PPOL564: Foundations of Data Science[1]

Mondays & Wednesdays (11:00am to 12:15pm)
Location: Reiss 559

Instructor: Eric Dunford
Office: Old North 404
Office Hours: Wednesdays 2pm to 4pm or by appointment (https://ericdunford.youcanbook.me)
Email: eric.dunford@georgetown.edu

COURSE DESCRIPTION

This critical first semester course aims to prepare all incoming students for the rigor of their upcoming coursework in the rest of the program. The course will accelerate the students to a level of mathematical and computer science competency that will enable them to excel in their ensuing data science courses. On the mathematical side, students will cover linear algebra with a special focus on matrix algebra (e.g. matrix notation, projections, determinants, inversions, and eigenvectors) and multivariate calculus up until constrained optimization. On the computer science side, students will be introduced to working at the command line and python programming, with introductions to data structures, data manipulation and programming paradigms. Students will work in notebooks and use Git/GitHub to submit coding assignments, developing literate programming and reproducible research skills they will use throughout the program.

The course is divided into three sections. Section 1 will focus on building competency with programming in python. Emphasis will be placed on producing readable, reproducible, and well-documented code. We will develop competency using Git/Github for version control. The skills acquired in Section 1 will be applied throughout the course, both in the form of coding "discussions" and applied assignment for the later mathematical discussions. Section 2 will delve into linear algebra. Students will gain a vital intuition of many of the mathematical properties that inform statistical estimation and data decompositions. Finally, Section 3 will cover uni- and multivariate calculus with an emphasis on optimization.

COURSE OBJECTIVES

The course aims to provide students with the following competencies:

- General understanding of python's programming syntax and data structures.
- Understanding of key mathematical concepts in linear algebra and multivariate calculus.
- Applied experience programming and debugging in python using Jupyter Notebooks.
- Familiarity with version control using Git from the command line.

[1] I reserve the right to change and adjust this syllabus at any time, but will given ample notice of any changes. All changes can be marked by the version index.

READINGS

We will rely primarily on two texts throughout the duration of the course:

- A Mathematics Course for Political and Social Research. Moore, Will H., and David A. Siegel. Princeton University Press, 2013.
- Learning Python. Lutz, Mark. O'Reilly Media, 5th Edition, 2013.

Both texts will serve as important reference materials. I will assume that students will have read the materials assigned for each week, especially when we delve into mathematical parts of the course. The Lutz text will serve as a useful reference as students apply their knowledge of python to the coding discussions each week and checkpoints. Any additional readings will be made available on the class Canvas site.

GRADING

This course will be graded using a traditional scale (A+ (97 – 100), A (93 – 96), A- (90 – 92), B+ (87-89), B (83 – 86), B- (80 – 82), etc.). It is expected that you will complete all of the assignments and attend class regularly.

- **Weekly Coding Discussions** (9) 20%
- **Checkpoint Assignments** (4) 40%
- **Final** (1) 40%

COURSE WORK

Weekly Coding Discussions

Every **Wednesday** there will be a coding problem/prompt pushed to the **class Github repository** by the end of the day (5pm). Students will be required to submit an **original response to the prompt by Friday evening 11:59pm**. Each student must then respond to another student's coding solution in the form of a substantive contribution to the existing code or by debugging an issue. **All coding responses to another student's push must be pushed by Sunday 11:59pm**. All edits should be pushed as branches, which the original author can then merge the elements of with their solution.

Submission and response will be assigned two points each week (one for submitting and one for commenting). It is vital that all students submit their answers in a timely fashion in order for others to respond and for us to discuss outcomes on Monday in class. Thus, any late submissions will not receive points.

There will be a total of 9 weekly coding discussions throughout the course of the semester. Weeks where a coding assignment is posted will be marked with a double asterisk (**). There will be no discussions for the weeks when checkpoint assignments are due.

All coded responses must be thoroughly commented: that is, each line of code must have a comment briefly explaining the intent of that line (i.e. why are you doing what you are doing?). **Failure to fully comment coded responses will result in half credit.** In addition, all students must be respectful when commenting on, making contributions, and debugging others code. Failure to be respectful will result in a loss of credit for that discussion.

The goal of the coding discussions is to apply a concept learned during the week in a way that helps build a greater level of programming fluency. Programming skills are honed through active usage and repetition. Learning to read other people's code and detecting issues is vital to successful collaborations in applied work. These discussions will be graded according to participation. The point is not to be "right" necessarily but rather to try, learn, and collaborate. As such, when receiving suggested edit requests

Checkpoint Assignments

There will be four assignments throughout the course of the semester. These assignments will take the form of questions that the student must independently complete. The goal of the assignment is to test the student's comprehension of the materials covered in each section. **All assignments will be posted Monday afternoon by 5pm on the class canvas site for the weeks marked on the syllabus**.

The assignments will be in the form of a Jupyter Notebook (.ipynb). These notebooks will be specially formatted for the graded assignments. **The completed assignment notebook must be submitted to Canvas by Friday 11:59pm for the weeks where a checkpoint was assigned**.

All assignment submissions must adhere to the following guidelines:

- All mathematical formulas should be written in LaTex.
- All code should be commented.
    - For each line of code, write a brief comment describing what the line intends to do.
    - For each functions, write a brief doc string outlining what the function does.
    - For scripts, offer a header comment outlining what the script or notebook intends to accomplish.
- All code must execute fully.

Final

We will have one final for this course at the end of the semester. The final seeks to test each student's understanding and comprehension of the main topics of the course. Students will be given two hours to complete the final. Details on the final will be announced in class as we approach the end of the term.

IMPORTANT DATES

- **9/3: No Class (Labor Day)**
- **9/21: Assignment 1 Due (by 11:59pm)**
- **10/8: No Class (Mid-Semester Holiday)**
- **10/12: Assignment 2 Due (by 11:59pm)**
- **11/9: Assignment 3 Due (by 11:59pm)**
- **12/7: Assignment 4 Due (by 11:59pm)**

- **12/18: Final (9:00am – 11:00am)**

POLICIES

Attendance/Participation: As a Master's level course, attendance will not be taken; however, participation in the form of questions and the weekly coding discussions are assumed. Frequent absences may result in failure of the class as student may fall behind. Each student is responsible to make up the materials missed during a lecture on their own. All lecture notes will be posted to canvas; however, note that these lecture slides are intentionally sparse. Thus, students who missed a lecture should reach out to their peers in the class for lecture notes.

Assignments: Late assignments will be penalized a third of a letter grade each day. Assignments later than a week will not receive credit. Assignments must be coherently written and all submitted assignments should be completely reproducible. Put simply, (1) answers should make sense and be concisely written and (2) all submitted code should run. Any assignments (or answers on assignments) that fail to meet these criteria will get no or partial credit. See the "Check Point Assignments" section above for details regarding submission.

Academic Integrity: Academic integrity is central to the learning and teaching process. Students are expected to conduct themselves in a manner that will contribute to the maintenance of academic integrity by making all reasonable efforts to prevent the occurrence of academic dishonesty. Academic dishonesty includes (but is not limited to) obtaining or giving aid on an examination, having unauthorized prior knowledge of an examination, doing work for another student, and plagiarism of all types, including copying code.

Plagiarism: Plagiarism is the intentional or unintentional presentation of another person's idea or product as one's own. Plagiarism includes, but is not limited to the following: copying verbatim all or part of another's written work; using phrases, charts, figures, illustrations, code, or mathematical / scientific solutions without citing the source; paraphrasing ideas, conclusions, or research without citing the source; and using all or part of a literary plot, poem, film, musical score, or other artistic product without attributing the work to its creator. Students can avoid unintentional plagiarism by following carefully accepted scholarly practices.

Disability Services: If you believe you have a disability, then you should contact the Academic Resource Center (arc@georgetown.edu) for further information. The Center is located in the Leavey Center, Suite 335 (202.687.8354). The Academic Resource Center is the campus office responsible for reviewing documentation provided by students with disabilities and for determining reasonable accommodations in accordance with the Americans with Disabilities Act (ADA) and University policies. Please contact me during the first week about necessary accommodations.

Religious Holidays: Georgetown is an inclusive community that recognizes all religions. If you anticipate that you will have a conflict due to a religious observance, please contact me early in the semester. Go to https://campusministry.georgetown.edu/religious_holy_days for a list of major holidays and to read the Provost's policy on accommodating religious observances.

General:

- **Proof of Diligent Debugging** (policy on asking technical questions)

  - When reaching out to me regarding a technical question, error, or issue you must demonstrate that you made a good faith effort to debugging/isolate your problem prior to reaching out. In as concise a way as possible, send me the record of what you tried to do. Ultimately, I am your resource of last resort. As software is continually being refined in data science and new approaches continually emerge and changing, learning how to frame your question and find a similar solution online is a key tool for success in this domain. If you make a diligent effort beforehand to solve your problem, I'll do the same in trying to help you figure out a solution.
- **Email Policy**
  - I will respond to all emails within 24 hours of being sent during a weekday. I will not be responded to emails sent late Friday (after 5pm) or during the weekend until Monday. Please plan accordingly if you have questions regarding current or upcoming assignments.

INSTALLATIONS

All students should plan on bringing a personal computer to class each session.

We will employ the following software throughout the semester in some capacity. This list outlines the software versions that *I will use* in class.[1] This is merely a list of suggested installations for those looking to follow along verbatim with the software I use. If/when you find an arrangement that works better for you on your machine, feel free to use that setup. Note, however, that all assignments must be submitted using Jupyter Notebooks (.ipynb).

NOTE: **Python 3 is required for this class. Students may not use Python 2 or R.**

- **Command Line (zsh via "oh-my-zsh")**
  - Note the Ubuntu shell (activated on windows Settings > Update & Security > For developers) and the Terminal use bash, which will work fine for everything we do. Zsh is a useful facelift for the terminal to make it easier to use.
  - Installation instructions:
    - Mac & Linux: https://github.com/robbyrussell/oh-my-zsh/wiki/Installing-ZSH
    - Windows: https://www.maketecheasier.com/install-zsh-and-oh-my-zsh-windows10/
      - Note: for Windows 10, the command line can be turned on by activating the developer mode: Settings > Update & Security > For developers
- **Package Managers**:
  - Homebrew (Mac): https://brew.sh
  - Chocolatey (Windows): https://chocolatey.org
- **Git**
  - Installation instructions: https://git-scm.com/book/en/v2/Getting-Started-Installing-Git
    - TL;DR: assuming one has a functioning shell, run: sudo apt-get install git
  - Git is already downloaded on most current versions of MacOS. However, the standard is python 2.7.x. This version of python will not be used in this class.

  - Git can also be downloaded with one's respective package manager, e.g. brew install git
- **Python 3.6.5 (>=) & Jupyter Notebooks**
  - Installation: https://www.anaconda.com/download/#macos
- **Pip**
  - pip is a method to download python modules. An alternative approach is to use the conda command, which will come with the anaconda distribution. Conda does not always play well with zsh. Thus, you'll see me use pip.
  - To download pip: https://pip.pypa.io/en/stable/installing/
- **Atom (+ Hydrogen Package)**
  - https://atom.io/
  - Atom is a hackable text editor created by Github. The editor offers a dazzling array of packages (some of them more useful than other) developed by the open-source community. The Hydrogen package allows users to initiate a Jupyter kernel in a script, which allows for interactive coding in Python. Interacting with the code is a valuable way to learn, thus I find packages like Hydrogen to be useful when both learning and developing.

Note: I primarily work off a Mac, and fully acknowledge that there are many nuances of other operating systems that may cause issues with some of the instructions outlined in this syllabus. That's okay. Try to find your respective platform's solution to the concept at hand. If you cannot resolve the issue, keep with the 'Proof of Diligent Debugging' protocol. If you do resolve the issue, please let me know what it was and how you resolved it. This will help minimize these sorts of occurrences moving forward.

COURSE SCHEDULE

**** = 'coding discussion assigned'**

COMPUTATION

Week 1: Introductions

- Review syllabus/establish expectations for the course.
- Note: getting up and running & schedule one-on-ones

Week 2: Command line, Version Control, and Documentation **

- No Class (9/3) – Labor Day
- Readings:
  - Download: Chacon, Scott and Ben Straub. (2014). 'Pro Git'. Ed. 2: https://git-scm.com/book/en/v2
    - Read Ch. 1 – 3, 6
  - Review: https://guides.github.com/

Week 3: Cont. Command line/Version Control + Introduction to Data Types in Python**

- Readings:
  - Review Chacon and Straub readings.

- Lutz Ch. 1 – Ch. 7

Week 4: Manipulating Data Types and Control Sequences in Python

- Readings:
  - Lutz Ch. 8 – Ch. 10, Ch. 13
- **Assignment 1 (Due Friday, Sept. 21)**

Week 5: Iterators, Comprehensions, Scope and Functions **

- Readings:
  - Lutz Ch. 14 – Ch. 18
  - Lutz Ch. 19 (only section on "Anonymous Functions: Lambda")

Week 6: Classes and Modules

- No Class (10/8) (Mid-Semester Holiday)
- Reading:
  - Lutz. Ch. 21 – Ch.26
  - Readings TBA: See Canvas.
- **Assignment 2 (Due Friday, Oct. 12)**

LINEAR ALGEBRA

Week 7: The Geometry of Vectors **

- Brief Introduction to writing mathematical notation using LaTex
- Readings:
  - Moore and Siegel Ch. 1, 2, 12.1, 13.1

Week 8: Transformations and Matrices **

- Readings:
  - Moore and Siegel Ch. 12.3 – Ch. 12.6, Ch. 13

Week 9: Decompositions **

- Readings:
  - Moore and Siegel Ch. 14.1

Week 10: Data Science Applications with Matrices

- Readings:
  - Readings TBA: See Canvas.
- **Assignment 3 (Due Friday, Nov. 9)**

CALCULUS

Week 11: Functions, Limits, and Derivatives **

- Readings:
  - Moore and Siegel Ch. 3 – Ch. 6

Week 12: Univariate Optimization and Partial Derivatives **

- Readings:
    - Moore and Siegel Ch. 8, 15.1-15.2.2

Week 13: Multivariate Optimization and Constrained Optimization **

- Readings:
    - Moore and Siegel Ch.15.3 - Ch. 17

Week 14: Computational Approaches to Optimization

- Readings:
    - Readings TBA: See Canvas.
- **Assignment 4 (Due Friday, Dec. 7)**

Final: Tuesday, December 18, 9:00am – 11:00am