Computing in PHYS 310

Background

In experimental physics, computers are tools on the workbench. Building and operating experiments often requires turning different knobs to see what they do, what their limitations are, and how the instruments can be configured to achieve our goals.

Experimenting also *sometimes* requires our "reading the manual."

Our goal in teaching programming and much of this course is motivated by this openness to experiment and knob-turning. To that end:

- We (your instructors) believe that you will get better with practice.
- We encourage you to work at your own pace to get comfortable with syntax. We will point you to self-guided exercises and resources
- We are available to work with you individually. Please ask for help. Don't spin your wheels.

Why python and jupyter?

Python is an open-source widely accessible, free programming language that is widely used in the scientific community which we will be using for programming and data analysis exercises in this course.

Jupyter is a web-based interactive computing platform in the form of notebooks. Notebooks combine live code, equations, narrative text, graphs, pictures etc. They allow for rapid tests and computations of the character that is needed in the physics lab.

Core competencies

In this course, we expect that in completing the assignments and experiments, you will be able to:

1. Recognize and know when to use python data types such as strings, int, float, booleans,
2. Construct and manipulate containers of objects using lists and arrays (and tuples, dictionaries as needed)
3. Be able to construct loops to iterate calculations.
4. Be able to use pre-built functions and libraries
5. Recognize the difference between required and optional arguments in function definitions
6. Be able to build custom functions to perform specific operations for your task.
7. Learn to use the objects and methods of the numpy package (Numerical python)

8. Learn to use objects and methods of the scipy package (Scieintific python)
9. Learn to use the objects and methods of matplotlib for plotting data
10. Be able to make effective, clearly labeled plots of acquired data.
11. Be able to combine different plots such as data, model and to distinguish them.
12. Be able to perform least squares curve-fitting using scipy.
13. Be able to generate random numbers for Monte Carlo simulations.
14. Be able to load data acquired by some instrument (e.g. oscilloscope, logger-pro, x-ray apparatus etc) into the working memory of a python notebook and analyze the data.
15. Be able to document and test your code so that it is understandable by someone else.
16. .... discover programming features that enable work flow efficiency and help you gain insight on different problems

Supporting materials

- Python for Physicists: https://lucydot.github.io/python_novice/:

  Python tutorials curated by Prof. Lucy Whalley from Northumbria University.
- Computational Physics https://websites.umich.edu/ mejn/cp2/

  Computational Physics textbook by Mark Newman from University of Michigan
- Scipy Tutorials https://docs.scipy.org/doc/scipy/tutorial/
- NumPy Tutorials https://numpy.org/learn/
- Matplotlib Examples matplotlib.org