CME 193: Introduction to Scientific Python
Lecture 5: Data Visualization

Austin Benson
Dan Frank
Institute for Computational and Mathematical Engineering (ICME)

April 18, 2013
Homework

- Homework 4 is due Tuesday, April 23

- Homework 5 will be extra credit (in case you failed an assignment). To be posted.
A lesson in Python style

Matlab and SciPy

matplotlib
The **pythonic** way is the simple, correct, most beautiful way of doing things.
There can be beauty in brevity (and vice versa).

```python
def scale(A):
    shape = A.shape
    m = shape[0]
    n = shape[1]
    for i in range(n):
        x = A[0, i]
        A[0, i] = x * 2
    for i in range(n):
        x = A[m - 1, i]
        A[m - 1, i] = x * 2
    if m < n:
        k = m
    else:
        k = n
    for i in range(k):
        x = A[i, i]
        A[i, i] = x * 5
```
In general, use *=, + =, -=, etc.

```python
def scale(A):
    shape = A.shape
    m = shape[0]
    n = shape[1]
    for i in range(n):
        A[0, i] *= 2
    for i in range(n):
        A[m - 1, i] *= 2
    if m < n:
        k = m
    else:
        k = n
    for i in range(k):
        A[i, i] *= 5
```
Try to use the Python built-in functions (don’t re-invent the wheel).

```python
def scale(A):
    shape = A.shape
    m = shape[0]
    n = shape[1]
    for i in range(n):
        A[0, i] *= 2
    for i in range(n):
        A[m - 1, i] *= 2
    k = min(m, n)
    for i in range(k):
        A[i, i] *= 5
```
Know the normal syntax of your data structures.

```python
def scale(A):
    A[0, :] *= 2
    A[-1, :] *= 2
    shape = A.shape
    m = shape[0]
    n = shape[1]
    k = min(m, n)
    for i in range(k):
        A[i, i] *= 5
```
Avoid defining too many variables.

```python
def scale(A):
    A[0, :] *= 2
    A[-1, :] *= 2
    m, n = A.shape
    k = min(m, n)
    for i in range(k):
        A[i, i] *= 5
```
Avoid defining too many variables...

```python
def scale(A):
    A[0, :] *= 2
    A[-1, :] *= 2
    k = min(A.shape)
    for i in range(k):
        A[i, i] *= 5
```
Avoid defining too many variables...

```python
def scale(A):
    A[0, :] *= 2
    A[-1, :] *= 2
    k = min(A.shape)
    for i in range(k):
        A[i, i] *= 5
```
def scale(A):
    A[0, :] *= 2
    A[-1, :] *= 2
    for i in range(min(A.shape)):
        A[i, i] *= 5
Know a little more advanced syntax.

```python
def scale(A):
    A[[0, -1], :] *= 2
    for i in range(min(A.shape)):
        A[i, i] *= 5
```
Search for functions to do the work for you.

```python
def scale(A):
    A[[0, -1], :] *= 2
    np.fill_diagonal(A, 5 * np.diag(A))
```
```python
def scale(A):
    shape = A.shape
    m = shape[0]
    n = shape[1]
    for i in range(n):
        x = A[0, i]
        A[0, i] = x * 2
    for i in range(n):
        x = A[m - 1, i]
        A[m - 1, i] = x * 2
    if m < n:
        k = m
    else:
        k = n
    for i in range(k):
        x = A[i, i]
        A[i, i] = x * 5
```

Do not be afraid to re-write code!

```python
def scale(A):
    A[[0, -1], :] *= 2
    np.fill_diagonal(A, 5 * np.diag(A))
```
A lesson in Python style

Matlab and SciPy

matplotlib
"I want to use Matlab because all my data is in .mat format"
-ICME colleague

- scipy.io.loadmat() and scipy.io.savemat()
plotting

- “I like Matlab’s plotting” -Me (a while ago)

- `matplotlib.pyplot` “Provides a MATLAB-like plotting framework.”
  - http://matplotlib.org/api/pyplot_api.html
A lesson in Python style

Matlab and SciPy

matplotlib
matplotlib.org: matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB or Mathematica), web application servers, and six graphical user interface toolkits.

- matplotlib is the standard Python plotting library
- We will primarily be using matplotlib.pyplot for data analysis
- Can create histograms, power spectra, bar charts, errorcharts, scatterplots, etc with a few lines of code
Line Plot

```python
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 10, 1000)
y = np.power(x, 2)
plt.plot(x, y)
plt.savefig('line_plot.png')
```
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 10, 1000)
y = np.power(x, 2)
plt.plot(x, y)
plt.xlim((1, 5))
plt.ylim((0, 30))
plt.xlabel('my x label')
plt.ylabel('my y label')
plt.title('plot title, including $\Omega$')
plt.savefig('line_plot_plus.png')
Scatter Plot++

Adding multiple lines and a legend

```python
import numpy as np
import matplotlib.pyplot as plt

x = np.linspace(0, 10, 1000)
y1 = np.power(x, 2)
y2 = np.power(x, 3)

plt.plot(x, y1, 'b-', x, y2, 'go')
plt.xlim((1, 5))
plt.ylim((0, 30))
plt.xlabel('my x label')
plt.ylabel('my y label')
plt.title('plot title, including $\Omega$')
plt.legend((r'$x^2$', r'$x^3$'))
plt.savefig('line_plot_plus2.png')
```
Histogram
```python
import numpy as np
import matplotlib.pyplot as plt

data = np.random.randn(1000)

# histogram (pdf)
plt.subplot(1, 2, 1)
plt.hist(data, bins=30, normed=True, facecolor='b')

# empirical cdf
plt.subplot(1, 2, 2)
plt.hist(data, bins=30, normed=True, color='g', cumulative=True)

plt.savefig('histogram.png')
```
Box Plot
import numpy as np
import matplotlib.pyplot as plt

tsamp1 = np.random.normal(loc=0., scale=1., size=100)
tsamp2 = np.random.normal(loc=1., scale=2., size=100)

plt.boxplot((samp1, samp2))
plt.savefig('boxplot.png')
Scatter Plot Matrix
scatter_matrix doesn’t have everything, especially functions that are designed to act on more than one axis at once.

```python
import matplotlib.pyplot as plt
import numpy as np
from pandas.tools.plotting import scatter_matrix
from pandas import DataFrame

df = DataFrame(np.random.normal(loc=0., scale=1., size=(1000, 5)),
               columns=['a', 'b', 'c', 'd', 'e'])
scatter_matrix(df, alpha=0.4, diagonal='kde')
plt.savefig('scattermatrix.png')
```
The Debian VM has pandas installed (Python data analysis library)
import numpy as np
import matplotlib.pyplot as plt

A = np.random.random((100, 100))
plt.imshow(A)
plt.hot()
plt.colorbar()
plt.savefig('imageplot.png')
Wire Plot
matplotlib toolkits extend functionality for other kinds of visualization

```python
from mpl_toolkits.mplot3d import axes3d
import matplotlib.pyplot as plt

ax = plt.subplot(111, projection='3d')
X, Y, Z = axes3d.get_test_data(0.1)
ax.plot_wireframe(X, Y, Z)
plt.savefig('wire.png')
```
End

- Thanks for a great class!
- Please provide feedback (online material, homework, etc.)
- End-of-quarter course surveys