IPython Notebooks

By

Tobie Nortje

A talk at PyCon ZA

October 2013

Cape Town

South-Afrca

IPython Notebooks and the Python Scientific Stack

IP[y]: IPython Interactive Computing

http://ipython.org/static/IPyheader.png
[IPython](#) in action creating reproducible and publishable interactive work.

What is this?

This `repo` contains the full [talk](#) I intend to deliver (have delivered) at [PyConZA2013](#). It contains all the files needed to build a final publishable PDF document from an interactive notebook and even adds a custom front page.

[The Complete Talk GitHub Website can be accessed here](#)

Background

IPython had become a popular choice for doing interactive scientific work. It extends the standard Python interpreter and adds many useful new futures. There is really no need to use the standard Python interpreter anymore. In addition to this IPython offers a web based Notebook that makes interactive work much easier, and have been used to write repeatable scientific papers and more recently a book has been written using this platform, the online Notebook Viewer and GitHub. The development of this material and tool chain to compile the notebook to a publishable PDF, has inspired me to maybe even try and turn this into a complete (free) book. Let's see what happens.

Combining the most common scientific packages with IPython makes it a formidable tool and serious competition to R. (R is still awesome!)
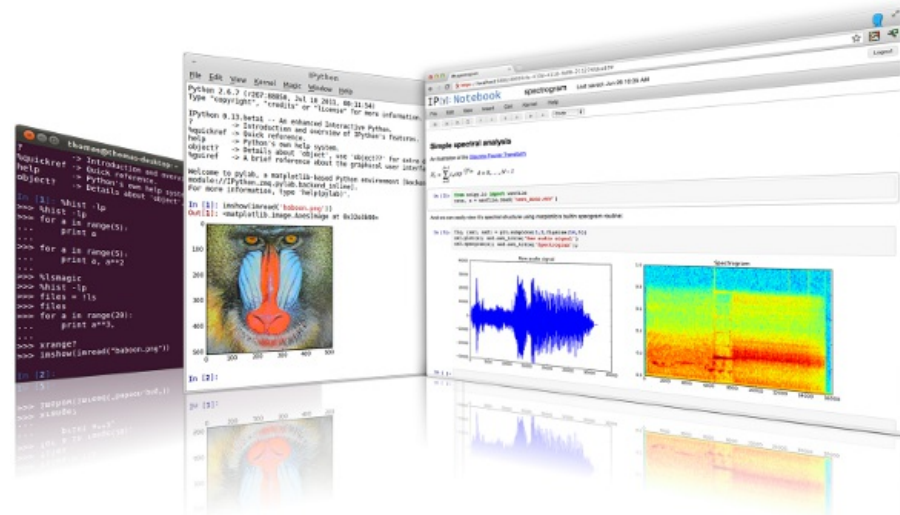
http://ipython.org/static/ipy0.13.png
As a matter of fact you can run R in the notebook session, embed YouTube Videos, Images and lots more but let me not get ahead of myself....

The science stack consists of (but not limited to):

package	description
[pandas](#)	`dataframe` implementation (based on numpy)
[scipy](#)	efficient numerical routines
[sympy](#)	`symbolic mathematics`
[matplotlib](#)	python standard `plotting` package

[sci-kit learn](#)	machine learning and `well documented!`

Talk contents

The talk will aim to introduce these tools and explore some practical interactive examples. Once completed it will be shown how easy it is to publish your work to various formats. Some of the topics covered in the talk are listed below:

item	description
ipython	quick intro to ipython and the notebook
setup	set up your environment / get the talk files
notebook basics	navigate the notebook
notebook magic's	special notebook commands that can be very useful
getting input	as from IPython 1.00 getting input from sdtin is possible
local files	how to link to local files in the notebook directory
plotting	how to create beautiful inline plots
symbolic math	quick demo of sympy model
pandas	quick intro to pandas dataframe
typsetting	include markdown, Latex via MathJax
loading code	how to load a remote .py code file
gist	paste some of your work to gist for sharing
js	some javascript examples
customising	loading a customer css and custom matplotlib config file
git cell	add code to a special cell that would commit to git
output formats	how to publish your work to html, pdf or jeveal.js presentation

Get the processed presentation files here:

format	description
[IPython notebook](#)	.ipynb file to run in browser
[IPython html notebook](#)	converted to HTML and served online
[IPython pdf notebook](#)	converted to PDF for download (to be added, needs pandoc)
[IPython pdf book](#)	converted to pdf and a front-page stitched to it)
[Ipython reveal.js presentation](#)	converted to a reveal.js presentation and served online
[Online IPython NBveiwer](#)	view on the ipython notebook viewer

Dependencies

I was given the challenge to develop all of this on a Windows machine as some of my sponsors want to demonstrate that this stuff can not only be done on GNU/Linux/OSX. So all the tool chains are Windows based. If you know Linux, then you are the type of person that would easily port this. That being said the Windows GitHub client is refreshing.

package	description
IPython	To use NBConvert you need V1.00. If you only want to use the interactive notebook then v0.13 will be ok.

pandoc	The document converter used by IPythonr
MikeTex	If you want to do a TEX to PDF transform. I had so many issues with the TEX to PDF conversion by NBConvert, so settled for wkhtmltopdf(below) to convert HTML to PDF rather. (Convert notebook to HTML with NBconvert and then from HTML to PDF with wkhtmltopdf
wkhtmltopdf	Convert HTML to PDF
pdftk	Can be used to combine PDF's. In this case add a frontpage to the generated IPython notebook PDF.
ImageMagick	for compressing the PDF. Still experimenting with this.
GhostScript	needed by ImageMagick
anaconda	install anaconda from Continuum Analytics. Almost all the Python packages are included and it has a virtual environment manager via it's console application `conda'

How to run the Interactive Notebook

Navigate to the `src` directory and run from the command line:

`python        ipython notebook`

If everything works your browser should open and you can select the `notebook` and start experimenting!

PDF, HTML, Slideshow Build Script

There is a build script in the `src` directory. It is an IPython file. You can basically build shell scripts this way. To use the power of IPython commands save the file with the `.ipy` extension and call it with IPython. Even the magic's work. To build the document use `ipython builddocs.ipy` You will have to change the paths to the software however.

Cross Platform Output Rendering

I have tested the HTML outputs on my Galaxy S3 and S4, IPAD and Nexus7. They render very well. Even the downloaded PDF was easily readable on the NEXUS 7 in landscape mode. In conclusion the produces work is really very well packaged and easily consumed on most platforms. This is not bad, and all done with open source software.

Some interesting links

- A book written with IPython Notebook
- Notebook Viewer
- Anaconda - Installing almost everything you need

About the presenter

- I am an Electrical Engineer and is currently working for a consulting firm where I manage the Business Analytics and Quantitative Decision Support Services division.
- I use python in my day to day work as a practical alternative to the limitations of EXCEL in using large data sets.
- LinkedIn
- I am also a co-founder at House4Hack

IPython Notebook - A cookbook?

Quick IPython Introdution

IPython provides a rich architecture for interactive computing with:

- Powerful `interactive shells` (terminal and Qt-based).
- A `browser-based notebook` with support for code, text, mathematical expressions, inline plots and other rich media.
- Support for `interactive data visualization` and use of GUI toolkits.
- Flexible, embeddable interpreters to load into your own projects.
- Easy to use, high performance tools for parallel computing.

The main reasons I have been using it includes:

- A `superior` shell
- Plotting is possible in the QT console or the Notebook
- the `magic` functions makes life easier
- I also use it as a `replacement shell` for Windows Shell or Terminal
- `Code Completion`
- `GNU Readline` based editing and command history

Some Imports and Settings

The following code cells make sure that plotting is enabled and also loads a customised matplotlib confirguration file that spices up the inline plots. The custom matplotlib file has been taken from the [Bayesian Methods for Hackers Project](#)

```
# makes sure inline plotting is enabled
%pylab --no-import-all inline
```

Populating the interactive namespace from numpy and matplotlib

```
#loads a customer marplotlib configuration file
import json
s = json.load( open("static/matplotlibrc.json") )
matplotlib.rcParams.update(s)
figsize(18, 4)
print "Imported customer plotting"
```

Imported customer plotting

Changing the Notebook Layout

The code cell below is an example of how you should not be chaning the layout and css of the notebook. From IPython V1.00 it is possible to include custom css by creating IPython profiles. Since this file needs to be distributable I have opted for the hack below as used by the `Bayesian Methods for Hackers Team`

```
from IPython.core.display import HTML
def css_styling():

    styles = open("static/custom.css", "r").read()
    return HTML(styles)
css_styling()
```

Notebook Basics

The IPython Notebook is a web-based interactive computational environment where you can combine

code execution, text, mathematics, plots and rich media into a single document.

- Code Completion
- Help
- Docstrings
- Markdown cells
- Slide Cells
- Running Code

Using Markdown

Notebook Magics

Timeit

```
%%timeit x = 0     # setup
for i in range(100000):
    x += i**2
```

```
100 loops, best of 3: 12.2 ms per loop
```

User Input

```
raw = raw_input("enter your input here >>> ")
print "hallo ", raw
```

```
enter your input here >>> world
hallo  world
```

Linking to Local Files

```
from IPython.display import FileLink, FileLinks
FileLinks('.')
```

./
  .DS_Store
  builddocs.ipy
  pycon13_ipython.ipynb
  README.md
./.ipynb_checkpoints/
  pycon13_ipython-checkpoint.ipynb
./output/
  .DS_Store
  pycon13_ipython.html
  pycon13_ipython.slides.html
  pycon13_ipython_complete.pdf
  pycon13_ipython_pdf.pdf
./static/
  .DS_Store
  custom.css
  frontpage.docx
  frontpage.pdf
  matplotlibrc.json

Adding Images

Adding Youtube Videos

I am making the video small as it does not embed into the final output pdf.
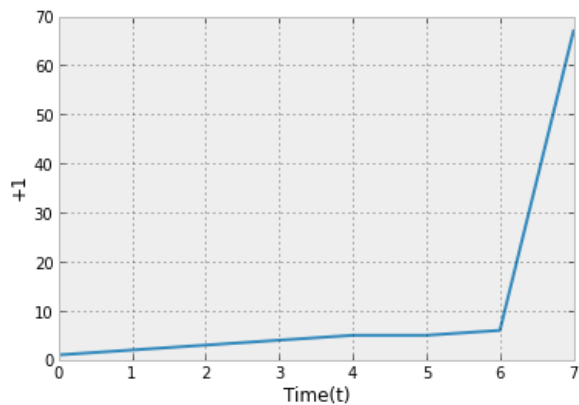
```
from IPython.display import YouTubeVideo
YouTubeVideo('iwVvqwLDsJo', width=200, height=200)
```

Plotting with Matplotlib

```
%pylab
xlabel('Time(t)')
ylabel('+1')
plot([1,2,3,4,5,5,6,67])
```

```
Using matplotlib backend: module://IPython.kernel.zmq.pylab.backend_inline
Populating the interactive namespace from numpy and matplotlib
```

```
[<matplotlib.lines.Line2D at 0x10621a4d0>]
```

Symbolic math

Massaging some data with Pandas

Typesetting

Latex

Latex is rendered using the mathjax javascript library

$$c = \sqrt{a^2 + b^2}$$

Using the Python Debugger - pdb

```
%pdb
```

Automatic pdb calling has been turned ON

```
foo = 1
bar = 'a'
print foo+bar
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-12-08464a413a31> in <module>()
      1 foo = 1
      2 bar = 'a'
----> 3 print foo+bar

TypeError: unsupported operand type(s) for +: 'int' and 'str'
> <ipython-input-12-08464a413a31>(3)<module>()
      1 foo = 1
      2 bar = 'a'
----> 3 print foo+bar

ipdb> ?
```
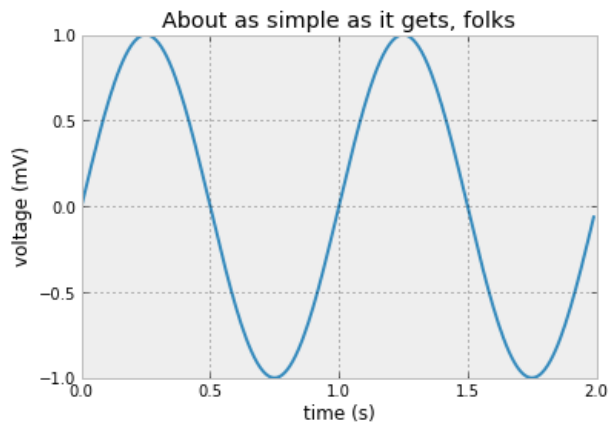
Loading Code Snippets

```
%load http://matplotlib.org/mpl_examples/pylab_examples/simple_plot.py
```

```
from pylab import *

t = arange(0.0, 2.0, 0.01)
s = sin(2*pi*t)
plot(t, s)

xlabel('time (s)')
ylabel('voltage (mV)')
title('About as simple as it gets, folks')
grid(True)
```

```
from pylab import *

t = arange(0.0, 2.0, 0.01)
s = sin(2*pi*t)
plot(t, s)

xlabel('time (s)')
ylabel('voltage (mV)')
title('About as simple as it gets, folks')
```

Saving a Gist

It is possible to save spesific lines of code to a GitHub gist. This is achieved with the `pastebin` magic as demonstrated below.

```
%pastebin "cell one" 0-10
```

u'https://gist.github.com/6521768'

Publishing Your Work

- NBConvert
- HTML
- PDF
- LATEX
- Reveal.js slideshow

File links to exported content

The links below can be used to verify the output from the convertion process. This saved me a lot of time as I could just click below and have a look at the files without exiting the notebook.

```
FileLinks('output/')
```

Building(exporting) from within the notebook

You can even call the build script from the notebook. The script will convert this page to an html and slide file. It will also compile to PDF and stitch a front page to it. Some of the last text in the building process wont appear as this notebook is being updated as it is being compile. Maybe not the best idea but saved a lot of time...

```
!ipython builddocs.ipy
```

srunning on mac

******************** Converting to HTML

[NbConvertApp] Using existing profile dir: u'/Users/tobie/.ipython/profile_default'

[NbConvertApp] Converting notebook pycon13_ipython.ipynb to html

[NbConvertApp] Support files will be in pycon13_ipython_files/

Links to some interesting notebooks

Sources / References

Since this demonstration focussed on the life cycle of the analysis to publication many of the code examples were taken from their respective websites. If I have not given credit at any point please let me know and I will make sure that the work is updated

- IPython
- Matplotlib
- Bayesian Methods for Hackers

Embedding the final presentation into the notebook!

The build script generates a slideshow version of this notebook and saves it in the output directory. Using an inline html iframe it is embedded below.