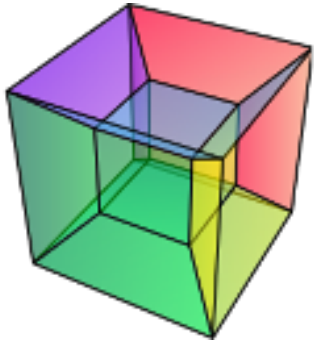


HyperSpy Community & Github

Eric Prestat

University of Manchester and SuperSTEM Laboratory



- HyperSpy on the web
- What is Github? Why is it so popular?
- What makes the HyperSpy community?



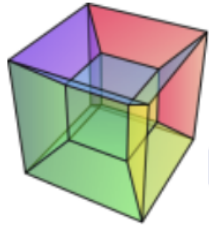
'Enhanced Data Generated by Electrons'

Following the very successful workshops at Lake Tahoe, Leukerbad, Port Ludlow, Guadeloupe, Grundlsee and Banff, the next international workshop on electron energy loss spectroscopy and imaging will be held in

Sainte Maxime, France,

- Francisco's talk on HyperSpy
 - Mostly about open source and scientific python community
 - NOT about the cool stuff that HyperSpy could already do at the time
- Open source is not only about opening the code but also about community
- *Small* paradigm shift required





HyperSpy

multi-dimensional data analysis

[Home](#) · [Download](#) · [Documentation](#) · [Demos](#) · [News](#) · [Support](#) · [Citing](#) · [Credits](#)

HyperSpy: multi-dimensional data analysis toolbox

HyperSpy is an open source Python library which provides tools to facilitate the interactive data analysis of multi-dimensional datasets that can be described as multi-dimensional arrays of a given signal (e.g. a 2D array of spectra a.k.a spectrum image).

HyperSpy aims at making it easy and natural to apply analytical procedures that operate on an individual signal to multi-dimensional arrays, as well as providing easy access to analytical tools that exploit the multi-dimensionality of the dataset.

Its modular structure makes it easy to add features to analyze different kinds of signals.

Highlights

- Two families of named and scaled axes: *signal* and *navigation*.
- Visualization tools for multi-dimensional spectra and images.
- Easy access multi-dimensional curve fitting and blind source separation.
- Built on top of NumPy, SciPy, matplotlib and scikit-learn.
- Modular design for easy extensibility.

HyperSpy documentation/community support

- Main website:
 - <http://hyperspy.org/>
- Documentation:
 - <http://hyperspy.readthedocs.io/en/stable/>
- Users list:
 - <https://groups.google.com/forum/#!forum/hyperspy-users>
 - Discuss/ask use case of HyperSpy
- Gitter: online chat room
 - <https://gitter.im/hyperspy/hyperspy>

HyperSpy demos

- Source code of the demo on github
 - <https://github.com/hyperspy/hyperspy-demos>
- Non-interactive version (using nbviewer)
 - <http://nbviewer.jupyter.org/github/hyperspy/hyperspy-demos/tree/master/>
 - Github now displays nicely the notebook, nbviewer is not really necessary anymore
- Interactive version (using mybinder)
 - <https://mybinder.org/v2/gh/hyperspy/hyperspy-demos/master>
 - Run demos online (on a remote server) without any installation
 - May be a bit slow...

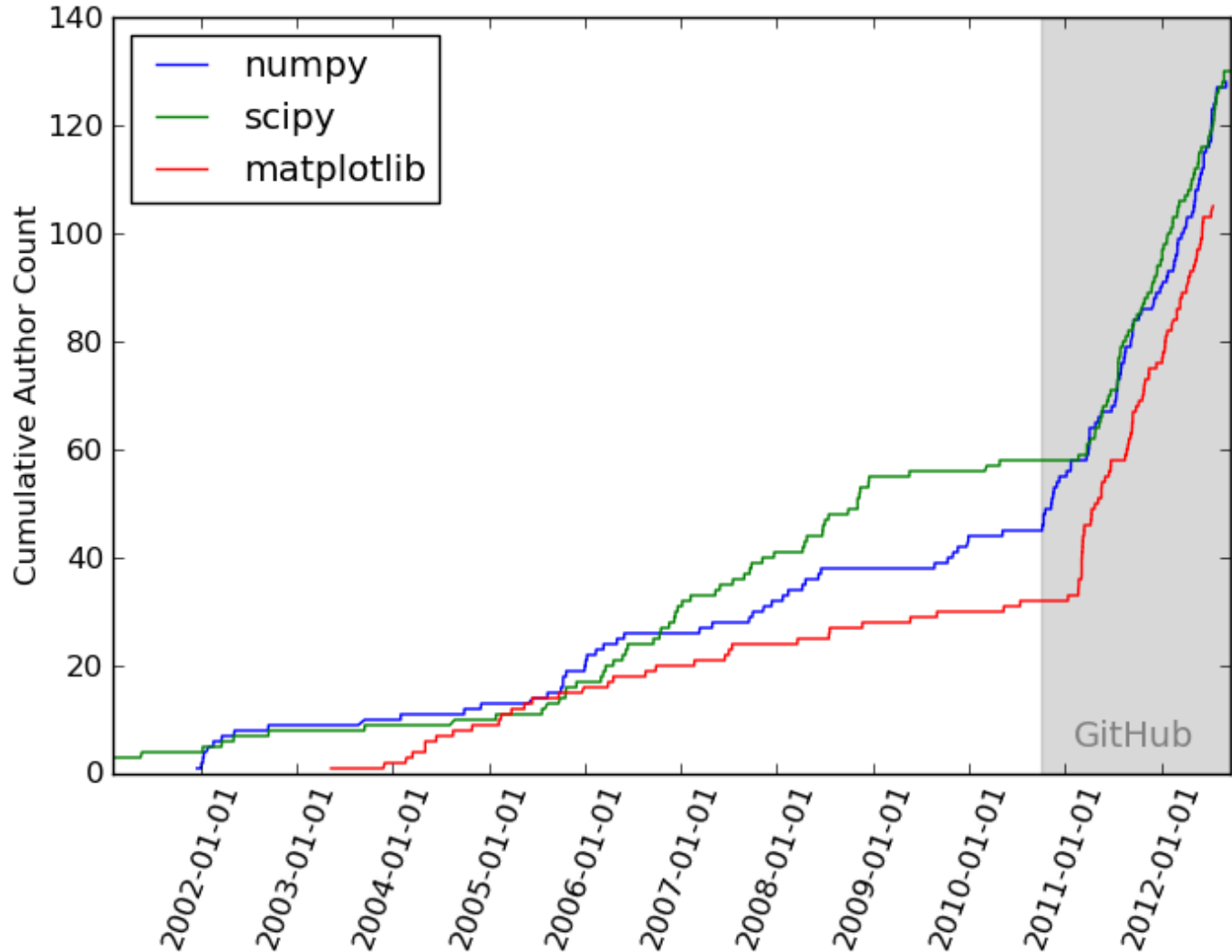
HyperSpy development

- Development site on Github
 - <https://github.com/hyperspy/hyperspy>
 - Issues tracker: report bug, propose new features and other business
 - Pull requests: discuss the merge of any changes with the upstream branches
- Developer guide
 - http://hyperspy.readthedocs.io/en/stable/dev_guide.html

HyperSpy is built on many python libraries

- Strong scientist user base
- The python scientific stack is huge!
 - Generic array computation: [Numpy](#), [scipy](#)
 - More specialise libraries: [scikit-image](#), [scikit-learn](#)
 - Visualisation: [matplotlib](#) & others
 - High-performance computing: [Cython](#), [numba](#), [dask](#)
 - Interactivity/user interface: [jupyter](#), [ipython](#), [spyder](#)
- One big strength of python is his large and dynamic community
 - Progress is fast, which is great!
 - Sometimes, it breaks because it move quickly...

The github effect



From Pythonic perambulations: Why Python is the last language you'll have to learn:

<https://jakevdp.github.io/blog/2012/09/20/why-python-is-the-last/>

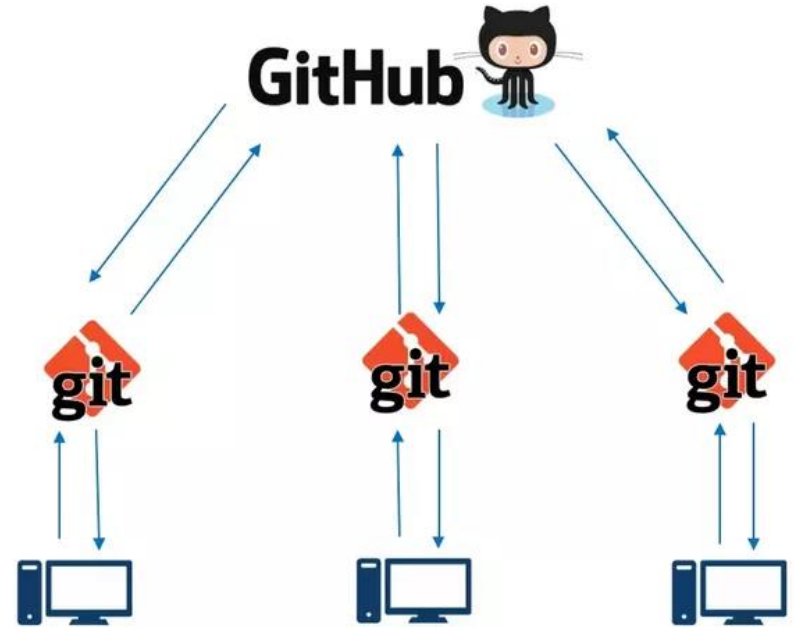
What is github?

- At the heart of Github is Git
 - A version control systems: manages and stores revisions of projects
 - Git is a distributed version control systems
 - Each contributor has is own *remote* (online) repository
 - Code is merged in the *upstream* repository

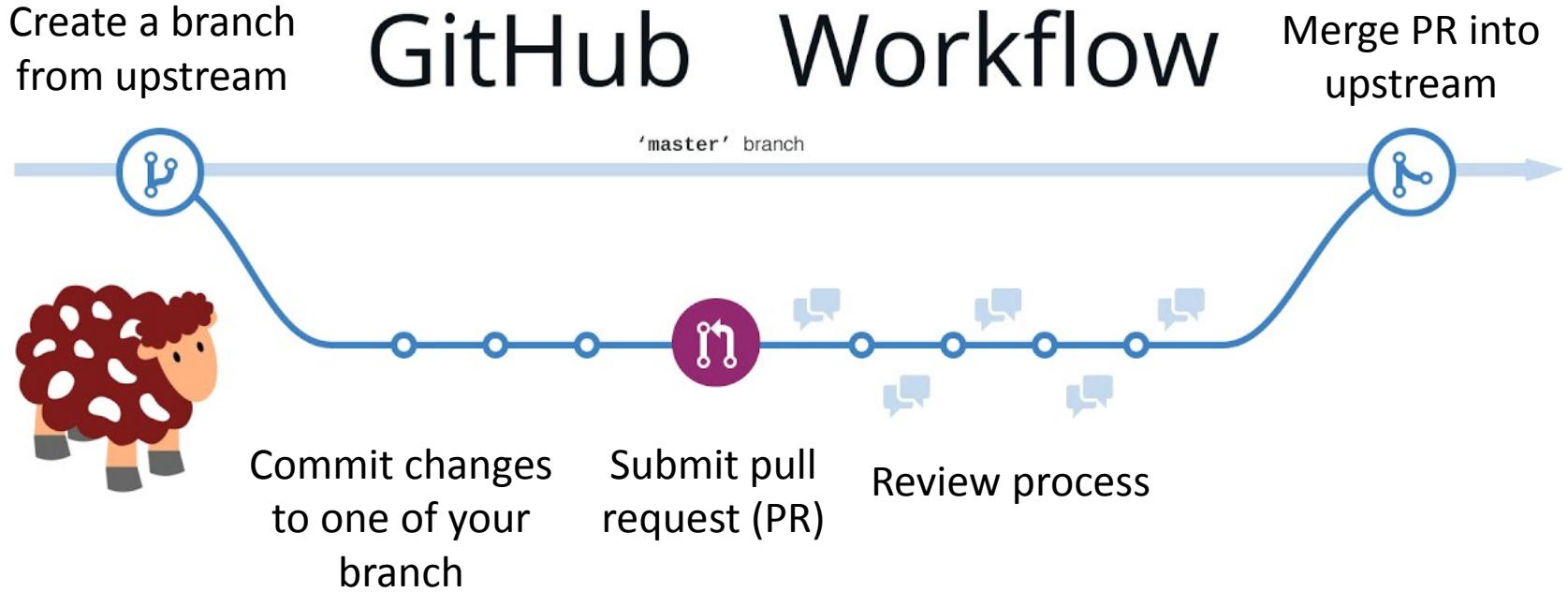


What is github?

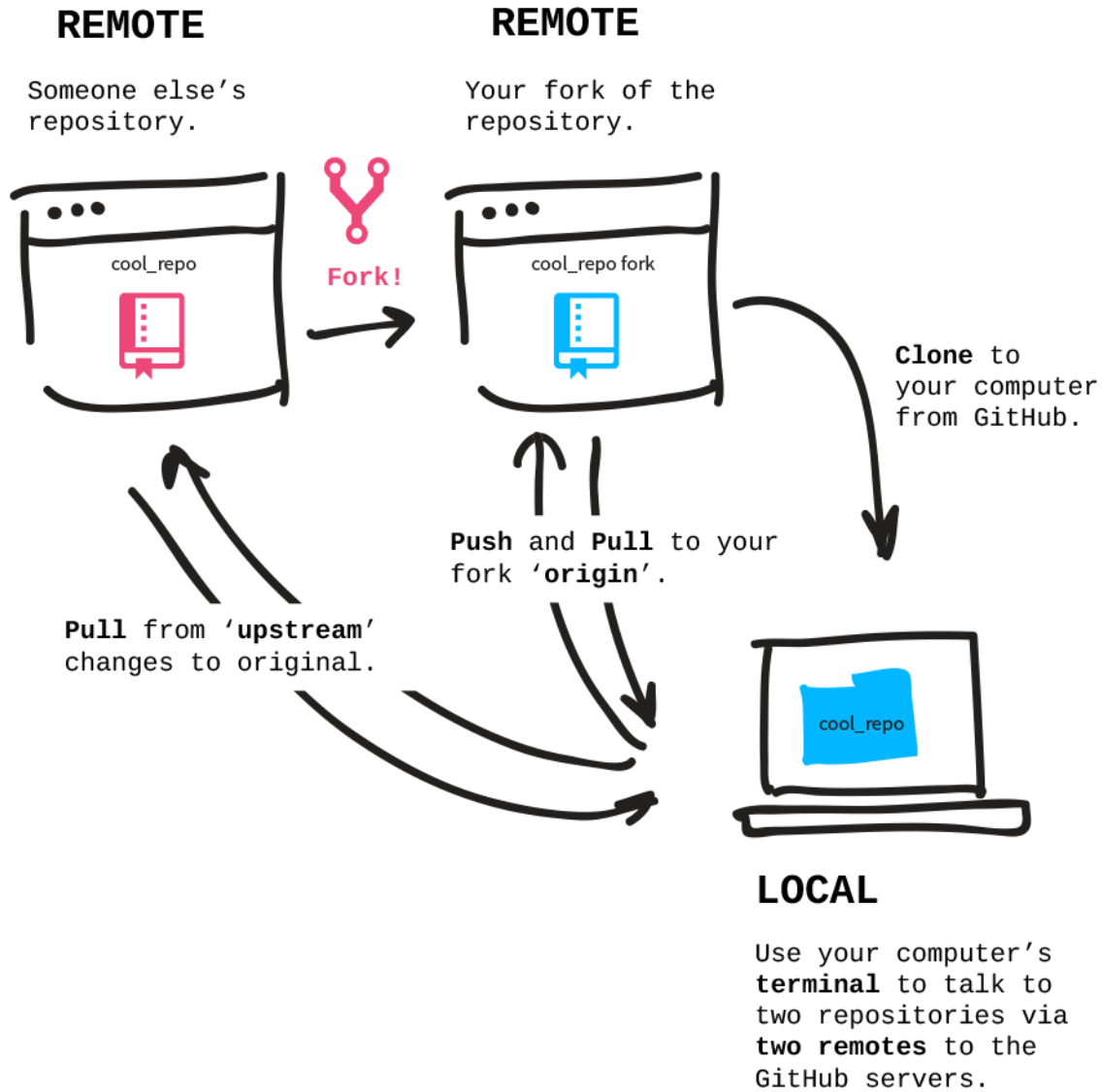
- GitHub is a Git repository hosting service with its own features
 - Web-based graphical interface
 - provides a centralised place where people discuss the changes
 - “Social coding”
 - Open discussion in the issue tracker or PR submission, wiki, etc



How to use github?



How to use github?



Contribute to an open source project before git/github

1. manually download the project's source code
2. make your changes locally
3. create a list of changes called a 'patch' and then e-mail the patch to the project's maintainer
4. The maintainer would then have to evaluate this patch, possibly sent by a total stranger and decide whether to merge the changes.

Tedious and not manageable

Why is github so useful/popular?

github web interface makes things easier

- Share code with other developers and the public
- Manage issues
- Manage pull request (code comparison and review)
- Maintain code


The screenshot shows a GitHub pull request (PR) titled "Fix tests and tidy up CI #2065". The PR is merged and was created by francisco-dlp on 18 Oct. It includes 12 commits and 10 files changed. The PR description details the changes made to fix Travis and Appveyor failures, including updates to matplotlib, numba, and sparse dependencies, and the installation of dependencies from conda-forge. The PR is reviewed by sem-geologist on 10 Oct. A code diff is shown for the file `hyperspy/misc/io/fei_stream_readers.py`, highlighting the removal of a try block and the addition of `import sparse`. The PR is labeled with "status: needs review", "type: bug-fix", "release: next patch", and "affects: tests".

Why is github so useful/popular?


github web interface makes things easier

- Review and discuss changes during pull request (PR) review


<pre> 41 - DEPS: "numpy scipy matplotlib ipython h5py sympy scikit-learn dill setuptools natsort scikit-image cython ipyparallel dask numexpr" 42 43 44 @@ -60,15 +60,11 @@ install: 60 61 # Install the dependencies of the project. 62 - ps: Add-AppveyorMessage "Installing conda packages..." 63 - - "%CMD_IN_ENV% conda install -yq %TEST_DEPS%" </pre>	<pre> 41 + DEPS: "numpy scipy matplotlib=2.2.3 ipython h5py sympy scikit-learn dill setuptools natsort scikit-image cython ipyparallel dask numexpr sparse numba" 42 43 44 @@ -60,15 +60,11 @@ install: 60 61 # Install the dependencies of the project. 62 - ps: Add-AppveyorMessage "Installing conda packages..." 63 + - "conda install -yq -c conda-forge %TEST_DEPS%" </pre>
--	---

 **francisco-dlp** on 18 Oct Member


Why should we use conda-forge to install all the deps? In a standard installation most people will have downloaded the full anaconda (vs miniconda here), hence most of the dependencies would be already installed and coming from the standard anaconda repos.

 **ericpre** on 18 Oct Member


This was part of removing the "test" optional dependency installation, so this should be removed now but pytest-cov still need to be installed.

 **francisco-dlp** on 18 Oct Member

My point was more about why conda-forge instead of the standard anaconda repository.

 **ericpre** on 18 Oct Member

pytest-mpl is not in defaults.

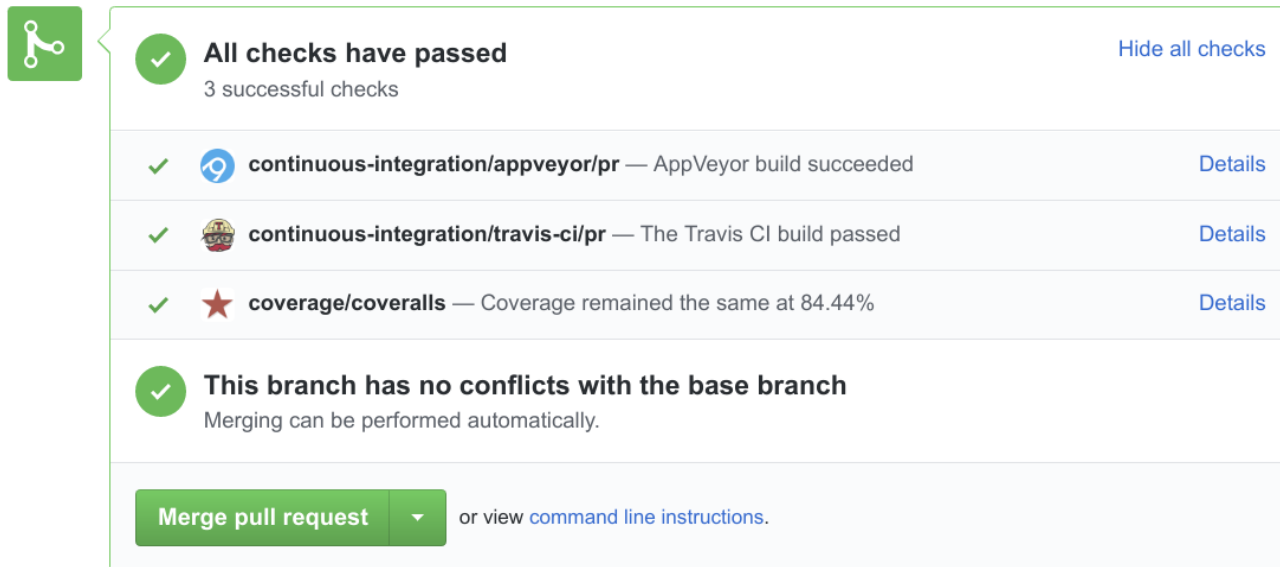
 **francisco-dlp** on 18 Oct Member

OK




[Resolve conversation](#)

Why is github so useful/popular?

- Continuous integration
 - For each PR, the code is tested automatically against a suite of tests using external services (travis, appveyor, etc.)
 - ~2270 *unit tests* checked that no regression is introduced by new changes

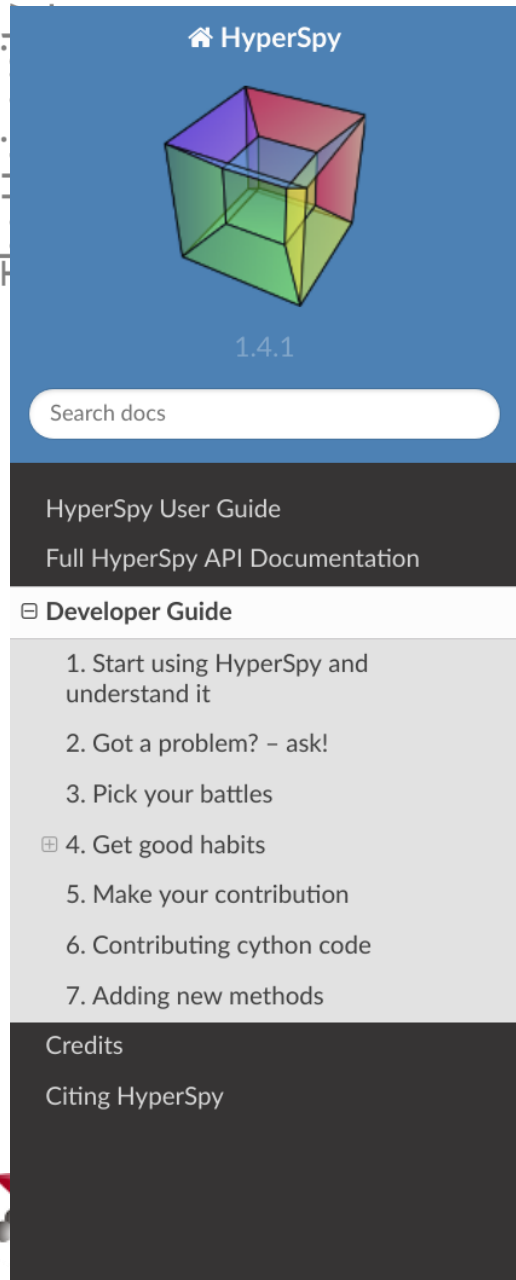


The screenshot shows a GitHub pull request status summary. At the top left is a green square icon with a white branching diagram. To its right is a green checkmark icon, followed by the text "All checks have passed" and "3 successful checks". A link "Hide all checks" is on the right. Below this are four rows of check status:

✓	 continuous-integration/appveyor/pr — AppVeyor build succeeded	Details
✓	 continuous-integration/travis-ci/pr — The Travis CI build passed	Details
✓	 coverage/coveralls — Coverage remained the same at 84.44%	Details
✓	This branch has no conflicts with the base branch Merging can be performed automatically.	

At the bottom, there is a green button "Merge pull request" with a dropdown arrow, followed by the text "or view [command line instructions](#)."

HyperSpy developer guide



HyperSpy

1.4.1

Search docs

HyperSpy User Guide

Full HyperSpy API Documentation

☰ Developer Guide

1. Start using HyperSpy and understand it
2. Got a problem? – ask!
3. Pick your battles
- ☰ 4. Get good habits
5. Make your contribution
6. Contributing cython code
7. Adding new methods

Credits

Citing HyperSpy

[Docs](#) » Developer Guide

[View page source](#)

Developer Guide

This 6-step guide is intended to give people who want to start contributing their own tools to HyperSpy a foothold to kick-start the process. This is also the way to start if you ultimately hope to become a member of the developer team.

We anticipate that many potential contributors and developers will be scientists who may have a lot to offer in terms of expert knowledge but may have little experience when it comes to working on a reasonably large open-source project like HyperSpy. This guide is aimed at you - helping to reduce the barrier to make a contribution.

Before you start you should decide which platform (i.e. Linux, Windows, or Mac) you are going to work in. All are possible and the advice below is the same it's only the specifics that change.

1. Start using HyperSpy and understand it

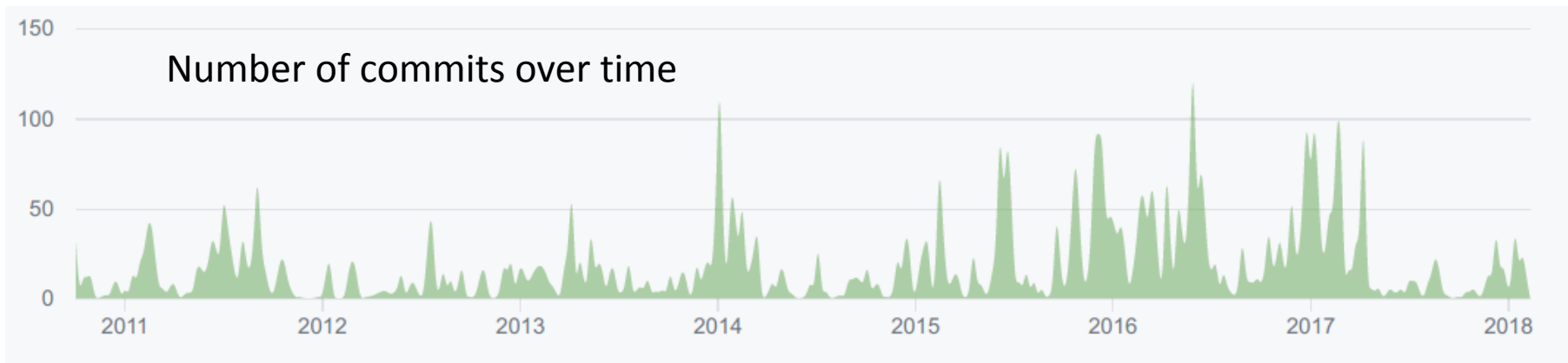
The best way to start understanding how HyperSpy works and to build a broad overview of the code as it stands is to use it – so what are you waiting for? [Download HyperSpy](#).

The user-guide also provides a good overview of all the parts of the code that are currently implemented as well as much information about how everything works – so read it well: [HyperSpy User-Guide](#).

For developing the code the home of HyperSpy is on github and you'll see that a lot of this guide boils down to using that platform well. so visit the following link and poke around the code, issues, and pull requests: [HyperSpy on Github](#).

HyperSpy community

- Code contributors
 - 37 contributors in total from many different labs
 - A few contributors change jobs
 - Their github profile may have been useful for their successful application
 - ~ 10 of them is a one off contribution
 - But most importantly, there are regular contributors

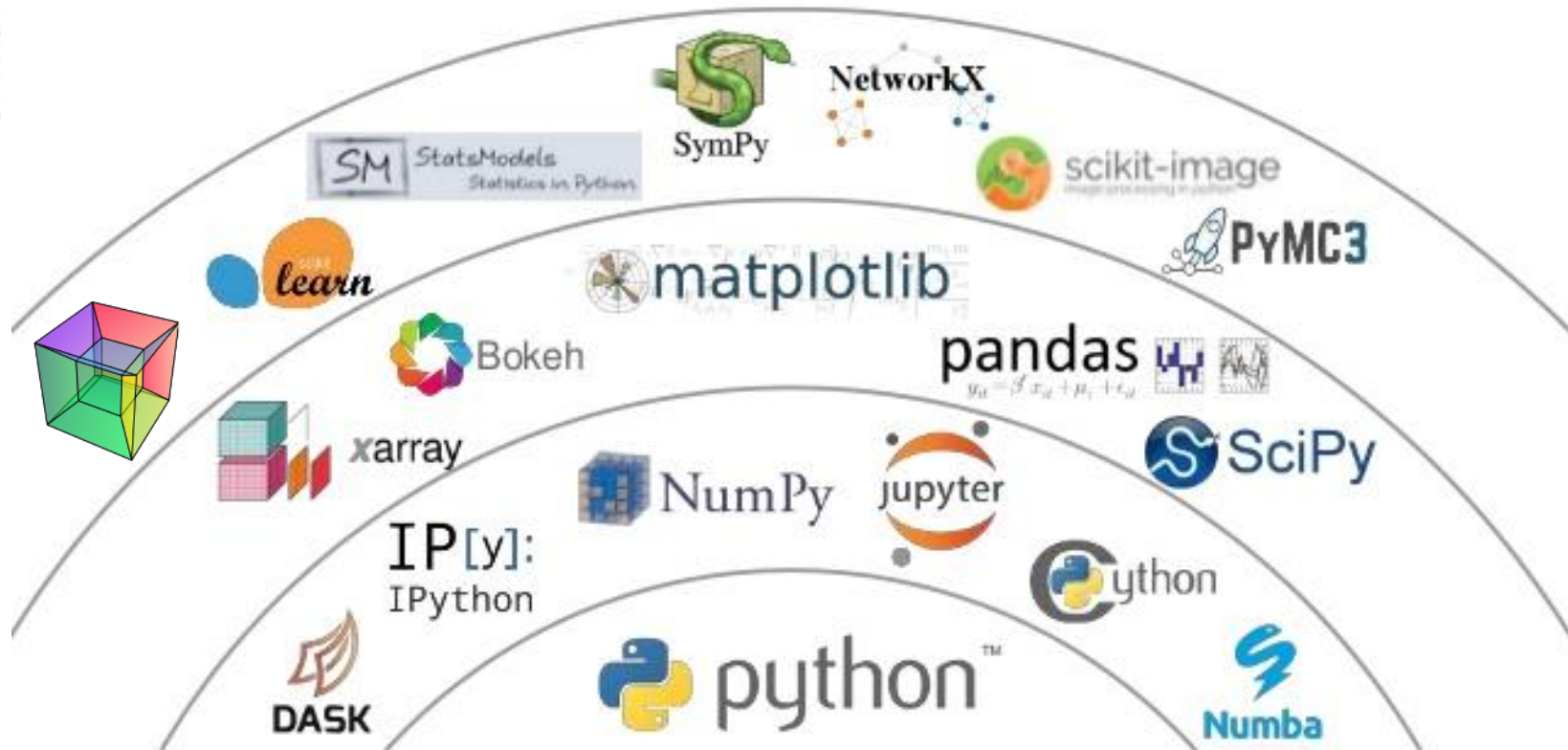


- Many more people following the gitter chat or the google user list contribute to the HyperSpy community
 - Bug report, feedback, complain, etc.

Python scientific stack: each library find its natural position

Python's Scientific Stack

Jake Vanderplas PyCon 2017 Keynote



- Strength of the major libraries for the user:
 - work nicely with each other
 - Similarities in the syntax

Current HyperSpy ecosystem

HyperSpyUI

HyperSpy-ipywidget

HyperSpy-traitsui

EELS holography

Big data

Multi-dimensional

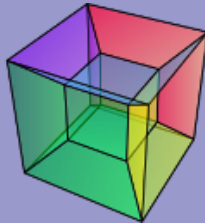
Visualisation and interactivity

Machine learning

Curve-fitting

EDS

IO



4D STEM Magnetic measurement

pixSTEM

Differential contrast imaging

Strain mapping Orientation mapping

Electron **pyXem**

cristallography Scanning Electron diffraction

Atomic position fitting

Atomap

Quantitative STEM

Strain mapping

Future HyperSpy ecosystem and other relevant libraries

EELS

IO

EDS

Holography

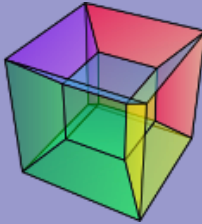
HyperSpyUI

HyperSpy-ipywidget

HyperSpy-traitsui

Multi-dimensional Visualisation and interactivity

Curve-fitting



Big data

Machine learning

Nion Swift

GMS 3

LiberTEM

tomviz

pycroscopy

4D STEM Magnetic measurement

pixSTEM

Differential contrast imaging

Strain mapping Orientation mapping

pyXem

Electron cristallography Scanning Electron diffraction

Atomic position fitting

Atomap

Quantitative STEM Strain mapping

Other extension

Common issues with research software development

- Many publications present interesting data analysis workflow or methodological development
- These are generally not easily accessible
 - proprietary licence
 - not easily to install or to run (platform dependent, etc.)
 - not supported/up to date anymore
- May be difficult to maintain for the developer
- Implementation need to be generic and robust
- Rely on a specific research group (most of the time by one or a few PhD student or a post-doc..)
 - Not sustainable

What makes HyperSpy today?

- Use tools and development practises which have proven to be successful for open source project
- HyperSpy is supported by its own community
- HyperSpy doesn't rely on a specific research group/institution
- Peer-reviewed and open-source development
- HyperSpy is a mature library
 - API fairly stable
 - doesn't break as much as before
- HyperSpy can be integrated easily in other software
 - offer a powerful platform for the development of other libraries
 - HyperSpy will be split in the near future
 - Easier, faster implementation of new features

Achievement of HyperSpy and its community

- HyperSpy managed to built a *distributed* community of users/contributor
 - Led by its own contributors
 - Decision based on contributors consensus
- Motivate (at least not discourage) users
 - Pay attention to users feedback
 - From user to contributor: make the learning curve easier
- What are contributors doing?
 - Contribution to user guide, tutorials and online discussion
 - Code writing and/or code review

This is one way to make a library
useful and sustainable

As individual is it worth contributing to HyperSpy?

- HyperSpy acknowledgement through zenodo DOI
 - One DOI for each release (important for reproducibility)
 - New contributors will get acknowledged
- Very good training
- Contribution to HyperSpy (or any other library) can be useful for career development
 - Github profil can be used as linkedin, etc.
 - Recognition by the community

As a group/organisation/company, why supporting HyperSpy?

- As PI/group leader
 - PI are not acknowledged but they can benefit a lot from the expertise gained through HyperSpy
 - Fairly useful “training” for post-doc/student
- Companies, conflict of interest?!
 - Some companies start to show interest in open source
 - Still a bit difficult: paradigm shift required
 - Customers are pushing enough to convince companies
 - Extend data processing capabilities of their software
 - By being compatible with open source software, companies can offer solution there could not afford otherwise