Adding Modules to Python

Jason Wang

October 17, 2013

1 Overview

For this next lab, you will need the `pyfits` module in Python to read and access FITS files, the standard data format for astronomy. However, `pyfits` is not installed on the UGAstro machines by default, so you will need to install it first.

Bill has generously put in the time to install `numpy` as `matplotlib` onto all of the UGAstro machines so you always have access to them. But what if you want to install some other Python module to use? Luckily, Python has developed a convenient way of managing modules called `pip` (`https://pypi.python.org/pypi/pip`). `pip` is an UNIX command line tool that can install any Python module from the PyPI (Python Package Index) onto your computer hassle free and is one of the infinite reasons Python is better than IDL.

One thing to note about the UGAstro setup is that you may have noticed you do not have administrative privileges and cannot install programs on a computer. This is where a quick understanding of Python's `import` statement is useful. Have you ever wondered the magic that happens when you type `import numpy` and you can suddenly use numpy? When you import something, python looks in two places usually to try to find the module you are importing: your current working directory and `sys.path`. What is `sys.path`? Try typing the following two lines in Python:

```
import sys
print sys.path
```

This tells you everywhere on the computer Python will search to try to import the module you just asked for it to import. You will notice most (if not all) of the paths listed are to folders that you do not have permission to access. However, you can actually add to that list of places to search for and `pip` gives you the option to do so.

2 Practical Instructions

The documentation for using `pip` can be found at `http://www.pip-installer.org/en/latest/usage.html`. To install a package for only yourself, use the following UNIX command in a terminal window:

```
pip install --user packagename
```

where `packagename` is the package you want to install (e.g. `pyfits`). Now if you print out `sys.path` you will see a filepath that corresponds to a folder in your home direction (e.g. /home/jwang/.local/lib/python2.7/site-package) and that is where `pip` has installed your package. This is convenient since if you log into any other UGAstro machine, you will also have that module installed and ready to go.

3 Modifying The Search Path

If you want to add a python library you wrote to `sys.path` so you can import it else where, I recommend one of these two ways of doing it. The first way is to temporarily add it to `sys.path`. Use the following lines of code in Python to import your module:

```
import sys
sys.path.append('/the/path/to/the/module')
import mymodule
```

for a file /the/path/to/the/module/mymodule.py. You should do this for the most part since you probably are not intending to permanately adding it to your Python library and it is pretty convenient to do.

If you really want to add something permanately you can add it to the environment variable `$PYTHONPATH` which is one thing Python uses to locate all the libraries available for import. To do this, first you need to edit the configuration file for the terminal. Open up the file ∼/.cshrc in your favorite editor and add the following line to the end of the file:

```
setenv PYTHONPATH /path/to/file
```

You can check it has been succesfully added by restarting your terminal window and typing:

```
echo $PYTHONPATH
```

Note: if you already have something set as `$PYTHONPATH`, you will have to append your new filepath instead of just setting it equal. So in the ∼/.cshrc file, append the following line instead:

```
setenv PYTHONPATH ${PYTHONPATH}:/path/to/new/file
```