**How I work**

and

# Contents

- cool stuff
- nbflow
- jupyter lab
- spyder 3.0
- bokeh
  - holoviews
  - datashader
- parallel python

# cool stuff

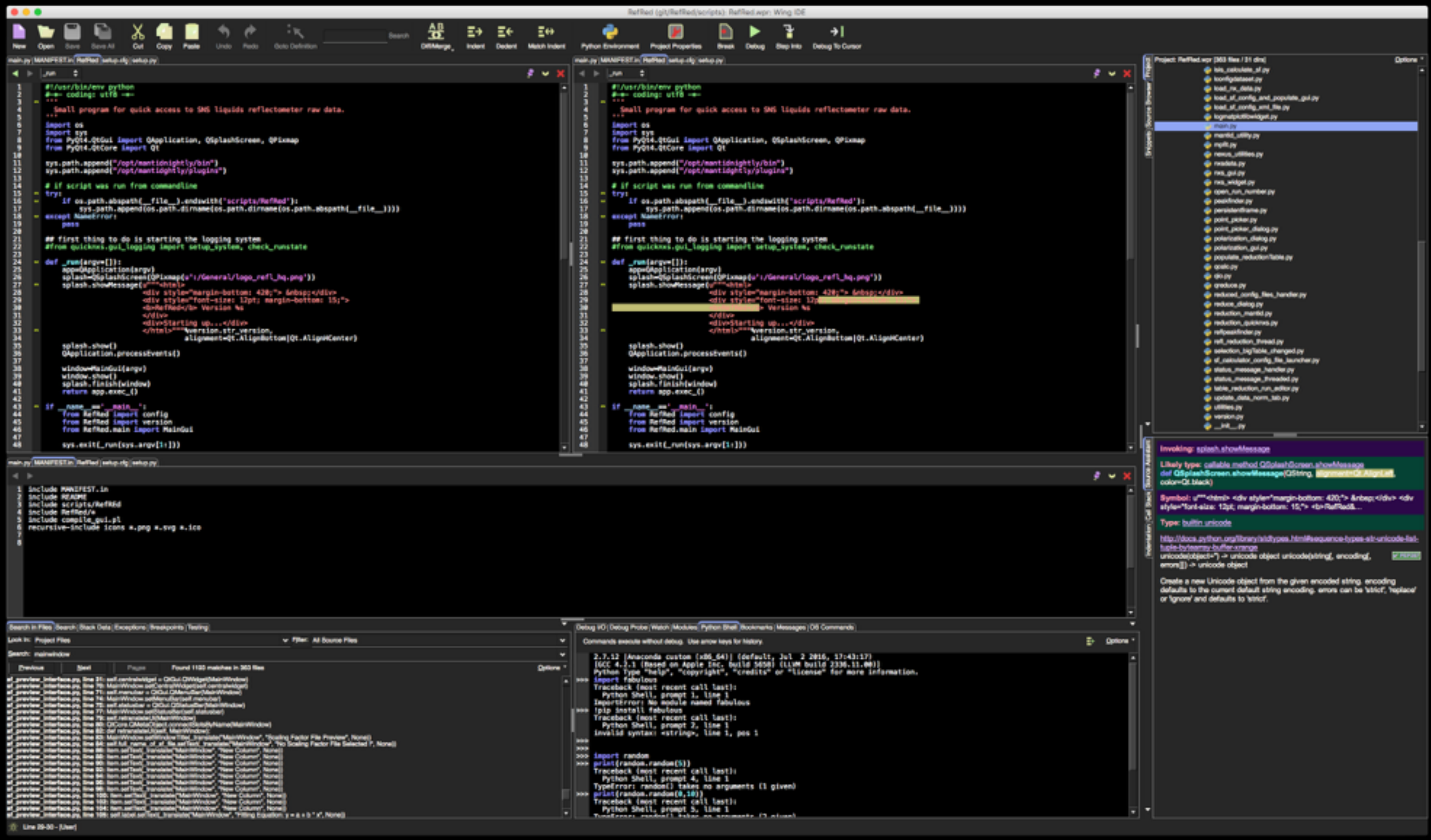
> conda environment

> > conda create env -n *name_of_env* python=3.5 matplotlib
> > source activate *name_of_env*
> > conda env list

> python IDE

> > wingIDE (https://wingware.com/)
> > pyCharm (https://www.jetbrains.com/pycharm/)
> > eclipse
> > spyder
> > jupyter lab (soon)

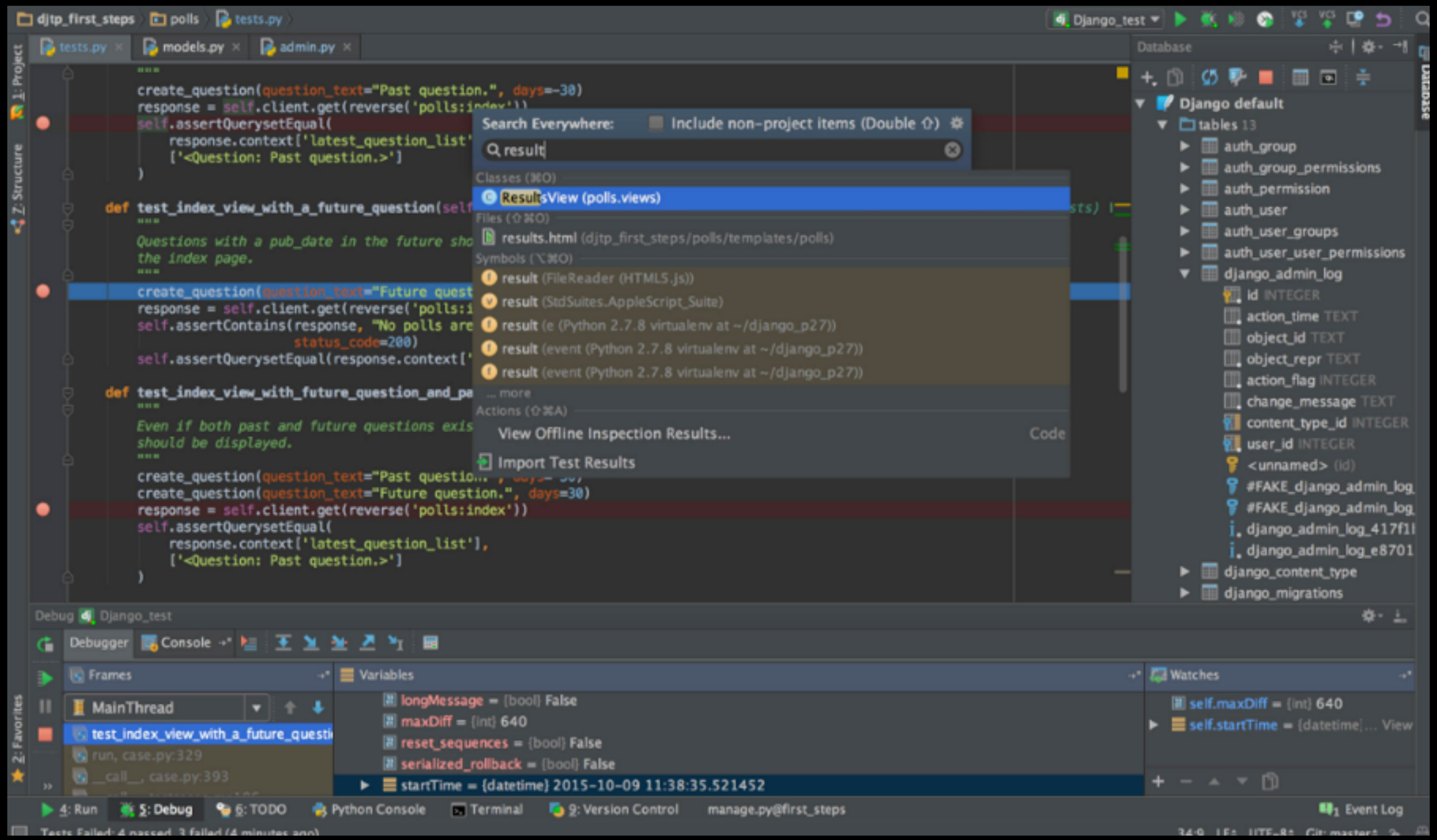
# wingIDE


#!/usr/bin/env python
# -*- coding: utf8 -*-
"""
Small program for quick access to SNS liquids reflectometer raw data.
"""
import os
import sys
from PyQt4.QtGui import QApplication, QSplashScreen, QPixmap
from PyQt4.QtCore import Qt
sys.path.append("/opt/mantidnightly/bin")
sys.path.append("/opt/mantidnightly/plugins")
# if script was run from commandline
try:
if os.path.abspath(__file__).endswith('scripts/RefRed'):
sys.path.append(os.path.dirname(os.path.dirname(os.path.abspath(__file__))))
except NameError:
pass
## first thing to do is starting the logging system
#from quicknxs.gui_logging import setup_system, check_runstate
def _run(argv=[]):
app=QApplication(argv)
splash=QSplashScreen(QPixmap(u':/General/logo_refl_hq.png'))
splash.showMessage(u"""<html>
  

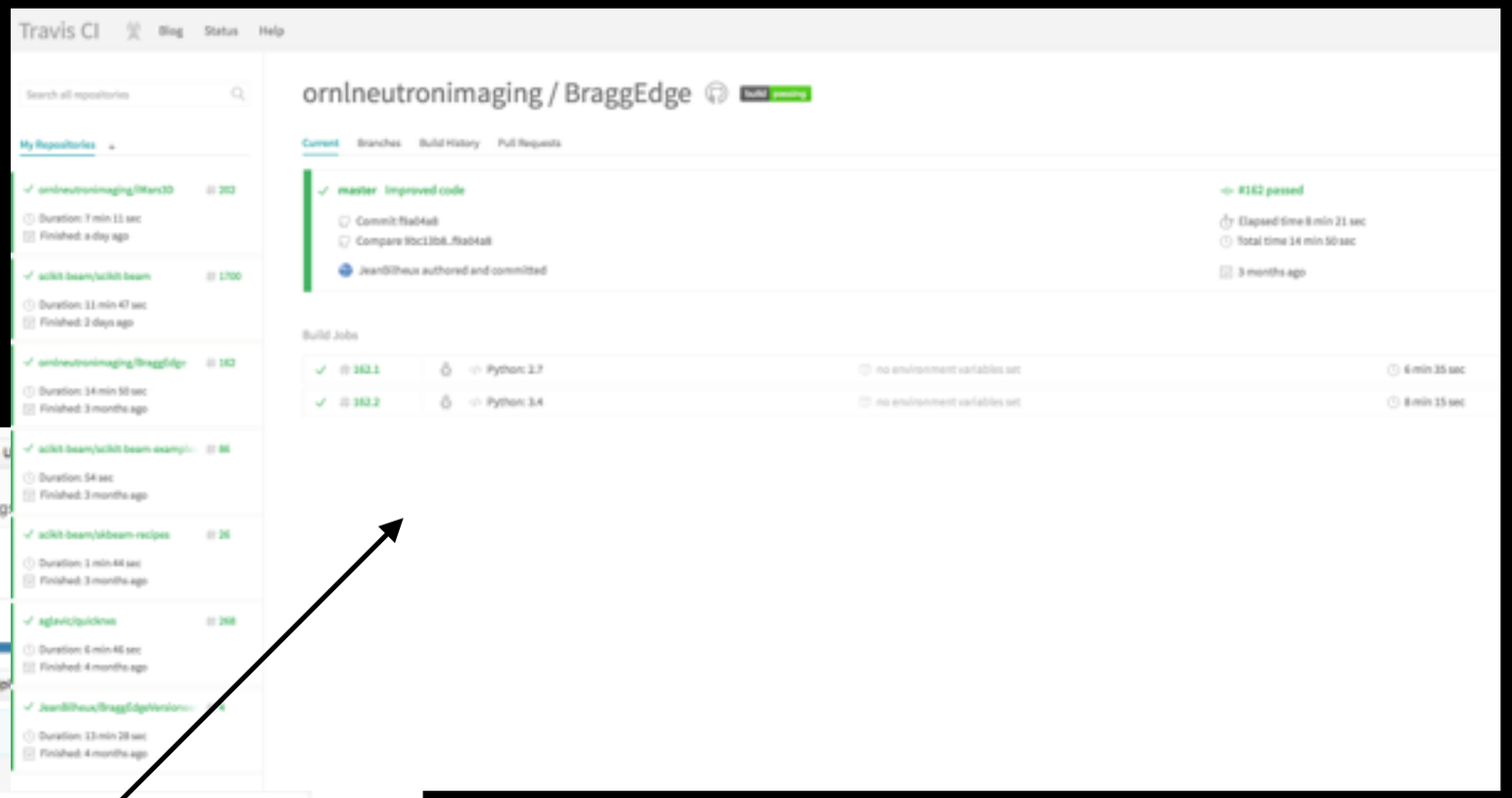
RefRed Version %s

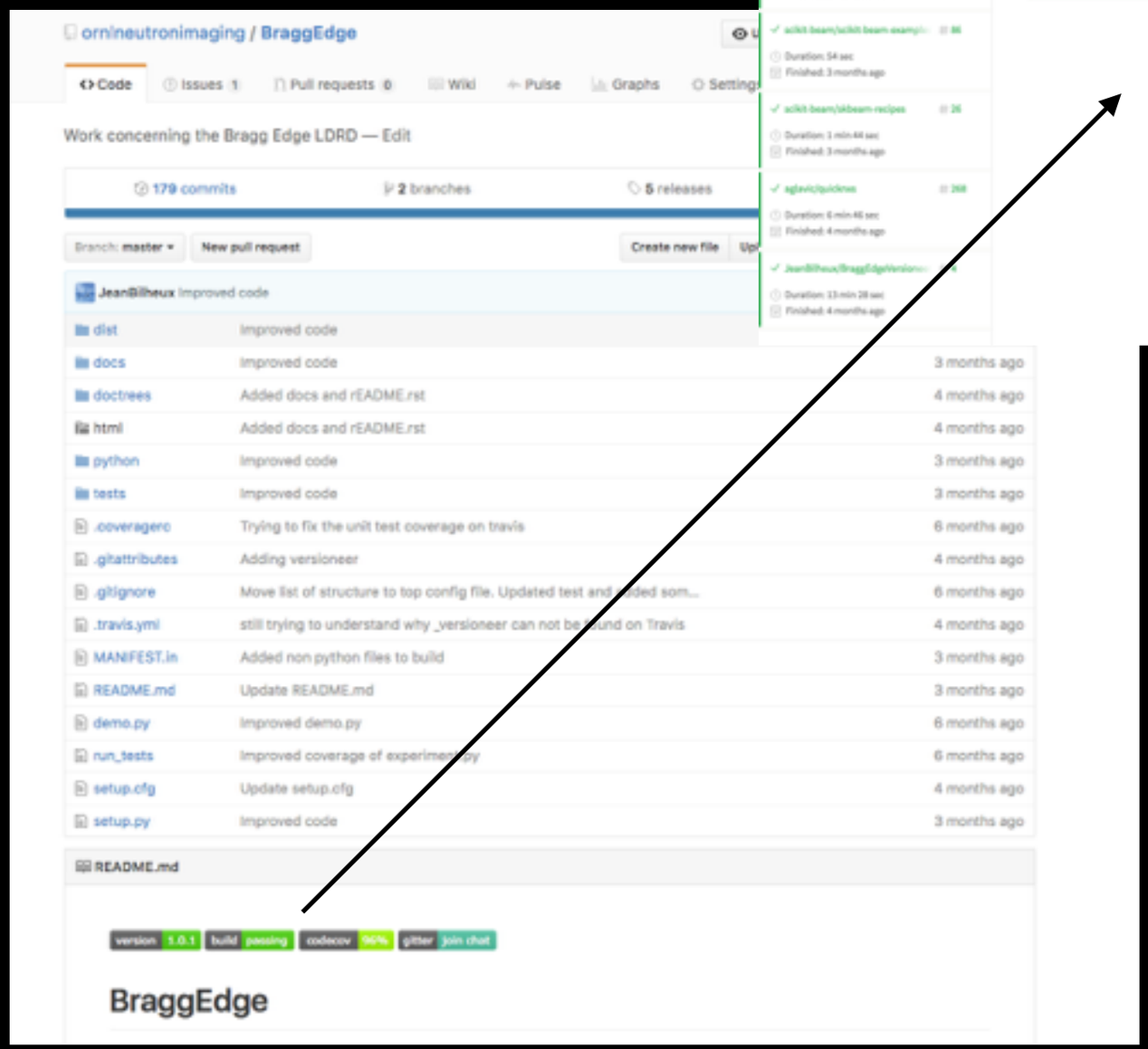
Starting up...
</html>"""%version.str_version,
alignment=Qt.AlignBottom|Qt.AlignHCenter)
splash.show()
QApplication.processEvents()
windowMainGui(argv)
window.show()
splash.finish(window)
return app.exec_()
if __name__=='__main__':
from RefRed import config
from RefRed import version
from RefRed.main import MainGui
sys.exit(_run(sys.argv[1:]))
include MANIFEST.in
include README
include scripts/RefRed
include RefRed/*
include compile_gui.pl
recursive-include icons *.png *.svg *.ico
Python Type "help", "copyright", "credits" or "license" for more information.
>>> import fabulous
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
ImportError: No module named fabulous
>>> !pip install fabulous
Traceback (most recent call last):
File "<stdin>", line 1
!pip install fabulous
^
SyntaxError: invalid syntax
>>>
>>> import random
>>> print(random.random())
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: random() takes no arguments (1 given)
>>> print(random.random(0,10))
Traceback (most recent call last):
File "<stdin>", line 1, in <module>
TypeError: random() takes no arguments (2 given)
Invoking: splash.showMessage
Likely type: callable method QSplashScreen.showMessage
def QSplashScreen.showMessage(QString, alignment=Qt.AlignLeft, color=Qt.black)
Symbol: u"""<html>     RefRed&...
Type: builtin unicode
http://docs.python.org/library/stdtypes.html#sequence-types-str-unicode-list-tuple-bytearray-buffer-xrange
unicode(object='') -> unicode object unicode(string[, encoding[, errors]]) -> unicode object
Create a new Unicode object from the given encoded string. encoding defaults to the current default string encoding. errors can be 'strict', 'replace' or 'ignore' and defaults to 'strict'.

# pyCharm


djtp_first_steps
polls
tests.py
Django_test
tests.py
models.py
admin.py
create_question(question_text="Past question.", days=-30)
response = self.client.get(reverse('polls:index'))
self.assertQuerysetEqual(
response.context['latest_question_list'],
['<Question: Past question.>']
)
def test_index_view_with_a_future_question(self):
"""
Questions with a pub_date in the future should not be displayed on
the index page.
"""
create_question(question_text="Future question.", days=30)
response = self.client.get(reverse('polls:index'))
self.assertContains(response, "No polls are available.",
status_code=200)
self.assertQuerysetEqual(response.context['latest_question_list'], [])
def test_index_view_with_future_question_and_past_question(self):
"""
Even if both past and future questions exist, only past questions
should be displayed.
"""
create_question(question_text="Past question.", days=-30)
create_question(question_text="Future question.", days=30)
response = self.client.get(reverse('polls:index'))
self.assertQuerysetEqual(
response.context['latest_question_list'],
['<Question: Past question.>']
)
Search Everywhere:
Include non-project items (Double ⇧)
result
Classes (⌘O)
ResultsView (polls.views)
Files (⇧⌘O)
results.html (djtp_first_steps/polls/templates/polls)
Symbols (⌥⌘O)
result (FileReader (HTML5.js))
result (StdSuites.AppleScript_Suite)
result (e (Python 2.7.8 virtualenv at ~/django_p27))
result (event (Python 2.7.8 virtualenv at ~/django_p27))
result (event (Python 2.7.8 virtualenv at ~/django_p27))
... more
Actions (⇧⌘A)
View Offline Inspection Results... Code
Import Test Results
Database
Django default
tables 13
auth_group
auth_group_permissions
auth_permission
auth_user
auth_user_groups
auth_user_user_permissions
django_admin_log
id INTEGER
action_time TEXT
object_id TEXT
object_repr TEXT
action_flag INTEGER
change_message TEXT
content_type_id INTEGER
user_id INTEGER
<unnamed> (id)
#FAKE_django_admin_log
#FAKE_django_admin_log
django_admin_log_417f1
django_admin_log_e8701
django_content_type
django_migrations
Debug Django_test
Debugger
Console
Frames
MainThread
test_index_view_with_a_future_questi
run, case.py:329
__call__, case.py:393
Variables
longMessage = {bool} False
maxDiff = {int} 640
reset_sequences = {bool} False
serialized_rollback = {bool} False
startTime = {datetime} 2015-10-09 11:38:35.521452
Watches
self.maxDiff = {int} 640
self.startTime = {datetime}... View
1: Project
7: Structure
2: Favorites
4: Run
5: Debug
6: TODO
Python Console
Terminal
9: Version Control
manage.py@first_steps
Event Log
Tests Failed: 4 passed, 3 failed (4 minutes ago)

# unit testing using travis



# documentation using sphinx

SPHINX
Python Documentation Generator

ornlneutronimaging.github.io/BraggEdge/index.html

Apps | Google Maps | Gmail | Google Contacts | Google Calendar | Adobe | Aviation | Python | Temporary | Programme soiree - ( | TweetDeck | SciPy2016

## braggedge 1.0.0 documentation

WELCOME TO BRAGGEDGE'S DOCUMENTATION!

Contents :: How to use This library »

### Welcome to braggedge's documentation!

Contents:

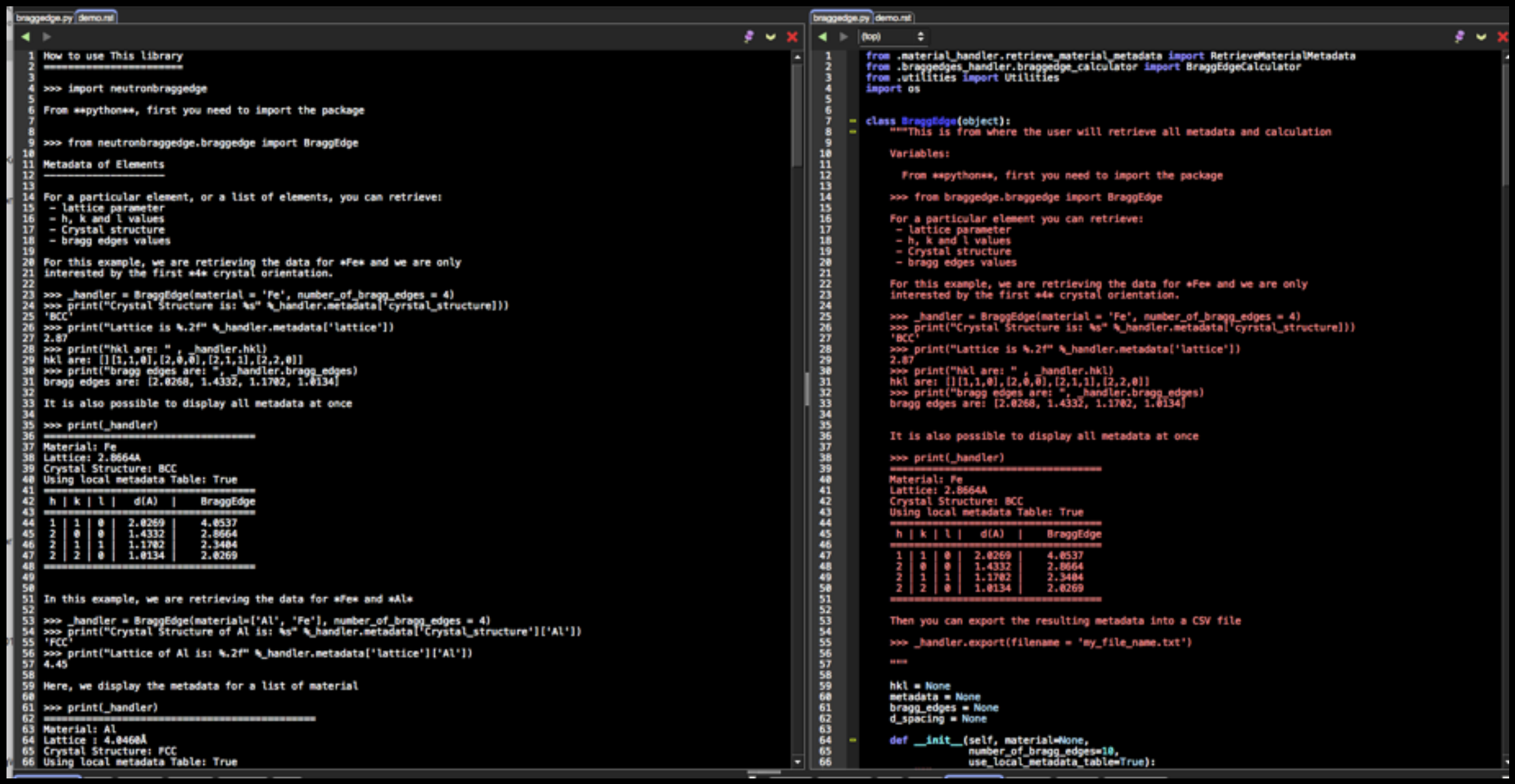


### Indices and tables



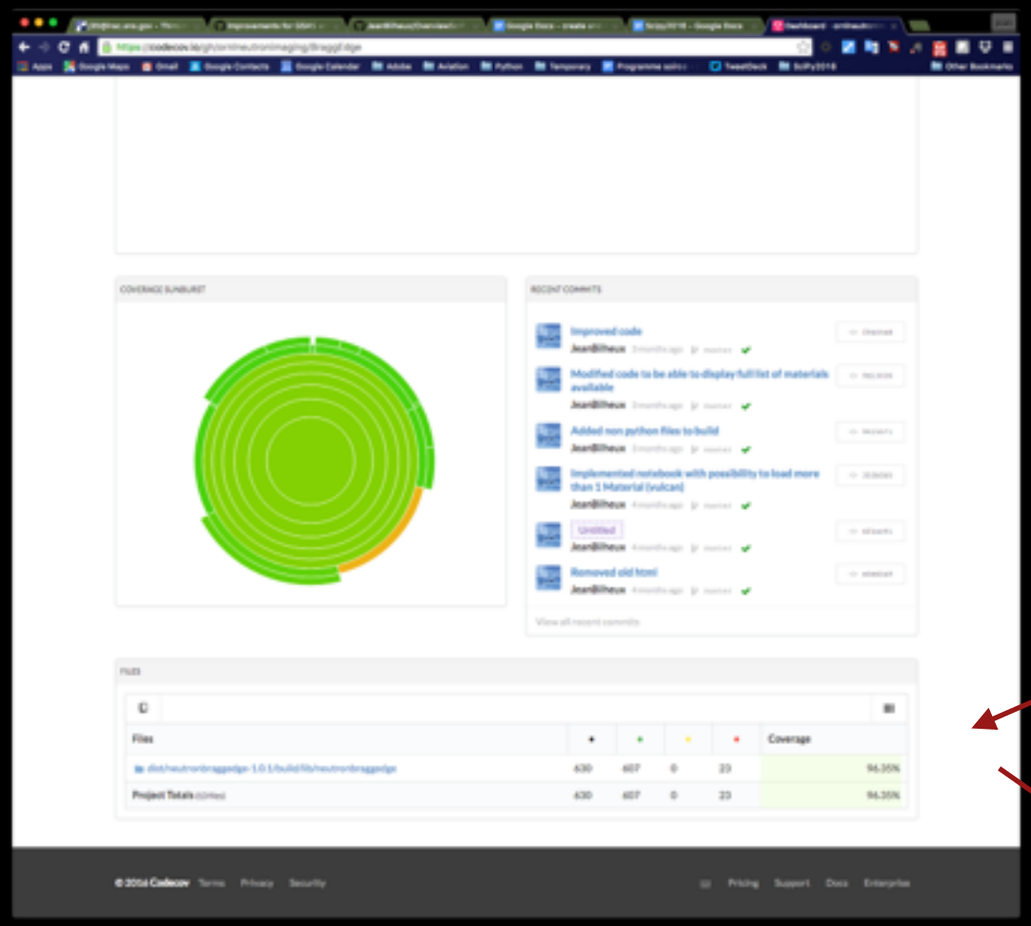
Contents :: How to use This library »

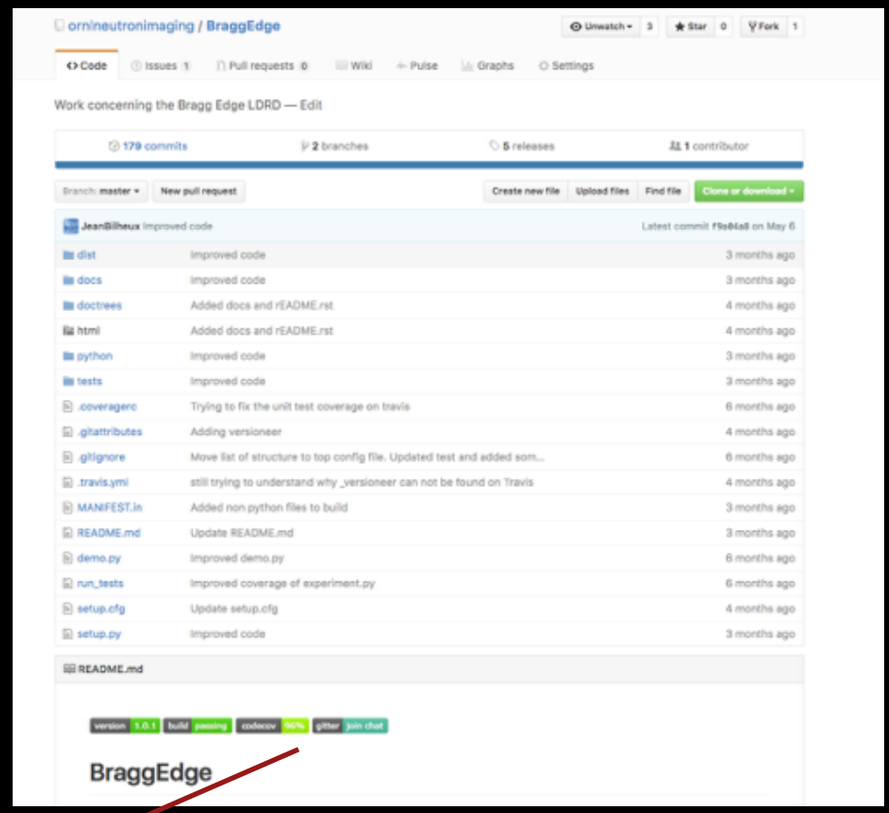
# documentation using sphinx

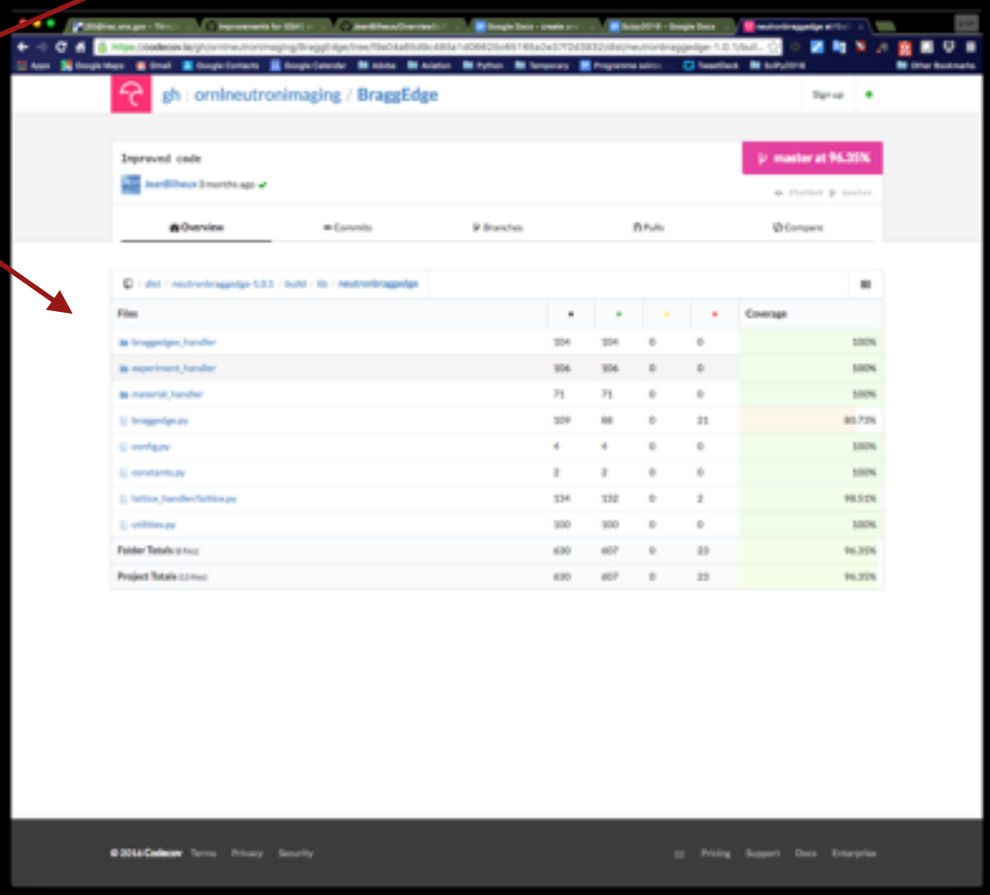



http://www.sphinx-doc.org/en/stable/

# test coverage





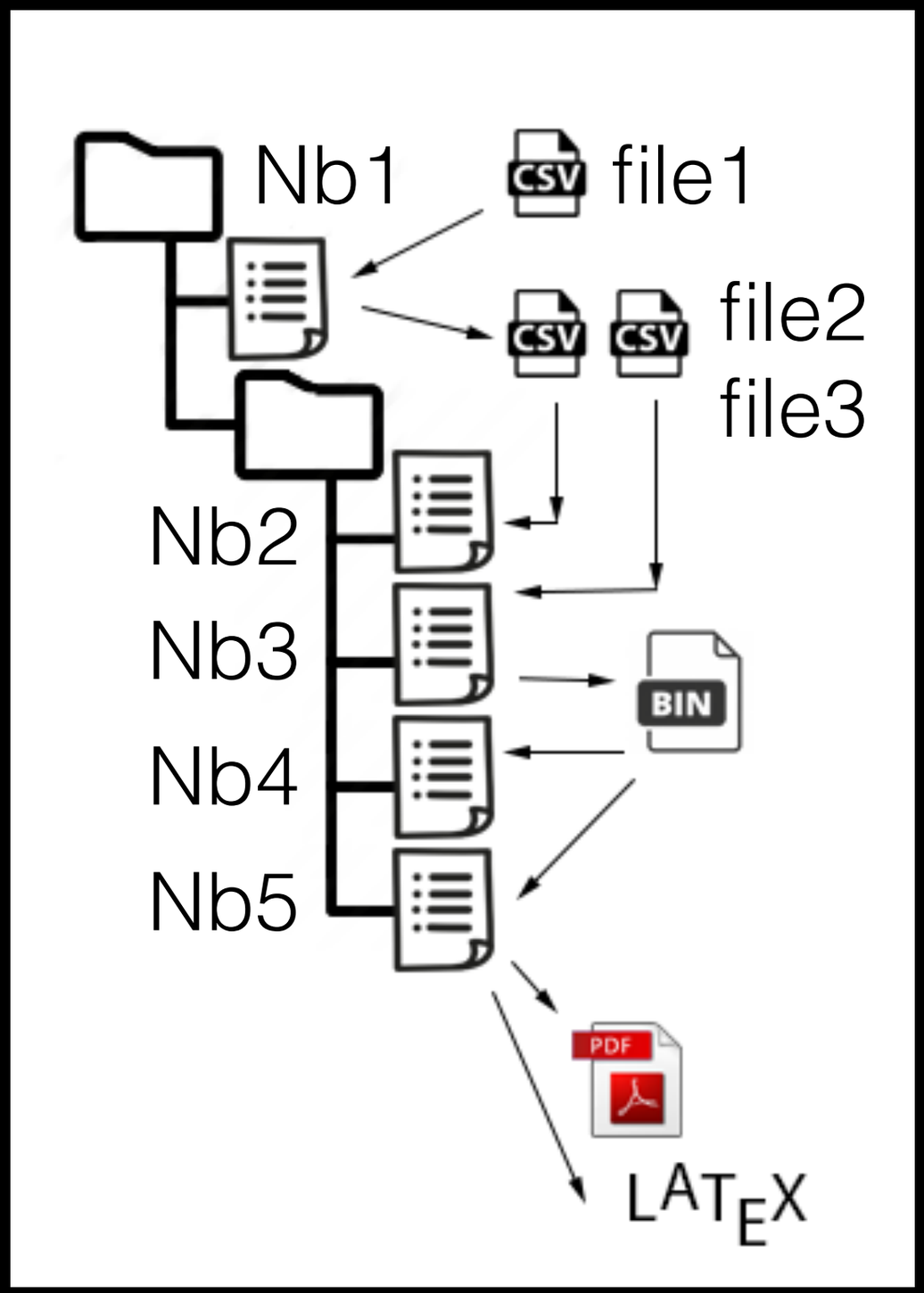

https://codecov.io/


SciPy2016

# Contents

- nbflow
- jupyter lab
- spyder 3.0
- bokeh
  - holoviews
  - datashader
- parallel python

# nbflow




## Problem

If I modify Nb3, which other notebooks do I need to run ?

## Solution

Define in each notebook its dependencies

# nbflow

Define the decencies at the top of the notebook

**__depends__ = ["../results/data.json"]**
**__dest__ = "../results/stats.json"**

92 lines (91 sloc) | 1.73 KB — Raw | Blame | History

In [ ]:
```
__depends__ = ["../data/human_raw.json"]
__dest__ = "../results/human.csv"
```

In [ ]:
```
import pandas as pd
import json
import datetime
```

In [ ]:
```
# Load the JSON formatted data
with open(__depends__[0], "r") as fh:
    data = json.load(fh)
data[0]
```

In [ ]:
```
# Convert it to a pandas DataFrame
df = pd.DataFrame(data).set_index(["participant", "stimulus"]).sortlevel()
df['timestamp'] = df['timestamp'].apply(datetime.datetime.fromtimestamp)
df.head()
```

In [ ]:
```
# Save it to CSV
df.to_csv(__dest__)
```

# nbflow

**To run the magic library**

**install nbflow**
**> pip2 install git+git://github.com/jhamrick/nbflow.git**

**Move to top folder and edit SConstruct file**

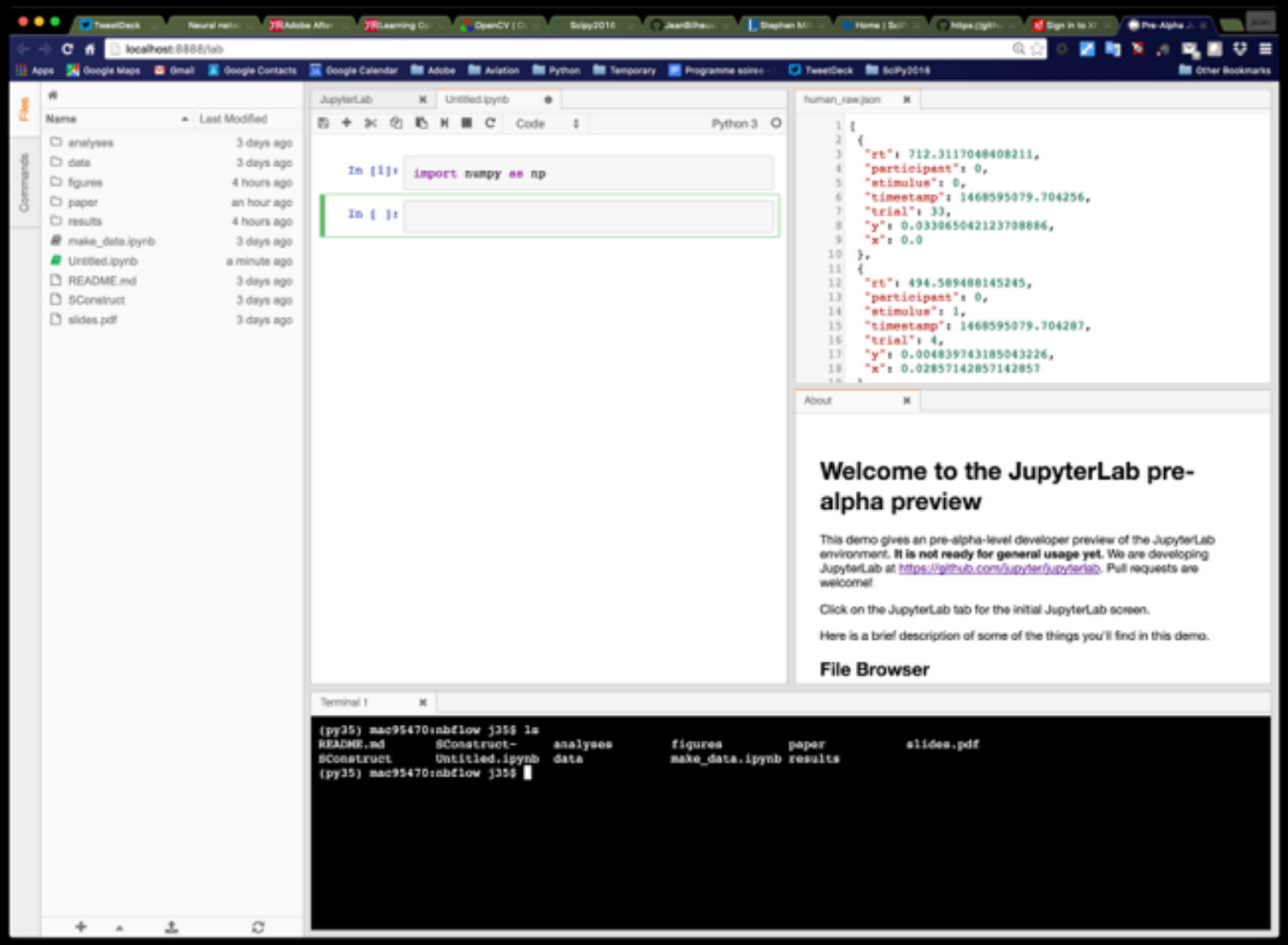
12 lines (9 sloc) | 300 Bytes

```
1  import os
2  import glob
3  from nbflow.scons import setup
4
5  env = Environment(ENV=os.environ)
6  setup(env, ["analyses"])
7
8  results = sorted(glob.glob("results/*.tex"))
9  figures = sorted(glob.glob("figures/*.pdf"))
10 env.PDF("paper/paper.pdf", "paper/paper.tex")
11 env.Depends("paper/paper.pdf", results + figures)
```

**Then run library**

**> scons**

# jupyter lab




File browser, notebooks, terminal, file editor, all in one window, inside the browser.

beta version and aiming for a 1.0 release before next SciPy

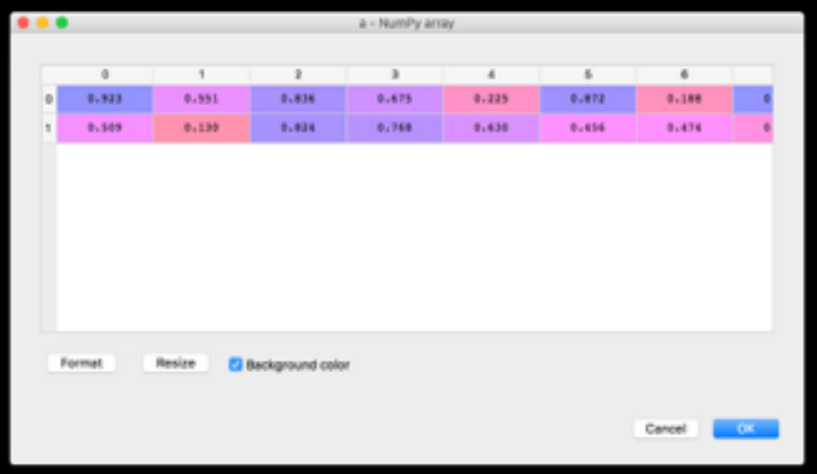
Current version should not be used for development!

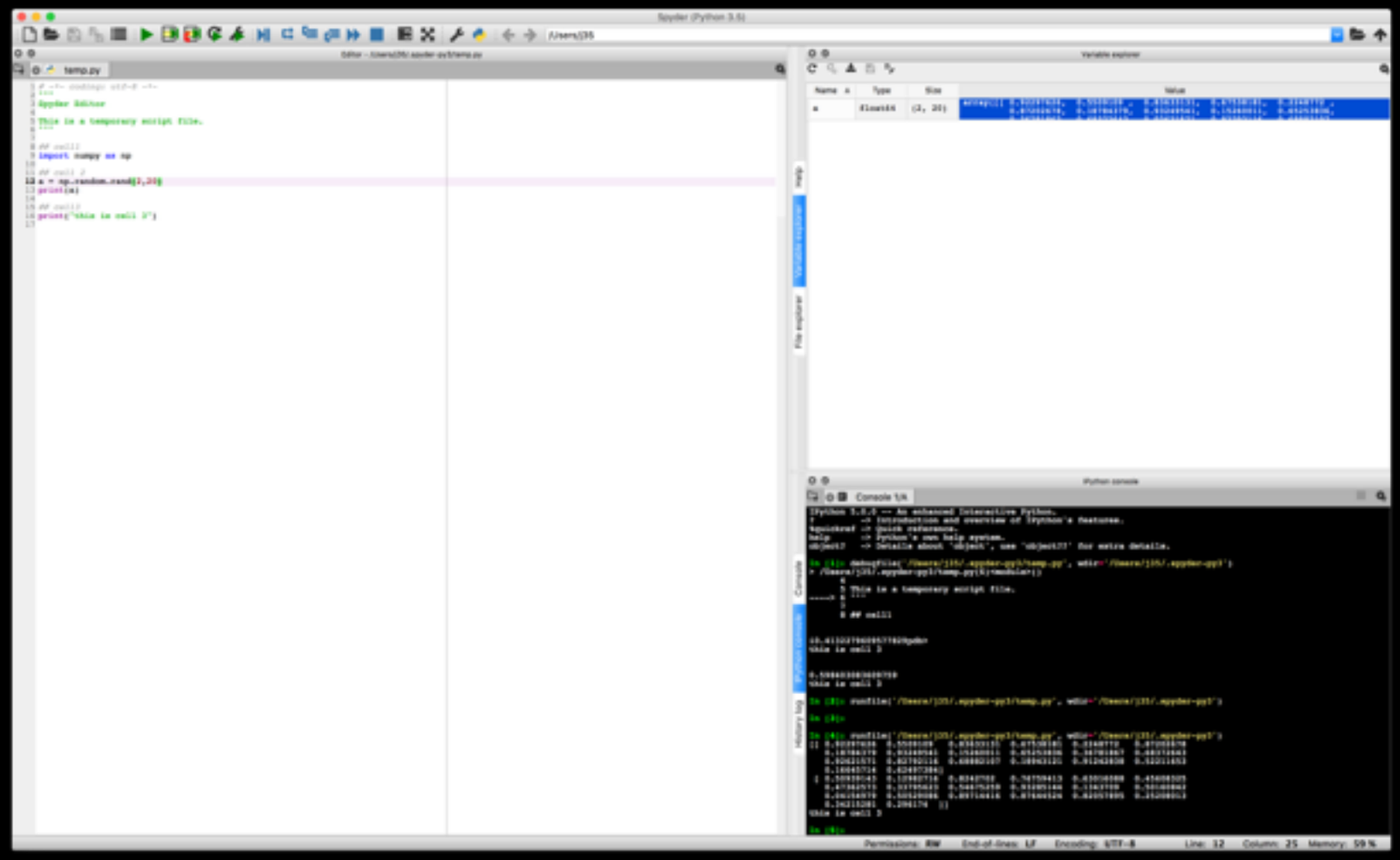
https://github.com/jupyter/jupyterlab

# spyder 3.0

## What's new

- work with projects
- plugins
  - ex: new buttons
- conda graphical installer
- line profiler



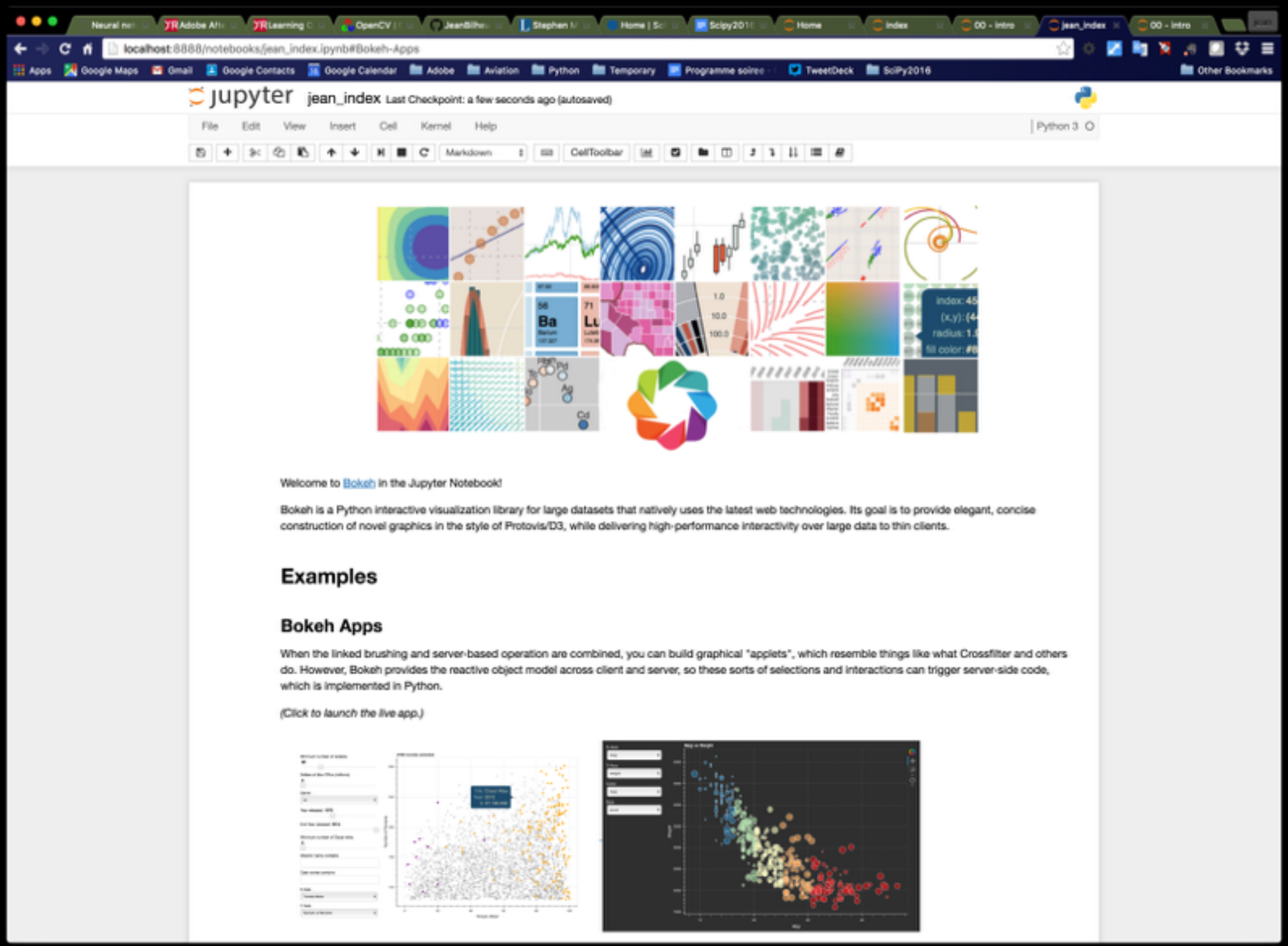



Tool to make python looks like matlab IDE

## My opinion

- concurrent of jupyter lab
- will be a great tool for playing with data
- still has some GUI issues

# bokeh (0.12.0)




demo -> OverviewSciPy2016/bokeh/jean_index.ipynb

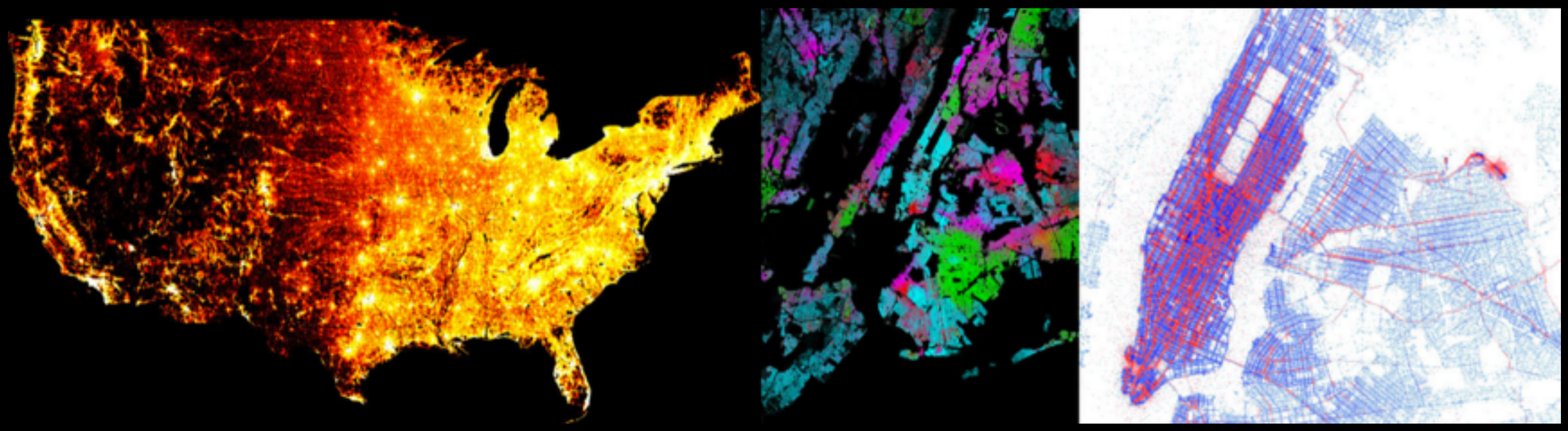
# holoviews

Stop plotting your data -
annotate your data and let it
visualize itself

> conda install bokeh pandas ipython-notebook ipywidgets

demo -> Overview/SciPy2016/holoviews/index.ipynb

# datashader: visualizing big data sets



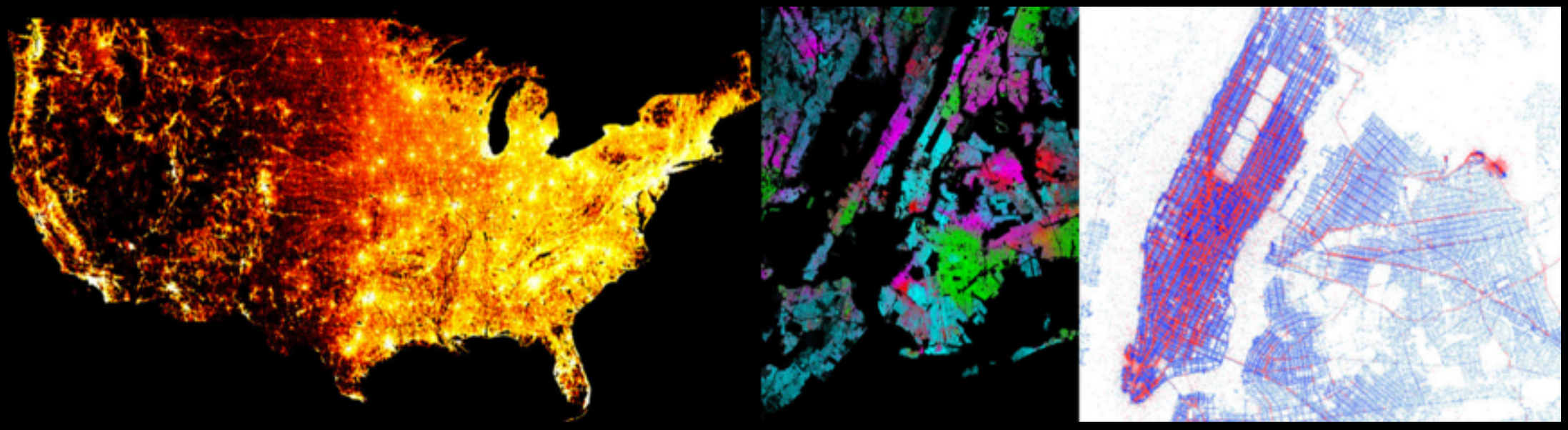
## When not to use data shader

- Plotting less than 1e5 or 1e6 data points
- When every datapoint matters; standard Bokeh will render all of them
- For full interactivity (hover tools) with every datapoint

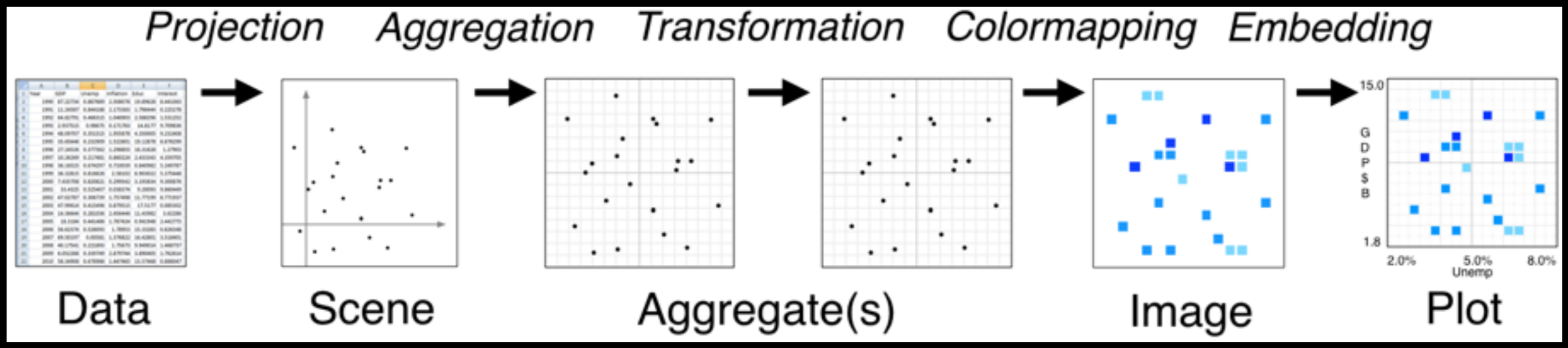
## When to use data shader

- Actual big data; when Bokeh/Matplotlib have trouble
- When the distribution matters more than individual points
- When you find yourself sampling or binning to better understand the distribution

# datashader: visualizing big data sets

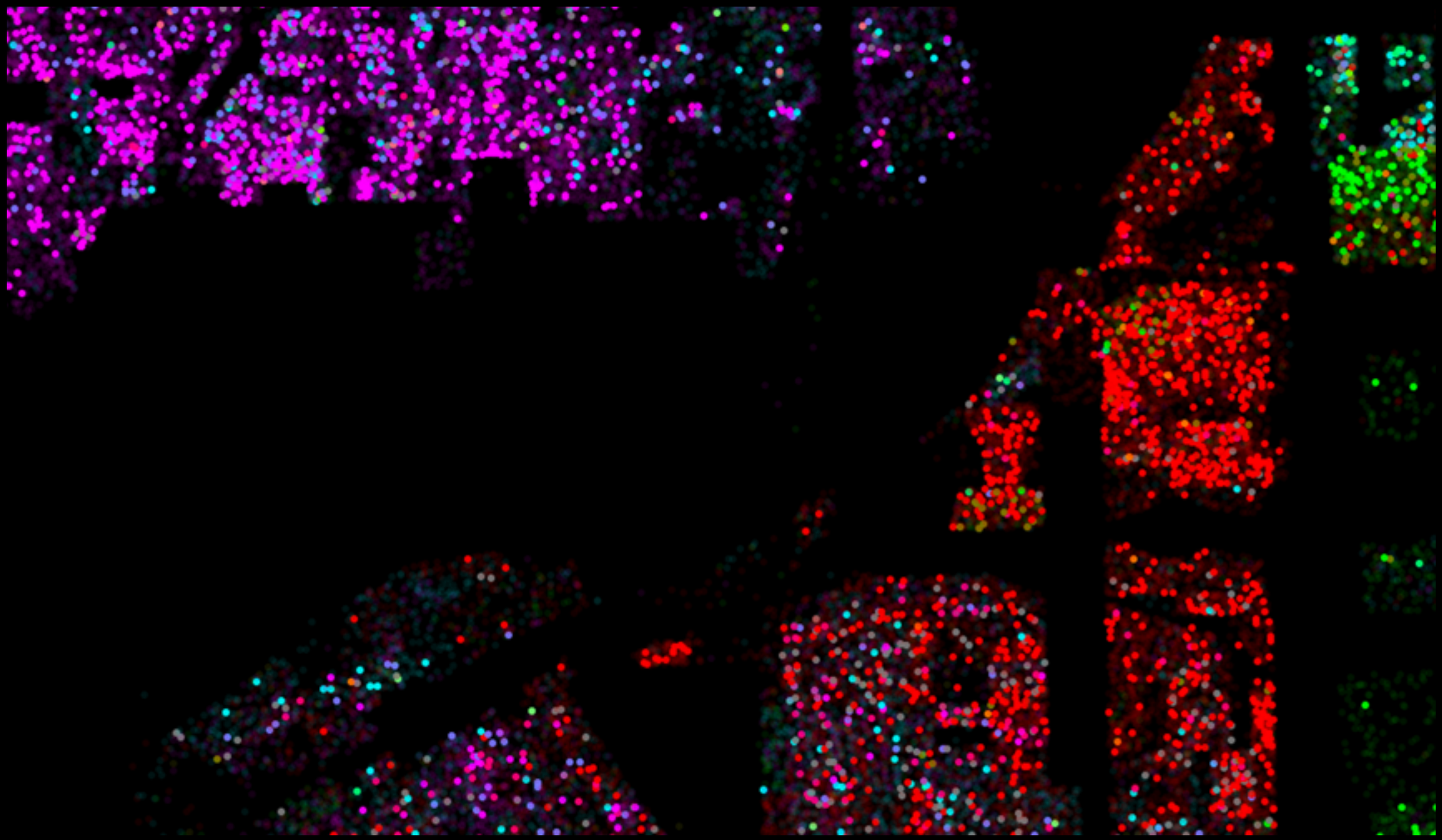


## How does it work




demo -> OverviewSciPy2016/datashader/index.ipynb

# datashader: visualizing big data sets



# parallel python

- mpi4py (https://github.com/JeanBilheux/python_tutorial/tree/master/python/parallelization)
- map
- Submit

http://matthewrocklin.com/scipy-2016-parallel/algorithms.html#/0/1

demo -> Overview/SciPy2016/parallel/

# Thank you

github.com/JeanBilheux/OverviewSciPy2016