ECON 255 — STATA Labs

Jihye Heo — Oberlin College

Contents

0 Introduction

0.1 Interface

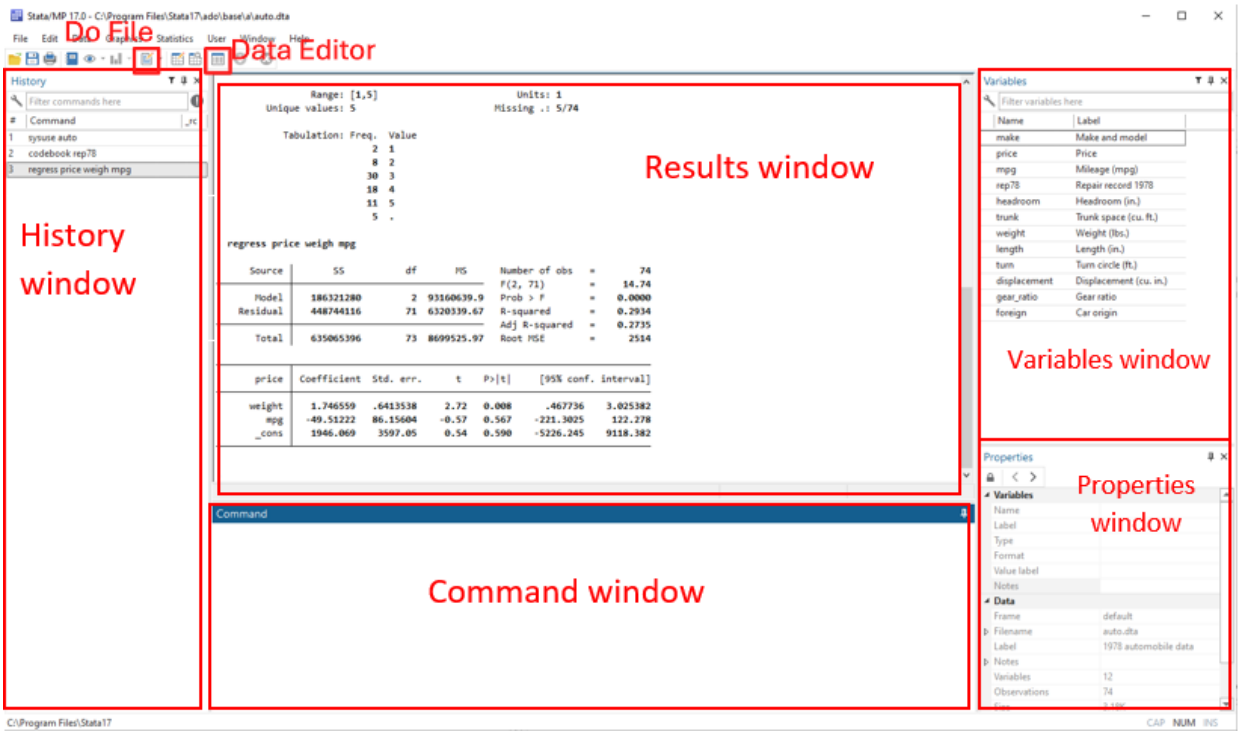

Figure 1: The STATA Interface

Key parts of the interface:

- **Command Window:** where you type Stata commands (not recommended for reproducibility).
- **Results Window:** displays command output.
- **Variables Window:** lists dataset variables currently in memory.
- **History Window:** shows your recent command history.

0.2 Working Directory

- **Default behavior:** Stata saves to and searches for files only in the current working directory, unless you specify a full file path.

> **Note**
>
> This means that if you do not set or change the working directory, Stata will not know where your dataset is located and will be unable to load it.

Tip: Working Directory for Our Lab Sessions

For lab sessions, set your working directory to the `Downloads` folder (where most datasets will be saved). To find the correct folder path:

1. Open your `Downloads` folder (or wherever you saved the dataset).
2. Click once on the dataset file to highlight it.
3. Right-click the file and select **Properties**.
4. In the Properties window, locate the field labeled **Location**.
5. Copy and paste this Location into Stata's `cd` command.

Set your working directory

```
* Display current directory
pwd

* Change working directory to Downloads (common in labs)
cd "C:\Users\YourUniquePath\Downloads"

* List files in the current directory
ls
```

0.3 File Types

- **Do-files (.do):** Contain the commands you write and execute. → We will always use Do-files instead of typing directly into the command window, because they let you:
  - Run multiple commands at once
  - Save your workflow for reproducibility
- **Data files (.dta):** Stata's native data format. Stata can also import `.csv`, `.xlsx`, and other formats.
- **Log files (.log):** Store both your commands and Stata's output. → Think of them as a record of what you did. Always start a log file at the beginning of each session.
  - You can add comments inside your do-file (which will also appear in the log) using an asterisk `*` at the start of a line.
    Example: `* This is a comment and will not be executed.`
  - Alternatively, use `//` for end-of-line comments.
    Example: `summarize mpg // quick summary of miles per gallon`

Note

When you have a question or encounter an error in Stata, always save both your **Do-file** and the corresponding **Log file**. Send them together to the teaching fellow or to me — this allows us to *reproduce exactly what you did* and quickly identify where the error occurred.

0.4 Default Log Template

Below is the default template you should use at the beginning of every lab session. It ensures your work is reproducible, well-documented, and easy to debug.

Default Do-file Template

```
*****************************************************
* [Your Name]
* Lab Class #1
*****************************************************

cls                  // clear command window
clear all            // remove any existing dataset
set more off         // prevent pauses in output

capture log close    // close any open log file
cd "C:\Users\YourName\Downloads"   // set working directory

log using "lab1_initials.log", replace // starting log file named "
    lab1_initials"

* --- your work goes here ---
* use comments (*) to explain steps

log close            // close the log at the end
*****************************************************
```

Below is the same template again for you to **copy and paste directly**:

```
*****************************************************
* [Your Name]
* Lab Class #1
*****************************************************

cls                  // clear command window
clear all            // remove any existing dataset
set more off         // prevent pauses in output

capture log close    // close any open log file
cd "C:\Users\YourPath\Downloads"   // set working directory

log using "lab1_initials.log", replace // starting log file named "lab1_initials"

* --- your work goes here ---
* use comments (*) to explain steps

log close            // close the log at the end
*****************************************************
```

Week 1 — Introduction to STATA (Teaching)

1.1 Goals for Week 1

1. Change the working directory (`cd`, `pwd`, `ls`).
2. Load datasets (.dta, .csv)
3. Construct new variables using `gen` and `replace`:
   - Apply conditions with `if`, logical operators (`&`, `|`)
   - Create dummy variables
   - Transform variables (squared, logarithm, interaction terms $A \times B$)
4. Summarize and describe data:
   - `describe`, `summarize`, `tabulate`
5. Keep or drop variables (`keep`, `drop`).
6. Create plots (histogram, scatter plots)
7. Save and export plots.
8. Save datasets for later use.

1.2 Setting for Week 1

In this week's lab, we will work with a *simulated* dataset that represents students who have taken **ECON255**. Each observation corresponds to one student in a given term, identified by a unique `StudentID`. The dataset includes demographic and academic information such as `Gender`, `Year` (Sophomore, Junior, Senior), and whether the student is an `EconMajor` or has a `DoubleMajor`. It also records measures of engagement, including the number of `Absences` and `OfficeHours` attended. Performance is captured by three midterm scores (`Mid1`, `Mid2`, `Mid3`) and a `FinalProject` score. This dataset is designed to illustrate how different academic behaviors (e.g., attending office hours, missing classes) and characteristics (e.g., major, year in school) are related to student outcomes.

1.3 Tasks for Week 1

0 Don't forget to start with your default do-file template!

1 Change the working directory to Downloads folder

```
pwd
cd "C:\...\Downloads"
```

2 Load the dataset (You can load either .dta, or .csv file)

```
* if using the DTA version:
use "simulated_students.dta", clear

* Or, if using the CSV version:
import delimited "simulated_students.csv", clear
```

3 Get descriptives of the data

- Use `describe`(**des**) to see an overview of variables.

```
des
des Mid1 Mid2 FinalProject
```

- Use `list` to inspect rows of data.

```
list StudentID Gender Year Mid1 Mid2 in 1/10
list StudentID Absences OfficeHours if Year=="Sophomore"
```

- Use `summarize`(**sum**) for summary statistics.

```
sum Mid1 Mid2 Mid3 FinalProject
sum Absences OfficeHours, detail
```

- Use `tabulate`(**tab**) for categorical variables.

```
tab Gender
tab Year, summarize(FinalProject)
```

- Use `count` to see how many observations meet conditions.

```
count if Absences > 5
count if OfficeHours > 10 & EconMajor==1
```

- Try logical conditions with `&`(and), `|`(or), `==`(equal to), `!=`(not equal to).

```
ls StudentID Mid1 Mid2 if Absences > 3 & EconMajor==1
```

4 Create new variables using `gen` and `egen`

- **Final score (0–100)**
  Midterms are 25% each; FinalProject is out of 25 points and counts fully.

```
gen FinalScore = 0.25*Mid1 + 0.25*Mid2 + 0.25*Mid3 +
    FinalProject
```

- **Letter grade from FinalScore**

```
gen LetterGrade = ""
replace LetterGrade = "A" if FinalScore >= 90
replace LetterGrade = "B" if FinalScore >= 80 & FinalScore <
    90
replace LetterGrade = "C" if FinalScore >= 70 & FinalScore <
    80
replace LetterGrade = "D" if FinalScore >= 60 & FinalScore <
    70
replace LetterGrade = "F" if FinalScore < 60
```

- **Log of office hours (add 1 to handle zeros)**

```
gen LogOH = log(OfficeHours + 1)
```

– **Squared absences**

```
gen AbsencesSq = Absences^2
```

– **Term average of FinalScore (using egen)**

```
egen TermAvgScore = mean(FinalScore), by(Term)
```

Tip: Difference between gen and egen

* `gen` (`generate`) is used to create a new variable based on a calculation for each observation.
  - Example: `gen LogOH = log(OfficeHours+1)` creates one value of `LogOH` per student.
* `egen` (`extensions to generate`) is used when you need to create variables that rely on **groups of observations** or more complex functions.
  - Example: `egen TermAvgScore = mean(FinalScore), by(Term)` gives every student in the same term the same average score.
* Rule of thumb: Use `gen` for simple, row-by-row calculations; use `egen` for grouped or advanced statistics.

5 Create and export Plots

– **Scatter plot: Office Hours vs. Final Score**

```
twoway scatter FinalScore OfficeHours, ///
    title("Office Hours vs. Final Score") ///
    xtitle("Office Hours Attended") ///
    ytitle("Final Score") ///
    name(sc_off, replace)
graph export "scatter_officehours_finalscore.png", replace
```

– **Histogram of Letter Grades by Term (overlayed)**

```
twoway ///
(histogram FinalScore if Term=="2024SP", width(5) start(50)
   fcolor(none) lcolor(red)) ///
(histogram FinalScore if Term=="2024FA", width(5) start(50)
   fcolor(blue%40) lcolor(blue)), ///
title("FinalScore by Term") ///
xtitle("FinalScore") ytitle("Frequency") ///
legend(order(1 "2024SP" 2 "2024FA") cols(2) position(11) ring
   (0) region(lcolor(none))) ///
name(h_overlay, replace)
graph export "histogram_overlay_by_term.png", replace
```

– **Joint Graph (1x2 layout)**

```
graph combine sc_off h_overlay, cols(2) name(gcombo, replace)
graph export "combined_plots.png", replace
```

Tip: Basic Structure of `twoway` Graphs in Stata

The general syntax is:

```
twoway (plottype yvar xvar [if] [in] [weight], options) ///
       (plottype yvar xvar, options) ... , ///
       graph_options
```

- `plottype`: the kind of plot (e.g., `scatter`, `line`, `histogram`).
- `yvar xvar`: the variables to plot (y-axis first, then x-axis).
- `options`: specific settings for that plot type (e.g., colors, titles).
- Multiple plot types can be combined by stacking them in parentheses, separated by spaces or line breaks with `///`.
- After the plot definitions, you can add `graph_options` to control the overall look (titles, axis labels, legends, etc.).

Example:

```
twoway (scatter y x, mcolor(blue)) ///
       (lfit y x, lcolor(red)), ///
       title("Y vs X") legend(order(1 "Data" 2 "Fit"))
```

This produces a scatter plot of `y` on `x` with a fitted regression line.

6 Save dataset

Save dataset in different formats

```
* Save as Stata .dta file
save "modified_students.dta", replace

* Save as CSV file
export delimited using "modified_students.csv", replace
```

Week 2 — Running OLS, Fitted Values, and Exporting Results

(Teaching)

2.1 New Goals for Week 2

1. Load and prepare datasets for regression analysis.
2. Run a basic Ordinary Least Squares (OLS) regression using `reg`.
3. Obtain and interpret key regression outputs:
   - Coefficient estimates($\hat{\beta}$) and standard errors($s.e(\hat{\beta})$)
   - R^2 and goodness-of-fit
4. Generate fitted values (`fitted_val` - $\hat{y}$) and residuals (`resid` - $\hat{u}$).
5. Summarize regression diagnostics with plots:
   - Residual plots
   - Fitted vs. actual scatter plots
6. Export regression results using `outreg2`.

2.2 Setting for Week 2

For this week we use the dataset `WAGE1.dta`, one of the standard examples from Wooldridge's textbook *Introductory Econometrics: A Modern Approach*. This dataset is a cross-sectional sample of working individuals in the United States. The main focus is on the determinants of hourly wages, with special attention to the role of education. The data contain information on wages, education, experience, demographic characteristics, and job-related variables.

Key variables we will work with include:

- `wage` : hourly wage in U.S. dollars.
- `educ` : years of education completed.
- `exper` : years of work experience.
- `tenure` : years with current employer.
- `female` : indicator variable for gender (=1 if female).
- `married` : indicator variable for marital status (=1 if married).

The dataset is widely used to illustrate the **Mincer wage equation**, which studies the returns to education. In this lab, however, we begin with a simple regression of `wage` on `educ` to practice running OLS, interpreting regression output, and generating fitted values and residuals in Stata.

2.3 Tasks for Week 2

0 Start with your default do-file template.

1 Load the dataset:

```
use "WAGE1.dta", clear
```

2 Explore the data (quick diagnostics before OLS)

- **Average years of education; min/max**

```
sum educ
di "Mean educ = " %5.2f r(mean) "   min = " r(min) "   max = "
    r(max)
```

Tip: Displaying Results from `summarize`

- `r(mean)`, `r(min)`, `r(max)` : stores results from the last `summarize` command.
- `%5.2f` : numeric display format — width 5, with 2 decimal places (e.g., `12.3456` → `12.35`).
- You can change the format (e.g., `%8.3f`) to adjust spacing and decimals.

- **Wage summary by gender (female vs. male)**

```
bysort female: sum wage
```

- **Distribution of education (histogram)**

```
hist educ, width(1) start(0) freq ///
title("Distribution of Education") xtitle("Years")
```

- **How many women vs. men?**

```
tab female
```

- **Education by gender (cross-tab)**

```
tab educ female
```

- **Overlay histograms of education by gender**

```
twoway ///
(histogram educ if female==1, width(1) start(0) fcolor(none)
   lcolor(red)) ///
(histogram educ if female==0, width(1) start(0) fcolor(none)
   lcolor(blue)), ///
title("Education by Gender") xtitle("Years of Education")
   ytitle("Frequency") ///
legend(order(1 "Female" 2 "Male") cols(2) position(11) ring(0)
    region(lcolor(none))) ///
name(educ_gender_overlay, replace)
```

3 Run a simple regression

- Estimate the regression of `wage` on `educ`.
- Retrieve and display the estimated intercept and slope coefficient.
- Compute fitted values and residuals from the regression.

Tip: `reg` Syntax and Saved Results

- **Syntax**

```
regress depvar indepvars [if] [in] [weight], [options]
```

- **Accessing estimates (after `reg`)**
  - Coefficient for `x1`: `_b[x1]`
  - Std. error for `x1`: `_se[x1]`
  - Intercept: `_b[_cons]`
  - R^2: `e(r2)` Adjusted R^2: `e(r2_a)`
  - Sample size: `e(N)` Residual df: `e(df_r)`
  - RMSE: `e(rmse)` Model/Residual SS: `e(mss)`, `e(rss)`
- **Predictions (after running `reg`)**
  - Fitted values $\hat{y}$: `predict yhat, xb`
  - Residuals u: `predict uhat, residuals`
  - SE of $\hat{y}$: `predict se_yhat, stdp`

```
. reg wage educ
```

Source	SS	df	MS
Model	1179.73204	1	1179.73204
Residual	5980.68225	524	11.4135158
Total	7160.41429	525	13.6388844

Number of obs	526
F(1, 524)	103.36
Prob > F	0.0000
R-squared	0.1648
Adj R-squared	0.1632
Root MSE	3.3784

wage	Coefficient	Std. err.	t	P>|t|	[95% conf.	interval]
educ	.5413593	.053248	10.17	0.000	.4367534	.6459651
_cons	-.9048516	.6849678	-1.32	0.187	-2.250472	.4407687

Figure 2: Regression Results

Note: The top-right panel reports the number of observations, R^2, and related fit statistics. The coefficient table displays the estimated coefficients, their standard errors, t-statistics (two-sided tests of $H_0 : \beta = 0$), corresponding two-sided p-values, and 95% confidence intervals.

[**Question**] What is the estimated increase in wage for a person with four more years of education?

4 Plot fitted vs actual, and residual plots:

```
    twoway (scatter wage educ) (lfit wage educ)

    twoway scatter resid educ, ///
    title("Residuals vs. Education") ///
    yline(0, lcolor(red))
```

5 Export regression results:
In this course, we will use the `outreg2` command to export regression tables to Word or other file formats. `outreg2` is not part of Stata's built-in commands, so you must first install the package (only once per computer).

```
    // Install outreg2 if not already available
    capture which outreg2   // checks if outreg2 is installed
    if _rc ssc install outreg2   // if not found (_rc != 0), install
       from SSC

    // Run regression and export results
    reg wage educ
    outreg2 using "Lab2_results.doc", ///
        replace ctitle("OLS: wage on educ") dec(3) bdec(3) tdec(3) ///
        addstat("R-squared", e(r2))
```

Week 3 — Assignment 1: Housing Prices (Homework)

3.1 Setting

Public real estate data, such as listing prices and housing characteristics, are widely accessible. Many providers even offer programmatic access (e.g., the Zillow API). In this assignment, you will use the dataset `HPRICE1.dta` from our textbook, which contains information on Boston-area home sales in 1990. Your tasks include: loading the data, generating descriptive statistics, estimating a simple OLS regression of housing price on key characteristics, and exporting both tables and figures.

3.2 What to Submit

Upload your **final and working** versions of the `.do` file, the corresponding `.log` file, and the exported `lab3_summary.doc` and `lab3_results.doc` to Blackboard. Grading will be based only on the files you submit, so please double–check that they are correct and complete before uploading.

3.3 Assignment Tasks

1. **Set up your default do-file template and change the working directory.**
2. **Load `HPRICE1.dta` dataset.**
3. **Summarize the main variables.** Generate descriptive statistics (mean, standard deviation, minimum, and maximum) for `price`, `sqrft`, and `bdrms`. Export these results into a Word document using the `outreg2` command. Save the file as `lab3_summary.doc`.

   Tip: `outreg2` to save Summary Statistics

   Type `help outreg2` in the Stata command window to view the official manual. In the section *Example 7. Summary tables*, the subsection *Regular summary table* shows how to save summary statistics (instead of regression results). Be sure to specify the file name with the `.doc` extension so that Stata saves it as a Word document rather than as a plain text file.

4. **Run an OLS regression.** Regress housing `price` on `sqrft` and `bdrms`.
5. **Save regression results.** Store the estimated coefficients as `beta0hat`, `beta1hat`, and `beta2hat`. Also generate fitted values (call them `pricehat`) and residuals (call them `resid`).
6. **Interpret the coefficient on bedrooms.** Use the `display` command and your saved `beta2hat` to calculate the estimated increase in house price from one additional bedroom, holding square footage constant. Report this in `lab3_summary.doc` using *dollar terms*, not thousands.
7. **Extend the calculation.** Again using `display`, compute the estimated increase in price for a house that has *one additional bedroom and 200 more square feet*. Report this in `lab3_summary.doc` and briefly explain in words why this increase is larger than in the previous step.
8. **Export regression results.** Export the OLS results as `lab3_results.doc` using `outreg2`. Make sure your table includes coefficients, standard errors, t-statistics, p-values, and R^2.

9. **Plot the fitted regression line.** Create a graph with housing `price` on the vertical axis and `sqrft` on the horizontal axis. Overlay the fitted regression line using the `pricehat` values you generated earlier, and export the figure to a file. Then include the figure into the `lab3_summary.doc`
   Note: Because the fitted values were estimated using both `sqrft` and `bdrms`, the line plotted against only `sqrft` will not be perfectly smooth—it will appear jagged.

Week 4 — Merging, Reshaping, and Inference (Teaching)

4.1 New Goals for Week 4

1. Acquire public country–year data from the World Bank API.
2. Tidy data by reshaping from *wide-by-year* to *long*, and pivoting to *wide-by-variable*.
3. Merge multiple country–year sources into one analysis dataset.
4. Create log and growth variables and run two macro-style regressions.
5. Export tables and (optionally) simple diagnostic plots.

4.2 Setting for Week 4

In practice, the data required for regression analysis are rarely clean or ready to use. They are often scattered across multiple datasets, recorded in different formats, or contain inconsistencies. To analyze them properly, researchers must first unify their structure and merge them using common identifiers.

In this week, we will practice these skills by using the World Bank API to download data in multiple formats and construct a **country–year panel dataset** with several key indicators.

With the completed dataset, our objective is to study two relationships at the country–year level:

- **Growth & Trade:** Does trade openness predict faster GDP per capita growth?

$$\Delta \log(\text{GDPpc}_{ct}) = \alpha + \beta \frac{Exports_{ct}}{GDP_{ct}} + \gamma \, \text{Education}_{ct} + u_{ct}. \tag{1}$$

- **Unemployment & Growth:** Are richer or high-inflation countries associated with different unemployment rates?

$$\text{Unemp}_{ct} = \alpha + \beta \, \log(\text{GDPpc}_{ct}) + \gamma \, \text{Inflation}_{ct} + u_{ct}. \tag{2}$$

We will use the `wbopendata` package in Stata to access World Bank indicators. For more information, type `help wbopendata`. The main variables of interest are:

- `NY.GDP.PCAP.KD` : real GDP per capita (constant 2015 US$)
- `NE.EXP.GNFS.ZS` : exports of goods & services (% of GDP)
- `NE.IMP.GNFS.ZS` : imports of goods & services (% of GDP)
- `SL.UEM.TOTL.ZS` : unemployment rate (% of labor force)
- `FP.CPI.TOTL.ZG` : inflation, CPI (annual %)
- `SE.SEC.ENRR` : gross secondary enrollment (%, proxy for education)

Note

Recall the syntax of the `reg` command: for example, `reg y x1 x2`. Keeping this in mind, and referring back to equations (1) and (2), what should the structure of our dataset look like?

4.3 Tasks for Week 4

0 Start with your default do-file template and set your working directory.

1 **Download country–year data directly in Stata (World Bank).**

- Download GDP, EXP, IMP data as *wide format* and save it as dta file
- Download CPI, UEM, SEC data as *long format* and save it as dta file

```
* Install package for World Bank API
ssc install wbopendata, replace

* Download GDP, EXP, IMP as wide format
wbopendata, indicator(NY.GDP.PCAP.KD; NE.EXP.GNFS.ZS; NE.IMP.GNFS.
    ZS) year(1990:2015) clear
save "week4_data1.dta", replace
* Check with Data Editor to see how this data looks like

* Download unemployment, inflation, education as long format
wbopendata, indicator(FP.CPI.TOTL.ZG; SL.UEM.TOTL.ZS; SE.SEC.ENRR)
     long year(1990:2015) clear
save "week4_data2.dta", replace
* Check with Data Editor to see how this data looks like
```

Tip: Long vs Wide

- **Wide format:** Each row is a country–indicator, and each column is a year. Wide format is convenient for humans to read, but hard to merge across different indicators. Example: one row for "GDP per capita in Brazil" with columns `yr1990`, `yr1991`, ...
- **Long format:** Each row is a country–year–indicator observation. Long format is the standard for regression analysis (country–year variables stacked in one column). Example: one row for "Brazil, 1990, GDP per capita" and another row for "Brazil, 1991, GDP per capita".

2 **Understanding the structure of first data**

For our regression equations (1) and (2), we need the dataset organized at the *country–year* level. That is, each row should represent a unique country–year observation. For each row, we want variables such as $\Delta \log(GDPpc_{ct})$, $\frac{Exports_{ct}}{GDP_{ct}}$, $Education_{ct}$, $Unemp_{ct}$, and $Inflation_{ct}$ as separate columns. The target regression dataset should look conceptually like this:

Country	Year	$\Delta \log(GDPpc)$	Exports/GDP	Education	Unemployment	Inflation
Brazil	1990	0.021	13.5	67.2	6.1	3.5
Brazil	1991	0.018	14.0	68.1	6.3	5.2
India	1990	0.025	9.2	52.7	4.8	8.4
India	1991	0.031	10.1	53.5	5.0	9.0
⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 1: Desired regression-ready structure: one row per country–year, variables as columns.

Now let us first investigate the first dataset. If you reload the file and open the Data Editor, you will see that the data are stored in a *wide format*, where each indicator has values spread across multiple year columns:

```
use "week4_data1.dta", clear
browse
```

An excerpt of the data looks like this:

countrycode	indicatorname	yr1990	yr1991	yr1992	yr1993	...
BRA	NY.GDP.PCAP.KD	5211.3	5297.4	5402.1	5534.6	...
BRA	NE.EXP.GNFS.ZS	11.8	12.1	12.5	13.2	...
BRA	NE.IMP.GNFS.ZS	10.6	11.3	12.0	12.8	...
IND	NY.GDP.PCAP.KD	1347.2	1382.5	1425.6	1487.3	...
IND	NE.EXP.GNFS.ZS	7.1	7.5	8.0	8.6	...
⋮	⋮	⋮	⋮	⋮	⋮	...

Table 2: Example structure of `week4_data1.dta` (wide format).

To prepare the dataset for regression, we need to *reshape* the data. Currently, each year appears as a separate column, and indicators are stacked under the `indicatorname` column. Our goal is to convert the dataset so that:

- Each row corresponds to a unique `countrycode - year` pair.
- Each indicator (GDP per capita, exports, imports) becomes its own column.

This requires two steps in Stata: (i) reshape from wide to long for years, and (ii) reshape back from long to wide for indicators to become variables.

```
* Step 1. Reshape wide -> long (stack years into a single column)
* Wide format: separate columns for yr1990, yr1991, yr1992, ...
* After this step:
*   - Those year columns are combined into one variable "year"
*   - Their values go into a single variable "yr"
*   - (countrycode, indicatorname) identifies each panel (i)
reshape long yr, i(countrycode indicatorname) j(year)
```

After first step, we would have data structure that looks like below.

countrycode	**indicatorname**	**year**	**yr**
BRA	NY.GDP.PCAP.KD	1990	5211.3
BRA	NY.GDP.PCAP.KD	1991	5297.4
BRA	NE.EXP.GNFS.ZS	1990	11.8
BRA	NE.EXP.GNFS.ZS	1991	12.1
⋮	⋮	⋮	⋮

Table 3: Example structure *after* Step 1 (`reshape long`): each row is a country–indicator–year observation, with values stored in `yr`.

Now in the second step, we need to transform the dataset so that each value of `indicatorcode` becomes its own separate variable. In other words, instead of stacking all indicators under one column, we spread them out so that for each country–year we directly observe GDP per capita, exports, imports, and so forth as separate columns.

However, Stata does not allow dots (`.`) or spaces in variable names. Therefore, before reshaping, we first create short, valid names (e.g. `NY.GDP.PCAP.KD` → `gdp_pc`, `NE.EXP.GNFS.ZS` → `exp`, `NE.IMP.GNFS.ZS` → `imp`). This step ensures that the reshape command works properly.

```
* Map raw codes to short, valid names (no spaces/dots)
gen str20 ind_short = "" // First create an empty string variable
replace ind_short = "gdp_pc" if indicatorcode == "NY.GDP.PCAP.KD"
replace ind_short = "exp"    if indicatorcode == "NE.EXP.GNFS.ZS"
replace ind_short = "imp"    if indicatorcode == "NE.IMP.GNFS.ZS"

* Step 2. Reshape long -> wide (make indicators into separate
    columns)
reshape wide yr, i(countrycode year) j(ind_short) string

* Tidy variable names
rename yrgdp_pc gdp_pc
rename yrexp    exp
rename yrimp    imp
```

Our final dataset now has one observation per country–year, with clean variables for GDP per capita, exports, and imports.

countrycode	year	gdp_pc	exp	imp
BRA	1990	5211.3	11.8	10.6
BRA	1991	5297.4	12.1	11.3
IND	1990	1347.2	7.1	8.5
IND	1991	1382.5	7.5	9.0
⋮	⋮	⋮	⋮	⋮

Table 4: Example structure *after* Step 2 (`reshape wide`): each row is now a country–year observation with indicators in separate columns.

Finally, we save the final version of this data:

```
* Saving final version
save "week4_data1_final.dta"
```

3 Merging and Renaming

Now, we load the second dataset, which was downloaded in long format (containing unemployment, inflation, and education variables):

```
use "week4_data2.dta", clear
browse
```

We now merge this dataset with the reshaped file `week4_data1_final.dta`, using the common identifiers `countrycode` and `year`. In Stata, merging requires us to specify the relationship between the datasets:

Tip: Merging Datasets

- `1:1` merge: one observation in master matches exactly one in using dataset.
- `m:1` merge: many observations in master match one in using dataset.
- `1:m` merge: one observation in master matches many in using dataset.
- `m:m` merge: many-to-many merge (generally discouraged).

In this exercise, both datasets are organized at the `country-year` level, so we use a `1:1` merge.

```
* Merge the two datasets by countrycode and year
merge 1:1 countrycode year using "week4_data1_final.dta"

* Inspect merge result
tab _merge
```

Tip: Interpreting `_merge`

`_merge==3`: matched in both files.
`_merge==1`: master-only (house has no income match).
`_merge==2`: using-only (income row with no houses).

Finally, we rename the variables for readability and clarity:

```
* Clean up variable names for clarity
rename sl_uem_totl_zs unemp         // Unemployment rate
rename fp_cpi_totl_zg infl          // Inflation (CPI, %)
rename se_sec_enrr sec_enroll       // Secondary school enrollment
```

Our final dataset should look like the following:

countrycode	year	gdp_pc	exp	imp	unemp	infl	sec_enroll
BRA	1990	5211.3	11.8	10.6	6.1	3.5	67.2
BRA	1991	5297.4	12.1	11.3	6.3	5.2	68.1
IND	1990	1347.2	7.1	8.5	4.8	8.4	52.7
IND	1991	1382.5	7.5	9.0	5.0	9.0	53.5
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

Table 5: Final merged dataset at the country–year level: each row is a country–year observation with indicators as separate variables.

4 **Create log and growth variables.**
We need the country–year growth rate of real GDP per capita:

$$\Delta \log(\text{GDPpc}_{ct}) \equiv \log(\text{GDPpc}_{ct}) - \log(\text{GDPpc}_{c,t-1}). \tag{3}$$

To compute this correctly, we must take differences *within each country* across adjacent years. In Stata, we declare the panel structure (country as the panel identifier, year as the time index) and then use the time-series difference operator `D.`, which applies differences by panel.

```
* 1) Log level
gen lgdppc = log(gdp_pc)
label var lgdppc "log(GDP per capita)"

* 2) Declare panel structure so time operators work within country
*xtset needs a numeric id, so we will encode countrycode to an id
egen cid = group(countrycode), label
xtset cid year

* 3) First difference of log GDP per capita, within each country
gen dlgdppc = D.lgdppc
label var dlgdppc "Delta log GDP per capita (within-country)"
```

Tip: Why `xtset` + `D.`?

- `xtset cid year` tells Stata that observations are ordered by **country** (panel) and **year** (time).
- After `xtset`, the operator `D.var` computes $var_t - var_{t-1}$ *for each country separately.* It never "bleeds" across countries.
- If you skip `xtset`, `D.` won't know where one country ends and the next begins, and differences may be wrong.

5 **Regression 1: Growth & Trade (export table).**

$$\Delta \log(\text{GDPpc}_{ct}) = \alpha + \beta \cdot \frac{Exports_{ct}}{GDP_{ct}} + \gamma \cdot \text{Educ}_{ct} + u_{ct}.$$

```
* Baseline growth regression
reg dlgdppc exports_gdp educ

* Export table
capture which outreg2
if _rc ssc install outreg2

outreg2 using "lab4_results.doc", replace ///
    ctitle("Growth on Trade & Education") dec(3) bdec(3) tdec(3)
       ///
    addstat("R-squared", e(r2), "N", e(N))
```

Note

How should we interpret $\hat{\beta}$ when the explanatory variable `exports_gdp` is measured in percentage points, while the dependent variable is the change in log GDP per capita?

6 **Regression 2: Unemployment & Growth (append table).**

$$\text{Unemp}_{ct} = \alpha + \beta \cdot \log(\text{GDPpc}_{ct}) + \gamma \cdot \text{Inflation}_{ct} + u_{ct}.$$

```
* Levels regression: unemployment on income and inflation
reg unemp lgdppc infl

* Append to the same document
outreg2 using "lab4_results.doc", append ///
    ctitle("Unemployment on log(GDPpc) & Inflation") dec(3) bdec
       (3) tdec(3) ///
    addstat("R-squared", e(r2), "N", e(N))
```

Note

How should we interpret $\hat{\beta}$ here?

7 Optional quick plots.

```
* Scatter + fitted line: growth vs openness
twoway (scatter dlgdppc exp, msize(vsmall)) ///
       (lfit dlgdppc exp, lcolor(red)), ///
       title("Growth vs Exports/GDP") ///
       ytitle("Change in log(GDPpc)") xtitle("Exports/GDP (pct of
          GDP)")
graph export "figures\growth_vs_exports.png", replace

twoway (scatter unemp lgdppc, msize(vsmall)) ///
       (lfit unemp lgdppc, lcolor(red)), ///
       title("Unemployment vs log(GDPpc)")
graph export "figures\unemp_vs_lgdppc.png", replace
```

Week 5 — Assignment 2 : Merging Movie Data (Homework)

5.1 Setting

You will work with two simulated datasets about movies. Each file contains different information, and your task is to merge them together for analysis:

- `movies_char.csv`: movie characteristics, including ID, title, genre, and budget.
- `movies_outcomes.csv`: movie outcomes, including ID, release year, box office revenue, and rating.

Your goals are to (i) import and clean each dataset, (ii) merge them by the movie identifier `movie_id`, and (iii) investigate how characteristics such as budget and genre are related to outcomes such as box office performance and ratings. Core variables will include: budget (`budget`), genre (`genre`), box office revenue (`box_office`), and rating (`rating`).

5.2 What to Submit

Upload your **final and working** versions of the `.do` file, the corresponding `.log` file, and the exported `lab5_results.doc` to Blackboard. Grading will be based only on the files you submit, so please double–check that they are correct and complete before uploading.

5.3 Assignment Tasks

1. **Set up** your default do-file template and change the working directory to your project folder.
2. **Load the movie characteristics data (`movies_char.csv`)**
   This dataset contains one observation per movie and includes the following variables:
   - `movie_id`: the unique identifier for each movie.
   - `title`: the movie title.
   - `genre`: the movie genre (e.g., Action, Comedy, Drama).
   - `budget`: the production budget (in millions of dollars).

   After importing, save the dataset as `char.dta`.
3. **Load the movie outcomes data (`movies_outcomes.csv`)**
   This dataset contains one observation per movie and includes the following variables:
   - `movie_id`: the unique identifier for each movie.
   - `year`: the release year of the movie.
   - `box_office`: worldwide box office revenue (in millions of dollars).
   - `rating`: the average IMDb-style rating (0–10 scale).

   After importing, save the dataset as `outcomes.dta`.
4. **Merge the movie characteristics and movie outcomes data.** Be sure to include the answers to below question in your `lab5_results.doc` once you generate the document file after completing the regression analysis in question 7.
   - What was the common identifier for matching?

- Is this 1–1, m–1, or 1–m matching? Why?
- Report the match distribution using `_merge` and the match rate (share of `_merge==3`).

Tip: Hints for merging

- Load the movie characteristics dataset:

```
use "char.dta", clear
```

- Merge with the movie outcomes file. Fill in the merge type and common identifier (replace merge_type and common_identifier with the correct inputs):

```
merge merge_type common_identifier using "outcomes.dta"
```

- Check how many matches you got:

```
tab _merge
```

- Compute the match rate:

```
count if _merge == 3
count
display "Match rate = " r(N1)/r(N2)
```

- For later analysis, you may want to keep only matched observations:

```
keep if _merge == 3
drop _merge
```

5. **Explore the merged dataset (Part 1).** After merging, each row now represents an *individual movie*, including its characteristics (budget, genre) and outcomes (box office, rating). Next, classify movies into "blockbusters" or "flops" based on their box office relative to their budget:
   - Define a **blockbuster** as a movie with box office revenue at least twice its budget.
   - Define a **flop** as a movie with box office revenue less than its budget.
   - Generate indicator variables for each case (`blockbuster`, `flop`) and tabulate their frequency across genres.
   - **[Write in `lab5_results.doc` after your regression]:** Which genre has the highest share of blockbusters? Which genre has the highest share of flops?

Tip: Tabulating shares in Stata

To see the share of blockbusters within each genre, use:

```
tab genre blockbuster, row
```

This reports row percentages: for each genre, the proportion of movies that are blockbusters vs. non-blockbusters.

6. **Explore the merged dataset (Part 2).** Now that you have created indicators for blockbusters and flops, let's focus on how the relationship between budget and box office differs across genres.
   - Create a **scatter plot** of `box_office` (y-axis) against `budget` (x-axis) for **Fantasy** movies.
   - Overlay a second scatter plot for **Romance** movies on the same graph, using a different color.
   - Add a legend and clear axis titles so the two genres can be compared directly.
   - Export this graph and include it in `lab5_results.doc` later.
   - **[Write in `lab5_results.doc` after your regression]:** How do Fantasy and Romance movies differ in terms of budget size and box office performance?
7. **Estimate the regression model and export the table as `lab5_results.doc`.**

$$\log(\text{box\_office}) = \beta_0 + \beta_1 \log(\text{budget}) + \beta_2 \,\text{rating} + u. \tag{4}$$

   (Generate new variables: `lbox_office = log(box_office)`, `lbudget = log(budget)`.)
   - **[Write in `lab5_results.doc`]:** Interpret $\hat{\beta}_1$: approximately how much higher is box office revenue if a movie's budget doubles?
   - **[Write in `lab5_results.doc`]:** Interpret $\hat{\beta}_2$: how is box office revenue related to a one-point increase in rating (on the 0–10 scale)?
8. **Hypothesis testing** Use your regression model to formally test whether budget and rating matter for explaining box office revenues.
   - Conduct a **t-test** of the null hypothesis that the coefficient on log(budget) is equal to zero.
     (*Just reference t-stat column from regression results!*)
   - Conduct an **F-test** of the null hypothesis that the coefficients on both log(budget) and `rating` are equal to zero jointly.
     (*Use the tipbox below for F-test*)
   - **[Write in `lab5_results.doc`]:** Report the test statistics, p-values, and whether you reject the null hypotheses at the 5% significance level. Briefly interpret what these results imply in the context of movies.

Tip: Hints for hypothesis testing in Stata

- When you run a regression, Stata already reports the **t-statistic** for each coefficient. For example:

```
reg lbox_office lbudget rating
```

  The t-test for $H_0 : \beta_1 = 0$ (coefficient on `lbudget`) is shown in the regression output.

- To conduct a joint **F-test**, use the `test` command after regression:

```
test x1 x2
```

  This tests the null hypothesis $H_0 : \beta_{x1} = \beta_{x2} = 0$ against the alternative that at least one of the coefficients is nonzero.

9. **Wrap-up.** Your `lab5_results.doc` must include:
   - Answers to #4, #5, #6, #7, #8
   - Figures from #6
   - Regression table from #7
   - One factor that might pose endogeneity problem in your regression and short explanation as to why.

Week 6 — Nonlinearities in Regression (Teaching)

6.1 New Goals for Week 6

1. Understand how to incorporate nonlinearities in regression:
   - Quadratic terms
   - Interaction terms
2. Learn how to calculate and interpret marginal effects.
3. Apply results to economic questions such as diminishing returns or heterogeneous effects.

6.2 Setting for Week 6

For this week we use the dataset `twoyear.dta`, which contains individual-level data on wages, education, experience, and demographics.

Key variables:

- `lwage` : log hourly wage (dependent variable).
- `educ` : years of education.
- `exper` : years of work experience.
- `female` : gender indicator (=1 if female).

We focus on the relationship between wages, experience, and gender, illustrating how quadratic and interaction terms are estimated and interpreted.

6.3 Tasks for Week 6

0 Start with your default do-file template.

1 Load the dataset and explore the main variables

```
use "twoyear.dta", clear

sum lwage exper
tab female
```

2 Run the following baseline regression and export it using `outreg2` as Lab3_results.doc

$$\log(wage_i) = \beta_0 + \beta_1 \text{exper}_i + u_i \tag{5}$$

```
* Running simple regression
reg lwage exper
```

3 **Quadratics: does the return to experience differ by experience level?**
The effect of experience on log wages may be nonlinear, with returns diminishing as experience increases. To capture this, we now run a quadratic regression and append the results to `Lab3_results.doc` using `outreg2`:

$$\log(wage_i) = \beta_0 + \beta_1 \text{exper}_i + \beta_2 \text{exper}_i^2 + u_i \tag{6}$$

```
* Allowing decreasing/increasing returns
reg lwage c.exper##c.exper
```

Tip: Running Regression with Nonlinearities

There are two common ways to estimate quadratic regressions:

1. **Manual approach:**

```
gen expersq = exper^2
reg lwage exper expersq
```

This works for estimation, but when you later use `margins`, Stata does not automatically recognize that `expersq` is a function of `exper`. As a result, marginal effects of experience will be incorrect unless you compute them manually:

$$\frac{\partial \log(wage)}{\partial\, exper} = \hat{\beta}_1 + 2\hat{\beta}_2 \cdot exper$$

2. **## operator (recommended):**

```
reg lwage c.exper##c.exper
```

The `##` notation tells Stata to include both the main effect (`c.exper`) and the squared term (`c.exper##c.exper`). Importantly, Stata then knows the functional form, so `margins` correctly computes partial derivatives.

Key syntax:

* `c.var` : treat `var` as continuous.
* `i.var` : treat `var` as categorical (indicator/dummy).
* `c.var1##c.var2` : include both main effects and their interaction.
* `i.var##c.var` : include main effect of the indicator, main effect of the continuous variable, and the interaction term.

4 Now, let's get marginal effect of an additional year of experience when experience is minimum, median, mean, and maximum values.

$$\frac{\partial\, \log(wage_i)}{\partial\, \text{exper}} = \hat{\beta}_1 + 2\,\hat{\beta}_2 \cdot \text{exper} \tag{7}$$

```
* Get key reference points for exper
quietly summarize exper, detail
local exp_min  = r(min)
local exp_med  = r(p50)
quietly summarize exper
local exp_mean = r(mean)
local exp_max  = r(max)

```

```
    * Marginal effects at min / median / mean / max
    margins, dydx(exper) at( ///
        exper = (‘exp_min’ ‘exp_med’ ‘exp_mean’ ‘exp_max’) )


    * Plot (x-axis shows the exper values we chose)
    marginsplot, xdimension(exper) ///
        title("Marginal Effect of Experience at Key Points") ///
        ytitle("d ln(wage) / d exper") xtitle("Experience (years)"
            )
    * Is the return to experience increasing or decreasing an
        experience level increases?
```

5 **Interactions: does the return to experience differ by gender?**

If the payoff to an extra year of experience differs systematically for women and men (different slopes), a single coefficient on `exper` is too restrictive. An interaction allows the slope to vary by group:

$$\log(wage_i) = \beta_0 + \beta_1 \,\text{exper}_i + \beta_2 \,\text{female}_i + \beta_3(\text{female}_i \times \text{exper}_i) + u_i. \tag{8}$$

Here, β_1 is the experience effect for men (baseline), and $\beta_1 + \beta_3$ is the experience effect for women. Specifically,

- β_1: marginal effect of experience on log wage for men.
- β_2: female–male gap at zero experience.
- β_3: difference in the return to experience (women vs. men).

```
    * Gender-specific returns to experience
    reg lwage i.female##c.exper
```

Marginal effects with `margins`:

- *Effect of one more year of experience*, by gender, at chosen experience levels:

```
    * Average marginal effect of experience, by gender
    margins female, dydx(exper)

    * Marginal effect of experience at min/median/mean/max for
        each gender (we already saved these earlier)
    margins female, dydx(exper) at(exper = (‘exp_min’ ‘exp_med’ ‘
        exp_mean’ ‘exp_max’))
    marginsplot, title("ME of experience on log(wage) by gender")

    * Or plot marginal effect at various experience level
    margins female, dydx(exper) at(exper=(0(30) 150))
    marginsplot
    * Now at which dimension does the return to experience differ?
        Is it across experience levels or across genders?
```

6 **[Optional] Interactions & Quadratics**

Experience often shows diminishing returns, meaning the marginal effect of an extra year is smaller at higher levels of experience. Moreover, this curvature may differ by gender. To allow both the intercept, slope, and curvature to vary by gender, we estimate:

$$\log(wage_i) = \beta_0 + \beta_1 \,\text{exper}_i + \beta_2 \,\text{exper}_i^2 + \beta_3 \,\text{female}_i + \beta_4(\text{female}_i \times \text{exper}_i) + \beta_5(\text{female}_i \times \text{exper}_i^2) + u_i.$$

```
* Gender-specific nonlinear returns to experience
reg lwage i.female##c.exper##c.exper
```

Tip: Triple ##

Using `i.female##c.exper##c.exper` tells Stata to include:

- The main effects: `female`, `exper`, and `exper`^{2}.
- All interactions: `female` × `exper`, `female` × `exper`^{2}.
- By construction, this setup lets the quadratic wage–experience profile differ by gender.

This is much safer than generating your own squared or interaction terms by hand, because `margins` automatically recognizes the full functional form.

Week 7 — Assignment 3: Nonlinearities in Regression (Homework)

7.1 Setting

You will use the dataset `LOANAPP.dta`, which contains information on mortgage loan applications. Your goal is to examine whether loan approval is influenced by race, gender, and other applicant characteristics, while allowing for nonlinearities such as quadratic terms and interaction effects.

7.2 What to Submit

Upload your **final and working** versions of the `.do` file, the corresponding `.log` file, and the exported `lab7_results.doc` (along with any figures) to Blackboard. Grading will be based only on the files you submit, so please double–check that they are correct and complete before uploading.

7.3 Assignment Tasks

1. **Set up your default do-file template and change the working directory.**
2. **Load the `LOANAPP.dta` dataset.** Explore the following variables using `des` and `sum`.
   - *white* (race indicator)
   - *male* (gender)
   - Housing expenditure/income ratio (*hrat*)
   - Other obligations/income ratio (*obrat*)
   - Industry unemployment rate (*unem*)
   - Marital status (*married*)
   - Number of dependents (*dep*)
   - Schooling > 12 years (*sch*)
   - Applicant income (*appinc*)
   - Late payment dummies (one, two, more than two) (*mortlat1*, *mortlat2*)
3. **Descriptive analysis.**
   Be sure to include these results and figures in `lab7_results.doc` once you generate the document file after completing the regression analysis in question 4.
   - Compute the proportion of applicants who are white vs. non-white.
   - Calculate the approval rate for each group (four groups: white men, white women, non-white men, non-white women) and the difference in means.

     Tip: Getting Group-Specific Averages and Figures

     To compute averages by groups, use:

     ```
     bysort group1 group2: summarize var
     ```

   - *Figure exercise:* Above exercise is bit hard to see the full picture quickly. Let's create a bar chart showing approval rates by race and gender (four groups: white

men, white women, non-white men, non-white women). Save it so you can put it inside `lab7_results.doc` later.

Tip: Creating Visually Appealing Figures

Add readable category labels for 0/1 indicators:

```
label define race 0 "Non-white" 1 "White", replace
label values white race
label define sex 0 "Women" 1 "Men", replace
label values male sex
```

Keep the y-axis on 0–1 and label it clearly. Show bar labels with 2 decimals:

```
graph bar (mean) approve, over(male) over(white) ///
yscale(range(0 1)) ylabel(0(.1)1, format(%3.1f) angle
    (0)) ///
blabel(bar, format(%4.2f) position(outside)) ///
bargap(20) bar(1, lcolor(black)) ///
title("Approval Rate by Race and Gender") ///
ytitle("Mean approval (share)") ///
note("Mean of 0/1 outcome")
```

4. **OLS regression without nonlinearities.** Estimate the probability of loan approval as a function of the applicant characteristics listed in question 2 and export the regression results to `lab7_results.doc`.

Tip: Wildcard Character '*'

In Stata, the asterisk `*` can be used as a *wildcard* in variable lists.

- `prefix*` will include all variables whose names begin with `prefix`.
- `*suffix` will include all variables whose names end with `suffix`.
- `*text*` will include all variables that contain `text` anywhere in their names.

This feature is especially useful when variables follow a consistent naming pattern (e.g., multiple dummies, ratios, or survey items). It helps avoid typing each variable name manually and reduces the chance of typos.
For example, if your dataset has columns named `yr_2001`, `yr_2002`, ..., `yr_2024`, instead of listing them one by one, you can simply type `yr_*`.

5. **Partial effects in linear models.**
   - Compute the marginal effect of `appinc` at its minimum, average, and maximum values. (*Hint: This should be the same across different value because we didn't incorporate any non-linearities yet!*)
   - Calculate the partial effect of *white* for both male and female applicants.
   - Include interpretation of these marginal effects in your `lab7_results.doc`. (*You are writing interpretation of above questions manually in your document before you submit!!!*)
6. **OLS regression with nonlinearities.** Re-estimate the probability of loan approval,

this time including the interaction between *white* and *male*, as well as a quadratic term for the applicant's income. Append these regression results to `lab7_results.doc`. (Use the `##` operator so that we can use `margins` command.)

7. **Nonlinearities and partial effects.**
   - Compute the marginal effect of `appinc` at its minimum, average, and maximum values. (*Hint: Now that we have non-linearities, these should be different!!*)
   - Calculate the partial effect of *white* for both male and female applicants.
   - Include interpretation of these marginal effects in your `lab7_results.doc`. (*You are writing interpretation of above questions manually in your document before you submit!!!*)
8. **Inference with linear combinations.** Using your regression with nonlinearities, test whether the effect of being white differs significantly between men and women.
   - Recall: in the model with *white*, *male*, and *white* × *male*, the effect of being white is:
     - For women: coefficient on *white*.
     - For men: coefficient on *white* + coefficient on (*white* × *male*).
   - Use the `lincom` command to test whether the white–nonwhite gap is the same for men and women.
   - Interpret the result in `lab7_results.doc` : is there evidence that the impact of race on loan approval differs by gender?
9. **Wrap-up.** Your `lab7_results.doc` must include:
   - Answers to #3, #5, #7, #8
   - Figure from #3
   - Regression result from #4, #6
   - **Potential Endogeneity.** Where might endogeneity arise in our loan approval regression? Discuss possible sources and why they could bias the OLS estimates.

Week 8 — Introduction to Panel Data and Lagged Variables

(Teaching)

8.1 New Goals for Week 8

1. Learn practical tools for **model robustness**:
   - Compare alternative functional forms.
   - Use **heteroskedasticity-robust standard errors** and understand why they matter.
2. Begin working with data that have both a **cross-sectional** and a **time** dimension:
   - Recognize what panel data are and how they differ from a single cross-section.
   - Understand why many economic questions require tracking the same units over time.
   - Create and interpret a one-period lag of the dependent variable as a first step toward panel analysis.

8.2 Setting for Week 8

We will use the simulated student–year panel `lab8_students_panel.dta` (balanced panel with 600 students observed for 4 years). This is the first lab in which we explicitly use the **time dimension** of the data. While previous weeks analyzed only cross-sectional variation, many empirical studies in economics require examining how the same individuals or firms change over time. Such datasets—called *panel data*—allow researchers to investigate both *cross-sectional* and *time-series* variation simultaneously. Starting this week, we will begin to utilize this structure by creating and interpreting lagged variables, setting the stage for full panel techniques in later weeks.

Key dataset features

- Balanced panel: `student` (id), `year` (1–4).
- Major (categorical, 7 values): Economics, ComputerScience, Business, Biology, Education, Sociology, Psychology.
- Employment indicators:
  - `employed` : 1 if employed in that year, 0 otherwise.
  - `underemp` : 1 if employed but underemployed (major-specific probability).
- Wages:
  - `logw` : log(wage) for employed observations (missing if unemployed).
  - `wage` : level wage (0 if unemployed).
- Human-capital and demographics:
  - `exper` : years of experience.
  - `tenure` : an approximate tenure measure.
  - `age` : age in each year.
  - `female` : gender indicator.
  - `race` : categorical race indicator.

8.3 Tasks for Week 8

0 Start with your default do-file template.

1 **Load and explore the dataset.** The data are stored as a balanced panel, meaning each student (`id`) is observed for four years. Begin by loading the dataset and examining its main variables.

```
use "lab8_students_panel.dta", clear

describe
summarize logw wage exper tenure age
tab major
tab year
```

Discuss: What variables vary over time? Which are time-invariant?

2 **Check the panel structure.** In Stata, panel data must be *declared* using `xtset`, which tells Stata the unique identifier for each entity and the time variable.

```
* Declare panel structure: (id = student identifier, year =
    time)
xtset id year

* Describe the panel characteristics
xtdescribe
```

Stata now knows that observations are ordered by student and year. The output of `xtdescribe` confirms that this dataset is **strongly balanced**—each student has 4 yearly observations.

Tip: Panel terminology

- **Cross section:** a single snapshot across many individuals (e.g., one year).
- **Time series:** one individual observed over many periods.
- **Panel:** combines both—many individuals observed over time.

3 **Visualize average trajectories by group (race).** To illustrate how outcomes move over time at the group level, compute and plot the mean log wages among employed by race across years.

```
* Save working data, compute group-year means, and restore
    original data
preserve

* 1) Mean log wage by race (condition on employed so logw exists)
collapse (mean) mean_logw = logw if !missing(logw), by(race year)

* Small-multiples: one panel per race
graph twoway line mean_logw year, ///
    by(race, cols(3) compact title("Mean log(wage) by race (
        employed only)")) ///
    ytitle("Mean log(wage)") xtitle("Year") note("Conditioned on
        employed (logw non-missing)")
```

```
11
12 graph export "mean_logw_by_race_year.png", replace
13
14 restore
```

Tip: Why `preserve/restore` when using `collapse`

Using `collapse` replaces the current dataset with a new aggregated dataset (one row per group). That *permanently* removes the original observation-level data in memory unless you save it first. In class we want to compute group-year summaries for plotting without destroying the full panel dataset we will continue to use.

* **Quick and safe (recommended):**

```
preserve
collapse (function) newvar = newvar_func, by(groupvar
    timevar)
restore
```

Here, `function` can be any summary operator such as `mean`, `sum`, `sd`, or `count`, and `by()` lists the grouping variables that define the aggregation. This temporarily swaps in the aggregated data and then restores the original dataset exactly as it was.

Short reminders:

* Don't call `xtset` on the collapsed (group-level) data — it no longer has the original id/year structure.
* `preserve/restore` is easy and safe for short in-class analyses; use `collapse + save` if you want to keep the aggregated table for reuse.

4 **Wages among employed vs. overall average wage by major.**
To see how employment status affects average earnings, create a variable that equals wage only for employed students. Then, compare the mean wage among employed students ($E[\text{wage} \mid \text{employed} = 1]$) with the overall mean wage (which includes zeros for unemployed students).

```
* Create a variable for wage among employed students
gen wage_emp = wage if employed==1

* Compare mean wage among employed vs. overall mean wage
tabstat wage_emp wage, by(major) statistics(mean) ///
    columns(variables) format(%9.2f)
```

This table reports two columns for each major:

* `wage_emp`: mean wage among employed students only (conditional mean)
* `wage`: mean wage across all students, including zeros for unemployed (unconditional mean)

Discuss: Which majors have higher average wages among employed students? Are those rankings the same when you include non-employed students in the average?

5 **Baseline regression and heteroskedasticity.**
Now estimate a pooled wage regression using employed observations only. Because variance of residuals may increase with experience or differ by major, we use `, robust` to obtain heteroskedasticity-consistent standard errors. Then, plot residuals versus `exper` (experience) to visually check for heteroskedasticity by examining the distribution of residuals conditional on an independent variable.

```
* Baseline regression using employed observations
reg logw i.major exper tenure female i.race if employed==1, robust
    baselevels
* The option "baselevels" forces Stata to display the omitted (
   base) categories in the regression table.
* For example, the output should show "(base)" next to Economics (
   major==1) and White (race==1).
* All other category coefficients are interpreted relative to
   these base groups.

* Generate residuals
predict resid, resid

* Plot residuals versus experience (more direct for our DGP where
   sd grows with exper)
twoway (scatter resid exper, msymbol(o) msize(small) ///
       (lfit resid exper), ///
       ytitle("Residuals") xtitle("Experience (years)") ///
       title("Residuals vs Experience") ///
       note("Increasing spread in residuals across exper indicates
           heteroskedasticity; robust SEs needed")

graph export "resid_vs_experience.png", replace
```

Tip: Interpreting residuals conditional on an independent variable

To detect heteroskedasticity visually, examine how the distribution (vertical spread) of residuals changes *conditional* on an independent variable such as experience or education. If the spread grows (a "fan" shape) as the conditioning variable increases, this indicates heteroskedasticity: $Var(u \mid X)$ depends on X. Using `, robust` produces consistent standard errors in this case.

6 **Lagged variable and clustering intuition.**
Next, we examine persistence by adding a lagged dependent variable. This helps us understand how outcomes today are related to past outcomes, and why repeated observations of the same student may have correlated errors.

```
* Create one-year lag of log wage within each student
bysort id (year): gen L1_logw = logw[_n-1]

* Check the correlation between current and lagged log wages
corr logw L1_logw if year>1 & employed==1

* Regression with lagged dependent variable
reg logw L1_logw i.major exper tenure female i.race ///
```

```
    if employed==1 & year>1, robust

* Visualize persistence (strong correlation between L1_logw and
    logw)
twoway (scatter logw L1_logw if year>1 & employed==1, msymbol(o)
    msize(small)) ///
       (lfit logw L1_logw if year>1 & employed==1), ///
       ytitle("Current log(wage)") xtitle("Lagged log(wage)") ///
       title("Persistence of Wages") ///
       note("Strong correlation across years within the same
           student")
graph export "logw_vs_L1logw.png", replace
```

Tip: Understanding the `bysort id (year)` prefix

The prefix `bysort id (year):` tells Stata to:

1. **Sort** the data by `id` and then by `year` within each individual.
2. **Execute** the command (after the colon) separately within each `id` group.

Generalized pattern for panel data operations:

`bysort panel_id (time_variable): gen newvar = oldvar[_n - k]`

Where:

- `panel_id` = unit of observation (firm, person, region, etc.)
- `time_variable` = time dimension (year, quarter, month, etc.)
- `oldvar` = the variable you are lagging
- `k` = number of periods back (1 for a one-period lag)

Examples:

```
bysort firmid (year): gen L1_sales = sales[_n-1]    // one-year
    lag of sales per firm
bysort country (year): gen L2_gdp   = gdp[_n-2]     // two-year
    lag of GDP per country
```

Tip: From Heteroskedasticity to Clustering: Why We Need `, robust` and `, cluster(id)`

Heteroskedasticity means that the noise term (u_i) does not have a constant variance across observations—some data points are "louder" than others. This variation typically occurs *across individuals or firms.*

* Example: In a wage regression, workers with more experience or education may have more variable earnings than entry-level workers.
* We address this issue using the option `, robust`, which allows Stata to compute standard errors that are valid even when the error variance differs across observations.

Clustering (Serial Correlation) arises once we introduce a **time dimension**. In panel data, we observe the same individual (or firm) repeatedly, and their unobserved characteristics can persist over time. For instance, when we saw that wages are highly persistent, it indicates that some part of the residual variation—perhaps coming from ability, motivation, or background—remains similar from year to year.

* Example 1: The same student's unobserved ability affects wages in every year, so if their error is high in Year 1, it is likely high in Year 2 as well.
* Example 2: A firm's unobserved management quality influences profits across multiple quarters—errors move together within the firm.
* Example 3: People living in the same city experience common shocks such as weather or local market conditions, generating correlated errors within that city.

Because these within-group observations are not independent, we need to adjust our inference to account for this *within-cluster correlation.* Clustering the standard errors by a group variable (here, the student identifier) tells Stata:

> "Treat all years of the same student as one bundle: allow errors to be correlated within that bundle but independent across different students."

Week 9 — Assignment 4: Difference-in-Differences (DID) (Homework)

9.1 Setting

Difference-in-differences (DID) compares how an outcome changes over time in a group that receives a treatment (the *treated* group) to how the outcome changes in a group that does not (the *control* group). The key identifying assumption is **parallel trends**: in the absence of treatment, the average outcomes for treated and control groups would have followed the same time path.

A simple motivating example. Imagine a state-level policy (e.g., a minimum-wage increase) implemented in a subset of counties in 2005. We observe county-level outcomes (e.g., average hourly wage) for several years before and after 2005. Some counties ("treated") are exposed to the policy starting in 2005; others ("controls") never receive the policy. DID asks: how much did wages in treated counties change after the policy, relative to the change in control counties?

Note

Notation. Let Y_{it} be the outcome for unit (county) i at time t. Define

$$D_i = \begin{cases} 1 & \text{if unit } i \text{ is in the treated group,} \\ 0 & \text{if unit } i \text{ is in the control group,} \end{cases} \quad \text{and} \quad \text{Post}_t = \begin{cases} 1 & \text{if } t \text{ is a post-treatment period,} \\ 0 & \text{if } t \text{ is a pre-treatment period.} \end{cases}$$

A simple DID regression (without fixed effects) can be written as:

$$Y_{it} = \beta_0 + \beta_1 D_i + \beta_2 \text{Post}_t + \beta_3 (D_i \times \text{Post}_t) + \varepsilon_{it}. \tag{9}$$

The DID estimate of the treatment effect is the interaction coefficient, $\hat{\beta}_3$.

2-by-2 table. The table below illustrates the logic of DID: take the change (After − Before) for each group, then difference those changes.

	Before	**After**	**Difference (After–Before)**
Treated ($D_i = 1$)	$\beta_0 + \beta_1$	$\beta_0 + \beta_1 + \beta_2 + \beta_3$	$\Delta_T = \beta_2 + \beta_3$
Control ($D_i = 0$)	β_0	$\beta_0 + \beta_2$	$\Delta_C = \beta_2$
Difference (Treated–Control)	$\Delta_B = \beta_1$	$\Delta_A = \beta_1 + \beta_3$	$\widehat{\text{DID}} = \beta_3$

Table 6: DID decomposition corresponding to regression equation (9).

As you can see from the table, in equation (9):

- β_1 captures any **pre-existing difference in levels** of Y between treated and control groups before the policy.
- β_2 represents the **common time trend** that affects both groups equally (for example, general economic growth).

– β_3 measures the **additional change in Y for the treated group** beyond the common time trend — the causal effect of treatment under the parallel trends assumption.

9.2 What to Submit

Upload your **final and working** versions of the `.do` file, the corresponding `.log` file, and the exported `lab9_results.doc` to Blackboard. Grading will be based only on the files you submit, so please double–check that they are correct and complete before uploading.

9.3 Assignment Tasks

We will use a simulated monthly panel dataset called `lab9_soda_tax_panel.dta`. The data represent 150 counties observed from 2010–2019. In 2015, a group of counties introduced a **soda tax** aimed at reducing sugary drink consumption. Other counties did not adopt the tax and will serve as comparison groups.

Each observation corresponds to a county–month pair and includes information on soda sales and several local characteristics. Key variables are:

- `group` – indicates the county group: there are three groups in total. One group implemented the soda tax (`treated`), while the other two did not (comparison groups).
- `soda_sales_pc` – monthly soda sales per capita (liters per person).
- `treated` – indicator for whether the county implemented the soda tax (1 = treated, 0 = otherwise).
- `post` – indicator for months in 2015 or later (post–policy period).
- `income_pc_month` – average monthly income per person (USD).
- `unemp_rate` – county unemployment rate (percent).
- `store_count` – number of retail outlets selling soda.
- `price_soda` – average local soda price (USD per liter).
- `diabetes_rate` – annual diabetes prevalence rate (percent).

Our goal is to use the **difference-in-differences (DID)** framework to estimate how the soda tax affected soda sales, test key assumptions such as parallel trends, and visualize the results. You will begin by exploring the dataset, checking pre-trends across groups, estimating the DID regression, and then extending the analysis to clustered inference and event-study plots.

1. **Set up your working environment.** Start with your default do-file template. Change the working directory to the folder where you saved `lab9_soda_tax_panel.dta`. Before starting, ensure that the following Stata packages are installed:

```
ssc install outreg2, replace
ssc install coefplot, replace
```

2. **Explore and describe the data.** Load the dataset and examine its structure and main variables.

```
describe
summarize soda_sales_pc diabetes_rate income_pc_month unemp_rate
    store_count
tab group
```

Answer the following questions in your lab9_results.doc file:

- What are the three groups in the data, and how many counties belong to each? (*Hint : We have monthly data for 10 years*)
- Which variables change over time within a county (i.e., vary within panel units)?
- Report the average monthly soda sales for each group (using `collapse` or `table`).

3. **Visual check: parallel trends.** Before running the DID regression, we first need to check whether the **parallel-trends assumption** appears reasonable. Compute and plot the group means of `soda_sales_pc` over time. (This corresponds to $E[Y_{it} \mid D_i, Post_t]$). Compare the treated group to each control group using the code template below. Fill in the blanks where indicated.

```
preserve
collapse (mean) soda_sales_pc, by(year group)
twoway (line __ __ if group=="__", lcolor(red)) ///
       (line __ __ if group=="__", lcolor(blue)) ///
       (line __ __ if group=="__", lcolor(green)), ///
       legend(order(1 "__" 2 "__" 3 "__")) ///
       xline(2015, lpattern(dash) lcolor(black)) ///
       title("Parallel Trends Check") ///
       ytitle("Average soda sales per capita")
graph export "parallel_trends.png", replace
restore
```

In your lab9_results.doc, include the following:

- Which control group appears to satisfy the parallel-trends assumption? Briefly explain your reasoning.
- Insert and label the exported graph (`parallel_trends.png`).

4. **Baseline DID regression.** We are going to modify equation (9) slightly to match our setting. Equation (9) was based on a simple two-group and two-period DID framework. However, our dataset contains multiple treated and control counties observed over many years. In this case, we generalize the model to the following two-way fixed effects DID specification:

$$\texttt{soda\_sales\_pc}_{it} = \alpha_i + \delta_t + \beta\big(\texttt{treated}_i \times \texttt{post}_t\big) + u_{it}. \tag{10}$$

Here, α_i represents a dummy variable for each county (**county fixed effect**), which captures all time-invariant differences across counties — such as local preferences for soda, climate, or population characteristics. The δ_t terms represent dummy variables for each year (**year fixed effects**), which capture common shocks that affect all counties in a given year — such as national health campaigns or changes in soda prices. Finally, the DID estimator β measures the **causal effect of the soda tax**: how much more (or less) soda sales changed in treated counties after 2015 relative to control counties, after removing county and year effects.

To estimate equation (10), follow the steps below.

1 Drop observations that belong to the non-chosen control group.

```
* keep treated and the chosen control group
keep if inlist(group,"treated","____")
```

2 Create a dummy variable called `post` equal to 1 if the year is 2015 or later.

```
gen post = (write condition here)
```

3 Declare the panel structure to Stata using the `xtset` command, and read the tipbox below before running your first fixed-effects regression.

```
    * Generate monthly time variable for panel declaration
    -> We have Year, and Month. We need Year-Month
    gen ym = ym(year, month)

    * This makes newly created ym be a "time" variable to STATA
    format ym %tm

    xtset __ __
```

Tip: Quick guide: `xtreg`, fixed effects DID

Generic syntax:

```
xtset panel_id time_var
xtreg y x1 x2 i.timevar, fe vce(cluster id)
```

What `xtreg, fe` does:

- Estimates the model with α_i (unit fixed effects) absorbed. This removes all time-invariant unobserved heterogeneity across counties (e.g., persistent taste for soda or local climate differences).
- Including `i.year` adds year fixed effects δ_t, capturing common time shocks that affect all counties (e.g., national soda trends).
- Use `vce(cluster county_id)` to allow serial correlation within counties.

4 Run the baseline two-way fixed effects DID regression and export the table as `lab9_results.doc`.

```
* Run baseline two-way fixed effects DID regression
xtreg soda_sales_pc post##treated i.year, fe vce(cluster
    county_id)
* i.year adds year dummies (delta_t) to control for common
    time shocks
* vce(cluster county_id) gives clustered standard errors

outreg2 using "lab9_results.doc", replace ctitle("Original")
```

Interpretation task (write in `lab9_results.doc`): Interpret the coefficient on the interaction term `post#treated` as the causal effect of the soda tax. Explain whether the sign (positive or negative) makes sense.

Note

Why didn't we include other covariates in the baseline? We can, but the two-way FE DID (county FE + year FE) identifies the causal effect under the parallel-trends assumption using within-county changes over time. Time-varying covariates (e.g., `income_pc_month`, `unemp_rate`, `price_soda`, `store_count`) can be added to improve precision or adjust for pre-determined confounders, but do not "fix" a violated parallel-trends assumption. If you include covariates, prefer pre-treatment or lagged versions and cluster SEs at the county level.

5. **Inference: why clustering matters.** Let's first visually understand what clustering means.

```
* Visualize average residuals by county
* For xtreg , e gives residuals
predict resid, e

bysort county_id: egen mean_resid = mean(resid)
hist mean_resid, title("Distribution of county mean residuals")

graph export ....
```

If the error terms (u_{it}) were *independent across counties and over time*, the mean residuals by county should:

- fluctuate randomly around zero, and
- have a distribution tightly centered around zero.

If the histogram instead shows that:

- many counties have **consistently positive or negative mean residuals**, or
- the distribution is **wide or skewed**,

then **residuals are systematically correlated within counties**. This means certain counties tend to have residuals that are mostly positive (consuming more soda than predicted) and others mostly negative (consuming less than predicted). In other words, residuals are not independent within clusters. If a county has unobserved features that make soda consumption persistently high, these features will cause its residuals to remain correlated over time.

When this happens, the standard errors from the usual heteroskedasticity-robust specification are **too small**, and the resulting t-statistics are **too large**. To correct this, you should cluster standard errors at the county level. `vce(cluster county_id)` allows arbitrary correlation of errors within each county while assuming independence across counties.

Question (write in `lab9_results.doc`): The histogram shows that several counties have mean residuals noticeably different from zero (i.e., some density at the extremes). What does this imply about the independence of residuals across time within a county? Based on this evidence, should we be concerned about clustering?

6. **Placebo test.** To further check the credibility of your DID estimate, run a **placebo test** by pretending that the soda tax was implemented two years earlier (in 2013 instead of 2015). If the DID framework is valid and the parallel-trends assumption holds, **you**

should not find a statistically significant treatment effect when the "fake" policy date is used.

```
* Create a fake post-treatment dummy (pretend policy started in
    2013)
gen fake_post = (condition here)

* Re-estimate the DID regression using the fake treatment timing
xtreg ...

* Export results for comparison
outreg2 using "lab9_results.doc", append ctitle("Placebo")
```

Note

If the placebo DID estimate (interaction `fake_post#treated`) is small and statistically insignificant, it suggests that there were no pre-existing differences in soda sales trends between treated and control counties before the actual 2015 policy. A significant placebo effect, however, would cast doubt on the parallel-trends assumption.

7. **Extension: link to health outcome.** Aggregate data to the annual level and test whether lower soda sales correspond to lower diabetes rates.

```
preserve
collapse (mean) soda_sales_pc diabetes_rate income_pc_month
    unemp_rate, by(county_id year group)
regress diabetes_rate soda_sales_pc income_pc_month unemp_rate,
    vce(cluster county_id)
restore
```

Interpret the coefficient on `soda_sales_pc`. Does it have the expected sign?

8. **Wrap-up:** Include in `lab9_results.doc`:
   - Answers to #2, #3, #4, #5, #7
   - Graphs from #2, #5
   - Regression result tables from #4

Week 10 — Fixed Effects (Within Estimator) (Teaching)

10.1 New Goals for Week 10

1. Understand why and when we use **fixed effects** in panel data:
   - Account for unobserved time-invariant heterogeneity correlated with regressors.
   - Interpret the fixed-effects (within) estimator as using only *within-unit* variation.
   - Recognize that including a full set of entity dummies (the *Least Squares Dummy Variable* or LSDV approach) produces the same coefficient estimates as the within estimator.
2. Learn the practical Stata workflow for fixed-effects estimation:
   - Declare a panel (`xtset`) and convert string IDs (`encode`).
   - Run `reg` with entity dummies, then `xtreg, fe`, and compare results.
   - Include time fixed effects and add lags to capture delayed responses.
   - Perform diagnostics: joint test for time FE, groupwise heteroskedasticity, serial correlation, and cross-sectional dependence.
3. Know alternatives and computational shortcuts:
   - Use `reghdfe` to absorb many fixed effects efficiently.
   - When and why to cluster standard errors by panel id.

10.2 Setting for Week 10

We will use the panel dataset `lab10_fixed_panel.dta`, which contains data for multiple countries observed over several years. The goal is to estimate how within-country changes in trade, labor, and related macroeconomic variables are associated with changes in GDP per capita. This lab focuses on applying fixed-effects estimators and conducting diagnostic tests in `Stata`.

Key dataset features

- Panel identifiers:
  - `country` – country name.
  - `country_code` – ISO country code.
  - `year` – calendar year.
- Main economic variables:
  - `gdppc` – GDP per capita (current US$).
  - `export` – export value index (2015 = 100).
  - `import` – import value index (2015 = 100).
  - `trade` – total trade index (sum of exports and imports).
  - `labor` – labor force participation rate (% of population ages 15+).
  - `capital` – gross capital formation (current US$).
  - `pop_growth` – population growth (annual %).
  - `inflation` – inflation rate based on GDP deflator (annual %).
  - `school_enroll` – tertiary school enrollment rate (% gross).
- Data source: World Development Indicators (World Bank), years 2000–2024.
- The panel may be unbalanced due to missing observations in some variables or years.

10.3 Tasks for Week 10

0 Start with your default do-file template.

1 **Prepare the panel and declare the panel structure.**
Load the dataset `Week10_long_panel.csv`, convert the string identifier `country_code` into a numeric id, and declare the panel in Stata.

```
* Load the dataset (CSV format)
import delimited "Week10_long_panel.csv", clear

* Encode string country code to numeric id
encode country_code, gen(country_id)

* Declare panel structure: panel id = country_id, time = year
xtset country_id year

* Quick panel summary
xtdescribe
```

Tip: Encoding the panel identifier

The variable `country_code` is a string (e.g., "USA", "KOR"). Stata's panel commands require a numeric identifier, so we use:

```
encode country_code, gen(country_id)
```

This creates a numeric id while preserving the original string labels, allowing `xtset` to recognize the panel structure correctly.

2 **Construct key variables and explore the data.**

```
* Construct total trade (export + import)
gen trade = your code here

* Create log-transformed variables
gen ln_gdppc = your code here
gen ln_trade = your code here
gen ln_labor = your code here
gen ln_capital = your code here

* Basic pooled summary
sum gdppc trade labor

* Panel summary: overall / between / within variation
xtsum gdppc trade labor

* Histograms: GDP per capita (level vs log)
hist gdppc, normal name(h1, replace) ///
    title("GDP per capita (level)")

hist ln_gdppc, normal name(h2, replace) ///
    title("log(GDP per capita)")
```

```
graph combine h1 h2, col(2) ///
    title("Distribution of GDP per capita (level vs log)") ///
    note("Visual comparison shows that logging reduces skewness")
```

Tip: Interpreting `xtsum`

The command **`xtsum`** reports three components of variation for each variable:

- **Overall:** total variation across all observations.
- **Between:** variation across units (countries).
- **Within:** variation within each unit over time.

The fixed-effects estimator relies on the *within* variation. If within variation is small, FE estimates may be imprecise or unstable.

3 Visualize trends in GDP per capita across time.

Before running regressions, it is helpful to **visualize how GDP per capita evolves over time**. This allows us to see whether growth patterns differ across countries and whether global income levels move together. For example, if most countries experience a similar upward trend, this suggests that **time effects (such as global business cycles or common shocks)** may be important to control for later with year fixed effects.

```
* Visualize cross-year variation in GDP per capita
preserve

* Compute yearly mean GDP per capita across countries
bysort year: egen mean_gdppc = mean(gdppc)

* Scatterplot of GDP per capita by year, with yearly mean line
twoway (scatter gdppc year, msymbol(circle_hollow) color(gs14)) ///
    (connected mean_gdppc year, sort msymbol(diamond)), ///
    title("GDP per capita over time") ///
    xtitle("Year") ytitle("GDP per capita (current US$)") ///
    legend(label(1 "Country observations") label(2 "Yearly mean")) ///
    note("The upward trend in the mean line indicates global income growth.")

graph export "gdppc_mean_by_year.png", replace

restore
```

4 Fixed Effects Estimation (stepwise)

In this section, we explore how within-country changes in key macroeconomic variables relate to within-country changes in GDP per capita. We begin by estimating a simple pooled OLS model that **ignores** country-specific effects, and then progressively introduce fixed effects to account for unobserved heterogeneity. For now, we ignore non-linearities

and lag effects so that we can focus on understanding the mechanics and interpretation of fixed effects.

$$\ln(\text{gdppc}_{it}) = \alpha_i + \beta_1 \ln(\text{trade}_{it}) + \beta_2 \ln(\text{labor}_{it}) + \beta_3 \ln(\text{capital}_{it}) + \beta_4 \text{pop\_growth}_{it} + \beta_5 \text{inflation}_{it} + \beta_6 \text{school\_enroll}_{it} + u_{it}.$$

The term α_i represents all unobserved, time-invariant characteristics of each country that may influence its level of GDP per capita—factors such as *geography, colonial history, institutional quality, or long-run technological capacity.* By including α_i, we allow each country to have its own baseline intercept, acknowledging that nations differ in ways that do not change over time but still affect economic performance. If these unobserved characteristics are correlated with any of the explanatory variables, omitting α_i would bias the estimated coefficients. Fixed-effects estimation effectively removes α_i by focusing on how changes within a country over time relate to changes in its explanatory variables, thereby isolating the *within-country* variation that identifies $\beta_1, \ldots, \beta_6$.

1. **Pooled OLS (no fixed effects).**

   Begin by estimating a pooled OLS model, treating all observations as independent. This ignores any country-specific effects (α_i) and assumes all heterogeneity is captured in the error term.

```
* 4.1 Pooled OLS regression (ignores country-specific effects)
reg ln_gdppc ln_trade ln_labor ln_capital pop_growth inflation
    school_enroll, robust
outreg2 using week10_results.doc, replace ctitle("Pooled OLS")
```

Tip: Interpretation

In this pooled regression, unobserved country-specific factors such as geography, institutions, or natural resources are absorbed into the error term. If these factors are correlated with the regressors, OLS estimates will be biased.

2. **LSDV: OLS with country dummies (explicit fixed effects).**

   Add a full set of country dummies (`i.country_id`) to estimate the Least Squares Dummy Variable (LSDV) model. This explicitly controls for time-invariant country effects.

```
* 4.2 LSDV regression (OLS with country dummies)
reg ln_gdppc ln_trade ln_labor ln_capital pop_growth inflation
    school_enroll i.country_id, ///
    vce(cluster country_id) hascons
outreg2 using week10_results.doc, append ctitle("LSDV") ///
    keep(ln_trade ln_labor ln_capital pop_growth inflation
       school_enroll)
```

Tip: LSDV vs. Within Estimator

The LSDV model estimates the same slope coefficients (β) as the fixed-effects (within) estimator but explicitly includes one dummy variable for each country. This can become computationally intensive when the number of groups is large, which is why we typically use `xtreg, fe`.

3. **Within estimator using `xtreg, fe`.**

   Next, estimate the fixed-effects model using the within transformation.

```
* 4.3 Fixed effects (within estimator)
xtreg ln_gdppc ln_trade ln_labor ln_capital pop_growth
    inflation school_enroll, fe robust
outreg2 using week10_results.doc, append ctitle("xtreg") ///
    keep(ln_trade ln_labor ln_capital pop_growth inflation
        school_enroll)
```

Tip: Interpreting the FE coefficients

The fixed-effects estimator removes all time-invariant country characteristics by demeaning the data within each country. Coefficients capture the effect of *within-country changes* in the explanatory variables on *within-country changes* in GDP per capita.

Discussion: If we included time-invariant variables such as `latitude` or `area`, they would be dropped automatically. This is because these variables do not vary over time within a country: once we include country fixed effects (α_i), knowing which country an observation belongs to already determines its latitude or area exactly. In other words, such variables are perfectly collinear with the country dummies and provide no additional within-country variation for estimation.

4. **Diagnostics for the FE model.**

   After estimating the fixed-effects model, perform diagnostic tests to check key assumptions.

```
* 4.4 Diagnostic tests after FE estimation
* Groupwise heteroskedasticity
ssc install xttest3, replace
xttest3

* Serial correlation (Wooldridge test)
capture which xtserial
if _rc {
    net from http://www.stata-journal.com/software/sj3-2/
    net describe st0039
    net install st0039
}

* Run Wooldridge test for first-order serial correlation
xtserial ln_gdppc ln_trade ln_labor ln_capital pop_growth
    inflation school_enroll
```

```
16
17 * Re-run with clustered SEs if issues are detected
18 xtreg ln_gdppc ln_trade ln_labor ln_capital pop_growth
      inflation school_enroll, fe vce(cluster country_id)
19 outreg2 using week10_results.doc, append ctitle("xtreg (
      clustered)") ///
20     keep(ln_trade ln_labor ln_capital pop_growth inflation
         school_enroll)
```

Tip: Why cluster?

If heteroskedasticity or serial correlation is detected, clustered standard errors allow arbitrary correlation of residuals within each country while assuming independence across countries.

5. **Adding time fixed effects.**

   From Task 3, we saw that global GDP per capita trends upward over time, suggesting that time effects may be important. Add year dummies (`i.year`) to capture global shocks or common trends.

$$\ln(\text{gdppc}_{it}) = \alpha_i + \gamma_t + \beta X_{it} + u_{it}$$

   Here, X_{it} represents the same set of explanatory variables used in the earlier specifications—$\ln(\text{trade}_{it})$, $\ln(\text{labor}_{it})$, $\ln(\text{capital}_{it})$, pop_growth_{it}, inflation_{it}, $\text{school\_enroll}_{it}$. Including the year effects γ_t allows the model to account for macroeconomic shocks or global trends that influence all countries simultaneously.

```
1 * 4.5 Two-way fixed effects: country and year FE
2 xtreg ln_gdppc ln_trade ln_labor ln_capital pop_growth
      inflation school_enroll i.year, fe robust
3 outreg2 using week10_results.doc, append ctitle("xtreg + year
      FE") ///
4     keep(ln_trade ln_labor ln_capital pop_growth inflation
         school_enroll)
5
6 * Test whether year fixed effects are jointly significant
7 testparm i.year
```

Tip: What do year effects capture?

Time fixed effects (γ_t) absorb common shocks affecting all countries in a given year, such as global recessions, commodity price swings, or pandemics. The test `testparm i.year` checks whether these year effects are jointly significant.

6. **Optional: Using `reghdfe` for many fixed effects.**

   For large datasets with many groups or additional dimensions of fixed effects, use the `reghdfe` command (after installing it). It is computationally efficient and produces identical coefficient estimates to `xtreg, fe` for the same specification.

```
1 * Install if needed
```

```
ssc install reghdfe, replace
ssc install ftools, replace

* Country fixed effects
reghdfe ln_gdppc ln_trade ln_labor ln_capital pop_growth inflation school_enroll, ///
    absorb(country_id) vce(cluster country_id)

* Country and year fixed effects
reghdfe ln_gdppc ln_trade ln_labor ln_capital pop_growth inflation school_enroll, ///
    absorb(country_id year) vce(cluster country_id)
```

Tip: When to use `reghdfe`

`reghdfe` is ideal when you need to absorb multiple high-dimensional fixed effects, such as country, industry, and year simultaneously. It is faster and more memory-efficient than the standard `xtreg` command.

Week 11 — Assignment 5: Endogeneity & Instrumental Variables (IV) (Homework)

11.1 Setting

In many empirical applications, we are interested in the causal effect of a firm's decision or strategic choice on an economic outcome. However, these decisions are often **not randomly assigned**. This leads to an **endogeneity** problem: the explanatory variable of interest is correlated with unobserved factors that also influence the outcome, causing simple OLS regressions to be biased.

A motivating example: advertising and product sales. Consider a panel of retail stores $i = 1, \ldots, N$ observed over several years $t = 1, \ldots, T$. Each store chooses how much to spend on advertising in each period, and we observe:

- $\texttt{sales}_{it}$ – product sales at store i in year t,
- $\texttt{ad\_spending}_{it}$ – advertising expenditures in that year,
- $\texttt{foot\_traffic}_{i}$ – average pedestrian traffic near the store (time–invariant; reflects baseline local demand),
- $\texttt{budget\_shock}_{it}$ – a firm-wide advertising budget shock (e.g., corporate marketing push; varies by year and is unrelated to individual store characteristics),
- $\texttt{ad\_cost\_index}_{it}$ – a local media advertising cost index (e.g., fluctuations in radio/TV slot prices; varies over time and across markets).

Our goal is to estimate whether increasing advertising expenditures actually raises sales. A key difficulty in this setting is that stores differ in many unobserved ways. Some stores are located in busy commercial areas, have a loyal customer base, or enjoy strong brand recognition. These features are not recorded in our dataset but clearly affect both advertising decisions and sales.

To make this concrete, suppose the true (but unobserved) relationship between advertising and sales is:

$$\log \texttt{sales}_{it} \;=\; \beta_0 \;+\; \beta_1 \log \texttt{ad\_spending}_{it} \;+\; \beta_2 \, \texttt{popularity}_i \;+\; u_{it}, \tag{11}$$

where $\texttt{popularity}_i$ is a time–invariant store characteristic that captures location quality, demand potential, and other persistent advantages.

Because more popular stores tend to spend more on advertising *and* have higher sales for reasons unrelated to advertising, $\log \texttt{ad\_spending}_{it}$ is correlated with $\texttt{popularity}_i$. If we estimate the simplified OLS regression

$$\log \texttt{sales}_{it} = \beta_0 + \beta_1 \log \texttt{ad\_spending}_{it} + \varepsilon_{it}, \tag{12}$$

the omitted variable $\texttt{popularity}_i$ becomes part of the error term, leading to a biased estimate of β_1. In other words, OLS confuses the effect of advertising with the underlying differences in store popularity.

To address endogeneity, we use **instrumental variables (IVs)**: factors that shift advertising spending but are otherwise unrelated to unobserved determinants of sales. In this lab, you will analyze a simulated panel dataset in which each store faces multiple potential sources of variation in advertising:

- a time–varying *firm-wide advertising budget shock* (`budget_shock`), which raises or lowers advertising resources across all stores in a given year and serves as a plausibly **valid and strong** instrument;
- a time–varying *local advertising cost index* (`ad_cost_index`), which captures fluctuations in media slot prices (e.g., radio, online ads) and acts as a plausibly **valid but weak** instrument;
- a time–invariant measure of pedestrian activity near each store (`foot_traffic`), which affects advertising decisions but is also directly related to local demand and store popularity, making it a **potentially invalid** instrument.

The goals of this assignment are to:

- show how endogeneity leads OLS to produce biased advertising–sales estimates,
- use IV methods to recover the causal effect of advertising under the IV assumptions, and
- compare two candidate instruments based on their **relevance** and **exogeneity**.

11.2 What to Submit

Upload your **final and working** versions of the `.do` file, the corresponding `.log` file, and the exported `lab11_results.doc` to Blackboard. Grading will be based only on the files you submit, so please double–check that they are correct and complete before uploading.

11.3 Assignment Tasks

1. **Set up your default do-file template.** Change your working directory to the location where you saved `ads_iv_full.csv`. Install the following packages if needed:

```
ssc install outreg2, replace
```

2. **Load and explore the dataset.** Import the dataset and examine its structure:

```
import delimited "ads_iv_full.csv", clear
describe

* Summarize main variables
summarize sales ad_spending foot_traffic budget_shock ///
          ad_cost_index popularity
```

   Create log values for `sales` and `ad_spending`. Then, answer the following questions in `lab11_results.doc`:

   - How many stores and how many years are in the dataset?
   - Which variables vary over time? Which are time-invariant?
   - Report the average values of `sales`, `ad_spending`, and `foot_traffic`.

3. **Correct OLS regression (with popularity).** Suppose the magic cat Coco did wonders and, for this particular assignment, you now have access to each store's `popularity`. Estimate the "correct" regression from equation (11):

$$\log \texttt{sales}_{it} \;=\; \beta_0 \;+\; \beta_1 \log \texttt{ad\_spending}_{it} \;+\; \beta_2 \texttt{popularity}_i \;+\; u_{it}.$$

   Run this regression in Stata and export the results to `lab11_results.doc` using `outreg2` with an appropriate column title (e.g., "Correct OLS").

4. **Naive OLS regression (biased).** Now the whimsical cat Coco vanishes, taking the `popularity` variable with her. From this point on, you no longer observe `popularity`, just as in real empirical work. Estimate the naive log–log OLS model from equation (12):

$$\log \texttt{sales}_{it} = \beta_0 + \beta_1 \log \texttt{ad\_spending}_{it} + \varepsilon_{it}.$$

   Export the results to `lab11_results.doc` (e.g., column title "OLS (Biased)").

   In `lab11_results.doc`, answer:

   - What is the estimated effect of advertising on sales according to the **correct** model?
   - What is the estimated effect according to the naive model?
   - By how much does the naive OLS estimate differ from the correct one when `popularity` is omitted? (Report the difference in estimated elasticities.)

5. **Visual exploration: relationships among variables.** You now decide to use an **IV** approach. Before running any regressions, it is helpful to check visually whether each potential instrument is correlated with advertising.

   Produce scatter plots of:

   - `ln_ad_spending` vs. `budget_shock` (valid, strong IV),
   - `ln_ad_spending` vs. `ad_cost_index` (valid, weak IV),
   - `ln_ad_spending` vs. `foot_traffic` (potentially invalid IV).

   Export all three plots and include them in `lab11_results.doc`. Briefly describe what each graph suggests about the strength and plausibility of each instrument.

6. **First-stage regressions: instrument relevance.** Now formally test whether each candidate IV is **relevant** by running separate first-stage regressions where the endogenous variable `ln_ad_spending` is regressed on each instrument:

   - First stage 1: `ln_ad_spending` on `budget_shock` (valid, strong IV),
   - First stage 2: `ln_ad_spending` on `ad_cost_index` (valid, *weak* IV),
   - First stage 3: `ln_ad_spending` on `foot_traffic` (potentially invalid IV).

   In `lab11_results.doc`, answer the following:

   - Is each instrument **statistically relevant**? In other words, does it significantly predict advertising in the first-stage regression?
   - Which instrument is **strongest**? When evaluating instrument strength, look not only at the size of the coefficient but also at the *first-stage F-statistic*. (A larger F-statistic means the instrument explains more variation in advertising and is therefore "stronger.")
   - Which instrument appears **weak**? An instrument is considered weak if it has only a small effect on advertising *and* produces a low first-stage F-statistic. Explain your reasoning using both pieces of evidence.

- Even if an instrument is strong (for example, `foot_traffic`), does that guarantee it is *valid*? Briefly explain why strength alone does **not** ensure exogeneity.

7. **IV estimation: recover the causal effect of advertising.** We now want to estimate the causal effect of advertising using each instrument separately. You will run three 2SLS regressions for the model:

$$\log \texttt{sales}_{it} = \beta_0 + \beta_1 \log \texttt{ad\_spending}_{it} + \varepsilon_{it},$$

where `ln_ad_spending` is treated as endogenous.

Tip: Quick guide: `ivregress 2sls`

The basic syntax for a single endogenous regressor and a single instrument is:

```
ivregress 2sls depvar (endogvar = instrument), vce(robust)
```

For example:

```
ivregress 2sls y (x = z), vce(robust)
```

Here:

- `y` is the dependent variable,
- `x` is the endogenous regressor,
- `z` is an instrumental variable that shifts `x` but is otherwise unrelated to the error term.

Important: If an instrument is weak (i.e., has a low first-stage F-statistic), the IV estimate may be very imprecise or unstable. If an instrument is invalid (i.e., correlated with omitted variables), the IV estimate will be biased even if the first stage is strong.

Using this template, run:

- an IV regression using `budget_shock` as the instrument for `ln_ad_spending` (valid, strong IV),
- an IV regression using `ad_cost_index` as the instrument for `ln_ad_spending` (valid, *weak* IV),
- an IV regression using `foot_traffic` as the instrument for `ln_ad_spending` (potentially *invalid* IV).

Export all IV results to `lab11_results.doc` with clear column titles (e.g., "IV: budget_shock", "IV: ad_cost_index", and "IV: foot_traffic").

In your write-up:

- Which IV produces an advertising elasticity closest to the **correct** OLS estimate from equation (11)?
- How does the estimate from the `ad_cost_index` IV illustrate the issues that arise with a **weak** instrument?
- Based on the economic story, why is `foot_traffic` likely invalid, even if it strongly predicts advertising?

(*Optional: You may try a fixed-effects regression for comparison. In this particular setting, FE performs remarkably well and provides a useful benchmark!*)

12 Command Directory

12.1 Basic

12.1.1 Comments & Line Control

```
* single-line comment
/* multi-line comment */
/// line continuation (use at end of line)
```

12.1.2 Working Directory

- `pwd` : show current working directory.
- `cd "path"` : change working directory.
- `dir` or `list` : list files in current directory.

12.1.3 Using `help` to check syntax

```
help regress
help summarize
search outreg2
viewsource regress.ihlp    // view help file source (read-only)
```

12.1.4 Loops (`forvalues` and `foreach`)

```
// Numeric loops
forvalues y = 2018/2024 {
    display "Year = `y'"
    // use file`y'.dta, clear
}

// Over lists (variables, strings, macros)
foreach v in price sqrft bdrms {
    summarize `v'
}
foreach ext in png pdf eps {
    graph export "myplot.`ext'", replace
}
```

12.1.5 Session Control

- `set more off` : prevent "–more–" pauses when running do-files.
- `clear all` : remove data and value labels from memory.
- `cls` : clear the Results window for a fresh output screen.

12.1.6 Log Files

- `capture log close` : safely close any open log (avoids an error if none is open).
- `log using "path\logfile.smcl", replace` : start a new log in SMCL format.
- `log using "path\logfile.log", replace` : start a new plain-text log.
- `log close` : end the current log.

```
capture log close
log using "output\session_log.smcl", replace
// ... your commands ...
log close
```

12.2 Data

12.2.1 Load, Browse, Export

- `use filename [, clear]` : load `.dta`.
- `import delimited filename [, options]` : import CSV/TXT.
- `browse` : open data browser (read-only).
- `save filename [, replace]` : save `.dta`.
- `export delimited using filename [, options]` : export CSV/TXT.

```
use "housing.dta", clear
browse
save "housing_clean.dta", replace
export delimited using "housing_clean.csv", replace
```

12.2.2 Descriptive tools

Variable overview and listing

- `des [varlist]` : describe variables.
- `list varlist [if] [in]` : display rows with conditions/ranges.

```
des
des price sqrft bdrms

list price sqrft in 1/10
list price sqrft bdrms if bdrms>=3
```

Summaries and tabulations

- `sum [varlist] [, detail]` : summary stats; `detail` adds percentiles.
- `tab var` : one-way tabulation; `tab var1 var2` two-way.
- `count [if]` : count obs meeting a condition.

```
sum price sqrft bdrms
sum price, detail

tab bdrms
tab bdrms region
count if price > 500000
```

12.2.3 Visual descriptive tools (graphs & export)

Flexible `twoway` overlays

- General pattern:
  `twoway (plottype1 y1 x1 [, opts1]) (plottype2 y2 x2 [, opts2]) ..., graph_opts`
  - Common `plottypes`: `scatter`, `line`, `connected`, `lfit`, `qfit`, `kdensity`, `function`.
  - Useful `graph_opts`: `title("...")`, `xtitle("...")`, `ytitle("...")`, `legend(...)`, `name(gname, replace)`.

```
// Example A: scatter + fitted line
twoway (scatter price sqrft) (lfit price sqrft), ///
      title("Price vs. Square Feet") xtitle("Sq Ft") ytitle("Price")

// Example B: overlay two kernel densities
twoway (kdensity wage if female==0) (kdensity wage if female==1), ///
      legend(order(1 "Male" 2 "Female")) title("Wage Distributions")

// Example C: add a theoretical function
twoway (scatter y x) (function y = 2 + .5*x, range(x)), ///
      title("Data with Theoretical Line")
```

Histograms (with overlays)

- `histogram var [, freq | percent | density bin(#) start(#) width(#) ...]`
  To *overlay* other plots (e.g., `kdensity`), use `addplot()`.

```
// Basic histogram
hist price, bin(30) percent title("Price Distribution")

// Histogram + kernel density overlay (natural pairing)
hist price, bin(30) percent addplot(kdensity price) ///
      title("Price: Histogram with Kernel Density")

// Two histograms by group (save and combine)
hist price if bdrms==2, percent name(h2, replace) title("Price: 2
    Bedrooms")
hist price if bdrms==3, percent name(h3, replace) title("Price: 3
    Bedrooms")
graph combine h2 h3, cols(2) name(h23, replace)
```

Exporting graphs

- `graph export "filename.png", replace` : also `.pdf/.eps`.

```
graph export "outputs/price_scatter.png", replace
```

12.2.4 Generating and transforming variables

`gen` and `egen` Stata provides two main commands to create new variables:

- `gen newvar = expression`
  Used for **row-wise transformations**. Each new value is computed directly from variables in the same observation. Example: taking logs, squares, or creating indicator dummies.
- `egen newvar = function(varlist) [, by(group)]`
  Used for **group-wise or more complex operations**. It can compute summary statistics (mean, max, count, rank, etc.) within groups, or create unique group identifiers. These are functions that `gen` cannot do directly.

```
// Row-wise transforms
gen lprice   = log(price)
gen sqrft2   = sqrft^2
gen large    = (sqrft >= 2000)

// Group stats
egen mean_price_by_city = mean(price), by(city)
egen gid = group(city neighborhood)
```

Difference and lag variables Many analyses require creating *lagged*, *lead*, or *differenced* versions of variables. These transformations allow us to capture dynamic relationships (e.g. how last year's trade affects current growth) or to compute growth rates from levels.

In Stata, you must first declare the data's panel/time structure with `xtset` (for panel data) or `tsset` (for pure time series). After this, you can use Stata's built-in time operators:

- `L.var` : lag of `var` (e.g. `L1.price` = last year's price).
- `F.var` : lead of `var` (e.g. `F1.price` = next year's price).
- `D.var` : difference of `var` (e.g. `D.price` = $\text{price}_t - \text{price}_{t-1}$).

```
// Example: city-by-year panel
xtset city year   // declare panel: city = panelvar, year = timevar

gen d_price  = D.price          // first difference of price
gen L1_price = L.price          // lag-1 price
gen g_price  = 100*(D.lprice)   // approx % growth ( log  * 100)
```

12.2.5 Reshaping data

Stata's `reshape` command reorganizes the layout of your dataset:

- `reshape long stub, i(id) j(time)`
  Converts from **wide** format (one row per id, many year columns) to **long** format (multiple rows per id, with year stacked in one column).
- `reshape wide stub, i(id) j(time)`
  Converts from **long** format (one row per id–time) back to **wide** format (one row per id, variables spread across year columns).

```
// Wide -> Long: price2019 price2020 ...
reshape long price, i(houseid) j(year)

// Long -> Wide
reshape wide price, i(houseid) j(year)
```

For a more detailed walk-through with World Bank panel data, see the Week 4 teaching materials.

12.2.6 Merging data

Stata's `merge` command combines datasets by matching on key variables:

- `merge 1:1 keyvars using file.dta`
  One observation in master matches one observation in using (1-to-1).
- `merge m:1 keyvars using file.dta`
  Many observations in master match one in using (many-to-1).
- `merge 1:m keyvars using file.dta`
  One observation in master matches many in using (1-to-many).

```
use "houses.dta", clear
merge m:1 zipcode using "zip_income.dta"

// Inspect merge result via _merge
tab _merge
drop if _merge==2  // drop using-only if appropriate
drop _merge
```

For more on interpreting merge diagnostics (such as _merge codes), see the Week 4 materials.

12.3 Regression

12.3.1 Basic `regress` syntax

- `regress depvar indepvars [if] [in] [weight], [options]`

```
reg price sqrft bdrms
reg lprice sqrft bdrms i.city, robust
```

12.3.2 Accessing results & robust SE

- `_b[var]` : coefficient; `_se[var]` : standard error.
- `e(r2)`, `e(N)` : stored results.
- `predict newvar, xb` (fitted) `predict newvar, resid` (residuals).
- `, robust` : heteroskedasticity-robust SE.

```
reg lprice sqrft bdrms, robust
di "Beta_sqft = " _b[sqrft]
di "R2 = " e(r2)
predict yhat, xb
predict uhat, resid
```

12.3.3 Factor variables: ##, interactions, and margins

Stata's **factor variable** notation (`i.`, `c.`, `#`, `##`) lets you specify interactions and polynomial terms directly inside regression commands. This avoids the need to manually generate new variables and ensures that Stata correctly handles dummies, interactions, and post-estimation tools such as `margins`.

- `c.x1##c.x2` : interaction of two continuous variables (includes x_1, x_2, and $x_1 \times x_2$).
- `i.group##c.x` : interaction of a discrete factor and a continuous variable (e.g. fixed effects × slope).
- `margins` : compute predicted values or contrasts at specific values of regressors.

```
// Interaction of square feet and bedrooms
reg lprice c.sqrft##c.bdrms, robust

// City fixed effects with interaction
reg lprice c.sqrft##i.city bdrms, robust

// Marginal effects at specified values
margins, at(sqrft=(1000(500)3000) bdrms=(2 3 4)) nose
marginsplot, xdimension(sqrft)
```

Tip: Why use ##?

- If you manually create $x_1 \times x_2$ with `gen`, Stata will not automatically include the main effects or treat categorical dummies correctly.
- Using `##` tells Stata to include the main effects *and* the interaction, and it also integrates seamlessly with `margins` and `marginsplot`.

Tip: Getting coefficient labels with factor variables

When you use `##`, Stata generates expanded variable names internally (e.g. `1.city#c.sqrft`). To see the exact names you can use in `lincom` or `test`, run:

```
reg lprice c.sqrft##i.city, robust
reg ..., coeflegend
```

The option `coeflegend` displays the internal coefficient labels you can copy directly into post-estimation commands.

12.3.4 Inference : `test` and `lincom`

Stata's `test` and `lincom` commands allow you to form and test linear combinations of coefficients.

- `test varlist` Wald test of linear or joint hypotheses.

```
// Example regression with interaction
reg lprice c.sqrft##c.bdrms, robust

// ---- Using test ----
// (1) Does the marginal effect of sqrft depend on bdrms?
//    H0: beta3 = 0
test c.sqrft#c.bdrms

// (2) Does sqrft have no effect on lprice at all?
//    H0: beta1 = 0 and beta3 = 0
test c.sqrft c.sqrft#c.bdrms
```

- `lincom expression` Estimate a specific linear combination of coefficients.

```
// ---- Using lincom ----
// (3) Effect of sqrft when bdrms = 3
//     H0: beta1 + 3*beta3 = 0
lincom c.sqrft + 3*(c.sqrft#c.bdrms)

// (4) Difference in sqrft effect between 2 vs 4 bedrooms
//     H0: (beta1 + 4*beta3) - (beta1 + 2*beta3) = 0
//     i.e., the marginal effect of sqrft is the same at 2 vs 4
   bedrooms
lincom (c.sqrft + 4*(c.sqrft#c.bdrms)) - (c.sqrft + 2*(c.sqrft#c.
   bdrms))
```

Tip: Interpreting interaction effects

- `test` is useful for joint significance of coefficients: e.g. $\beta_3 = 0$ (interaction irrelevant) or $\beta_1 = \beta_3 = 0$ (no effect at all).
- `lincom` is best for computing the effect of x_1 at a particular value of x_2, or differences across values.
- Always check coefficient labels using `reg ..., coeflegend` so you know the correct names to use in `lincom`.

12.3.5 Exporting using `outreg2`

Most exports use user-written commands. In this class we standardize on `outreg2` for Word/Excel.

- `outreg2 using filename, [options]` : export summary or regression tables.

```
* One-time installation
capture which outreg2
if _rc ssc install outreg2

* [A] Summary statistics to Word
outreg2 using "output\lab3_summary.doc", replace ///
    sum(log) keep(price sqrft bdrms) dec(3)

* [B] Regressions to Word (append models)
reg lprice sqrft bdrms
outreg2 using "output\reg_results.doc", replace ///
    ctitle("OLS: lprice on sqrft, bdrms") dec(3) bdec(3) tdec(3) ///
    addstat("R-squared", e(r2))

reg lprice c.sqrft##c.bdrms i.city, robust
outreg2 using "output\reg_results.doc", append ///
    ctitle("OLS: + interactions + city FE, robust") dec(3) bdec(3)
       tdec(3)
```