Scientific Programming, Analysis, and Visualization with Python

Mteor 2270 – Fall 2025

Python – The Big Picture

- Interpreted
- General purpose, high-level
- Dynamically type
- Multi-paradigm
  - Object-oriented
  - Functional programming features

Python is Interpreted, not Compiled

- Python is an *interpreted* programming language.

- The difference between an interpreted language and a compiled language (such as Fortran or C++) is:

  - Compiled language: the entire source code text file is read and then and then converted into an executable code (in machine language) that can be directly executed by the operating system. This executable code is often optimized for the operating system.

  - Interpreted language the source code text file is read line-by-line and each statement converted into machine language and executed before the next line is read, or the text-file is read completely and converted into an intermediate code that is then further converted into machine language for execution by the operating system.

About Python

- Compiled languages are generally more efficient and faster
  - Conversion to machine language only occurs once and the executable can be run whenever needed.

- In an interpreted language, the interpretation process occurs every time the program is run.

- Interpreted languages are often more flexible and changes can be easily made to the program at runtime.

Python is Dynamically Typed

- Most variables do not need to be declared as real, integer, string, etc.
- The type of a variable is defined by the value assigned to it.
- The type of a variable can change throughout the program execution.
- The opposite of dynamically-typed is *statically-typed*. FORTRAN and C are examples of statically-typed languages.

Python is Object-oriented

- Although Python is an object-oriented language, we will often use it in a more traditional procedural or functional programming paradigm.

- For those who want to truly learn about what object-oriented means, and how to use it to its fullest, I recommend a book titled *The Object-oriented Thought Process* by Matt Weisfeld.

Python is Open-source and Free

- Unlike with MATLAB or IDL there are no licenses required.

- There are companies that put together high quality Python bundles and support them for a fee. The advantage is that the libraries supported by these bundles work together with few bugs.

- However, the same libraries can be assembled and installed freely from other sources, though there may be some additional headaches sorting out dependencies, etc.

Versions of Python

- Python is a relatively young and fast moving language
  - Python 1.0: 1994
  - Python 2.0: 2000
  - Python 2.7: 2010
  - Python 3.0: 2008
  - Python 3.4: 2014
  - Python 3.5: 2015
  - Python 3.6: 2016
  - Python 3.7: 2018
  - Python 3.8: 2019
  - Python 3.9: 2020
  - Python 3.10: 2021
  - Python 3.11: 2022
  - Python 3.12: 2023
  - Python 3.13: 2024
  - Python 3.14: 2025
  - Python 3.16: in the work for 2026

- Python 3.X does not maintain backward compatibility with the older versions of Python
  - Thus, code developed for Python 2.X may not work with Python 3.X, and vice-versa.
  - New, large projects should be written in 3.X
  - For research code, often depending on exotic modules, use 2.7.

- We will use Python 3.x in this course.

Basic Shell Scripting/Programming with Python

- Shell: a user interface for access to an operating system's services.
  - The outer layer between the user and the operating system.

- The first line in your program needs to be:

#!/usr/bin/python

- This line tells the computer what python interpreter to use.

Pycharm/IDLE/Eclipse/Ipython or Jupyter

- Pycharm and IDLE: Interactive development environments.
- Eclipse: a multi-language integrated development environment that is popular on campus.
- For now, we will focus on JupyterHub or JupyterLab for development (Ipython).

Advantages for Scientific Computing

- Large and fast growing user community.
- Large and fast growing number of libraries for scientific computing.
- Well integrated with popular existing codes.
- Highly extensible, large collection of add-on packages.
- Easy communication with R, C, Fortran

When to use Python

- Always!

- Except…..
  - You need the speed of C or Fortran
  - You can re-use significant code written in other languages.
  - You do HPC (using MPI, OpenMP)
  - You can profit from a non-Python tradition in your field or workgroup.
  - Something else is better for the task
    - R, Matlab at times, for example

Documentation and Books

- One book that I use more than any other for reference is:
  - *Python: Essential Reference* (4th ed.) by David M. Beazley.
- Another book that may be helpful is: *Python Pocket Reference* by Mark Lutz.

Online Documentation

- See course web page.