

ECOG 315 / ECON 181, Summer 2025
Advanced Research Methods and Statistical Programming
Week 1 Lecture Slides

Matthew N. White

Howard University

May 30, 2025

Who Are These Guys? Matt White

- ▶ Instructor of record: Matthew N. White (Matt)
- ▶ Email: mnwhite@gmail.com; Cell: (603) 566 0413
- ▶ I will be here **most** weeks; others will be here **sometimes**
- ▶ Previously at University of Delaware, now with Econ-ARK
- ▶ Primary interests: health economics, heterogeneous agents macro

Who Are These Guys? Matt White



Who Are These Guys? Chris Carroll

- ▶ Most senior instructor: Christopher D. Carroll (Chris)
- ▶ Email: ccarroll1@jhu.edu
- ▶ Professor at Johns Hopkins economics department
- ▶ World expert in consumption-saving theory and empirics
- ▶ Has advised many PhDs, taught research skills course

Who Are These Guys? Chris Carroll



Who Are These Guys? Alan Lujan

- ▶ Content-producing instructor: Alan E. Lujan-Solis (Alan)
- ▶ Email: alujan@jhu.edu
- ▶ Program coordinator and lecturer at JHU Advanced Academic Programs
- ▶ Primary interests: computational methods for economics
- ▶ Will make and share **asynchronous** Zoom videos

Who Are These Guys? Alan Lujan



Who Are These Guys? Econ-ARK

- ▶ All of your instructors work for or with Econ-ARK
- ▶ Non-profit org that makes open source software for economists
- ▶ And tools/structures for archiving research projects
- ▶ Website: <http://www.econ-ark.org>
- ▶ We didn't expect to be teaching the course, please bear with us!



Course Communication Channels

- ▶ We don't have access to Canvas or class list (yet)
- ▶ GitHub repository: <https://github.com/econ-ark/aeasp.2025>
- ▶ The repo is “live” and edited frequently; you will submit via pull requests
- ▶ If you have questions, please email us, especially Matt
- ▶ If it's something of concern to others, open an issue on GitHub
- ▶ Each team will meet weekly with their advisor, scheduled independently
- ▶ Matt and Alan use Discord; do you want to use Discord for the course?

Big Picture: How to Be an Economist

- ▶ You will learn a **lot** in an economics PhD program
- ▶ Lots of economics, lots of math, lots of econometrics
- ▶ How to **do** research: asking question, lit review, forming strategy, iterating
- ▶ How to **communicate**: academic writing, presentations/slides, selling yourself
- ▶ There are tools to help you with those things; often not explicitly taught
- ▶ Being familiar with the tools of the trade is part of being an academic / economist

What Are We Doing Here? Course Overview

- ▶ We are going to help you learn to be an economist
- ▶ In lectures: teaching you about tools used in research and collaboration
- ▶ In lectures: advice for executing research steps and applying those tools
- ▶ In lectures: presenting your research progress and getting feedback
- ▶ Asynchronous Zoom: specific applied econometrics and computational methods
- ▶ Adviser meetings: ongoing personal discussions of your research progress
- ▶ Grading: qualitative hand-waving

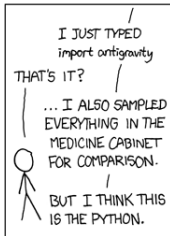
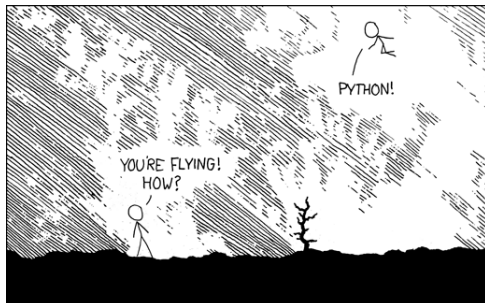
The Platinum Rule

- ▶ Google is your friend; the internet is full of lies, but use it anyway
- ▶ Someone else almost surely had the same Q you did, and got an answer
- ▶ We will also show you how to **responsibly** use AI in your work
- ▶ WRONG: “Please write me a 1000 word essay about X, and provide citations.”
- ▶ RIGHT: “I think these two paragraphs are too wordy and repetitive. Please help me shorten and clarify them.”

Agenda for Week 1

- ▶ ~~Personal and course introductions~~
- ▶ Installing Anaconda distribution of Python
- ▶ Setting up GitHub account and GitHub Desktop
- ▶ Interacting with course repo via GitHub
- ▶ Basics of conda environments
- ▶ Basics of Python / IDEs / Jupyter notebooks

Why Python?



Why Python?

- ▶ High level language: closer to human language than machine code
- ▶ Well documented and supported: **many** packages available via PyPI
- ▶ Widely used in scientific computing
- ▶ Completely free and open source
- ▶ Easy to program and run native Python code; human time is valuable
- ▶ Easy to extend and link to other languages: acceleration via `jit`
- ▶ Not the best at anything, but good enough at everything

Anaconda Distribution

- ▶ Python comes in **distributions**: specific implementations with packages
- ▶ Anaconda: widely used distribution for scientific computing
- ▶ Automatically includes all of the most commonly used packages for scientific work
- ▶ Has its own virtual environment and package manager, `conda`
- ▶ Also comes with a pretty nice interactive development environment (IDE), Spyder
- ▶ Spyder is set up for IPython: better graphical presentation

Installing Anaconda

- ▶ Go to <https://www.anaconda.com/download>
- ▶ Submit your email or click “skip registration”
- ▶ Select the distribution installer for your OS
- ▶ You probably want the graphical installer if it's an option
- ▶ Download it, then install Anaconda; this will take a bit

Collaboration and Version Control with `git`

- ▶ You have done group work; probably shared docs by email or Dropbox
- ▶ Academic research is a really big group project: so many files
- ▶ Multiple people working on same file at the same time: oh no!
- ▶ Realize in November that you need some file how it looked back in May
- ▶ Academic / programmer solution: version control via `git`
- ▶ Track **entire** history of file changes, easy to revert to prior version
- ▶ Make “branches” from main, can merge separate work paths via pull requests
- ▶ Work on files locally, but long term archived on a remote server

Making git Easier: GitHub

- ▶ `git` was created by Linus Torvalds, the Linux guy; it's a command line tool
- ▶ GitHub is a website and desktop app that provides an easier interface for `git`
- ▶ You can use command line `git`, but I will show you GitHub Desktop
- ▶ Go to <https://github.com>, enter your email address and sign up
- ▶ Everything you need for this course is on free tier, but you can pay for more
- ▶ Next: Go to <https://github.com/apps/desktop> and install GitHub Desktop
- ▶ Then open it and sign in to your GitHub account; only have to do this once

What the Fork is a Repository?

- ▶ A collection of files in `git` is called a repository (repo)
 - ▶ All of your work for one academic project would be a repository
 - ▶ The (non-Canvas) course website is a GitHub repository
 - ▶ The development of a new Python package would happen in a repo
- ▶ GitHub repos can be **public** and available to everyone who finds it
- ▶ Or can be **private** and only accessible to invited collaborators
- ▶ A repo can be **forked**, making a new personal copy– it can be private!
- ▶ You can issue a **pull request** (PR) to send changes from your fork back to the “upstream” repo

Forking the Course Repository

- ▶ Each of you will make a personal fork of the course repository
- ▶ You will submit assignments (etc) by issuing a PR back to the upstream repo
- ▶ Go to course website: <https://github.com/econ-ark/aeasp.2025>
- ▶ Click the menu arrow near “Fork” in top right, select “Create a new fork”
- ▶ Default options should be fine; click green “Create” button



Cloning Your Fork to Local Machine

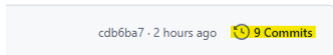
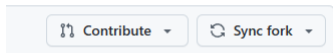
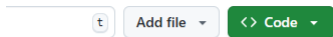
- ▶ Repos (and forks) actually live on the remote GitHub server
- ▶ You must **clone** them to your computer to work with files
- ▶ Creates a local working copy of the repo (or your fork)
- ▶ Open GitHub Desktop and:
 1. Click “Repository” button in top left
 2. Then “Add” and “Clone repository...”
 3. Select YourHandle/aeasp.2025, probably only option
 4. Choose a local directory to put it in
- ▶ All of the course repo files are now on your computer!

Where's the Remote? Making Commits

- ▶ Changes that you make to your local clone don't automatically go to remote
- ▶ Need to **commit** them: “put a pin” in changes, with (label and comments)
- ▶ Easiest to show by example: open up `/materials/setup/Survey.txt`
- ▶ Read the instructions, then fill out the short survey below
- ▶ Save file as instructed in step (3) of the survey document; my survey is there
- ▶ In GitHub Desktop, click on “Changes”; name your commit, comments optional
- ▶ Click “commit to main”, then click “push to remote” at the top
- ▶ Pushing to remote is what actually shares your local work with the server

Where's the Remote? Making Pull Requests

- ▶ Navigate on `github.com` to your fork of the course repo
- ▶ Click on the “1 commits” button; the commit you just made should be there!
- ▶ But I want your survey in the upstream course repo; need to make a **pull request**
- ▶ PR: Request to merge “downstream” work into “upstream” project
- ▶ At top left, click “Pull requests”, then “New pull request”
- ▶ Ensure you're going from `YourHandle/aeasp.2025` to `Econ-ARK/aeasp.2025`
- ▶ Give a title and descriptive comments to your PR, then create/submit it
- ▶ Then we can see your PR(s) on the course repo!



Anaconda and Virtual Environments

- ▶ There are a lot of Python packages, and you will have multiple projects
- ▶ Project A requires CoolPackage v0.8 or higher; Project B requires v0.7 or lower
- ▶ Not a problem! conda can manage virtual Python environments
- ▶ Can have multiple collections/configurations of packages on your computer
- ▶ Manage each separately, switch between them with a single command
- ▶ We probably won't use this much, but will install more packages
- ▶ Open up Anaconda Prompt (from Windows start menu, e.g.); terminal will open

Installing New Packages

- ▶ Example: installing HARK, Econ-ARK's primary software package
- ▶ In Anaconda prompt, type `pip install econ-ark`, accept
- ▶ `pip` is the most common **package manager**, draws on Python Package Index
- ▶ `conda` is also a package manager, but **slightly** fussier
- ▶ HARK is now available for use on your computer
- ▶ But we won't be working with it yet

Actually Working with Python: Spyder

- ▶ You **can** write and edit Python code in any text editor
- ▶ And then run Python code from a command line with `python`
- ▶ Using an **interactive development environment** (IDE) just make things nicer
- ▶ Open up Spyder (e.g. from Windows start menu)
- ▶ There are a lot of panes; I close all of them except console and editor
- ▶ Can open them later from View menu, Panes option
- ▶ Console pane: live Python environment, can run code commands one by one
- ▶ Editor pane: file editor; write scripts / programs, run with green arrow in toolbar

How to Learn Python

- ▶ We will give you hands-on instruction with Python, don't worry
- ▶ If you want to learn on your own, I recommend Kevin Sheppard's notes
- ▶ There's a copy in `/materials/setup/KevinSheppardPython.pdf`
- ▶ It's from four years ago; current Python versions are 3.10 to 3.13
- ▶ Recommended reading order: Ch 1, 2, 4, 9, 10, 12, 18, 3, 6, 7, 5, 11, 15, 29
- ▶ Data operations: Ch 8, 16, 14, 17, 23
- ▶ More math stuff: Ch 19, 20, 21, 22
- ▶ General coding: Ch 13, 24, 25, 26, 28

Introduction to Jupyter Notebooks

- ▶ Communicating scientific ideas is sometimes aided by reader interaction
- ▶ **Jupyter notebooks** provide a way to integrate text, math, and code
- ▶ Go to <https://github.com/econ-ark/HARK> and fork our repo
- ▶ Then clone it to your personal computer
- ▶ In Anaconda prompt, type `jupyter notebook`
- ▶ Navigate to your HARK clone directory, then go to `/examples/Gentle-Intro/`
- ▶ Open `Gentle-Intro-To-HARK.ipynb`
- ▶ Cells can be run individually with Shift+Enter