ECON 370 Quantitative Economics with Python

Lecture 6: Python Fundamentals (Part 4)

Zhen Huo [zh335@nyu.edu]
Matthew McKay [mm8390@nyu.edu]
Thomas Sargent [thomas.sargent@nyu.edu]

Spring 2016

Agenda

Visualizing Data

1. matplotlib
2. line charts
3. bar charts
4. histograms
5. Other packages

Numeric Computing

1. The numpy package
2. Arrays
3. Matrices
4. Linear Algebra
5. Solving Systems of Equations

matplotlib reading

`http://quant-econ.net/py/matplotlib.html`

matplotlib

matplotlib is a versatile Python plotting package.

It is very general and capable of producing very simple to highly complex plots.

The programmer has a lot of control, but the tradeoff is convenience in defining the plots.

Some recent packages provide subsets of plots that are easier to implement quickly but are typically less versatile tools.

Simple Example

```
import matplotlib.pyplot as plt #Import plotting library
import numpy as np
x = np.linspace(0, 10, 200)    #Generate Some Data
y = np.sin(x)
plt.plot(x, y, 'b-', linewidth=2)  #Request a Plot
plt.show()
```

Using matplotlib

There are two primary ways to use the matplotlib library

1. **pylab** which provides a matlab like interface with convenient plotting functions
2. **object-oriented** interface which provides a high degree of control

Basic Plot Types

1. Line Charts
2. Histograms
3. Scatter Plots

See: **intro-to-matplotlib.ipynb**

Other packages

Some other packages include:

1. **ggplot** `http://ggplot.yhathq.com/`
2. **seaborn** `http://stanford.edu/~mwaskom/software/seaborn/`
3. **pandas** `http://pandas.pydata.org/`
4. **bokeh** `http://bokeh.pydata.org/en/latest/`
5. many others ...

Numerical Computing Reading

NumPy: `http://quant-econ.net/py/numpy.html`
Linear Algebra: `http://quant-econ.net/py/linear_algebra.html`

Documentation:

1. **NumPy** `http://docs.scipy.org/doc/numpy/user/`

The NumPy package

NumPy is the key package for scientific computing with Python.

NumPy provides:

1. N-dimensional array object (ndarray)
2. broadcasting functions
3. shape manipulation
4. sorting
5. basic linear algebra
6. random number simulation
7. Fourier transforms
8. basic statistical operations

and is widely used as a foundation by other packages, it is fast and it is stable.

Why is it fast?

Underlying implementation is written in C (mostly) or Fortran.

Python is used as a higher level environment to work productively while the computation occurs in libraries that are statically compiled and execute quickly.

There are other ways to get **speed** but this is the best place to start as it is the foundation of so many Python projects.

NumPy Arrays

```
import numpy as np
a = np.array([1.0, 2.0, 3.0])
b = np.array([[1.0, 2.0, 3.0],
              [4.0, 5.0, 6.0]])
```

How are ndarray objects different to standard Python sequences (lists)?

1. NumPy arrays are homogenous in data type (dtype)
2. NumPy arrays have fixed size when they are created

These limitations come with much improved performance for numerical computing.

NumPy Arrays

See **intro-to-numpy.ipynb** on GitHub

Matrices

`http://quant-econ.net/py/linear_algebra.html#matrices`

Solving Systems of Equations

`http://quant-econ.net/py/linear_algebra.html#solving-systems-of-equations`