Jupyter in HPC

Feb 28th, 2018

Matthias Bussonnier

bussonniermatthias@gmail.com
GitHub: @carreau
Twitter: @mbussonn

About Me

Matthias Bussonnier

- A Physicist/Bio-Physicist
- Core developer of IPython/Jupyter since 2012
  - Co-founder, and Steering Council member
- Post doctoral Scholar on Jupyter at BIDS

Demo(s)

Just to find bugs and make things crash

Webinar & Outline

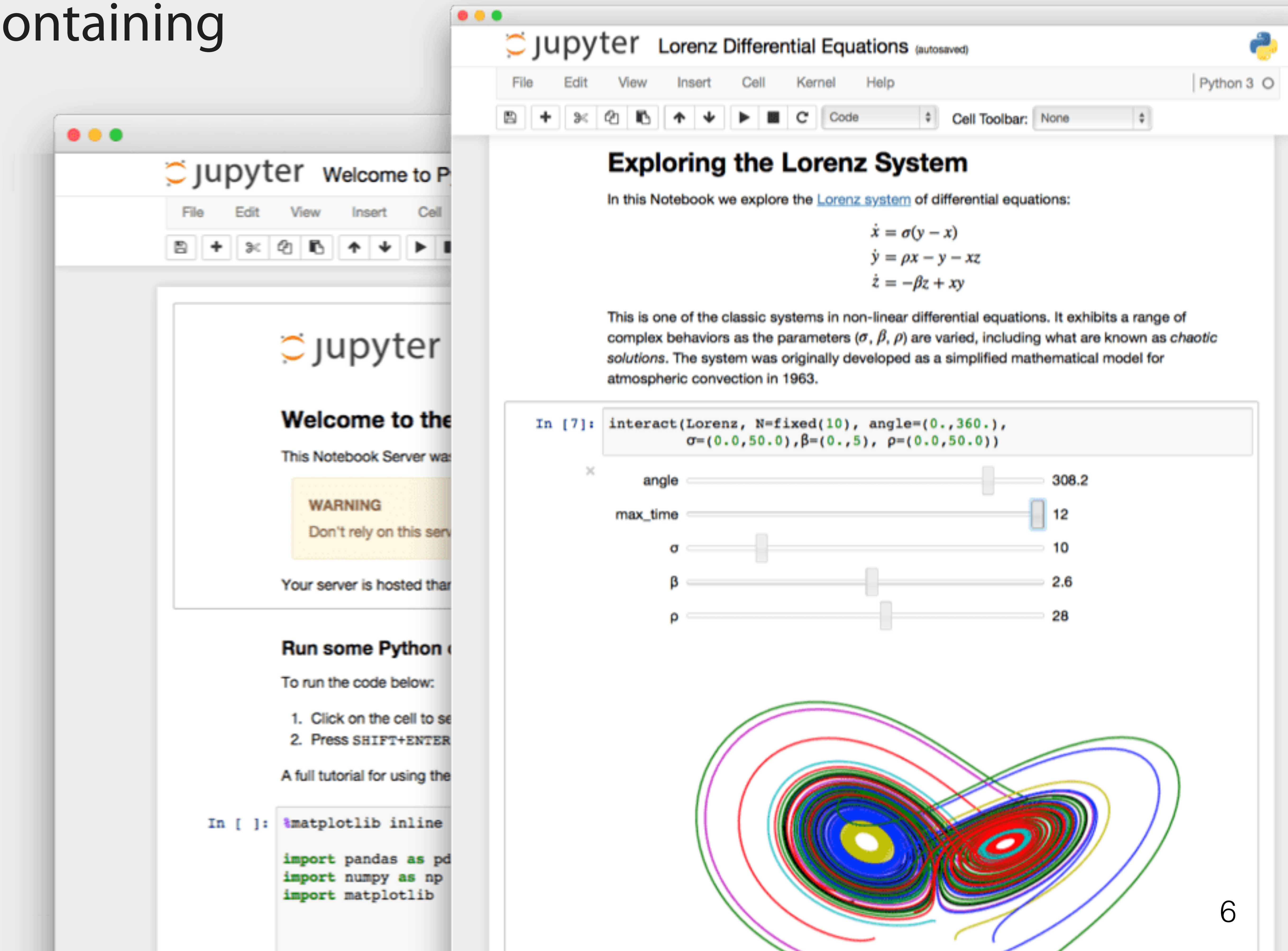
- This webinar will be in 3 parts
  - Overview of what is Jupyter + HPC
  - Use case : Suhas Somnath
  - Use case : Shreyas Cholia
- Outline Part 1
  - From IPython to Jupyter
  - What is Jupyter
  - Jupyter Popularity
  - Some Jupyter Usage

From IPython to Jupyter

- 2001: Fernando Perez Wrote “**IPython**”
  - Create IPython for Interactive Python with prompt number, gnu plot integration
  - Replace a bunch on perl/make/C/C++ files with only Python.
- 2011: QtConsole
- 2012: Birth of current **Notebook** (6th prototype)
  - Make IPython “network enabled”
  - Made possible by mature web tech.
- 2013: First non-Python (**Julia)** kernel
- 2014: we **renamed** the Python-Agnostic part to **Jupyter**.
- 2018: several millions users & **JupyterLab** released

What is Jupyter

- Mainly Known for **The Notebook**
  - Web server, a web app, load .ipynb (json), containing code, narrative, math and results.
  - Attached to a **Kernel** doing computation.
- Results can be:
  - Static (Image)
  - Interactive (client-side scoll/pan/brush)
  - Dynamic (Call back into Kernel)



Welcome to the

This Notebook Server wa

WARNING

Don't rely on this serv

Your server is hosted thar

Run some Python

To run the code below:

1. Click on the cell to se
2. Press SHIFT+ENTER

A full tutorial for using the

```
In [ ]: %matplotlib inline

        import pandas as pd
        import numpy as np
        import matplotlib
```

Exploring the Lorenz System

In this Notebook we explore the Lorenz system of differential equations:

$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = \rho x - y - xz$$
$$\dot{z} = -\beta z + xy$$

This is one of the classic systems in non-linear differential equations. It exhibits a range of complex behaviors as the parameters (σ, β, ρ) are varied, including what are known as *chaotic solutions*. The system was originally developed as a simplified mathematical model for atmospheric convection in 1963.

```
In [7]: interact(Lorenz, N=fixed(10), angle=(0.,360.),
                 σ=(0.0,50.0),β=(0.,5), ρ=(0.0,50.0))
```

angle 308.2

max_time 12

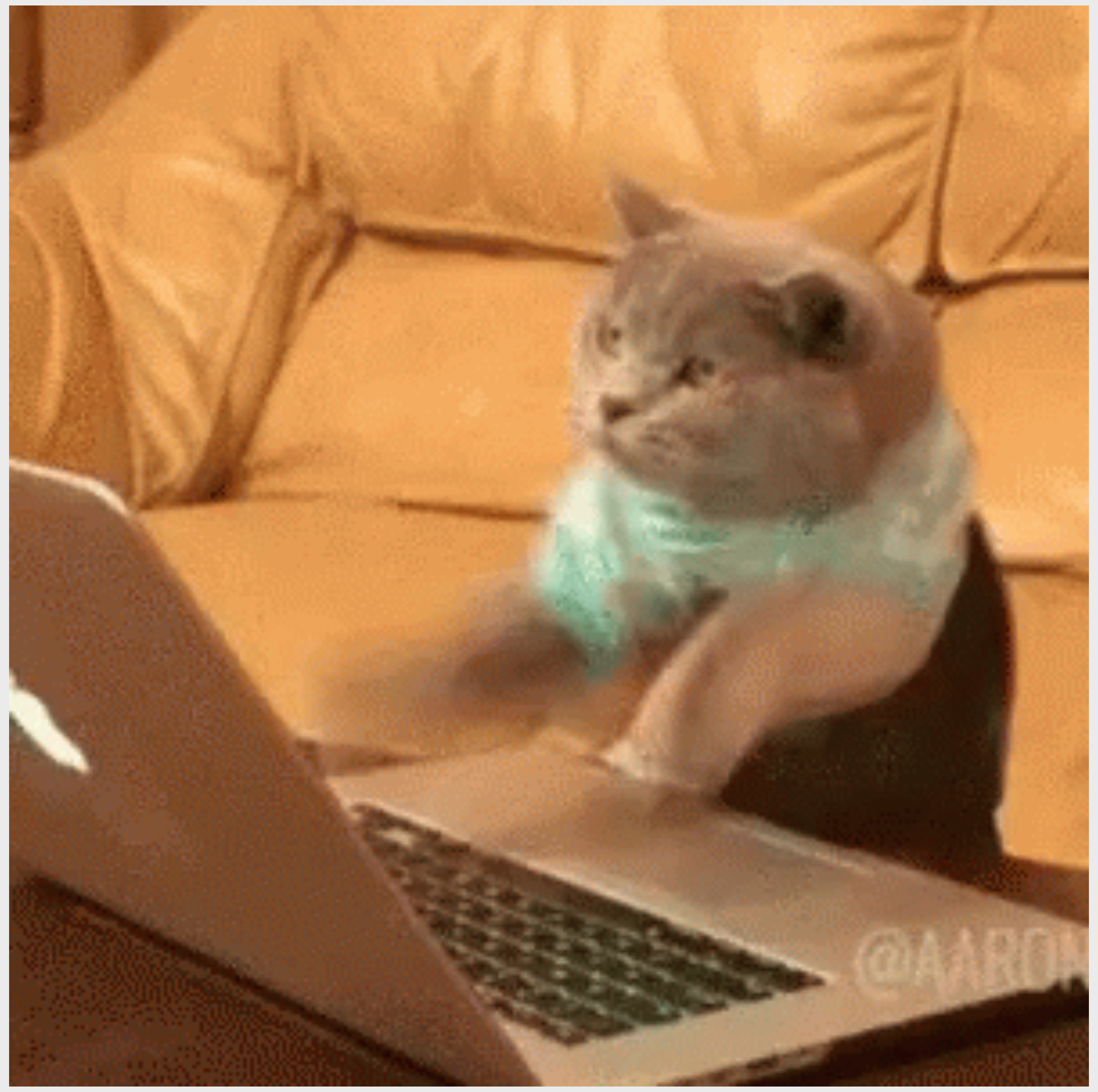
σ 10

β 2.6

ρ 28

Focused on Exploratory Programming

- IPython was designed for exploratory programming, as a REPL (Read Eval Print Loop) and grew popular, especially among scientist who loved it to explore.



> “IPython have weaponized the tab [completion] key”
> – Fernando Pérez

Open Organisation

- Organisation with Open Governance (https://GitHub.com/jupyter/governance)
- Funded by Grants and Donations, and Collaborations

NUMFOCUS

OPEN CODE = BETTER SCIENCE

GORDON AND BETTY MOORE FOUNDATION

Alfred P. Sloan FOUNDATION

THE LEONA M. AND HARRY B. HELMSLEY CHARITABLE TRUST

Protocols and Formats

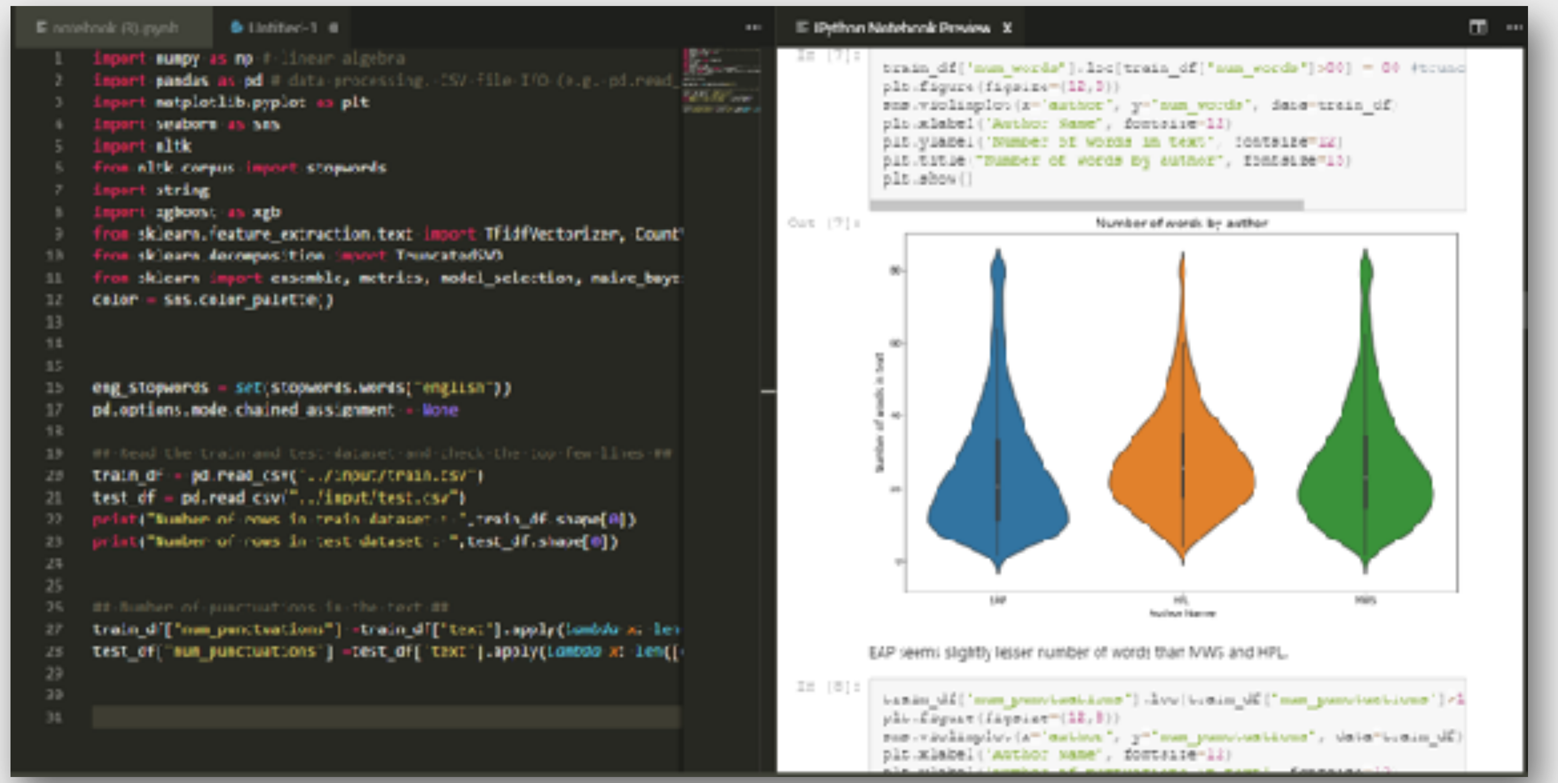
- Jupyter is also a set of **Protocols and Formats** that reduce the **N-frontends** × **M-backends** problem to a **M-Frontends + N-backends**,
  - Open, Free and Simple.
    - JSON (almost) everywhere
    - Notebook document format,
    - Wire protocol
  - Thought for Science and **Interactive** use case.
    - Results embedded in documents no "Copy past" mistake.
    - Scale from Education to HPC jobs.

Ecosystem

Frontends: Notebook, JupyterLab, CLI, *Vim, Emacs, Visual Studio Code, Atom, Nteract, Juno...*

Kernels: Python, *Julia, R, Haskell, Perl, Fortran, Ruby, Javascript, C/C++, Go, Scala, Elixir... 60+*

Building Blocks: Nbformat, JupyterHub, Kernel Gateway...



```python
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_
import matplotlib.pyplot as plt
import seaborn as sns
import nltk
from nltk.corpus import stopwords
import string
import xgboost as xgb
from sklearn.feature_extraction.text import TfidfVectorizer, Count
from sklearn.decomposition import TruncatedSVD
from sklearn import ensemble, metrics, model_selection, naive_bay
color = sns.color_palette()



eng_stopwords = set(stopwords.words("english"))
pd.options.mode.chained_assignment = None

## Read the train and test dataset and check the top few lines ##
train_df = pd.read_csv("../input/train.csv")
test_df = pd.read_csv("../input/test.csv")
print("Number of rows in train dataset : ",train_df.shape[0])
print("Number of rows in test dataset : ",test_df.shape[0])


## Number of punctuations in the text ##
train_df["num_punctuations"] =train_df["text"].apply(lambda x: len
test_df["num_punctuations"] =test_df["text"].apply(lambda x: len([
```

In [7]:

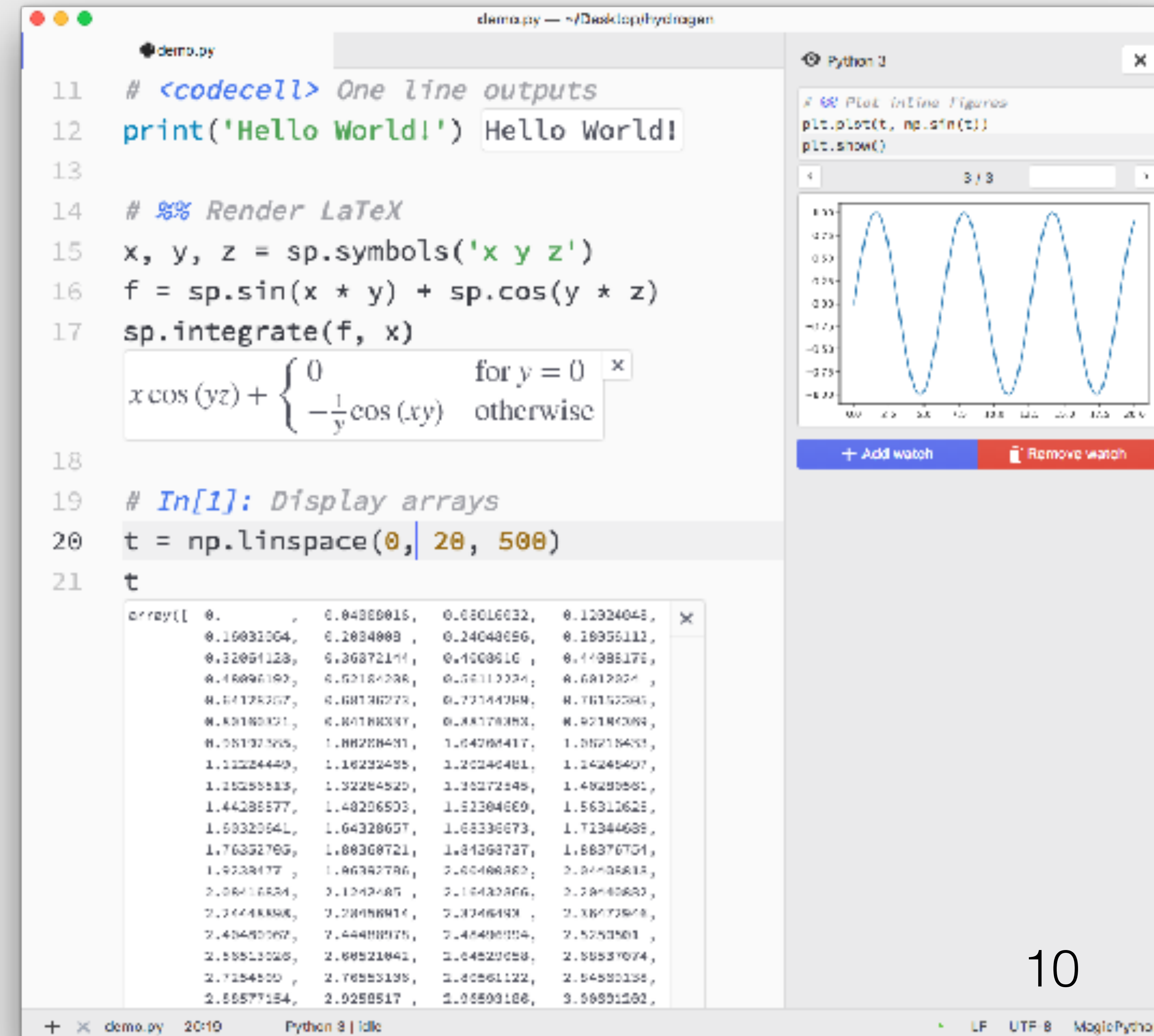
```python
train_df['num_words'].loc[train_df['num_words']>80] = 80 #trunc
plt.figure(figsize=(12,8))
sns.violinplot(x='author', y='num_words', data=train_df)
plt.xlabel('Author Name', fontsize=12)
plt.ylabel('Number of words in text', fontsize=12)
plt.title("Number of words by author", fontsize=15)
plt.show()
```

Out [7]:

EAP seems slightly lesser number of words than MWS and HPL.

In [8]:

```python
train_df['num_punctuations'].loc[train_df['num_punctuations']>1
plt.figure(figsize=(12,8))
sns.violinplot(x='author', y='num_punctuations', data=train_df)
plt.xlabel('Author Name', fontsize=12)
```



```python
# <codecell> One line outputs
print('Hello World!')
```

```
Hello World!
```

```python
# %% Render LaTeX
x, y, z = sp.symbols('x y z')
f = sp.sin(x * y) + sp.cos(y * z)
sp.integrate(f, x)
```

$$x \cos(yz) + \begin{cases} 0 & \text{for } y = 0 \\ -\frac{1}{y}\cos(xy) & \text{otherwise} \end{cases}$$

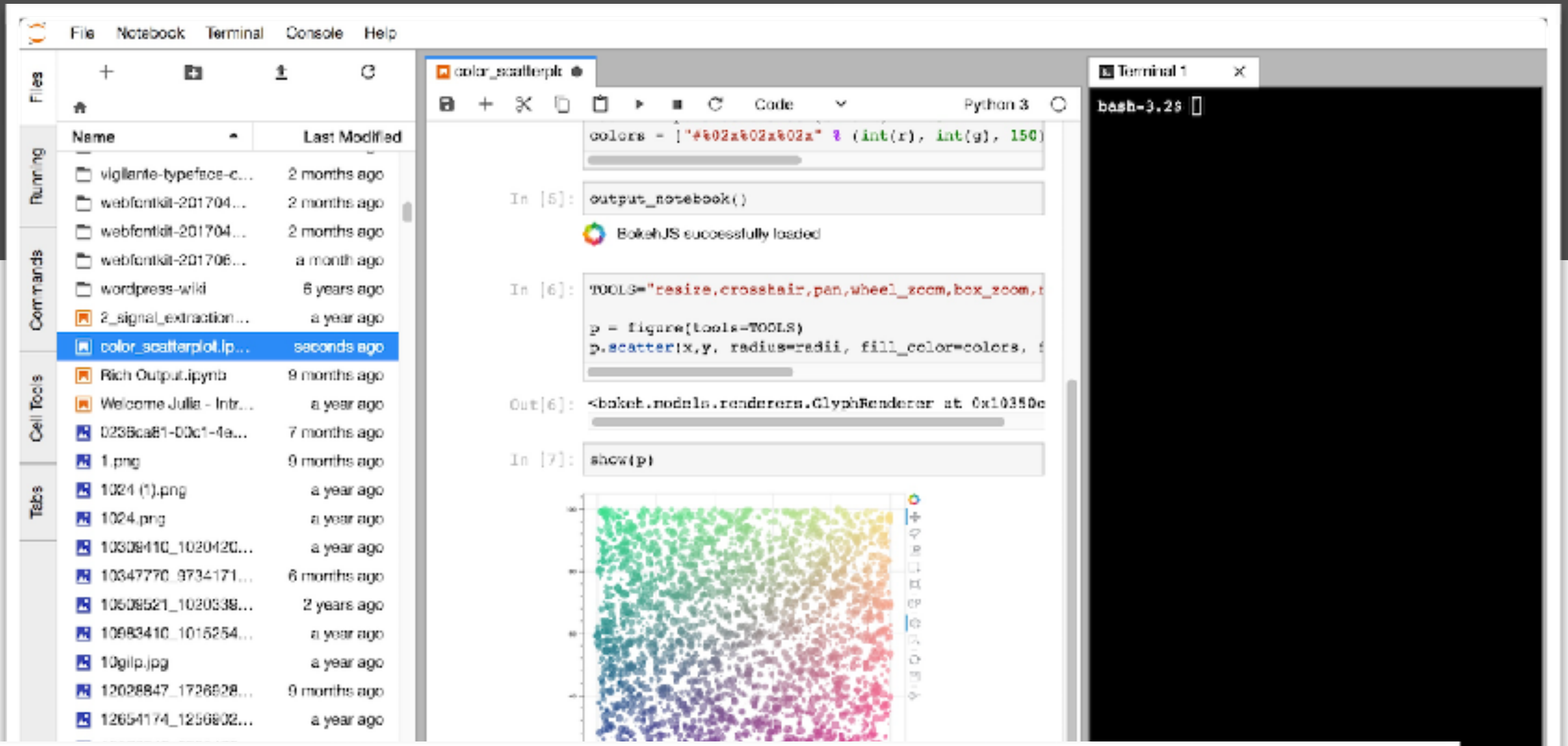
```python
# In[1]: Display arrays
t = np.linspace(0, 20, 500)
t
```

```
array([ 0.        ,  0.04008016,  0.08016032,  0.12024048,
        0.16032064,  0.2004008 ,  0.24048096,  0.28056112,
        0.32064128,  0.36072144,  0.4008016 ,  0.44088176,
        0.48096192,  0.52104208,  0.56112224,  0.6012024 ,
        0.64128257,  0.68136273,  0.72144289,  0.76152305,
        0.80160321,  0.84168337,  0.88176353,  0.92184369,
        0.96192385,  1.00200401,  1.04208417,  1.08216433,
        1.12224449,  1.16232465,  1.20240481,  1.24248497,
        1.28256513,  1.32264529,  1.36272545,  1.40280561,
        1.44288577,  1.48296593,  1.52304609,  1.56312625,
        1.60320641,  1.64328657,  1.68336673,  1.72344689,
        1.76352705,  1.80360721,  1.84368737,  1.88376754,
        1.9238477 ,  1.96392786,  2.00400802,  2.04408818,
        2.08416834,  2.1242485 ,  2.16432866,  2.20440882,
        2.24448898,  2.28456914,  2.3246493 ,  2.36472946,
        2.40480962,  2.44488978,  2.48496994,  2.5250501 ,
        2.56513026,  2.60521042,  2.64529058,  2.68537074,
        2.7254509 ,  2.76553106,  2.80561122,  2.84569138,
        2.88577154,  2.9258517 ,  2.96593186,  3.00601202,
```

```python
# %% Plot inline figures
plt.plot(t, np.sin(t))
plt.show()
```

JupyterLab

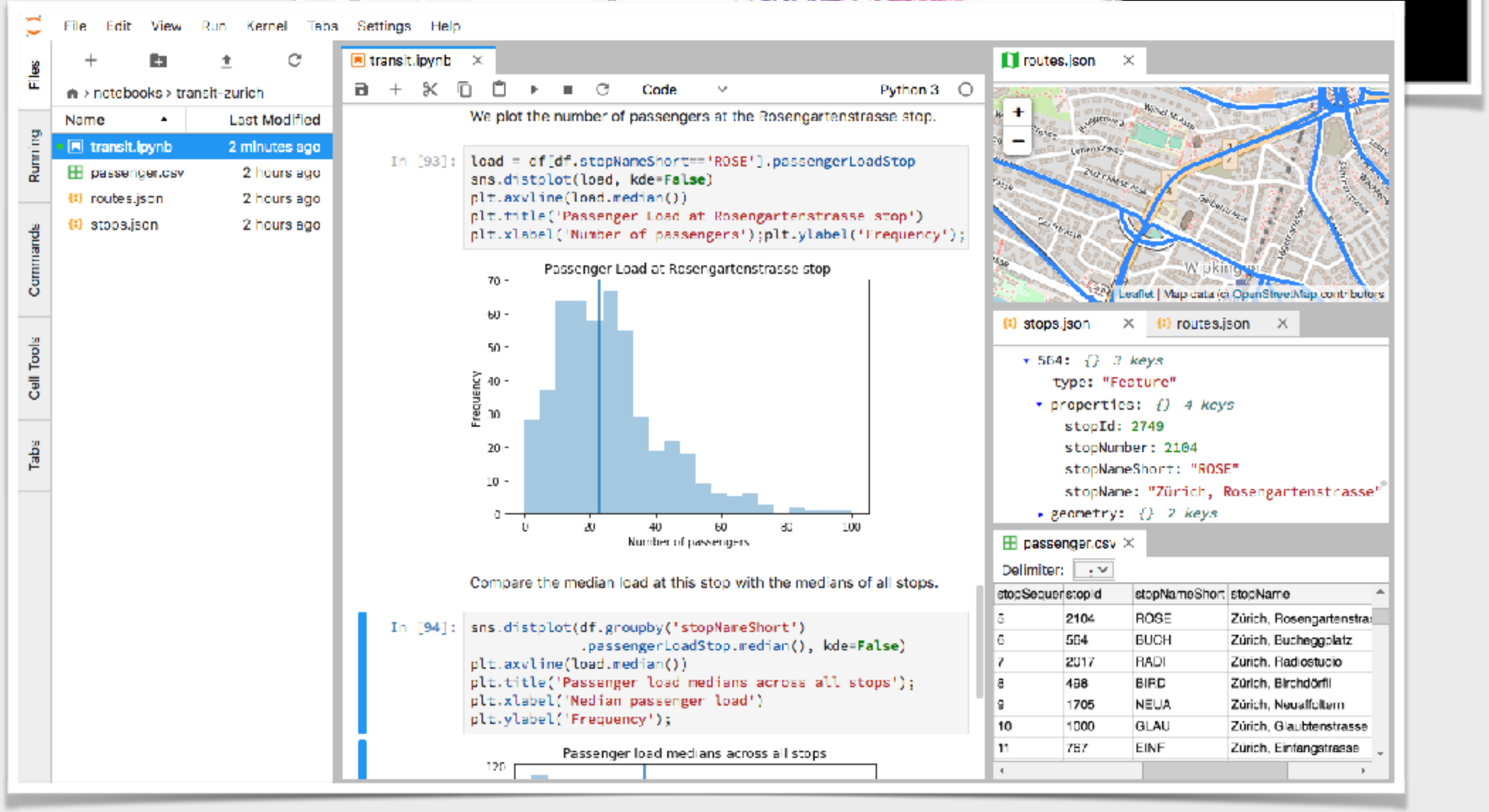
- Extends the notebook interface with text editor, shell, ...etc
- is it and IDE ?
- If by I you mean Interactive, then yes



Name	Last Modified
vigilante-typeface-c...	2 months ago
webfontkit-201704...	2 months ago
webfontkit-201704...	2 months ago
webfontkit-201706...	a month ago
wordpress-wiki	6 years ago
2_signal_extraction...	a year ago
color_scatterplot.ip...	seconds ago
Rich Output.ipynb	9 months ago
Welcome Julia - Intr...	a year ago
0238ca81-00c1-4e...	7 months ago
1.png	9 months ago
1024 (1).png	a year ago
1024.png	a year ago
10308410_1020420...	a year ago
10347770_9734171...	6 months ago
10508521_1020338...	2 years ago
10983410_1015254...	a year ago
10gilp.jpg	a year ago
12028847_1726828...	9 months ago
12654174_1256802...	a year ago

```
colors = ["#%02x%02x%02x" % (int(r), int(g), 150)

In [5]: output_notebook()

        BokehJS successfully loaded

In [6]: TOOLS="resize,crosshair,pan,wheel_zoom,box_zoom,r

        p = figure(tools=TOOLS)
        p.scatter(x,y, radius=radii, fill_color=colors, f

Out[6]: <bokeh.models.renderers.GlyphRenderer at 0x10350c

In [7]: show(p)
```

```
bash-3.2$ 
```



Name	Last Modified
transit.ipynb	2 minutes ago
passenger.csv	2 hours ago
routes.json	2 hours ago
stops.json	2 hours ago

We plot the number of passengers at the Rosengartenstrasse stop.

```
In [93]: load = df[df.stopNameShort=='ROSE'].passengerLoadStop
         sns.distplot(load, kde=False)
         plt.axvline(load.median())
         plt.title('Passenger Load at Rosengartenstrasse stop')
         plt.xlabel('Number of passengers');plt.ylabel('Frequency');
```

Compare the median load at this stop with the medians of all stops.

```
In [94]: sns.distplot(df.groupby('stopNameShort')
                        .passengerLoadStop.median(), kde=False)
         plt.axvline(load.median())
         plt.title('Passenger load medians across all stops');
         plt.xlabel('Median passenger load')
         plt.ylabel('Frequency');
```

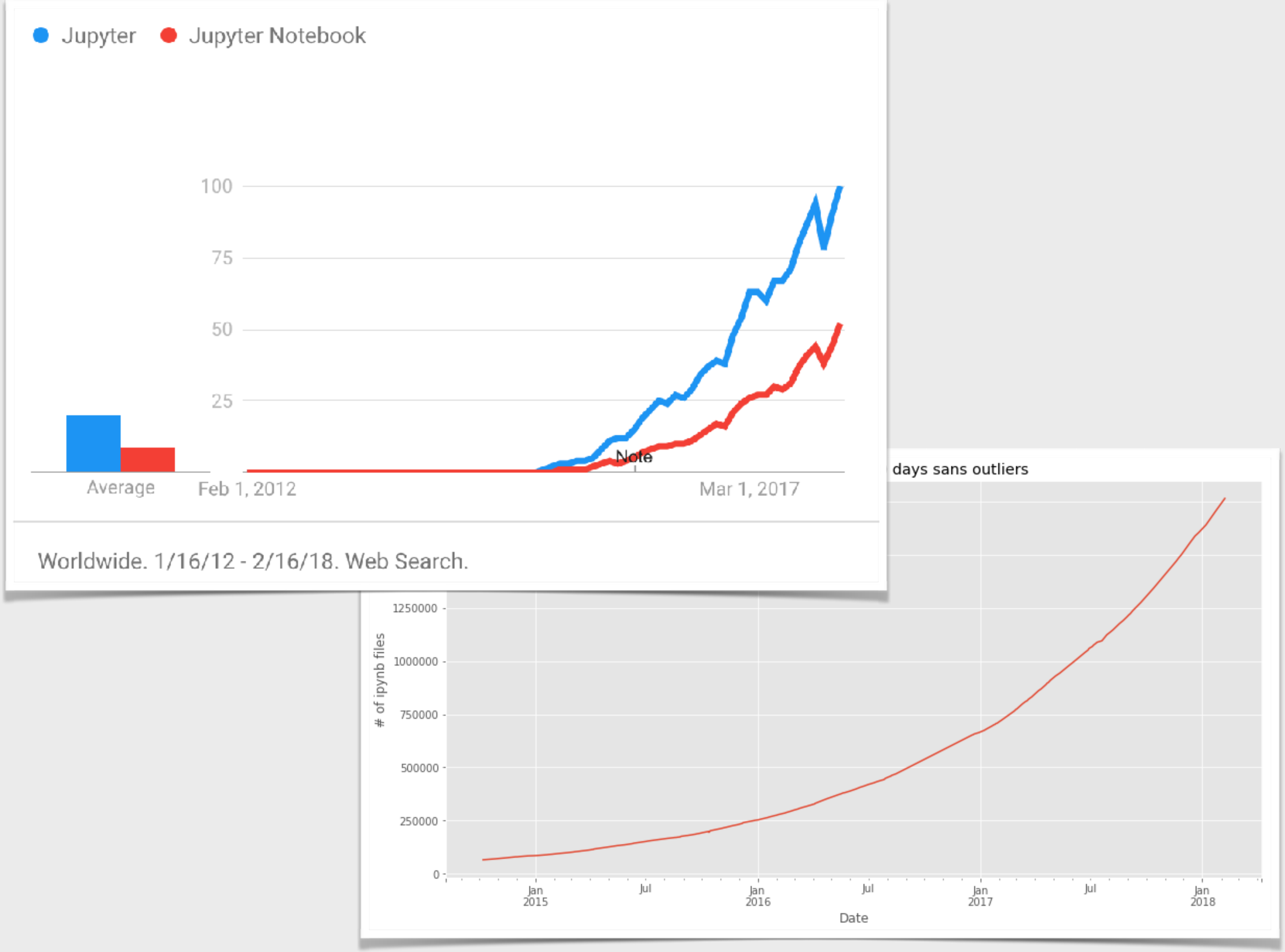
```
▾ 564: {} 3 keys
    type: "Feature"
  ▾ properties: {} 4 keys
      stopId: 2749
      stopNumber: 2104
      stopNameShort: "ROSE"
      stopName: "Zürich, Rosengartenstrasse"
  ▸ geometry: {} 2 keys
```

Delimiter: ,

stopSequen	stopId	stopNameShort	stopName
5	2104	ROSE	Zürich, Rosengartenstra
6	564	BUCH	Zürich, Bucheggplatz
7	2017	RADI	Zürich, Radiostudio
8	498	BIRD	Zürich, Birchdörfli
9	1705	NEUA	Zürich, Neuaffoltern
10	1000	GLAU	Zürich, Glaubtenstrasse
11	787	EINF	Zürich, Einfangstrasse

https://blog.jupyter.org/jupyterlab-is-ready-for-users-5a6f039b8906

Popularity



https://github.com/parente/nbestimate

Popularity

Interactivity

- Coding is not the end goal of most of our users. A simple, single tool, with friendly interface helps.
- Persisting kernel state allows to iterate only on part of an analysis.
- Notebook interface give the interactivity of the REPL with the edit-ability and linearity of a script with intermediate result. Aka "Literate Computing”

Popularity

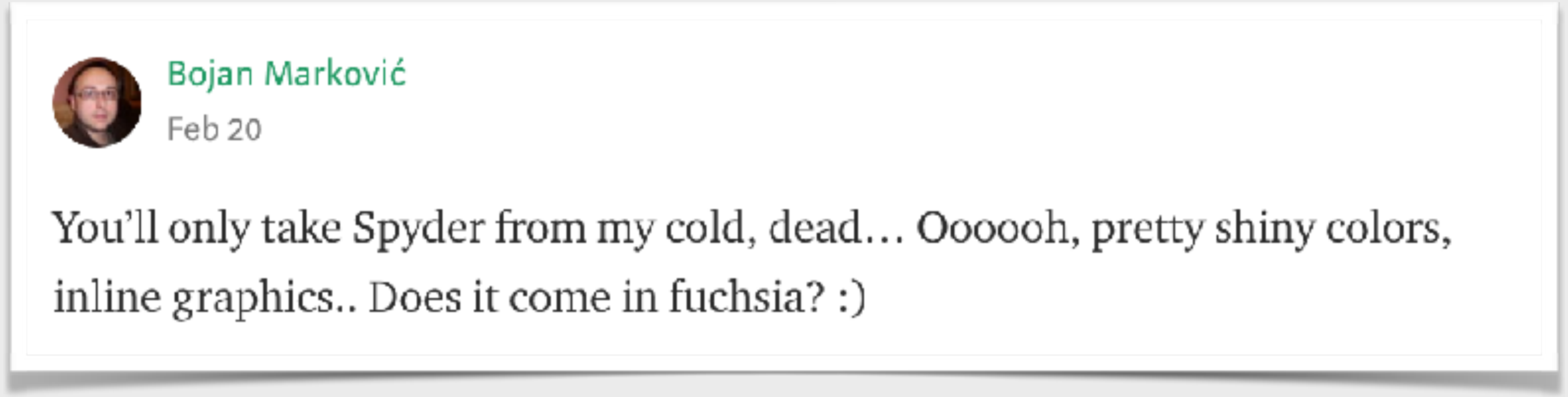
Separation of states

- Computation, narrative/visualisation in different processes.
    - Robust to crashes
    - Can "Share" and analysis / notebook without having to “rerun"
    - Trustworthy (No copy-past issues).
- Cons:
    - Understanding that document/kernel can have different states can be challenging.
    - Notebook format is not as widespread as others.

Popularity

Network enabled / web based

- User love fancy colors and things moving. Using D3 and other



Bojan Marković

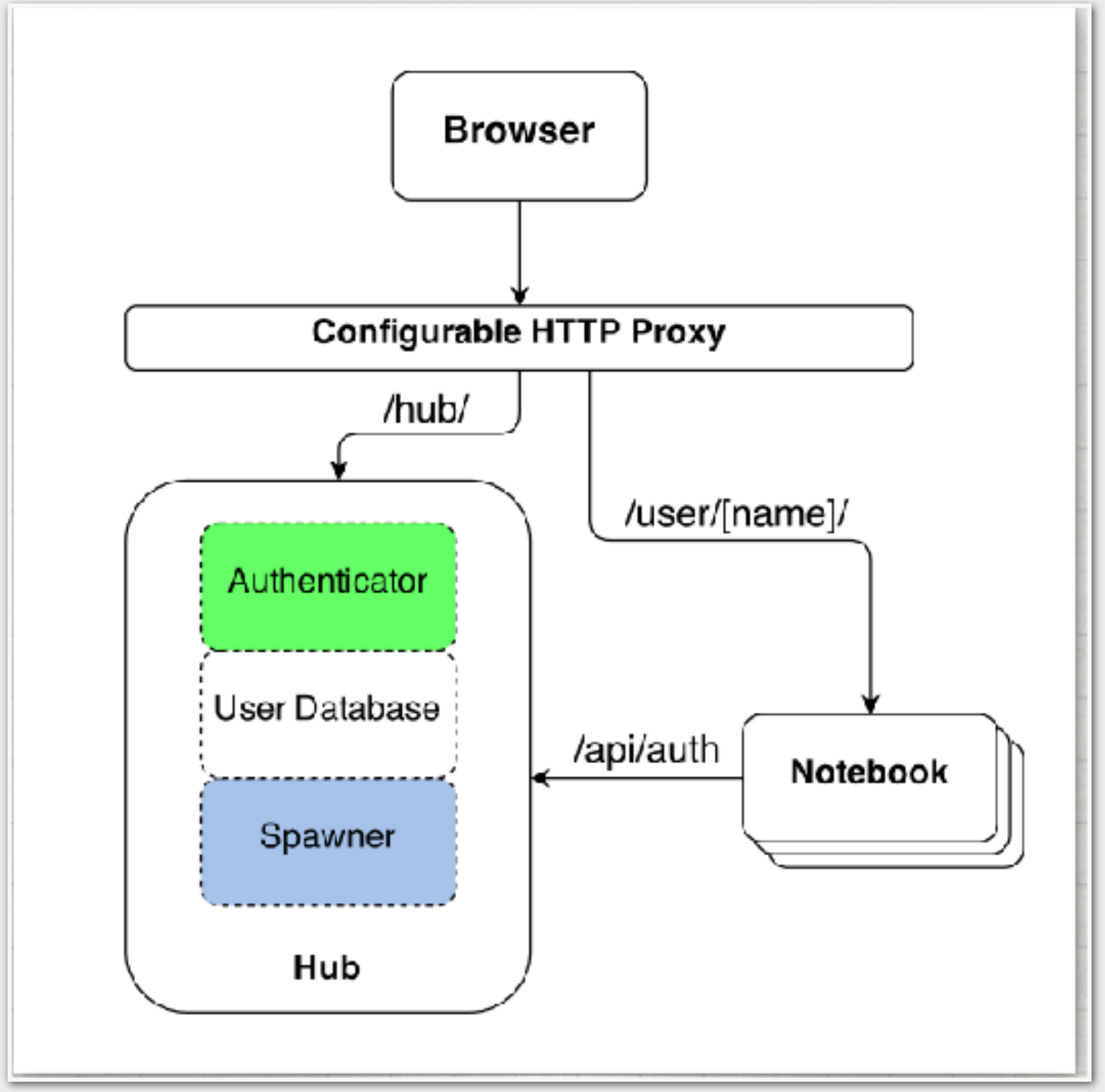
Feb 20

You'll only take Spyder from my cold, dead… Oooooh, pretty shiny colors, inline graphics.. Does it come in fuchsia? :)

  - dynamic libraries are highly popular
  - Usable by novices and power-users
  - Users w/ different expertise (Numerical Methods, Visualization,...)
- Seamless transition to HPC: Kernel Menu > Restart on Cluster
- Document persist if code crash.
- Can be Zero-Installation (See JupyterHub).
  - A web browser is all you need.

JupyterHub

- Multi-users Jupyter deployment
  - Not (Yet) Realtime collaboration
- Each user can get their own process/version(s)/configuration(s)
  - Hooks into any Auth
  - Only requires a browser
- Not limited to running Jupyter (e.g. work with RStudio, OpenRefine...)

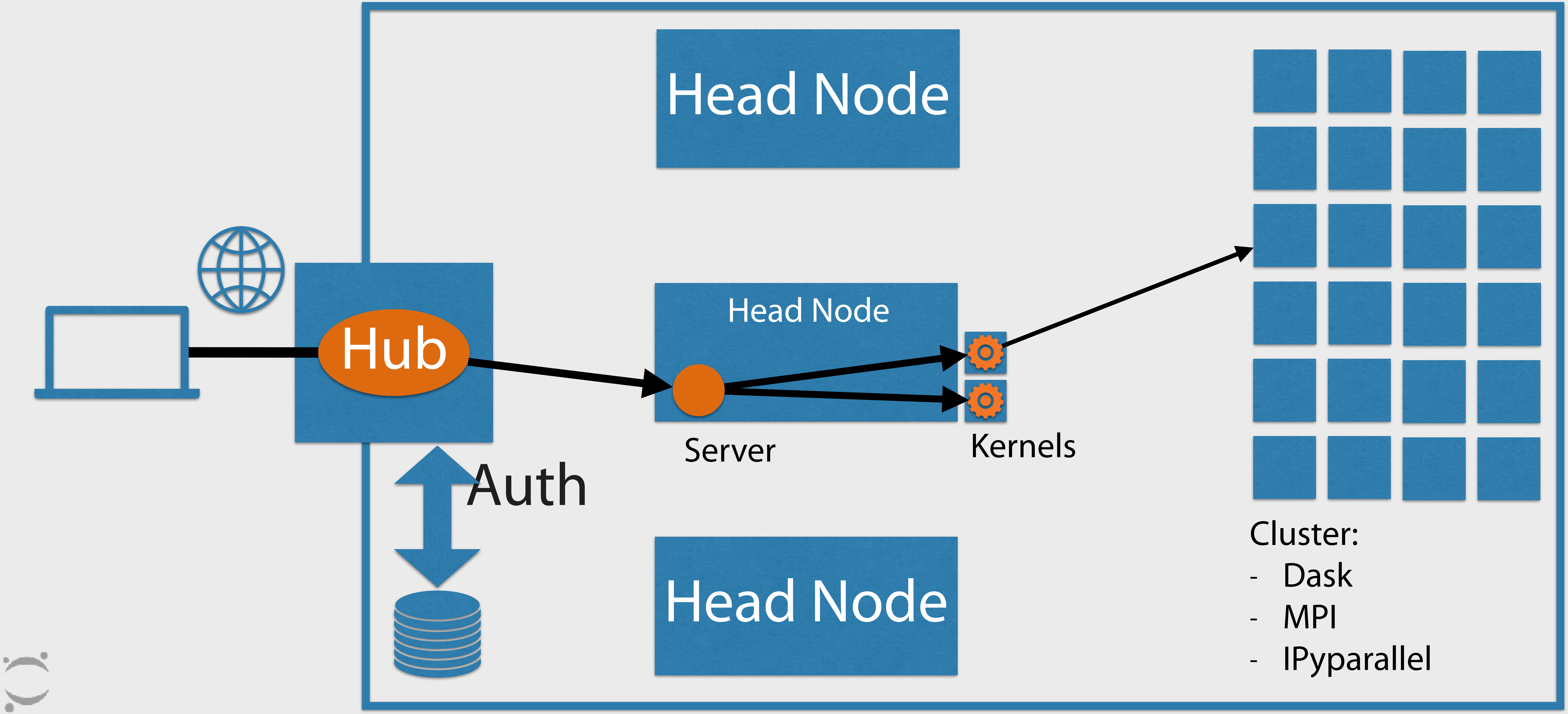


Use Cases

HPC

- Batch Jobs
  - You can run notebook “headless”
  - Parametrized notebook as “reports” you can interact with later
- Interactive Cluster.
  - Run a Hub (hook into LDAP/PAM...)
  - Run notebook servers on a Head node
  - Run Kernels on head Node/fast queue
  - Extra Workers (e.g. dask) on Batch queue/cluster.

HPC deployment



Cluster:

- Dask
- MPI
- IPyparallel

HPC Misconceptions

We need to run JupyterHub on the cluster: No

- Hub, Server, Kernels, (and Workers):
  - Do not have to be on the same machine
  - Do not have to use the same environment

A Kernel is a (single) language: No

- A Kernel is a preconfigured computation environment. It can be:
  - A queue, a hardware resource (GPU, SSD...), A location (like a beam-line)
- Example of Python, Cython, Julia, R, Fortran, Rust, C calling each other in same notebook

Every User have the same environment: No/No

- Kernels and notebook server can be configured independently
- Subset of users could use different server versions w/ different extensions.

JupyterHub is Limited to Jupyter: No

JupyterHub Can run RStudio, Open Refine.

Danger !

Despite Notebook being great, some limitations:

Most if not all **document state** is **in your browser** !

- Watch out for flaky network connections !
- Do Not close your Laptop Lid*/Tab* !

Workaround:

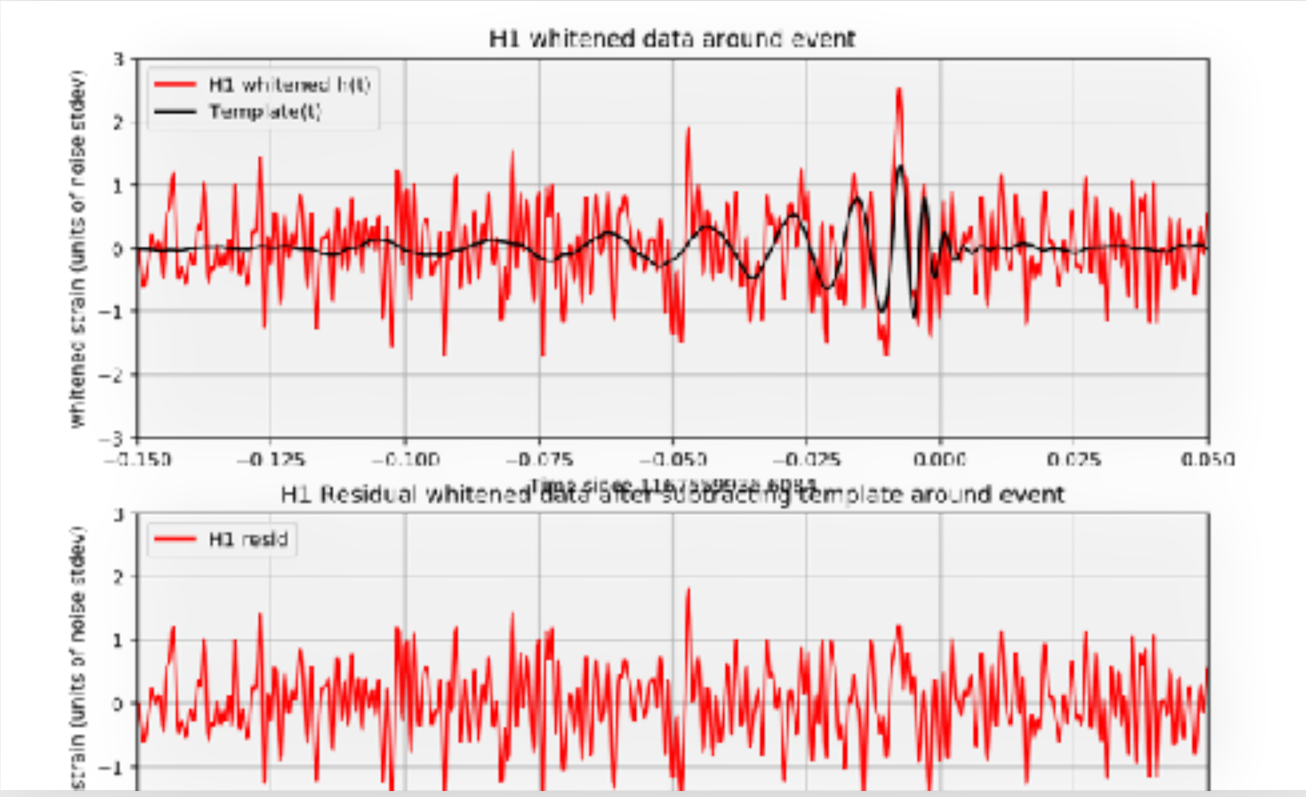
- Wrap computations (especially long), in Futures
- Use Caching.

Interrupting in compiled code is hard.

Large outputs/notebooks can crash the browser

*recent versions have buffering if client disconnected properly

Some Jupyter Usage



Ligo

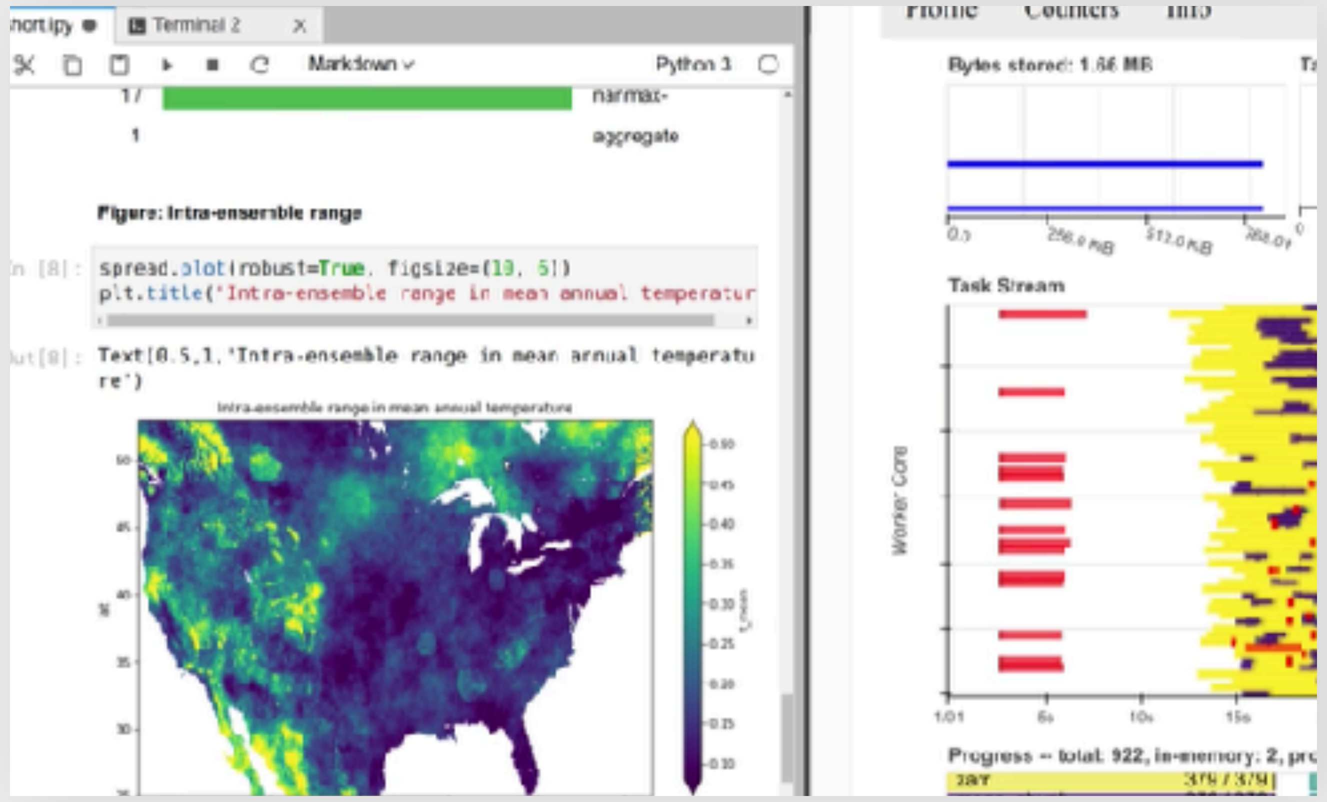


Figure: Intra-ensemble range

```
In [8]: spread.plot(robust=True, figsize=(13, 5))
        plt.title('Intra-ensemble range in mean annual temperatur
Out[8]: Text(0.5,1,'Intra-ensemble range in mean annual temperatu
        re')
```

Progress -- total: 922, in-memory: 2, pr

Pangeo

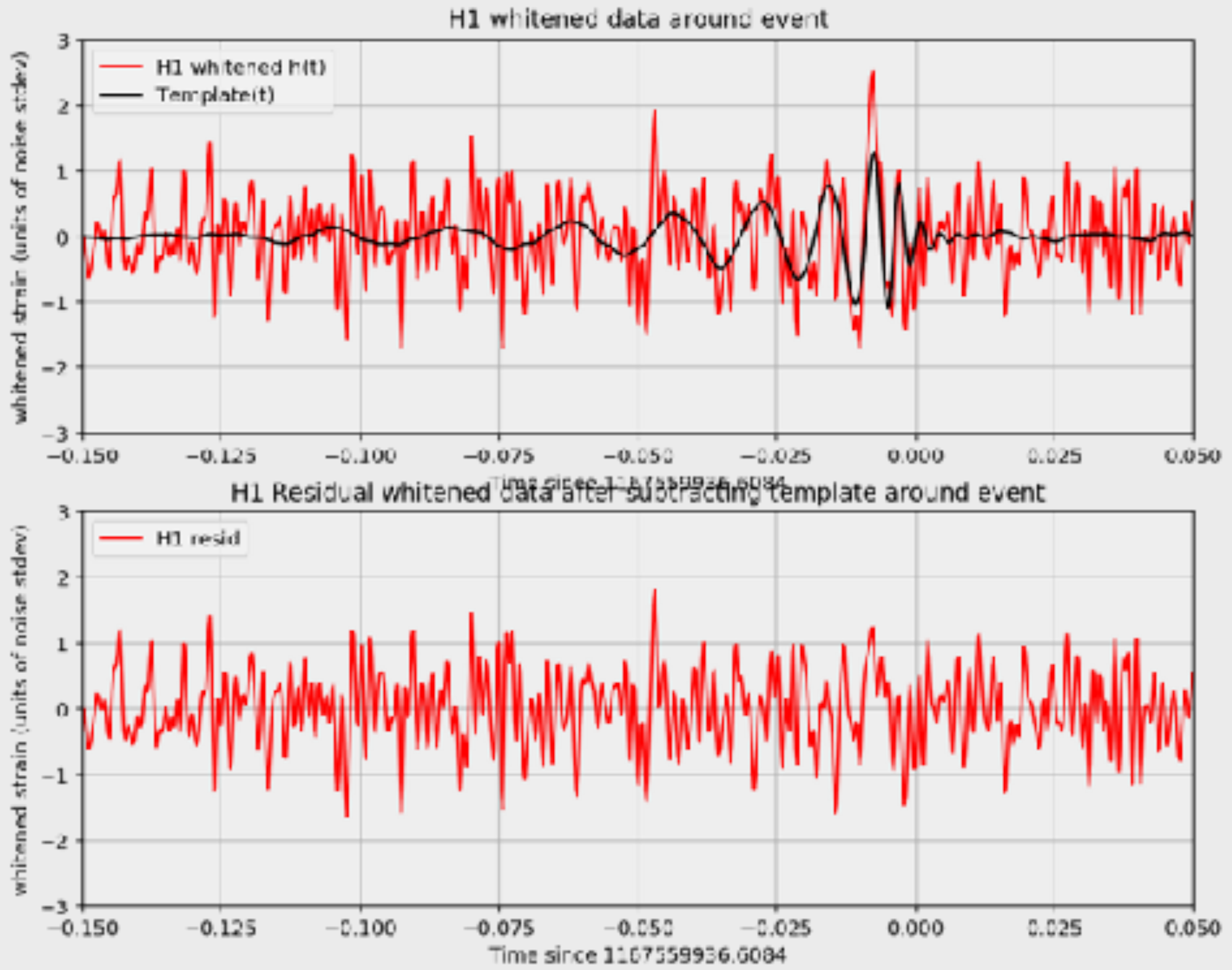
(Interactive usag

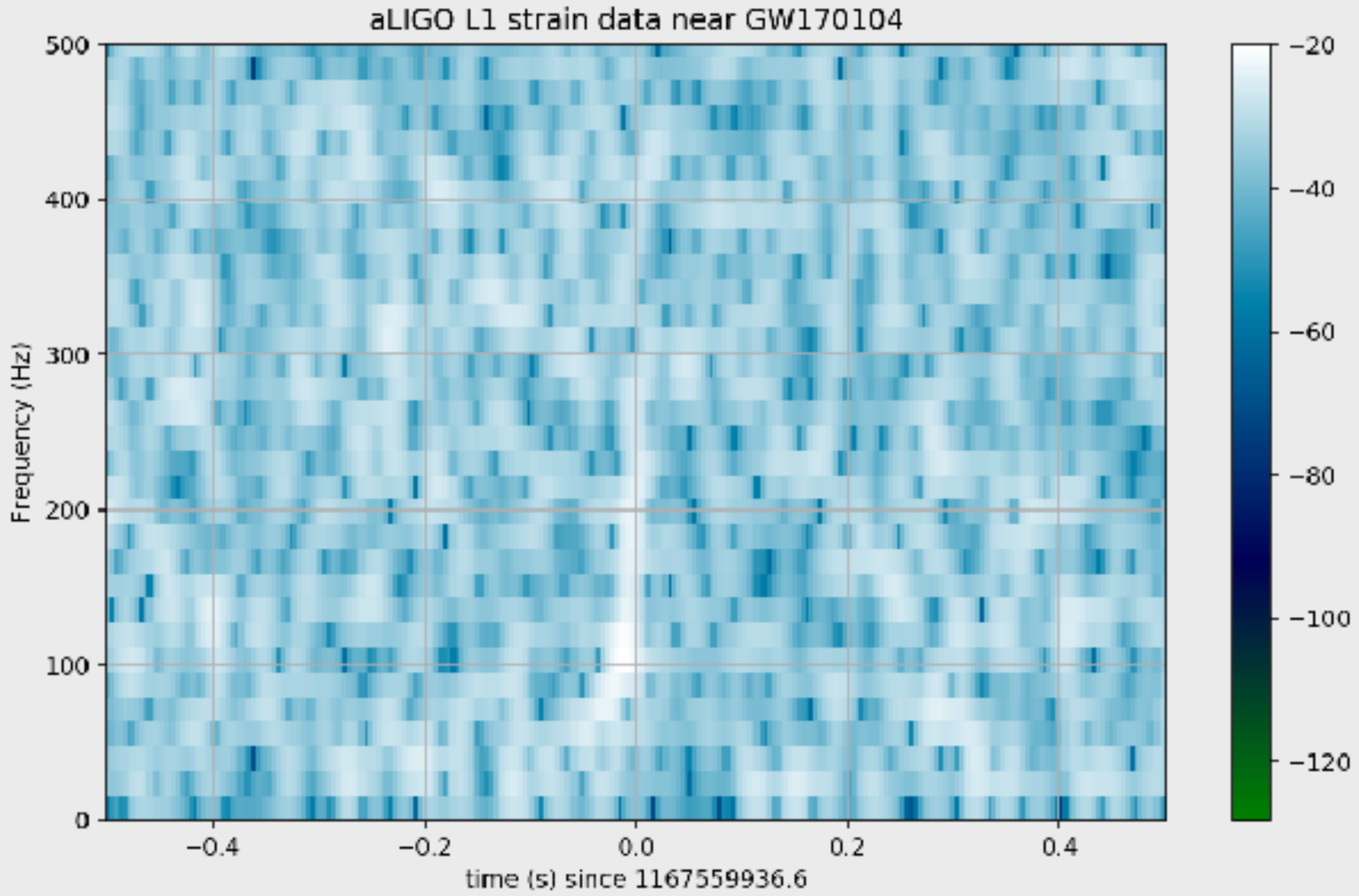
```
jetComponents->Draw("LEGO1 0");
c.Draw();

#------------------------------------------
#                         FastJet release
#                M. Cacciari, G.P. Salam
```

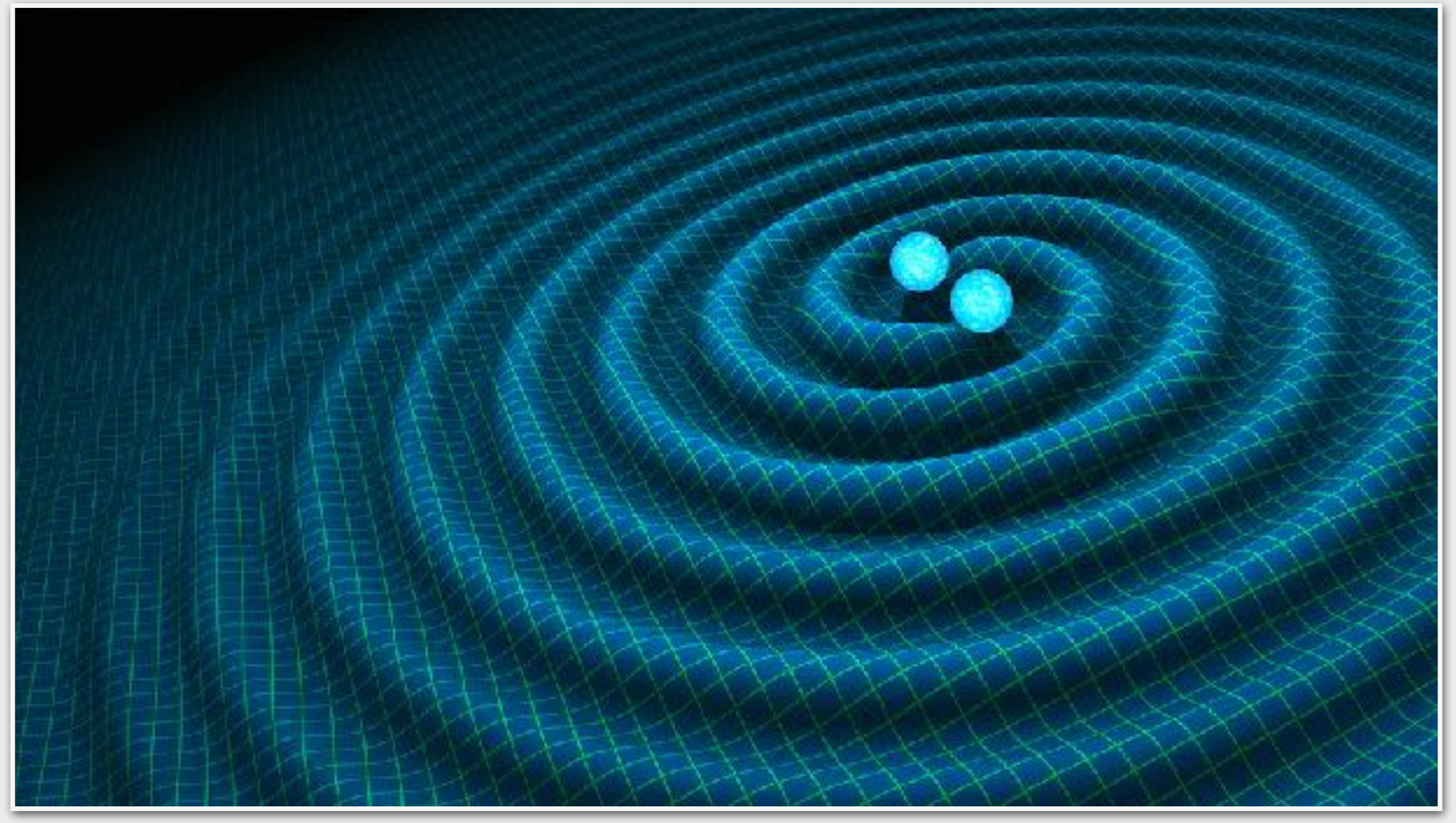
Cern’s SWAN

Ligo

- Some events analysis with Jupyter
- Subset of data + env put online
- Run the analysis yourself on Binder[1] and listen to the waves


H1 whitened data around event
H1 whitened h(t)
Template(t)
whitened strain (units of noise stdev)
3
2
1
0
−1
−2
−3
−0.150
−0.125
−0.100
−0.075
−0.050
−0.025
0.000
0.025
0.050
Time since 1167559936.6084
H1 Residual whitened data after subtracting template around event
H1 resid
whitened strain (units of noise stdev)
Time since 1167559936.6084





[1] https://github.com/minrk/ligo-binder

Pangeo (pangeo-data.github.io)

- Effort from Atmosphere / Ocean / Land / Climate (AOC) science community
- unified effort
- Cloud based
- Recent Technologies
  - Dask, Jupyter

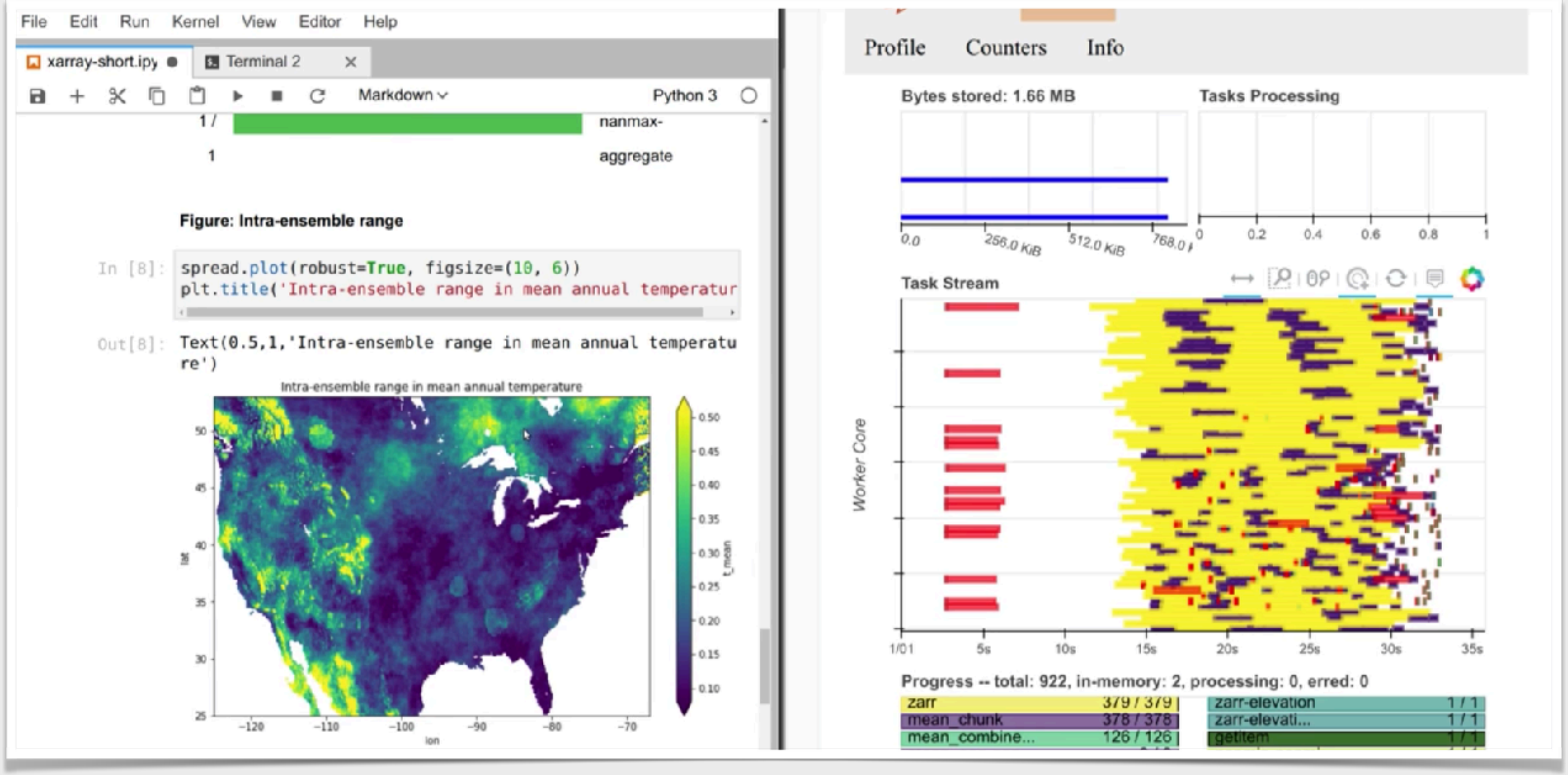


Figure: Intra-ensemble range

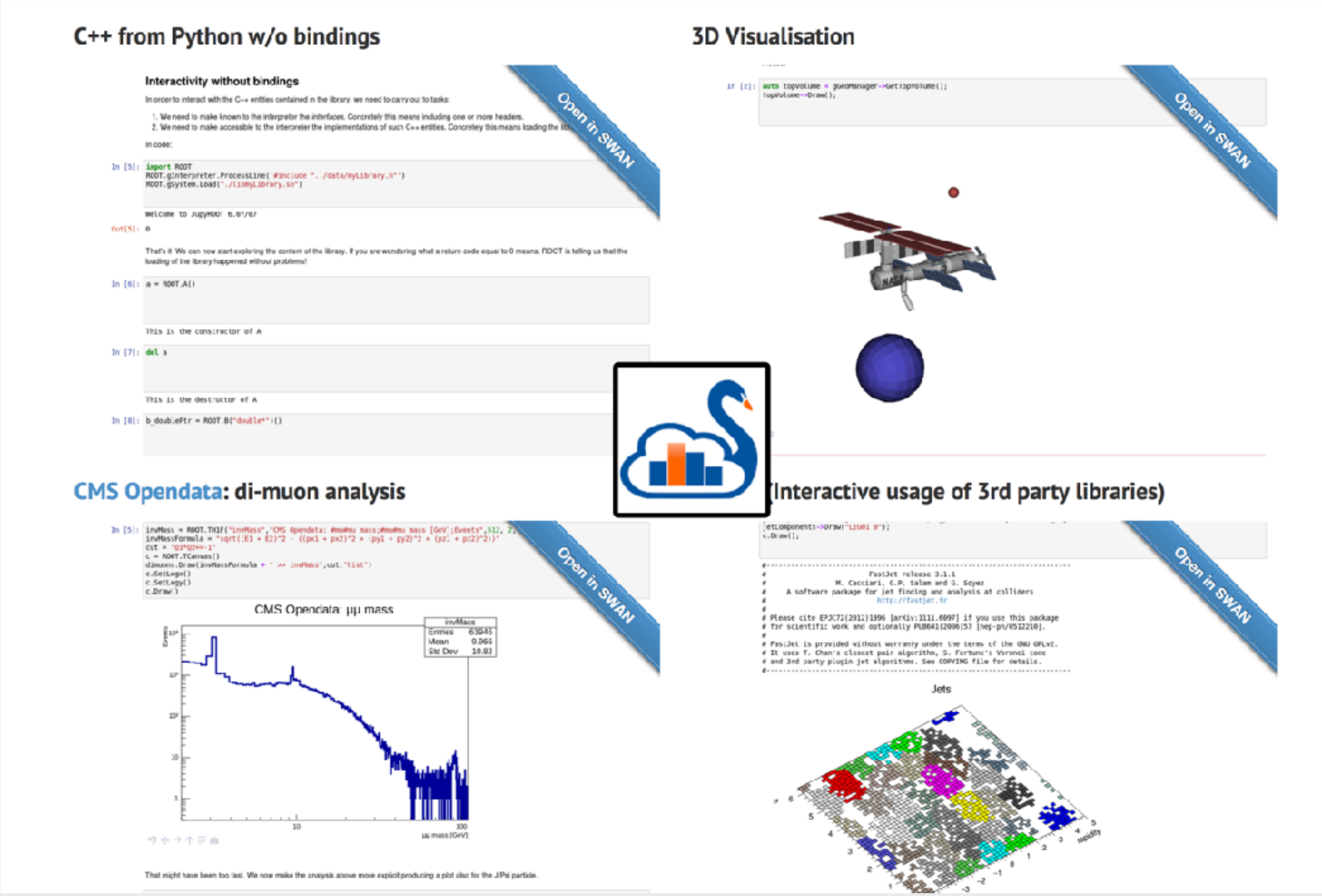
```
In [8]: spread.plot(robust=True, figsize=(10, 6))
        plt.title('Intra-ensemble range in mean annual temperatur

Out[8]: Text(0.5,1,'Intra-ensemble range in mean annual temperatu
        re')
```

```
Progress -- total: 922, in-memory: 2, processing: 0, erred: 0
zarr              379 / 379    zarr-elevation    1 / 1
mean_chunk        378 / 378    zarr-elevati...   1 / 1
mean_combine...   126 / 126    getitem           1 / 1
```

Matt Rocklin Blog post on pangeo-data.github.io

Cern Swan (swan.web.cern.ch)

- Share platformed for Data Analysis
- Sync W/ $HOME directory
- 0-install
- Share Data
- Provide example gallery with 1-click-fork



C++ from Python w/o bindings

Interactivity without bindings

In order to interact with the C++ entities contained in the library we need to carry out to tasks:

1. We need to make known to the interpreter the interfaces. Concretely this means including one or more headers.
2. We need to make accessible to the interpreter the implementations of such C++ entities. Concretey this means loading the lib

In code:

```
In [5]: import ROOT
        ROOT.gInterpreter.ProcessLine('#include "../data/myLibrary.h"')
        ROOT.gSystem.Load("./libmyLibrary.so")

        Welcome to JupyROOT 6.07/07
Out[5]: 0
```

That's it! We can now start exploring the content of the library. If you are wondering what a return code equal to 0 means, ROOT is telling us that the loading of the library happened without problems!

```
In [6]: a = ROOT.A()

        This is the constructor of A
In [7]: del a

        This is the destructor of A
In [8]: b_doublePtr = ROOT.B("double*")()
```

3D Visualisation

```
In [2]: auto topVolume = geoManager->GetTopVolume();
        topVolume->Draw();
```

CMS Opendata: di-muon analysis

```
In [5]: invMass = ROOT.TH1F("invMass","CMS Opendata: #mu#mu mass;#mu#mu mass [GeV];Events",512, 2,
        invMassFormula = "sqrt((E1 + E2)^2 - ((px1 + px2)^2 + (py1 + py2)^2 + (pz1 + pz2)^2))"
        cut = "Q1*Q2==-1"
        c = ROOT.TCanvas()
        dimuons.Draw(invMassFormula + " >> invMass",cut,"hist")
        c.SetLogx()
        c.SetLogy()
        c.Draw()
```

CMS Opendata: μμ mass

invMass	
Entries	63946
Mean	0.066
Std Dev	10.82

That might have been too fast. We now make the analysis above more explicit producing a plot also for the J/Psi particle.

(Interactive usage of 3rd party libraries)

```
jetComponents->Draw("LEGO1 0");
c.Draw();

#--------------------------------------------------------------------------
#                         FastJet release 3.1.1
#                 M. Cacciari, G.P. Salam and G. Soyez
#     A software package for jet finding and analysis at colliders
#                           http://fastjet.fr
#
# Please cite EPJC72(2012)1896 [arXiv:1111.6097] if you use this package
# for scientific work and optionally PLB641(2006)57 [hep-ph/0512210].
#
# FastJet is provided without warranty under the terms of the GNU GPLv2.
# It uses T. Chan's closest pair algorithm, S. Fortune's Voronoi code
# and 3rd party plugin jet algorithms. See COPYING file for details.
#--------------------------------------------------------------------------
```

Jets

The Shape of Things to come

The Shape of Things to come

Classic Notebook -> JupyterLab transition

- Stabilisation
- Transfer of extensions
- Collaboration:
  - Google retired Real-time API
  - Who “executes” problem
- Long Running Jobs

JupyterHub

- Horizontal (and Vertical) Scaling
- Audits APIs (Hippa Compliance ?)
- “Federation” (binder) / Intercommunication

O'REILLY®

jupytercon

Brought to you by NumFOCUS Foundation and O'Reilly Media Inc.

August 21–24, 2018

New York, NY

jupytercon.com

CFP- Ends March 6th

Question(s)
while we change
speakers ?