Python for Scientific Computing

Course Goals

- Simply: to learn how to use python to do
  - Numerical analysis
  - Data analysis
  - Plotting and visualizations
  - Symbol mathematics
  - Write applications
  - …

Class Participation

- We'll learn by interactively trying out some ideas and looking at code
  - I want to learn from all of you—share you experiences and expertise
- We'll use slack to communicate out of class
  - Everyone should have received an invite to our slack: https://python-sbu.slack.com
- Try out ideas and report to the class what you've learned
  - You should have an installation of python on your laptop
  - Alternately, you can use the Virtual SINC site on campus, which has Anaconda Python installed

Slack

- Log onto our slack as soon as possible
  - If you didn't get an invite, e-mail me, and I'll add you
- Slack is a web-based team chat tool
  - I've setup a number of channels for us to focus our discussions
  - Everyone is expected to participate
- Your course grade is based on your participation
  - I have a simple script(https://github.com/zingale/slack_grader) that I'll use for grading
    - Good contributions will get a comment on the slack, counting as "+1"
    - The script will record these points over the semester
    - Help, by using slack reactions for useful comments from your classmates
- In "free" mode, slack only keeps 10k messages—I don't think we'll go over that during the semester

Why Slack?

- One of the goals of this class is to teach you tools that are used by computational science groups
- Slack has gained enormous adoption by research groups over the past few years
  - We'll see how to integrate github repos to it, so everyone can keep on top of code developments
  - It's easy to post code snippets in conversations
  - There are lots of integrations that are available to extend its usefulness

Why Python?

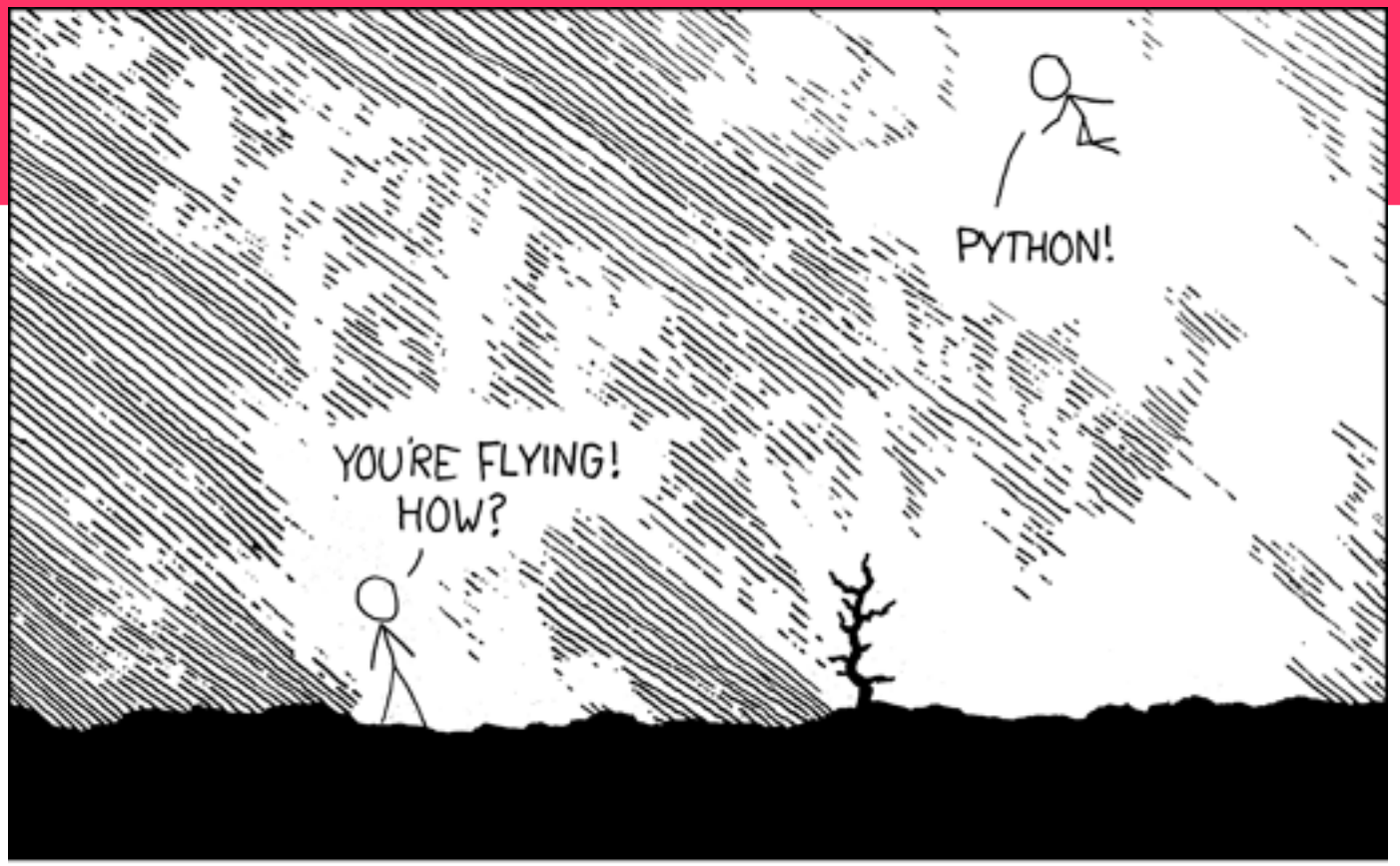
- Very high-level language
  - Provides many complex data-structures (lists, dictionaries, …)
  - Your code is shorter than a comparable algorithm in a compiled language
- Many powerful libraries to perform complex tasks
  - Parse structured inputs files
  - send e-mail
  - interact with the operating system
  - make plots
  - make GUIs
  - do scientific computations
  - …
- Easy to prototype new tools
- Cross-platform and Free

Why Python?

- Dynamically-typed
- Object-oriented foundation
- Extensible (easy to call Fortran, C/C++, ...)
- Automatic memory management (garbage collection)
- Ease of readability (whitespace matters)

... and for this course ...

- Very widely adopted in the scientific community
  - Mostly due to the very powerful array library NumPy


PYTHON!
YOU'RE FLYING! HOW?


I LEARNED IT LAST NIGHT! EVERYTHING IS SO SIMPLE!
HELLO WORLD IS JUST print "Hello, world!"
I DUNNO... DYNAMIC TYPING? WHITESPACE?
COME JOIN US! PROGRAMMING IS FUN AGAIN! IT'S A WHOLE NEW WORLD UP HERE!
BUT HOW ARE YOU FLYING?
I JUST TYPED import antigravity
THAT'S IT?
... I ALSO SAMPLED EVERYTHING IN THE MEDICINE CABINET FOR COMPARISON.
BUT I THINK THIS IS THE PYTHON.

(xkcd)

Installing Python

- Linux
  - Python is probably already installed
  - The dependencies we need for our class should be available through your package manager
- OS X / Windows
  - The easiest way to get everything we need for class is by installing Anaconda: https://www.continuum.io/downloads
    - You'll have a choice of python 2.7 or 3.6—choose python 3.6
- If you run into problems ask on slack (there is an “installation” channel), or come by my office

Hello, World!

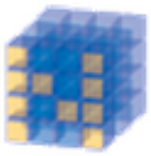
- If you have python installed properly, you can start python simply by typing python at your command line prompt
  - The python shell will come up
  - Our hello world program is simply:

    ```
    print("hello, world")
    ```
  - Or, in python 2 (more on this later…):

    ```
    print "hello, world"
    ```

Scientific Python Stack

- Most scientific libraries in python build on the Scientific Python stack:



NumPy
Base N-dimensional array package

SciPy library
Fundamental library for scientific computing

Matplotlib
Comprehensive 2D Plotting

IPython
Enhanced Interactive Console

Sympy
Symbolic mathematics

pandas
Data structures & analysis

(scipy.org)

IPython Shell

- Type ipython (or ipython3) at your prompt
- Like the standard shell, this runs in a terminal, but provides many conveniences (type %quickref to see an overview)
  - Scrollback history preserved between sessions
  - Built-in help with ?
    - function-name?
    - object?
  - Magics (%lsmagic lists all the magic functions)
    - %run script: runs a script
    - %timeit: times a command
    - Lots of magics to deal with command history (including %history)
  - Tab completion
  - Run system commands (prefix with a !)
  - Last 3 output objects are referred to via _, __, ___

Jupyter Notebooks

- A web-based environment that combines code and output, plots, plain text/stylized headings, LaTeX, ...
  - Notebooks can be saved and shared
  - Viewable on the web via: http://nbviewer.ipython.org/
  - Provides a complete view of your entire workflow
- Start with jupyter notebook
- I'll provide notebooks for a lot of the lectures to follow so you can play along on your own
  - The best way to learn is to experiment—download the notebooks and play around
  - Discuss anything you don't understand in the discussion forum

Python 2.x vs. Python 3

- See https://wiki.python.org/moin/Python2orPython3
- Mostly about cleaning up the language and making things consistent
  - e.g. print is a statement in python 2.x but a function in 3.x
- Some trivial differences
- .pyc are now stored in a __pycache__ directory
- Some gotyas:
  - 1/2 will give different results between python 2 and 3
- It's possible to write code that works with both python 2 and 3—often we will do so by importing from __future__
- We will focus on python 3.x

Class Organization

- We'll work mostly with Jupyter notebooks from now one (with a few exceptions)
- Each week, I'll ask you to work through some notebooks on your own, outside of class
  - These will always be posted on the class website
  - Great opportunity to ask questions on slack
- The hope is that we'll get a lot of the basic concepts for the week covered by working through the notebooks
- In class, we'll work on some exercises together
- We'll fine-tune some of this as we go through the semester