Course Intro

Gabriel Petrini

August 25th, 2020

Contents

1 Introduction

1.1 What this class is about

This class is a smattering of advanced econometrics topics.

- Understanding the usefulness of structural modeling
- Learning the computational tools for estimating structural models
- Advanced topics in treatment effects and measurement error models

1.2 Applicability of topics

The techniques we will cover are used in a wide variety of fields of applied microeconomics:

- Labor
- Education
- IO
- Public
- Development
- Health
- Urban/Regional
- Environmental
- Others

1.3 What we will cover in the class

1. Basic computing and things you need to think about
2. Coding, version control, reproducibility, workflow
3. Estimating and simulating structural models
4. Subjective expectations models
5. Measurement error correction
6. Treatment effects
7. Machine learning

1.4 Grading

- Problem sets:
  - You must use Julia and write .jl scripts, no Jupyter
  - You can work in groups of up to 3, but you must turn in your own code
- Class participation:
- Midterm exam:
- Paper presentation:
  - You must consult with me at least 1 week prior to your scheduled presentation date to ensure the paper is appropriate for a presentation
- Paper referee report:
  - The paper shouldn't be published, or if it has been published, you should use the earliest pre-print version
- Research proposal:

2 More about Julia

2.1 Basic operations

- **Array indexing:** use `[ ]`
- **Show output:** use `println()`
- **Commenting:** use `#` for single line, `#= ... =#` for multi-line
- **Element-wise operators:** must put a `.` in front, e.g. `x .+ y` if `x` and `y` are arrays
- **Load installed package:** `using Random`
- **Execute script:** `include("myfile.jl")`

2.2 Creating and executing a Julia script

In Julia, even scripts should have functions wrapped around them. Then the script is executed at the REPL by typing `include("myscript.jl")`. Wrapping code in a function allows the JIT compiler to optimize the code.

2.3 Comprehensions

Allows the user in 1 line of code to create an object that could be a complex formula. Exemple:

$$\sum_{t=1}^{T} \beta^t Y_t$$

```
PV = sum([^t*Y[t] for t=1:T])
```