Quantitative Methods Bootcamp

Rocio Mendez Pineda & Tobias Rüttenauer

2023-09-28

Before We Start

- Go to: **https://github.com/ruettenauer/Bootcamp**
- Download:
  - The slides for this bootcamp
  - The dataset we will use (“WDI_Data.dta”)
- Make sure to save the dataset in an easy-to-access folder
  - A place you can also access from elsewhere (e.g., N-drive)

Objectives of This Bootcamp

Overarching goal

Be on common starting point + ready to take on the quantitative MSc modules

- Refresh key statistical concepts
  - E.g., sampling distributions, hypothesis testing
- Obtain basic familiarity with R & Stata

Statistical Inference

Statistical inference is used to learn from incomplete data
We wish to learn some characteristics of a population (e.g., the mean and standard deviation of the heights of all women in the UK), which we must estimate from a sample or subset of that population

Two types of inference:

1. **Descriptive inference**: What is going on, or what exists?
2. **Causal inference**: Why is something going on, why does it exist?

Example data

The code below loads the WDI packages and searches for an indicator on CO2 per capita.

```
# load package
library(WDI)

# Search GDP per capita
WDIsearch("CO2.*capita")
```

```
          indicator
6032 EN.ATM.CO2E.PC
6048 EN.ATM.METH.PC
6059 EN.ATM.NOXE.PC
                                                                    name
6032                                CO2 emissions (metric tons per capita)
6048                  Methane emissions (kt of CO2 equivalent per capita)
6059 Nitrous oxide emissions (metric tons of CO2 equivalent per capita)
```

The code below uses the WDI API to retrieve the data and creates a dataframe of three indicators.

```r
# Define countries, indicators form above, and time period
wd.df <- WDI(country = "all",
             indicator = c('population' = "SP.POP.TOTL",
                           'gdp_pc' = "NY.GDP.PCAP.KD",
                           'co2_pc' = "EN.ATM.CO2E.PC"),
             extra = TRUE,
             start = 2019, end = 2019)

# Drop all country aggregrates
wd.df <- wd.df[which(wd.df$region != "Aggregates"), ]

# Save data
save(wd.df, file = "WDI_short.RData")
```

Descriptive Inference

▶ expand for full code

Descriptive: What are the average CO2 emissions of all highly populated countries countries?

Causal Inference

▶ expand for full code

Causal: How does GDP influence the amount of CO2 emissions?

Variables and Observations

A variable is anything that can vary across units of analysis: it is a characteristic that can have multiple values

- E.g., sex, age, diameter, financial revenue, temperature

A unit of analysis is the major entity that you analyse

- E.g., individuals, objects, schools, countries

An observation is the value of a particular variable for a particular unit (sometimes a unit is in its entirety referred to as observation)

- E.g., the individual King Charles III is 73 years of age

Different Types of Variables

Continuous / interval-ratio variables:

They have an ordering, they can take on infinitely many values, and you can do calculations with them

- E.g., income, age, weight, minutes

Categorical variables:

Each observation belongs to one out of a fixed number of categories

- Ordinal variables: there is a natural ordering of the categories
- Nominal variables: there is no natural ordering of the categories
- E.g., education level, Likert scales, gender, vote choice

Describing variables

Decribing variables

Describing data is necessary because there is usually too much of it, so it does not make any sense to look at every data point
We thus have to look for ways to summarize central tendencies, variation, and relationships that exist in the data
There are many different ways to do this

1. Visual depictions
2. Numerical descriptions

- E.g. mean, mode, median, standard deviation

Distribution

Variables can be characterized by their **frequency distribution**:
The distribution of the (relative) frequencies of their values

- E.g., we can graph the world income distribution:

- ▶ expand for full code

Distributions: Examples

- Normal distribution
- Chi-squared distribution

- ▶ expand for full code

- ▶ expand for full code

Distributions can take on many different shapes

Measures of Central Tendency

Mean: conventional average calculated by adding all values for all units and dividing by the number of units

$$\overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i = \frac{1}{n}(x_1 + \cdots + x_n)\text{, with units } i = (1, 2, \ldots, n)$$

- May give a distorted impression if there are outliers

Median: value that falls in the middle if we order all units by their value on the variable
Mode: most frequently occurring value across all units

Measures of Dispersion

Variance: average of the squared differences between each observed value on a variable and its mean

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

- Why the square? To treat + and – differences alike

Standard deviation: average departure of the observed values on a variable from its mean

$$\sigma = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

- It is the square root of the variance; it “reverts” the square-taking in the variance calculation, to bring the statistic back to the original scale of the variable

Notation

Typically, we use Roman letters for sample statistics and Greek letters for population statistics:

- Sample mean $= \bar{x}$ population mean $= \mu$
- Sample variance $= s^2$, population variance $= \sigma^2$
- Sample standard error $= s$, population standard deviation $= \sigma$

Recall: the sample is what we observe, the population is what we want to make inferences about

Decribing relationships

Decribing relationships

Often, we are not just interested in the distribution of a single variable. [1]
Instead, we are interested in **relationships** between several variables. If there is a relationship between variable, knowing one variable can tell you something about the other variable.

- Age and height
- GDP and CO2 emissions
- Education and income

Hypothesis testing

A hypothesis is a theory-based statement about a relationship that we expect to observe

- E.g., girls achieve higher scores than boys on reading tests

For every hypothesis there is a corresponding null hypothesis about what we would expect if our theory is incorrect

- E.g., there is no association between Y and X in the population
- In our example: girls are not better readers than boys

Covariance

Covariance refers to the idea that the pattern of variation for one variable corresponds to the pattern of variation for another variable: the two variables "vary together"
Statistically speaking, covariance is the multiplication of the deviations from the mean for the first variables and the deviations from the mean for the second variable:

$$cov_{x,y} = \frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})$$

- The covariance tells us the direction of an association: + or –
- It does not tell us about the strength of the association (reference to underlying distributions of variables is missing)

Pearson's Correlation

For continuous data, we can calculate Pearson's correlation (ρ)

- ρ measures strength & direction of association for linear trends
- ρ rescales the covariance to the underlying distributions of the variables involved:

$$\rho = \frac{cov_{x,y}}{\sigma_X \sigma_Y} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

Dividing the covariance by the product of the standard deviations normalizes the covariance to a range from -1 to +1

- -1 = perfectly negative correlation (all points on decreasing line)
- +1 = perfectly positive correlation (all points on increasing line)
- 0 = no correlation (random cloud of points)
- Correlation is weaker closer to 0 and stronger closer to +/-1

An Example

- Code

An Example I

- Code

An Example II

- Code

---

1. Yeah, you could have spend the last 30 minutes on instagram!

Statistical Software

Statistical software

Stata & **R** are powerful software packages that allows you to do:

- Data management and manipulation
- Data visualization
- Statistical analysis
- (Writing papers and presentations)

Writing Stata syntax files and R script facilitates **reproducibility**

Stata's Interface: Variables Window

The variables window displays all variables in your dataset

- Single click on variable names to see details in properties window
- Double click to make variables appear in command window

Stata's Interface: Properties Window

The variables window displays all variables in your dataset

The properties window displays details about selected variables as well as the entire dataset (e.g., number of observations, sort order)

Stata's Interface: Command Window

.

The command window is for entering and executing commands

- But it is better to use do-files (same applies to drop-down menus)

Stata's Interface: Results Window

The results window displays all output of your commands

Stata's Interface: Command History and Current Working Directory

The command history window lists previously run commands

- At the bottom you can see the current working directory: the folder where any files will be loaded from and saved to

Stata's Interface: Opening a Do File

Do files are text files where you can store commands for reuse

- Huge payoffs for reproducibility, debugging, adapting commands

Running Commands from a Do File

After entering a command, you select it, and then click the “execute” button or press “Ctrl+D”

Do File Dos and Don’ts

1. Use annotations to facilitate replicability (incl. for future self!):
    - Use * for single-line comments and /* */ for multiple lines

2. Break down code into clearly labelled sections / subsections
3. Use tab indentations to making things easy to read

4. Don’t put too much information on a single line

- Use /// to continue your command on the next line and
- write “top-to-bottom” instead of “left-to-right”

R & R Studio Interface

Structure of Commands

General Stata Command Syntax

Stata commands mirror everyday commands in their structure:

- They often start with a verb: “Bring me…”
- They then list an object: “… a pint of milk…”
- They may add a condition: “… if it is still before noon…”
- They may specify further details after the comma: “, quickly please” or “, I want semi-skimmed”

In nearly all cases, Stata syntax consists of four parts:

- **Command**: What action do you want to see performed?
- **Names of variables**, files, objects: On what objects is the command to be performed (“varlist”)
- **Qualifier(s) on observations**: Which observations are to be taken into account (and how)? (“if”, “in”, “weight”)
- **Options**: What special things should be done in the execution?

“help [command]” is your friend

```
1  help summarize
```

General R Workflow

```
object_name <- value

# Example
a <- 3
b <- 4
c <- a + b
c
```

- Assign a value to an object

General R Workflow

```
object_name <- value

# Example
a <- 3
b <- 4
c <- a + b
c
```

- Define the objects a and b

General R Workflow

```r
object_name <- value

# Example
a <- 3
b <- 4
c <- a + b
c
```

- Perform an operations with them

General R Workflow

```r
object_name <- value

# Example
a <- 3
b <- 4
c <- a + b
c
```

```
[1] 7
```

- Return the results

General R Syntax

```
function_name(arg1 = val1[which()], arg2 = val2[which()], option1 = v
```

- **function_name**: What action do you want to see performed?
- **args**, files, objects: On what objects is the command to be performed
- **Qualifier(s) on observations**: Which observations are to be taken into account (and how)? (“which”)
- **options**: What special things should be done in the execution?

General R Syntax

Produce a sequence between 0 and 10 with 20 values

```
y <- seq(0, 10, length.out = 20)
y
```

```
 [1]  0.0000000  0.5263158  1.0526316  1.5789474  2.1052632  2.6315789
 [7]  3.1578947  3.6842105  4.2105263  4.7368421  5.2631579  5.7894737
[13]  6.3157895  6.8421053  7.3684211  7.8947368  8.4210526  8.9473684
[19]  9.4736842 10.0000000
```

Calculate the mean

```
mean(y)
```

```
[1] 5
```

Calculate the mean of all values above 5

```
mean(y[which(y >= 5)])
```

```
[1] 7.631579
```

“?[command]” is your friend

```
1  ?seq
```