Programming for Computational Social Science (CSS1), Data Analysis in Python

John Serences, jserences@ucsd.edu

January 6th, Winter 2020
Class 00

1

Introductions…

- **John Serences (jserences@ucsd.edu)**
- Professor in Department of Psychology, Neuroscience Graduate Program
- UCSD undergrad, Johns Hopkins graduate school, Salk Institute postdoc
- Research: selective attention and memory systems in humans, focus on using computational models to link brain activity and behavior
- http://serenceslab.ucsd.edu/

2

Introductions…

- **Abhilash Kasarla (akasarla@eng.ucsd.edu)**
- Undergrad in India
- 2nd-year master's student in the ECE, specializing in Machine Learning and Data Science (MLDS)
- Hobbies are playing tennis and following Cricket

3

Introductions…

- Shreya Satish Nayak (snayak@eng.ucsd.edu)
- Undergrad in India.
- 2nd year Master's student in ECE, specialization in Communication Theory and Systems.
- Hiking, binge watching TV shows, and swimming

4

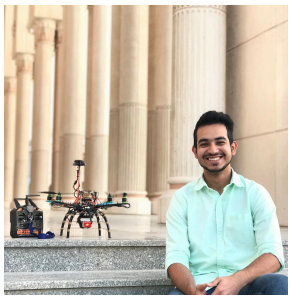
Introductions…

- **Aaditya Bharanidharan (abharani@eng.ucsd.edu)**
- Undergrad Amrita University, Coimbatore, India
- M.S. Chemical engineering; Lithium Battery research
- Hobbies Playing Indian traditional percussion instrument

5

Introductions…

- **Adnan Shahpurwala (ashahpur@eng.ucsd.edu)**
- Undergrad in Dubai
- 2nd year M.Sc. student in ECE, pursuing the Intelligent Systems, Robotics and Control (ISRC) track.
- Hobbies include tinkering with an arduino to make small gadgets and my most recent hobby is app development.



6

Central repository for class material

- https://github.com/JohnSerences/CSS01_W2020
- Good to familiarize yourself with GitHub – commonly used tool for collaborative programing

7

Goals of the course

- Develop solid understanding of the Python language and the Jupyter environment.
  - Open science, data and code sharing
  - Replicability, best practices
- Introductory course for people new to Python and new to coding
  - Experience in another language may help, but no programming experience is necessary
- Why learn to code?
  - Its actually really fun to solve complex problems…by the end of this course you will be impressed with how much you can do

8

Goals of the course…

- Bring everyone along – coding is something that many of you may fear, but all of you can do!
- If you're good at it, get even better by helping other people…best way to REALLY figure out how much you know
- If you're struggling, reach out to get help...from me, from the TAs, from your classmates

9

Most important slides in the class!

- **Don't be afraid to make mistakes**.
  - You'll see me make plenty of mistakes, and that **is a normal part of the process**.
- **Embrace mistakes as learning opportunities.**
  - While learning how to fix a mistake, you will also learn 10 other cool things that you didn't even know that you needed to know.

10

Most important slides in the class!

- Learn how to navigate frustration – its pretty unavoidable in programming.
  - For some, that means googling around for 8 hours straight
  - For others, might be more productive to spend 20 minutes on a problem, and then take a break. Repeat.
  - **In either case, do not wait until the last minute.** Often times solutions come to you if you step away from programming for a day or two (or even while you're sleeping), so give yourself time!

11

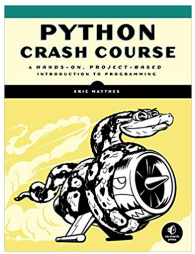
Most important slides in the class!

- Try to ignore the snarky jerks who lurk in stack overflow and other forums…
- Meta example of snarky comments coming into a discussion about how to prevent snarky comments:
  - https://meta.stackoverflow.com/questions/372285/flagging-snarky-comments-to-is-it-possible-questions

12

Textbook (not required, but might help)

- Python crash course : a hands-on, project-based introduction to programming by Eric Matthes, **Nov 1, 2015**
- Not essential, but may help, especially if you are new to programming
- New on Amazon for about $30, used on eBay for $2.95
- There are many other books out there that will work just as well (or maybe better)…take a look around and find one that works for you
- **PYTHON3 (not 2)**



13

Grading: Exams/Final Project

- **Short Quizzes**
  - Starting week 2: At the start of section there will be a short quiz with questions focused on topics from material covered in the previous week.
  - Multiple choice.
  - You can drop two quizzes to give you some flexibility if you need to be late for a class or if you have other obligations that prevent you from preparing.
    - Best 7 out of 9 quiz scores will count.
  - Each quiz will be worth 5 points (35 points total).

14

Grading: Exams/Final Project

- **Midterm**
  - In class midterm that will cover material from the first 1/2 of the class.
  - Multiple choice and written answers.
  - The midterm will be worth 50 points.
- **Final Project**
  - In lieu of a final exam you will have a project where you will need to generate code to address a novel problem.
  - More details will be provided in next few weeks.
  - The final project will be worth 50 points.
  - Due date TBD (whenever the registrar assigns our final time).

15

Problem sets

- In-section problem sets
  - Each week, starting in week 2, there will be a problem set to work on during section and outside of class.
  - Provides hands-on practice that ***is necessary*** to develop fluency.
  - Each problem set worth 5 points
  - Must complete 7 out of 9 problem sets (will drop two lowest scores, so 35 total points)

16

In class participation

- In class participation and attendance
  - Class participation and attendance is worth 10 total points.
  - You will receive the full 10 points if you attend 80% of the lectures (and your grade will be proportionally lowered from there).
  - Attendance will be monitored based on responses to questions using iClickers.
  - And ask questions! It will help everybody.

17

Grading scale

- **Example grades earned by a student:**
  - Quizzes (best 7 of 9): 30/35
  - Midterm: 45/50
  - Final Project: 48/50
  - Problem sets (best 7 of 9): 33/35
  - In class participation: 8.75/10
- **To compute:**
  - (your points) / (total points)
  - 164.75/180 = 91.527777777%, which is an A-
- **Grades will not be rounded**. A 92.99999% is still an A-

A+	97-100
A	93-96.99999
A-	90-92.99999
B+	87-89.99999
B	83-86.99999
B-	80-82.99999
C+	77-79.99999
C	73-76.99999
C-	70-72.99999
D+	67-69.99999
D	63-66.99999
D-	60-62.99999
F	0-59.99999

18

Academic Integrity

- "Integrity of scholarship is essential for an academic community. The University expects that both faculty and students will honor this principle and in so doing protect the validity of University intellectual work. For students, this means that all academic work will be done by the individual to whom it is assigned, without unauthorized aid of any kind."
- This course will make use of online quizzes and exams via Google Forms and/or other similar platforms.
  - Doing assigned work while logged in as another student will be treated as a violation of academic integrity and you will be referred for disciplinary action.
  - Similarly, any communication with other students or anyone else during a quiz or exam will be treated as a violation and also referred for disciplinary action.
- If you have questions about what is and what is not acceptable, please ask me (or the TAs).

19

Important resources for students

- **UCSD's principles of community**
- **Counseling and Psychology Services (CAPS).** "CAPS provides FREE, confidential, psychological counseling and crisis services for registered UCSD students. CAPS also provides a variety of groups, workshops, and drop-in forums."
- **CARE** at the Sexual Assault Resource Center is the UC San Diego confidential advocacy and education office for sexual harassment, sexual violence and gender-based violence (dating violence, domestic violence, stalking).
- **Office for the Prevention of Harassment & Discrimination (OPHD).** OPHD "works to resolve complaints of discrimination and harassment through formal investigation or alternative resolution."

20

Course Schedule (approximate)

- Jan 6,8: What is Python?, Jupyter Environment (Google Colab), First Program, Intro to object types and methods
- Jan 13,15: More on object types, lists, for loops, list comprehensions, slicing lists
- **Jan 20 (no class)**,22: If…elif…else statements, dictionaries
- Jan 27,29: while statements, writing functions
- **Midterm on Feb 3rd, no class on Feb 5th**
- Feb 10,12: Classes, object-oriented programming
- **Feb 17 (no class)**,19: File Input/Output, data formats for files (e.g. JSON, HDF5)
- Feb 24,26: Pandas (data frames)
- March 2,4: NumPy (numerical computing)
- March 9,11: Plotting (Matplotlib)
- Final Project Due: TBD (when our final exam time is allocated by the Registrar)

21

Why learn Python?

- Incredibly flexible for data analysis (modules/libraries)
- Quick development for prototyping/production, excellent GUI support
- Support for generating and compiling C code (faster execution)
- Good balance of flexibility and power against complexity of language/constructs (e.g. Visual Basic/Matlab vs C/C++ vs. Assembly)

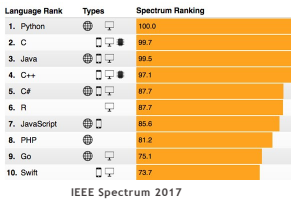
22

Python vs. Matlab

- Programming style
  - 0 vs 1 based indexing
  - block indent
  - () vs [] for function calls, array indexing
- Use in industry/academia
  - Python is far more common in industry – prototyping to full development
  - Many new branches of analysis/computing are led by the Python community with Matlab playing catch-up
- Bleeding edge – good and bad
- Open development community – good and bad
- Some references (note the affiliation of authors …)
  - https://www.mathworks.com/products/matlab/matlab-vs-python.html
  - https://pyzo.org/python_vs_matlab.html
  - http://phillipmfeldman.org/Python/Advantages_of_Python_Over_Matlab.html
  - Perhaps the most balanced (and relevant): https://blog.thedataincubator.com/2017/10/matlab-vs-python-numpy-for-academics-transitioning-into-data-science/

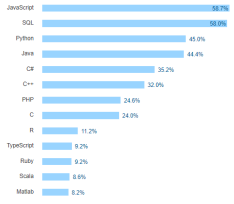
23

Bottom line on Python vs Matlab (and other languages)



Language Rank	Types	Spectrum Ranking
1. Python		100.0
2. C		99.7
3. Java		99.5
4. C++		97.1
5. C#		87.7
6. R		87.7
7. JavaScript		85.8
8. PHP		81.2
9. Go		75.1
10. Swift		73.7

IEEE Spectrum 2017



https://insights.stackoverflow.com/survey/2017

Most popular language for data scientist/engineer [check out this page for other interesting stats]

24

Vy Vo
PhD, computational neuroscience UCSD

- **Research Scientist @ Intel Labs**
- Studying ways to improve machine learning & artificial intelligence, inspired by findings in computational & cognitive neuroscience.
- Skilled at distilling large data sets using Python, Matlab, and R.
  - Specialties include: supervised learning with linear and nonlinear classifiers; dimensionality reduction with PCA/ICA; multivariate, linear, logistic, and other types of regression; model fitting using gradients or grid search; and more.
  - Optimizing analysis on large and noisy datasets using parallel programming.

25

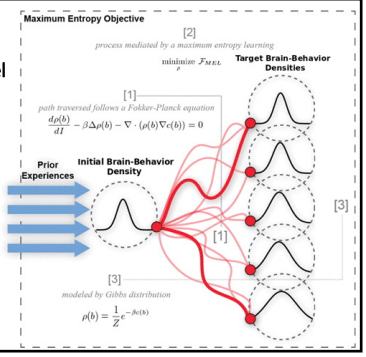
Javier Omar Garcia
Postdoc, cognitive neuroscience UCSD

- **Neuroscientist at Army Research Laboratory**
- Research focused on summarizing whole-brain networks using novel analytical approaches and attempting to understand basic cognitive processes (e.g., attention, visual perception, decision making, motor control).
- Implements models of neural activity in naturalistic environments to provide foundational research for adaptive neurotechnology (i.e. 'brain/computer interfaces', or BCIs).

26

A Minimum Free Energy Model of Motor Learning

Javi Garcia



27

Python programming environments

- Many approaches/environments to develop code
  - Command line interface…pretty basic, no frills
  - Traditional IDE (Integrated Development Environment)…from simple to fancy (.py files)
    - IDLE, ATOM, Sublime, Spyder
  - Notebooks…Integrated web-based environment
    - iPython notebook, aka: Jupyter

28

Jupyter Notebook environment (https://jupyter.org/)

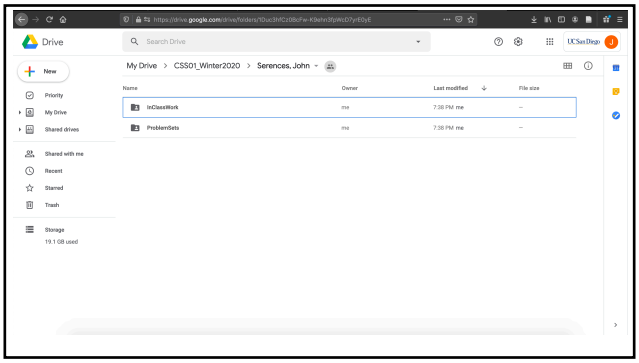
- Contains live code, equations, visualizations and narrative text all in one place
- Easy to share – cross platform and (should) run on any computer and any OS and will produce the same output
- Google Colab is a Jupyter notebook environment that requires no additional setup
  - Runs on virtual machine that is set up when your session starts (and is recycled after session idle)
  - Supports Python 2.7 (deprecated) and Python 3.7 (current active version)
  - All major extensions (modules/libraries)
  - Easy to share directly on drive or after downloading in open source .ipynb format

29

File structure for in-class work

- Shared folder on google drive
  - Use this to store your in-class work from each week
- These folders are shared only with me + TAs
  - Do not share code/notebooks/folders directly with classmates
- Navigate there now and make two sub-folders
  - InClassWork
  - ProblemSets

30



31

Organization of In-class work

- To launch a new notebook and do basic setup
  - Log in to: https://colab.research.google.com/notebooks/welcome.ipynb
    - Can just google: "google colab"
    - Use your AD credentials to login
  - File menu -> "New Python 3 Notebook"
  - File menu -> "Rename" (or just type in the name field in upper left corner)
    - File name should be: Lastname_CSS01_InClass00.ipynb
    - Use exactly this convention EVERY time, only updating the 00 counter (so next class is 01, etc).
  - File menu -> "Locate in drive"
    - This will launch a new window with a file list
    - Right click, Move, navigate to our shared folder and move file to "InClassWork"
    - Now all your work will be saved in our shared folder and you will be able to easily find your notes from each class

32

Organization of problem sets

- Follow directions for setting up a new notebook (see last slide)
- In this case, file name should be: Lastname_CSS01_ProblemSet01.ipynb
  - Use exactly this convention EVERY time, only updating the 00 counter (so next class is 01, etc).
- File menu -> "Locate in drive"
  - This will launch a new window with a file list
  - Right click, Move, navigate to our shared folder and move file to "ProblemSets"
  - Now all your work will be saved in our shared folder we will be able to find your work and grade it

33

Questions?

- Next time – more on Jupyter notebooks, objects, variables.

34