

IP[y]:
IPython

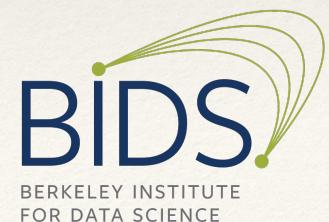


IPython & Project Jupyter

A language-independent
architecture for open
computing and data
science

Fernando Pérez
([@fperez_org](https://twitter.com/fperez_org) & fperez@lbl.gov)

LBL & UC Berkeley



IPython: CU Boulder, 2001

or how to best procrastinate on a Physics dissertation

```
In [13]: run ~/scratch/error
reps: 5
-----
ValueError                                Traceback (most recent call last)
/home/fperez/scratch/error.py in <module>()
    70 if __name__ == '__main__':
    71     #explode()

--> 72     main()
    73     g2='another global'

/home/fperez/scratch/error.py in main()
    60     array_num = zeros(size,'d')
    61     for i in xrange(reps):
--> 62         RampNum(array_num, size, 0.0, 1.0)
    63     RNtime = time.clock()-t0
    64     print 'RampNum time:', RNtime

/home/fperez/scratch/error.py in RampNum(result, size, start, end)
    43     tmp = zeros(size+1)
    44     step = (end-start)/(size-1-tmp)
--> 45     result[:] = arange(size)*step + start
    46
    47 def main():

ValueError: shape mismatch: objects cannot be broadcast to a single shape
In [14]: 
```

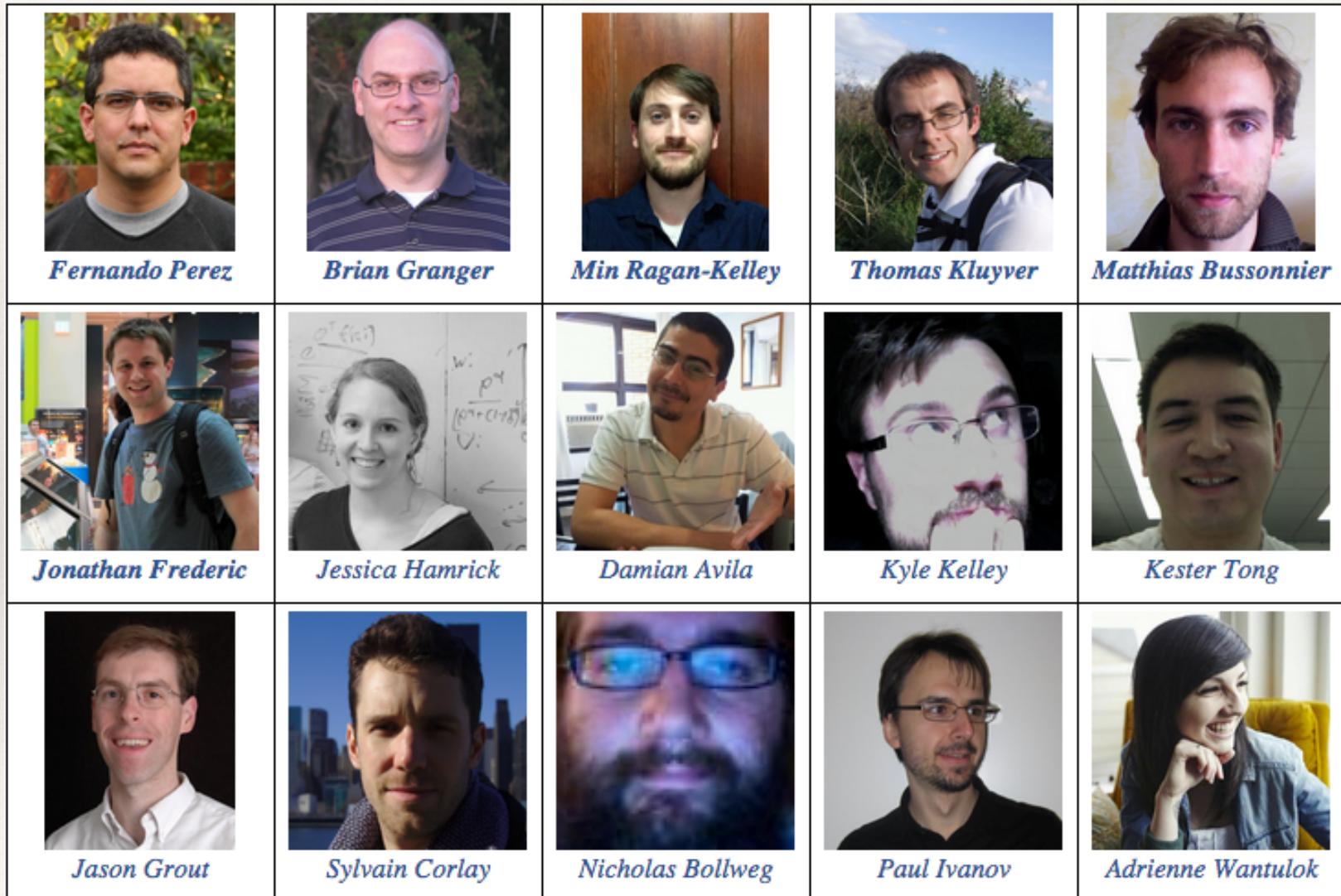
November 2001: "Just an afternoon hack"

- ❖ 259 Line Python script.
- ❖ `sys.ps1 -> In [N].`
- ❖ `sys.displayhook -> Out [N]`, caches results.
- ❖ Plotting, Numeric, etc.

Now (Openhub stats)

- ❖ 19,279 commits
- ❖ 442 contributors
- ❖ Total Lines: 187,326
- ❖ Number of Languages : 7 (JS, CSS, HTML, ...)

Real credit goes to whole team



Plus ~ 500 more Open source contributors!

Current and recent funding



**ALFRED P. SLOAN
FOUNDATION**

GORDON AND BETTY
MOORE
FOUNDATION

The logo for Continuum Analytics, consisting of two overlapping curved bands, one blue and one green.
CONTINUUM
ANALYTICS

The logo for Enthought, featuring a blue 3D cube icon followed by the word 'ENTHOUGHT' and 'SCIENTIFIC COMPUTING SOLUTIONS' below it.

THE LEONA M. AND HARRY B.
HELMESLEY
CHARITABLE TRUST

SIMONS FOUNDATION

 POWERED BY
rackspace[®]
the open cloud company

 **Microsoft**

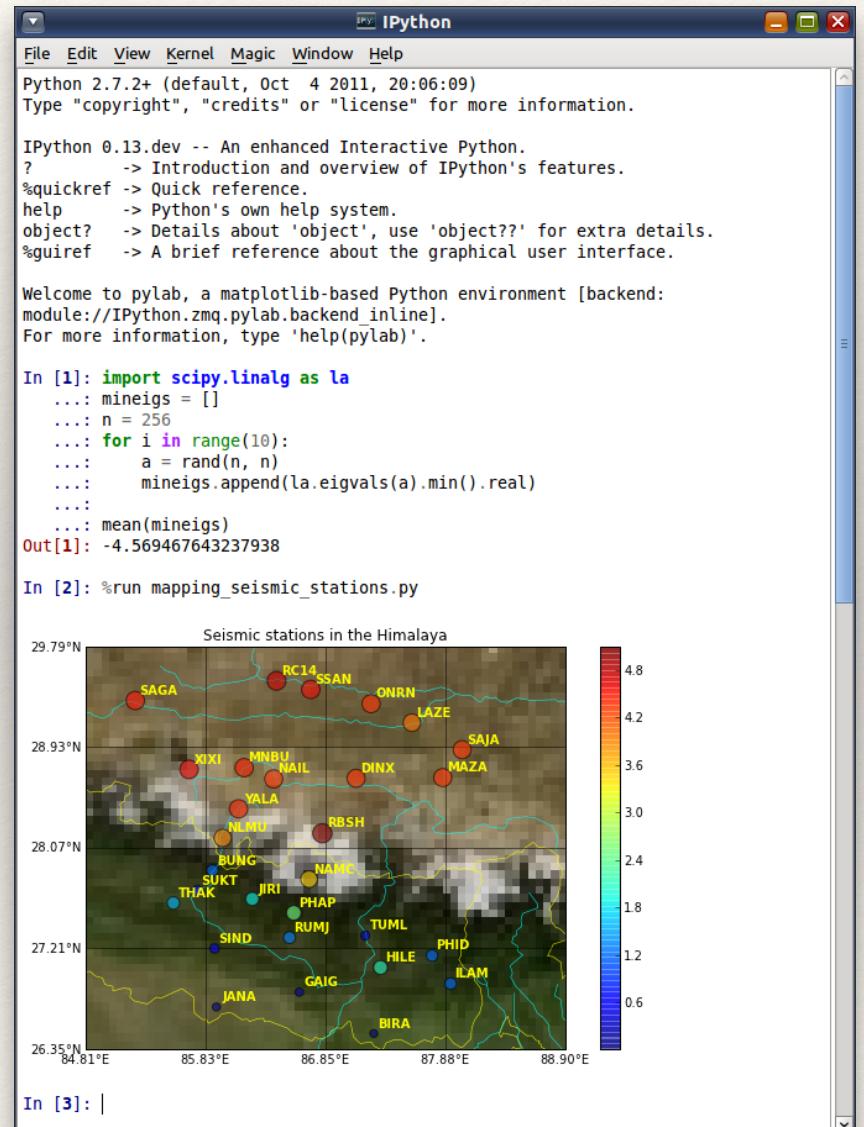
 **Google**

Bloomberg

Beyond the Terminal...

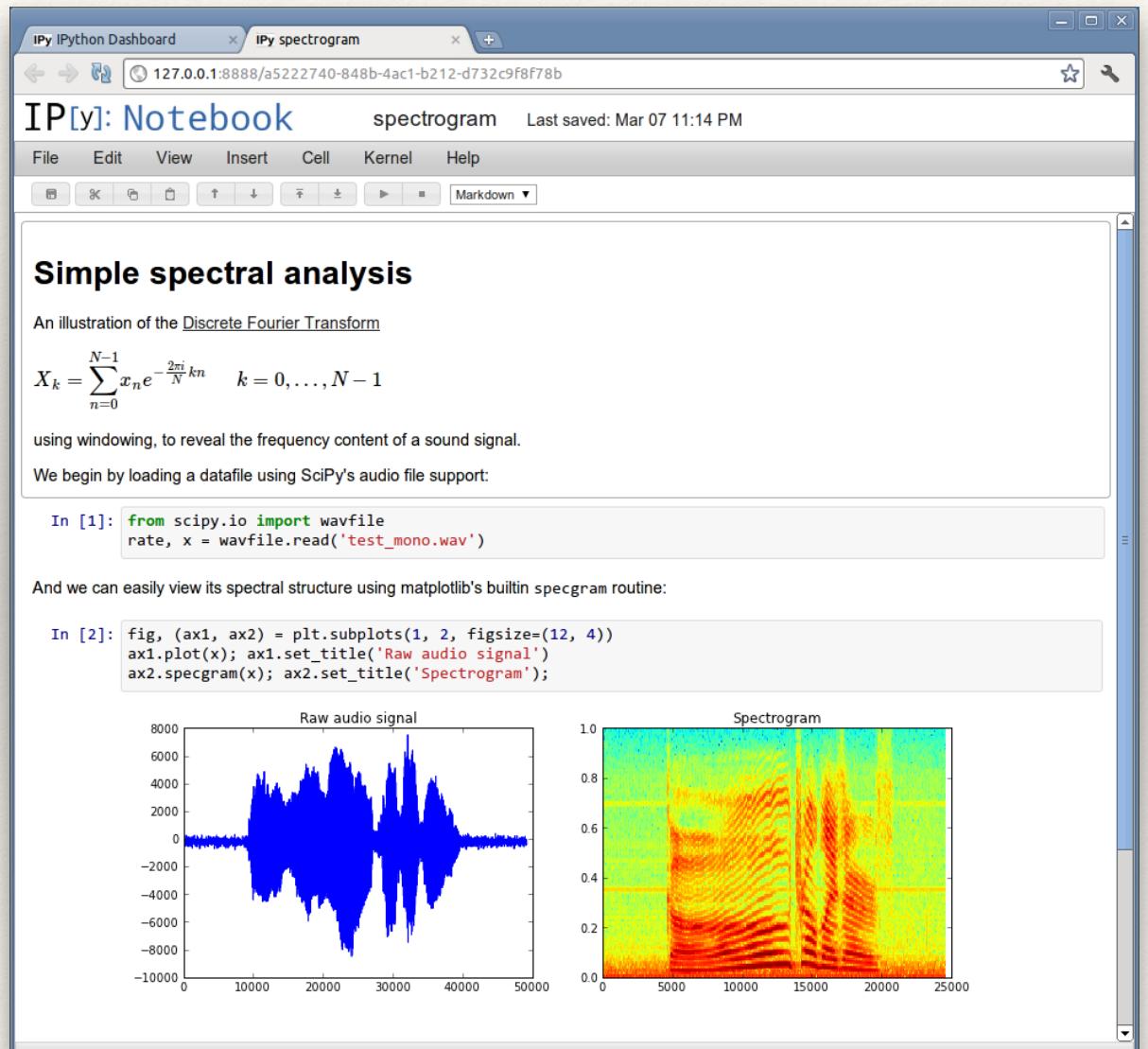
- ❖ The REPL as a network protocol
- ❖ Kernels
 - ❖ execute code
- ❖ Clients
 - ❖ Read input
 - ❖ Present output

Simple abstractions enable rich,
sophisticated clients



2011: The IPython Notebook

- ❖ Rich web client
- ❖ Text & math
- ❖ Code
- ❖ Results
- ❖ Share, reproduce.

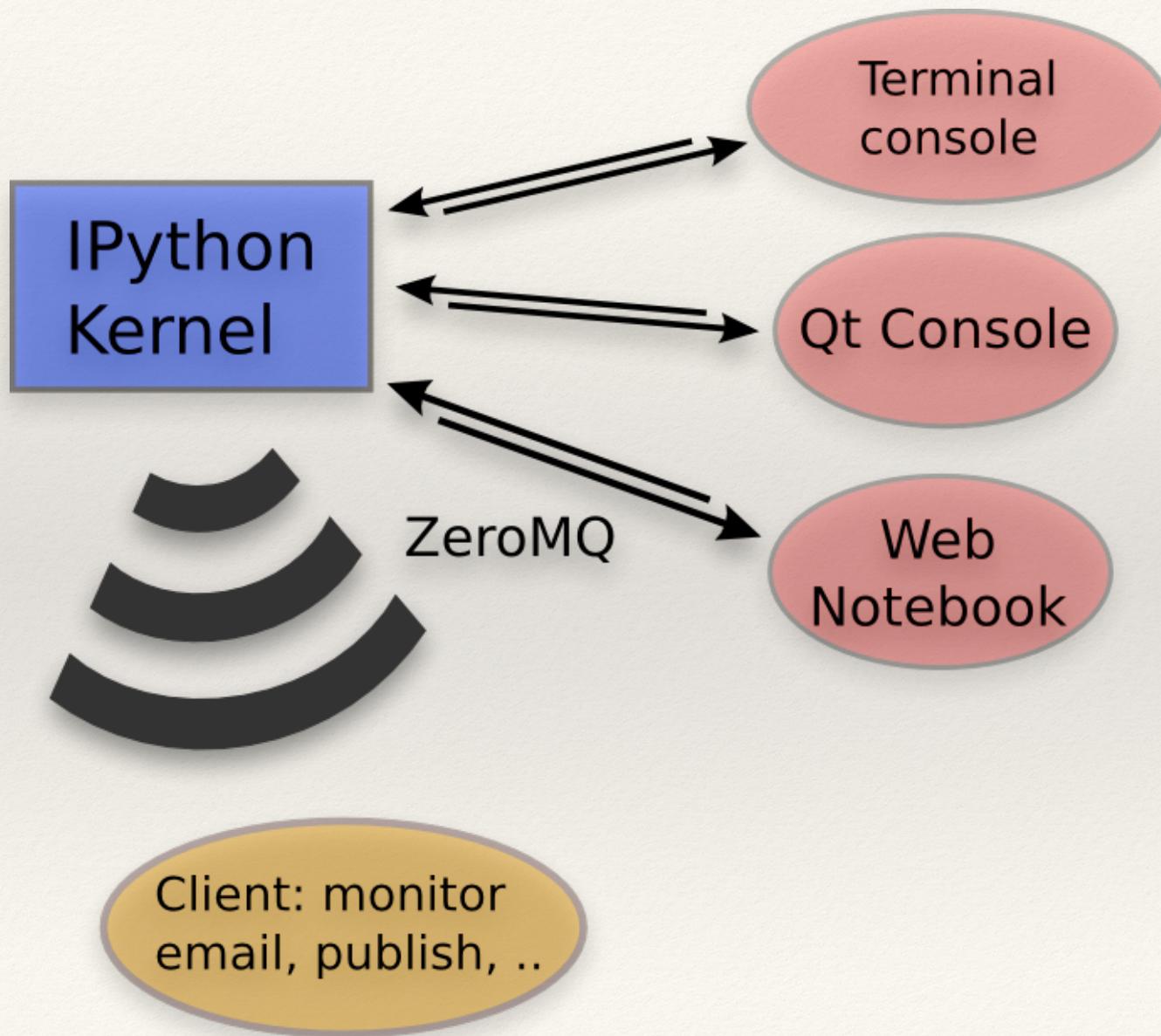


From IPython to Project Jupyter

IP[y]:
IPython



A simple and generic architecture



Not just about Python: Kernels in any language

- ❖ IPython "Official", we ship it.
- ❖ IJulia
- ❖ IRKernel
- ❖ IHaskell
- ❖ IFSharp
- ❖ Ruby
- ❖ IScala
- ❖ IErlang
- ❖ Lots more! ~37 and counting

“Why is it called IPython,
if it can do Julia, R, Haskell, Ruby, ... ?”

IPython

- ❖ Interactive Python shell at the terminal
- ❖ Kernel for this protocol in Python
- ❖ Tools for Interactive Parallel computing
 - ❖ Network protocol for interactive computing
 - ❖ Clients for protocol
 - ❖ Console
 - ❖ Qt Console
 - ❖ Notebook
 - ❖ Notebook file format & tools (nbconvert...)
 - ❖ Nbviewer

IPython

...

Jupyter

- ❖ Interactive Python shell at the terminal
- ❖ Kernel for this protocol in Python
- ❖ Tools for Interactive Parallel computing

- ❖ Network protocol for interactive computing
- ❖ Clients for protocol
 - ❖ Console
 - ❖ Qt Console
 - ❖ Notebook
- ❖ Notebook file format & tools (nbconvert...)
- ❖ Nbviewer



Language Agnostic

What's in a name?

- ❖ *Inspired* by the open languages of science:
 - ❖ Julia, Python & R
 - ❖ *not* an acronym: *all languages* equal class citizens.
- ❖ **Astronomy** and Scientific Python:
 - ❖ A long and fruitful collaboration
- ❖ **Galileo's** notebooks:
 - ❖ the original, open science, data-and-narrative papers
 - ❖ Authorea: “Science was Always meant to be Open”

The Jupyter Notebook Ecosystem

nbviewer: seamless notebook sharing

- ❖ Zero-install reading of notebooks
- ❖ Just share a URL
- ❖ nbviewer.ipython.org

nbviewer FAQ IPython Jupyter

nbviewer

A simple way to share Jupyter Notebooks

URL | GitHub username | GitHub username/repo | Gist ID Go!

IPython

IRuby

IJulia

An IJulia Preview

Programming Languages

Books

Python for Signal Processing

Mining the Social Web

Probabilistic Programming

Data Visualization with Lightning

Interactive plots with Plotly

Misc

Interactive data visualization with Bokeh

Reproducible Research

The screenshot shows a web browser displaying the *ISME Journal* website. The URL in the address bar is www.nature.com/ismej/journal/v7/n3/full/ismej2012123a.html. The page title is "The ISME Journal" with the subtitle "Multidisciplinary Journal of Microbial Ecology". The main content area features a green header image of microorganisms. A sidebar on the left contains links for "Journal home", "Advance online publication", "Current issue", "Archive" (which is highlighted), "Focuses", "Browse by subject", and "Press releases". The main content area has a "Commentary" section with the following text:
The ISME Journal (2013) 7, 461–464; doi:10.1038/ismej.2012.123;
published online 25 October 2012
Collaborative cloud-enabled tools allow rapid,
reproducible biological insights
[Open](#)
Benjamin Ragan-Kelley^{1,12}, William Anton Walters^{2,12},
Daniel McDonald^{3,6,12}, Justin Riley⁴, Brian E Granger⁵,
Antonio Gonzalez⁶, Rob Knight^{7,8}, Fernando Perez⁹ and J
Gregory Caporaso^{10,11}
Footnotes:
¹Graduate Group in Applied Science and Technology, University of California at Berkeley, Berkeley, CA, USA
²Department of Molecular, Cellular and Developmental Biology, University of Colorado at Boulder, Boulder, CO, USA
³Biofrontiers Institute, University of Colorado at Boulder, Boulder, CO, USA
⁴Office of Educational Innovation and Technology, Massachusetts Institute of Technology, Cambridge, MA, USA
⁵Physics Department, California Polytechnic State University, San Luis Obispo, CA, USA
⁶Department of Computer Science, University of Colorado at Boulder, Boulder, CO, USA
A sidebar on the right provides links to "FULL TEXT" options like "Download PDF", "Send to a friend", and "View interactive PDF in ReadCube", as well as links to "Data availability", "References", "Acknowledgements", "Figures and Tables", "Supplementary info", "Export citation", "Export references", and "Papers by Ragan-Kelley".

Paper, Notebooks and Virtual Machine

This notebook is intended to calculate the positions of primers in an alignment, using functions from PrimerProspector.

Import the needed functions, and define the primer sequences

```
In [8]: # Code modified from PrimerProspector library slice_aligned_region.py (development version)
# Imports and definitions
from string import lower, upper
from operator import itemgetter

from cogent import LoadSeqs, DNA
from cogent.core.alphabet import AlphabetError
from cogent.align.align import make_dna_scoring_dict, local_pairwise
from cogent.parse.fasta import MinimalFastaParser
from cogent.core.moltype import IUPAC_DNA_ambiguities

DNA_CODES = ['A', 'C', 'T', 'G', 'R', 'Y', 'M', 'K',
             'W', 'S', 'B', 'D', 'H', 'V', 'N']

# Note that these are all written 5'->3', the reverse primers are reverse complemented for
# the local alignment

# If one wanted to test different primers, they would be defined here.

# 27f/338r = V2 (also includes V1, but generally just referred to as V2)
# 349f/534r = V3
# 515f/806r = V4
# 967f/1046r = V6
# 1391f/1492r = V9

primer_seqs = {
    '27f': 'AGAGTTTGATCMTGGCTCAG',
    '338r': DNA.rc('GCTGCCTCCCGTAGGAGT'),
    '349f': 'GYGCASCAGKCGMGAAN',
    '534r': DNA.rc('ATTACCGCCGCTGCTGG'),
    '515f': 'GTGCCAGCMGCCGCGTAA',
    '806r': DNA.rc('GGACTACVSGGGTATCTAAT'),
    '967f': 'CAACCGGAAGAACCTTACCC',
    '1048r': DNA.rc('CGRCRGCCATGYACCWC'),
    '1391f': 'TGYACACACCGCCGCTC',
    '1492r': DNA.rc('GGCTACCTTGTATTACGACTT'),
    '1391r': 'TGYACACACCGCCGTC' # Need this rather than forward primer to get proper 3' position of reverse version
}

reference_aligned_file = '/home/ubuntu/qiime_software/gg_otus-4feb2011-release/rep_set/gg_76_otus_4feb2011_aligned.fasta'
```

Instructions and supporting data for the QIIME/IPython/StarCluster demo at the 2012 NIH Cloud Computing the Microbiome workshop and our corresponding paper in the ISME Journal.

The analysis made use of the [IPython Notebook](#), [QIIME](#), [StarCluster](#), [PyCogent](#), and [PrimerProspector](#). All of these tools are pre-installed in the ami-9f69c1f6 public Amazon EC2 instance, which was used in this study.

Supporting Files

The IPython notebooks supporting this study can be viewed [here](#) and are available here in PDF format:

- [NIH Cloud Demo \(Complete\)](#)
- [NIH Cloud Demo \(Fast\)](#)
- [Timing*](#)
- [Variable Region Position Boundaries](#)
- [Pearson v Robinson-Foulds Distances](#)
- [V3 and V4 Regions Only](#)

* Note that the Timing notebook is for reference as related to the paper only - it will not be directly reproducible on re-runs of the above notebooks as it relies on the semi-manual creation of the tasks.log file. The tasks.log file used to generate the original timing data is available for [download here](#).

The Greengenes reference OTU collection used in this study is available for [download here](#).

The IPython notebook files (.ipynb) are available for [download here](#).

The tree metadata mapping file used in generating the coloring categories in the 3D PCoA plot is [available here](#).

The paper for this analysis, "Collaborative cloud-enabled tools allow rapid, reproducible biological insights", is available [here](#).

Reproducing the analysis

Four m2.4xlarge instances were booted using StarCluster to create a 32 core cluster with approximately 280GB of RAM (70GB per 8 core instance). This was used for the full analysis (a more complete analysis then was done during the workshop, where the workshop analysis was optimized to run quickly). To support the large quantity of data that is generated during the analysis, you should create an EBS volume which will be attached to the running instance. A 20 GB volume will be sufficient. The volume used for running these notebooks is available as snap-75eb8005.

To reproduce the analyses presented in this paper you should install StarCluster locally, and configure it according to the [instructions on the StarCluster website](#). You can then add the following to your ~/.starcluster/config file:

```
[plugin ipcluster]
setup_class = starcluster.plugins.ipcluster.IPCluster
enable_notebook = true
# If you leave notebook_passwd out, a random password
# will be generated instead.
notebook_passwd = YOUR-PASSWORD

[cluster qiime-ipython]
node_image_id = ami-9f69c1f6
cluster_user = ubuntu
keyname = YOUR-KEY
cluster_size = 4
node_instance_type = m2.4xlarge
plugins = ipcluster
volumes = qiime-ipython-data

[volume qiime-ipython-data]
VOLUME_ID = YOUR-VOLUME-ID
MOUNT_PATH = /home/ubuntu/data
```

Scientific Blogging

SCIENTIFIC AMERICAN™

Sign In | Register

Search ScientificAmerican.com

Subscribe News & Features Topics Blogs Videos & Podcasts Education Cit

SA en español

Blogs About

Like 0 Tweet 2 g+ 3 Share 3 reddit this!

SA Visual Illustrating science since 1845

SA Visual Home About Contact

Visualizing 4-Dimensional Asteroids

By Jake VanderPlas | September 16, 2014

Multicolor plot

Let's put these all together. Rather than using two separate color scales to identify these asteroid groups, we can define a single two-dimensional color scale reflecting the asteroid chemistry and use these colors when plotting the same points in orbital space. The result is a plot very similar to the one that appeared in Parker et al., 2008, where this work was first reported:

Multicolor plot

Let's put these all together. Rather than using two separate color scales to identify these asteroid groups, we can define a single two-dimensional color scale reflecting the asteroid chemistry and use these colors when plotting the same points in orbital space. The result is a plot very similar to the one that appeared in Parker et al., 2008, where this work was first reported:

In [13]: `plot_multicolor()`

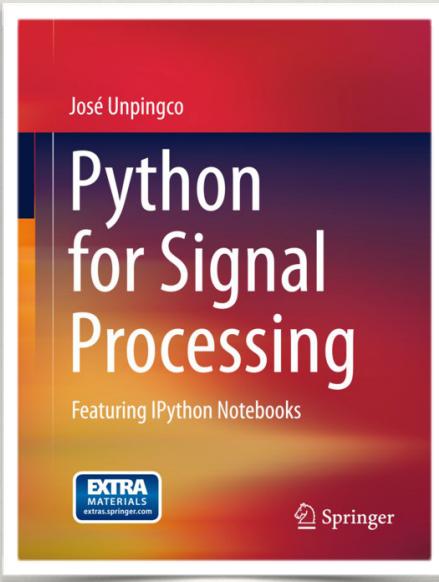
Jake van der Plas @ UW

<http://blogs.scientificamerican.com/sa-visual/2014/09/16/visualizing-4-dimensional-asteroids>

Executable books

Python for Signal Processing, by José Unpingco

- ❖ Springer hardcover book
- ❖ Chapters: IPython Notebooks
- ❖ Posted as a blog entry
- ❖ All available as a Github repo



The screenshots illustrate the content of the blog post:

Top Screenshot (Summary):

- A plot showing a discrete signal (blue dots) and its corresponding DFT spectrum (blue sinc-like curve).
- The x-axis is labeled "time sample index" and ranges from 0 to 16.
- The y-axis ranges from 0.0 to 1.0.

Bottom Screenshot (IPython Notebook):

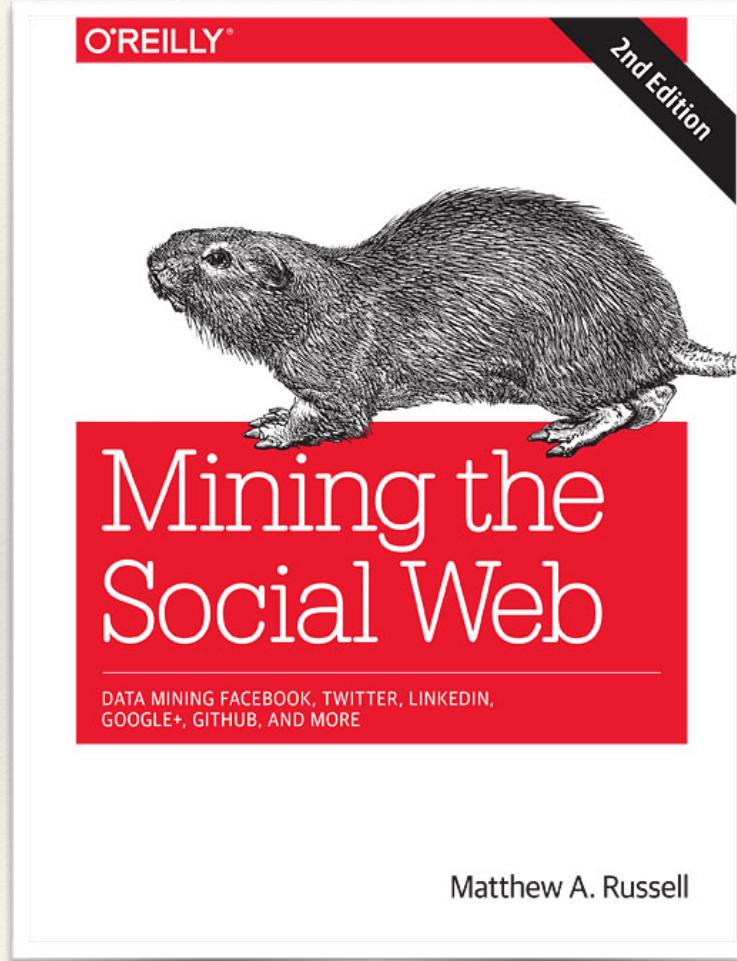
- Code:**

```
def abs_sinc(k=None,N=64,Ns=32):
    if k is None: k = arange(0,N-1)
    y = where(k == 0, 1.0e-20, k)
    return abs(sin( Ns*2*pi/N*y)/sin(2*pi*y/N))/sqrt(N)

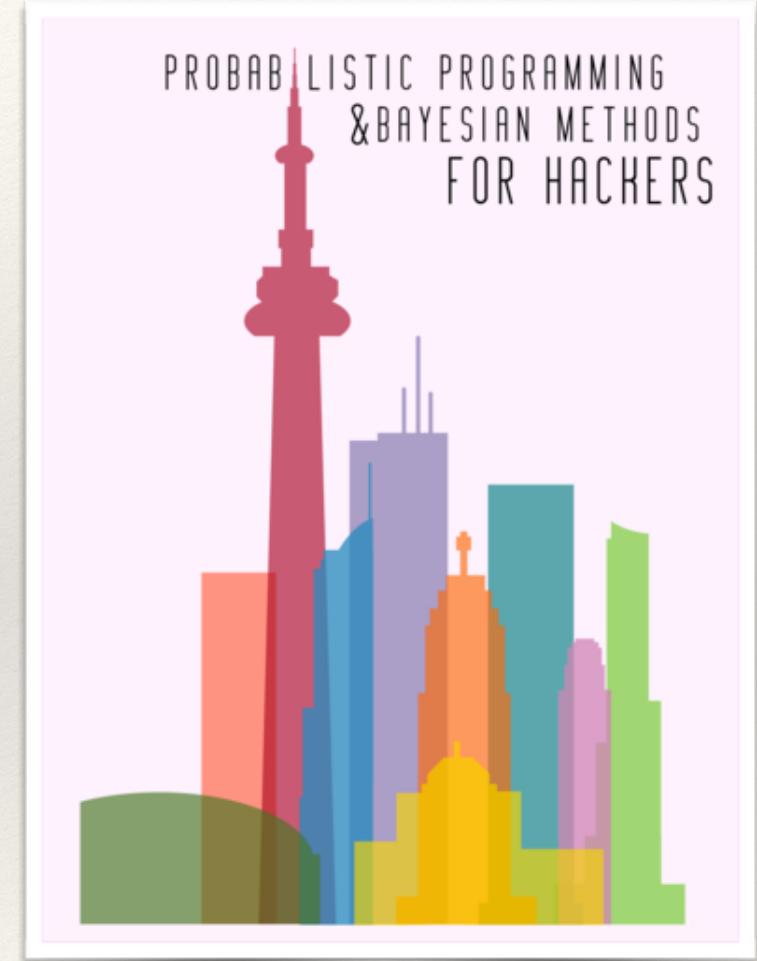
fig,ax=subplots()
fig.set_size_inches((8,3))

ax.plot(abs_sinc(N=512,Ns=10),label='duration=10')
ax.plot(abs_sinc(N=512,Ns=20),label='duration=20')
ax.set_xlabel('DFT Index',fontsize=18)
ax.set_ylabel(r'$|X(\Omega_k)|$',fontsize=18)
ax.set_title('Rectangular Windows DFTs',fontsize=18)
ax.grid()
ax.legend(loc=0);
```
- Plot:** A plot titled "Rectangular Windows DFTs" showing two peaks. The x-axis is "DFT Index" (0 to 600) and the y-axis is \$|X(\Omega_k)|\$ (0.0 to 0.9). One peak is blue (duration=10) and the other is green (duration=20).

More authors creating books this way



By Matthew Russell



By Cameron Davidson-Pilon

University Courses

	Course	University	Instructor
0	Data Science and Visualization with Python	Santa Clara	Brian Granger
1	Python for Data Science	UC Berkeley	Josh Bloom
2	Introduction to Data Science	UC Berkeley	Michael Franklin
3	Working with Open Data	UC Berkeley	Raymond Yee
4	Introduction to Signal Processing	UC Berkeley	Miki Lustig
5	Data Science (CS 109)	Harvard University	Pfister and Blitzstein
6	Practical Data Science	NYU	Josh Attenberg
7	Scientific Computing (ASTR 599)	University of Washington	Jake Vanderplas
8	Computational Physics	Cal Poly	Jennifer Klay
9	Introduction to Programming	Alaskan High School	Eric Matthes
10	Aerodynamics-Hydrodynamics (MAE 6226)	George Washington University	Lorena Barba

11	HyperPython: hyperbolic conservation laws	KAUST	David Ketcheson
12	Quantitative Economics	NYU	Sargent and Stachurski
13	Practical Numerical Methods with Python	4 separate universities + MOOC	Barba, et al.
14	Data Science: Algorithms	Columbia - Lede Program	Chris Wiggins
15	Data Science: Databases	Columbia - Lede Program	Chris Wiggins
16	Data Science: Foundations	Columbia - Lede Program	Chris Wiggins
17	Data Science: Platforms	Columbia - Lede Program	Chris Wiggins

These are just some we are aware of!

A collaborative MOOC on OpenEdX

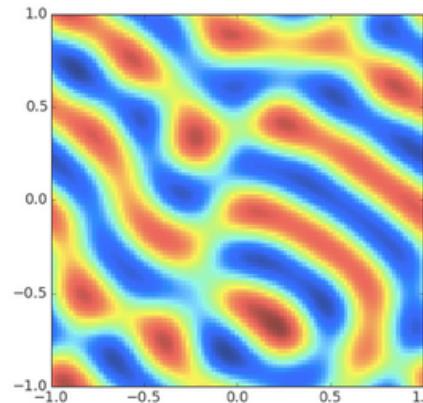
- ❖ Lorena Barba at George Washington University, USA.
- ❖ Ian Hawke at Southampton, UK
- ❖ Carlos Jerez at Pontifical Catholic University of Chile.
- ❖ All materials on [GitHub](#).

Lorena A. Barba group



Announcing "Practical Numerical Methods with Python" MOOC

Posted on 07.26.2014

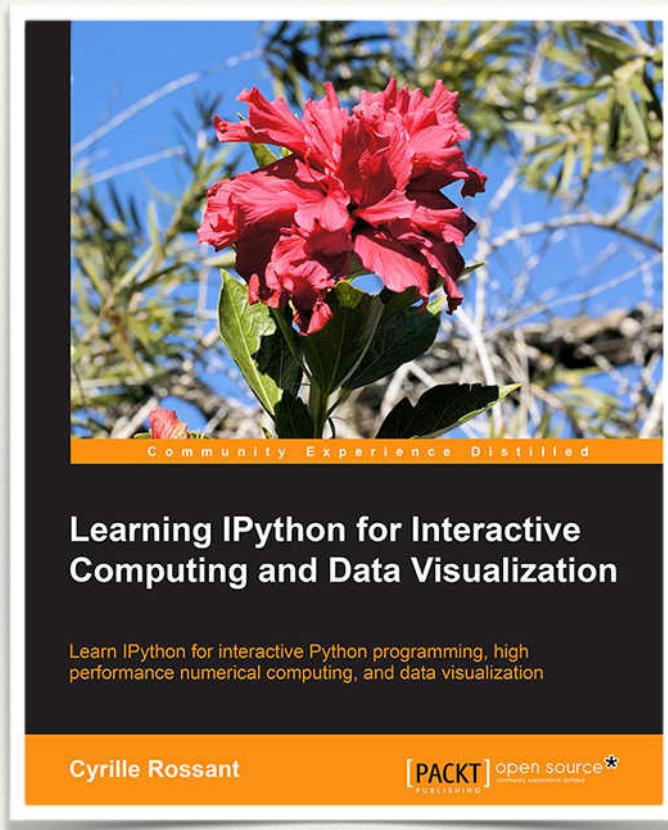


Pattern formation:
► solution for a reaction-diffusion system like:

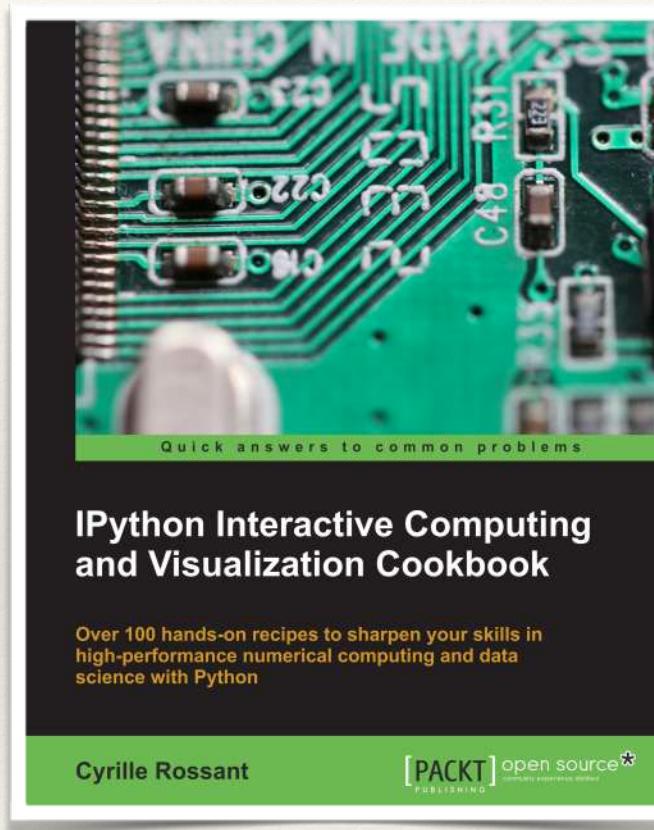
$$\begin{aligned} u_t &= \delta D_1 \nabla^2 u + f(u, v) \\ v_t &= \delta D_2 \nabla^2 v + g(u, v) \end{aligned}$$

An example of the types of problems we will learn to solve in this course, among others governed by differential equations.

Books about IPython



Learning IPython for Interactive Computing and Data Visualization



IPython Interactive Computing and Visualization Cookbook



Cyrille Rossant
cyrille.rossant.net

Changing the scientific culture

nature
International weekly journal of science

Search Advanced search

Home | News & Comment | Research | Careers & Jobs | Current Issue | Archive | Audio & Video | For Authors

Archive > Volume 515 > Issue 7525 > Toolbox > Article

NATURE | TOOLBOX Share | Email | Print

E-mail RSS Facebook Twitter

Interactive notebooks: Sharing the code

The free IPython notebook makes data analysis easier to record, understand and reproduce.

Helen Shen

05 November 2014

[PDF](#) [Rights & Permissions](#)



Illustrations by The Project Twins

Top story



USA 25 Brontosaurus

Beloved *Brontosaurus* makes a comeback

Jurassic giant's taxonomic status is restored.

Recent Read Comments Emailed

1. History: Women at the edge of science
Nature | 08 April 2015
2. Scientific instrumentation: The aided eye
Nature | 08 April 2015
3. Books in brief
Nature | 08 April 2015
4. Antibody shows promise as

<http://www.nature.com/news/interactive-notebooks-sharing-the-code-1.16261>

Executable papers: the future?

nature.com | Sitemap

Login Register Close

IP[y]: Notebook Nature (autosaved) IPython (Python 3)

File Edit View Insert Cell Kernel Help

Cell Toolbar: None

nature rackspace

Introduction

Welcome! You have just launched a live example of an IPython Notebook. The notebook is an open-source, interactive computing environment that lets you combine live code, narrative text, mathematics, plots and rich media in one document. Notebook documents provide a complete reproducible record of a computation and its results and can be shared with colleagues (through, for example, email, web-hosting services such as GitHub, Dropbox, and nbviewer).

You can edit anything in this temporary demonstration notebook, including the text you are reading. To see it full-screen, click on the 'Expand' icon in the lower right corner of the frame around this notebook.

This notebook showcases some of IPython's capabilities for researchers.

This demonstration is hosted by [Rackspace](#) and is running on its bare metal offering, [OnMetal](#). Try out these cloud services yourself through [Rackspace's developer+ page](#).

Basic Python code and plotting

The box below (known as a code cell) contains the Python code to plot $y = x^2$ over the range $[0,5]$. The blue comments preceded by # explain what the code does.

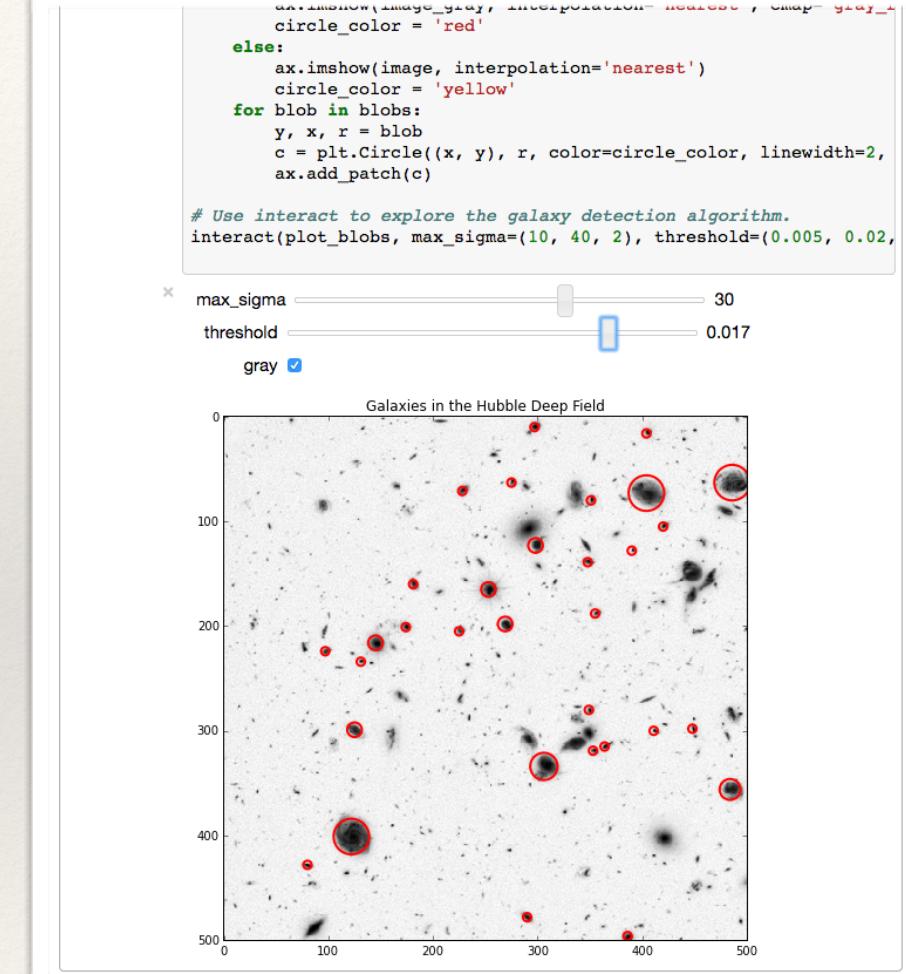
To run the code:

1. Click on the cell to select it.
2. Press SHIFT+ENTER on your keyboard or press the play button (▶) in the toolbar above.

A full tutorial for using the notebook interface is available [here](#).

```
In [ ]: # Import matplotlib (plotting) and numpy (numerical arrays).
# This enables their use in the Notebook.
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np

# Create an array of 30 values for x equally spaced from 0 to 5.
x = np.linspace(0, 5, 30)
```



<http://www.nature.com/news/ipython-interactive-demo-7.21492?article=1.16261>

Notebook Workflows: The Big Picture

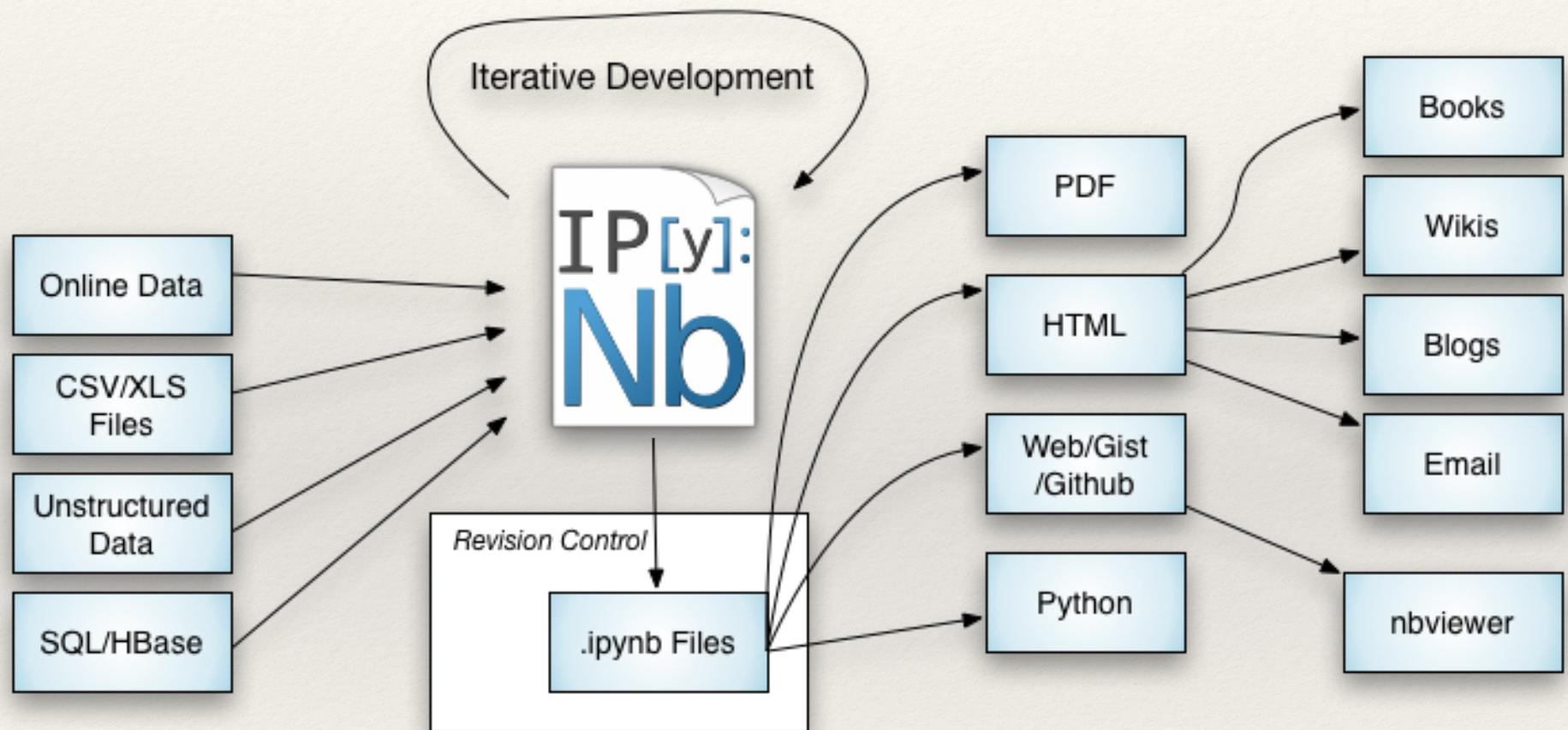


Image credit: [Joshua Barratt](#)

Lots more! The IPython Gallery

A gallery of interesting IPython Notebooks

Fernando Perez edited this page 8 days ago · 229 revisions

This page is a curated collection of IPython notebooks that are notable for some reason. Feel free to add new content here, but please try to only include links to notebooks that include interesting visual or technical content; this should *not* simply be a dump of a Google search on every ipynb file out there.

Important contribution instructions: If you add new content, please ensure that for any notebook you link to, the link is to the rendered version using [nbviewer](#), rather than the raw file. Simply paste the notebook URL in the nbviewer box and copy the resulting URL of the rendered version. This will make it much easier for visitors to be able to immediately access the new content.

Note that Matt Davis has conveniently written a set of [bookmarklets and extensions](#) to make it a one-click affair to load a Notebook URL into your browser of choice, directly opening into nbviewer.

Table of Contents

1. Entire books or other large collections of notebooks on a topic
 - [Introductory Tutorials](#)
 - [Programming and Computer Science](#)
 - [Statistics, Machine Learning and Data Science](#)
 - [Mathematics, Physics, Chemistry, Biology](#)
 - [Earth Science and Geo-Spatial data](#)
 - [Linguistics and Text Mining](#)
 - [Signal Processing](#)
2. Scientific computing and data analysis with the SciPy Stack
 - [General topics in scientific computing](#)
 - [Social data](#)
 - [Psychology and Neuroscience](#)
 - [Machine Learning](#)
 - [Physics, Chemistry and Biology](#)
 - [Economics](#)
 - [Earth science and geo-spatial data](#)

Reproducible academic publications

This section contains academic papers that have been published in the peer-reviewed literature or pre-print sites such as the [ArXiv](#) that include one or more notebooks that enable (even if only partially) readers to reproduce the results of the publication. If you include a publication here, please link to the journal article as well as providing the nbviewer notebook link (and any other relevant resources associated with the paper).

1. [Reply to 'Influence of cosmic ray variability on the monsoon rainfall and temperature': a false-positive in the field of solar-terrestrial research](#) by Benjamin Laken, 2015. Reviewed article will appear in JASTP. The [IPython notebook](#) reproduces the full analysis and figures exactly as they appear in the article, and is available on Github: link via [figshare](#).
2. [The probability of improvement in Fisher's geometric model: a probabilistic approach](#), by Yoav Ram and Lilach Hadany. ([Theoretical Population Biology](#), 2014). An [IPython notebook](#), allowing figure reproduction, was deposited as a [supplementary file](#).
3. [Stress-induced mutagenesis and complex adaptation](#), by Yoav Ram and Lilach Hadany ([Proceedings B](#), 2014). An [IPython notebook](#), allowing figures reproduction, was deposited as a [supplementary file](#).
4. [Automatic segmentation of odor maps in the mouse olfactory bulb using regularized non-negative matrix factorization](#), by J. Soelter et al. ([Neuroimage](#) 2014, Open Access). The [notebook](#) allows to reproduce most figures from the paper and provides a deeper look at the data. The [full code repository](#) is also available.
5. [Multi-tiered genomic analysis of head and neck cancer ties TP53 mutation to 3p loss](#), by A. Gross et al. ([Nature Genetics](#) 2014). The [full collection of notebooks](#) to replicate the results.
6. [powerlaw: a Python package for analysis of heavy-tailed distributions](#), by J. Alstott et al.. Notebook of examples in manuscript, [ArXiv link](#) and [project repository](#).
7. [Collaborative cloud-enabled tools allow rapid, reproducible biological insights](#), by B. Ragan-Kelley et al.. The [main notebook](#), the [full collection of related notebooks](#) and the [companion site](#) with the Amazon AMI information for reproducing the full paper.
8. [A Reference-Free Algorithm for Computational Normalization of Shotgun Sequencing Data](#), by C.T. Brown et al.. Full notebook, [ArXiv link](#) and [project repository](#).
9. [The kinematics of the Local Group in a cosmological context](#) by J.E. Forero-Romero et al.. The [Full notebook](#) and also all the data in a [github repo](#).

Jupyter as Infrastructure

OSS and commercial adoption

The KBase Narrative Interface

The screenshot shows the KBase Narrative Interface in a web browser window titled "Untitled". The interface has a top navigation bar with tabs for "Analyze", "Narratives", and "Jobs", and a dropdown menu for "DATA". Below the navigation is a search bar and a red "Add Data" button. A sidebar on the left lists "APPS & METHODS" with four items: "Assemble and Annotate Microbial Genome v0.1.0", "Build and Normalize Metagenomic Functional Abundance Data v0.1.0", "Build and Normalize Metagenomic Taxonomic Abundance Data v0.1.0", and "Build Plant Metabolic Model v0.1.0". The main content area features the KBase logo and a "Welcome to the Narrative Interface!" message. It includes a section titled "What's a Narrative?" explaining the purpose of the interface, and a "Get Some Data" section with instructions on how to add data. At the bottom, there is a code input field showing "%pylab inline" and "plot(rand(100));", followed by a generated plot of a noisy blue line from 0 to 100.

Microsoft: Visual Studio & Azure Cloud

Shahrokh Mortazavi, Dino Viehland, Wenming Ye, Dennis Gannon.

The image shows two side-by-side screenshots of Microsoft Visual Studio and an IPython Notebook.

Visual Studio Screenshot:

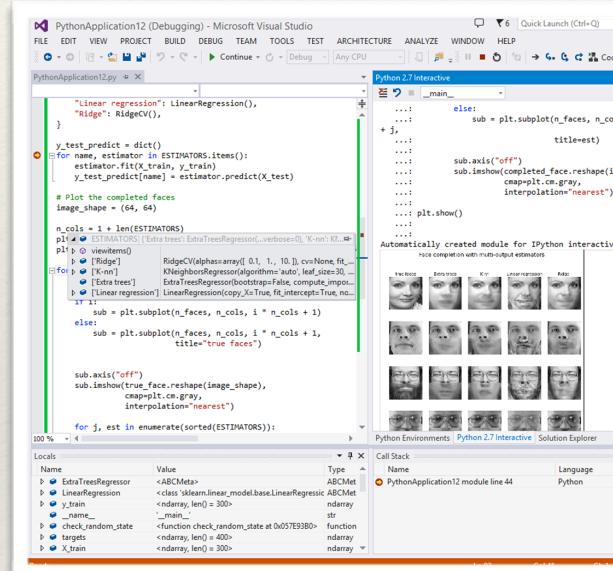
- Code Editor:** Shows Python code for face recognition using various estimators. The code includes imports for `sklearn` and `matplotlib`, defines an `ESTIMATORS` dictionary, and iterates through it to fit models and plot results.
- Python 2.7 Interactive Window:** Displays the execution of the script, showing the creation of a module for the IPython environment and the resulting plots.
- Locals Window:** Shows variables and their values, including estimators like `ExtraTreesRegressor`, `Knn`, and `Ridge`.
- Call Stack Window:** Shows the current call stack, indicating the code is running in the main module.

IPython Notebook Screenshot:

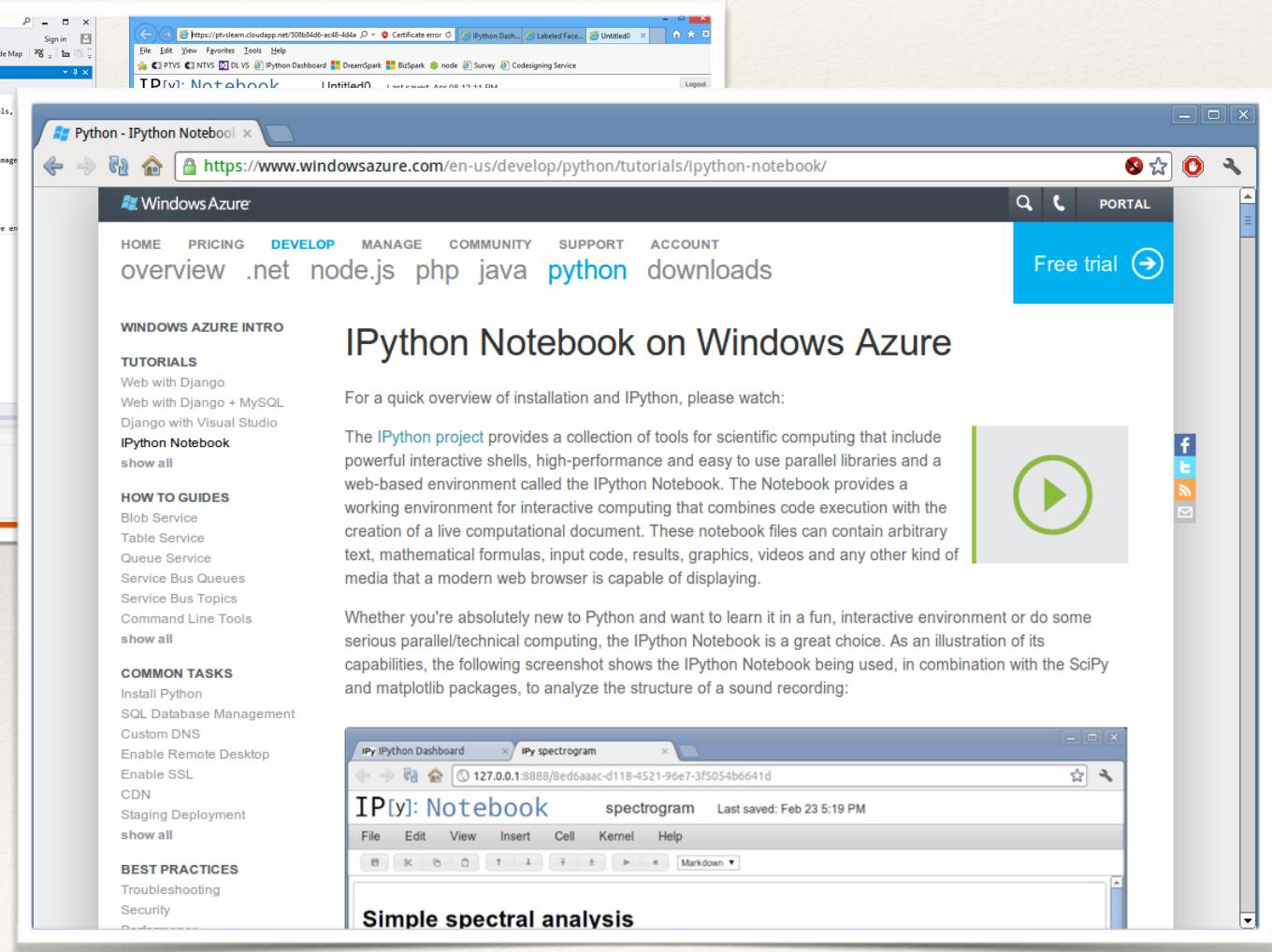
- Code Cell:** Shows the same Python code being run in an IPython notebook.
- Output:** Displays a grid of 15 face images, each labeled with its corresponding estimator name: true faces, Extra trees, Knn, Linear regression, and Ridge. The images show varying degrees of model performance.

Microsoft: Visual Studio & Azure Cloud

Shahrokh Mortazavi, Dino Viehland, Wenming Ye, Dennis Gannon.



The screenshot shows the Microsoft Visual Studio IDE. On the left, a code editor displays Python code for face recognition, utilizing libraries like scikit-learn and matplotlib. On the right, the 'Python 2.7 Interactive' window shows a script running, displaying a grid of faces. Below the code editor is the 'Locals' window, showing variables like 'LinearRegression', 'RidgeCV', 'y_train', and 'X_train'. A green arrow points from the 'Locals' window towards the IPython Notebook interface.



The screenshot shows an IPython Notebook running on Windows Azure. The main area displays a grid of faces. To the right, there is a detailed description of the IPython Notebook on Windows Azure, including a 'Free trial' button and social sharing icons. At the bottom, a separate window titled 'IP[y]: Notebook' shows a spectrogram analysis with the title 'Simple spectral analysis'.

IPython Notebook on Windows Azure

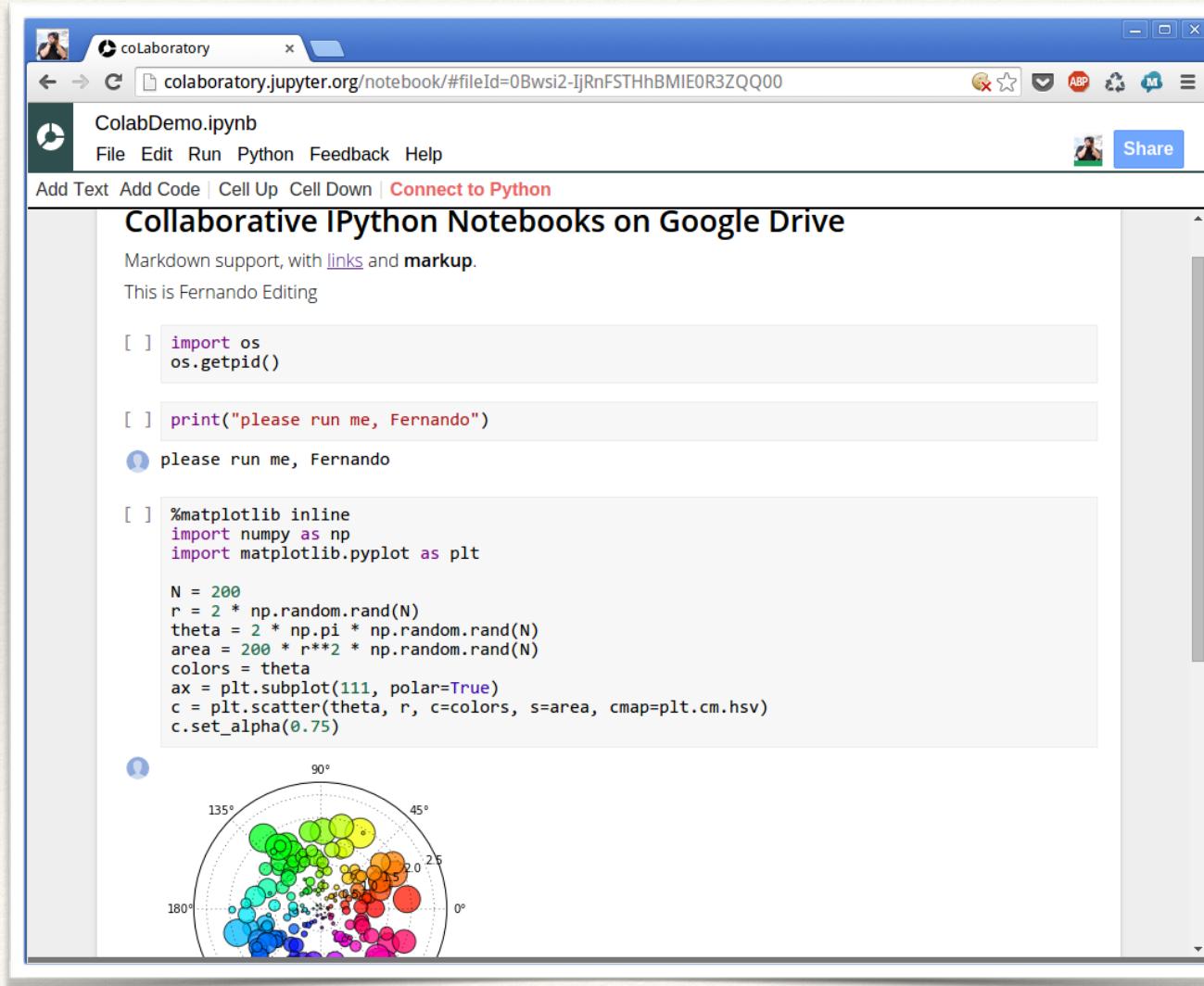
For a quick overview of installation and IPython, please watch:

The [IPython project](#) provides a collection of tools for scientific computing that include powerful interactive shells, high-performance and easy to use parallel libraries and a web-based environment called the IPython Notebook. The Notebook provides a working environment for interactive computing that combines code execution with the creation of a live computational document. These notebook files can contain arbitrary text, mathematical formulas, input code, results, graphics, videos and any other kind of media that a modern web browser is capable of displaying.

Whether you're absolutely new to Python and want to learn it in a fun, interactive environment or do some serious parallel/technical computing, the IPython Notebook is a great choice. As an illustration of its capabilities, the following screenshot shows the IPython Notebook being used, in combination with the SciPy and matplotlib packages, to analyze the structure of a sound recording:

Google CoLaboratory

Kayur Patel, Kester Tong, Mark Sanders, Corinna Cortes @ Google
Matt Turk @ NCSA/UIUC



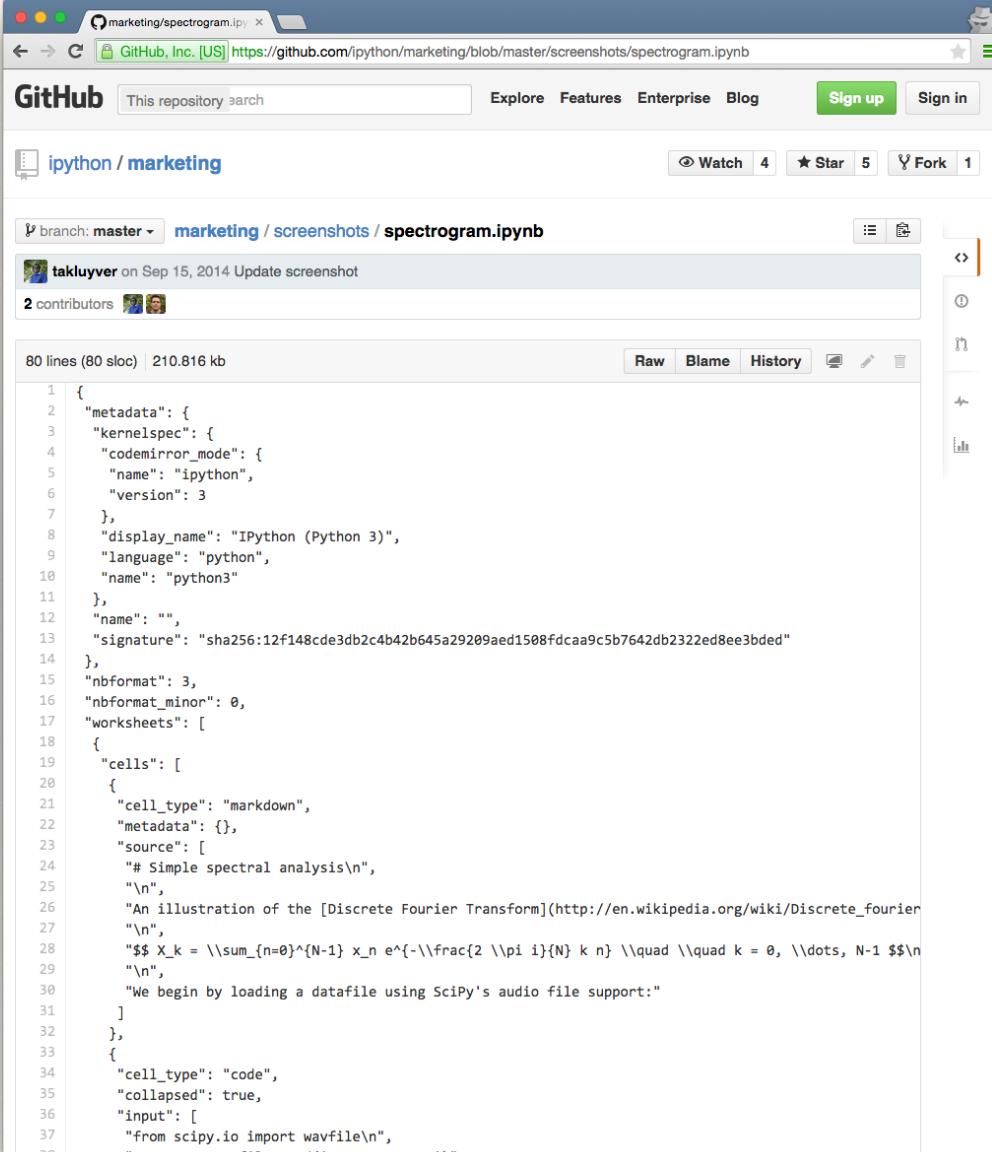
Authorea: notebooks in papers

The screenshot shows the Authorea website interface. At the top, there is a navigation bar with links for INSTITUTIONS, ARTICLES, ABOUT, PLANS, BLOG, HELP, and LOG IN. Below the navigation bar, there are buttons for PUBLIC, WORKING DRAFT, Index, 12 Comments, Export, Share, and Follow. On the left side, there is a sidebar with icons for writing, sharing, and other features, and a button labeled "Start writing It's free". The main content area features a large title: "Data-driven, interactive science, with d3.js plots and IPython Notebooks" by Alberto Pepe, Matteo Cantiello, and Nathan Jenkins. Below the title, there is a text block: "This week we are launching a brand new look for Authorea and a couple of exciting new features aimed at making scientific research more interactive. Since the very beginning of Authorea, we have been striving to make collaborative scientific writing as easy as possible. But in addition to **writing**, we are also creating a space for new ways of **reading** science, and **executing** it." There is also a smaller text block below: "For example, if you are a scientist, chances are that you do a lot of data analysis and you might want to visualize and provide access to your data in some **fun, new, interactive, more meaningful, data-driven** ways, rather than the usual static, data-less plot. There are many ways to create this kind of interactive plots. In this short blog post we will look at two of them."

https://www.authorea.com/users/3/articles/3904/_show_article

Recent Developments

Over 200,000 notebooks on GitHub



A screenshot of a GitHub notebook file, `spectrogram.ipynb`, displayed in a web browser. The browser window has a dark theme. At the top, the address bar shows the URL `https://github.com/ipython/marketing/blob/master/screenshots/spectrogram.ipynb`. The GitHub header includes the repository name "ipython / marketing", the branch "master", and statistics: 4 watches, 5 stars, and 1 fork. Below the header, the notebook's metadata is shown, including its creation date (Sep 15, 2014) and contributors (takluyver). The main content area displays the notebook's code cells. The first cell is a Markdown cell containing a mathematical formula for the Discrete Fourier Transform:

```
# Simple spectral analysis\n,\n,\nAn illustration of the [Discrete Fourier Transform](http://en.wikipedia.org/wiki/Discrete_fourier\n,\n,$\$ X\_k = \\\sum_{n=0}^{N-1} x_n e^{-\\\frac{2 \\\pi i}{N} k n} \\\quad \\\quad k = 0, \\\dots, N-1 \$\$)\n,\n,\nWe begin by loading a datafile using SciPy's audio file support:\n]
```

The notebook contains 80 lines of code across 37 cells.

As of May 6, 2015:

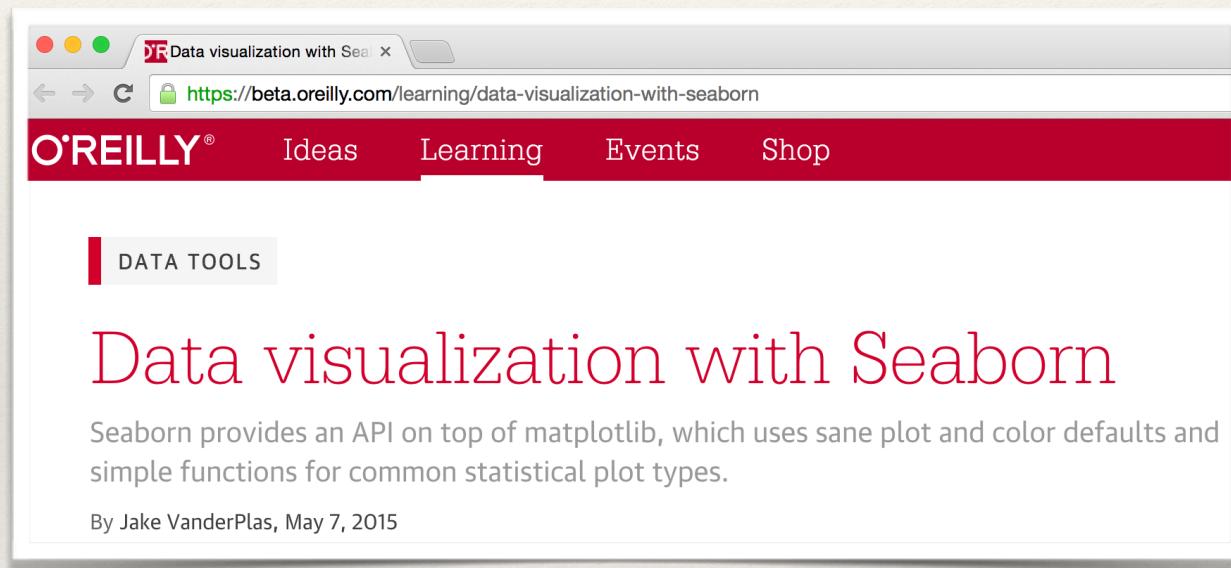
The screenshot shows an IPython notebook interface running on GitHub. The title bar says "marketing/spectrogram.ipynb". The notebook content includes:

- A header section titled "Simple spectral analysis" with a brief description of the Discrete Fourier Transform.
- Code in In [1]:

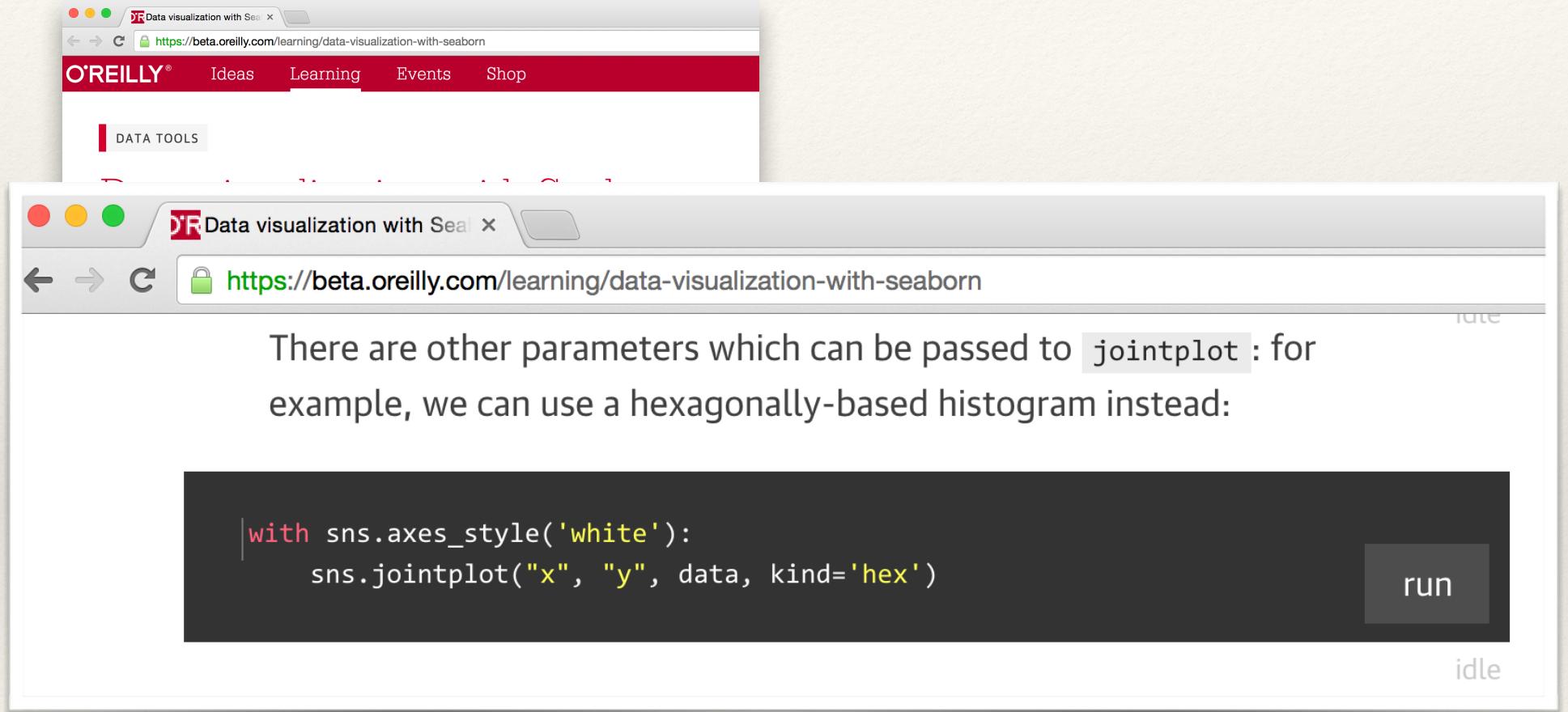
```
from scipy.io import wavfile
rate, x = wavfile.read('test_mono.wav')
```
- A note: "We begin by loading a datafile using SciPy's audio file support:"
- Code in In [2]:

```
#matplotlib inline
from matplotlib import pyplot as plt
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');
```
- Two plots:
 - "Raw audio signal": A line plot of the raw audio waveform.
 - "Spectrogram": A spectrogram showing frequency over time.

O'Reilly: authoring and delivering executable books



O'Reilly: authoring and delivering executable books



The image shows a screenshot of an O'Reilly executable book. At the top, there's a browser window titled "Data visualization with Se" with the URL <https://beta.oreilly.com/learning/data-visualization-with-seaborn>. Below it is a dark-themed code editor window. The code editor has a "run" button in the bottom right corner and an "idle" status message at the bottom right. The code itself is:

```
with sns.axes_style('white'):
    sns.jointplot("x", "y", data, kind='hex')
```

O'Reilly: authoring and delivering executable books

The screenshot shows a web browser window with two tabs. The top tab is titled "Data visualization with Seaborn" and the URL is <https://beta.oreilly.com/learning/data-visualization-with-seaborn>. The bottom tab is also titled "Data visualization with Seaborn" and has the same URL. The main content area displays a snippet of Python code:

```
|with sns.axes_style('white'):
|    sns.jointplot("x", "y", data, kind='hex')
```

To the right of the code, there is a red-bordered button labeled "run". A red arrow points from the text "This is a live post!" at the bottom left towards the "run" button.

This is a live post!

O'Reilly: authoring and delivering executable books

The screenshot shows a web-based executable book interface. At the top, there are two browser tabs: one for "Data visualization with Seaborn" and another for "Data visualization with Seaborn" with a lock icon. The main content area displays a page titled "Data visualization with Seaborn" by Jake VanderPlas, dated May 7, 2011. The page content includes a snippet of Python code using the Seaborn library to create a hexagonal jointplot:

```
with sns.axes_style('white'):
    sns.jointplot("x", "y", data, kind='hex')
```

Below the code is a scatter plot generated by the code. The plot shows a positive correlation between variables x and y. The x-axis ranges from -6 to 6, and the y-axis ranges from -3 to 4. A text annotation in the plot area states: "pearsonr = 0.64; p = 4.7e-231". To the right of the plot is a vertical histogram of the y-axis data. In the bottom right corner of the main content area, there is a "done" button.

O'Reilly's Thebe: Jupyter Cells As a Service

- ❖ Jupyter code cells that can be embedded in static web pages.
- ❖ Plugin by @zischwartz based on minrk/single cell.
- ❖ Supports widgets, plots, and the wide range of Jupyter display protocols.
- ❖ Separates serving of static assets from the notebook.

Credit: @odewahn, CTO of O'Reilly

IBM Knowledge Anyhow

The screenshot shows the IBM Knowledge Anyhow Workbench interface. At the top, there's a navigation bar with the IBM logo, Home, Register, and Login links. Below that is a main content area with a title "Welcome to your Workbench." and a subtitle explaining it's a walk-up-and-use cloud environment for ad-hoc analytics and creating data products in interactive notebooks.

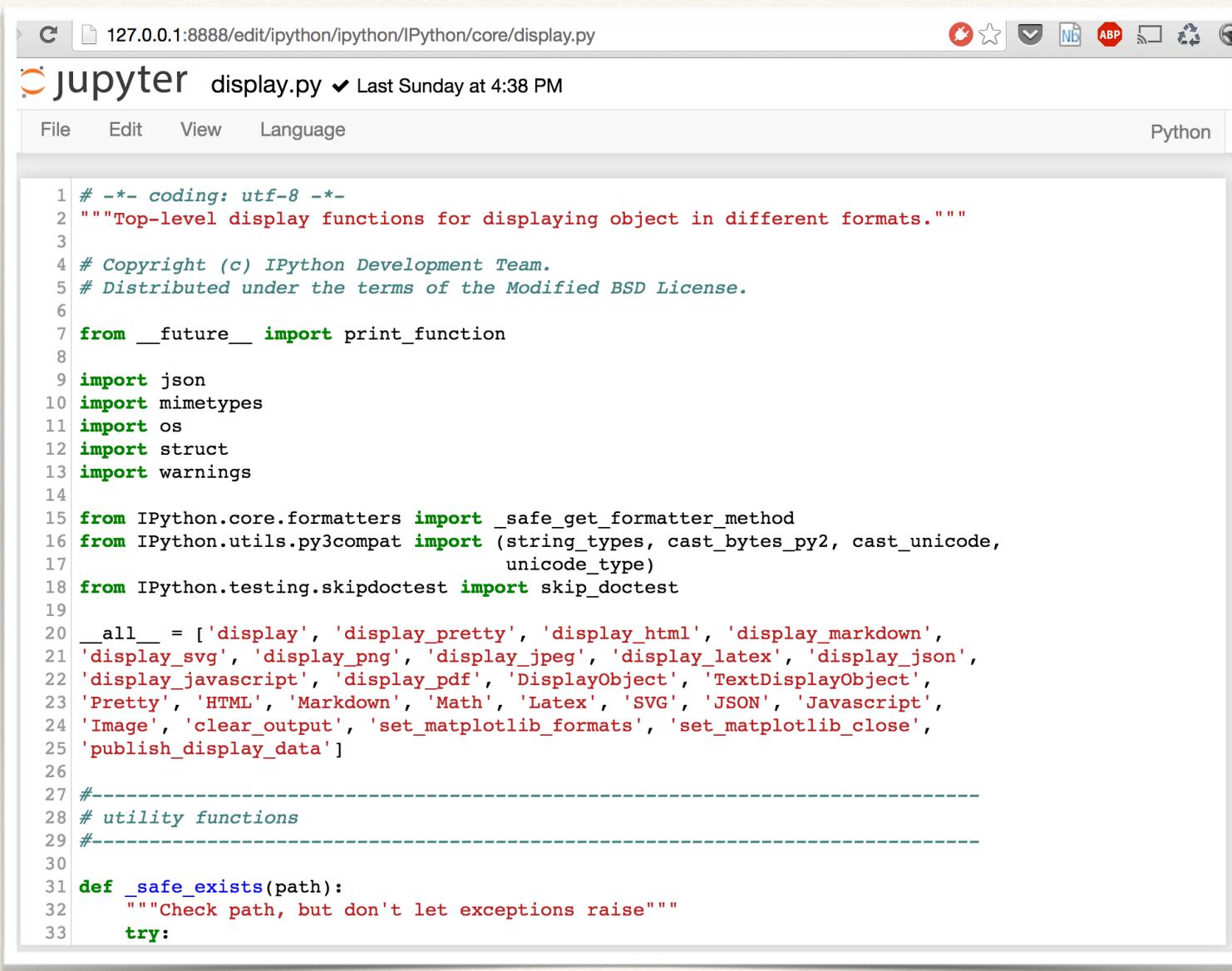
The central part of the interface is a Jupyter notebook titled "Exploration of Airline On-Time Performance.ipynb". It contains Python code for plotting arrival and departure delays by state. A bar chart titled "Number of delayed flights by state" is displayed, showing data for various US states. The chart has two bars for each state: "DEP DELAYS" (green) and "ARR DELAYS" (blue). The y-axis ranges from 0 to 25,000. The x-axis lists state abbreviations. A note at the bottom of the chart says: "Big states with big airports appear to be in the top five. But we haven't accounted for how many total flights these states service. We should plot the percentage of flights that are delayed."

To the right of the notebook is a sidebar titled "Recent" which lists recent projects and notebooks. Below the sidebar is a media player showing a video thumbnail and a play button, with the text "Background music sampled from Jahzzar/Montmartre CC SA-BY".

At the bottom left, there's a "Principal Component Analysis Plots" section with a brief description and some sample Python code for performing PCA on the Iris dataset.

On the right side of the bottom panel, there's a call-to-action: "Pick up your favorite tools." followed by a paragraph about the workbench's capabilities.

Full-page text editor



The screenshot shows a Jupyter Notebook interface with the following details:

- Title Bar:** Shows the URL `127.0.0.1:8888/edit/ipython/ipython/IPython/core/display.py` and various browser icons.
- Header:** Displays the Jupyter logo, the file name `display.py`, and the last modified time `Last Sunday at 4:38 PM`.
- Toolbar:** Includes standard file operations like File, Edit, View, Language, and a Python tab.
- Code Editor:** A large text area containing the source code for `display.py`. The code is color-coded for syntax, with numbers on the left indicating line numbers.

```
# -*- coding: utf-8 -*-
"""
Top-level display functions for displaying object in different formats.

# Copyright (c) IPython Development Team.
# Distributed under the terms of the Modified BSD License.

from __future__ import print_function

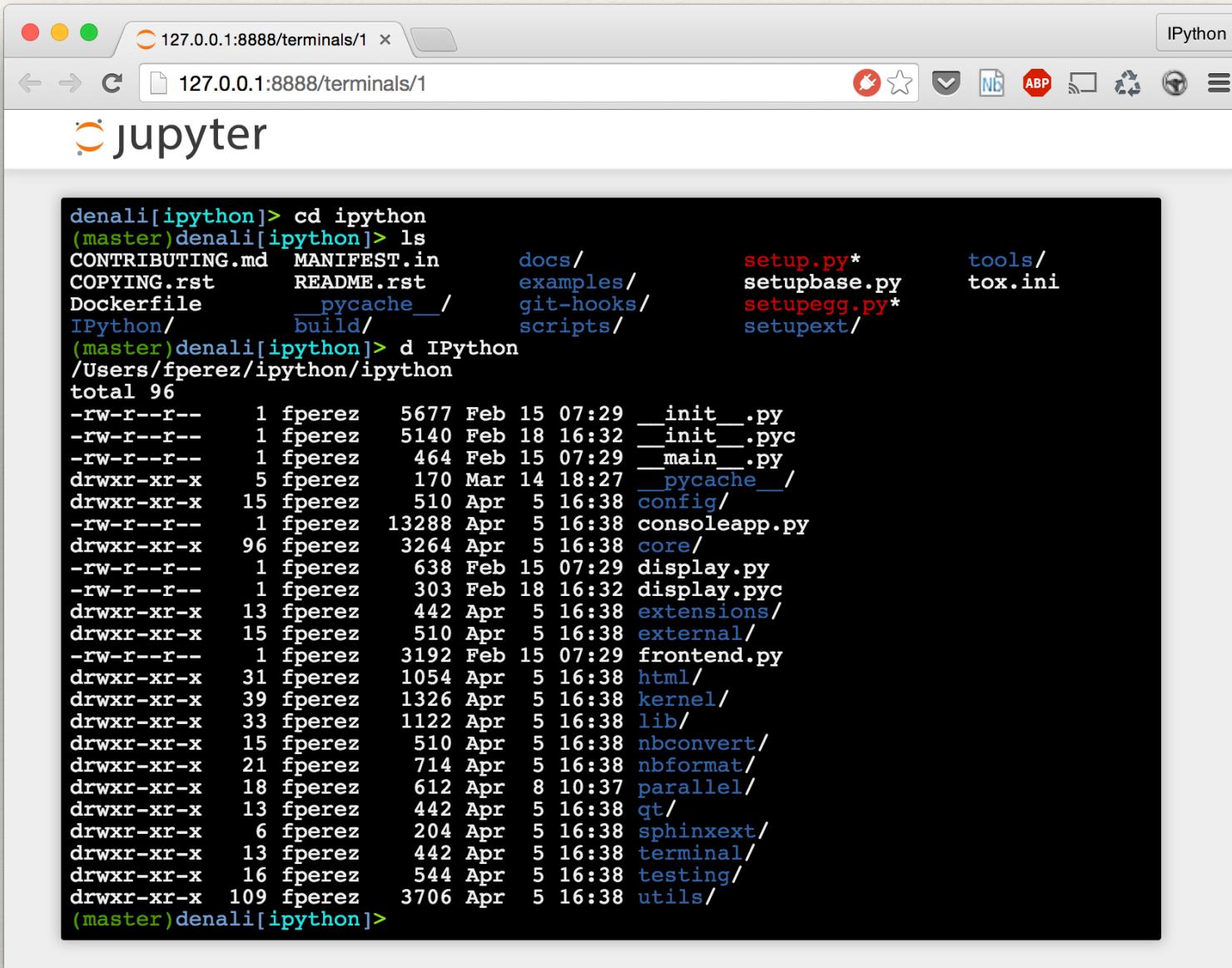
import json
import mimetypes
import os
import struct
import warnings

from IPython.core.formatters import _safe_get_formatter_method
from IPython.utils.py3compat import (string_types, cast_bytes_py2, cast_unicode,
                                    unicode_type)
from IPython.testing.skipdoctest import skip_doctest

__all__ = ['display', 'display_pretty', 'display_html', 'display_markdown',
'display_svg', 'display_png', 'display_jpeg', 'display_latex', 'display_json',
'display_javascript', 'display_pdf', 'DisplayObject', 'TextDisplayObject',
'Pretty', 'HTML', 'Markdown', 'Math', 'Latex', 'SVG', 'JSON', 'Javascript',
'Image', 'clear_output', 'set_matplotlib_formats', 'set_matplotlib_close',
'publish_display_data']

#-----
# utility functions
#-----
def _safe_exists(path):
    """Check path, but don't let exceptions raise"""
    try:
```

In-browser terminal (real-time sync)



The screenshot shows a web browser window with the URL `127.0.0.1:8888/terminals/1`. The title bar says "IPython". The content area displays a terminal session:

```
denali[ipython]> cd ipython
(master)denali[ipython]> ls
CONTRIBUTING.md  MANIFEST.in      docs/
COPYING.rst       README.rst      examples/
Dockerfile        __pycache__/_   git-hooks/
IPython/          build/         scripts/
                  tools/
                  tox.ini
(master)denali[ipython]> d IPython
/Users/fperez/ipython/ipython
total 96
-rw-r--r--    1 fperez    5677 Feb 15 07:29 __init__.py
-rw-r--r--    1 fperez    5140 Feb 18 16:32 __init__.pyc
-rw-r--r--    1 fperez     464 Feb 15 07:29 __main__.py
drwxr-xr-x    5 fperez    170 Mar 14 18:27 __pycache__/
drwxr-xr-x    15 fperez    510 Apr  5 16:38 config/
-rw-r--r--    1 fperez   13288 Apr  5 16:38 consoleapp.py
drwxr-xr-x    96 fperez    3264 Apr  5 16:38 core/
-rw-r--r--    1 fperez     638 Feb 15 07:29 display.py
-rw-r--r--    1 fperez     303 Feb 18 16:32 display.pyc
drwxr-xr-x    13 fperez    442 Apr  5 16:38 extensions/
drwxr-xr-x    15 fperez    510 Apr  5 16:38 external/
-rw-r--r--    1 fperez    3192 Feb 15 07:29 frontend.py
drwxr-xr-x    31 fperez    1054 Apr  5 16:38 html/
drwxr-xr-x    39 fperez    1326 Apr  5 16:38 kernel/
drwxr-xr-x    33 fperez    1122 Apr  5 16:38 lib/
drwxr-xr-x    15 fperez    510 Apr  5 16:38 nbconvert/
drwxr-xr-x    21 fperez    714 Apr  5 16:38 nbformat/
drwxr-xr-x    18 fperez    612 Apr  8 10:37 parallel/
drwxr-xr-x    13 fperez    442 Apr  5 16:38 qt/
drwxr-xr-x     6 fperez    204 Apr  5 16:38 sphinxext/
drwxr-xr-x    13 fperez    442 Apr  5 16:38 terminal/
drwxr-xr-x    16 fperez    544 Apr  5 16:38 testing/
drwxr-xr-x   109 fperez   3706 Apr  5 16:38 utils/
(master)denali[ipython]>
```

Google CoLab: next steps

- ❖ Google Research funding a postdoc @ Berkeley. Thanks!
- ❖ Integrate **real-time collaboration** into Jupyter architecture.
 - ❖ First, supported on **Google Drive**.
 - ❖ Then, generalize, support **other real-time backends**.



Matthias Bussonnier
@ Berkeley



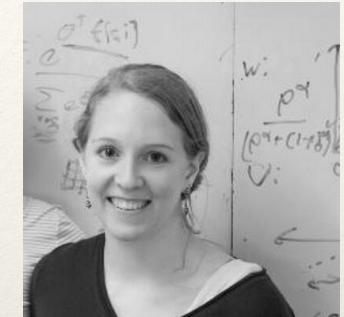
Kester Tong
@ Google

JupyterHub: multiuser support

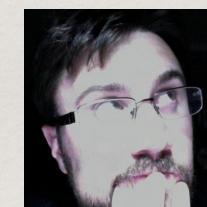
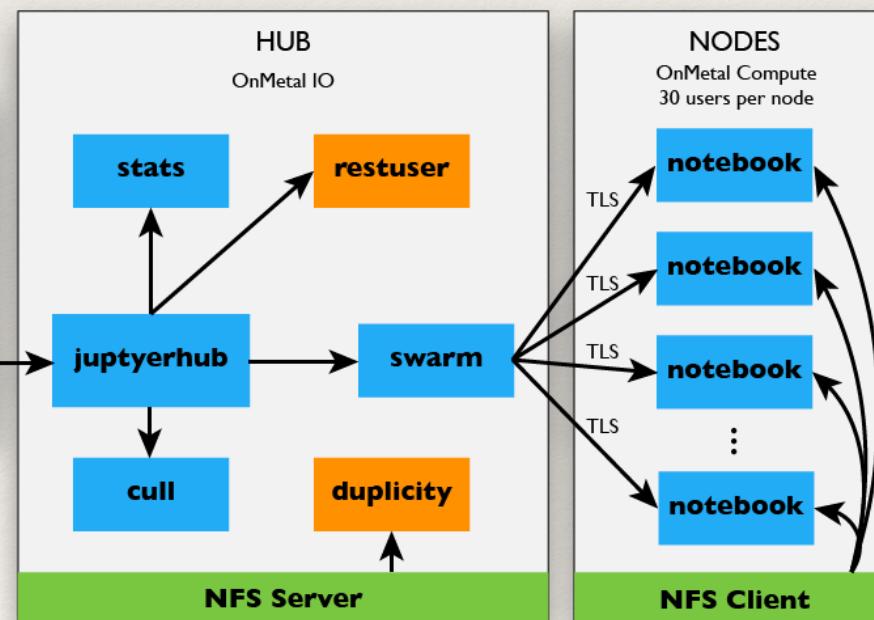
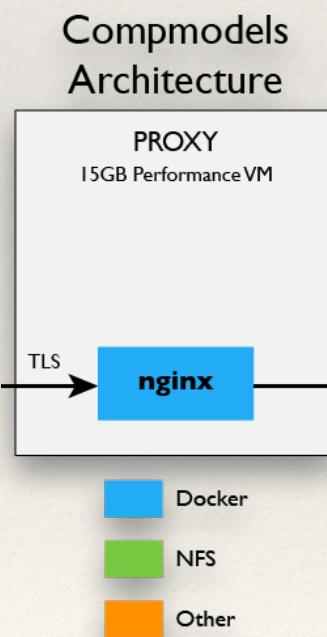
- ❖ Out of the box
 - ❖ Unix accounts
 - ❖ Local single-user notebooks
- ❖ Customizable
 - ❖ Authentication: OAuth, LDAP, etc.
 - ❖ Subprocess control: Docker, VMs, etc.

JupyterHub in Education @ Berkeley

- ❖ Computationally intensive course, ~220 students
- ❖ Fully hosted environment, zero-install
- ❖ Homework management and grading (w B. Granger)



Jess Hamrick @ Cal



K. Kelley
Rackspace M. Ragan-Kelley
Cal B. Granger
Cal Poly



<https://developer.rackspace.com/blog/deploying-jupyterhub-for-education>

Thank You!

@fperez_org fperez@lbl.gov

@ProjectJupyter @IPythonDev

Try it out at
try.jupyter.org