Software Engineering for Economists

What I cannot create, I do not understand.

– Richard Feynman

Description

We acquire basic software engineering skills to tackle computer-intensive economic research projects. During lectures and a group project, we explore the following set of topics:

- Version Control
- Unit Testing
- Debugging and Profiling
- Code Documentation
- Design Patterns
- Data Management
- Cloud Computing
- High Performance Computing

These basic techniques allow us to leverage tools from computational science, increase the transparency of our implementations, and ensure the recomputability of results. Thus, they expand the set of possible economic questions that we can address and improve the quality of our answers.

Additional information and background material is provided on the course website:

`http://policy-lab.org/teaching/softEcon`

This class is accompanied by the *Computational Economics Colloquium*, where experienced researchers present their innovative approaches to economic modeling, numerical methods, and

software engineering. Guest lectures by the *Research Computing Center*, the *Social Sciences Computing Services*, and the *Computation Institute* introduce us to the computing infrastructure at the University of Chicago and the advanced application of computational methods in scientific research.

For additional information, please contact us at `softEcon@policy-lab.org`.

Group Project

The group projects consist of a contribution to *Quantitative Economics*, an open source library for quantitative economic modeling. Each group prepares a high-quality implementation of a quantitative economic model of their choice. We will build on existing material provided as code supplements to several economic textbooks (see course website for selected references). As the course moves along, we use our newly acquired skills to iteratively improve our workflow and the quality of our implementations. Groups report on their progress and receive feedback in a series of presentations throughout the semester.

Tools

We rely five main tools in the class: (1) *Python*, (2) *Ubuntu*, (3) *Microsoft Azure*, (4) *struct-Toolbox*, and (5) *Quantitative Economics*.

- The ***Python*** programming language is accessible to novice programmers seeking to develop software engineering skills, and powerful enough for serious computation. *Python* is used by computer programmers and scientists alike. Thus, it provides the tools used in software engineering as well as numerous libraries for scientific computing. In addition, *Python* is an open source project easily linked with other languages such as *Fortran* and *C*.

- ***Ubuntu*** is a Linux-based operating system. It is based on the principle of open-source development and users are encouraged to use free software, study how it works, improve

upon it, and distribute it. Using *Ubuntu* serves as a preparation for the use of high performance computing clusters, which mostly rely on Linux-based operating systems.

- ***Microsoft Azure*** is a cloud computing platform which provides the power and scalability required for collaboration, high performance computation, and data-intensive processing. Class participants receive access to the cloud using *Microsoft Azure Academic Passes* for six months.

- The ***structToolbox*** is a computer program for the simulation and estimation of a dynamic model of female labor supply decisions (Keane et al., 2011). It is designed as a teaching tool to show how, by acquiring software engineering skills, structural econometricians can more readily absorb research from computational science, improve the transparency of implementations, and ensure recomputability of results.

- The ***Quantitiative Economics*** website provides a series of lectures on quantitative economic modelling using *Python*. Topics include economic theory and empirics, mathematical and statistical concepts related to quantitative economics, algorithms and numerical methods for studying economic problems, and coding skills.

Prerequisites

This class requires a basic knowledge of the *Ubuntu* operating system and *Python* programming language. For novices, we offer a *Software Engineering Bootcamp* in the week before regular classes start to catch up on these basics. The *Python Scientific Lecture Notes* and the *Ubuntu Manual* provide our starting point. We also provide additional resources on the course website that allow for independent study.

Team

Philipp Eisenhauer	Contact:	eisenhauer@policy-lab.org
	Office Hours:	by appointment
Yike Wang	Contact:	ywang@policy-lab.org
	Office Hours:	by appointment

Time

Lectures	Mondays and Wednesdays, 1:30-2:50, Saieh Hall for Economics, #247
TA Sessions	Fridays, 2:30-3:20, Saieh Hall for Economics, #247
Colloquium	Thursdays, 5:00-7:00, Saieh Hall for Economics, #021

References

The basic ideas behind software engineering, independent of any particular tools, are presented in:

- Steve McConnell, *Code Complete: A Practical Handbook of Software Construction*, Microsoft Press, Seattle, WA.

Numerous additional references are available on the course website.

Schedule

- **Week 1**
  - Introduction
- **Week 2**
  - Cloud Computing

    - Exploit the resources and scalability of the cloud in your research.
  - **Special Event:** Computational Economics Colloquium

- **Week 3**
  - Version Control
    - Record changes to your computer program over time to recall specific versions later.

- **Week 4**
  - Debugging and Testing
    - Examine the correctness of your computer program and its components.
  - **Special Event:** Computational Economics Colloquium

- **Week 5**
  - Profiling and Code Optimization
    - Measure the performance of your computer program and identify the frequency and duration of function calls. Modify it for more rapid execution and less memory use.

- **Week 6**
  - Code Documentation
    - Document your computer program to inform fellow researchers about what it does and how it does it.

- **Week 7**
  - Design Patterns
    - Create reusable components of your computer program to use them in multiple research projects.
  - **Special Guest:** Social Sciences Computing Services

- **Week 8**
  - Data Management
    - Control the information you use and generate during your research project.
  - **Special Guest:** Research Computing Center
  - **Special Event:** Computational Economics Colloquium
- **Week 9**
  - High Performance Computing
    - Aggregate computing power to tackle computation-intensive projects.
- **Week 10**
  - Virtual Machines
    - Take advantage of virtual machines to increase the transparency of your programming choices and ensure full recomputability of your results.

Links

Python	`https://www.python.org`
Ubuntu	`http://www.ubuntu.com`
Microsoft Azure	`http://www.azure.microsoft.com`
structToolbox	`http://policy-lab.org/structToolbox`
Quantitative Economics	`http://quant-econ.net`
Python Scientific Lecture Notes	`https://scipy-lectures.github.io`
Ubuntu Manual	`http://ubuntu-manual.org`
Social Sciences Computing Services	`https://sscs.uchicago.edu`
Research Computing Center	`https://rcc.uchicago.edu`
Computation Institute	`https://www.ci.uchicago.edu`

Computational Economics Colloquium

`http://bfi.uchicago.edu/events/computational-economics-colloquium`

References

Keane, M. P., Todd, P. E., and Wolpin, K. I. (2011). The Structural Estimation of Behavioral Models: Discrete Choice Dynamic Programming Methods and Applications. In Ashenfelter, O. and Card, D., editors, *Handbook of Labor Economics*, volume 4A, pages 331–461. Elsevier Science.