Python as a Scientific Language

A Brief Introduction

D. T. Welling

U. of Michigan Climate and Space

Play along at home: www-personal.umich.edu/~dwelling/python/

Python is

…a multi-p…ct
oriented…guage.

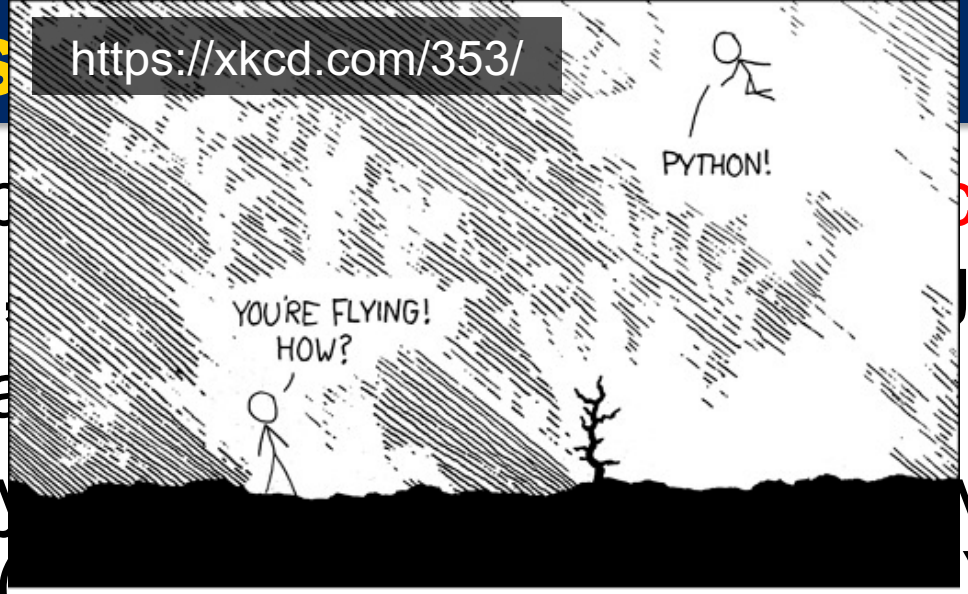
…named a…

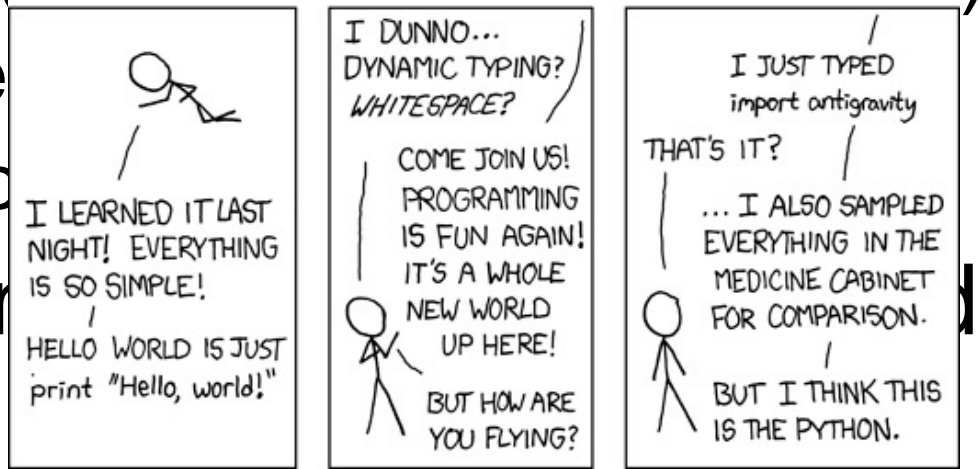
…relatively…version 2
in 2000 (…).

…extreme…

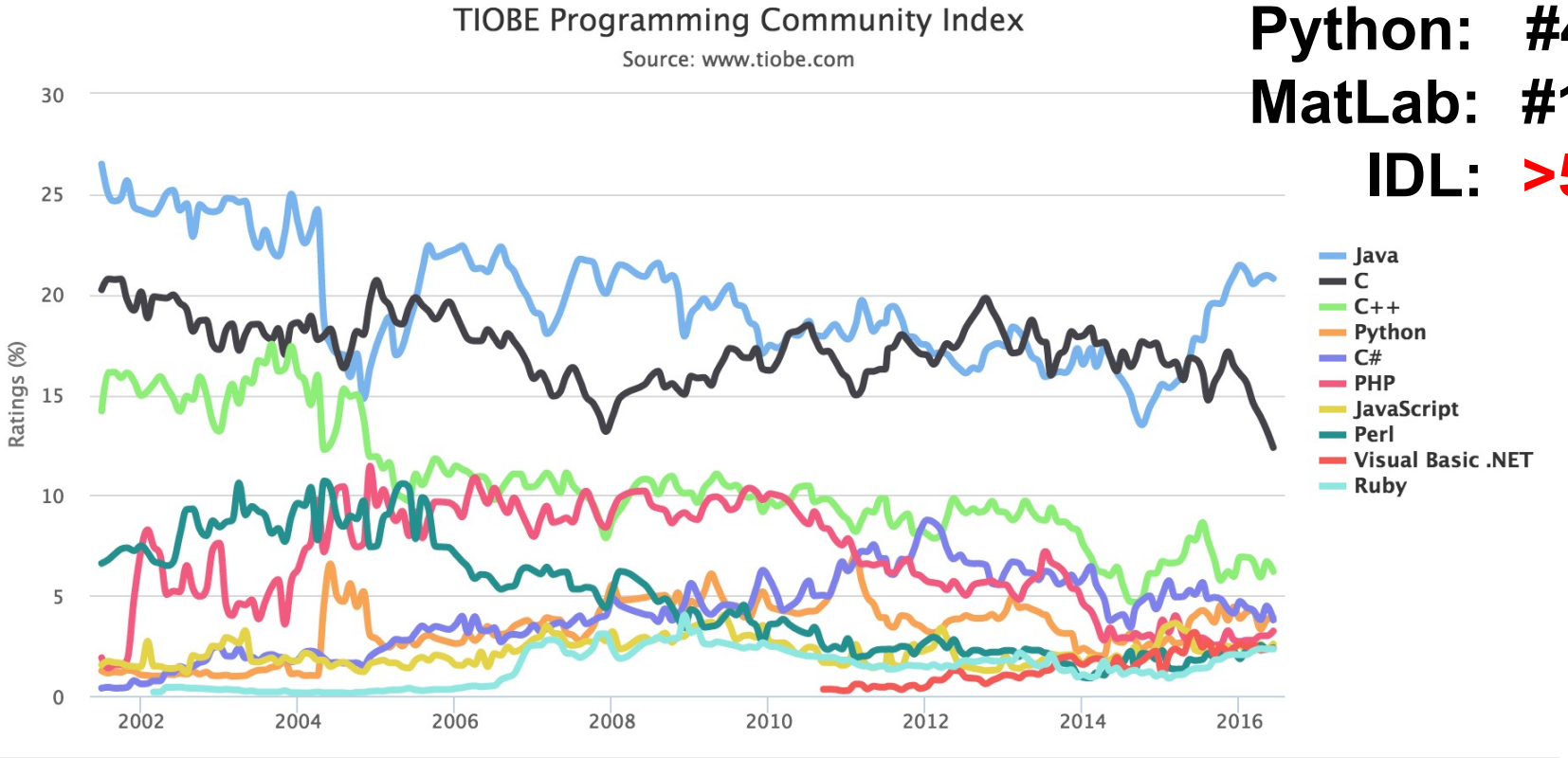
…Open so…

Curren…d 3.5.1





TIOBE Rankings



Python: #4

MatLab: #17

IDL: >50

Why Python?

- Powerful scripting rivaling Perl, Bash, etc.
- "Batteries included" (GUI, regex, web scripting, d… …ne box.)
- Ubiqui… (Win…*ni… …e).
- Extensi… … FORTRAN
- Emphasize… …larity of source code
- Natural, easy, powerful Object Oriented

IT'S FREE

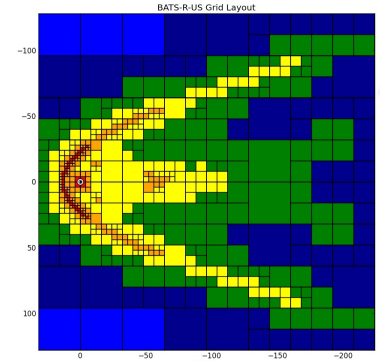
and I'm cheap.

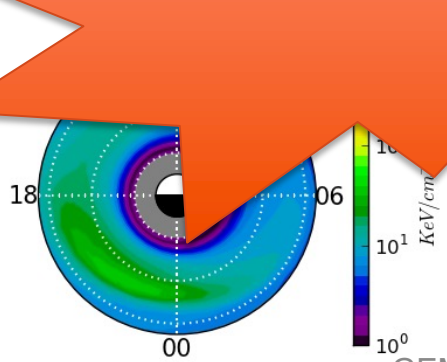
Why Python for Science?

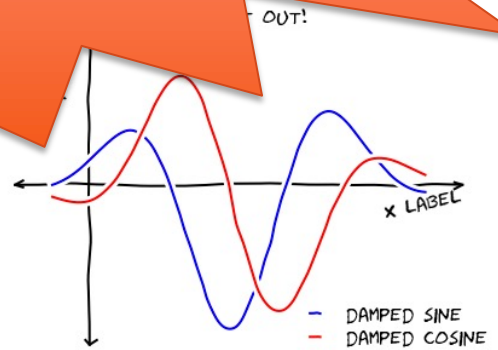
- Numpy (array algebra), SciPy (common scientific functions), and Matplotlib (plotting) combine to rival IDL, MatLab.
- Easy to drop int
- Combines s lization for *complet*

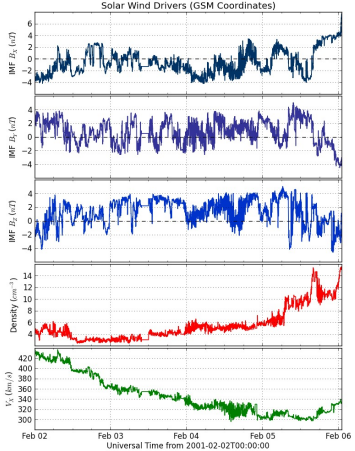
IT'S FREE

and I'm cheap.









Resources

www.python.org	Source, documentation, other resources.
…/dev/peps/pep-0008/	Style guide (suggested coding conventions)
enthought.com	Python think-tank; Canopy Python Distribution
Dive Into Python (Mark Pilgrim)	Open-source introduction (.net for website)
Core Python Programming (Wesley Chun)	Excellent introduction and reference; very thorough.
GDS	**G**oogle **D**at **S**htuff for nearly any issue
My Crap	www-personal.umich.edu/~dwelling/python/

Getting Python

***nix**	Already Done! Use your package manager to get additional modules.
OS X	Use internal installation (WARNING: non-standard!)
	Use package manager (Fink, MacPorts, Homebrew).
	Use Enthought Canopy Python distribution.
Windows	Get software from python.org and install.
	Use Enthought Canopy Python distribution.

Scientists will want Python, Numpy, SciPy, Matplotlib, and IPython.

Interfacing with Python

There are many, many ways to work with Python.

- Scripting and executing from the system shell.
- Command line interfaces through a Python shell, such as the default shell or IPython.
- Interactive Development Environments that combine text editors with shell prompts, such as Spyder.
- Jupyter Notebook, a web-based interactive session that combines mark-up and code.

An Example: example_imf.py

```
#!/usr/bin/env python

'''
An example module […]
'''

import numpy as np

def format_ax(ax, ylabel=None):
    '''
    Format an axes object […]
    '''
    <commands>
# Comments start with hashtags!
```

- "shebang" – tells shell how to execute.
- "docstring" – long form comments/documentation
- Imports – include code from other python files
- Function & variable definitions (note docstrings associated with definitions)

example_imf.py

```
class ImfData(dict):
    '''
    A class for handling Imf […]
    '''
    def __init__(self,
filename):
        <commands>



    def calc_v(self):
        '''
        Calculate […]
        '''
        <commands>
```

Class definitions:

- Class-level docstring
- Special method definitions: define basic object behavior
- Method definitions: functions that leverage object attributes

NOTE NESTED TABBING

example_imf.py

```
if __name__ == '__main__':
    <commands>
    <commands>
    <commands>
```

Optional code that runs in the “__main__” namespace (when file is executed, not imported).