Week 14 Lecture: Advanced Python Programming, Concepts and Examples

Introduction to Programming for GIS & Remote Sensing
GEO6938-1469
GEO4938-147A
Re-cap From Past Lectures

• Last week:
  – Package management and Python extension
    • Python Package Index – pip and pip-Win
    • Examples of using Python packages
This Week: Advanced Topics

• How do we take all we know and go further?
• Keeping it specific to Python:
  – Parallel processing
  – Advanced plotting and scientific computing
  – Spatial data in Python
Basics of Parallel Processing

• Multiple cores == workers

• How do we spread work among them?
  – “Multi-threaded” or “parallel” processes

• In Python, the simplest technique:
  – `import multiprocessing`
```python
import multiprocessing
import time

## Create a CPU intensive function:
def waste_time(n):
    i = 0
    while i < 1024*1024*64:
        i += 1
    return(n)

## We can create a callback function to log the result of our function:
result_list = []
def log_result(result):
    ## This is called whenever waste_time(n) returns a result.
    ## result_list is modified only by the main process, not the pool workers.
    result_list.append(result)

def apply_async_with_callback():
    ## We are going to create a worker pool, where we can control the number
    ## of workers with the processes= argument to the Pool() function:
    pool = multiprocessing.Pool(processes=2)

    ## We'll keep track of the time it takes to run the program:
    start_time = time.time()
    for i in range(4):
        pool.apply_async(waste_time, args=(i,), callback=log_result)
    pool.close()
    pool.join()

    print(result_list)
    print("Process took " + str(time.time() - start_time) + " seconds...")

if __name__ == '__main__':
    apply_async_with_callback()
```
Example 1: Multiprocessing

- Open the Python file located in this week’s /src folder
- Note that we are only using base Python packages
Multiprocessing Cont’d.

• We can send data to and receive data back from individual workers
  – To share data between workers you can use “thread and process safe” communication:
    • The multiprocessing.Queue object
  • The order in which the items are sent to workers does not correspond to the order you get them back when using `apply_async()`
    – If it matters, use `Pool.map()` instead
Multiprocessing is Complex

• This is barely scratching the surface of multi-threaded or parallel processes
• Piping, cores vs. threads, etc.
• Start-up and overhead cost in CPU processing
• Learn more:
  – http://docs.python.org/2/library/multiprocessing.html
  – Multiprocessing and ArcGIS:
Scientific Computing in Python

• Moving away from ArcGIS and toward more general scientific computing

• Many packages good for scientific computing
  – Often platform specific
  – Best way to get them?
    • Pre-built platform
My Suggestion for Python-based Scientific Computing

• Anaconda
  – [https://store.continuum.io/cshop/anaconda/](https://store.continuum.io/cshop/anaconda/)
  – It’s free, based on all open source software
  – Includes Python, numpy, matplotlib, scipy, sci-kit, etc.

  – Has its own package management called conda which also includes a C++ distributable:
    • e.g. at command prompt: `conda install matplotlib`
Examples 2-5, Anaconda

• We will use the Miniconda distribution of Anaconda (found in /lib of your assignment) and illustrate the use of the conda package management system
  – Which includes pip for non-conda packages:
    • conda install pip
Installing Conda Packages is Easy

- From the command prompt:

```
C:\Users\forrest>conda install Spyder
```
For Our Examples:

conda install scipy
conda install matplotlib
conda install pandas
conda install pip
conda install seaborn
conda install husl
conda install statsmodels
conda install moss
conda install scikit-learn
conda install bokeh
Example 2 - Seaborn Plotting

Example 3-5 - Bokeh Plotting

- [http://bokeh.pydata.org/index.html](http://bokeh.pydata.org/index.html)

Welcome to Bokeh

Bokeh is a Python interactive visualization library for large datasets that natively uses the latest web technologies for enhancing, concise and elegant, concise construction of novel graphics in the style of Protovis/D3, while delivering high-performance to thin clients.

For more information about the goals and direction of the project, please see the Technical Vision.

To get started quickly, follow the Quickstart.

Visit the source repository: [https://github.com/ContinuumIO/bokeh](https://github.com/ContinuumIO/bokeh)

Be sure to follow us on Twitter @bokehplots!
Example 6-7 – Plotting Maps and Spatial Data in Python

- Using the Basemap extension to matplotlib
  - [http://matplotlib.org/basemap/users/examples.html](http://matplotlib.org/basemap/users/examples.html)

- Install binaries:
Reading and Writing Shapefiles in Python - pyshp

• ... without arcpy!

• https://code.google.com/p/pyshp/
pyshp Interface for Shapefile

Reading Points in Shapes

```python
>>> import shapefile
>>> sf = shapefile.Reader("shapefiles/blockgroups")
>>> shapes = sf.shapes()
>>> # Read the bounding box from the 4th shape
>>> shapes[3].bbox
[-122.485792, 37.786931000000003, -122.446285, 37.811019000000002]
>>> # Read the 8th point in the 4th shape
>>> shapes[3].points[7]
[-122.471063, 37.787402999999998]
```

Reading Database Attributes

```python
>>> # Read the field descriptors for the database file
>>> sf.fields
[(["DeletionFlag", "C", 1, 0], ["AREA", "N", 18, 5],
... ["BKG_KEY", "C", 12, 0], ["POP1990", "N", 9, 0], ["POP90_SQMI", "N", 10, 1],
... ["HOUSEHOLDS", "N", 9, 0],
... ["MALES", "N", 9, 0], ["FEMALES", "N", 9, 0])
>>> # Read the 2nd and 3rd field values of the 4th database record
>>> sf.records[3][1:3]
[060750601001', 4715]
```
Creating Shapefiles From Scratch

Writing Shapefiles

```python
>>> import shapefile
>>> # Make a point shapefile
>>> w = shapefile.Writer(shapefile.POINT)
>>> w.point(90.3, 30)
>>> w.point(92, 40)
>>> w.point(-122.4, 30)
>>> w.point(-90, 35.1)
>>> w.field('FIRST_FLD')
>>> w.field('SECOND_FLD', 'C', '40')
>>> w.record('First', 'Point')
>>> w.record('Second', 'Point')
>>> w.record('Third', 'Point')
>>> w.record('Fourth', 'Point')
>>> w.save('shapefiles/test/point')

>>> # Create a polygon shapefile
>>> w = shapefile.Writer(shapefile.POLYGON)
>>> w.poly(parts=[[1, 5], [5, 5], [5, 1], [3, 3], [1, 1]])
>>> w.field('FIRST_FLD', 'C', '40')
>>> w.field('SECOND_FLD', 'C', '40')
>>> w.record('First', 'Polygon')
>>> w.save('shapefiles/test/polygon')
```
This isn’t the end...

• ... if you have any questions let’s talk about them next week!
• No specific lecture or topics will be covered unless you propose them
• Time to work on your projects
• Time to address concerns with me