Big data in economics (EC 410/510)

Spring 2022 syllabus

Grant R. McDermott
Dept. of Economics, University of Oregon

Summary

When: Mon & Wed, 10:00–11:20
Where: 140 TYKE
Web: https://github.com/orgs/uo-510-2022s (quarter)
https://github.com/uo-ec607 (lectures)

Who: Grant McDermott (instructor)
Assistant Professor of Economics
grantmcd@uoregon.edu
By appointment

Boyoon Chang (GE)
Doctoral student in economics
bchang@uoregon.edu
By appointment

Course description

Data are getting bigger. As data get big (i.e. they cannot fit in your computer's memory) the conventional empirical tools of the applied economist's toolbox often become inefficient or even ineffective. In this course, we will introduce, discuss, and implement some of the key tools that have been developed to overcome these challenges. We will also see how these "big data" tools be profitably repurposed for "medium data" settings too. We will start by laying the foundations for effective data science work, covering topics like version control and the shell (i.e. command line). From there, we will learn how to handle data and become efficient programmers in *R* (our primary computational environment). While our immediate focus will be on making the most of the local resources at our disposal, the skills that we master here will serve us well once we scale up to dedicated big data environments in the final section of the course. By the end of the quarter, you will have connected to cloud-based based services and high-performance computing clusters, queried petabyte-sized databases, and run distributed code across a network of computers. More importantly, you will have a better understanding of how computers work, what tools are at your disposal for tackling big data problems, and how to meaningfully integrate them into your everyday workflow.

Practical matters

Class rules

Important: We will be using a "flipped" classroom environment for this course. In other words, the bulk of the lecture content for the course will be delivered *asynchronously* via pre-recorded videos. My expectation is that you will watch and work through the videos on your own before we meet for class. Having you watch the videos beforehand yields several benefits in a course like this. It will save me (and you) from having to troubleshoot coding problems live during lectures, it will allow you to rewatch material that you didn't quite get upon first viewing, and it will generally free up time for discussion in an otherwise very coding-centric course. We'll reserve actual class time for two things: 1) troubleshooting and follow-up from the lecture videos, and 2) student presentations. Please note that you should attend the in-class sessions even though you will have watched the videos beforehand.

(Some lectures will still be "live" in the regular sense. Our first lecture will definitely be one of these. I'll let you know ahead of time if any others fall into this category too.)

Software requirements

All of the software requirements for this course are free and open-source. Please aim to have everything installed by the start of our first lecture. I will be available for installation troubleshooting during the first week of the quarter. If you want a detailed tutorial on how to achieve a perfect working setup, I can think of no finer guide than Jenny Bryan *et al.*'s `http://happygitwithr.com/` (see esp. chapters 4 – 11).

R and RStudio

We will mainly be using the statistical programming language ***R*** (download here).[1] Please make sure that you install the **RStudio IDE** too (download here).

Git and GitHub Classroom

We will also make extensive use of the **Git** version control system (follow the OS-specific installation instructions here). Once you have installed Git, please create an account on **GitHub** (here) and register for an education discount to get unlimited private repos (here).[2] Now is probably a good time to tell you that I am going to run the course through GitHub Classroom. You will receive an email invitation

[1] Mac users may consider installing R through Homebrew. That's fine, but please use `` `brew install --cask r` `` to do so (exactly as written here)... unless you're using one of the new M1-compatible machines. In the latter case, please speak to me about installation options first.

[2] GitHub offers unlimited free private repos for everyone. However, you are limited to three collaborators per private repo, so the education discount still makes sense.

to the course repo with instructions in due time, but suffice it to say that this is how we'll submit assignments, provide feedback, receive grades, etc.

Other

You are ready to start this course once you have installed R, RStudio, and Git (as well as created an account on GitHub). The last thing I want you to do for now is make sure that your system is configured to handle some additional

- **Linux:** You should be good to go.
- **Mac:** Install the Homebrew package manager. I also recommend some additional software, depending on whether you're using one of the new M1-compatible machines or not:
  - Non-M1 (older): Install the *R* toolchain for MacOS. This will automatically take care of several steps like installing Xcode, etc.
  - M1 (newer): While the new M1 chips offer incredible performance, unfortunately they require several finicky steps to play nicely with some of the other tools that we'll be using. Please install Xcode by opening your Terminal and running `` `xcode-select --install` ``. There are several more steps that you'll need to take, but we'll save that for class or office hours.[3]
- **Windows:** Install Rtools. While its not essential, I also recommend that you install the Chocolatey package manager for Windows.

I will provide instructions for any further software requirements as the need arises; i.e. when we get to the relevant lecture. On that note, the lectures have all been posted ahead of time on the course website. Each lecture lists all of the *R* packages and external libraries (if relevant) required for a particular class. I'll try to remind you, but my expectation is that you will look at these requirements and ensure that you have them installed *before* we start class.

Textbook and other readings

There's no set textbook for this course (Ed Rubin and I are working on one.). The lecture notes are very detailed and are thus "self-contained". However, I've drawn inspiration from various sources; a few of which are listed below. You don't *need* to buy or read any of these (excellent) books to complete the course. But I can eagerly recommend leafing through at least one or two of them. Each of these books is freely available online if you can't afford a hard copy:

- "***R* for Data Science**" (Garrett Grolemund and Hadley Wickham)[4]

[3] Here is a thorough discussion if you want to go through them by yourself.
[4] FWIW, Jake VanderPlas's "**Python Data Science Handbook**" is excellent option for anyone looking for a Python equivalent.

- "**Data Visualization: A practical introduction**" (Kieran Healy)
- "**Advanced *R***" (Hadley Wickham)
- "**Geocomputation with *R***" (Robin Lovelace, Jakub Nowosad and Jannes Muenchow)
- "**An Introduction to Statistical Learning**" (Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani)
- Etc.

Evaluation and grading

Grade determination

Grades will be determined as follows:

EC 410		EC 510	
1 × Short presentation	5%	1 × Short presentation	5%
1 × Long presentation	10%	1 × Long presentation	10%
1 × Intro HW assignment	10%	1 × Intro HW assignment	5%
3 × HW assignments (25% each)	75%	3 × HW assignments (20% each)	60%
		OSS contribution	20%

You will only be graded on the items that are listed above. No additional work or submissions will be considered. There is no final exam or project for this course. Any changes or specific requirements will be made clear as we proceed through the course. In the meantime, here are some additional details.

Homework assignments

With the exception of the first (short) introductory assignment, your homework assignments are to be completed in **teams of two**. As we'll see, these assignments will all be distributed and graded through GitHub Classroom. Links will be provided and I'll cover the details in class. Please note that late submissions will not be graded.

Presentations

Each person in the class will be expected to give two presentations. Both types are first-come, first-served.

- "Short". A 2–5 minute, show-and-tell talk on a "base" R function. I have a list of suggested options, but really this is just for you to explore and share some fun things about the language.

Base R has many hidden (underrated) gems that will make you a more productive and capable programmer in the long-run, even if you stick with user-written packages most of the time.

- "Long". A 10 minute talk accompanied by slides (which you'll share with the rest of the class). You'll choose from a list of prescribed presentation topics—corresponding to the lecture topic at hand—that I shall provide in due course.

In general, your presentations will be graded on how clearly you communicate the topic at hand. The primary question that I'll be asking myself when grading is: "Could other students easily understand the concept(s) based on your talk?"

OSS contribution (EC 510 only)

You are going to contribute to open-source software (OSS) in some way, shape, or form. This could be by identifying and correcting bugs in a package that you use. Or, it could be by contributing material (e.g. documentation) to an open-source project. I particularly want to encourage you to contribute to the Library of Statistical Techniques (https://lost-stats.github.io/). There's clearly quite a bit of leeway here and I'll need to sign off on whatever you propose. Similarly, depending on the scope and size, you may need to make several different contributions to fulfill the requirement.

Honesty and academic integrity

Students caught cheating or plagiarizing will automatically be assigned a zero grade. Please acquaint yourself with the Student Conduct Code at http://studentlife.uoregon.edu. More important for this class, individual team member will be expected to pull their weight for your paired assignments.[5]

Accessibility

If you have a documented disability and anticipate needing accommodations in this course, please make arrangements with me during the first week of the term. Please also request that the Accessible Education Center send me a letter verifying your disability. Students with infants or young children that need ongoing care should similarly come and see to me. We'll have to take it on a case-by-case basis, but I'll do my utmost to accommodate you.

[5] You'll see that we can actually track individual member contributions, although I really hope it doesn't come to that.

Tentative lecture outline

Note: We only have 80 minutes allocated for each lecture. I expect that several individual topics will run over two or more lecture slots. Please bear that in mind as you look over this tentative outline.

Foundations

Expected no. of lectures slots: 5 (6)

- Introduction: Motivation, software installation, and data visualization
- Version control with Git(Hub)
- Learning to love the shell
- *R* language basics (*optional*)

Data wrangling, I/O, and acquisition

Expected no. of lectures slots: 5

- Data cleaning and wrangling: (1) Tidyverse
- Data cleaning and wrangling: (2) data.table
- Big data I/O
- Webscraping: (1) Server-side and CSS
- Webscraping: (2) Client-side and APIs

Programming

Expected no. of lectures slots: 4

- Functions in *R*: (1) Introductory concepts
- Functions in *R*: (2) Advanced concepts
- Parallel programming

Cloud resources and distributed computation

Expected no. of lectures slots: 6

- Docker
- Cloud computation (Google Compute Engine)
- High performance computing (UO Talapas cluster)
- Databases
- Spark

Other potential topics (time permitting)

- Regression tools for big data problems
- Spatial analysis
- Networks
- Deep learning
- Automation and workflow
- Rcpp (i.e. integrating C++ with R)