CME 193: Introduction to Scientific Python
Lecture 5: Data Visualization & Web Scraping

Dan Frank

Institute for Computational and Mathematical Engineering (ICME)

January 23, 2014

Administrivia

- HW 4 due next class
- No HW 5
- Austin Benson will teach guest lecture

IDEs, Debugging, Version Control, etc.

Matplotlib

Web Scraping

Debugging: General Principles

How do I make my code bug free?

1. First, don't write bugs! (e.g. informative variable/function names, encapsulation, etc.)
2. Isolate the problem: module $\Rightarrow$ function $\Rightarrow$ variable
3. Be able to consistently reproduce the problem
4. Read the error messagees
5. print statements
6. Explore the code interactively with a debugger

Debugging

I've written a function or script and it's giving me a weird error... now what?

```
import numpy as np

def some_buggy_function():
    A = np.arange(1, 10)
    # import ipdb; ipdb.set_trace() # BREAKPOINT
    A /= 2.
    return np.sum(5 / A)
```

Debugging: ipdb

What can I do within the debugger?

1. n (next)
2. ENTER (repeat previous)
3. q (quit)
4. p `variable` (print value)
5. c (continue)
6. l (list where you are)
7. s (step into subroutine)
8. r (continue till the end of the subroutine)
9. plus anything you can normally do at a python terminal

Debugging: ipdb

```
danfrank@stiletto:~$
danfrank@stiletto:~$ vim src/python-course/part-5/lecture-5/code/debug.py
danfrank@stiletto:~$ python src/python-course/part-5/lecture-5/code/debug.py
src/python-course/part-5/lecture-5/code/debug.py:7: RuntimeWarning: divide by zero encountered in divide
  return np.sum(5 / A)
danfrank@stiletto:~$ vim src/python-course/part-5/lecture-5/code/debug.py
danfrank@stiletto:~$ python src/python-course/part-5/lecture-5/code/debug.py
> /home/danfrank/src/python-course/part-5/lecture-5/code/debug.py(6)some_buggy_function()
      5     import ipdb; ipdb.set_trace() # BREAKPOINT
----> 6     A /= 2.
      7     return np.sum(5 / A)

ipdb> n
> /home/danfrank/src/python-course/part-5/lecture-5/code/debug.py(7)some_buggy_function()
      6     A /= 2.
----> 7     return np.sum(5 / A)
      8

ipdb> A
array([0, 1, 1, 2, 2, 3, 3, 4, 4])
ipdb> A.dtype
dtype('int64')
ipdb> l
      2
      3 def some_buggy_function():
      4     A = np.arange(1, 10)
      5     import ipdb; ipdb.set_trace() # BREAKPOINT
      6     A /= 2.
----> 7     return np.sum(5 / A)
      8
      9 some_buggy_function()

ipdb>
```

IDEs

Not all software engineers use them but Integrated Development Environments can be very helpful in developing code. Functionality may include

- source code editor with syntax highlighting
- autocompletion and goto definitions
- debugging
- integration with terminal

No standard IDE for Python, but eclipse, emacs, vim, etc. can be configured. Each coder has their own setup. iPython provides many of these functions within a terminal.

Version Control

Version control enables different people working on the same code base to coordinate their efforts.

- maintain history of code development
- tracking differences between current versions and older versions
- create branches of code so different features can be worked on simultaneously and merged together later

There are many tools for version control but the standards are **subversion** and more recently **git**. This course is currently coordinated through `github.com`

IDEs, Debugging, Version Control, etc.

Matplotlib

Web Scraping

What is Matplotlib?

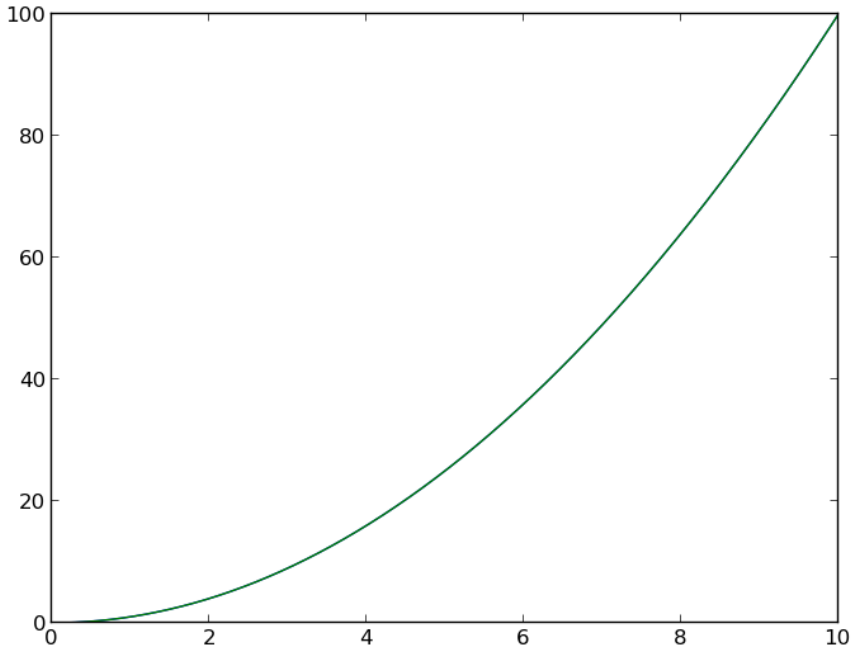
matplotlib.org: matplotlib is a python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. matplotlib can be used in python scripts, the python and ipython shell (ala MATLAB or Mathematica), web application servers,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

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

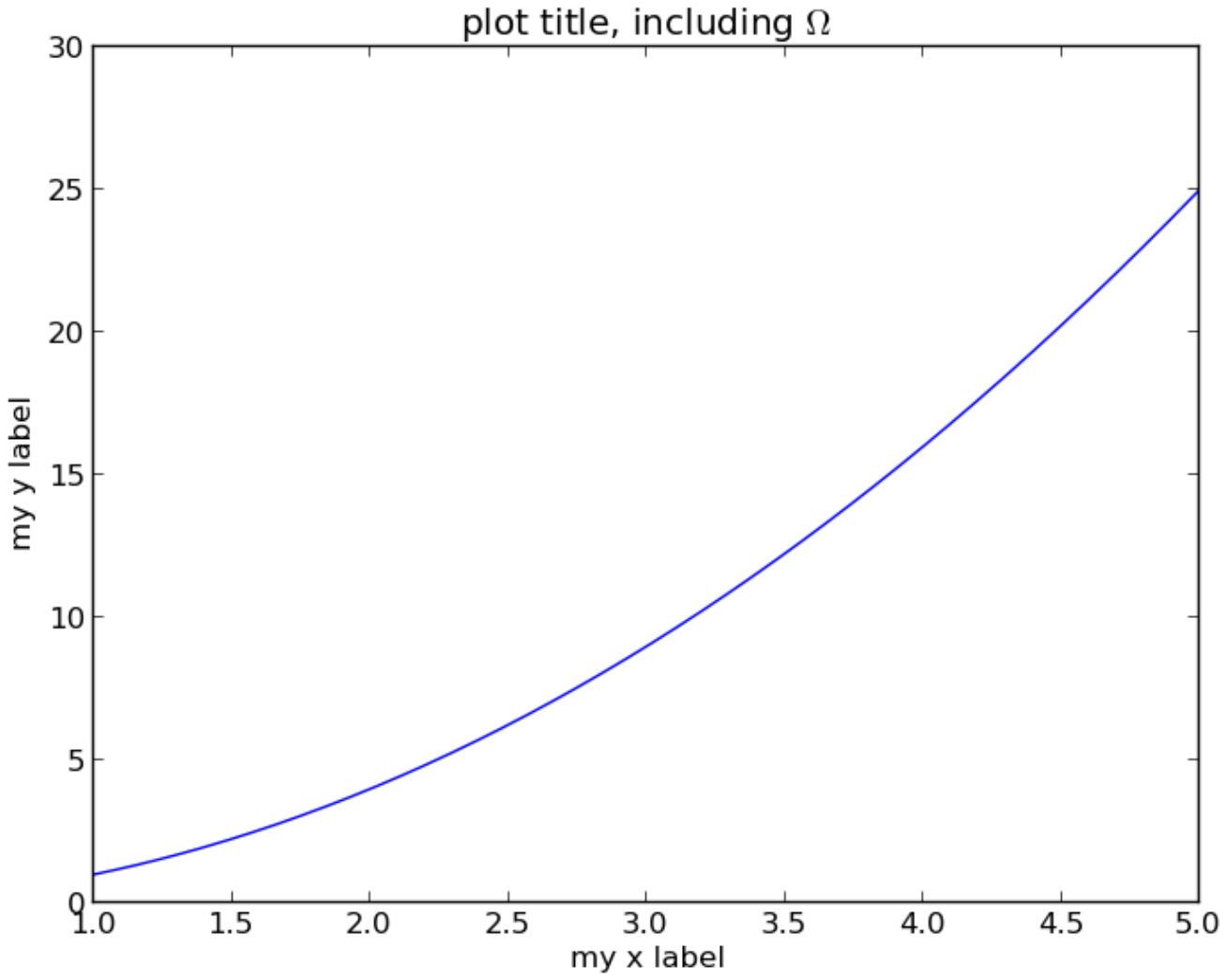
Scatter Plot

```
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')
```



Scatter Plot+



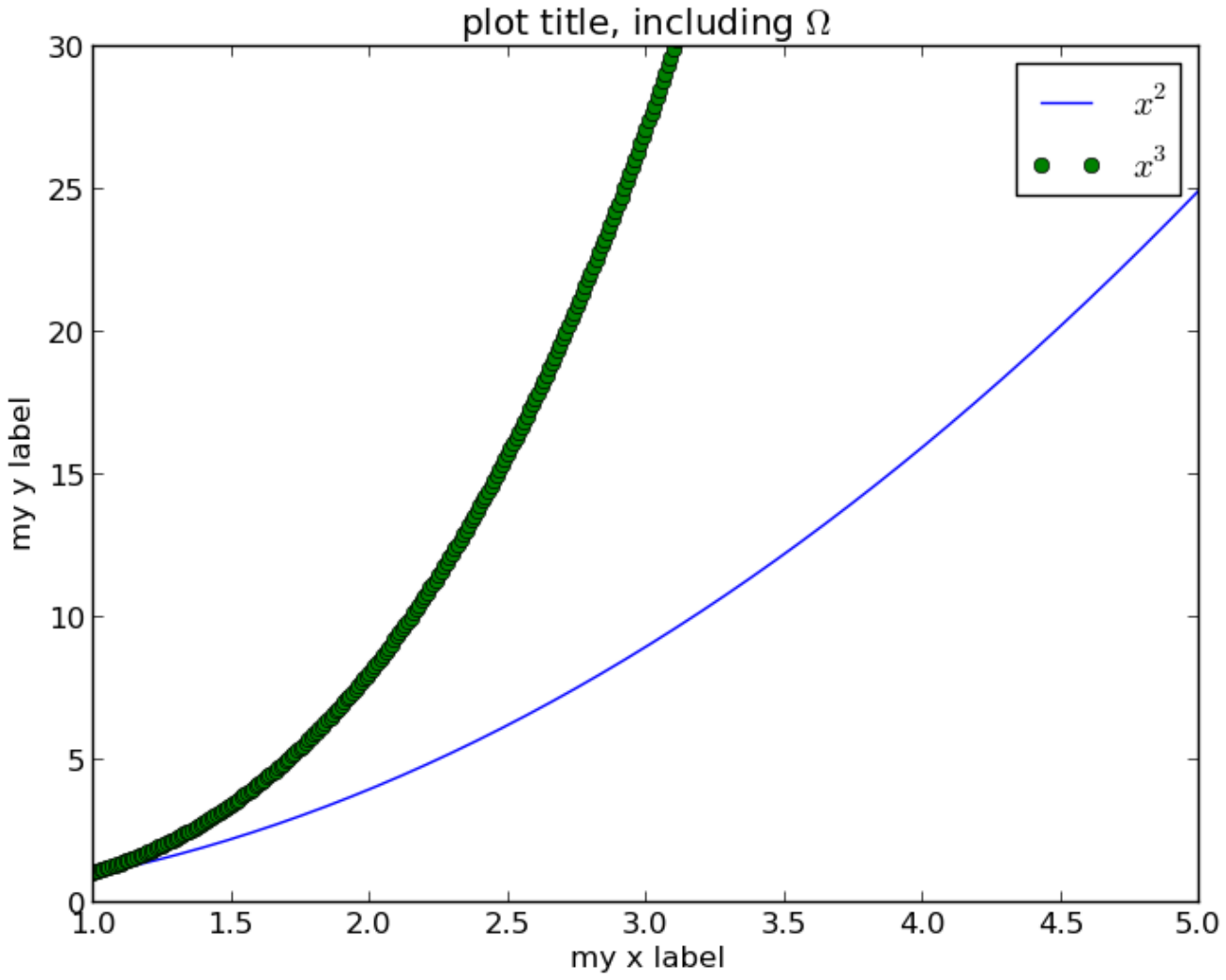
Scatter Plot+

Adding titles and labels

```
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++



Scatter Plot++

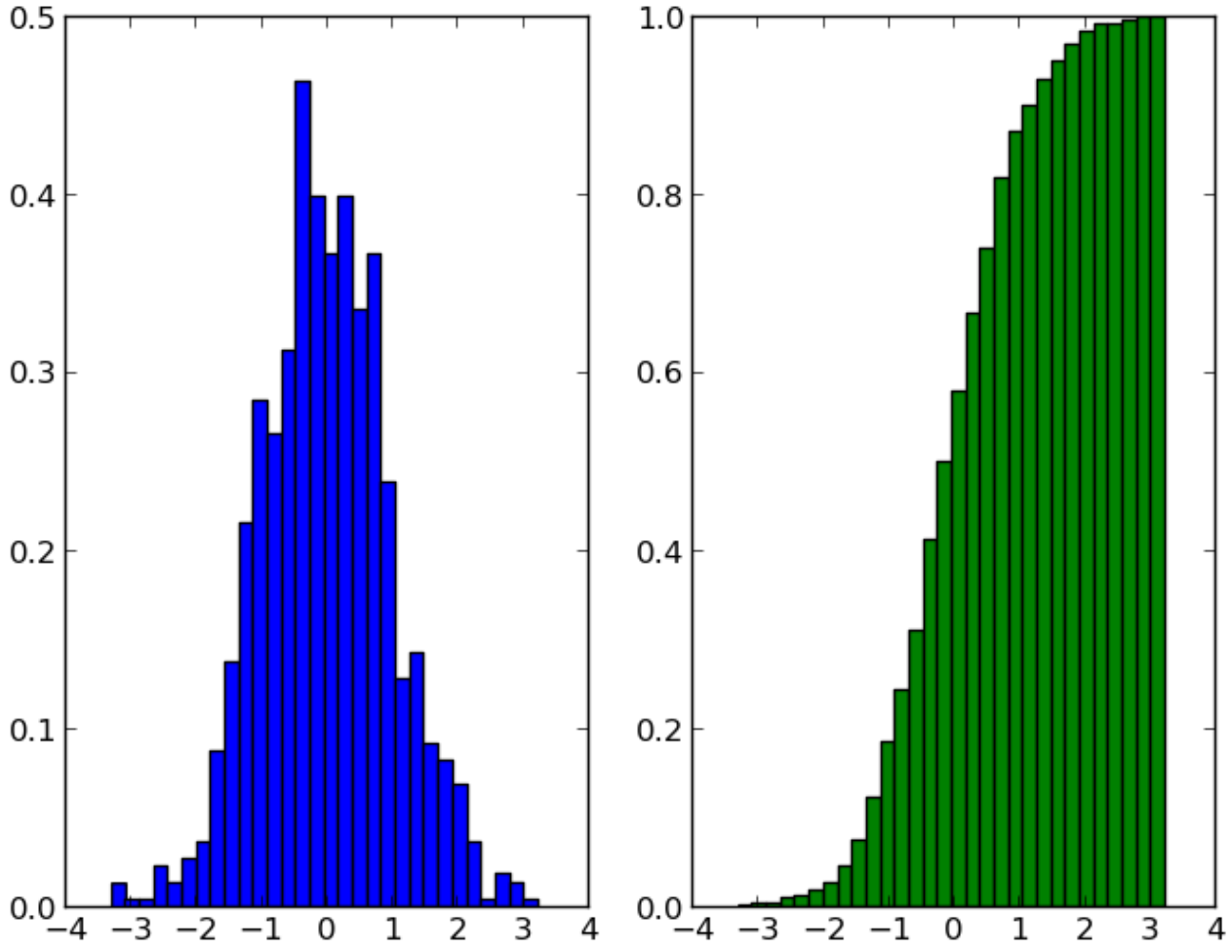
Adding multiple lines and a legend

```
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(('$x^2$', '$x^3$'))

plt.savefig('line_plot_plus2.png')
```

Histogram



Histogram

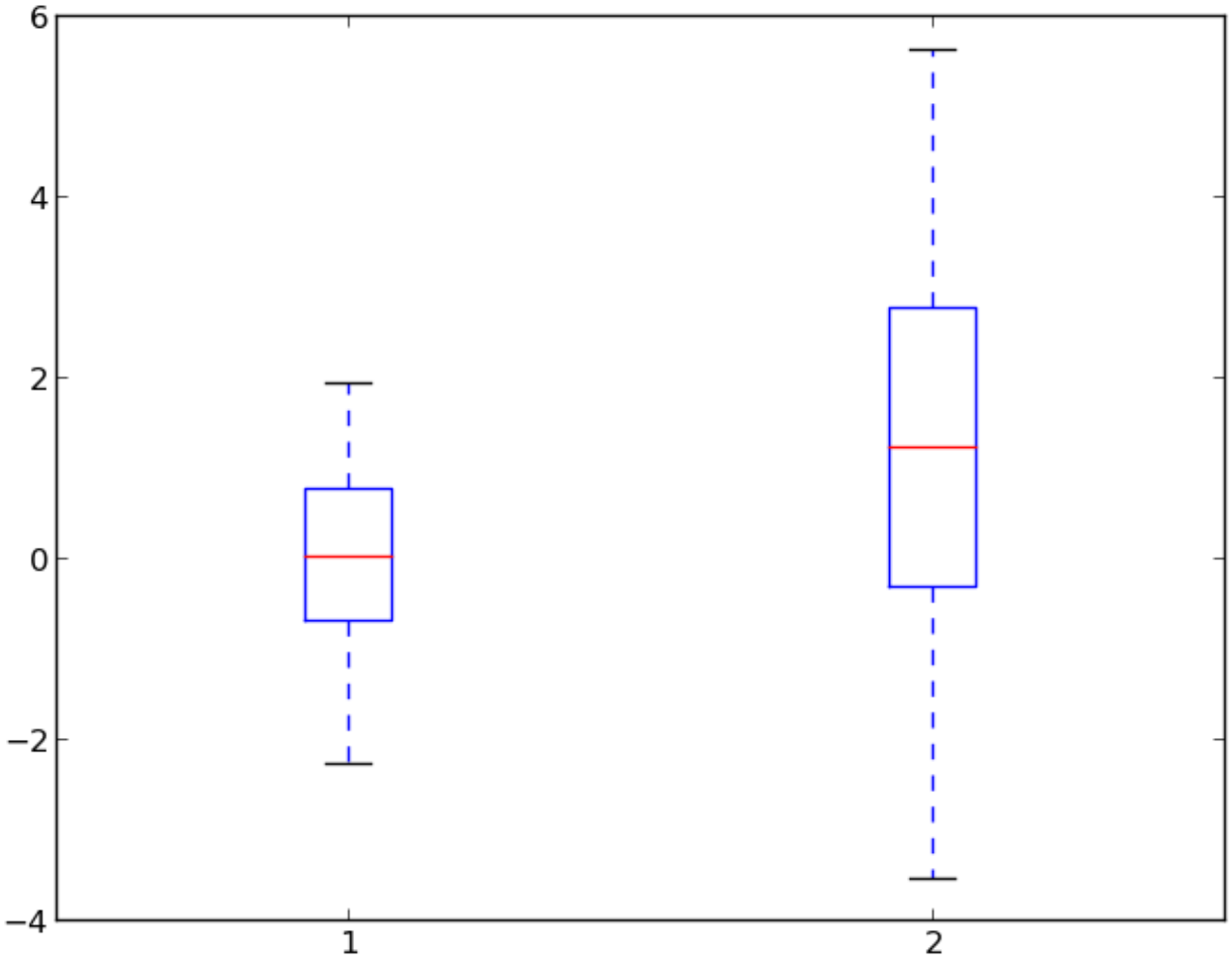
```
data = np.random.randn(1000)

# 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

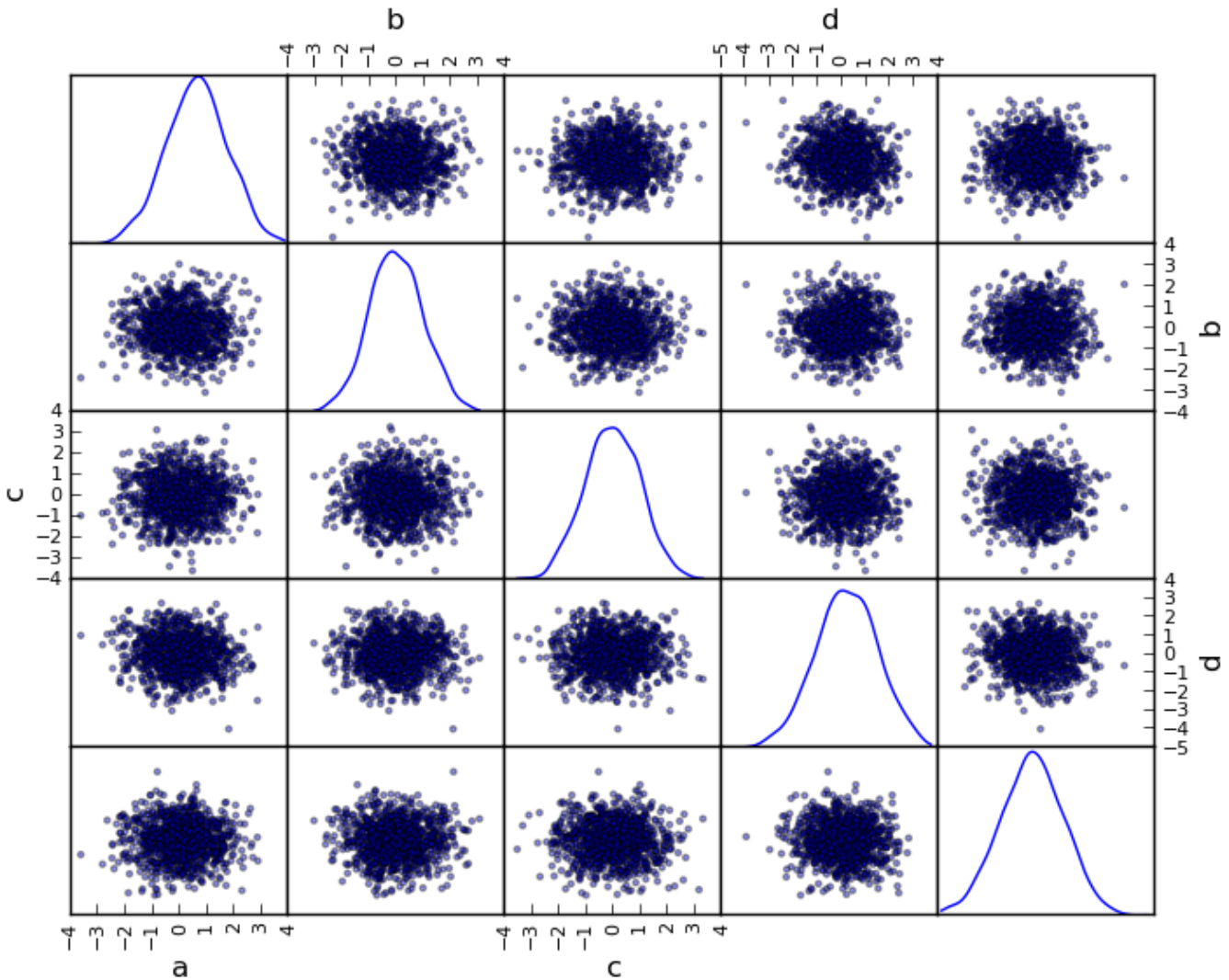


Box Plot

```
samp1 = np.random.normal(loc=0., scale=1., size=100)
samp2 = np.random.normal(loc=1., scale=2., size=100)

plt.boxplot((samp1, samp2))
plt.savefig('boxplot.png')
```

Scatter Plot Matrix



Scatter Plot Matrix

`matplotlib` doesn't have everything, especially functions that are designed to act on more than one axis at once.

```
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')
```

Image Plot

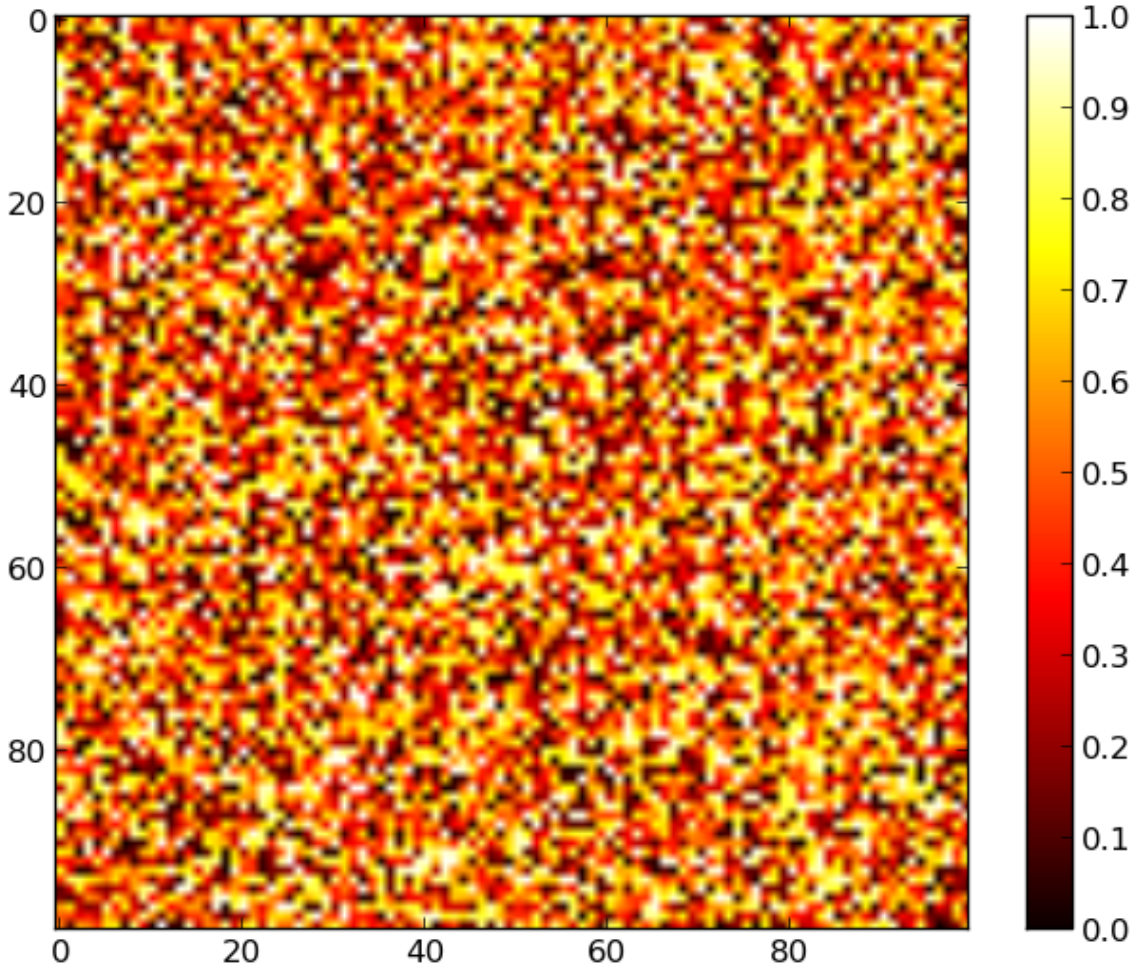
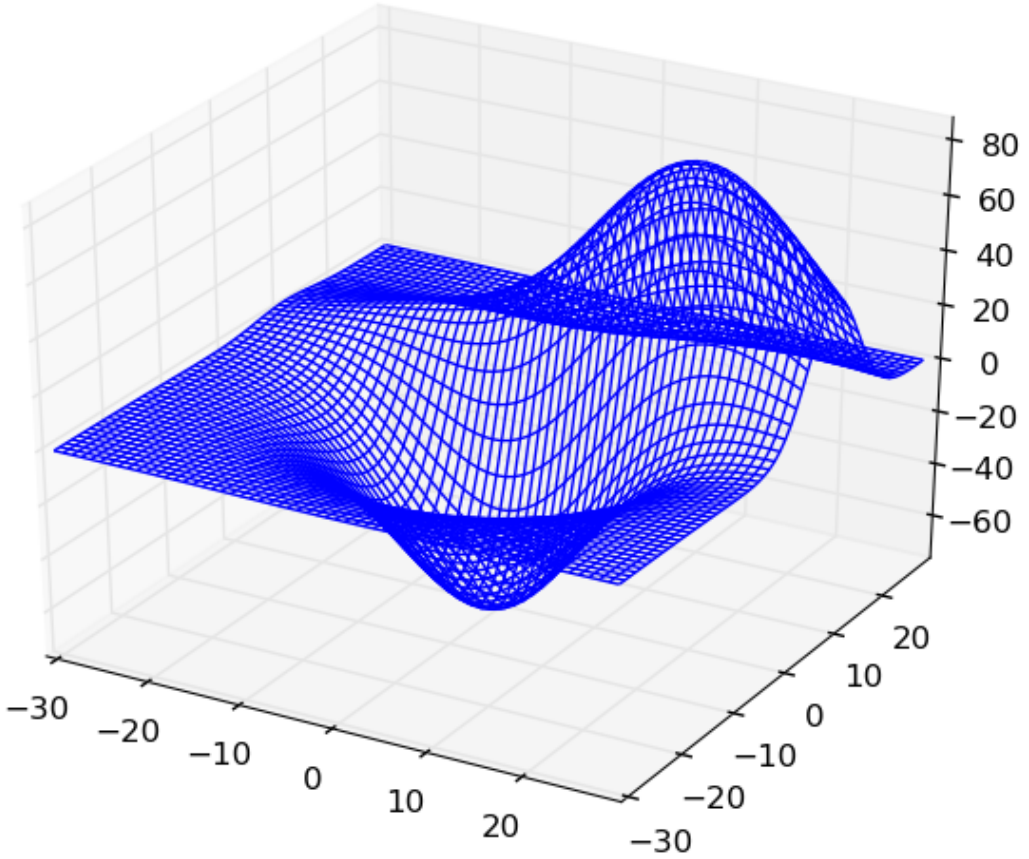


Image Plot

```
A = np.random.random((100, 100))
plt.imshow(A)
plt.hot()
plt.colorbar()

plt.savefig('imageplot.png')
```

Wire Plot



Wire Plot

matplotlib toolkits extend funtionality for other kinds of visualization

```
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')
```

IDEs, Debugging, Version Control, etc.

Matplotlib

Web Scraping

Web Scraping Ingredients

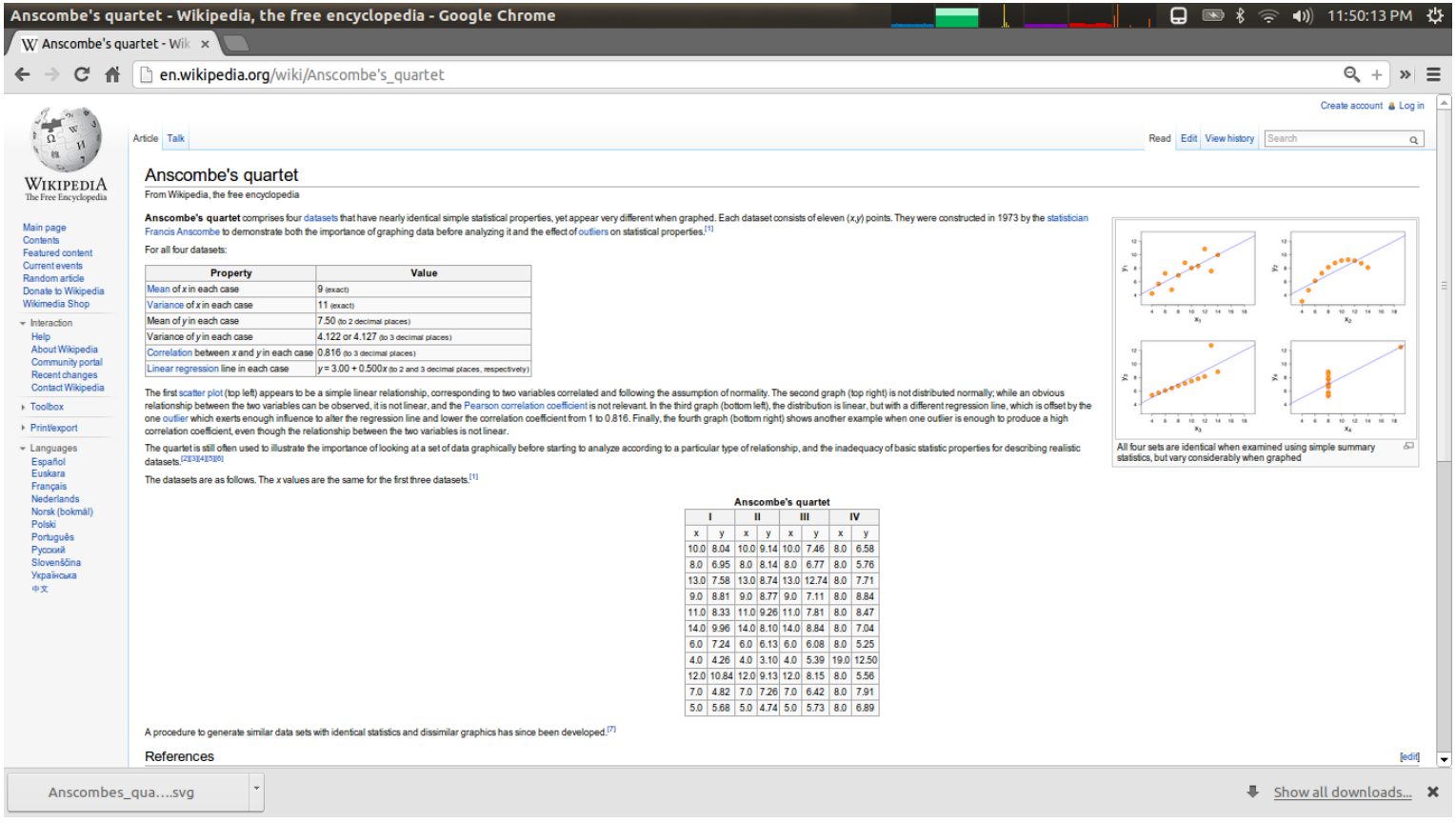
Webscraping HTML involves...

- visual inspection - `Chrome Developer Mode & Firebug`
- browser sessions and interacting with HTML - `mechanize`
- HTML parsing/searching - `BeautifulSoup`

  warning: javascript makes things tricky... check out *selenium* if you need to interact with javascript

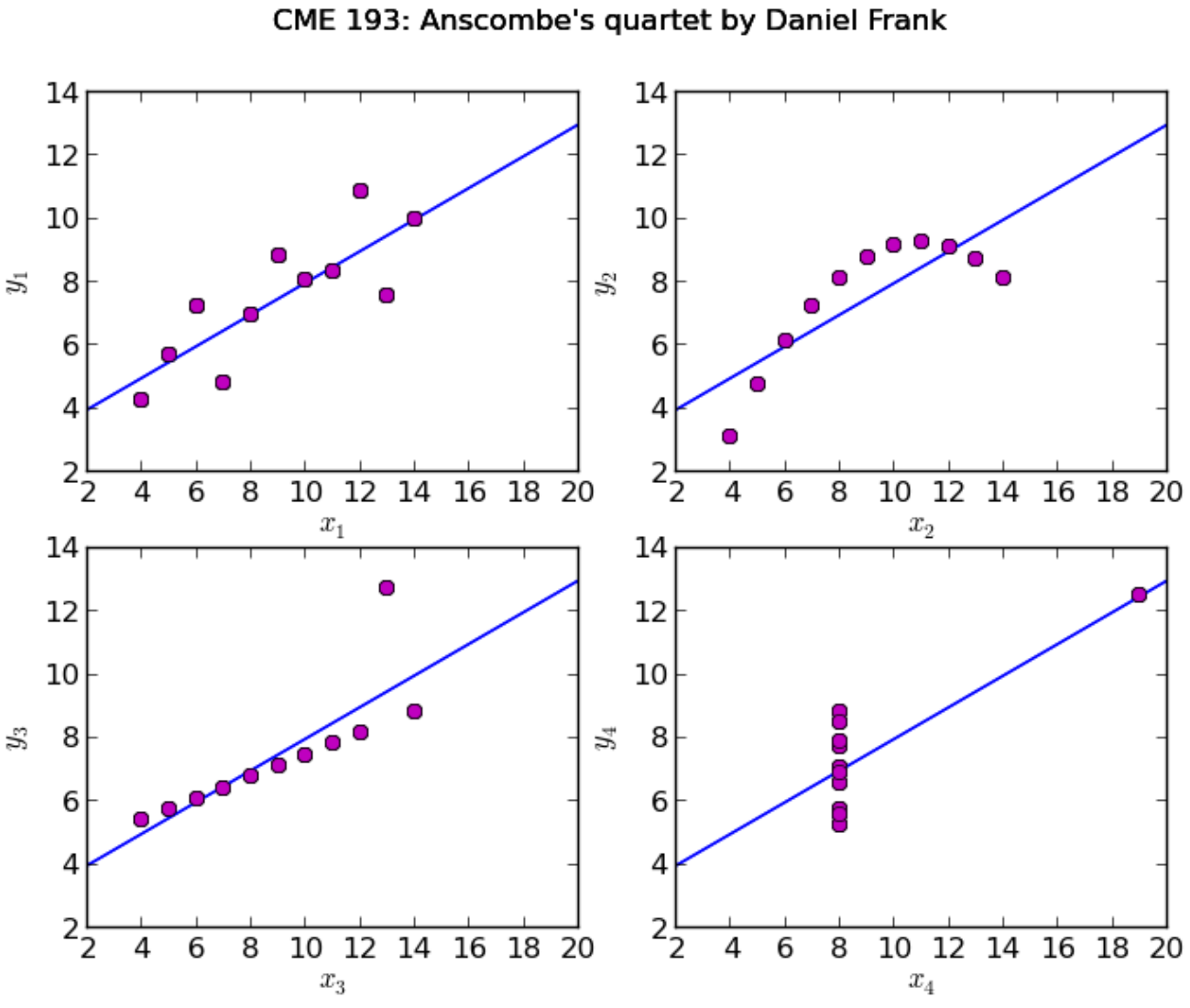
Webscraping Example

Want to verify that wikipedia's plot



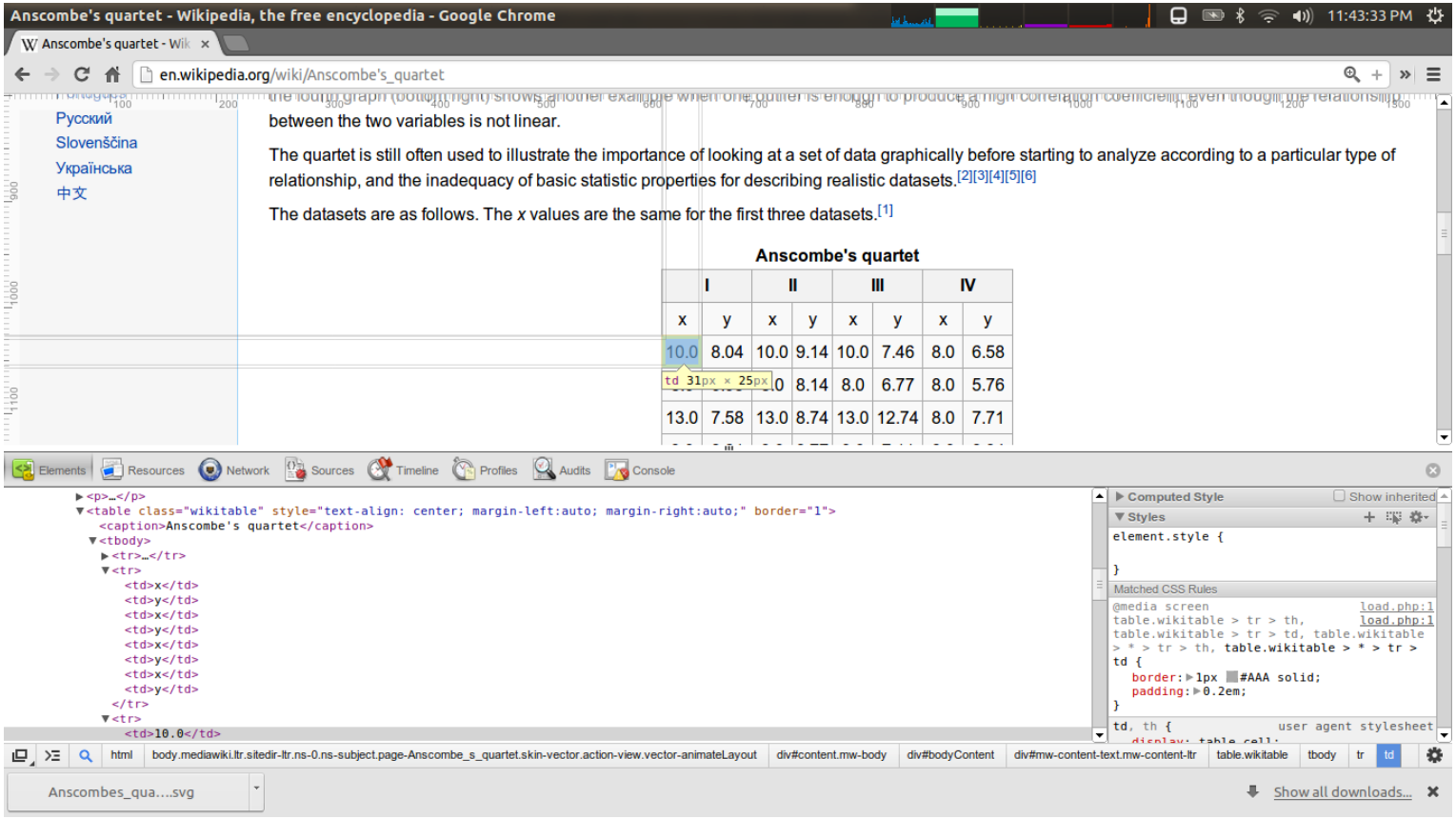
Webscraping Example

Recreate the plot ourselves by scraping the data and plotting



Webscraping Example: Visual Inspection

In Chrome Developer Mode we can use 'inspect element' to look at the HTML associated with the table we're interested in.



Webscraping Example: Visual Inspection

```
# missing imports and bad multilines!
br = mechanize.Browser()
br.addheaders = [('User-agent', 'Mozilla/5.0 (Macintosh; U; ' +
    ' Intel Mac OS X 10_6; en-us) ' +
    ' AppleWebKit/531.9 (KHTML, like Gecko) Version/4.0.3 Safari/531.9')]
soup = BeautifulSoup(br.open("http://en.wikipedia.org/wiki/Anscombe's_quartet").read())

tbl = soup.find(lambda tag: (tag.name == 'caption') and \
                            (tag.text == "Anscombe's quartet")).parent
arr_list = []
for row in tbl.findAll('tr'):
    elements = row.findAll('td')
    if len(elements) != 0:
        try:
            np.float(elements[0].string)
        except:
            continue
        arr_list.append(np.array([np.float(e.string) for e in elements]))
data = np.vstack(arr_list)

grid = np.linspace(2, 20, 100)
for i in xrange(4):
    x = data[:, 2 * i]; y = data[:, 2 * i + 1]
    a, b = scipy.polyfit(x, y, 1)
    plt.subplot(2, 2, i + 1)
    plt.plot(grid, a * grid + b, 'b-', x, y, 'mo')
    plt.xlabel("$x_" + str(i + 1) + "$"); plt.ylabel("$y_" + str(i + 1) + "$")
    plt.xlim((2, 20)); plt.ylim((2, 14))

plt.suptitle("CME 193: Anscombe's quartet by Daniel Frank")
```