Python Primer

Adam D. Myers

Introduction

Python is a high-level scripting language that is useful for manipulating data. As with any programming language, Python has some undesirable features, such as some relatively slow processes and a vast library of complicated dependencies. But Python can be used to wrap faster code such as C (using, e.g, the `os.system` or `subprocess.call` routines), and, perhaps most usefully, it has a large number of existing packages for manipulating data from large surveys.

Getting Started

You will need to set up our local `anaconda` (a.k.a. `conda`) install of Python, which contains tools relevant for ASTR 5160. To do so, place the following commands in the `.bashrc` file in your home directory:

```
export PATH=/usr/local/Anaconda/bin:$PATH
source activate astroconda
```

Note that to use conda, you will have to be in the bash shell. So, if this is not your usual shell then you will have to remember to issue:

```
bash
```

at the UNIX command line.

Most often, Python is used interactively, or by writing code in a text file and running that text file at the UNIX prompt. To use Python interactively, you can either type `python` at the UNIX prompt, or, you can use the interactive jupyter notebook tool by typing the following:

```
jupyter notebook
```

at the UNIX prompt. For the simple `python` option you should see, e.g.:

```
(astroconda) bash-4.1$ python
Python 2.7.12 |Continuum Analytics, Inc.| (default, Jul 2 2016, 17:42:40)
[GCC 4.4.7 20120313 (Red Hat 4.4.7-1)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
Anaconda is brought to you by Continuum Analytics.
Please check out:  http://continuum.io/thanks and https://anaconda.org
>>>
```

For the `jupyter` option, a web browser should be launched that displays your directory structure. Navigate to your SVN directory of interest (by clicking on folders) and then, in the top right corner, click on `New` and then launch a `Python2` kernel. If you choose to use jupyter, then it will be useful to read the following:

```
https://nbviewer.jupyter.org/github/jupyter/notebook/blob/master/docs/source/examples/
Notebook/Notebook%20Basics.ipynb
https://nbviewer.jupyter.org/github/jupyter/notebook/blob/master/docs/source/examples/
Notebook/Running%20Code.ipynb
```

As an example of how to run a Python script directly from the command line, create a text file, called, e.g. `myprogram.py` that contains the following lines of Python code:

```
def myprogram():
     print('hello world')

if __name__ == "__main__":
     myprogram()
```

and then run it from the UNIX command line as follows:

```
python myprogram.py
```

Note that *any piece of code you submit as a homework answer should be able to be run from the UNIX command line, and the homework directory should include a README file indicating exactly how to run the code*.

Python is heavily documented online. Usually, if you need a command that performs a specific task, you'll find something on the web. Science is a collaborative endeavor. I have no problem with you using other people's Python modules in this class, including modules from other student's homework submissions submitted *in weeks prior to the week of the current homework* (e.g., in Week 2 of the class it is fine to use other student's functions and procedures from Week 1). If you are completely unfamiliar with Python, then you should visit the Python tutorial at:

```
https://docs.python.org/2/tutorial/
```

and try sections 3–6 before the next class.

The PYTHON_PATH

You can import any modules to use in code if they are in your current directory or in a directory that is listed in your `PYTHON_PATH`. Look online for the UNIX commands to see how you can change your `PYTHON_PATH` to access other directories ... for instance, in the event that you want to directly import each other's work from previous weeks.

Common Commands and Plotting

For plotting, we will use the matplotlib library. Try the first few examples at the matplotlib tutorial:

`http://matplotlib.org/users/pyplot_tutorial.html`

Note that it if you to choose to use the jupyter notebook option for interactive Python, then adding the command:

```
%matplotlib inline
```

at any point in your jupyter notebook before you import matplotlib will allow you to plot figures inline in the actual notebook.

Good Practice with Python and SVN

1. *Write short pieces of code*. Longer code is less likely to be used by somebody else. If you need to write code to; 1) determine the area in a box, then; 2) randomly populate that box at a particularly number density, then 3) measure the clustering of points in that box; then *write three separate functions* and combine the three tasks using a single, short procedure.

2. Give each piece of code *an informative name*, and separate the words in the name by underscores. So, for instance, if you have code to populate the area on a sphere at random, call that code `populate_sphere_at_random`. To distinguish file names from code, separate file names by dashes, as in `random-points-ra-0-to-12-hrs.txt` .

3. Give each piece of code *an informative header*[1], which might be, for example (from some of my own work):

```
def htm(ra,dec,level=20):
   """Look up an HTM ID

   Return the Hierarchical Triangular Mesh coordinate that
   corresponds to the point on the sphere at a given RA and Dec.

   INPUTS:
   ra:  :class:`float`
      Right Ascension in degrees
   dec:  :class:`float`
      Declination in degrees
   level:  :class:`int', defaults to 20
      Level of the HTM tree at which to report

   OUTPUTS:
```

[1] In fact, I recommend using the example header as a template *whenever* you submit your own code in this class

```
:class:‘string’
   The HTM coordinate index-string that corresponds to the
   inputted point on the surface of the sphere.
:class:‘list‘
   3-vector containing the Cartesian coordinates of the input point

CAVEATS: Beyond Level 25, the resolution of the pixels becomes
smaller than 64-bit float precision used by PYTHON can handle.  For
any input at a resolution greater than level 25, the user will be
warned and the level 25 result will be returned.

NOTES: Typically performs 1250 (level 5) lookups/sec on a 1.6GHz processor.

EXAMPLES:
   >>> print( htm(123.45,67.89))
      (’N213312001013012303312’, (-0.20746735989196083, 0.3140440906223856,
      (0.926462953239156))

First written by Adam D. Myers, May 20, 2004
"""
```

This approach not only makes the code more transparent to others, you’ll also thank me in Week 10 when you want to use the code you wrote in Week 2 but you can’t remember what it does...