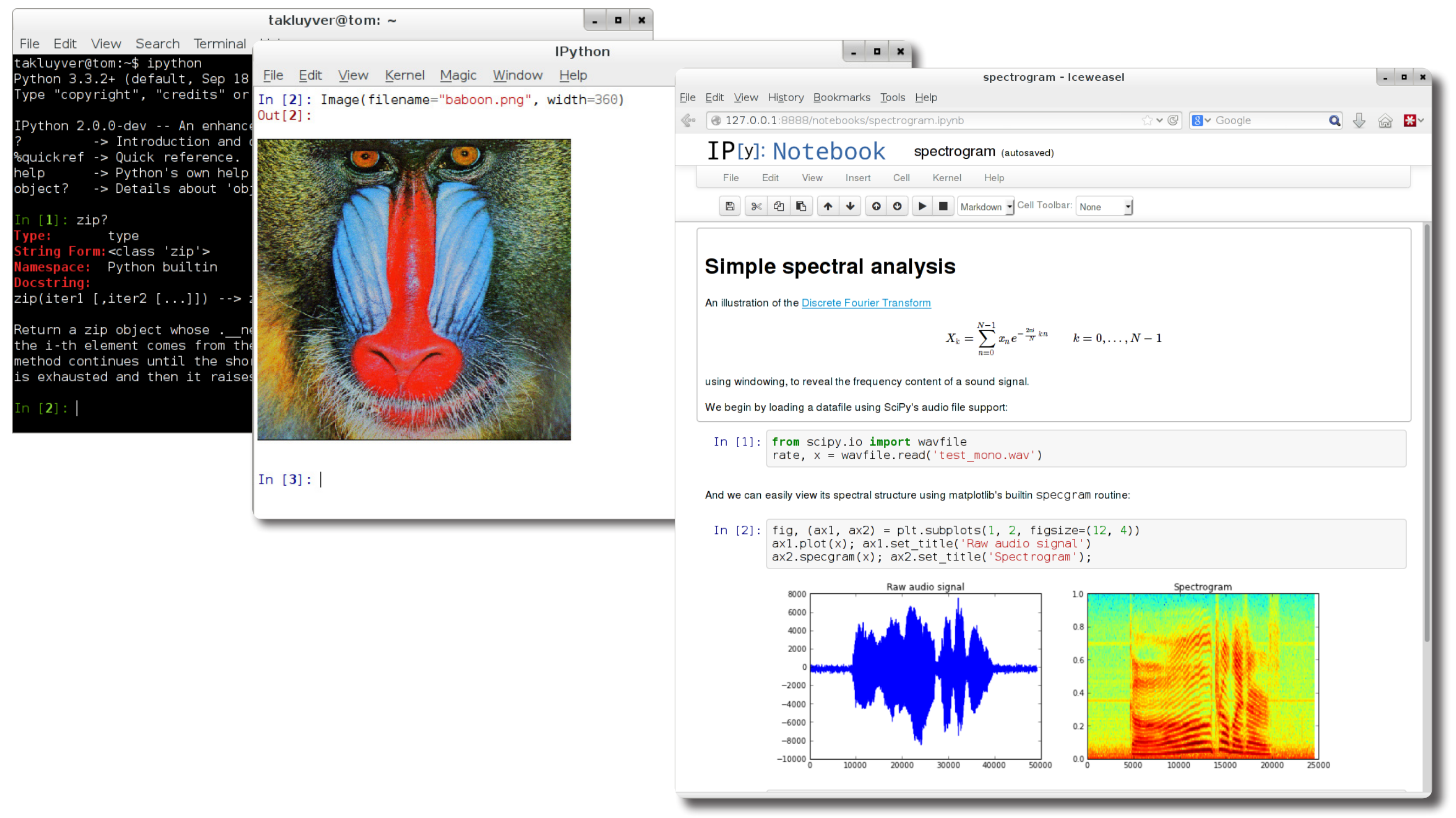


```
takluyver@tom:~$ ipython
Python 3.3.2+ (default, Sep 18
Type "copyright", "credits" or

IPython 2.0.0-dev -- An enhance
?         -> Introduction and 
%quickref -> Quick reference.
help      -> Python's own help
object?   -> Details about 'obj

In [1]: zip?
Type:        type
String Form:<class 'zip'>
Namespace:   Python builtin
Docstring:
zip(iter1 [,iter2 [...]]) --> z

Return a zip object whose .__ne
the i-th element comes from the
method continues until the shor
is exhausted and then it raises

In [2]: |
```

```
In [2]: Image(filename="baboon.png", width=360)
Out[2]:

In [3]: |
```

127.0.0.1:8888/notebooks/spectrogram.ipynb

IP[y]: Notebook spectrogram (autosaved)

Simple spectral analysis

An illustration of the Discrete Fourier Transform

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{2\pi i}{N} kn} \qquad k = 0, \ldots, N-1$$

using windowing, to reveal the frequency content of a sound signal.

We begin by loading a datafile using SciPy's audio file support:

```
In [1]: from scipy.io import wavfile
        rate, x = wavfile.read('test_mono.wav')
```

And we can easily view its spectral structure using matplotlib's builtin specgram routine:

```
In [2]: fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(12, 4))
        ax1.plot(x); ax1.set_title('Raw audio signal')
        ax2.specgram(x); ax2.set_title('Spectrogram');
```

takluyver@tom: ~
File Edit View Search Terminal
IPython
File Edit View Kernel Magic Window Help
spectrogram - Iceweasel
File Edit View History Bookmarks Tools Help
Google
File Edit View Insert Cell Kernel Help
Markdown
Cell Toolbar: None
Raw audio signal
Spectrogram

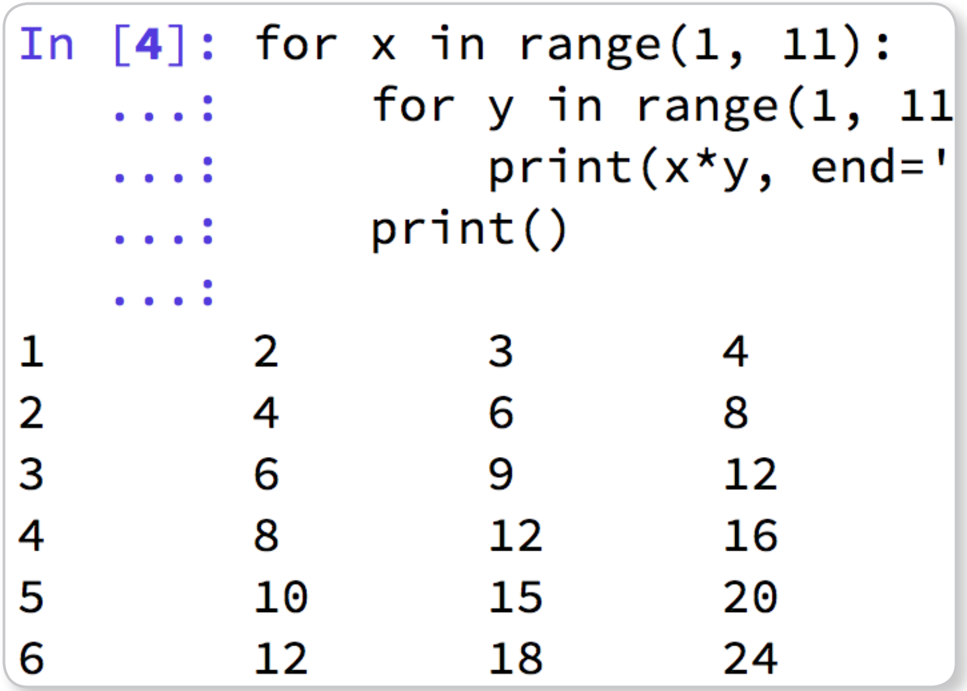
IP[y]: IPython

Interactive Computing

What is IPython?

IPython a collection of tools for interactive computing, including:

- A browser-based notebook with support for code, text, mathematical expressions, inline plots and other rich media.
- Powerful interactive consoles.
- Easy to use, high performance tools for interactive parallel computing.



```
In [4]: for x in range(1, 11):
   ...:     for y in range(1, 11
   ...:         print(x*y, end='
   ...:     print()
   ...:
1       2       3       4
2       4       6       8
3       6       9       12
4       8       12      16
5       10      15      20
6       12      18      24
```

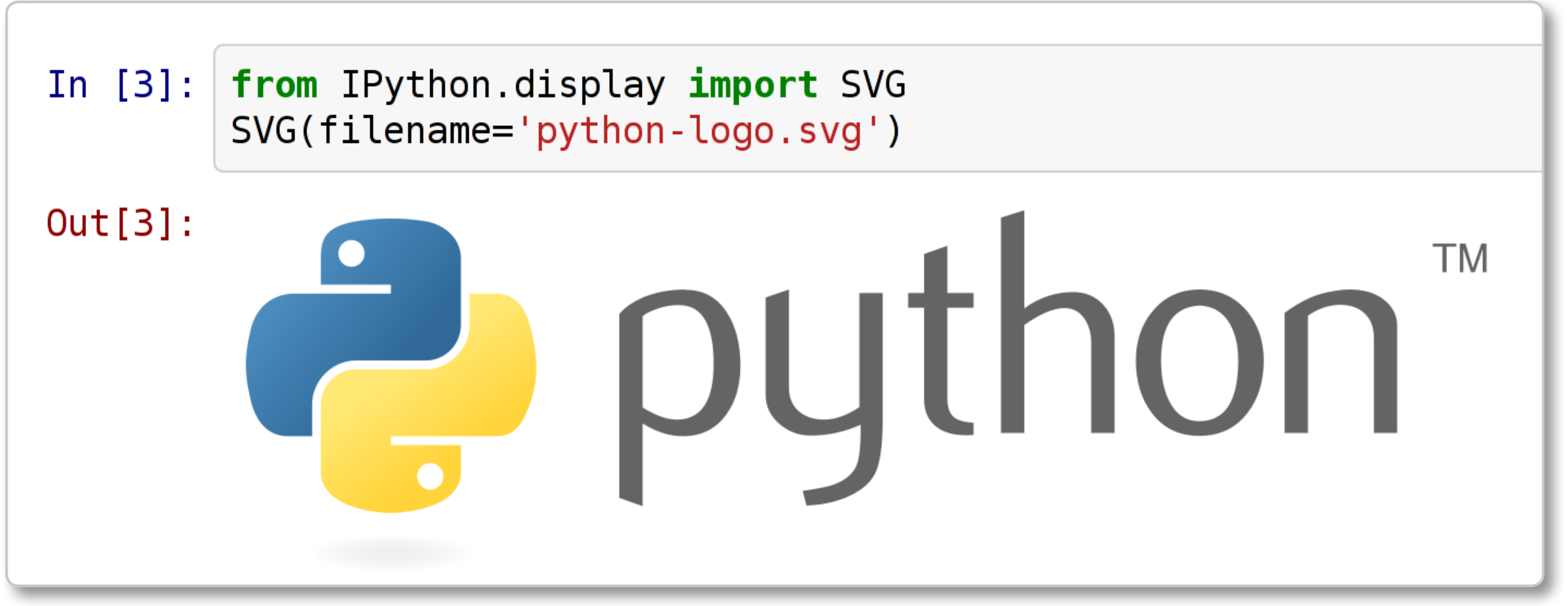
Despite the name, ***IPython is not just for Python***. The cell magics system lets you mix Python code with other languages, including Cython, R and Octave, in one document. Because IPython uses a language agnostic protocol, people have also created backends for languages including Julia, Ruby, and Haskell, which let you create entire notebooks in those languages.

julia

IPython Notebook

The IPython Notebook is the centerpiece of IPython. It lets you save your code in a document along with results, graphs, and rich annotations. These documents can be shared with others and support workflows for **collaborative** and **reproducible** computational research. With the Notebook, you can

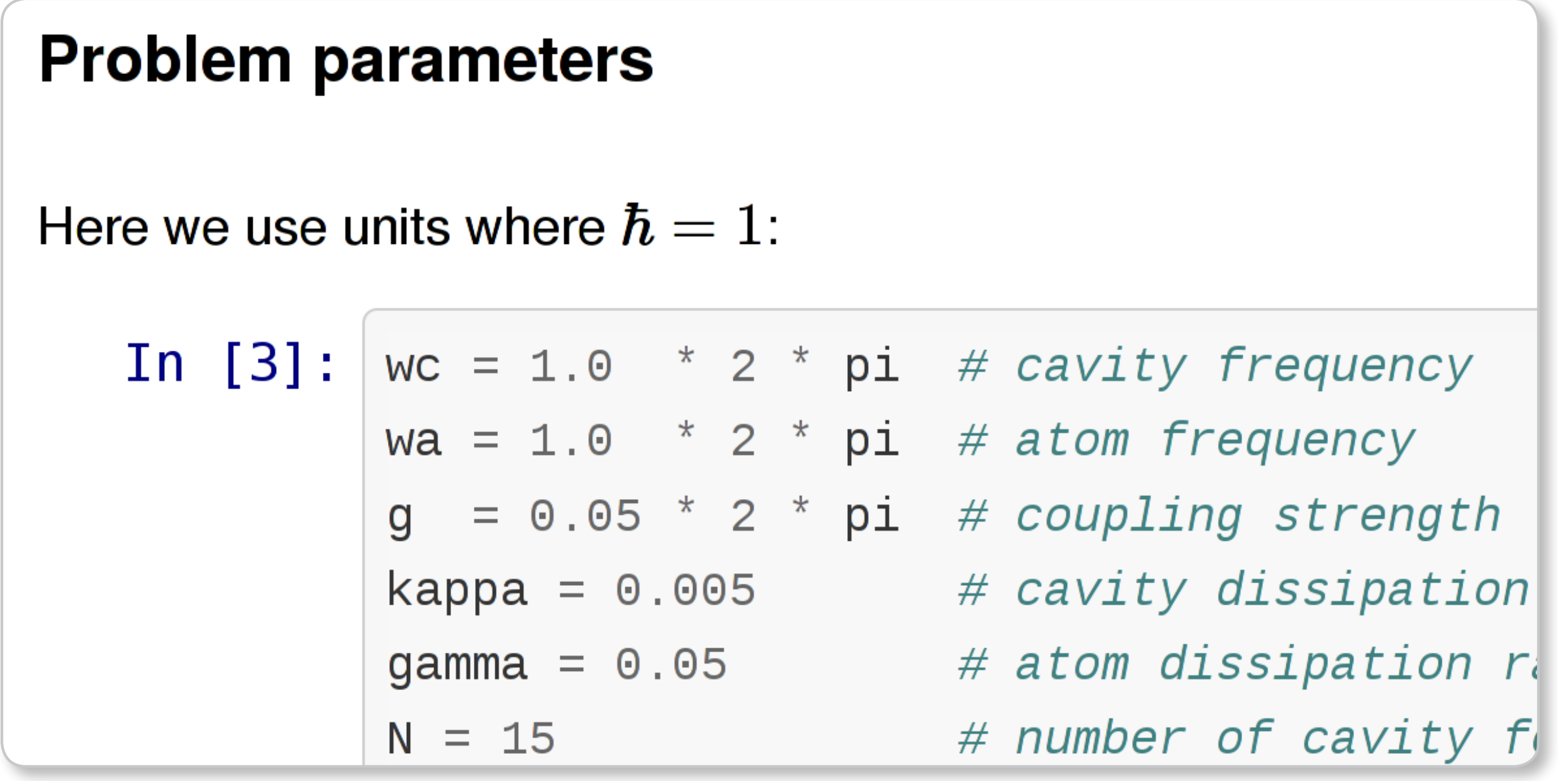
- Include rich output, such as images, graphs and videos:



```
In [3]: from IPython.display import SVG
        SVG(filename='python-logo.svg')

Out[3]:
```

- Describe and explain your code:

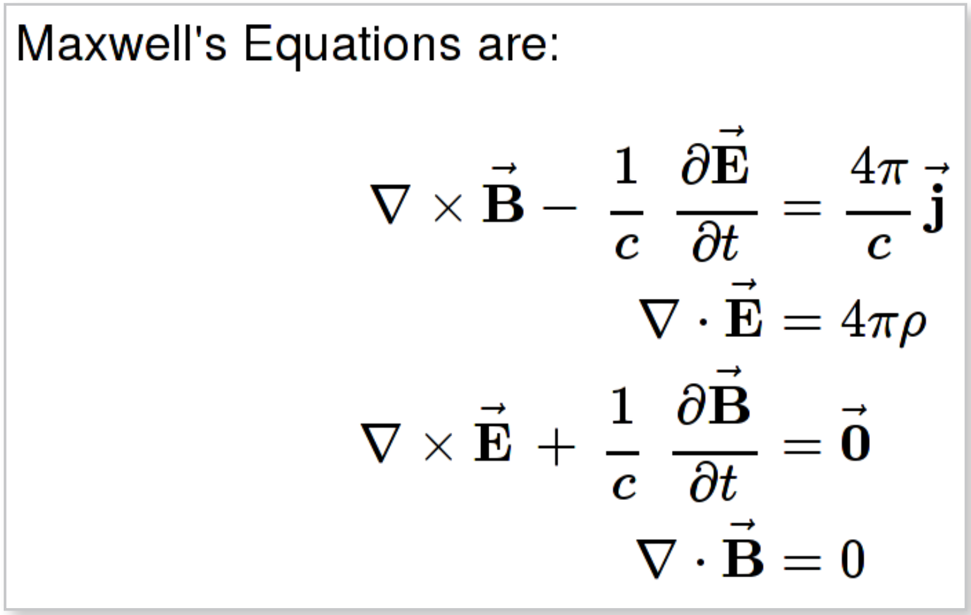


Problem parameters

Here we use units where $\hbar = 1$:

```
In [3]: wc = 1.0  * 2 * pi  # cavity frequency
        wa = 1.0  * 2 * pi  # atom frequency
        g  = 0.05 * 2 * pi  # coupling strength
        kappa = 0.005       # cavity dissipation
        gamma = 0.05        # atom dissipation r
        N = 15              # number of cavity f
```

- Include mathematical expressions:



Maxwell's Equations are:

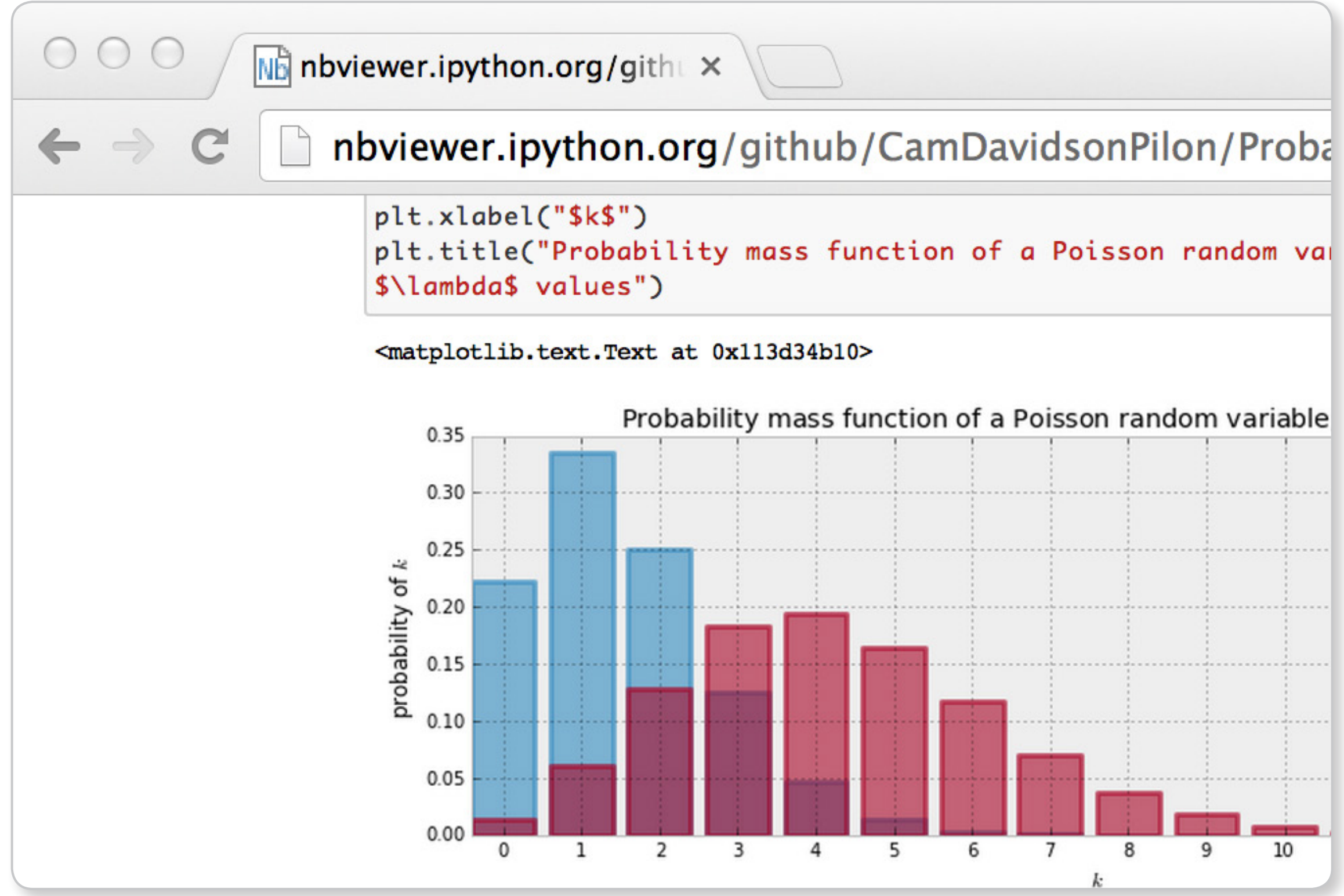
$$\begin{aligned} \nabla \times \vec{\mathbf{B}} - \frac{1}{c}\frac{\partial \vec{\mathbf{E}}}{\partial t} &= \frac{4\pi}{c}\vec{\mathbf{j}} \\ \nabla \cdot \vec{\mathbf{E}} &= 4\pi\rho \\ \nabla \times \vec{\mathbf{E}} + \frac{1}{c}\frac{\partial \vec{\mathbf{B}}}{\partial t} &= \vec{\mathbf{0}} \\ \nabla \cdot \vec{\mathbf{B}} &= 0 \end{aligned}$$

Use Cases for IPython

Sharing Analyses

Catherine has written some code to prepare and analyse her data. Now she wants her collaborator, Wes, to read it and comment on it, but his computer is not set up to run her code. She converts her notebook to HTML, which Wes can view in any browser. This includes the code, notes she has written about what it does, and results in the form of graphs and tables. By installing IPython, Wes can also modify and rerun the code.

If the analysis is public, they can share the notebook files on the web, using the public NBViewer service to show people the code and results.



nbviewer.ipython.org/github/CamDavidsonPilon/Proba

```
plt.xlabel("$k$")
plt.title("Probability mass function of a Poisson random va
$\lambda$ values")
```

```
<matplotlib.text.Text at 0x113d34b10>
```

nbviewer.ipython.org/githu
Probability mass function of a Poisson random variable
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
probability of k
0
1
2
3
4
5
6
7
8
9
10
k

Cloud Computing

Because the IPython notebook is an HTML interface inside your browser, you can easily run it on remote servers such as Amazon Web Services or Windows Azure. Projects such as Starcluster make it easy to set up IPython on cloud servers.

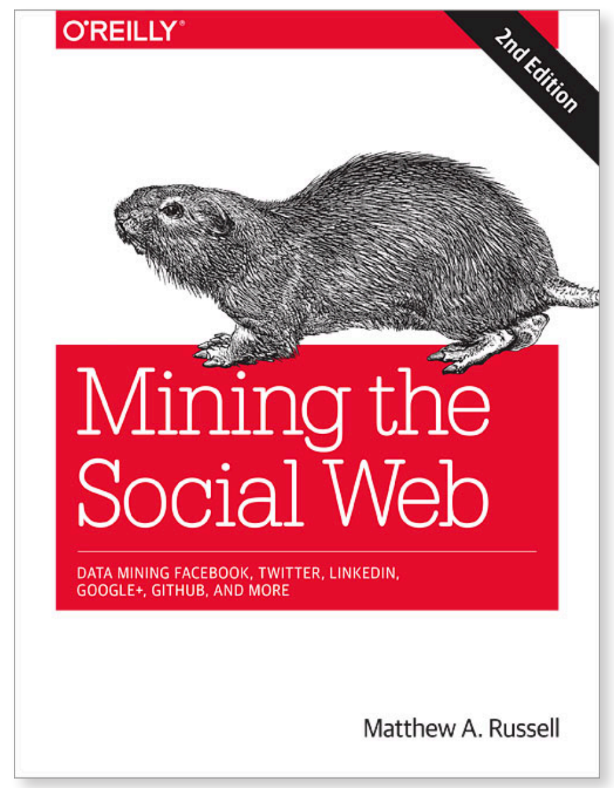
Executable Documents

Joe is giving a lab meeting talk. He prepares presentation 'slides' as IPython Notebooks, interleaving narrative text, images, LaTeX formulae, and code.

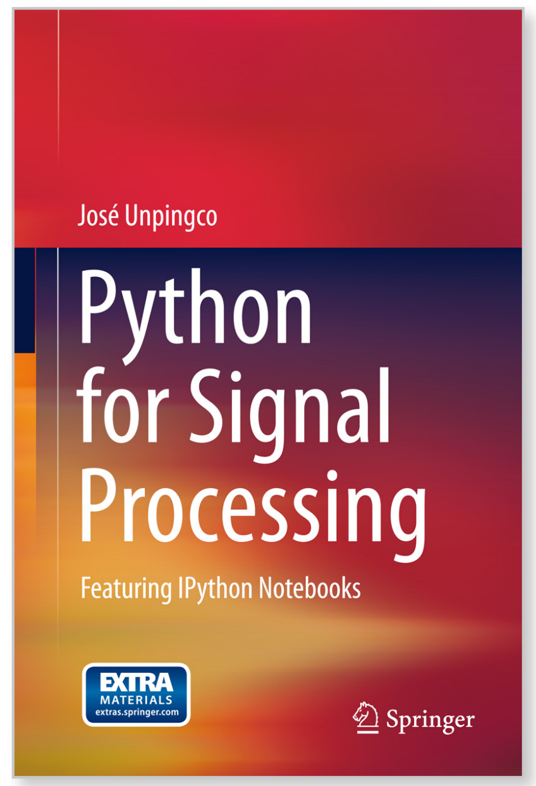
He prepares code cells to load up some data and perform some analysis. Joe can execute the code cells, getting the results in-line in the same environment that he's writing the rest of the talk materials. One cell plots a subset of the data, and Joe can tweak plotting parameters such as line width and line color, and re-execute until he's happy with its appearance.

Narrative cells among the code cells describe the method used, provide a rendered formula for the code implementation, suggest how parameters might be altered, and provide clickable links to the paper which first introduced the technique. Joe and his advisor expand the work into a full manuscript, with the notebook as the computational companion to the paper.

This practice has already been further extended to the scale of whole books:



O'REILLY
2nd Edition
Mining the Social Web
DATA MINING FACEBOOK, TWITTER, LINKEDIN, GOOGLE+, GITHUB, AND MORE
Matthew A. Russell



José Unpingco
Python for Signal Processing
Featuring IPython Notebooks
EXTRA MATERIALS extras.springer.com
Springer

For more information, go to ipython.org, or find us on GitHub at github.com/ipython