Introduction to Python

2022 Computational and Mathematical Modeling of Cognition

Instructor: Michael D. Nunez (University of Amsterdam)

Join the #python channel on Slack!

The main strategy of learning Python programming

Solving many problems!

Best way to do the problems: A mix of individual and group work, learn to debug errors, and ask many questions

See the PDF document

It is okay to struggle a little bit!

Course overview

Sunday

Session 1: Introduction to Python
Session 2: Flow control (for loops, if statements, etc.)

Monday

Session 3: Plotting and coding style
Session 4: Project work

Sunday: Tentative schedule

09:30 – 13:00 Session 1 Introduction to Python
(basics, errors, loading data, etc.)

- 09:30 - 10:20: Interactive lecture
- 10:30 - 11:20: Individual work w/ Q&A
- 11:30 – 12:20: Group work w/ Q&A
- 12:30 – 13:00: Session 1 review

13:00 – 14:00: Lunch

14:00 – 18:30 Session 2: Flow control
(for loops, if statements, etc.)

- 14:00 - 14:50: Interactive lecture
- 15:00 - 15:50: Individual work w/ Q&A
- 16:00 – 17:20: Group work w/ Q&A

17:30 – 18:30: Project work (SL)
20:00 – 24:00 Optional Python hotelwork ☺, Slack Q&A

Monday: Tentative schedule

09:00 – 09:50 Session 2 review

10:00 – 12:00 Session 3: Plotting and coding style

- 10:00 – 10:50: Interactive lecture
- 11:00 – 12:20: Optional group work w/ Q&A
- 12:30 – 13:00: Session 3 review

13:00 – 14:00: Lunch

14:00 – 15:00 Project work and/or challenge problem work

- 14:00 - 15:00: Individual work w/ Q&A
- 15:00 – 15:50: Project solutions

16:00 – 16:30: Put up posters

16:30 – 18:30 Poster session

??? Optional Python Hotelwork ☺

Why learn Python?

- Some packages are written in / for Python
  - TensorFlow, other neural network and machine learning packages
  - BayesFlow
  - Neuroscience packages, e.g. MNE
- Sometimes faster?
- Great for jobs
- You could be a true developer!

Should I learn Python instead of R?

- **No**, you should learn and use both languages
- It is not like learning two human languages at once
- Both have different communities and thus packages
- Statistics / Psychometrics / Cognitive Modeling -> R
- Machine Learning / Neural Networks / Neuroscience -> Python
- The languages are ultimately very similar with some key differences (e.g. indexing, copying variables in memory, stylistic differences, and use of definitions)

Anaconda Python 3

Download and install the .exe file from this location:
https://www.anaconda.com/products/individual

Run the .exe to install the program. Remember the destination folder, e.g. “C:\Users\[Your Name]\Anaconda3” on Windows

“/Users/[Your Name]/opt/anaconda3” on Mac

Keep “Register Anaconda3 as my default Python 3.9” selected. We will use Python 3.9 for this course, not Python 2.7

Installing your programming IDE

These are the two free recommended programs to use the languages (*Integrated Development Environments; IDEs*): **Rstudio** and **PyCharm Community**

- **RStudio is r**ecommended for those who are only learning R.
- Both **RStudio** and **PyCharm Community** are recommended for most who are learning both Python & R.
- Using both R and Python within **PyCharm Community** is also an option. But this is recommended for only those who already know some R and want both interpreters accessible within the same environment.

Other Python IDE possibilities:

- **Spyder**
- Jupyter Notebooks (not recommended)

https://towardsdatascience.com/5-reasons-why-jupyter-notebooks-suck-4dc201e27086

Download and set up PyCharm

Download and install the Community .exe file from this location:
Windows: https://www.jetbrains.com/pycharm/download/#section=windows
Mac: https://www.jetbrains.com/pycharm/download/#section=mac
Linux: https://www.jetbrains.com/pycharm/download/#section=linux

(Do not download the Profession version unless you are willing to pay for it in 30 days)

Python environments

- Python environments are a way to manage package dependencies
- Point to the correct environment by pointing to individual python.exe files

- Start a new “Pure Python” project with Base interpreter: Python 3.9
- OR: Make sure the we are using the correct verison of Python by going to File -> Settings -> Project: -> Python Interpreter -> Python 3.9 at either “C:\Users\[Your Name]\Anaconda3”\python.exe” or “~/opt/anaconda3/bin/python”
    - If you don’t see it here: First make sure you installed Anaconda Python 3 (see last slide) If you have installed Anaconda: nagivate to Settings symbol on the same page -> System Interpreter -> Navigate to one of the location, either: “C:\Users\[Your Name]\Anaconda3\python.exe” or “/Users/[Your name]/opt/anaconda3/bin/python

IPython terminal versus other terminals

- Python can be compiled and run in many ways
- IPython is the interactive terminal (e.g. like the R terminal)
- IPython has nice “magic” functions like %paste
- import os
- Some of you may prefer to work only in a IPython terminal **outside** of PyCharm / Spyder / etc.

import os
import numpy as np

- Python package imports, similar to R
- Stylistically different
- **numpy** always necessary
- **os** (operating system) often necessary

Installing new packages in Python

- pip install package-name
- This can be used directly in the Ipython terminal and other terminals
- Make sure you are installing to the Python environment (think python.exe location) that you want
- Note that you make break some other dependency or you may need to install a specific version of a package

Statistics in Python

- pip install pingouin
- pip install --user --upgrade pingouin

```
import numpy as np
from scipy import stats
import statsmodels.api as sm
import pingouin as pg
```

A Developer's Best *Secret* to good programming

- GOOGLE (or Bing / DuckDuckGo / etc.)
- StackOverflow
- Github Issues / Module community forums

Extra help from Hannes Rosenbusch (University of Amsterdam)

- https://www.youtube.com/watch?v=5U4WovYcgjE&list=PLY3JDK9oD57jhyqr43dP4JuhiAUhZ9JB-&index=3

- Extra help with PyCharm:
- https://www.youtube.com/watch?v=IO8H0qglovo&list=PLY3JDK9oD57jhyqr43dP4JuhiAUhZ9JB-&index=4

Session 1: Schedule

09:30 – 13:00 Session 1 Introduction to Python (basics, errors, loading data, etc.)

- 09:30 - 10:20: Interactive lecture
- 10:30 - 11:20: Individual work w/ Q&A
- 11:30 – 12:20: Group work w/ Q&A
- 12:30 – 13:00: Session 1 review

13:00 – 14:00: Lunch