Traitement de données avec

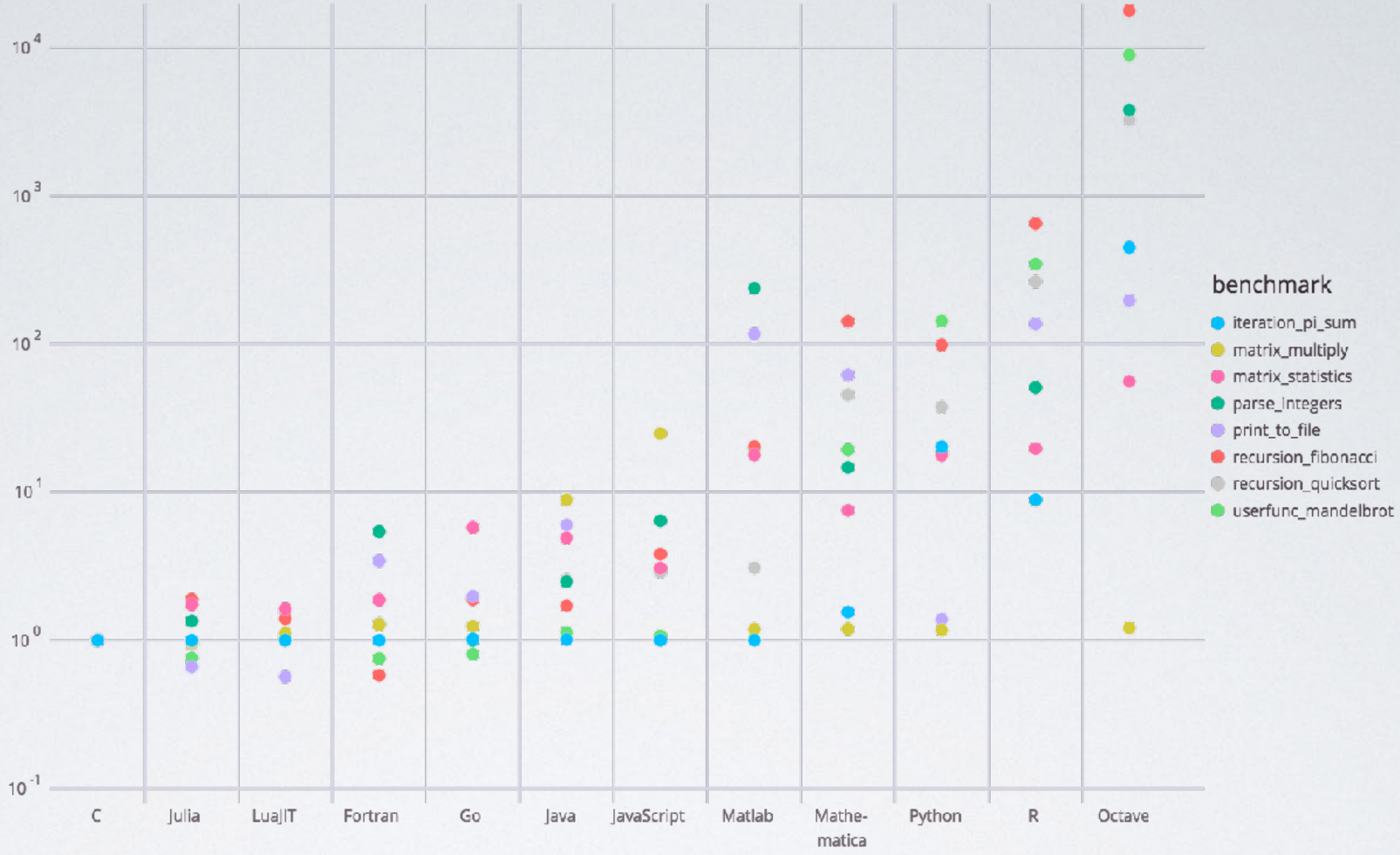
PYTHON

POURQUOI PYTHON?

Nous aide à être plus efficace dans notre travail de chercheur
en tirant avantage des outils disponibles

- Bénéficie d'un large écosystème d'outils matures : ex. : Numpy, Pandas, SciPy, Matplotlib, IPython/Jupyter, etc.
- Très large éventail d'outils développés par d'autres utilisateurs offert en code libre

Source



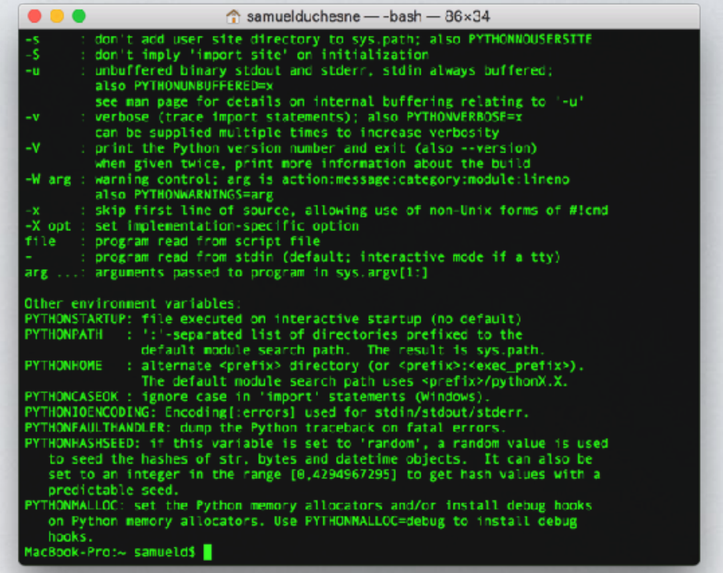
COMPARAISON

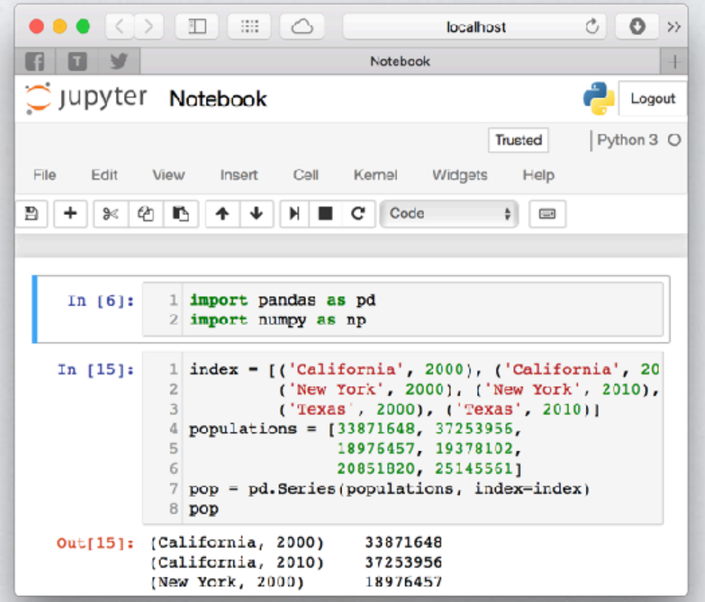
avec d'autres langages de programmation

TERMINAL OU NOTEBOOK

If Python is the engine of our data science task, you might think of IPython as the interactive control panel.

— Jake Vanderplast





Téléchargez l'example

EXEMPLE

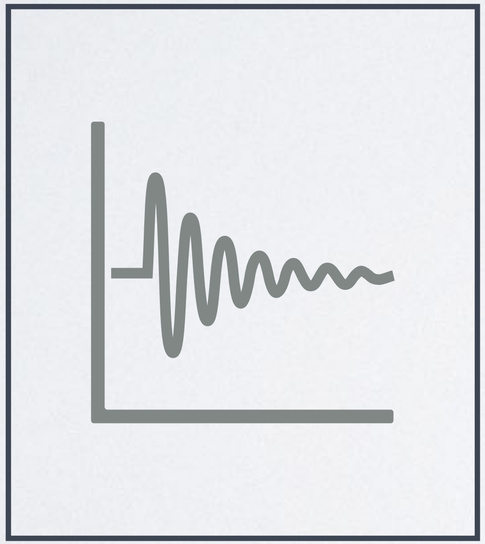
BemSolar

Démonstration d'un code simple permettant de récupérer, traiter et analyser les données de rayonnement solaire du National Oceanic and Atmospheric Administration.

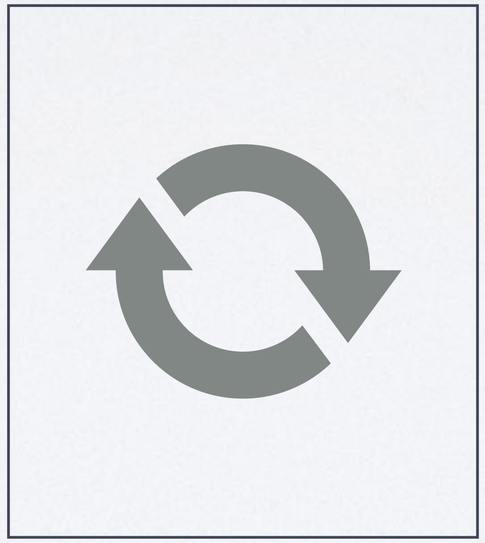
Python, Jupyter, Github

Une stratégie de travail efficace en trois parties

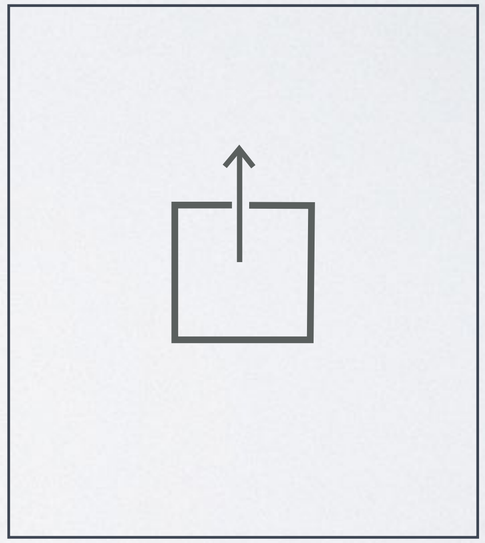
Explorer



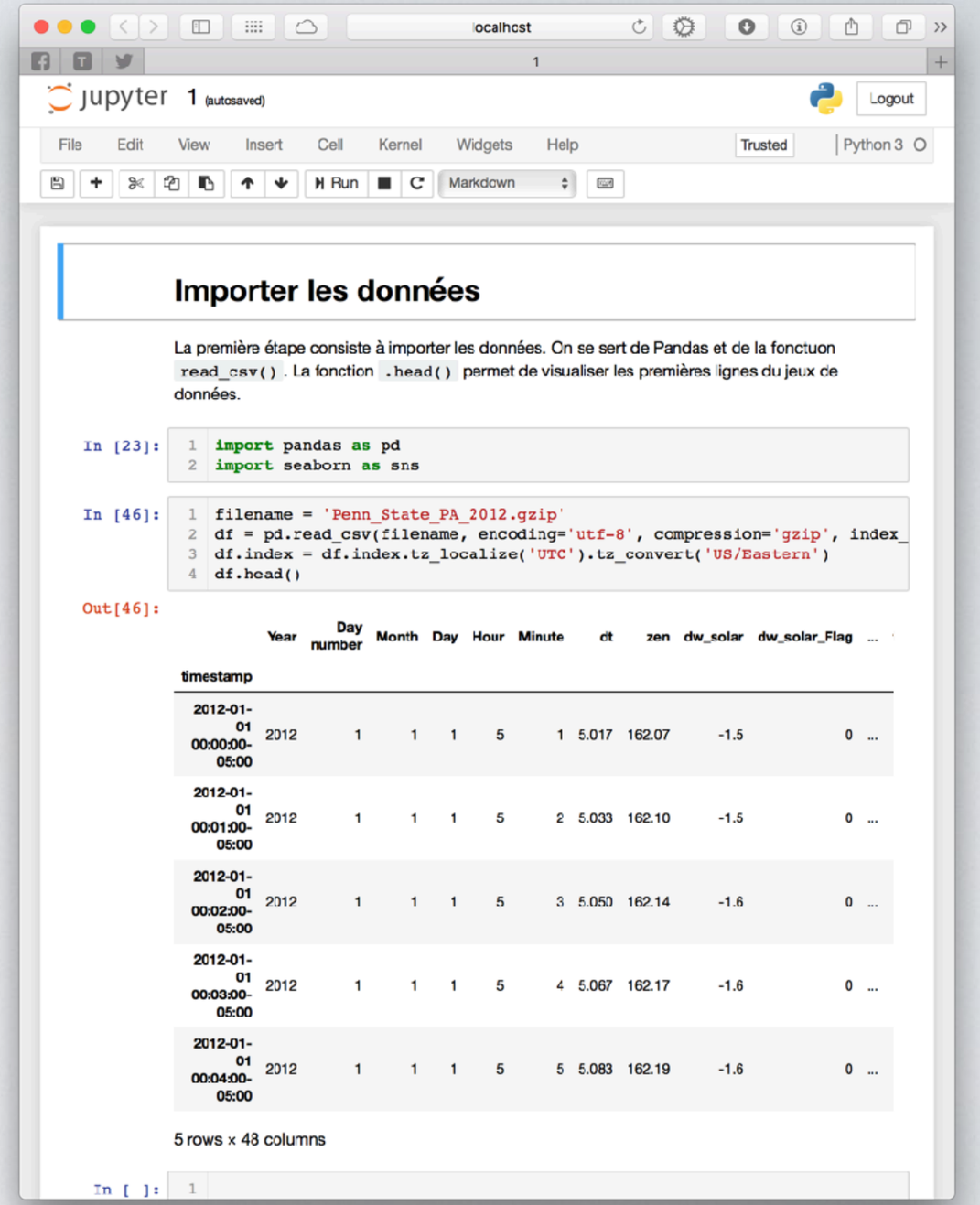
Reconstituer



Partager



IMPORTER LES DONNÉES



Importer les données

La première étape consiste à importer les données. On se sert de Pandas et de la fonctuon `read_csv()`. La fonction `.head()` permet de visualiser les premières lignes du jeux de données.

In [23]:
```
import pandas as pd
import seaborn as sns
```

In [46]:
```
filename = 'Penn_State_PA_2012.gzip'
df = pd.read_csv(filename, encoding='utf-8', compression='gzip', index_
df.index = df.index.tz_localize('UTC').tz_convert('US/Eastern')
df.head()
```

Out[46]:

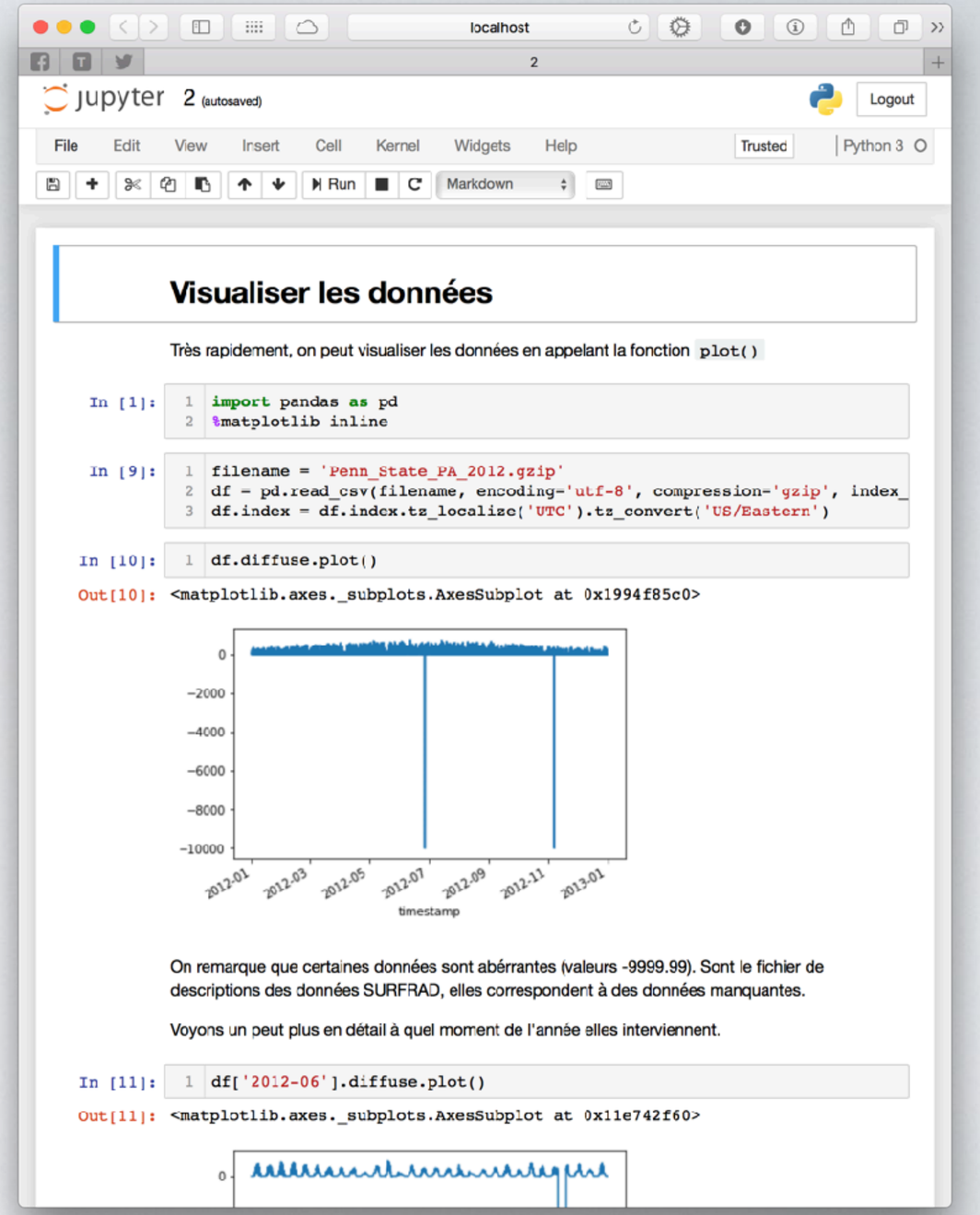
timestamp	Year	Day number	Month	Day	Hour	Minute	dt	zen	dw_solar	dw_solar_Flag	...
2012-01-01 00:00:00-05:00	2012	1	1	1	5	1	5.017	162.07	-1.5	0	...
2012-01-01 00:01:00-05:00	2012	1	1	1	5	2	5.033	162.10	-1.5	0	...
2012-01-01 00:02:00-05:00	2012	1	1	1	5	3	5.050	162.14	-1.6	0	...
2012-01-01 00:03:00-05:00	2012	1	1	1	5	4	5.067	162.17	-1.6	0	...
2012-01-01 00:04:00-05:00	2012	1	1	1	5	5	5.083	162.19	-1.6	0	...

5 rows × 48 columns

In [ ]:

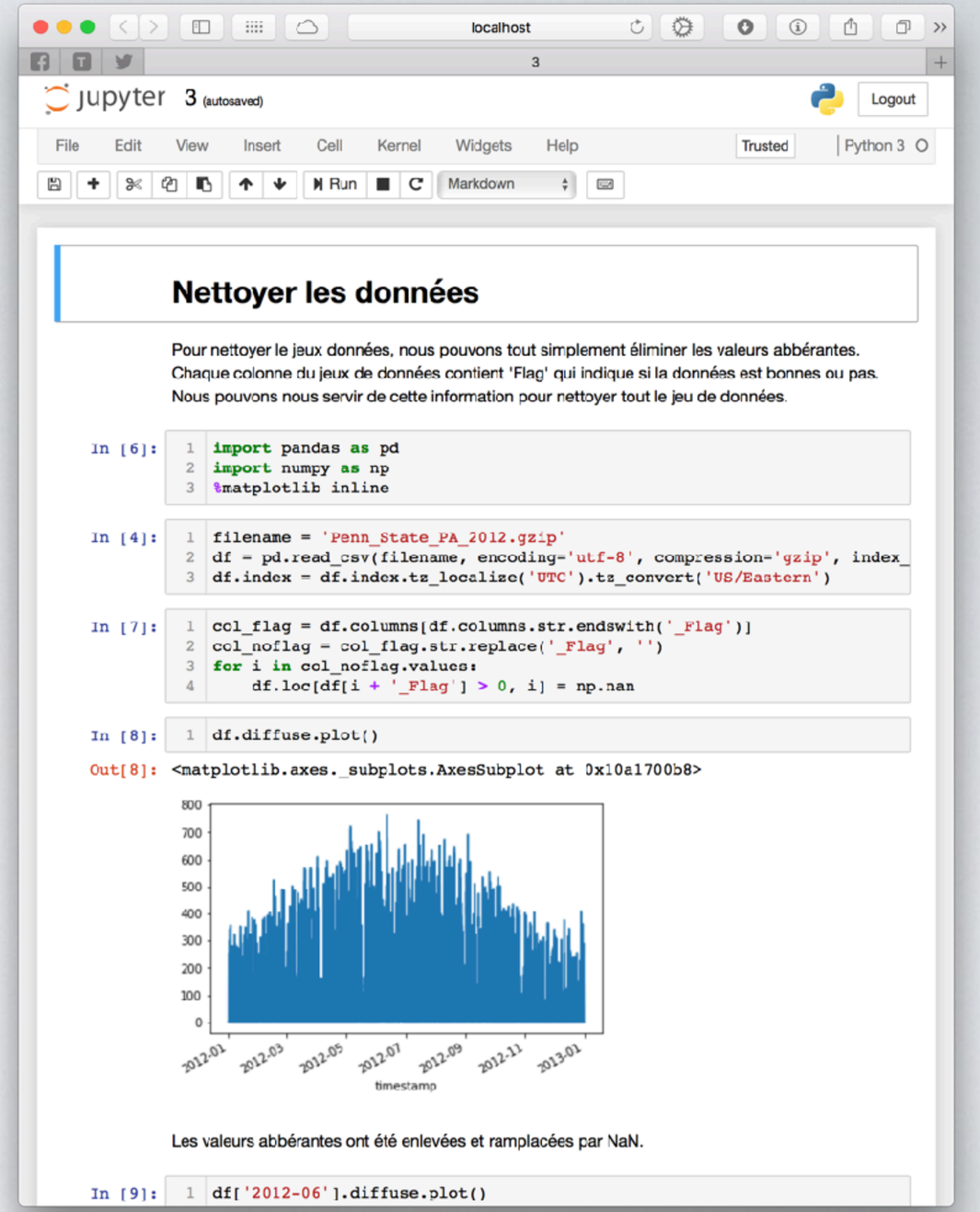
VISUALISER LES DONNÉES

2



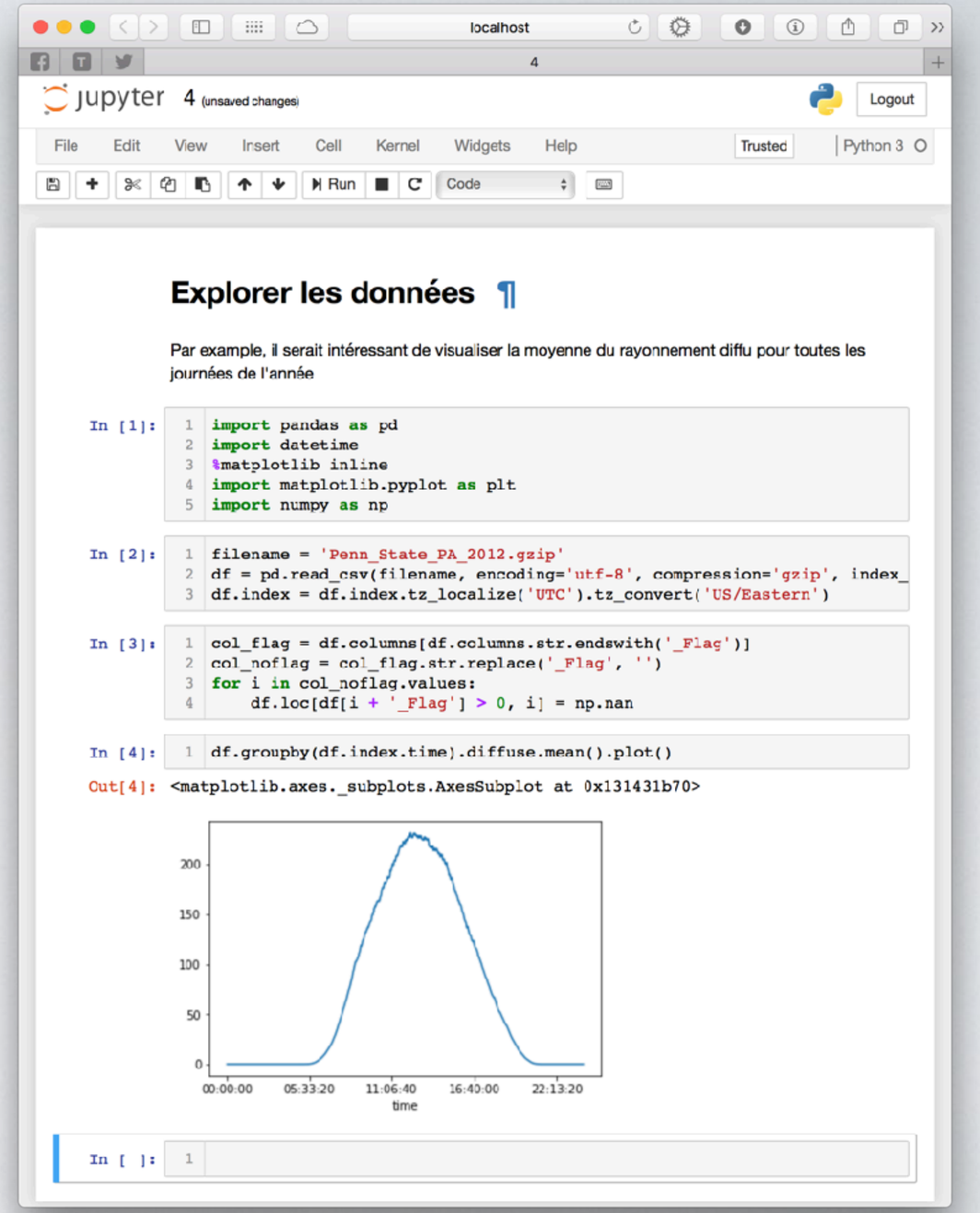
NETTOYER LES DONNÉES

3



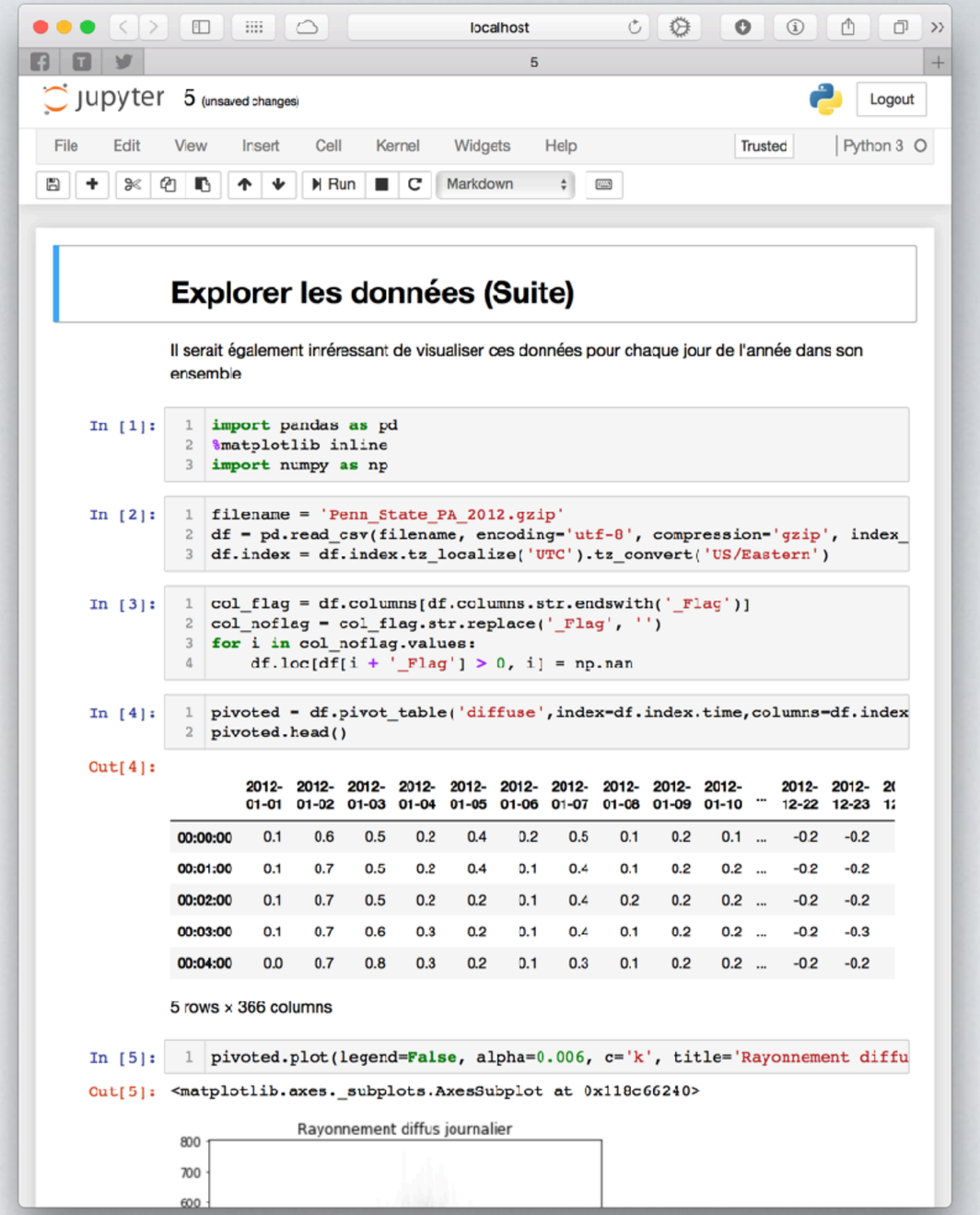
EXPLORER LES DONNÉES

4



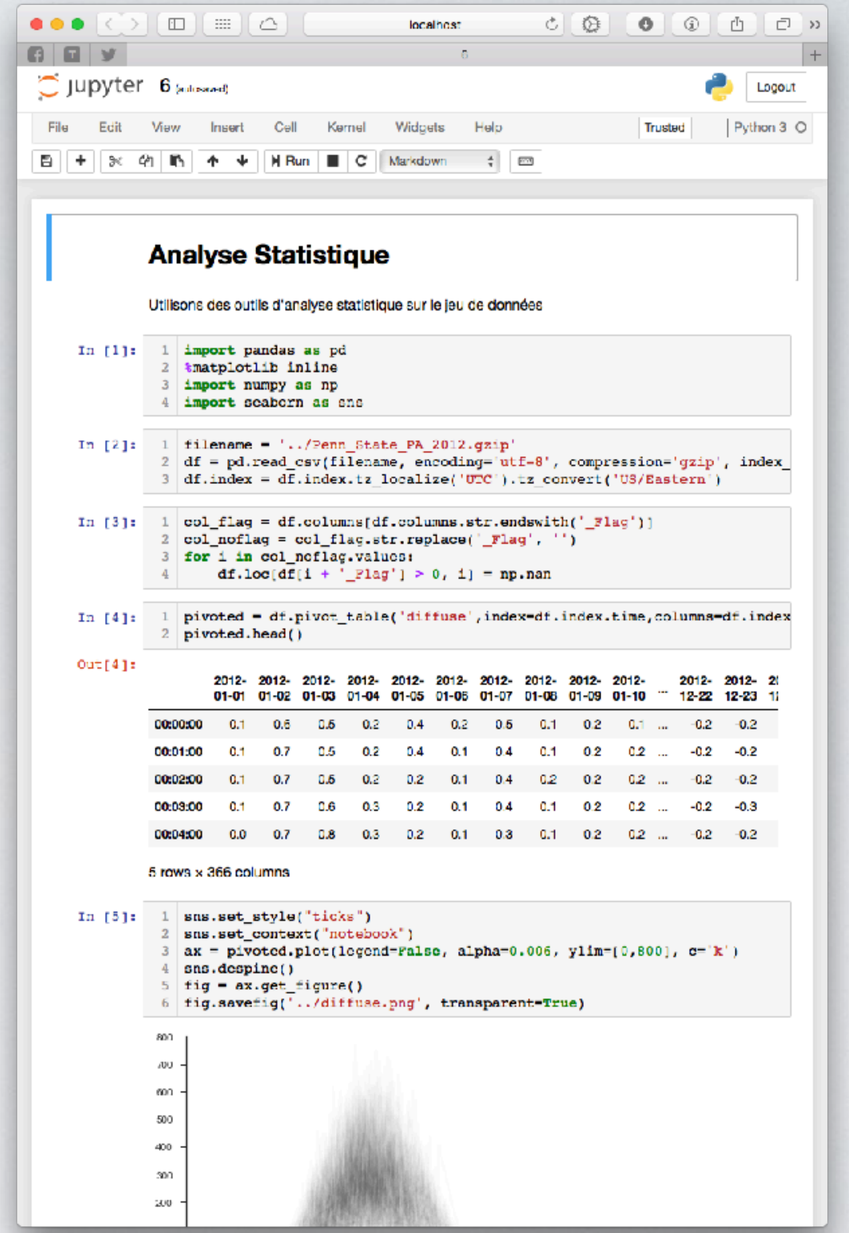
EXPLORER LES DONNÉES (SUITE)

5



ANALYSE STATISTIQUE

6



Analyse Statistique

Utilisons des outils d'analyse statistique sur le jeu de données

In [1]:
```
import pandas as pd
%matplotlib inline
import numpy as np
import seaborn as sns
```

In [2]:
```
filename = '../Penn_State_PA_2012.gzip'
df = pd.read_csv(filename, encoding= utf-8', compression='gzip', index_
df.index = df.index.tz_localize('UTC').tz_convert('US/Eastern')
```

In [3]:
```
col_flag = df.columns[df.columns.str.endswith('_Flag')]
col_noflag = col_flag.str.replace('_Flag', '')
for i in col_noflag.values:
    df.loc[df[i + '_Flag'] > 0, i] = np.nan
```

In [4]:
```
pivoted = df.pivot_table('diffuse',index=df.index.time,columns=df.index
pivoted.head()
```

Out[4]:

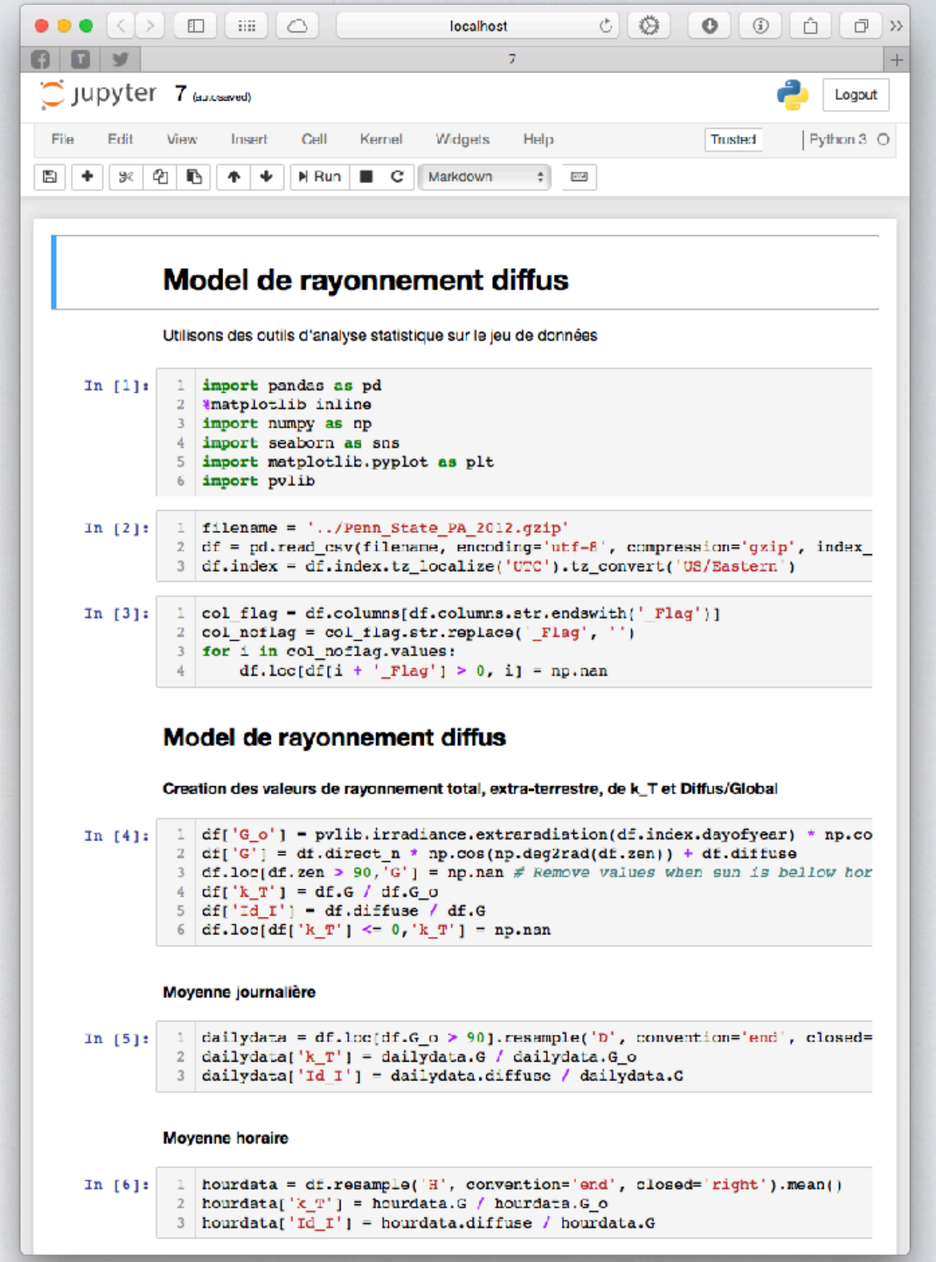
	2012-01-01	2012-01-02	2012-01-03	2012-01-04	2012-01-05	2012-01-06	2012-01-07	2012-01-08	2012-01-09	2012-01-10	...	2012-12-22	2012-12-23	2 1
00:00:00	0.1	0.6	0.5	0.2	0.4	0.2	0.5	0.1	0.2	0.1	...	0.2	0.2	
00:01:00	0.1	0.7	0.5	0.2	0.4	0.1	0.4	0.1	0.2	0.2	...	-0.2	-0.2	
00:02:00	0.1	0.7	0.5	0.2	0.2	0.1	0.4	0.2	0.2	0.2	...	0.2	0.2	
00:03:00	0.1	0.7	0.6	0.3	0.2	0.1	0.4	0.1	0.2	0.2	...	-0.2	-0.3	
00:04:00	0.0	0.7	0.8	0.3	0.2	0.1	0.3	0.1	0.2	0.2	...	0.2	0.2	

5 rows x 366 columns

In [5]:
```
sns.set_style('ticks')
sns.set_context('notebook')
ax = pivoted.plot(legend=False, alpha=0.006, ylim=[0,800], c='k')
sns.despine()
fig = ax.get_figure()
fig.savefig('../diffuse.png', transparent=True)
```

MODÈLE DE RAYONNEMENT DIFFUS

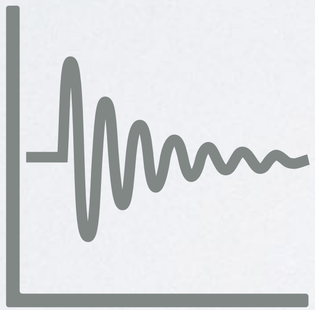
7



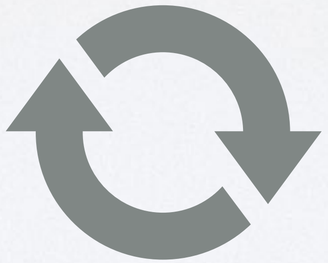
PYTHON

Nous aide à être plus efficace dans notre travail de chercheur
en tirant avantage des outils disponibles

Explorer



Reconstituer



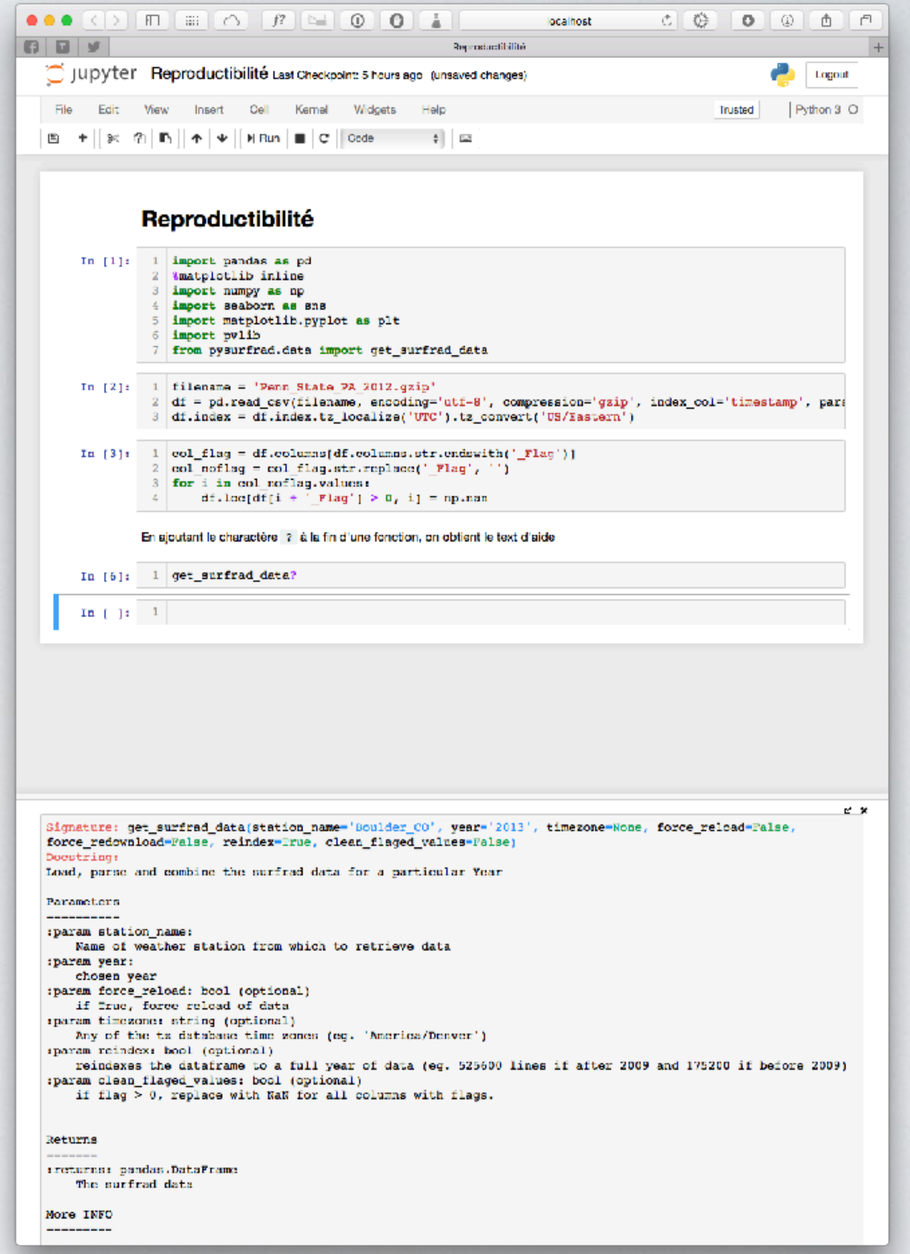
Partager

RÉPÉTER

S'assurer que le code est reproductible

simplement cliquer sur :

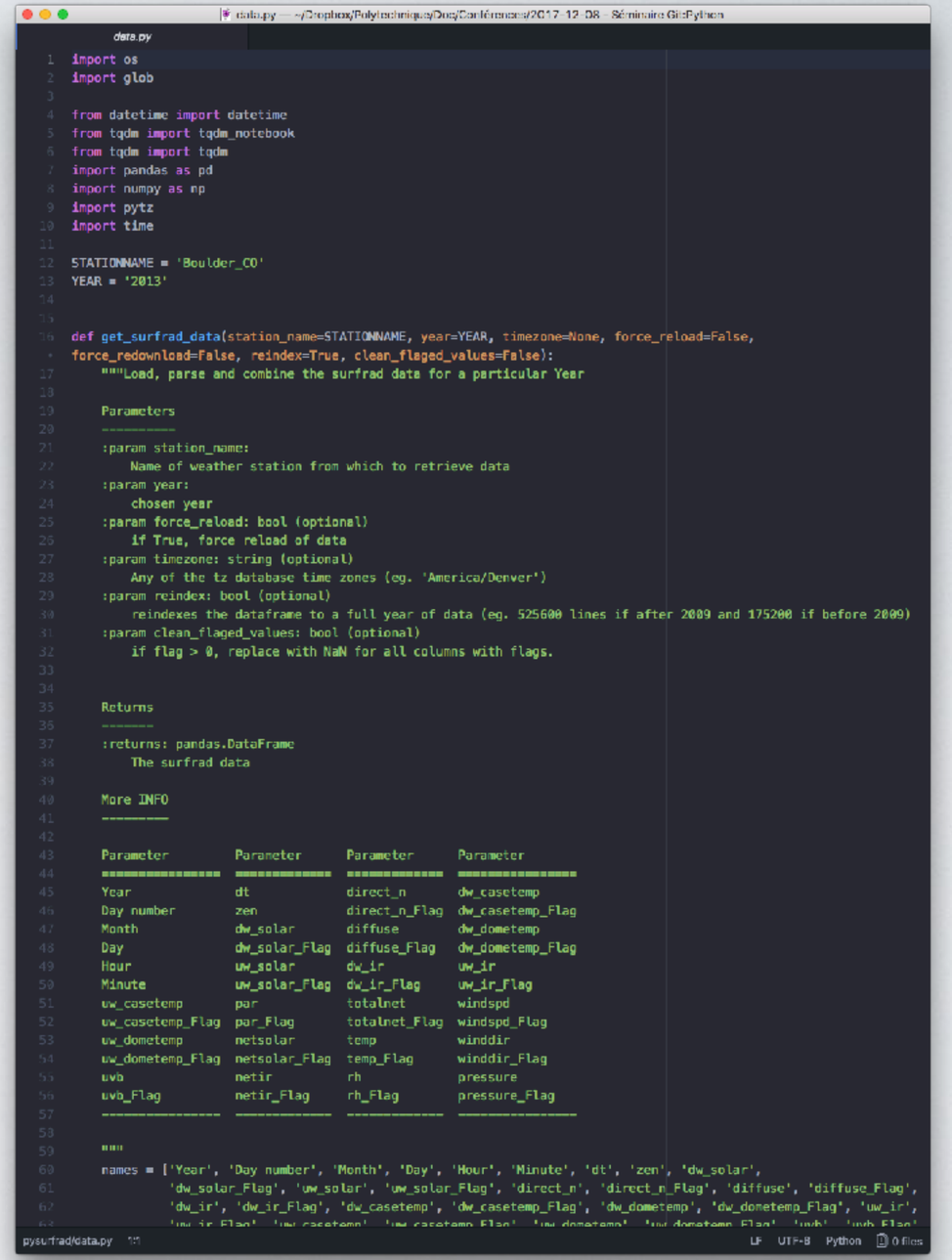
`Kernel –> Restart & Run All`



RÉPÉTER

Créer un « python package »

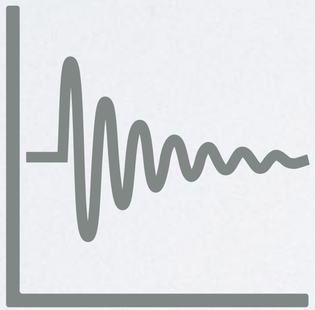
Une librairie de fonction que l'on peut importer dans les premières lignes du carnet



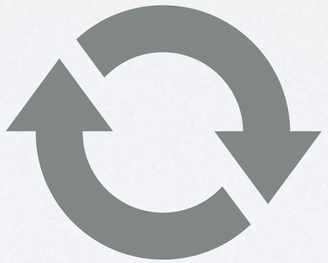
PYTHON

Nous aide à être plus efficace dans notre travail de chercheur
en tirant avantage des outils disponibles

Explorer



Reconstituer

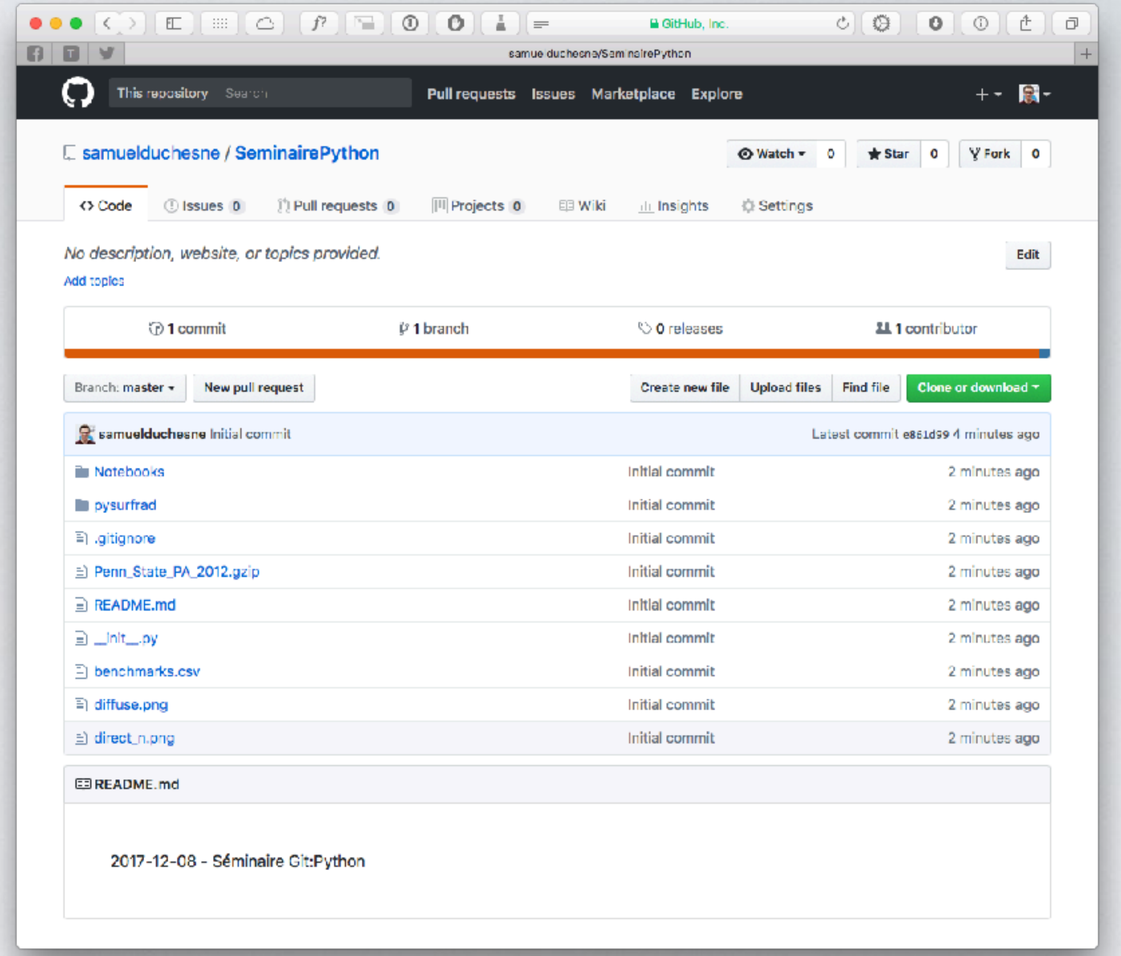


Partager

PARTAGER

Les « Notebooks » sont facilement partagés en lignes

github, nbviewer, etc.

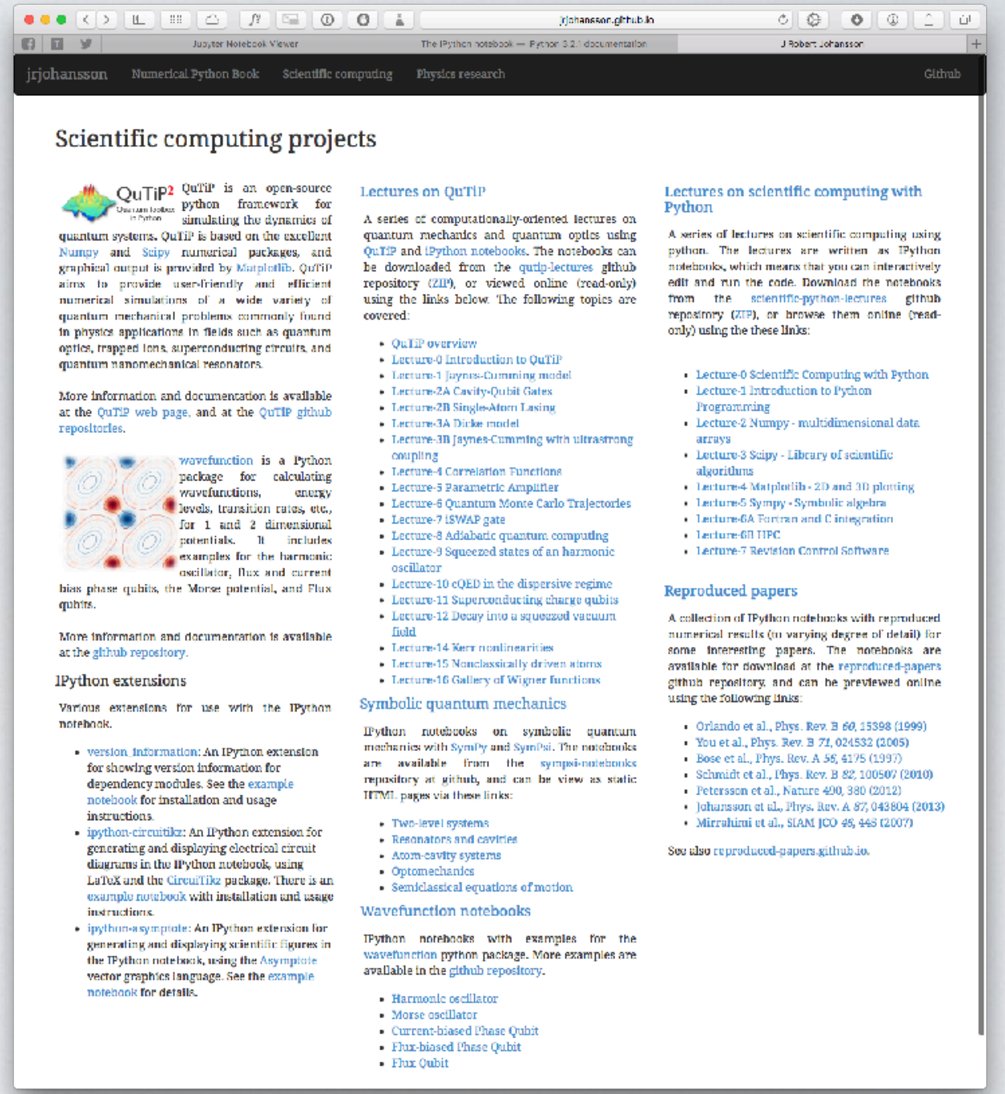


IPython Notebook

Pour des notes de cours en ligne

J Robert Johansson, Ph.D.

physique des solides théoriques et de calcul, mécanique quantique dans les circuits électriques supraconducteurs.



jrjohansson.github.io

Jupyter Notebook Viewer | The IPython notebook — IPython 3.2.1 documentation | J Robert Johansson

jrjohansson | Numerical Python Book | Scientific computing | Physics research | Github

Scientific computing projects

QuTiP² Quantum Toolbox in Python

QuTiP is an open-source python framework for simulating the dynamics of quantum systems. QuTiP is based on the excellent Numpy and Scipy numerical packages, and graphical output is provided by Matplotlib. QuTiP aims to provide user-friendly and efficient numerical simulations of a wide variety of quantum mechanical problems commonly found in physics applications in fields such as quantum optics, trapped ions, superconducting circuits, and quantum nanomechanical resonators.

More information and documentation is available at the QuTiP web page, and at the QuTiP github repositories.

wavefunction is a Python package for calculating wavefunctions, energy levels, transition rates, etc., for 1 and 2 dimensional potentials. It includes examples for the harmonic oscillator, flux and current bias phase qubits, the Morse potential, and Flux qubits.

More information and documentation is available at the github repository.

IPython extensions

Various extensions for use with the IPython notebook.

- version_information: An IPython extension for showing version information for dependency modules. See the example notebook for installation and usage instructions.
- ipython-circuitikz: An IPython extension for generating and displaying electrical circuit diagrams in the IPython notebook, using LaTeX and the CircuiTikz package. There is an example notebook with installation and usage instructions.
- ipython-asymptote: An IPython extension for generating and displaying scientific figures in the IPython notebook, using the Asymptote vector graphics language. See the example notebook for details.

Lectures on QuTiP

A series of computationally-oriented lectures on quantum mechanics and quantum optics using QuTiP and IPython notebooks. The notebooks can be downloaded from the qutip-lectures github repository (ZIP), or viewed online (read-only) using the links below. The following topics are covered:

- QuTiP overview
- Lecture-0 Introduction to QuTiP
- Lecture-1 Jaynes-Cumming model
- Lecture-2A Cavity-Qubit Gates
- Lecture-2B Single-Atom Lasing
- Lecture-3A Dicke model
- Lecture-3B Jaynes-Cumming with ultrastrong coupling
- Lecture-4 Correlation Functions
- Lecture-5 Parametric Amplifier
- Lecture-6 Quantum Monte Carlo Trajectories
- Lecture-7 ISWAP gate
- Lecture-8 Adiabatic quantum computing
- Lecture-9 Squeezed states of an harmonic oscillator
- Lecture-10 cQED in the dispersive regime
- Lecture-11 Superconducting charge qubits
- Lecture-12 Decay into a squeezed vacuum field
- Lecture-14 Kerr nonlinearities
- Lecture-15 Nonclassically driven atoms
- Lecture-16 Gallery of Wigner functions

Symbolic quantum mechanics

IPython notebooks on symbolic quantum mechanics with SymPy and SymPsi. The notebooks are available from the sympsi-notebooks repository at github, and can be view as static HTML pages via these links:

- Two-level systems
- Resonators and cavities
- Atom-cavity systems
- Optomechanics
- Semiclassical equations of motion

Wavefunction notebooks

IPython notebooks with examples for the wavefunction python package. More examples are available in the github repository.

- Harmonic oscillator
- Morse oscillator
- Current-biased Phase Qubit
- Flux-biased Phase Qubit
- Flux Qubit

Lectures on scientific computing with Python

A series of lectures on scientific computing using python. The lectures are written as IPython notebooks, which means that you can interactively edit and run the code. Download the notebooks from the scientific-python-lectures github repository (ZIP), or browse them online (read-only) using the these links:

- Lecture-0 Scientific Computing with Python
- Lecture-1 Introduction to Python Programming
- Lecture-2 Numpy - multidimensional data arrays
- Lecture-3 Scipy - Library of scientific algorithms
- Lecture-4 Matplotlib - 2D and 3D plotting
- Lecture-5 Sympy - Symbolic algebra
- Lecture-6A Fortran and C integration
- Lecture-6B HPC
- Lecture-7 Revision Control Software

Reproduced papers

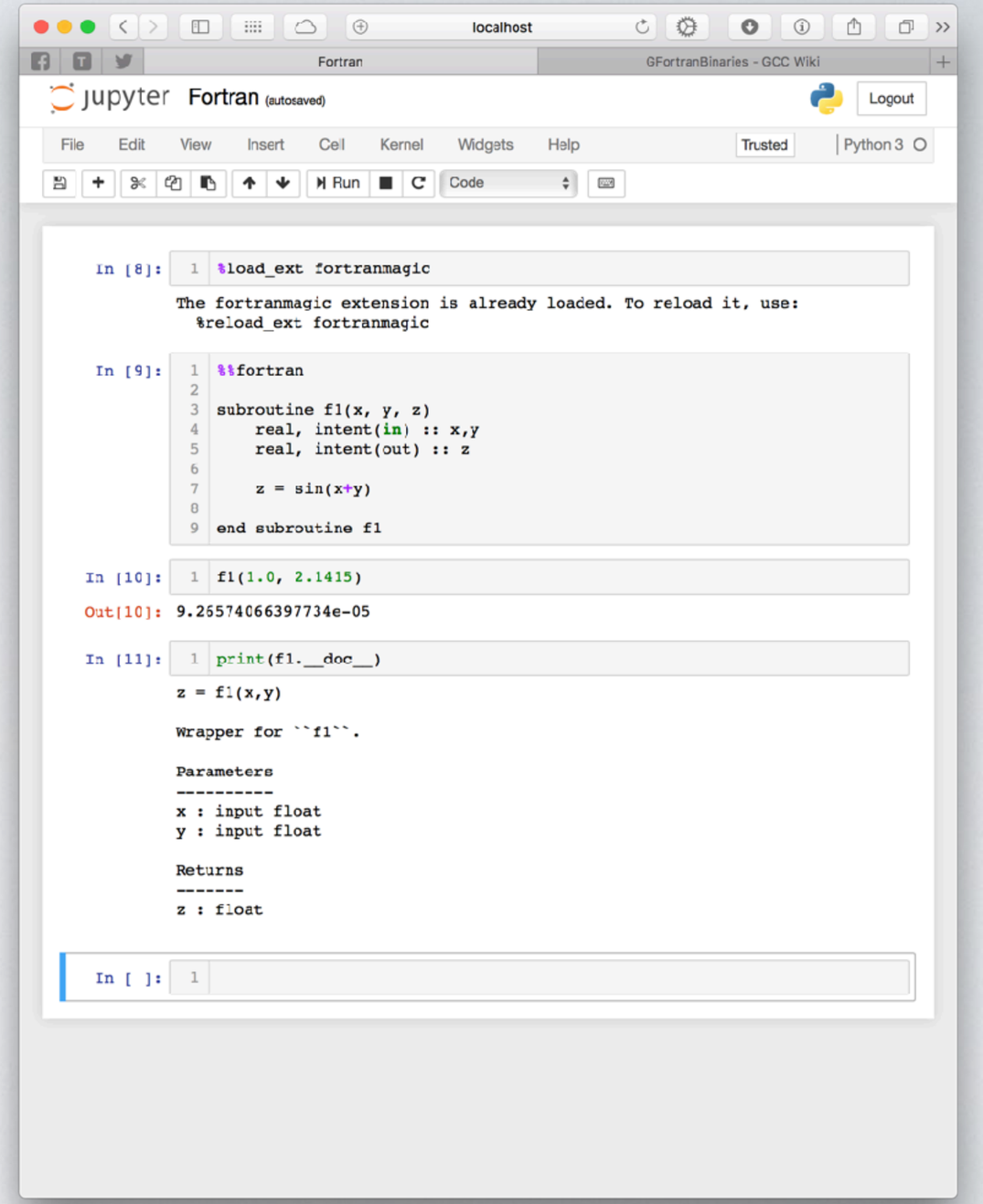
A collection of IPython notebooks with reproduced numerical results (to varying degree of detail) for some interesting papers. The notebooks are available for download at the reproduced-papers github repository, and can be previewed online using the following links:

- Orlando et al., Phys. Rev. B 60, 15398 (1999)
- You et al., Phys. Rev. B 71, 024532 (2005)
- Bose et al., Phys. Rev. A 56, 4175 (1997)
- Schmidt et al., Phys. Rev. B 82, 100507 (2010)
- Petersson et al., Nature 490, 380 (2012)
- Johansson et al., Phys. Rev. A 87, 043804 (2013)
- Mirrahimi et al., SIAM JCO 46, 448 (2007)

See also reproduced-papers.github.io.

FORTRAN

et oui!



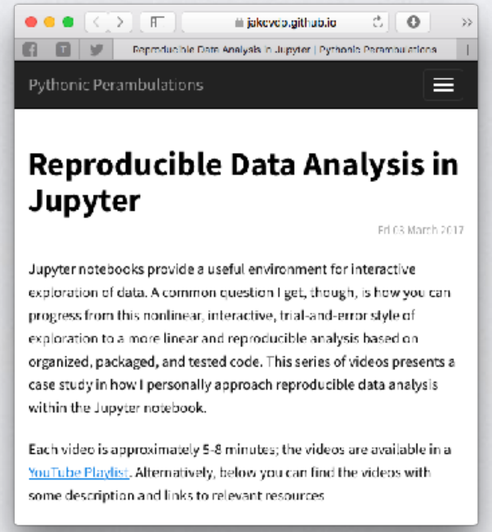
RESSOURCES

En ligne

Pour installer Python

Pour apprendre

Pour se rappeler



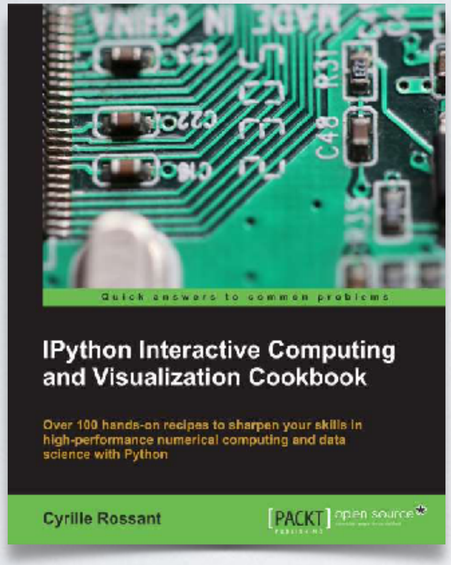
Anaconda
Installation facile de Python

Reproducible Data Analysis in Jupyter
Jake VanderPlas

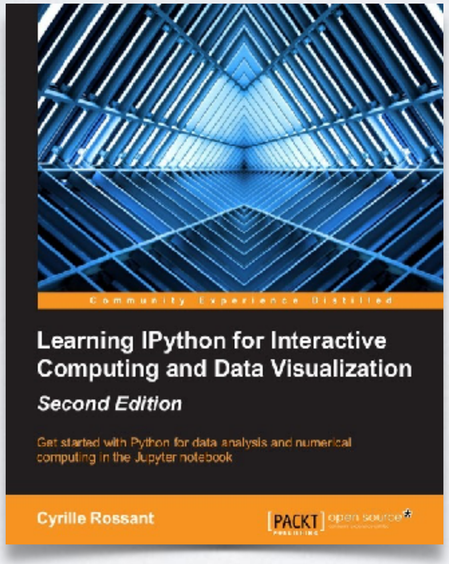
Python Pandas Cheat Sheet
Aide mémoire pour Pandas

RESSOURCES

Livres

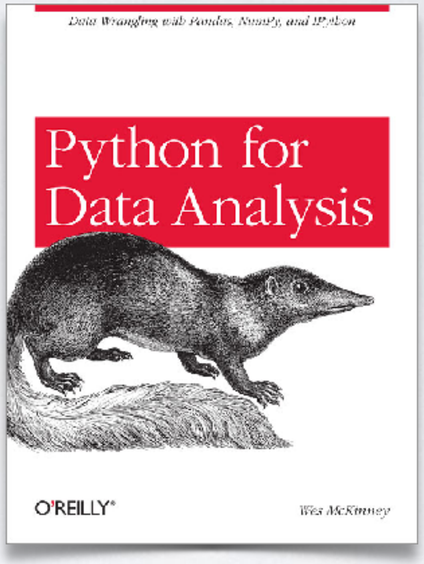


IPython Interactive Computing and Visualization Cookbook
Cyrille Rossant

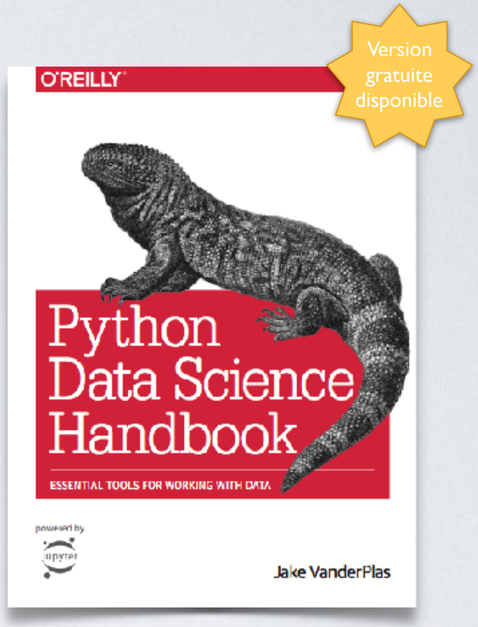


Learning IPython for Interactive Computing and Data Visualization, second edition
Cyrille Rossant

A Whirlwind Tour of Python
Jake VanderPlas



Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython
William McKinney



Python Data Science Handbook
Jake VanderPlas