

Introduction to Jupyter

Presented by Ben Winjum

IDRE

February 7, 2019

Outline

- Briefly, what is the Jupyter notebook?
- Using the notebook: text and code
- The growing use of Jupyter
- Using the notebook: output and interactive widgets
- Jupyter notebooks as part of a larger Jupyter ecosystem
- Methods for sharing notebooks
 - And words of caution
- Intro to JupyterLab

```
Python 3.6.3 | packaged by conda-forge | (default, Nov  4 2017, 10:13:32)
Type 'copyright', 'credits' or 'license' for more information
IPython 7.0.0.dev -- An enhanced Interactive Python. Type '?' for help.
```

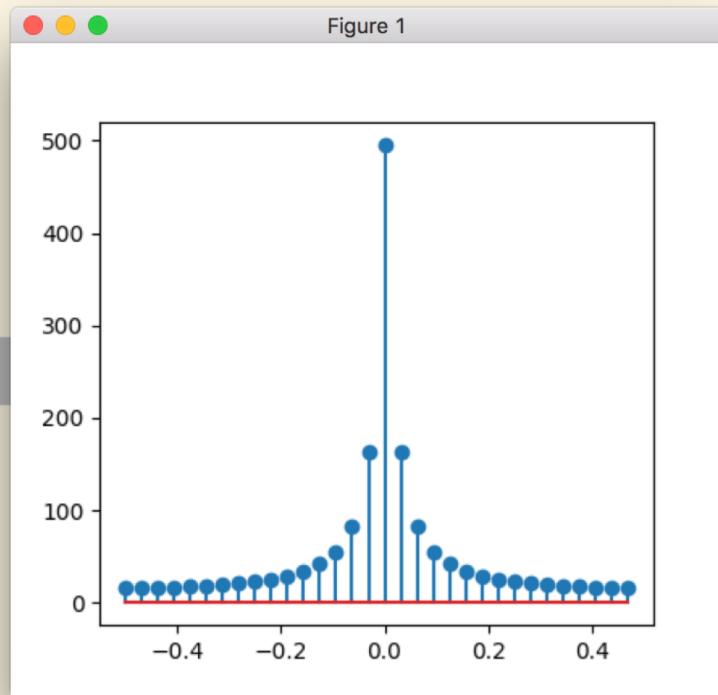
```
In [1]: from numpy.fft import *
...: from numpy import arange
...: a = arange(32)
...: A = fft(a)
...: f = fftfreq(32)

In [2]: %matplotlib tk

In [3]: from matplotlib.pyplot import stem

In [4]: stem(f, abs(A))
Out[4]: <Container object of 3 artists>

In [5]: _.
       add_callback      eventson
       baseline          get_children
       count()           get_label
```



Jupyter grew out of IPython, an interactive shell for computing with Python.

(By Mbussone - Own work, CC BY-SA 4.0, <https://commons.wikimedia.org/w/index.php?curid=68247365>)

jupyter spectrogram

(autosaved)



File Edit View Insert Cell Kernel Help

Python 3



Markdown

CellToolbar

Simple spectral analysis

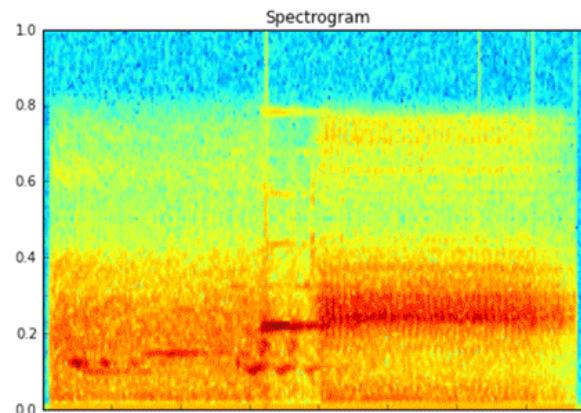
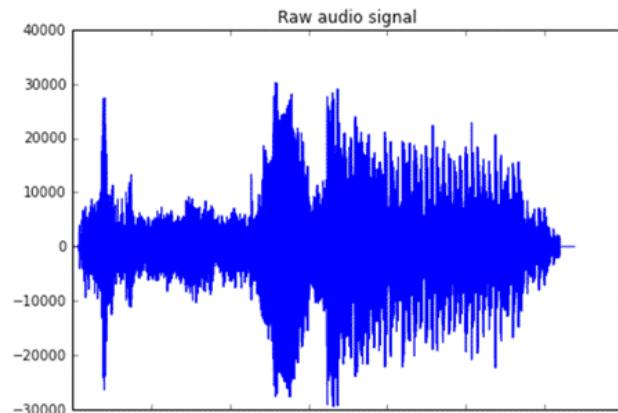
An illustration of the [Discrete Fourier Transform](#)

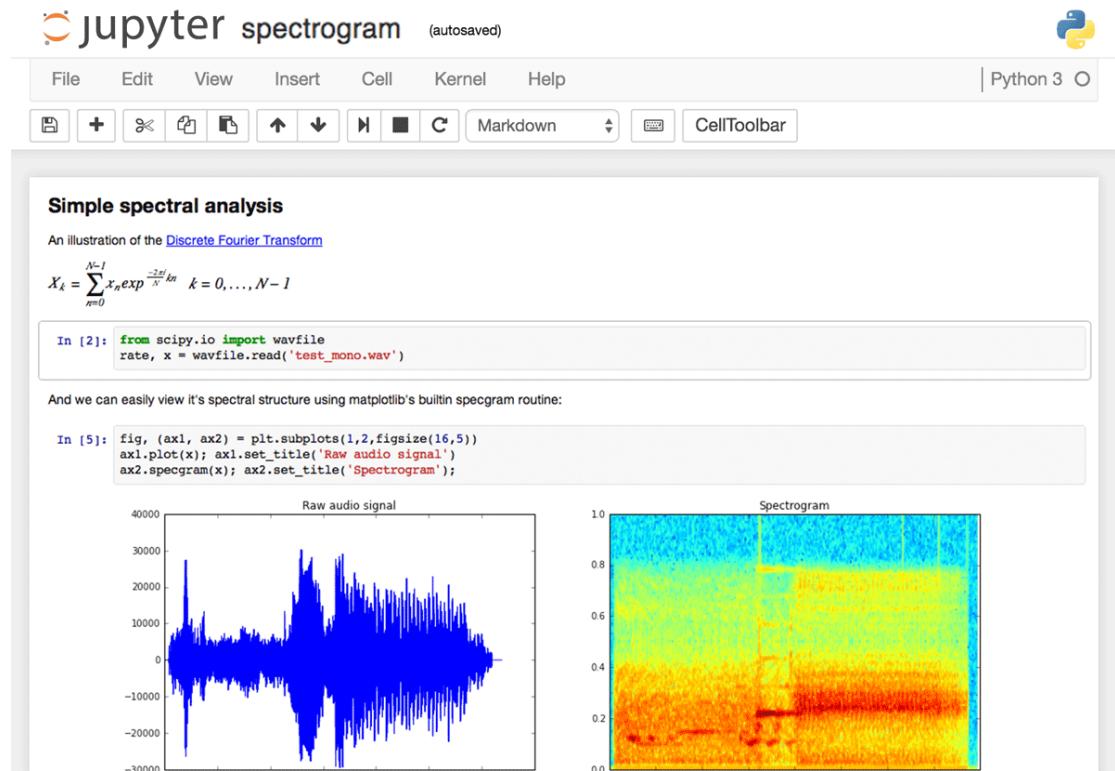
$$X_k = \sum_{n=0}^{N-1} x_n \exp \frac{-2\pi i}{N} kn \quad k = 0, \dots, N-1$$

```
In [2]: from scipy.io import wavfile  
rate, x = wavfile.read('test_mono.wav')
```

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

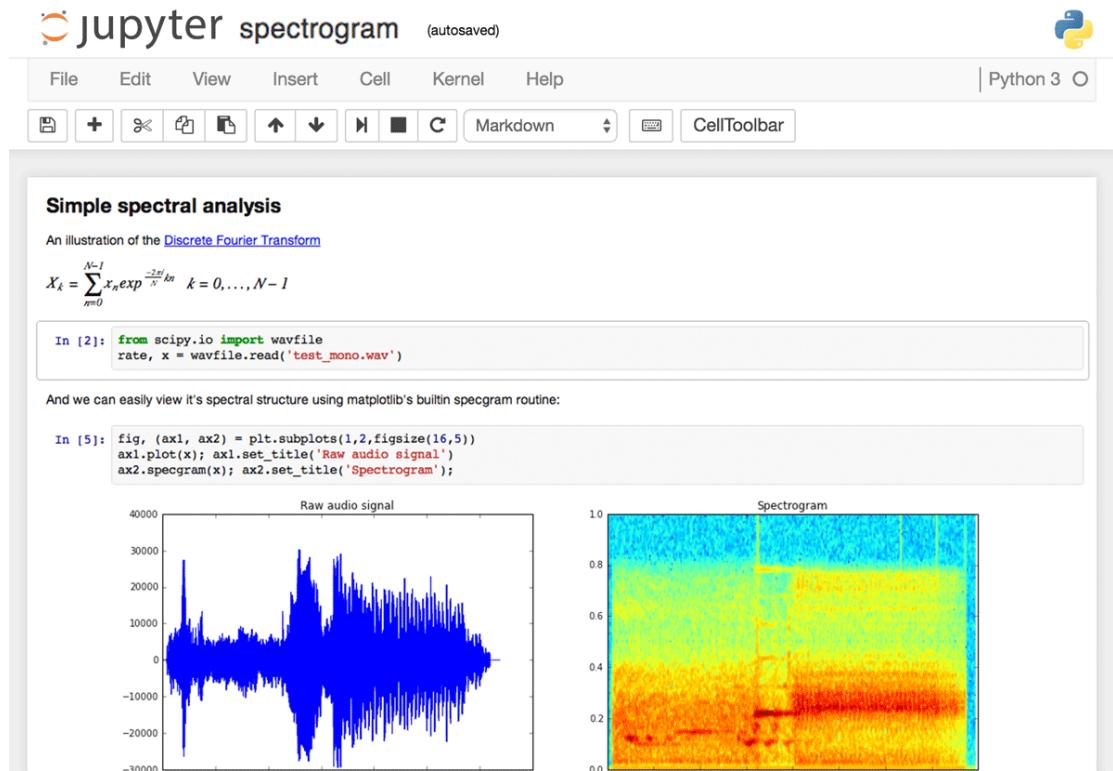
```
In [5]: fig, (ax1, ax2) = plt.subplots(1,2,figsize(16,5))  
ax1.plot(x); ax1.set_title('Raw audio signal')  
ax2.specgram(x); ax2.set_title('Spectrogram');
```





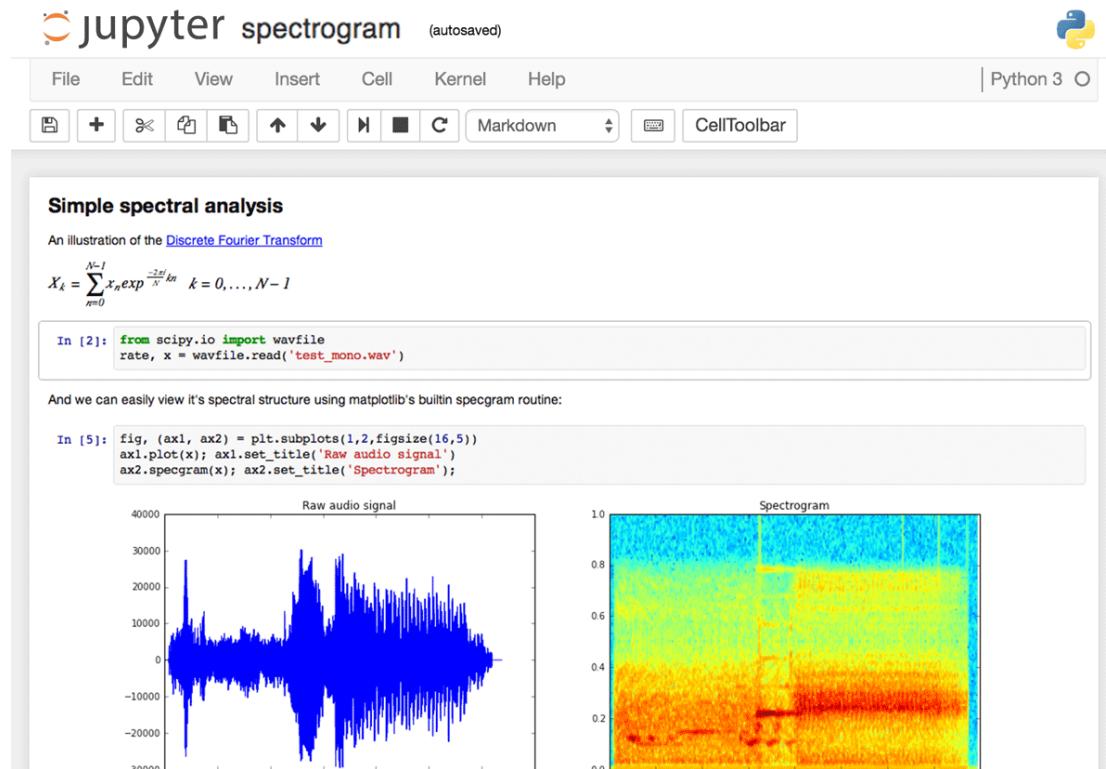
Main features of the web application

- In-browser editing for code, with automatic syntax highlighting, indentation, and tab completion/introspection.



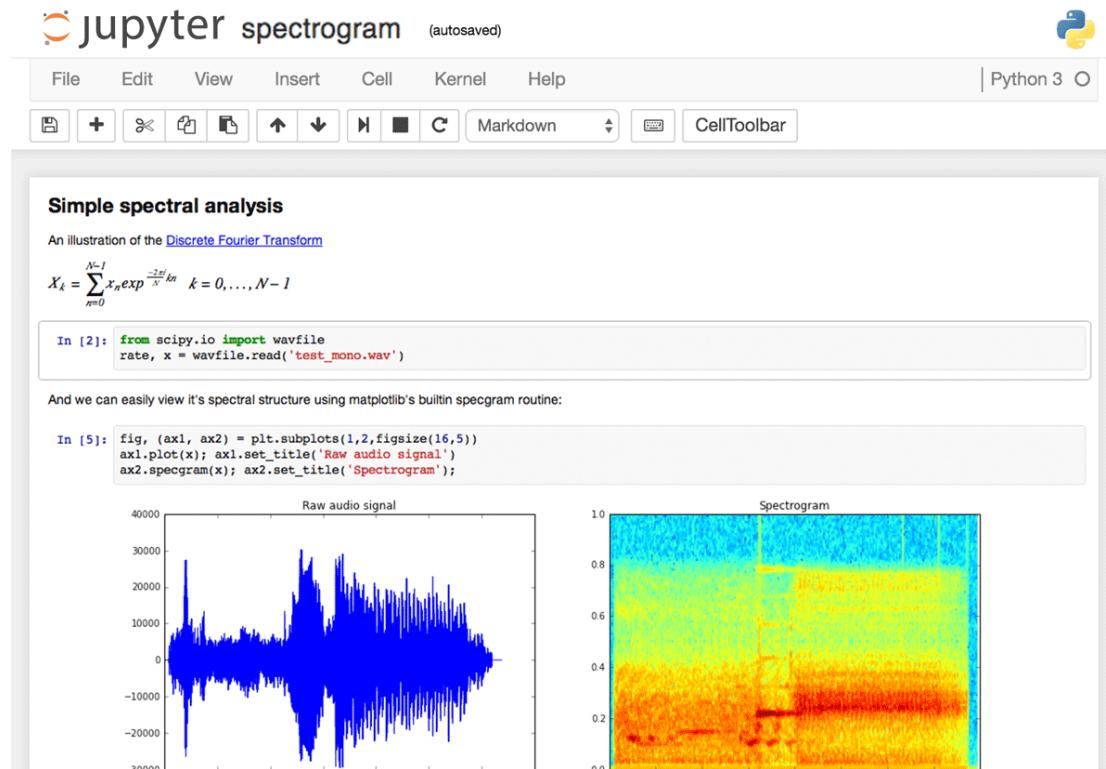
Main features of the web application

- The ability to execute code from the browser, with the results of computations attached to the code which generated them.



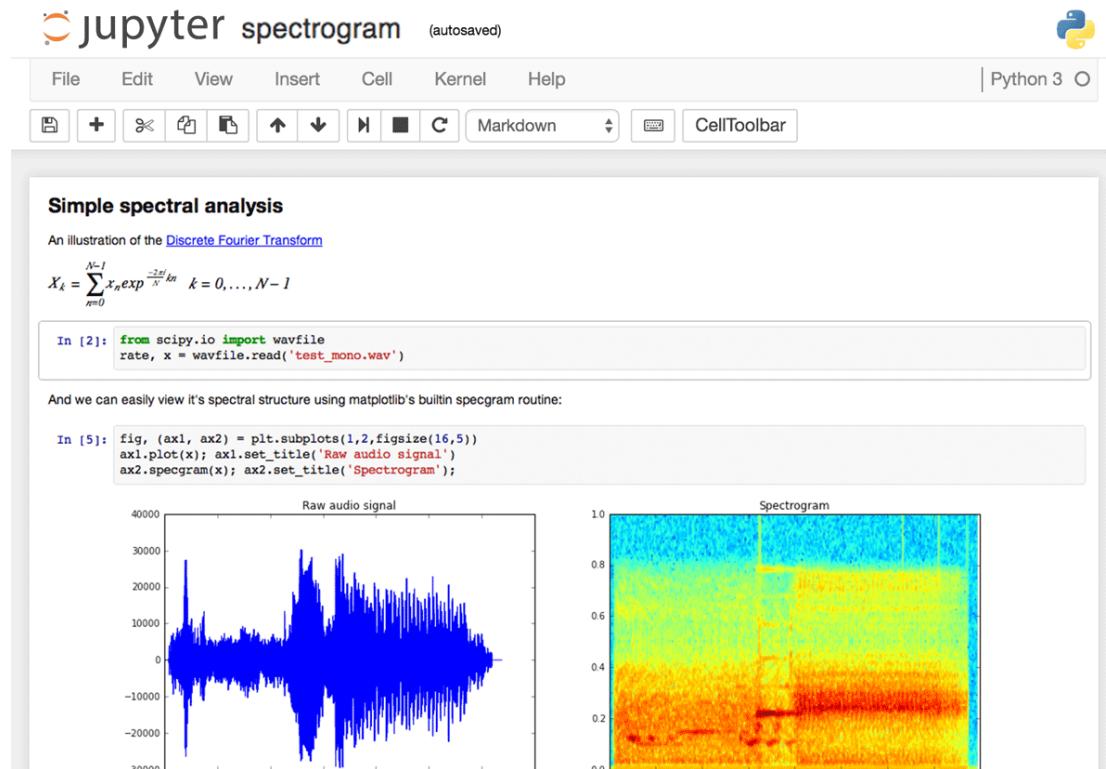
Main features of the web application

- Displaying the result of computation using rich media representations, such as HTML, LaTeX, PNG, SVG, etc. For example, publication-quality figures rendered by the matplotlib library, can be included inline.



Main features of the web application

- In-browser editing for rich text using the Markdown markup language, which can provide commentary for the code, is not limited to plain text.



Main features of the web application

- The ability to easily include mathematical notation within markdown cells using LaTeX, and rendered natively by MathJax.



Julia, Python, R
and now over 100 other languages

Image: <http://jupyter.org/>

Notebook basics

go to <https://jupyter.idre.ucla.edu>

```
$ cd work  
$ git clone https://github.com/benjum/idre-intro-to-jupyter.git
```

The Jupyter Notebook

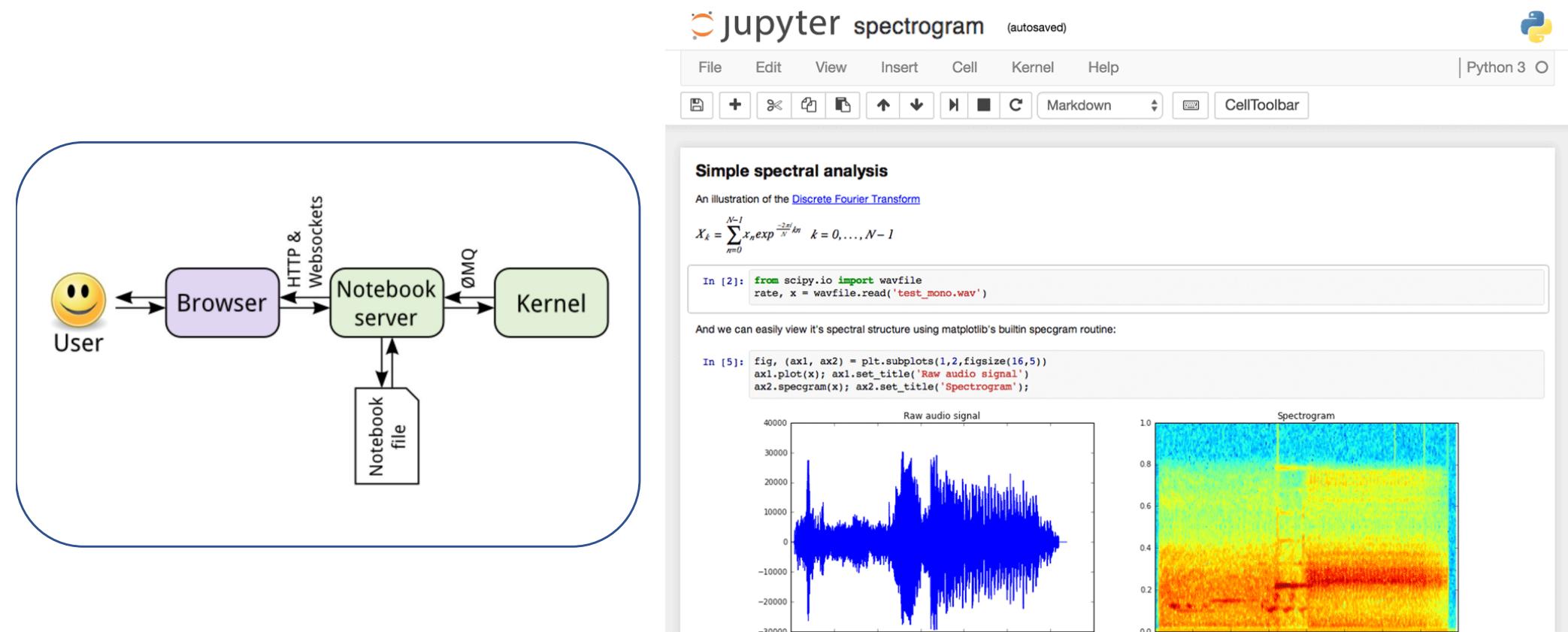


Image: https://jupyter.readthedocs.io/en/latest/architecture/how_jupyter_ipython_work.html

The Jupyter Notebook

Jupyter is comprised of several components

Front-end:

1. **Web Application:** Browser-based tool for interactive development of notebook documents
2. **Notebook Document:** A representation of all content visible in the web application, including inputs and outputs of the computations, explanatory text, mathematics, images, and rich media representations of objects. These documents are internally JSON files and are saved with the .ipynb extension. Since JSON is a plain text format, they can be version-controlled and shared with colleagues.

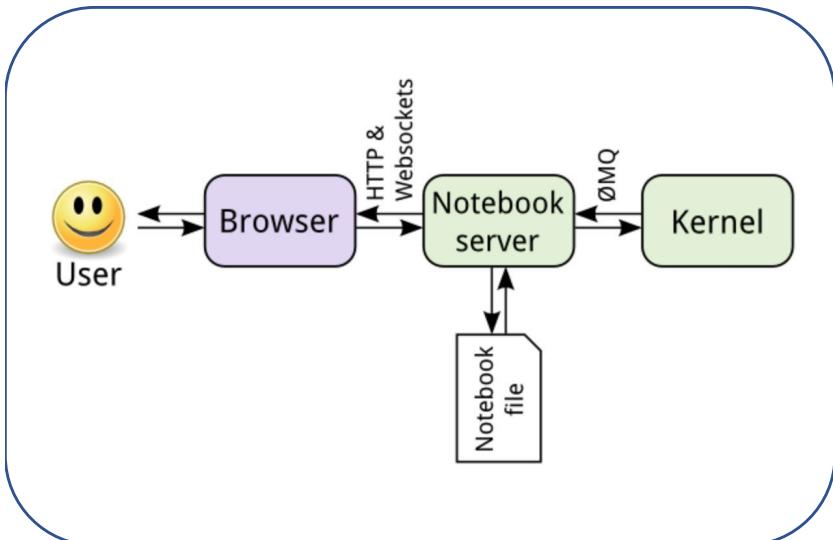


Image: https://jupyter.readthedocs.io/en/latest/architecture/how_jupyter_ipython_work.html

The Jupyter Notebook

Jupyter is comprised of several components

Back-end:

1. **Kernel:** A separate process responsible for running user code. Jupyter is capable of interfacing with many programming languages.
2. **Notebook Server:** Communicates with kernel and routes the programming language to the web browser.

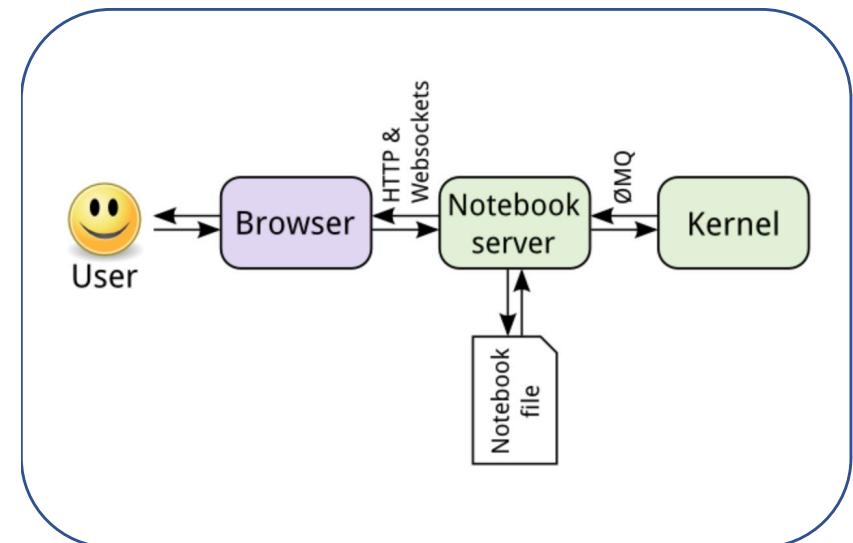


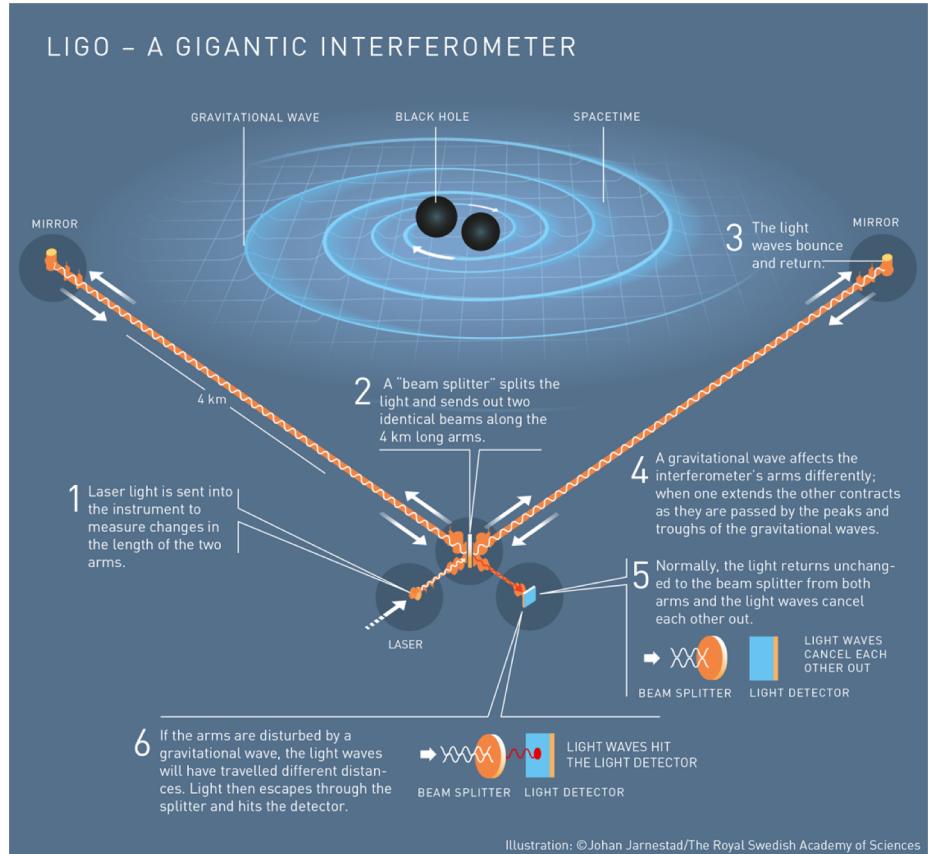
Image: https://jupyter.readthedocs.io/en/latest/architecture/how_jupyter_ipython_work.html



Supported kernels

Image: <http://jupyter.org/>

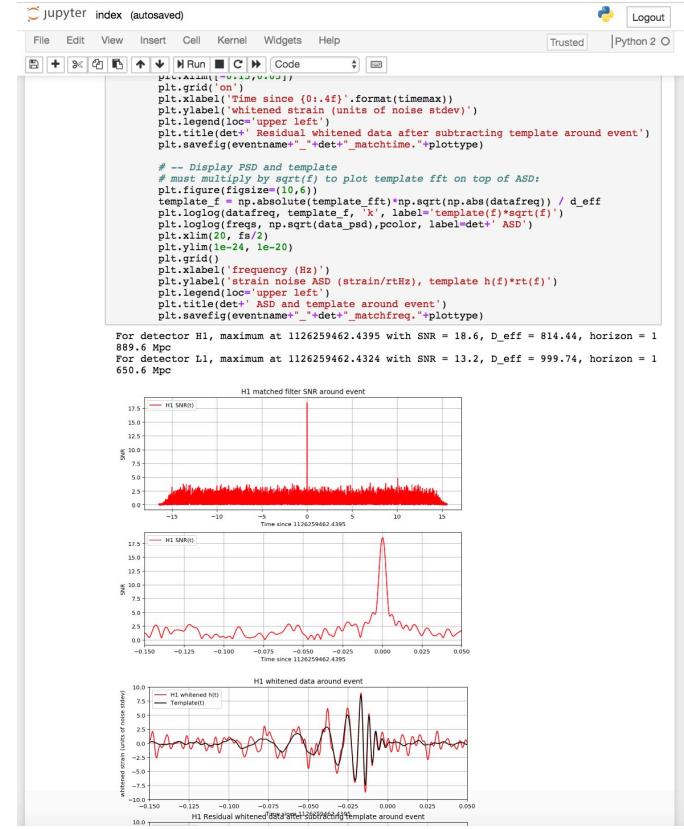
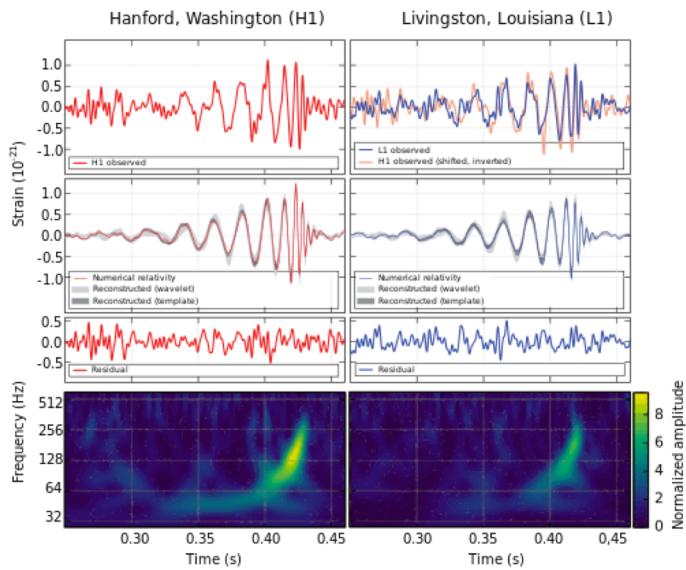
Jupyter Exploration of 2017 Physics Nobel Work



<http://physics.aps.org/featured-article-pdf/10.1103/PhysRevLett.116.061102>

<https://www.gwopenscience.org/tutorials/>

LIGO & Open Science



<http://physics.aps.org/featured-article-pdf/10.1103/PhysRevLett.116.061102>

<https://www.gw-openscience.org/tutorials/>

Jupyter notebooks engage learners

Convert from kern format to MusicXML

```
In [12]: c = converter.parse('/Users/carol/Downloads/duet/edokomuri.krn')
c.show()
```

Out[12]:

music21: a toolkit for computer-aided musicology

What is music21?

music21 is a set of tools for helping scholars and other active listeners answer questions about music quickly and simply. If you've ever asked yourself a question like, "I wonder how often measures begin with a half note?" or "I wonder what the most common note length is in this piece," or "I bet we'd know more about Renaissance counterpoint (or Indian ragas or post-tonal pitch structures or the form of minstrels) if I could write a program to automatically write more of them," then music21 can help you with your work.

How simple is music21 to use?

Extremely. After starting Python and typing `from music21 import *` you can do all of these things with only a single line of music21 code:

```
Display a short melody in musical notation:
converter.parse('lilypond/longfellow.lily').show()
```

```
Print the twelve-tone matrix for a tone row (in this case the opening of Schoenberg's Fourth String Quartet):
print( serial.rotateMatrix([1,2,3,4,5,6,7,8,9,10,11,12]) )

# or show off the cool 'Intervallic' results are already available as objects, you can type:
print( serial.intervallic.alwaysIntervallic('Paussetonenergo9')) .matrix )
```

```
Convert a file from Finale's *.kern data format to MusicXML, for editing in Finale or Sibelius:
converter.parse('/Users/catherine/Downloads/duet/edokomuri.krn').write('musicxml')
```

def closeComposition(self):
 return self._closeChordObjectWith:
 >>> chord = Chord(['C4', 'G5',
 >>> chord2 = Chord(['D4', 'A5']);
 >> print(chord2.lily.value)
 <3> "c' g'4"

 >>> newChord = copy.deepcopy(self)
 >>> tempChord = Chord(['C4', 'G5']);
 >>> tempChord._setBase();
 >>> for childChord in tempChord.pitches:
 >>> while thisPitch.ps > chordBase:
 >>> thisPitch.octave += thisChordBase
 >>> newChord.pitches = tempChord.pitches

- Get Started with music21
- Browse the [music21 API Documentation](#)
- Download the [Google Code](#)
- Get latest news and updates at the [music21 blog](#)
- Read the [Frequently Asked Questions List](#)
- Sign up for the [music21list](#) mailing list through Google Groups.



From C. Willing, JupyterCon 2017

R. Connelly and V. Gayle

An investigation of Social Class Inequalities in General Cognitive Ability in Two British Birth Cohorts

“Concern about the lack of the reproducibility of research persists across a range of academic disciplines (Nature [Editorial], 2016). There is a general appeal for extra materials to be routinely provided alongside research publications which include sufficient information for a third party to reproduce results without any additional information from the authors (Diggle, 2015; King, 1995; King, 2003). An innovative aspect of this work is that we go beyond providing the usual supplementary material and make the complete workflow openly available. We take the trailblazing step of rendering the complete workflow accessible and reproducible within a Jupyter notebook Jupyter notebooks have been used in Nobel Prize winning high-profile big science applications but are rarely used in Sociology.”

<https://github.com/RoxanneConnelly/Social-Class-Inequalities-in-General-Cognitive-Ability-in-Two-British-Birth-Cohorts>

Novel uses of Jupyter: Digital Publication (takes a long time for adoption to actually occur)

The "Paper" of the Future

-  Alyssa Goodman (Harvard University)
-  Josh Peek (Space Telescope Science Institute)
-  Alberto Accomazzi (Harvard-Smithsonian Center for Astrophysics (CFA))
-  Chris Beaumont (Harvard-Smithsonian Center for Astrophysics (CFA))
-  Christine L. Borgman (UCLA - University of California, Los Angeles)
-  Hope How-Huan Chen (Harvard University)
-  Merce Crosas (Harvard University)
-  Christopher Erdmann (North Carolina State University)
-  August Muench
-  Alberto Pepe (Authorea Team)
-  Curtis Wong (Microsoft)

show less

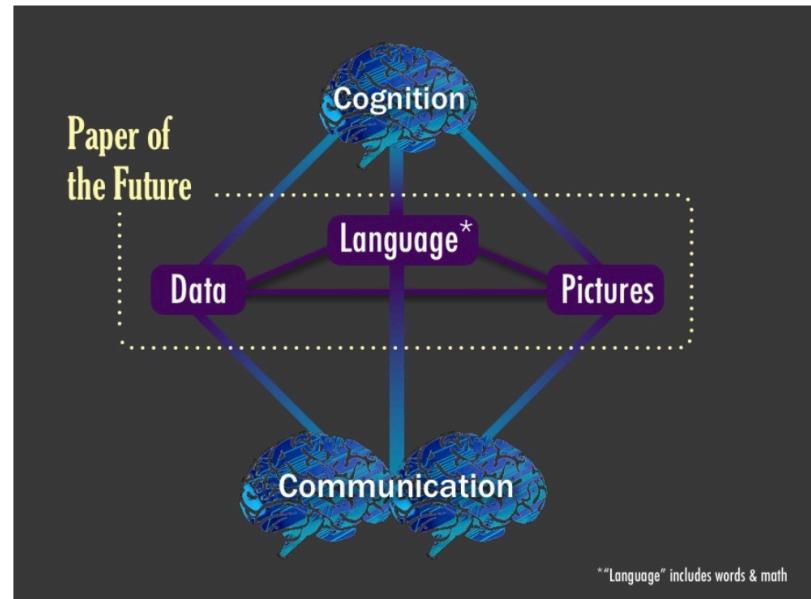


Fig. 1

*The Paper of the Future should include seamless linkages amongst **data**, **pictures**, and **language**, where "language" includes both words and math. When an individual attempts to understand each of these kinds of information, different cognitive functions are utilized: communication is inefficient if the channel is restricted primarily to language, without easy interconnection to data and pictures.*

<https://www.authorea.com/users/23/articles/8762-the-paper-of-the-future?commit=d4033594de841d252b3220927b39de4314d26409>

Notebooks: output and widgets

One more note on output and widgets

Jupyter widgets as a framework

Jupyter widgets forms a framework for representing python objects interactively. Some large open-source interactive controls based on Jupyter widgets include:

- bqplot - 2d plotting library
- pythreejs - low-level 3d graphics library
- ipyvolume - high-level 3d graphics library
- ipyleaflet - maps

The number of Jupyter users is growing very quickly

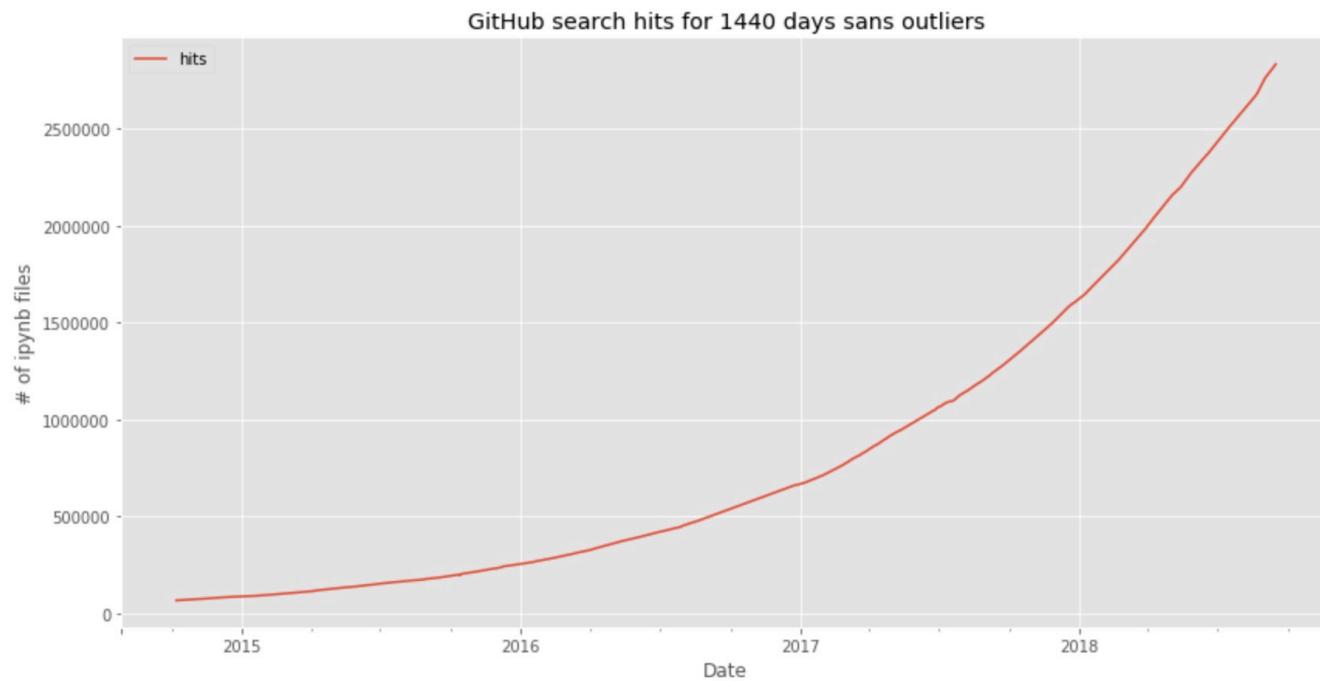


Image: <http://nbviewer.jupyter.org/github/parente/nbestimate/blob/master/estimate.ipynb>

Nevertheless, Jupyter tools are being used by some major companies, and they are finding it in their best interest to contribute to open source tools related to Jupyter



Bloomberg

Engineering

NETFLIX



Quilt

O'REILLY®



Quantopian

M Beyond Interactive: Notebook | +

A Medium Corporation [US] | https://medium.com/netflix-techblog/notebook-innovation-591ee3221233

Apps bookmarks science gateways r docker jupyterhub Software Carpentry Computational Inf... PLOS Biology: Bes... 13 M B ::

Follow

THE NETFLIX TECH BLOG

Commuter

view & share notebooks

notebook UI

interactive notebook

interactive compute

Jupyter Server → PySpark kernel → Python → Spark

output notebook

storage AWS

Amazon EFS

Amazon S3

data warehouse services

Metacat → Genie

job execution

job execution

storage

Parameterized Notebook

scheduler Meson

Meson Template

Parameterized notebook

ad hoc execution

configure

scheduled execution

store & view notebook

Commuter

read

store

read & write

store

output notebook

N

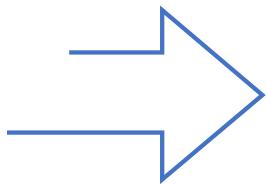
The diagram illustrates the Netflix Notebook Infrastructure. It starts with a user interface (notebook UI) which can lead to 'view & share notebooks' or 'ad hoc execution'. 'Ad hoc execution' leads to an 'interactive notebook' (represented by a document icon). This notebook can be executed via 'interactive compute' (Jupyter Server → PySpark kernel → Python → Spark) or 'scheduled compute' (Papernill → PySpark kernel → Python → Spark). The output of these executions is stored as 'output notebooks' (document icons). These notebooks can be 'read' from storage (AWS services: Amazon EFS and Amazon S3) or 'stored' back into storage. The infrastructure also includes a 'data warehouse services' component (Metacat → Genie) which interacts with the scheduled compute path. A 'scheduler' (Meson) is used to 'configure' the system and handle 'scheduled execution' of parameterized notebooks (Parameterized Notebook). The Parameterized Notebook is generated from a 'Meson Template' using the Meson scheduler. The entire system is monitored by a 'Commuter'.

Notebook Infrastructure at Netflix

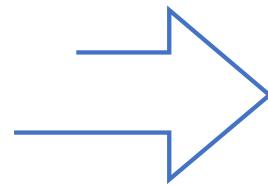
<https://medium.com/netflix-techblog/notebook-innovation-591ee3221233>

PayPal Notebooks Open Source Plans

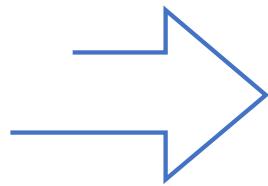
Slide from JupyterCon 2018



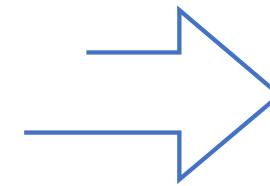
- PPMagics: available **now**
- Gimel: available **now**



- Airflow integration
- Github sharing
- Project collaboration
- Tableau integration



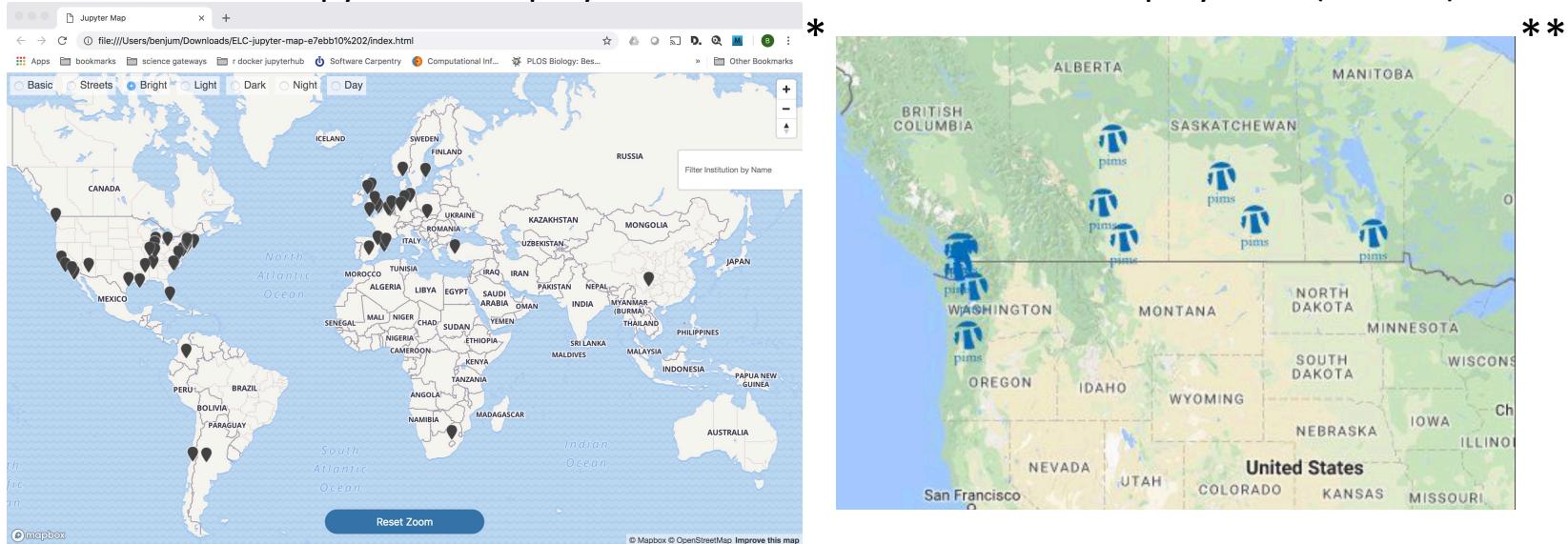
- Config UI
- Dataset browser
- Pipelines



- Data Science Workbench

Jupyter is being used by academic institutions of very high caliber all around the world and in many different disciplines

Institutional JupyterHub deployments, as well as a federation deployment (SYZYGY)



It's actually developing so quickly that many institutions are not on this map.... including UCLA and UC Berkeley!

* Institutional deployment map generated from data at <https://github.com/ELC/jupyter-map/>

** Canadians Land on Jupyter, JupyterCon 2018, <https://conferences.oreilly.com/jupyter/jup-ny/public/schedule/detail/68396>

Jupyter Projects

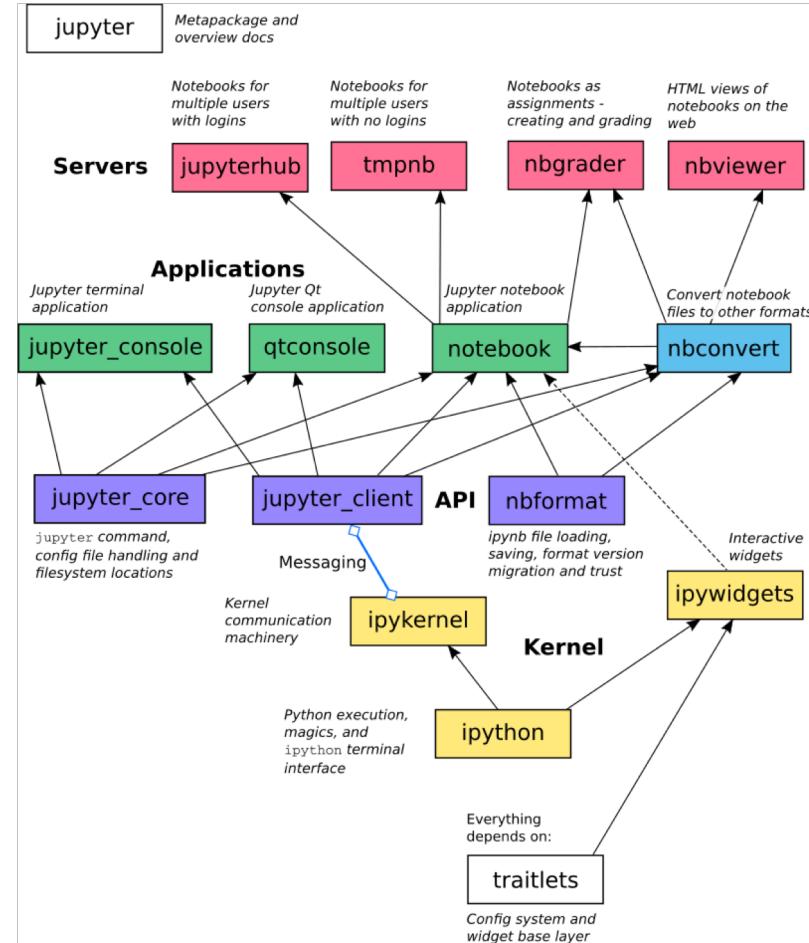
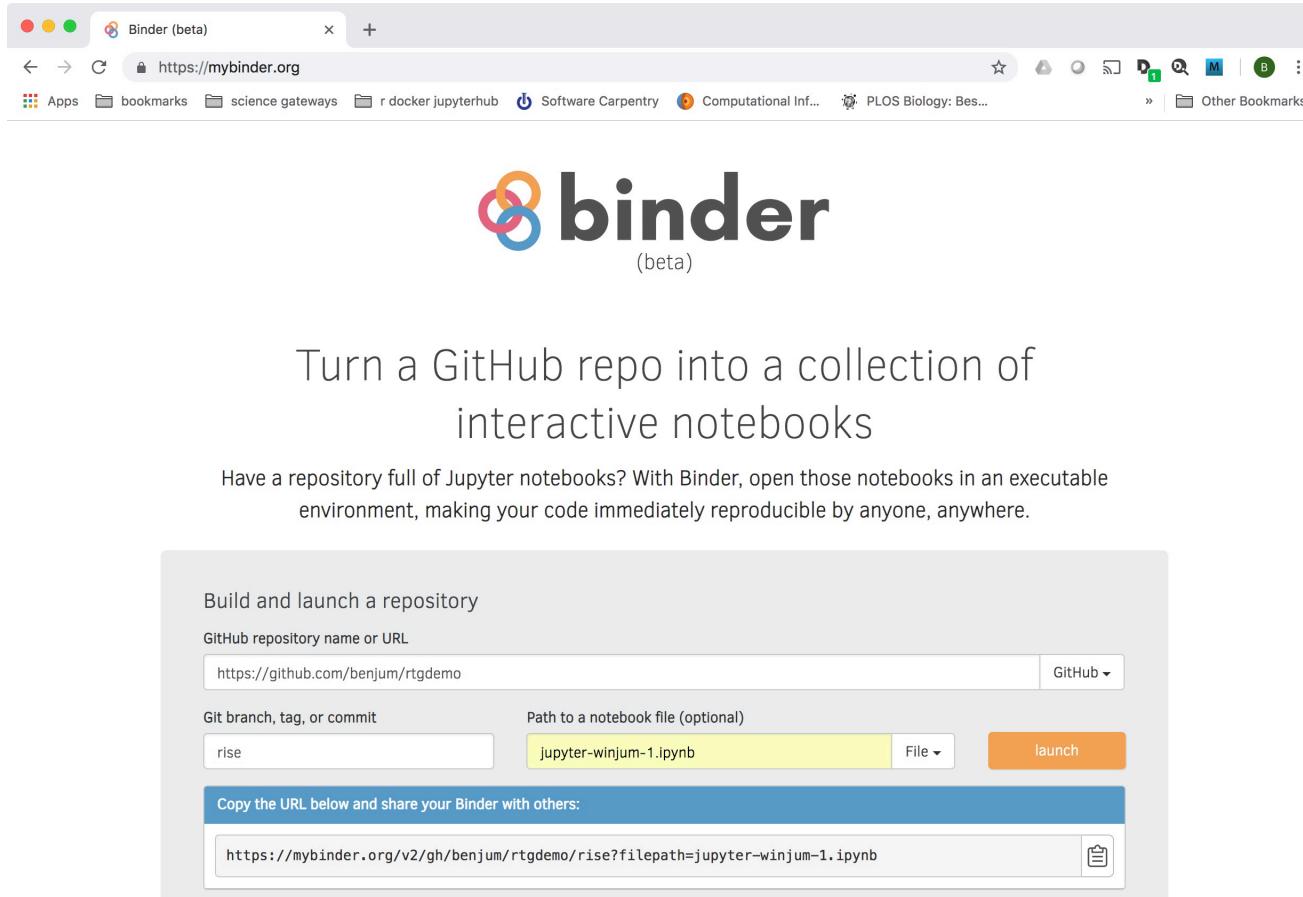


Image: https://jupyter.readthedocs.io/en/latest/architecture/visual_overview.html

How to share

- jupyter nbconvert --to html mynotebook.ipynb
- Or pdf or .py
- Via menu
- Binder
- nbviewer.jupyter.org

<https://github.com/jupyter/nbconvert>



Example taken from <https://mybinder.org/>

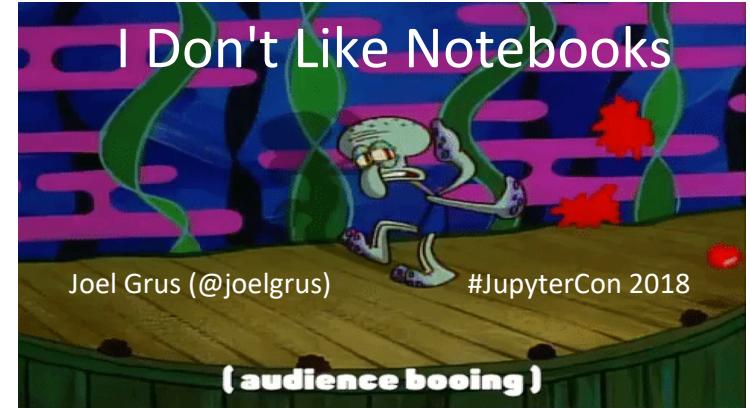
Not all uses of Jupyter are for clearly documented research and education

A. Rule, A. Tabard, J. Hollan. *Exploration and Explanation in Computational Notebooks*. ACM CHI Conference on Human Factors in Computing Systems, Apr 2018, Montréal, Canada.

- In a scan of > 1 million notebooks on GitHub, 1/4 had no explanatory text
- In analyzing 200 academic notebooks, < 40% discussed reasoning or explained results
- In interviews with 15 academic data analysts, most considered computational notebooks personal, exploratory, and messy

“Why not to use Jupyter”

- Hidden state and out-of-order execution
- Notebooks are difficult for beginners
- Notebooks encourage bad habits
- Notebooks discourage modularity and testing
- Jupyter’s autocomplete, linting, and way of looking up the help are awkward
- Notebooks encourage bad processes
- Notebooks hinder reproducible + extensible science
- Notebooks make it hard to copy and paste into Slack/Github issues
- Errors will always halt execution
- Notebooks make it easy to teach poorly
- Notebooks make it hard to teach well



<http://blog.revolutionanalytics.com/2018/09/notebooks-literate-programming.html>

So Beware!

- If you interact with others' notebooks (or even your own past notebooks), there can be hidden state due to:
 - Sloppiness
 - Out-of-order execution
 - Different libraries
 -

Different ways to view notebooks

- Lab notebook
 - Single author... and meant primarily for single viewer
 - Can be split into parts before they get too long
 - Can be split into different topics
 - Not meant to be anything other than place for experimentation and development
- Deliverable report
 - Single- or team-authored
 - Meant to share
 - Fully polished
 - Store the final analysis and outputs
- Interactive playground
 - Carefully crafted educational narratives to invite interaction
 - Interleaving regular notebook cells with test cells

Novel uses of Jupyter: Can Enable Data Re-use

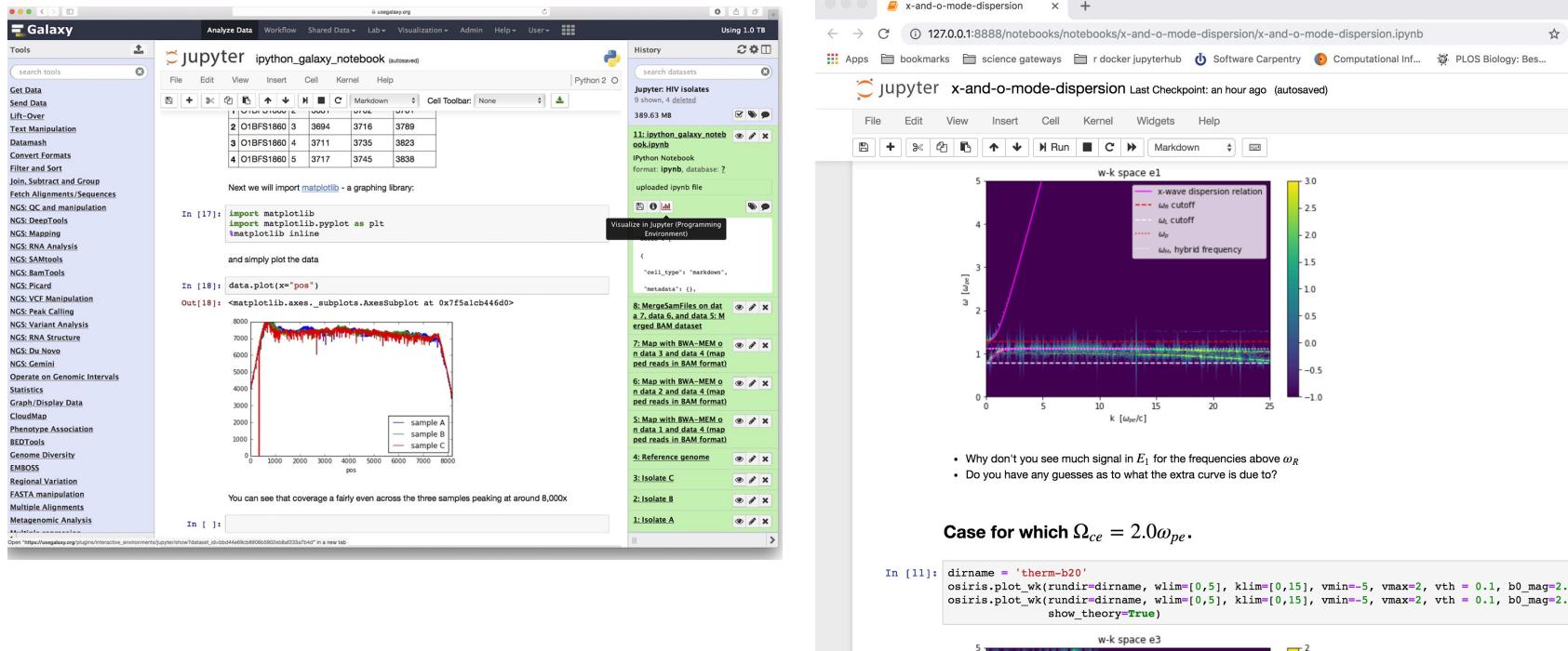
Simultaneous Matrix Diagonalization for Structural Brain Networks Classification

“... We also tried the proposed approach on UCLA Autism dataset [23] and UCLA APOE-4 dataset [2,3]. We note that ...”

“For the numerical experiments, we used Python programming language and Jupyter notebook environment. We did matrix calculations and numerical analysis using NumPy, SciPy, NetworkX and igraph libraries. The main classification pipeline was implemented using scikit-learn library [20], which we used for all the classifiers except GBDT. For Gradient Boosted Decision Trees, we used xgboost library [5]. The code is available from authors upon request.”

Novel uses of Jupyter: Integration with Research Software

Used as a data analysis tool of research results, or as a medium in which to run simulations



<http://blogs.nature.com/naturejobs/2017/08/01/techblog-jupyter-joins-the-galaxy/>

Jupyter and Education



Educational Examples

- Interactive notes
 - [12 Steps to Bioinformatics](#), Lorena Barba
- Online textbooks
 - [An Introduction to Applied Bioinformatics](#), Greg Caporaso
- Notebook projects (using research software for education)
 - [JupyterPIC](#) for UCLA EE/Phys M185

Jupyter and HPC



Connecting Notebooks with HPC Clusters

- University HPC Clusters
 - Like UCLA's Hoffman2
 - <https://www.hoffman2.idre.ucla.edu/access/jupyter-notebook/>
- National Supercomputing Centers
 - Like DOE's NERSC (National Energy Research Scientific Computing Center)
 - <http://www.nersc.gov/users/data-analytics/data-analytics-2/jupyter-and-rstudio/>

JupyterHub

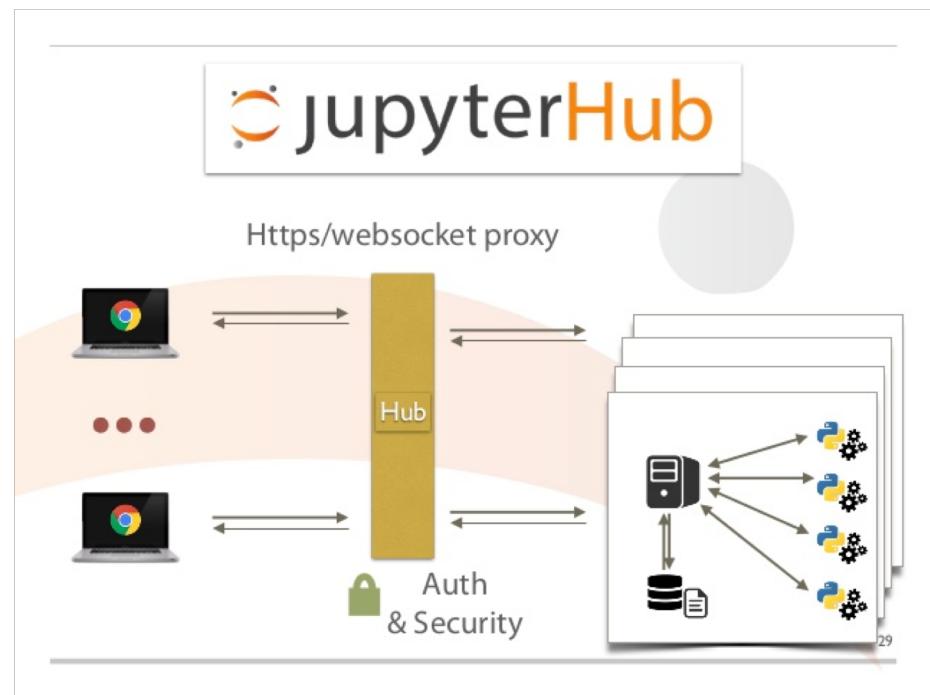
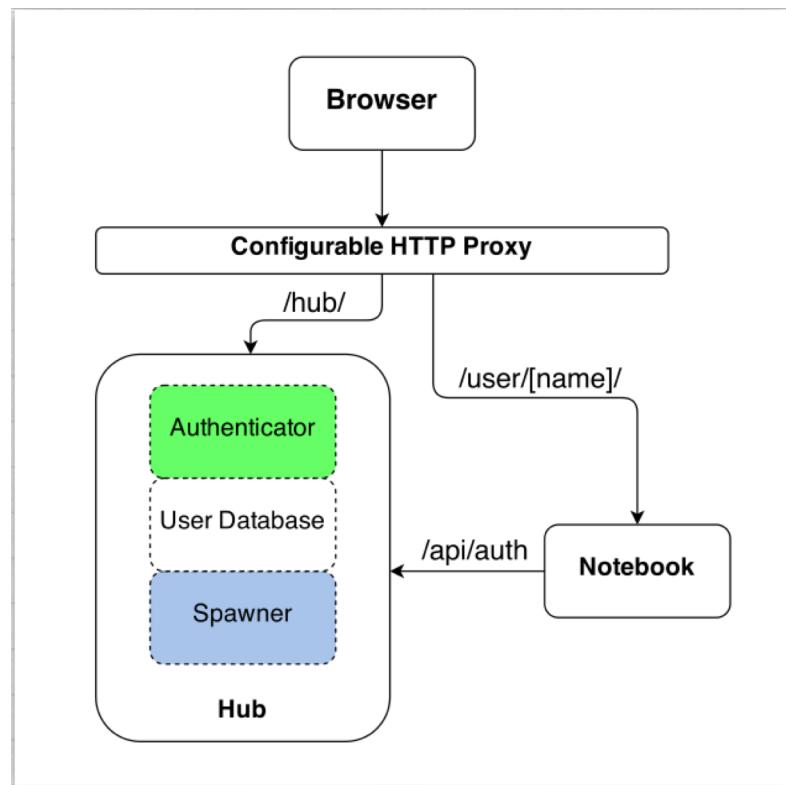


Image: <https://jupyterhub.readthedocs.io/en/stable/>

Next Generation: JupyterLab

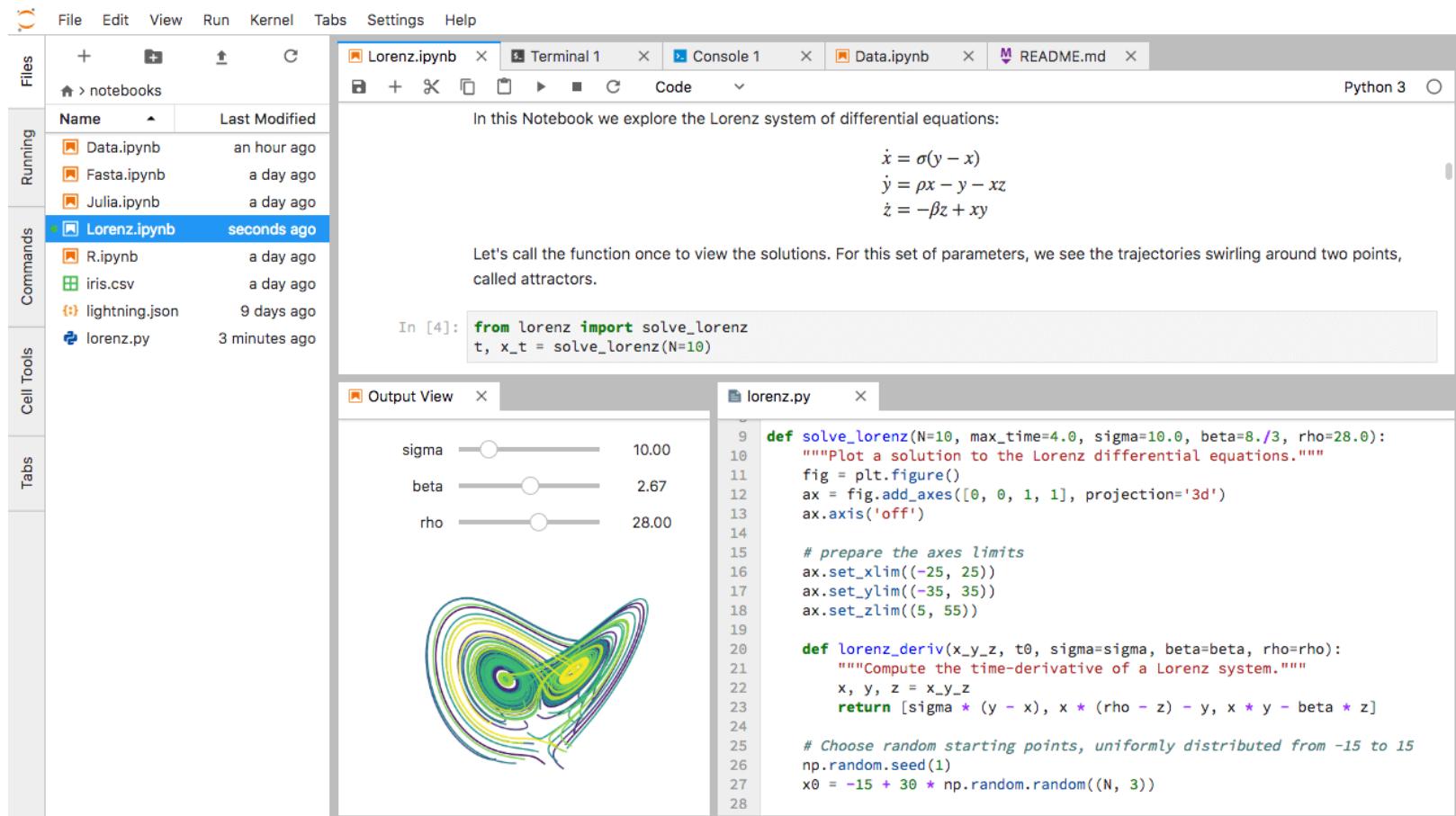


Image: <https://jupyterlab.readthedocs.io/en/stable/>

Notebooks: in JupyterLab