UK Atomic Energy Authority

HyperSpy Community & Github

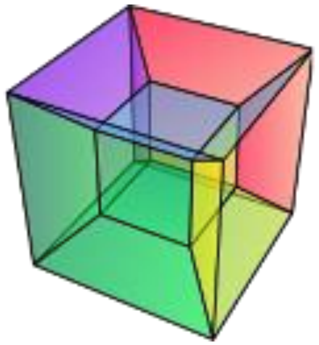
Eric Prestat

This work was part funded by the RCUK Energy Programme [grant number EP/T012250/1]. This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014-2018 and 2019-2020 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission.

CCFE CULHAM CENTRE FOR FUSION ENERGY

Outline

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

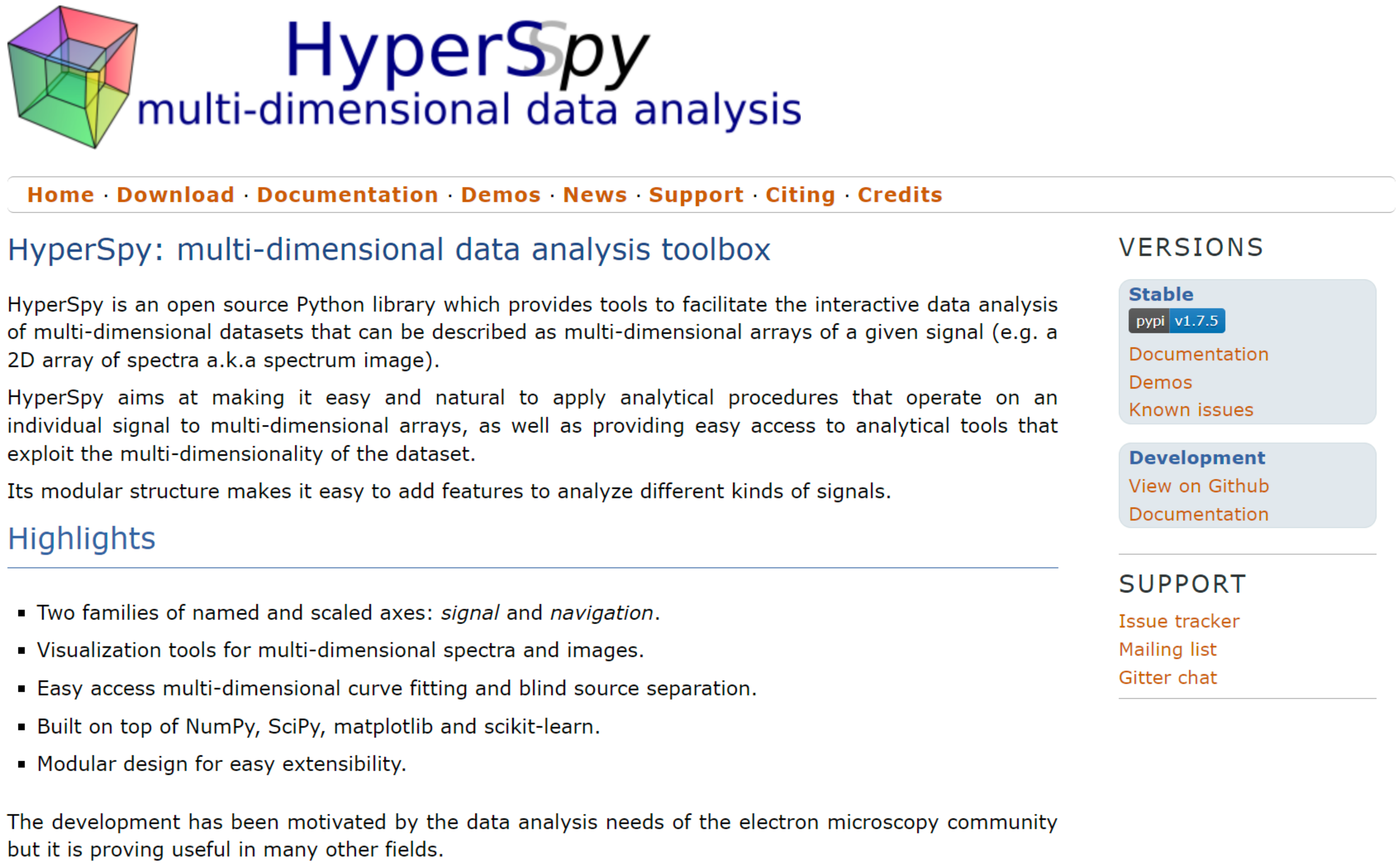


'Enhanced Data Generated by Electrons'

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 source code, but also about:
  - Accessibility to the users
  - Sustainable library development
  - Building a community



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.

The development has been motivated by the data analysis needs of the electron microscopy community but it is proving useful in many other fields.

VERSIONS

Stable
pypi v1.7.5
Documentation
Demos
Known issues

Development
View on Github
Documentation

SUPPORT

Issue tracker
Mailing list
Gitter chat

https://hyperspy.org

HyperSpy Documentation / Community Support

- Main website:
  - https://hyperspy.org/
- Documentation:
  - https://hyperspy.org/hyperspy-doc/current/index.html
- Gitter: online chat room
  - https://gitter.im/hyperspy/hyperspy
- Users list:
  - https://groups.google.com/forum/#!forum/hyperspy-users
  - Not very popular/used much anymore, using gitter is favourable

HyperSpy demos

- Source code of the demo on github
  - https://github.com/hyperspy/hyperspy-demos
- Non-interactive version on nbviewer
  - https://nbviewer.jupyter.org/github/hyperspy/hyperspy-demos/tree/master/
  - Github now displays nicely the notebook, but works well for short notebook (loading fail otherwise)
- Interactive version on mybinder.org
  - ~~https://mybinder.org/v2/gh/hyperspy/hyperspy-demos/master~~
  - ~~Run demos online (on a remote server) without any installation~~
  - ~~1 CPU core, between 1 and 2 GB available memory~~
  - ~~Shut down after 10 min inactivity~~

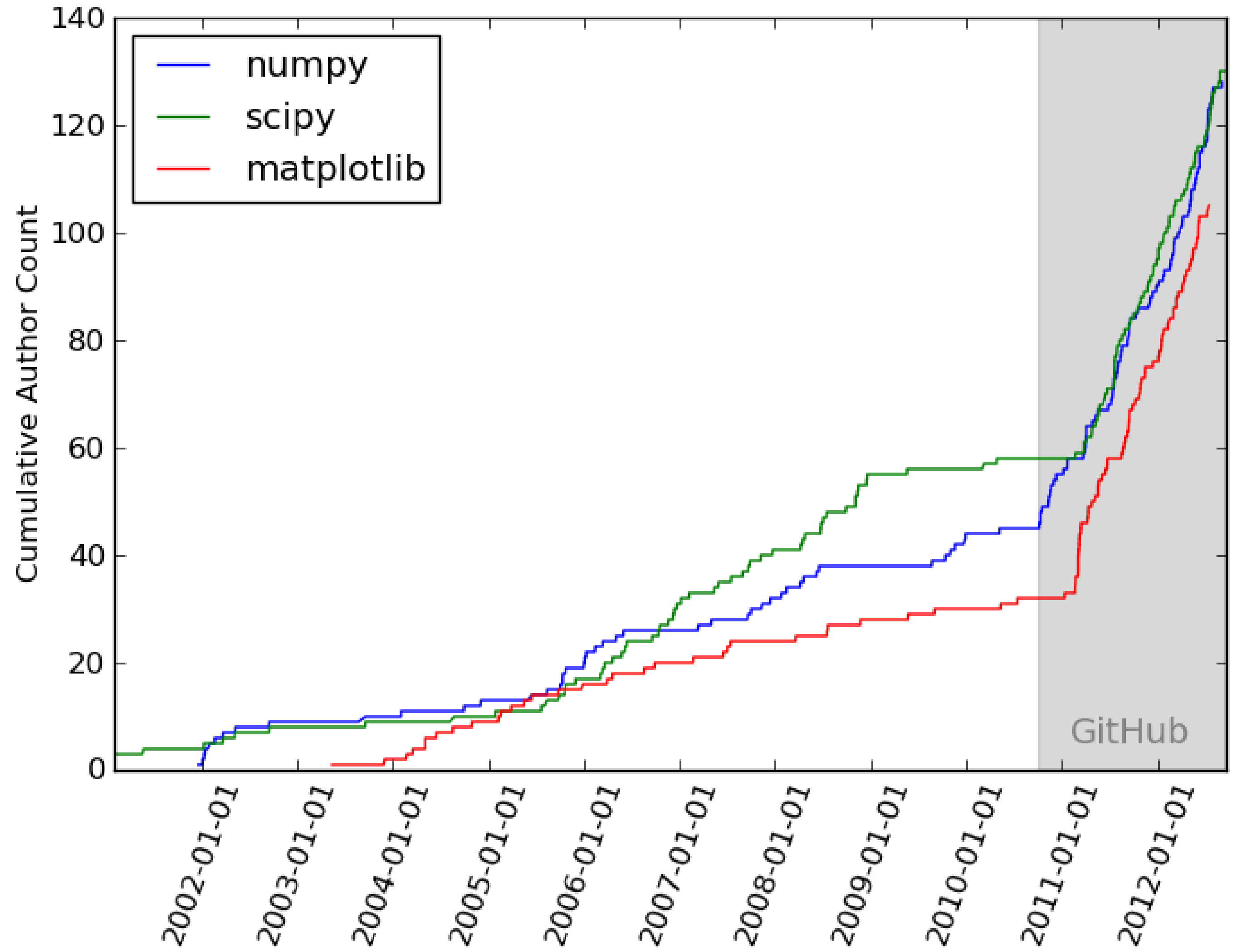
Since last month (April 2023), not working anymore because of lack of cloud resources, after Google sponsoring stops*

* https://blog.jupyter.org/mybinder-org-reducing-capacity-c93ccfc6413f

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

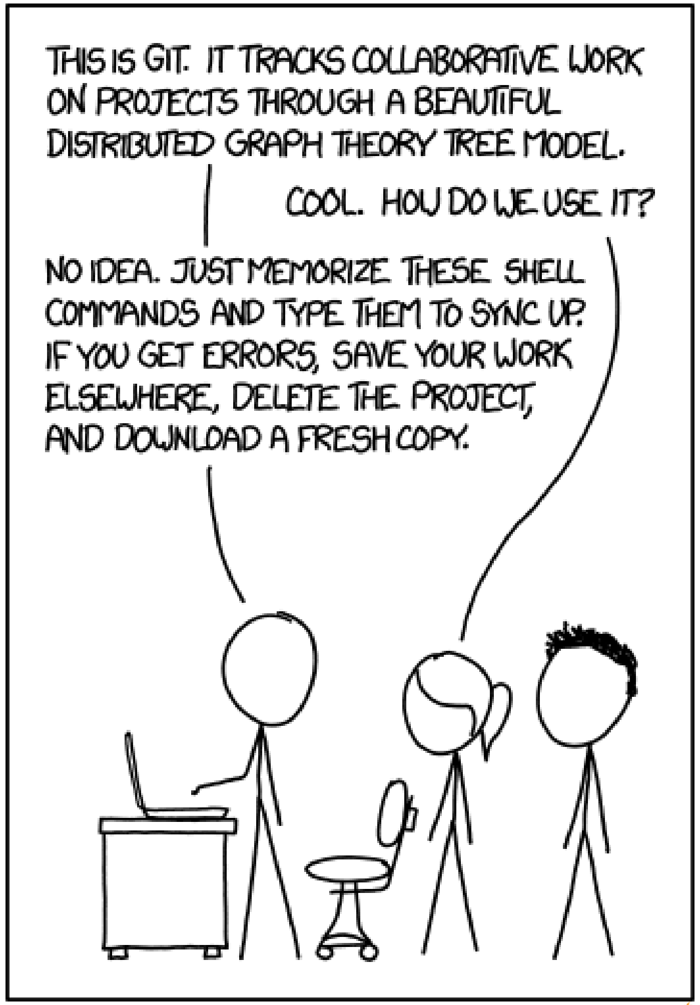
The GitHub Effect



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

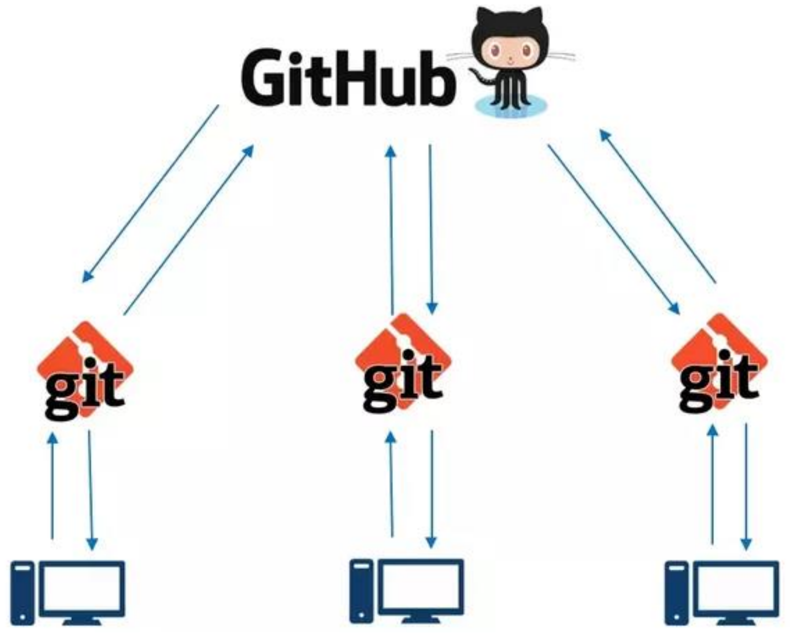
- At the heart of GitHub is git
  - A version control system: manages and stores revisions of projects
  - git is a distributed version control systems
    - Each contributor has is own *remote* (online) repository
    - Code is merged in the *upstream* repository


THIS IS GIT. IT TRACKS COLLABORATIVE WORK ON PROJECTS THROUGH A BEAUTIFUL DISTRIBUTED GRAPH THEORY TREE MODEL.
COOL. HOW DO WE USE IT?
NO IDEA. JUST MEMORIZE THESE SHELL COMMANDS AND TYPE THEM TO SYNC UP. IF YOU GET ERRORS, SAVE YOUR WORK ELSEWHERE, DELETE THE PROJECT, AND DOWNLOAD A FRESH COPY.

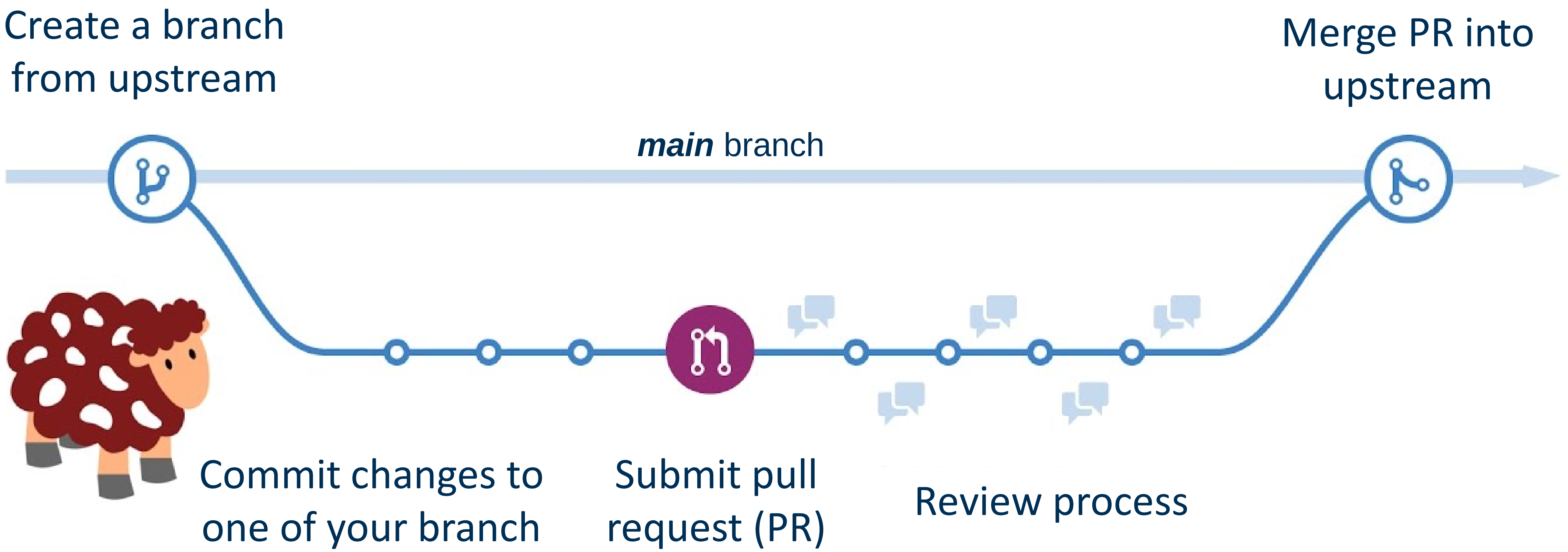
Short video introducing GitHub: https://www.youtube.com/watch?v=w3jLJU7DT5E

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
    - Open “pull request” to propose changes

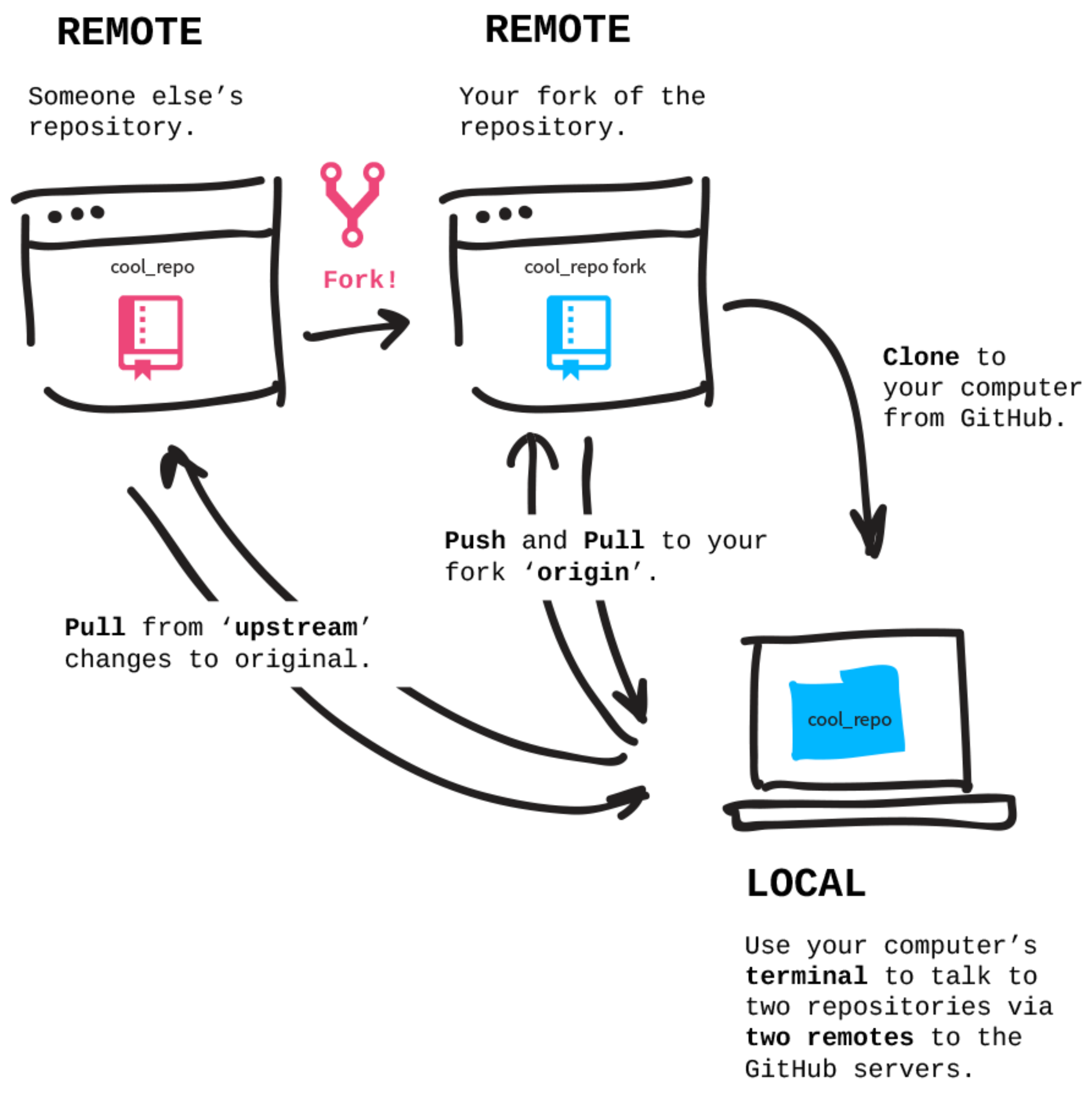


GitHub Workflow



GitHub Workflow

git is a distributed version control systems


REMOTE
Someone else's repository.
cool_repo
Fork!
REMOTE
Your fork of the repository.
cool_repo fork
Clone to your computer from GitHub.
Push and Pull to your fork 'origin'.
Pull from 'upstream' changes to original.
cool_repo
LOCAL
Use your computer's terminal to talk to two repositories via two remotes to the GitHub servers.

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
    - Patches are not easily to read…
4. The maintainer would then have to evaluate this patch, possibly sent by a total stranger and decide whether to merge the changes.

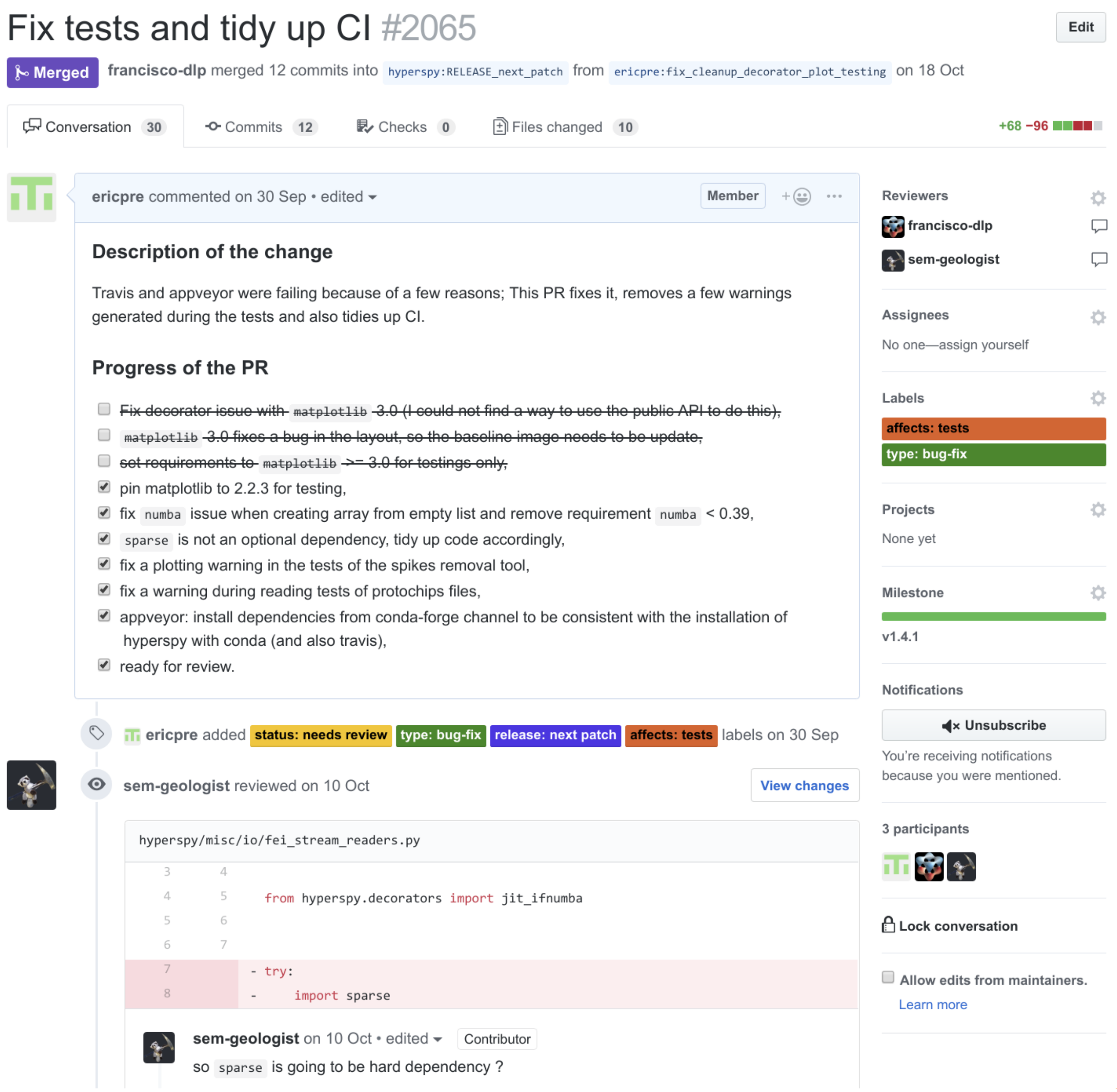
Tedious and not manageable

Why is GitHub so useful / popular?

UK Atomic Energy Authority

GitHub platform makes collaboration easier

- Share code with other developers
- Manage issues
- Manage pull request
  - Code comparison
  - Review
- Maintain code
  - Continuous integration
  - Third-party integration



Fix tests and tidy up CI #2065

Edit

Merged **francisco-dlp** merged 12 commits into `hyperspy:RELEASE_next_patch` from `ericpre:fix_cleanup_decorator_plot_testing` on 18 Oct

Conversation 30 | Commits 12 | Checks 0 | Files changed 10 | +68 −96

ericpre commented on 30 Sep • edited — Member

Description of the change

Travis and appveyor were failing because of a few reasons; This PR fixes it, removes a few warnings generated during the tests and also tidies up CI.

Progress of the PR

- [ ] ~~Fix decorator issue with `matplotlib` 3.0 (I could not find a way to use the public API to do this),~~
- [ ] ~~`matplotlib` 3.0 fixes a bug in the layout, so the baseline image needs to be update,~~
- [ ] ~~set requirements to `matplotlib` >= 3.0 for testings only,~~
- [x] pin matplotlib to 2.2.3 for testing,
- [x] fix `numba` issue when creating array from empty list and remove requirement `numba` < 0.39,
- [x] `sparse` is not an optional dependency, tidy up code accordingly,
- [x] fix a plotting warning in the tests of the spikes removal tool,
- [x] fix a warning during reading tests of protochips files,
- [x] appveyor: install dependencies from conda-forge channel to be consistent with the installation of hyperspy with conda (and also travis),
- [x] ready for review.

ericpre added `status: needs review` `type: bug-fix` `release: next patch` `affects: tests` labels on 30 Sep

sem-geologist reviewed on 10 Oct — View changes

hyperspy/misc/io/fei_stream_readers.py

```
     from hyperspy.decorators import jit_ifnumba


 - try:
 -     import sparse
```

sem-geologist on 10 Oct • edited — Contributor

so `sparse` is going to be hard dependency ?

Reviewers: francisco-dlp, sem-geologist

Assignees: No one—assign yourself

Labels: affects: tests, type: bug-fix

Projects: None yet

Milestone: v1.4.1

Notifications: Unsubscribe — You're receiving notifications because you were mentioned.

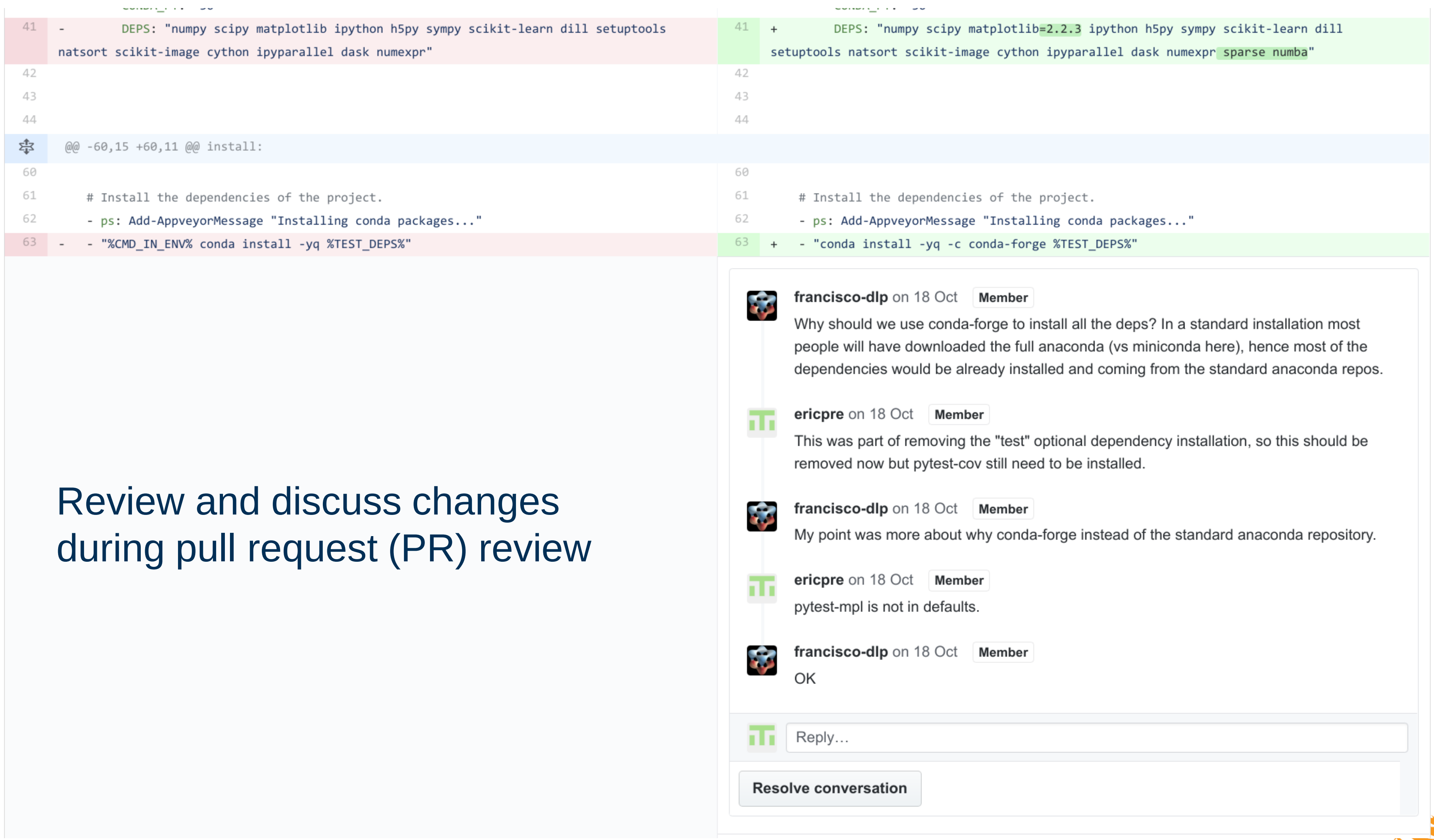
3 participants

Lock conversation

Allow edits from maintainers. Learn more

CCFE CULHAM CENTRE FOR FUSION ENERGY

Why is GitHub so useful / popular?



```diff
-        DEPS: "numpy scipy matplotlib ipython h5py sympy scikit-learn dill setuptools natsort scikit-image cython ipyparallel dask numexpr"
+        DEPS: "numpy scipy matplotlib=2.2.3 ipython h5py sympy scikit-learn dill setuptools natsort scikit-image cython ipyparallel dask numexpr sparse numba"

@@ -60,15 +60,11 @@ install:

    # Install the dependencies of the project.
    - ps: Add-AppveyorMessage "Installing conda packages..."
-   - "%CMD_IN_ENV% conda install -yq %TEST_DEPS%"
+   - "conda install -yq -c conda-forge %TEST_DEPS%"
```

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

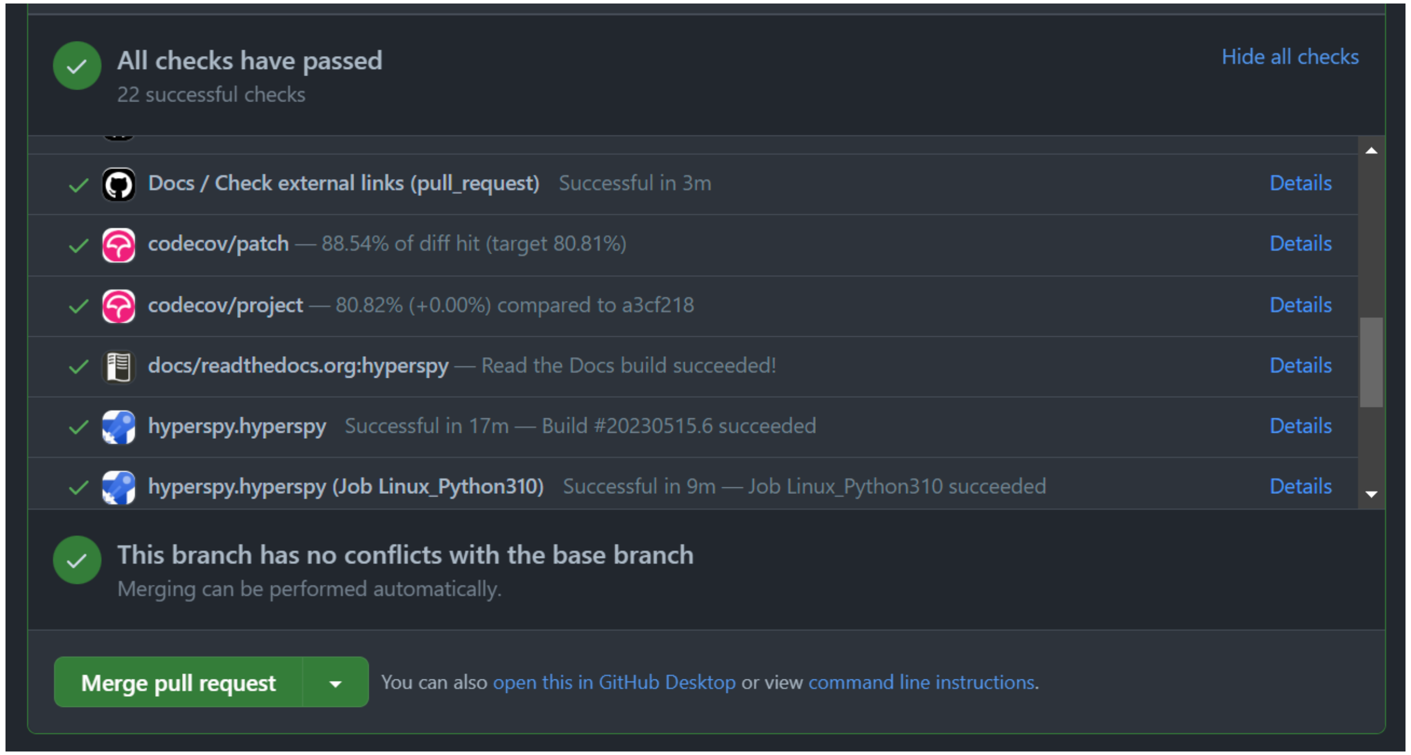
Reply…

Resolve conversation

Review and discuss changes during pull request (PR) review

Why is GitHub so useful / popular?

- Continuous integration
  - For each PR, the code is tested automatically against a suite of tests using external services (Github Actions, Azure Pipelines, documentation links, code formatting, code coverage, etc.)
  - ~5386 *unit tests* continuously checking that no regression is introduced by new changes



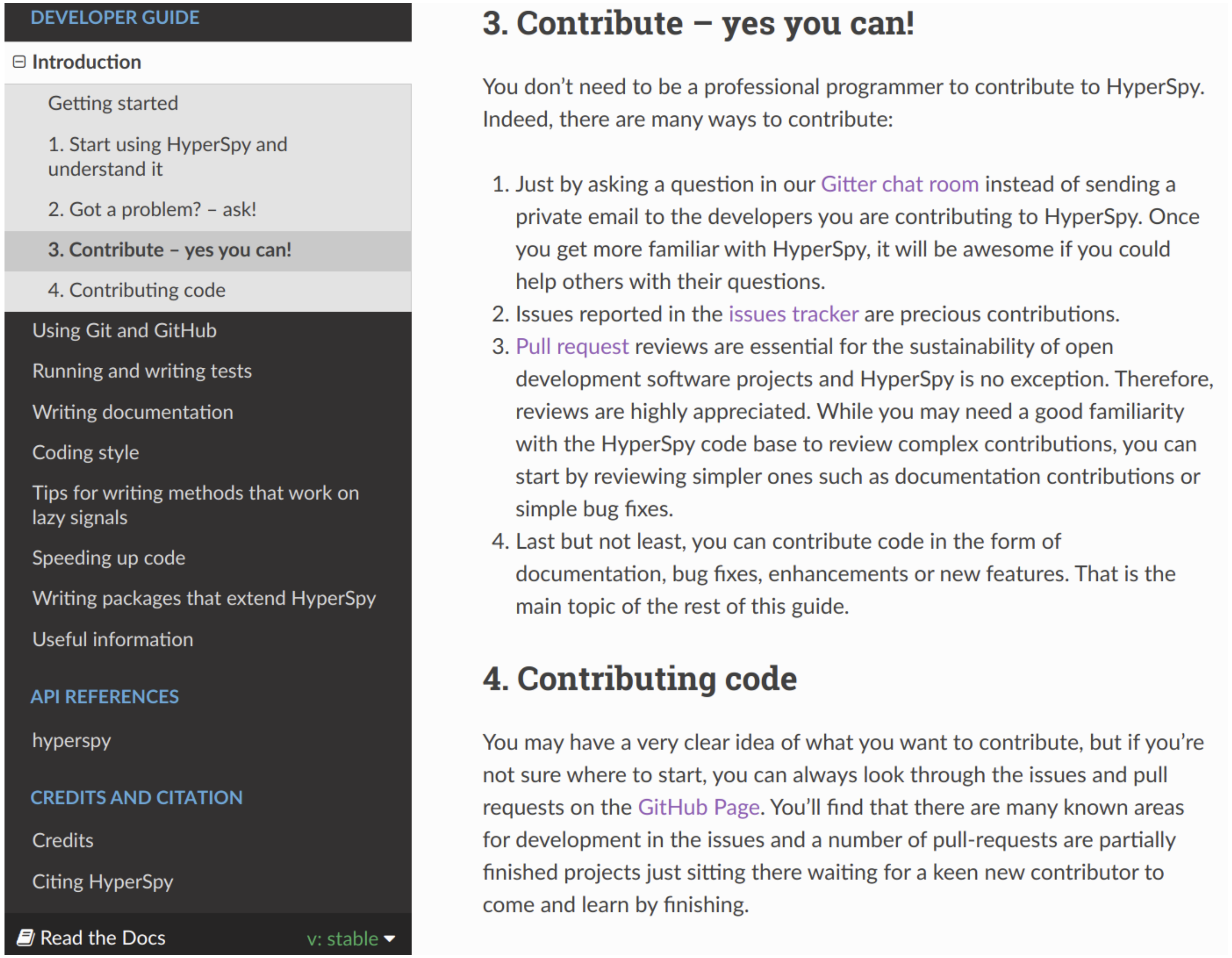
All checks have passed
22 successful checks
Hide all checks

✓ Docs / Check external links (pull_request) Successful in 3m Details
✓ codecov/patch — 88.54% of diff hit (target 80.81%) Details
✓ codecov/project — 80.82% (+0.00%) compared to a3cf218 Details
✓ docs/readthedocs.org:hyperspy — Read the Docs build succeeded! Details
✓ hyperspy.hyperspy Successful in 17m — Build #20230515.6 succeeded Details
✓ hyperspy.hyperspy (Job Linux_Python310) Successful in 9m — Job Linux_Python310 succeeded Details

This branch has no conflicts with the base branch
Merging can be performed automatically.

Merge pull request ▾ You can also open this in GitHub Desktop or view command line instructions.

Contributing


DEVELOPER GUIDE

⊟ Introduction

Getting started

1. Start using HyperSpy and understand it

2. Got a problem? – ask!

3. Contribute – yes you can!

4. Contributing code

Using Git and GitHub

Running and writing tests

Writing documentation

Coding style

Tips for writing methods that work on lazy signals

Speeding up code

Writing packages that extend HyperSpy

Useful information

API REFERENCES

hyperspy

CREDITS AND CITATION

Credits

Citing HyperSpy

Read the Docs v: stable

3. Contribute – yes you can!

You don't need to be a professional programmer to contribute to HyperSpy. Indeed, there are many ways to contribute:

1. Just by asking a question in our Gitter chat room instead of sending a private email to the developers you are contributing to HyperSpy. Once you get more familiar with HyperSpy, it will be awesome if you could help others with their questions.
2. Issues reported in the issues tracker are precious contributions.
3. Pull request reviews are essential for the sustainability of open development software projects and HyperSpy is no exception. Therefore, reviews are highly appreciated. While you may need a good familiarity with the HyperSpy code base to review complex contributions, you can start by reviewing simpler ones such as documentation contributions or simple bug fixes.
4. Last but not least, you can contribute code in the form of documentation, bug fixes, enhancements or new features. That is the main topic of the rest of this guide.

4. Contributing code

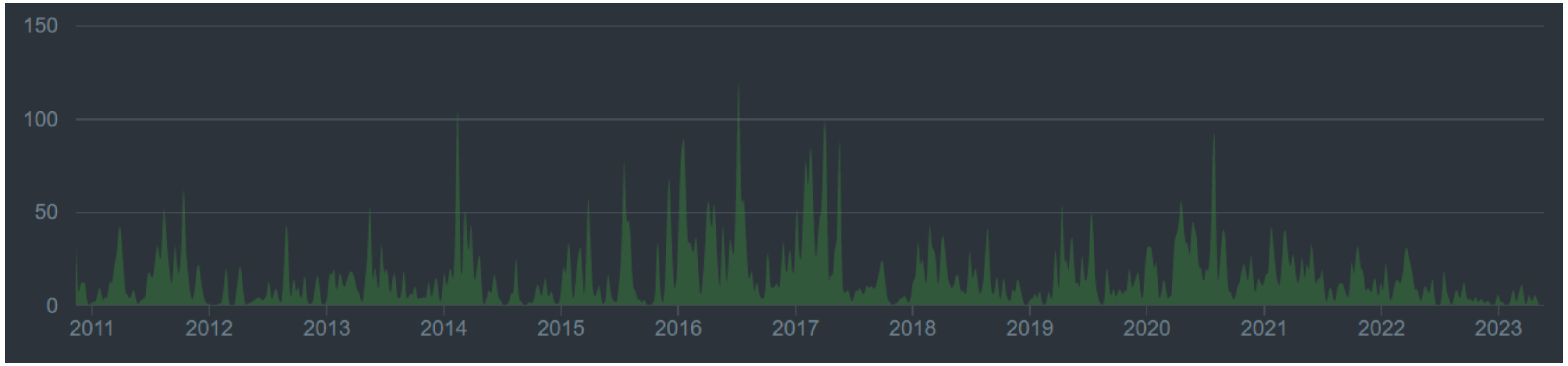
You may have a very clear idea of what you want to contribute, but if you're not sure where to start, you can always look through the issues and pull requests on the GitHub Page. You'll find that there are many known areas for development in the issues and a number of pull-requests are partially finished projects just sitting there waiting for a keen new contributor to come and learn by finishing.

https://hyperspy.readthedocs.io/en/stable/dev_guide/intro.html

HyperSpy Community

- Code contributors
  - ~62 contributors in total from many different labs
  - A few contributors change jobs
    - Their github profile may have been useful for their successful application
  - ~ 20 of them are a one off contribution
- Many more people following the gitter chat
  - Bug report, user feedback, feature requests, etc.

Number of commits over time



The HyperSpy Eco-System as of 2020

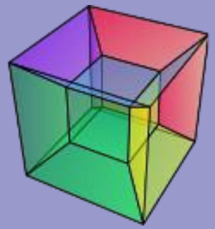
- Extension registration: other libraries can create their domain specific signal class and core functionalities of HyperSpy will be able to create instance of these class when necessary

HyperSpy-ipywidget

HyperSpy-traitsui

HyperSpyUI

Multi-dimensional | Big data | IO | Curve-fitting | EDS | EELS | holography | Visualisation and interactivity | Machine learning



pixStem: 4D STEM, Magnetic measurement, Differential contrast imaging

pyXem: Strain mapping, Orientation mapping, Electron cristallography, Scanning Electron diffraction

LumiSpy: cathodoluminescence, photoluminescence

Kikuchipy: EBSD, Strain mapping, Orientation mapping

Atomap: Atomic position fitting, Quantitative STEM, Strain mapping

EMpyer: Electron Correlation Microscopy, Angular Correlations, Fluctuation Electron Microscopy

inpystem: inpainting, reconstruction techniques

ParticleSpy: segmentation, Particles analysis

hyperspy_swift_library: Access nionswift libraries

The HyperSpy Eco-System as of 2021

UK Atomic Energy Authority

- Extension registration: other libraries can create their domain specific signal class and core functionalities of HyperSpy will be able to create instance of these class when necessary

HyperSpy-ipywidget

HyperSpy-traitsui

HyperSpyUI

Multi-dimensional

Big data

IO

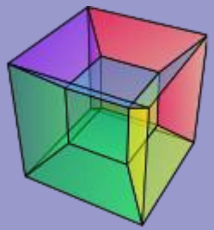
Curve-fitting

EDS EELS

holography

Visualisation and interactivity

Machine learning



EMpyer pixStem

pyXem

Strain mapping

Orientation mapping

Electron cristallography

Scanning Electron diffraction

Differential contrast imaging

Magnetic measurement

LumiSpy
cathodoluminescence
photoluminescence

Kikuchipy
EBSD
Strain mapping
Orientation mapping

Atomap
Atomic position fitting
Quantitative STEM
Strain mapping

pyMatchSeries
Non-rigid image registration

inpystem
inpainting
reconstruction techniques

ParticleSpy
segmentation
Particles analysis

hyperspy_swift_library
Access nionswift libraries

pgure-svt
Denoising time-resolved microscopy

CCFE CULHAM CENTRE FOR FUSION ENERGY

HyperSpy 2.0: Coming Soon

UK Atomic Energy Authority

- Extension registration: other libraries can create their domain specific signal class and core functionalities of HyperSpy will be able to create instance of these class when necessary

HyperSpy-ipywidget

HyperSpy-traitsui

HyperSpyUI

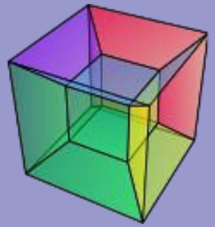
Multi-dimensional

Machine learning

Visualisation and interactivity

Curve-fitting

Big data



RosettaSciIO

EELS

EDS

Holography

EMpyer

pixStem

pyXem

Strain mapping

Orientation mapping

Electron cristallography

Scanning Electron diffraction

Differential contrast imaging

Magnetic measurement

cathodoluminescence
LumiSpy
photoluminescence

EBSD
Strain mapping
Kikuchipy
Orientation mapping

Atomic position fitting
Quantitative STEM
Atomap
Strain mapping

pyMatchSeries
Non-rigid image registration

inpainting
inpystem
reconstruction techniques

segmentation
ParticleSpy
Particles analysis

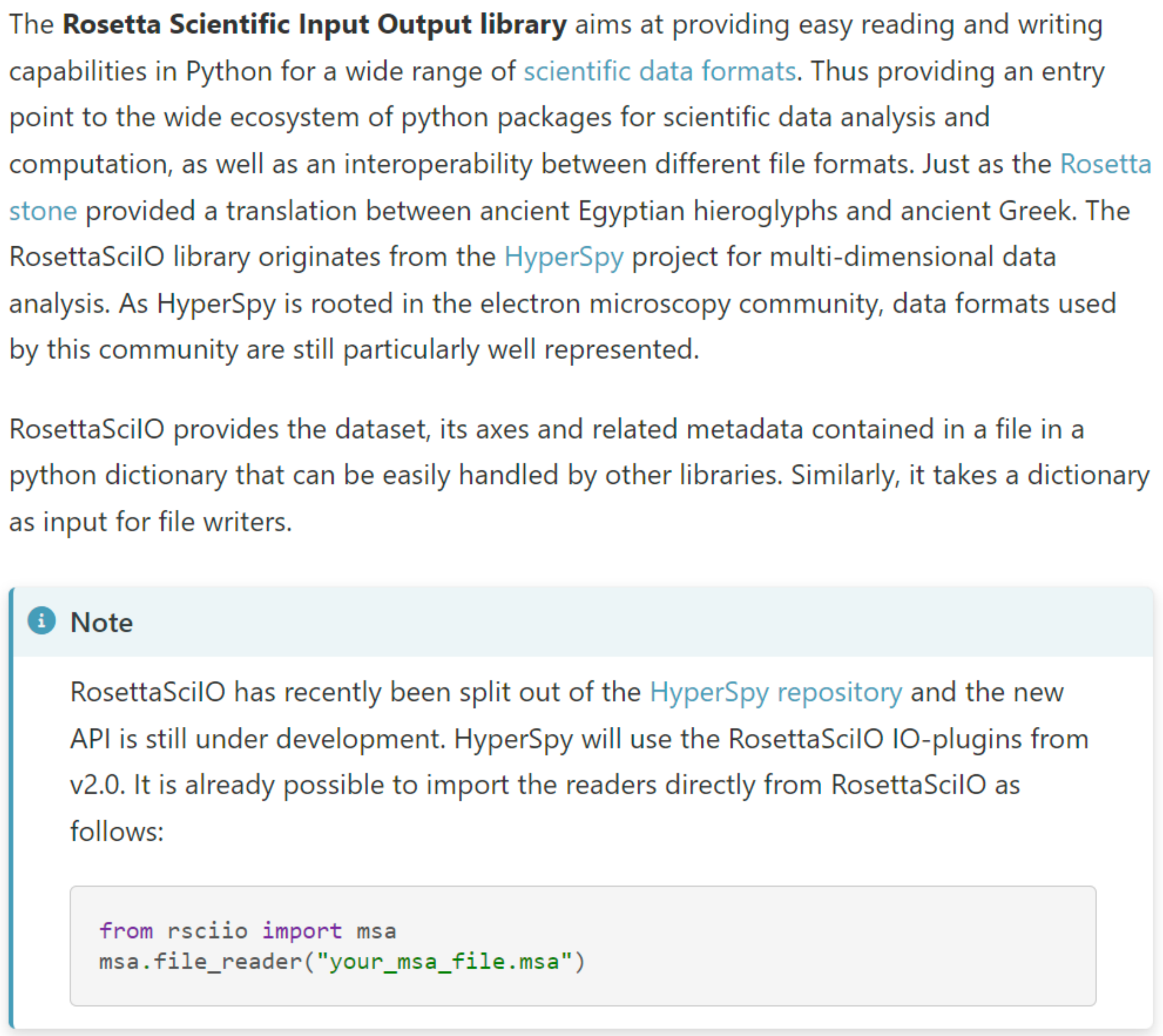
hyperspy_swift_library
Access nionswift libraries

pgure-svt
Denoising time-resolved microscopy

CCFE
CULHAM CENTRE FOR FUSION ENERGY

RosettaSciIO

- Target audience:
  - Developer of python libraries
  - No dependence on HyperSpy
- End user will continue to open/save data through HyperSpy as before
- Share IO code beyond the HyperSpy community
- API
  - Existing API stable
  - Experimental API to provide additional IO functionalities expected to be unstable
- Packaging being finalised


The **Rosetta Scientific Input Output library** aims at providing easy reading and writing capabilities in Python for a wide range of scientific data formats. Thus providing an entry point to the wide ecosystem of python packages for scientific data analysis and computation, as well as an interoperability between different file formats. Just as the Rosetta stone provided a translation between ancient Egyptian hieroglyphs and ancient Greek. The RosettaSciIO library originates from the HyperSpy project for multi-dimensional data analysis. As HyperSpy is rooted in the electron microscopy community, data formats used by this community are still particularly well represented.

RosettaSciIO provides the dataset, its axes and related metadata contained in a file in a python dictionary that can be easily handled by other libraries. Similarly, it takes a dictionary as input for file writers.

Note

RosettaSciIO has recently been split out of the HyperSpy repository and the new API is still under development. HyperSpy will use the RosettaSciIO IO-plugins from v2.0. It is already possible to import the readers directly from RosettaSciIO as follows:

```python
from rsciio import msa
msa.file_reader("your_msa_file.msa")
```

https://rosettasciio.readthedocs.io

Python libraries for 4D STEM data analysis

- pyXem
- pixstem
- EMpyer

Merged

- py4DSTEM
- pycroscopy
- stemtools
- Various nionswift plugins

All these open source libraries are doing very similarly things

- LiberTEM

Optimised for high throughput on small computer clusters

- …

What's Make HyperSpy Today

UK Atomic Energy Authority

- Eco-system provides state-of-the data analysis capabilities
- 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
  - Stable API: deprecation cycle
  - API break only on major release
- HyperSpy can be integrated easily in other software
  - Framework for the development of other libraries
  - Integration testing of the eco-system

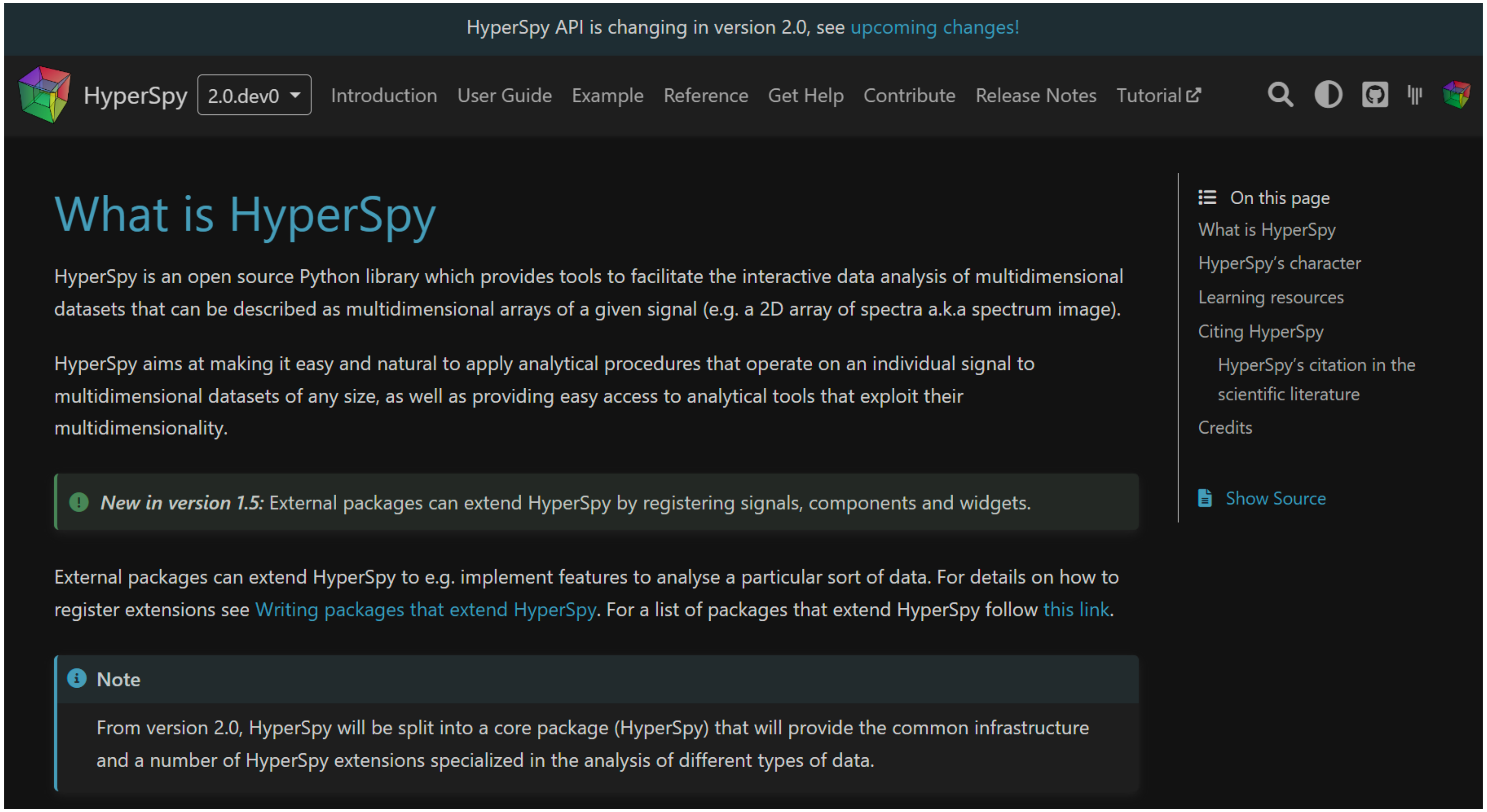
Community lead development

HyperSpy 2.0

- Domain specific functionalities split into extensions
- Improve modularity
- Independent development of extensions
- Easier, faster implementation of new features in extensions

CCFE CULHAM CENTRE FOR FUSION ENERGY

Documentation Improvement



HyperSpy API is changing in version 2.0, see upcoming changes!

What is HyperSpy

HyperSpy is an open source Python library which provides tools to facilitate the interactive data analysis of multidimensional datasets that can be described as multidimensional 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 multidimensional datasets of any size, as well as providing easy access to analytical tools that exploit their multidimensionality.

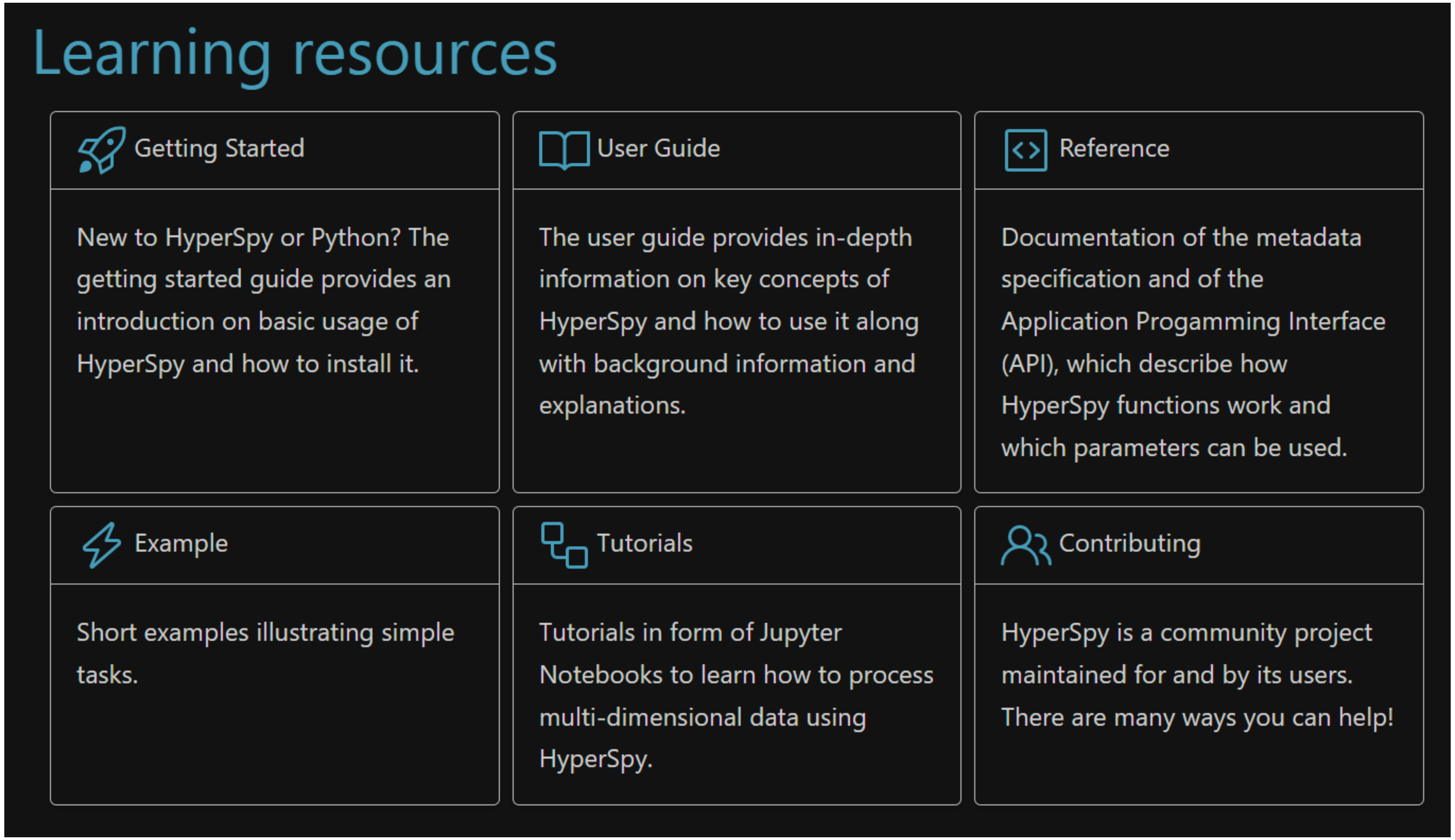
New in version 1.5: External packages can extend HyperSpy by registering signals, components and widgets.

External packages can extend HyperSpy to e.g. implement features to analyse a particular sort of data. For details on how to register extensions see Writing packages that extend HyperSpy. For a list of packages that extend HyperSpy follow this link.

Note

From version 2.0, HyperSpy will be split into a core package (HyperSpy) that will provide the common infrastructure and a number of HyperSpy extensions specialized in the analysis of different types of data.

Documentation Improvement



Learning resources

Getting Started

New to HyperSpy or Python? The getting started guide provides an introduction on basic usage of HyperSpy and how to install it.

User Guide

The user guide provides in-depth information on key concepts of HyperSpy and how to use it along with background information and explanations.

Reference

Documentation of the metadata specification and of the Application Progamming Interface (API), which describe how HyperSpy functions work and which parameters can be used.

Example

Short examples illustrating simple tasks.

Tutorials

Tutorials in form of Jupyter Notebooks to learn how to process multi-dimensional data using HyperSpy.

Contributing

HyperSpy is a community project maintained for and by its users. There are many ways you can help!

https://hyperspy.org/hyperspy-doc/dev/index.html

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

Change in how open-source tools is perceived:

- Users doesn’t need to be convinced to use open-source tools anymore
- Instrumentation suppliers engage with open source community

As an Individual is it Worth Contributing to HyperSpy?

- HyperSpy acknowledgement through zenodo DOI
  - One DOI for each release (important for reproducibility)
  - New contributors will be acknowledged
- Very good training
  - There is a lot to learn when contributing to open source project
- Contribution to HyperSpy (or any other library) can be useful for career development
  - Github profile 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
  - Remove dependence on expansive license for software or plugins
  - Improve reproducibility
- 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