Scientific Data Analysis, Data Science, and Machine Learning in Python

An independent study with two parts:

Part 1: Consolidate and advance our understanding of mainstream and cutting edge scientific data analysis techniques using Chapters 1-8 of Pasha, Astronomical Python

Part 2: Survey data science techniques up to and including neural net and deep learning implementations (which are the relevant preparation for a subsequent study of natural-language processing and LLMs) using Chapters 1-11, 13-15, and 18-19 Grus, Data Science from Scratch, 2nd Ed.

Term 6 of Academic Year 2024-2025, Deep Springs College

Mentor: **Prof. Brian Hill**

Student: Hexi Jin (DS 23)

Materials

Required

- Imad Pasha, *Astronomical Python*
  - Pasha's examples use data hosted at **https://zenodo.org/records/10732223**
- Joel Grus, *Data Science from Scratch, 2nd Edition*
  - We may need to copy over resources from Grus's GitHub repo for the book **https://github.com/joelgrus/data-science-from-scratch**

Optional

- Both Pasha and Grus include adequate introductions to Python features as they use them, but you may want a more systematic introduction to use as a reference. An excellent one is David Beazley, *Python Distilled*. It is actually a distillation and update of his time-tested *Python: Essential Reference*, which was growing overly-long as the Python language feature set kept growing.
- Since we will be using Git to keep and share all of our code and notes, and version control with this level of sophistication is *de rigueur* for working in a software team, consider supplementing the understanding that you get from the workflows we are using by reading Travis Swicegood's *Pragmatic Guide to Git*.

Actual Daily Schedules (Kept Retrospectively)

- **Daily Schedule - Part 1**
- **Daily Schedule - Part 2**

Looking Beyond

- **Looking Beyond**

Notes (mostly code samples)

- **Brian's Notes**
- **Hexi's Notes**

Daily Schedule Part 1 (Actual — Kept Retrospectively)

Regular meeting schedule is Wednesdays and Saturdays, 11:00-12:00

Back to **Course home page**

Part 1: Scientific Python (using Imad Pasha, *Astronomical Python*)

Part 1 Uses Pasha and lasts for the first three weeks of Term 6

Week 1 — Shell and Python Quick-Start/Review

- May 16 — Complete Chapters 1 to 3: Unix (shell) Basics, Installing Python, and the Astronomy/Scientific Data Analysis Stack — Problem Set 0: Get Anaconda downloaded and installed and use the IPython interface — Discussed Python language features, syntax, and style (PEP 8), differences between Windows and Unix shells, globbing, and Python's Operating System Insulation Layer (OSIL)
- May 17 — Complete Chapter 4: Introduction to Python — Problem Set 1: Use for loops to compute the first 20 Fibonacci numbers (screenshot your solution in IPython) — Discussed notebook tools and IDEs

Week 2 Matplotlib and Numpy

- May 21 — Complete Chapter 5: Visualization with Matplotlib — Install (if not already part of your Python distribution) and start working in Jupyter Lab — Problem Set 2: Make some histogram and scatter plots using the **iris dataset** (save your plots as a Jupyter Lab notebook)
- May 25 — Complete Chapter 6: Numerical Computing with NumPy — Create a github account, fork the repo: brianhill/scientific-data-analysis — Then figure out how to get a local copy onto your machine of your fork (`hexijin/scientific-data-analysis`) and this will involve installing git on your machine (which will be different for Mac or Windows) — Started learning shell access to git, and the add, commit, push cycle (which we will be adding more to once that is routine)

Week 3 — SciPy and AstroPy

- May 28 — Complete Chapter 7: Scientific Computing with SciPy — Problem Set 3 (in addition to working through all the code in the chapter): Do Exercise 7.1 — Introduced the linear algebra concepts and notation for column vectors, row vectors, and matrix and vector multiplication
- June 1 — Complete Chapter 8: Astropy and Astronomical Packages — As Problem Set 4 (in addition to working through all the code in the chapter): Do the Chapter 8 exercises — Finally, it's time to add to your git knowledge the ideas of origin and upstream, and a second cyle of operations: how to fetch from upstream (my GitHub repo), rebase (in your local repo), and push your rebased changes to your origin (your GitHub fork of my repo)

See also **Daily Schedule - Part 2**

Daily Schedule Part 2 (Actual — Kept Retrospectively)

Back to **Course home page**

See also **Daily Schedule - Part 1**

Part 2: Data Science Foundations (using Joel Grus, *Data Science from Scratch, 2nd Edition*)

Part 2 Uses Grus and lasts for the remaining four weeks of Term 6

Week 4 — Yet Another Review of Python — Some Vector and Matrix Algebra — Statistics and Probability

- June 4 — Chapters 1-3: Mostly redundant but excellent review of Python and Matplotlib — Review the three chapters, but completely stop using Jupyter or Jupyter lab, and instead get everything working in PyCharm Professional Edition (free for students) — When Grus says that you should not be tampering with your base Python environment, he is completely correct — So learn how to make a venv that you could call grus or dsfs and then switch to it
- June 7 — Chapters 4-6: Linear Algebra, Statistics, and Probability (due to having taken last fall's Bayesian Statistics class, much of the math in Chapters 5 and 6 will be review)

Week 5 — Optimization (aka Minimization and Maximization) — Working with Data

- June 11 — Chapters 7 and 8: Hypotheses & Inference and Gradient Descent — Make a local repo from the magic hexijin.github.io GitHub repo, put an index.md file in it, and then push to origin main — The only remaining step to having **your own home page** is to enable GitHub pages in this repo — For more advanced reading, Grus recommends this **Overview of Gradient Descent** by Eric Ruder
- June 15 — Chapters 9 and 10: Getting and Working with Data (including subtracting the mean and dividing by the standard deviation to get rescaled data sets, and a load of utilities for doing principal component analysis, that Grus somewhat-too-rapidly introduced at the end of Chapter 10)

Week 6 — Machine Learning — Linear Regression

- June 19 — Chapters 11 and 13: Machine Learning and Naive Bayes (and you may need to pick up some material from Chapter 12 on k-Nearest Neighbors which we are otherwise skipping)
- June 21 — Chapters 14 and 15: Simple Linear Regression and Multiple Regression — In Chapter 15, Grus squeezed in a digression on **The Bootstrap** which is a computational approach not just to estimating parameters, but to estimating uncertainties in those parameters

In the interest of getting to Neural Networks and Deep Learning in our final week, we are skipping Chapter 12 (on k-Nearest Neighbors), Chapter 16 (on Logistic Regression), and Chapter 17 (on Decision Trees)

Week 7 — Neural Networks — Deep Learning

- June 23 — Chapter 18: Neural Networks
- June 25 (no meeting, but do the live coding session) — Get a feeling for how a real pro codes, including type-hinting, systematic adherence to style choices, and code testing, by building the code in PyCharm as **Grus builds a deep learning libary** in VS Code, pausing the live coding session whenever you need to catch up with him, and fixing the style errors that PyCharm's linter catches and Grus's runs of mypy miss — Grus's live coding session is effectively a blindingly-fast introduction to the same material as is in Chapters 18 and 19
- June 26 (final meeting) — Chapter 19: Deep Learning — Only up to and including the section titled "Softmaxes and Cross-Entropy"

See also **Looking Beyond**

Supplementary Material and Looking Beyond

Back to **Course home page**

See also **Daily Schedule - Part 2**

Supplementary Material

Since Python's classes are something we only did much with toward the end, and they are often considered a fundamental part of an introduction to Python, you could do a quick introduction to classes (aka "object-oriented programming") such as Section 1.16 of *Python Distilled* by Beazley. The three things that are generally considered important about object-oriented programming are encapsulation (controlled access to the object's data), inheritance (behavior is inherited from superclasses in the class heirarchy and extended by subclasses), and polymorphism (behavior can be overriden in subclasses). Chapters 4 and 7 of Beazley are a much more complete introduction to object-oriented programming.

Looking Beyond Our Endpoint (Chapter 19 on Deep Learning)

Although the chapter we finished with did not attempt to cover the 2017 "**Attention is All You Need**" paper, Grus has almost perfectly set you up for a junior-level course in natural-language processing and LLMs that almost every computer science department is now or shortly going to have on offer. Before or instead of taking such a course, you can review or go beyond where we have gotten as of Chapter 19 of Grus in multiple ways:

(1) A concise and mathematically-sophisticated review of all that we have done (and then some) is "**A high-bias, low-variance introduction to Machine Learning for physicists**."

(2) Grus considers how LLMs have changed and will continue to change the workflow of a data scientist starting at the 15:00 mark in this late-2023 video "**Doing Data Science in the Time of ChatGPT**." This is a casual survey that may only serve to cement what you have already discovered you can do with a current-generation LLM like Grok 3 or ChatGPT 4.5.

Of the five possibilities for looking beyond enumerated here, the next possibility is probably the biggest bang for your buck:

(3) Recapitulate what we have done and then look inside the mathematics and implementation of LLMs, without actually doing any more implementation, by watching the seven Deep Learning videos by 3Blue1Brown (Grant Sanderson). Sanderson's visualizations are a joy to watch even when he is presenting something you already understand, but perhaps the first three in the series are not worth your time given how much we have learned from Grus. The fourth in the series (**Backpropagation Calculus**) will help cement the slick multi-variable calculus Grus is doing in Chapter 19. The final three ("**Transformers Explained Visually**," "**Attention in Transformers**," and "**How Might LLMs Store Facts?**") will definitely be new, and will give you an idea how neural nets and deep learning are applied to create LLMs like Grok, Gemini, and ChatGPT.

(4) How this is going to affect industry after industry is anybody's guess, but a recent and informed guess (from venture capitalist Marc Andreessen) is in this late-2024 **Lex Fridman interview of Marc Andreessen**. (The link deliberately jumps you to a point over three hours into the interview.)

(5) Following up on a recommendation for further reading given at the end of Chapter 19 of Grus, consider the preliminary version of ***Deep Learning with Python, Third Edition*** (the final version of the third edition is estimated to appear September 2025).