

Gammapy

Christoph Deil, MPIK, Heidelberg
for the Gammapy developers



October 2, 2017, Ateliér CTA, Observatoire de Meudon

Overview



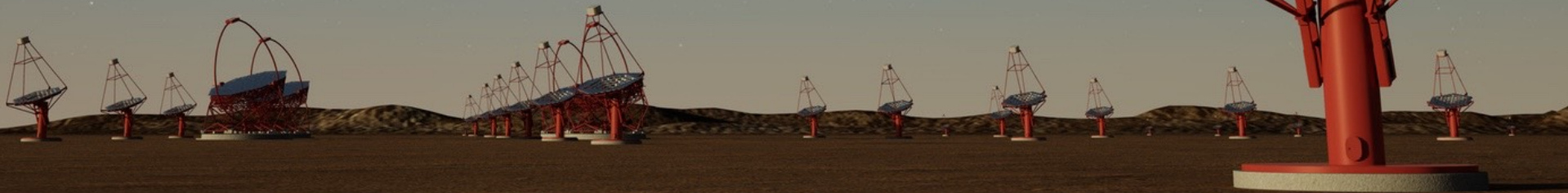
- What is Gammapy?
- Context for Gammapy (Python, Numpy, Astropy, ...)
- Gammapy features and recent developments
- Gammapy next steps
- Getting started with Gammapy
- Concluding remarks



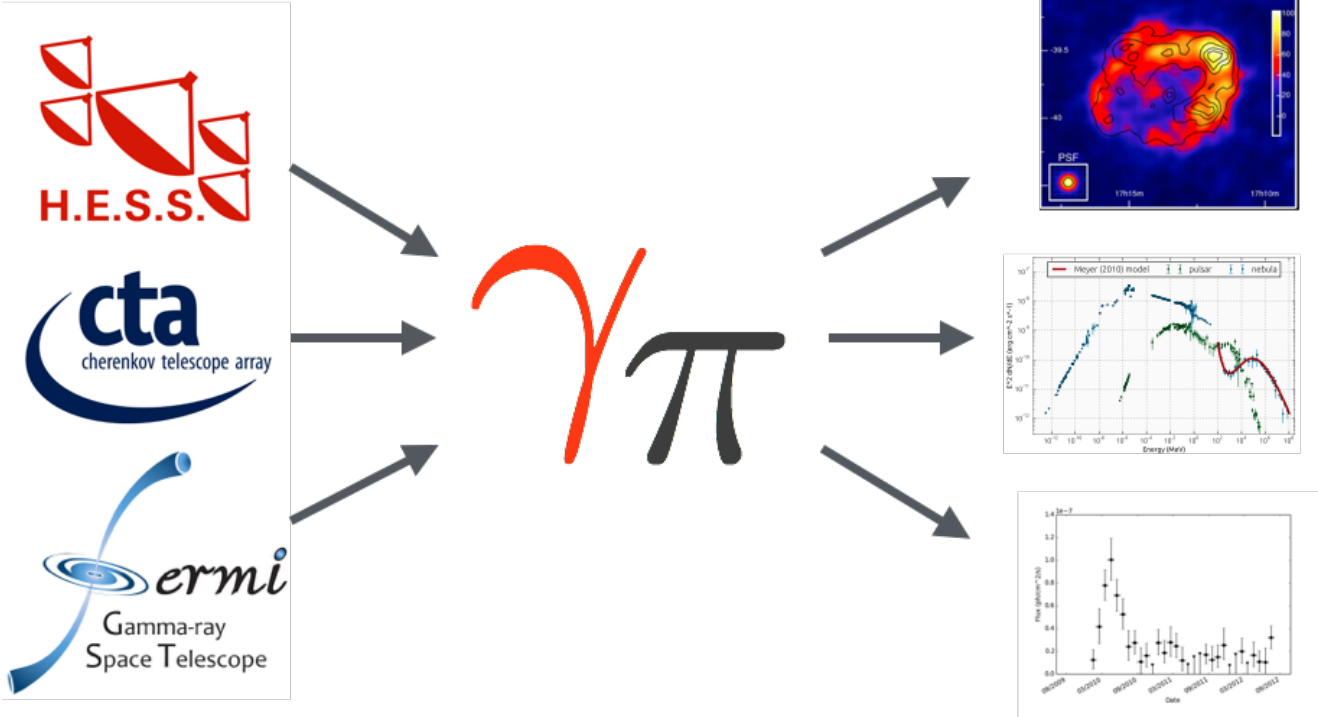
cherenkov
telescope
array

What is Gammapy?

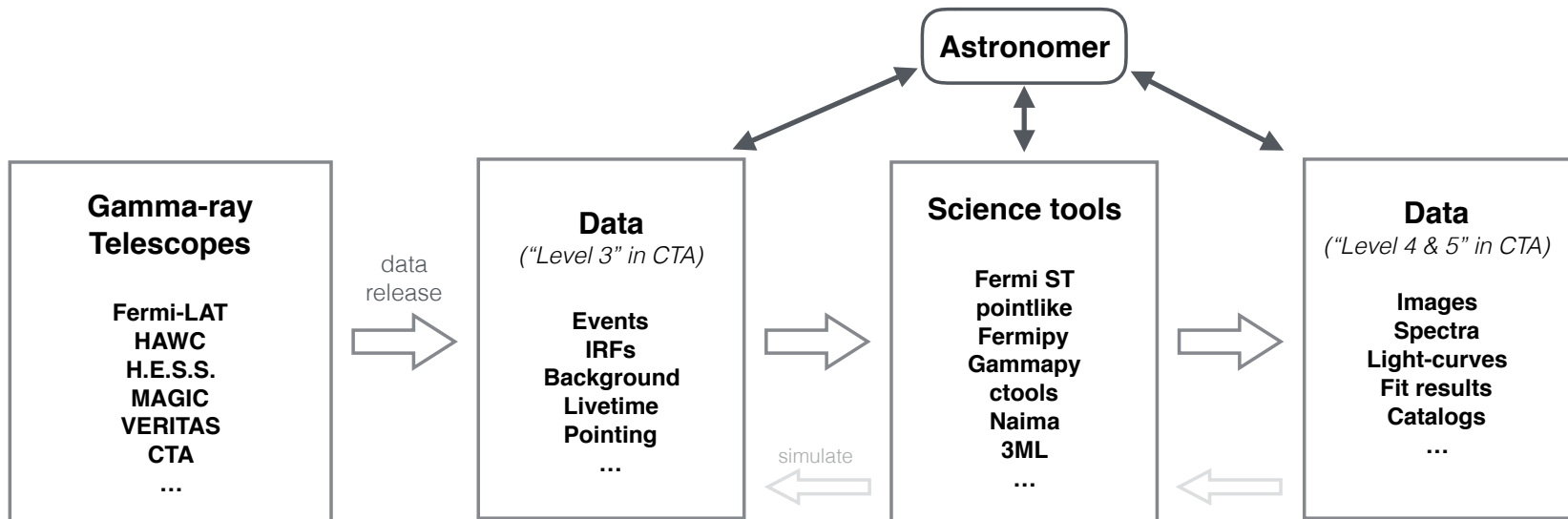
Introduction, idea, philosophy, status



What is Gammapy?



What is Gammapy?



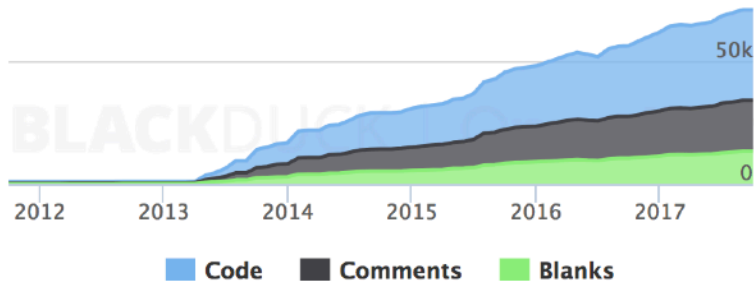
What is Gammapy?



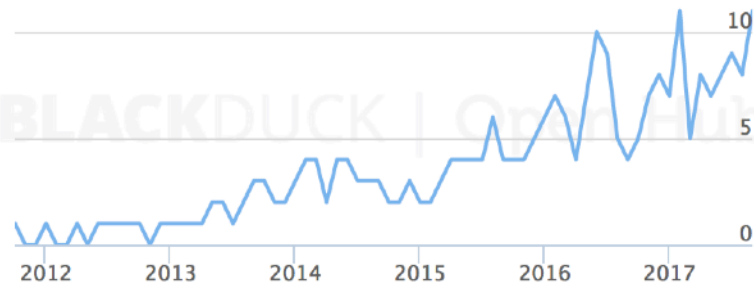
What is Gammapy?



Lines of Code



Contributors per Month



Gammapy – A prototype for the CTA science tools



C. Deil¹, R. Zanin¹, J. Lefaucheur², C. Boisson³, B. Khélifi⁴, R. Terrier⁵, M. Woody⁶, L. Mohrmann⁷, N. Chakraborty⁸, J. Watson¹⁷, R. López Coto⁹, S. Klepser¹⁰, M. Cerruti¹¹, J.-P. Lenain¹², F. Acero¹³, A. Djannati-Atai¹⁴, S. Pita¹⁵, Z. Bošnjak¹⁶, J.E. Ruiz¹⁶, C. Trichard¹⁷, T. Vaillanet¹⁸ for the CTA Consortium, A. Donath¹⁹, J. King²⁰, L. Julavin²¹, E. Owen²², M. Paz Arribas²³, B. Sipek²⁴, D. Lenzini²⁵, A. Vercigiani²⁶, M. Spisak²⁷

¹MPK, ²INiTH, ³APC, ⁴SLAC, ⁵FAU, ⁶DESY, ⁷LENHE, ⁸CRA, ⁹Rijeka University, ¹⁰IAA CSIC, ¹¹CPPM, ¹²LAPP, ¹³UIUC, ¹⁴MSSU, ¹⁵Humboldt University, ¹⁶Cambridge, ¹⁷Georgia Tech, ¹⁸Oxford. See www.cta-observatory.org

ABSTRACT

Gammapy is a Python package for gamma-ray astronomy, built on Numpy and Astropy, and a prototype for the Cherenkov Telescope Array (CTA) science tools. Here we give an introduction to Gammapy and show an application example how to create a sky image using simulated CTA event data and instrument response functions. For further information, visit docs.gammapy.org and follow the links to "tutorials" and "CTA".

Supported by:



CTA science tools

The Cherenkov Telescope Array (CTA) will observe the sky in very-high-energy gamma-ray light soon. All astronomers will have access to CTA high-level data, as well as CTA science tools (ST) software. The ST can be used for example to generate sky images and to measure source properties such as morphology, spectra and light curves, using event lists as well as instrument response function (IRF) and auxiliary information as input.



Fig. 1: Gammapy is a Python package, built on Numpy and Astropy as core dependencies.

```
***Make a simple image with Gammapy***
from gammapy.data import EventList
from gammapy.maps import SkyImage
events = EventList.read('events.fits')
image = SkyImage.from_events(
    events, geom=Geom.from_regions(
        [Rect(0, 0, 2, 2)], unit='deg',
    )
)
image.plot(geom=geom)
```

Fig. 2: Gammapy code example: efficiently work with events and pixels from Python, by always storing data in Numpy arrays

Behind the scenes ...

Figure 2 shows a Gammapy code example that generates a counts image from an event list. The key point here is that all data are stored in Numpy arrays and processed efficiently via calls into existing C extensions in Numpy and Astropy. For instance, in the example, the EventList and SkyImage objects store coordinates and pixel data as Numpy arrays, respectively. Data processing routines such as `image.from_events()` are based on Numpy histograms and calls into the CFITSIO and WCSLIB C libraries.

Gammapy

Gammapy is a prototype for the CTA ST, built on the scientific Python stack and Astropy, optionally using Shorpa or other packages for modeling and fitting (see Figure 1). Our initial focus was to implement the common TeV analysis methods, i.e. using 2-dimensional sky images for source detection and morphology characterization, followed by spectral analysis or light curve computation for a given source region. A 3-dimensional analysis with a simultaneous spatial and spectral models of the gamma-ray emission, as well as background is in development. Further developments and verification using data from existing Cherenkov Telescope arrays such as H.E.S.S. and MAGIC, as well as simulated CTA data is ongoing.

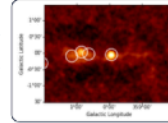


Fig. 3: Significance image computed with Gammapy. Galactic centre region using 1.5 hours of simulated data with CTA South.

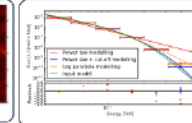


Fig. 4: Spectrum simulated and analysed with Gammapy. Example of a very short exposure (100 seconds) with CTA of a very bright source.

Getting started

Visit docs.gammapy.org and browse the tutorial/Jupyter notebooks to get an overview of what Gammapy can do. To install Gammapy, use one of `pip install gammapy` or `conda install -c astropy gammapy` try out some of the tutorials locally, and then adapt them to your application and use cases. Note that Gammapy is under very active development. If you have any questions, feature requests or find an issue, please contact the Gammapy mailing list or issue tracker.

CTA applications

We show an example of a sky image and spectrum computed with Gammapy in Figures 3 and 4.

For further CTA application examples using Gammapy, see the following poster here at ICRC 2017: R. Zanin (Galactic plane), C. Trichard (PeVatrons), J. Lefaucheur (extra-galactic sources)

ACKNOWLEDGEMENTS: We gratefully acknowledge financial support from the agencies and organizations listed here: www.cta-observatory.org/consortium_acknowledgments

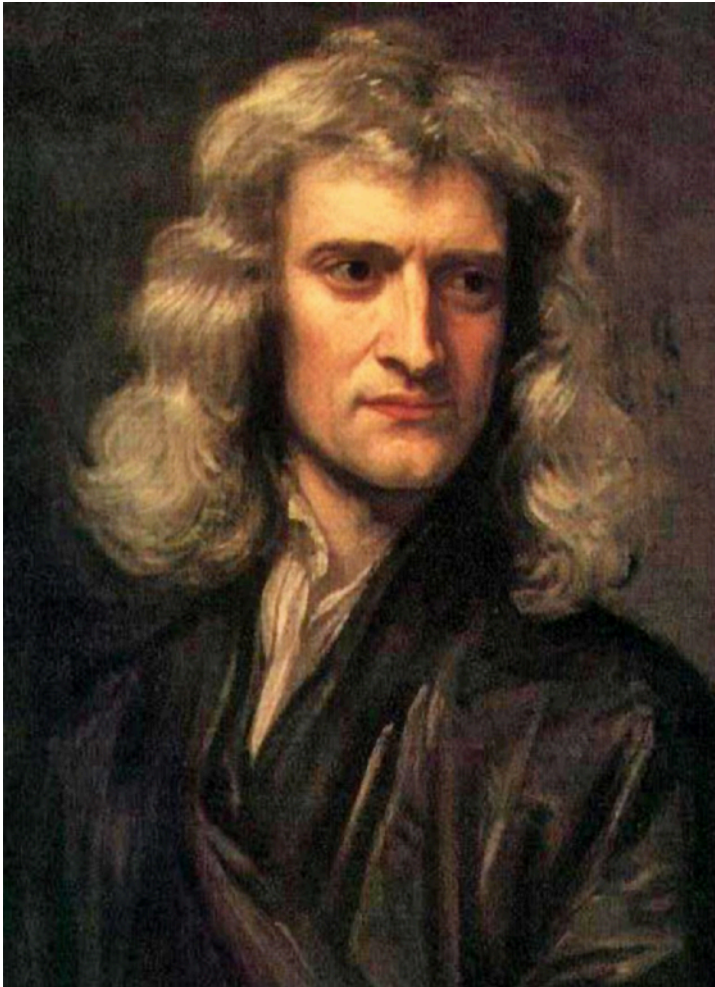
Conclusions

In the past two years, we have developed Gammapy as an open-source analysis package for existing gamma-ray telescope and as a prototype for the CTA science tools.

We find that the Gammapy approach, to build on the powerful and well-tested Python packages Numpy and Astropy, brings large benefits: A small codebase that is focused on gamma-ray astronomy in a single high-level language is easy to understand and maintain. It is also easy to modify and extend as new use cases arise, which is important for CTA, since it can be expected that the modeling of the instrument, background and astrophysical emission, as well as the analysis method in general will evolve and improve over the next decade.

Last but not least, the Gammapy approach is inherently collaborative, sharing development effort as well as know-how with the larger astronomical community, that to a large degree already has adopted Numpy and Astropy as the basis for astronomical analysis codes in the past 5 years.

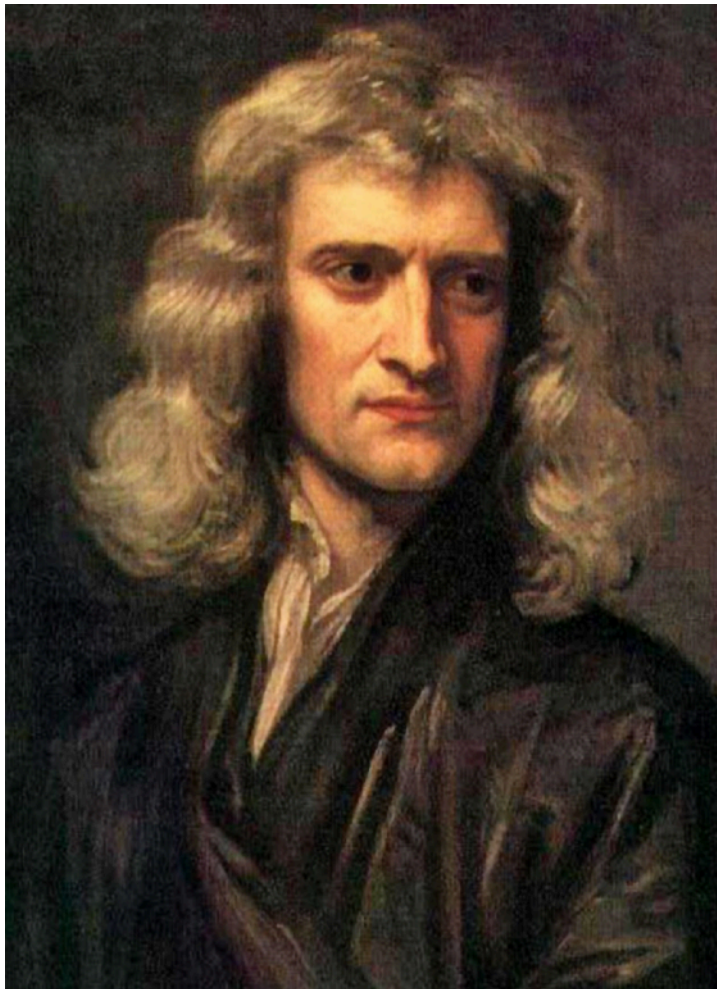
Gammapy philosophy



“If I have seen further, it is by standing on the shoulders of giants.”

— *Issac Newton*

Gammapy philosophy



“If I have seen further, it is by importing
from the code of giants.”

— *Gammapy developer*

Gammapy philosophy

- Python first
 - High-level nice language
- Build on existing scientific Python and astronomy packages
 - Concentrate on gamma-ray astronomy
- Interoperable and flexible
 - Event and pixel data in numpy arrays
- Collaborate
 - Development on Github
 - An Astropy-affiliated package
 - Contribute to related packages (healpix, regions, ...)

Gammapy status

- Gammapy development started in 2013, it is under heavy development, not a complete and finished science tool package
- Note that the same is true for the CTA data model and IRFs that are the Gammapy input, event classes and types as well as IRFs and background models are very preliminary
- With Gammapy, you can currently read event data and IRFs from HESS, MAGIC, CTA, do a “classical” VHE data analysis (2-dim images, 1-dim spectra, lightcurve)
- A first working Sherpa-based prototype for 3D analysis exists, but probably should be re-written (see comments later)
- More info about available features later and tomorrow in the tutorials

Context for Gammapy

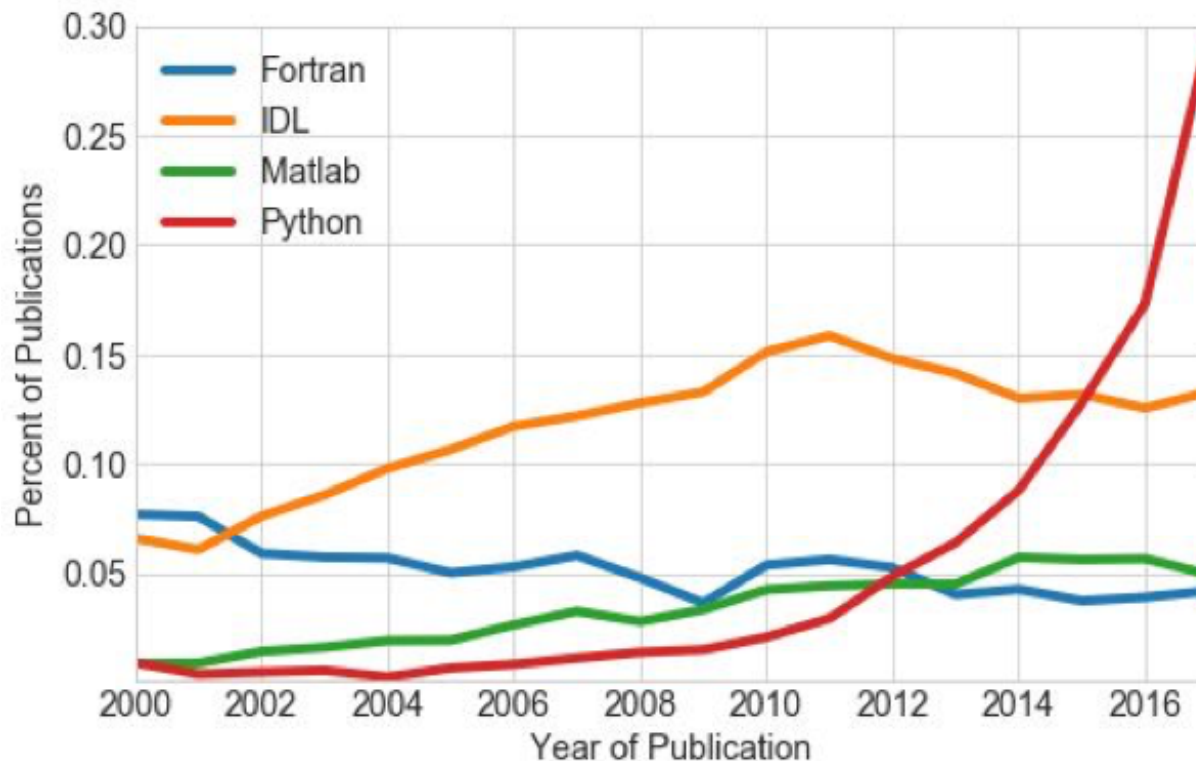
Origins of Python, Numpy, Astropy



Python in astronomy (science tools)



Over the past decade, Python has become the favourite language for astronomers (but also for many other domains, not discussed here)



Python in astronomy (pipelines)



Over the past decade, Python has also become the favourite language for astronomy data pipelines

Kepler/K2 Guest Observer Office

Repositories: 25

Top languages: Python, CSS, C, Makefile, Jupyter Notebook

Most used topics: k2, kepler

K2fov
Check whether targets are in the field of view of NASA's K2 space telescope
Python 5 Updated 3 hours ago

oktopus
A soft-bodied, eight-armed package for beautiful inference.

STScI-JWST

Repositories: 2

Top languages: Python, Jupyter Notebook

Most used topics: astronomy, jwst

jwst
JWST Calibration Pipeline
python astronomy jwst
Jupyter Notebook 32 Updated 2 days ago

wss_tools
Python tools for JWST Wavefront Sensing Software

LSST

Large Synoptic Survey Telescope - Astronomy that's Wider, Faster, Deeper

Repositories: 328

Top languages: Python, Shell, C++, TeX, Jupyter Notebook

Most used topics: k2, kepler

versiondb
1 Updated 2 hours ago

qserv
LSST Query Services
C++ 16 Updated 13 hours ago

cta-observatory

unofficial CTA repo

Repositories: 12

Pinned repositories:

- ctapipe**
CTA Low-level Data Processing Pipeline Prototype
Python 12 Updated 82

But why Python?

- Python was created by Guido van Rossum ~ 1990.
- *“My initial goal for Python was to serve as a second language for people who were C or C++ programmers, but who had work where writing a C program was just not effective.”*
- *“Bridge the gap between the shell and C.”*



The genesis of scientific Python



- Numpy and Scipy were created by Travis Oliphant and others in ~ 2006
- (based on the earlier Numeric and Numarray packages from the 90s)



- *“Prior to Python, I used Perl (for a year) and then Matlab and shell scripts & Fortran & C/C++ libraries. When I discovered Python, I really liked the language... But, it was very nascent and lacked a lot of libraries. I felt like I could add value to the world by connecting low-level libraries to high-level usage in Python. “*
— Travis Oliphant

Python enters astronomy

- Python became the #1 language in astronomy in the past few years, but it entered astronomy over a decade ago.
- StSci / Hubble were an early adopter and contributor, led by Perry Greenfield
- PyData 2015 talk: "How Python Found its way Into Astronomy" (https://youtu.be/uz53IV1V_Xo)
- Python in Astronomy 2015 talk: "The Development and Future of Python at STSci" (https://youtu.be/R_UcjjUC8bE)
- *"Python is a language that is very powerful for developers, but is also accessible to Astronomers. Getting those two classes of people using the same tools, I think, provides a huge benefit that's not always noticed or mentioned. "*
— Perry Greenfield



The genesis of Astropy



June 9th 2011 on the Astropy mailing list ...

- [\[AstroPy\] PyAstronomy](#) *Stefan Czesla*
 - [\[AstroPy\] Proliferating py-astro-libs](#) *Marshall Perrin*

On Jun 9, 2011, at 12:54 PM, Stefan Czesla wrote:

Dear all,

we would like to let you know about our recent release of a -- hopefully -- useful contribution to Python's astronomy community, namely, our PyAstronomy package

On 10/06/11 8:25 AM, Marshall Perrin wrote:

Hopefully without sounding too critical of you in particular, I'm going to ask: do we as a community really need /yet another/ separate python library for astronomy

The situation before Astropy

Astronomical coordinates, FITS, WCS, ...

—> too many packages, many ways to do the same thing

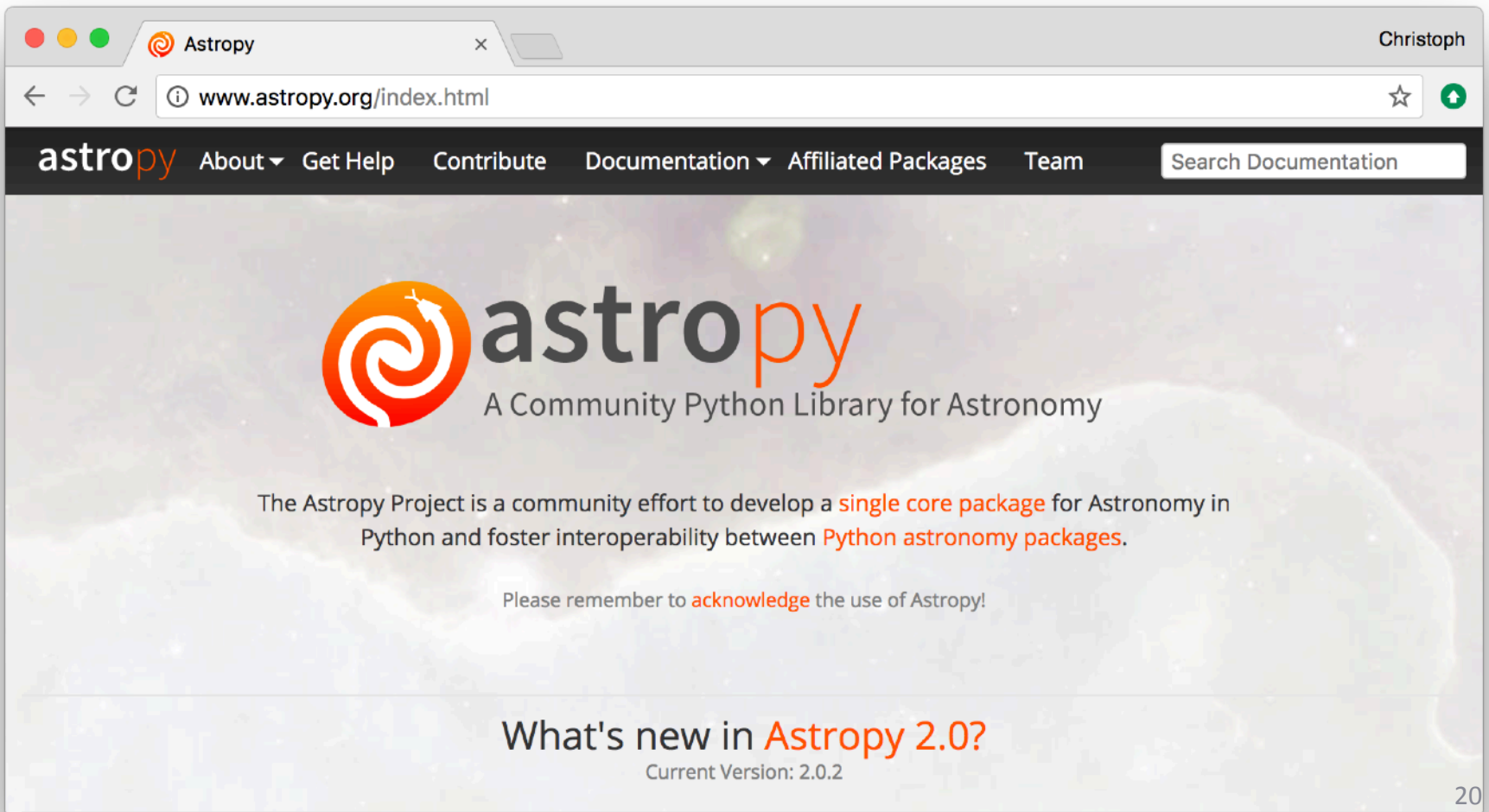
—> often quality not very high and long-term support uncertain




Astropy project



Basic idea: **Astropy core package** with functionality that many astronomers need (with only one required dependency: Numpy), plus an ecosystem of **affiliated packages** that build on the core package.

A screenshot of a web browser displaying the Astropy website. The browser's address bar shows 'www.astropy.org/index.html'. The website's navigation bar includes links for 'About', 'Get Help', 'Contribute', 'Documentation', 'Affiliated Packages', and 'Team', along with a search box for 'Search Documentation'. The main content area features the Astropy logo, which consists of a red spiral with a white arrow pointing upwards, followed by the text 'astropy' in a large, dark font. Below the logo, it reads 'A Community Python Library for Astronomy'. A paragraph of text explains that the Astropy Project is a community effort to develop a 'single core package' for Astronomy in Python and foster interoperability between 'Python astronomy packages'. A note asks users to 'acknowledge' the use of Astropy. At the bottom, there is a section titled 'What's new in Astropy 2.0?' with the current version listed as 2.0.2. The browser's name 'Astropy' and the user's name 'Christoph' are visible in the top right corner of the window.

astropy About ▾ Get Help Contribute Documentation ▾ Affiliated Packages Team Search Documentation

 **astropy**
A Community Python Library for Astronomy

The Astropy Project is a community effort to develop a **single core package** for Astronomy in Python and foster interoperability between **Python astronomy packages**.

Please remember to **acknowledge** the use of Astropy!

What's new in **Astropy 2.0?**
Current Version: 2.0.2

Astropy project



- Overall very successful in it's first six years
- Set up in a good way technically: Github, code review, docs, tests, ...
- Set up in a good way from a project management perspective: coordination committee, other roles, meetings, ...
- Some institutional support, especially from StScI, but getting direct funding remains a challenge ([2016arXiv161003159M](https://arxiv.org/abs/1610.03159)).
- Funding for "infrastructure" or "base" projects that aren't project-specific has always been a problem. E.g. Numpy only managed in 2017 to get it's first direct funding (700k\$).

A&A 558, A33 (2013)
DOI: 10.1051/0004-6361/201322068
© ESO 2013

Astronomy
&
Astrophysics

Astropy: A community Python package for astronomy

The Astropy Collaboration, Thomas P. Robitaille¹, Erik J. Tollerud^{2,3}, Perry Greenfield⁴, Michael Droettboom⁴, Erik Bray⁵, Tom Aldcroft⁶, Matt Davis⁷, Adam Ginsburg⁸, Adrian M. Price-Whelan⁹, Wolfgang E. Kerzendorf⁸, Alexander Conley⁹, Neil Crighton⁴, Kyle Barbary⁹, Demitri Muna¹⁰, Henry Ferguson⁴, Frédéric Grollier¹², Madhura M. Parikh¹¹, Prasanth H. Nair¹², Hans M. Günther⁵, Christoph Deil¹³, Julien Woillez¹⁴, Simon Conseil¹⁵, Roban Kramer¹⁶, James E. H. Turner¹⁷, Leo Singer¹⁸, Ryan Fox¹², Benjamin A. Weaver¹⁹, Victor Zabalza¹³, Zachary I. Edwards²⁰, K. Azalee Bostroem⁴, D. J. Burke⁵, Andrew R. Casey²¹, Steven M. Crawford²², Nadia Dencheva⁴, Justin Ely⁴, Tim Jenness^{23,24}, Kathleen Labrie²⁵, Pey Lian Lim⁴, Francesco Pierfederici⁴, Andrew Pontzen^{26,27}, Andy Ptak²⁸, Brian Refsdal², Mathieu Servillat^{29,3}, and Ole Streicher³⁰

¹ Max-Planck-Institut für Astronomie, Königstuhl 17, 69117 Heidelberg, Germany
e-mail: robitaille@mpia.de

² Department of Astronomy, Yale University, PO Box 208101, New Haven, CT 06510, USA

³ Hubble Fellow

⁴ Space Telescope Science Institute, 3700 San Martin Drive, Baltimore, MD 21218, USA

⁵ Harvard-Smithsonian Center for Astrophysics, 60 Garden Street, Cambridge, MA 02138, USA

⁶ Center for Astrophysics and Space Astronomy, University of Colorado, Boulder, CO 80309, USA

⁷ Department of Astronomy, Columbia University, Pupin Hall, 550W 120th St., New York, NY 10027, USA

⁸ Department of Astronomy and Astrophysics, University of Toronto, 30 Saint George Street, Toronto, ON M5S3H4, Canada

⁹ Argonne National Laboratory, High Energy Physics Division, 9700 South Cass Avenue, Argonne, IL 60439, USA

¹⁰ Department of Astronomy, Ohio State University, Columbus, OH 43210, USA

¹¹ S.V.National Institute of Technology, 395007 Surat., India

¹² Independent developer

¹³ Max-Planck-Institut für Kernphysik, PO Box 103980, 69029 Heidelberg, Germany

¹⁴ European Southern Observatory, Karl-Schwarzschild-Str. 2, 85748 Garching bei München, Germany

¹⁵ Laboratoire d'Astrophysique de Marseille, OAMP, Université Aix-Marseille et CNRS, 13388 Marseille, France

¹⁶ ETH Zürich, Institute for Astronomy, Wolfgang-Pauli-Strasse 27, Building HIT, Floor J, 8093 Zurich, Switzerland

¹⁷ Gemini Observatory, Casilla 603, La Serena, Chile

¹⁸ LIGO Laboratory, California Institute of Technology, 1200 E. California Blvd., Pasadena, CA 91125, USA

¹⁹ Center for Cosmology and Particle Physics, New York University, New York, NY 10003, USA

²⁰ Department of Physics and Astronomy, Louisiana State University, Nicholson Hall, Baton Rouge, LA 70803, USA

²¹ Research School of Astronomy and Astrophysics, Australian National University, Mount Stromlo Observatory, via Cotter Road, Weston Creek ACT 2611, Australia

²² SAAO, PO Box 9, Observatory 7935, 7925 Cape Town, South Africa

²³ Joint Astronomy Centre, 660 N. A'ohōkū Place, Hilo, HI 96720, USA

²⁴ Department of Astronomy, Cornell University, Ithaca, NY 14853, USA

²⁵ Gemini Observatory, 670 N. A'ohōkū Place, Hilo, HI 96720, USA

²⁶ Oxford Astrophysics, Denys Wilkinson Building, Keble Road, Oxford OX1 3RH, UK

²⁷ Department of Physics and Astronomy, University College London, London WC1E 6BT, UK

²⁸ NASA Goddard Space Flight Center, X-ray Astrophysics Lab Code 662, Greenbelt, MD 20771, USA

²⁹ Laboratoire AIM, CEA Saclay, Bât. 709, 91191 Gif-sur-Yvette, France

³⁰ Leibniz-Institut für Astrophysik Potsdam (AIP), An der Sternwarte 16, 14482 Potsdam, Germany

Received 12 June 2013 / Accepted 23 July 2013

ABSTRACT

We present the first public version (v0.2) of the open-source and community-developed Python package, Astropy. This package provides core astronomy-related functionality to the community, including support for domain-specific file formats such as flexible image transport system (FITS) files, Virtual Observatory (VO) tables, and common ASCII table formats, unit and physical quantity conversions, physical constants specific to astronomy, celestial coordinate and time transformations, world coordinate system (WCS) support, generalized containers for representing gridded as well as tabular data, and a framework for cosmological transformations and conversions. Significant functionality is under active development, such as a model fitting framework, VO client and server tools, and aperture and point spread function (PSF) photometry tools. The core development team is actively making additions and enhancements to the current code base, and we encourage anyone interested to participate in the development of future Astropy versions.

Key words. methods: data analysis – methods: miscellaneous – virtual observatory tools

Astropy core package



Documentation in Sphinx, some Jupyter tutorial notebooks

The screenshot shows the Astropy documentation website. The browser address bar is `docs.astropy.org/en/stable/`. The page header includes the `astro.py:docs` logo and navigation links for `Index` and `Modules`. The main content area features the Astropy logo and the text: "A Community Python Library for Astronomy". Below this, a paragraph describes the package's role in the Astropy Project. The page is organized into sections: "Getting Started" with a list of links (Installation, What's New in Astropy 2.0?, Importing astropy and subpackages, Getting started with subpackages, Example Gallery, Tutorials, Get Help, Contribute and Report Problems, About the Astropy Project), "User Documentation", and "Data structures and transformations" with a list of links (Constants, Units and Quantities, N-dimensional datasets, Data Tables, Time and Dates, Astronomical Coordinate Systems, World Coordinate System, Models and Fitting). A version selector at the bottom right shows "v: stable".

The screenshot shows a page from the Astropy documentation website, displaying a list of modules. The browser address bar is `docs.astropy.org/en/stable/`. The page content is organized into sections: "Files, I/O, and Communication" (Unified file read/write interface, FITS File handling, ASCII Tables, VOTable XML handling, Miscellaneous: HDF5, YAML, pickle, SAMP), "Computations and utilities" (Cosmological Calculations, Convolution and filtering, Data Visualization, Astrostatistics Tools), "Nuts and bolts" (Configuration system, I/O Registry, Logging system, Python warnings system, Astropy Core Package Utilities, Astropy Testing Tools, Try the development version), and "Developer Documentation" (instructions on contributing, coding, documentation, and testing guidelines, and a link to the project vision). A footer note mentions additional tools for developers in the `astropy/astropy-tools` repository. A version selector at the bottom right shows "v: stable".

Astropy development



GitHub

Repository for the Astropy core package <http://www.astropy.org> — Edit

13,977 commits 11 branches 34 releases 142 contributors

Branch: master **astropy / +**

File/Folder	Description	Last Commit
<code>.continuous-integration</code>	For consistency / clarity, rename the PIP variable to PIP_INSTALL [sk...	a month ago
<code>astropy</code>	Use string in skipif() for pytest 2.3 compatibility	3 days ago
<code>astropy_helpers @ 01a97a2</code>	Updated astropy-helpers again to the latest master, incorporating sev...	a month ago
<code>cextern</code>	Merge pull request #4045 from mdboom/wcs/precision	26 days ago
<code>dev</code>	Add tool to fix up parsing tables	3 months ago
<code>docs</code>	Merge pull request #4238 from bspocz/docs_fixing_typos_VI_II	10 days ago
<code>licenses</code>	Upgrade PLY to 3.6	3 months ago
<code>static</code>	Fixed support on Python 3, and got rid of .astropy-root per astropy/a...	11 months ago
<code>.astropy-root</code>	Don't rely on .git to enable auto-build when importing from source tr...	4 months ago
<code>.gitattributes</code>	Use union merge for changelog	11 months ago
<code>.gitignore</code>	ignore .swf files generated by vim	6 months ago
<code>.gitmodules</code>	Update the astropy_helpers URL to the real astropy-helpers.	2 years ago
<code>.mailmap</code>	update .mailmap for impending release	9 months ago
<code>.travis.yml</code>	Adding pytz to .travis.yml	a month ago

Code

- Issues 514
- Pull requests 103
- Wiki
- Pulse
- Graphs
- Settings

SSH clone URL
`git@github.com:ast`

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

[Clone in Desktop](#)

[Download ZIP](#)

Astropy development



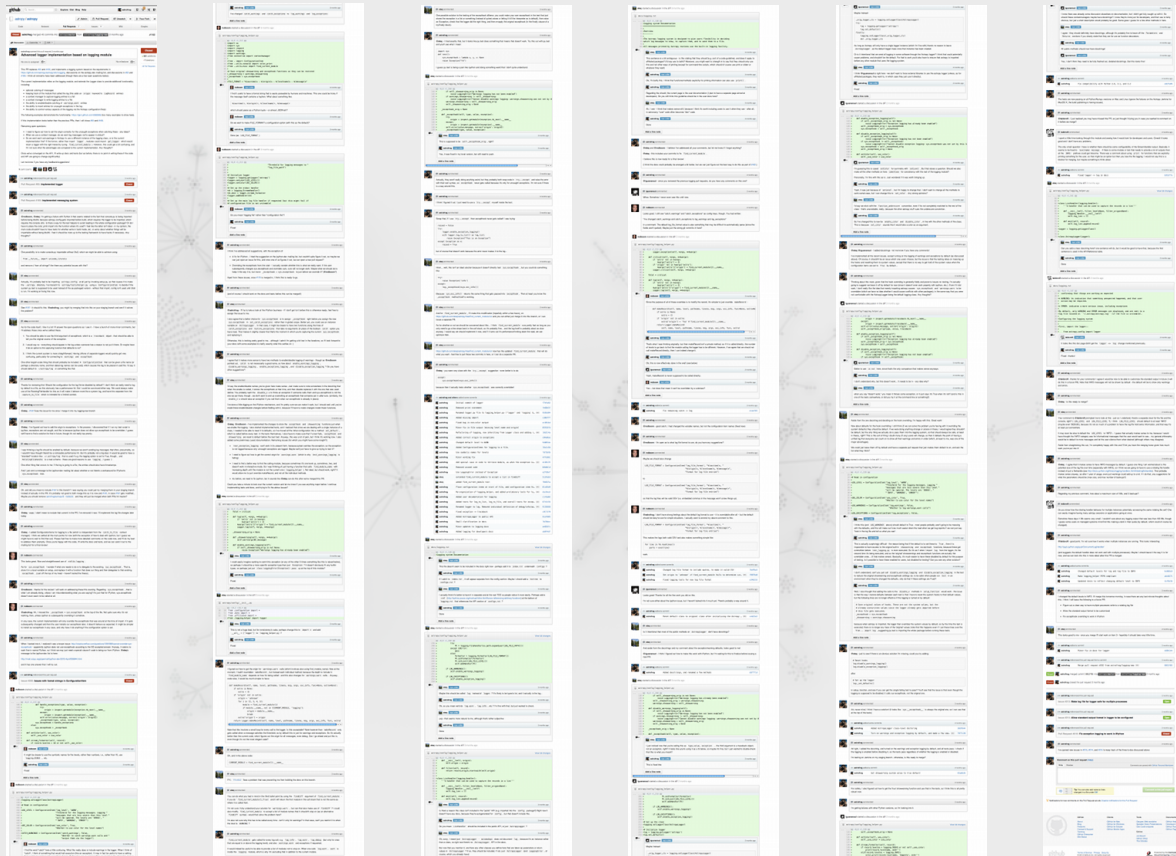
The screenshot shows the GitHub interface for the astropy/astropy repository. The browser address bar displays the URL: <https://github.com/astropy/astropy/pulls?page=2&q=is%3Apr+is%3Aopen>. The repository name is 'astropy / astropy'. The page shows a list of pull requests with the following details:

Issue	Status	Labels	Author	Created	Comments
Fix for issue [3739]	Open	Affects-release, Enhancement, io.fits	anizami	Aug 11	11
Editing modeling docs	Open	Docs, modeling	anizami	Aug 11	12
Handle sorting NaNs and masked values in jsviewer	Open	Affects-release, Bug, table	sargas	Aug 7	14
Some polynomial model refactoring (WIP?)	Open	Affects-release, modeling	embray	Aug 7	9
Update `bounding_box` property and new `Model.render()` method in `astropy.modeling`	Open	Affects-dev, modeling	patti	Aug 5	45
Add sample plots to built-in models	Open	Docs, modeling	sYnfo	Jul 25	9
Retry --open-files check once after garbage collection	Open	Affects-release, Bug, testing	embray	Jul 24	18
POC: tables within tables within tables!	Open	Affects-dev, table	taldraft	Jul 15	3

Astropy development



The code review discussion for a pull request (in an extreme, too large case, where the pull request should probably have been split in multiple parts).




Astropy tests



Automated tests and continuous integration ensure that additions work and changes don't break anything.


```
def test_constellations():  
  
    # the actual test for accuracy is in test_funcs - this is just meant to make  
    # sure we get sensible answers  
  
    sc = SkyCoord(135*u.deg, 65*u.deg)  
    assert sc.get_constellation() == 'Ursa Major'  
    assert sc.get_constellation(short_name=True) == 'UMa'
```


(x 10,000)



 **Some checks were not successful** [Hide all checks](#)
1 errored and 1 successful checks

 continuous-integration/travis-ci/pr — The Travis CI build could not complete...	Details
 continuous-integration/appveyor — AppVeyor build succeeded	Details

 **This branch is up-to-date 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](#).

Astropy affiliated packages

- Roughly two categories:
 - In-development functionality for Astropy core package
Examples: wcsaxes, regions, healpix, reproject
 - Specialised packages that aren't needed by most astronomers. Examples:
 - PINT - pulsar timing
 - Naima - non thermal SED modeling
 - sncosmo - supernova light curve modeling
 - astroplan - astronomical observation planning
 - ... many more ...
- Main idea: collaborate, avoid duplication, increase quality, ...
- More infos on the webpage and in a second Astropy paper that is currently being written (first one was 2013 about Astropy v0.2).

What is Gammapy?

- Gammapy started ~ 2013 by people in H.E.S.S., now used for Fermi-LAT, H.E.S.S., MAGIC, CTA
- Gammapy is an Astropy-affiliated package for gamma-ray astronomy
- Gammapy is a prototype for the CTA science tools



Gammapy development




Gammapy is set up exactly the same way as Astropy
(using the standard open source and Python tools)

GitHub Version control, issue tracker,
contributions via pull requests & code review

Tests automatically run on Linux & Mac
on each pull request and master branch



Travis CI

 **pytest** Python testing framework
(makes it easy to write and run tests)

Python documentation generator
API and narrative docs pages
cross-linked, full-text search



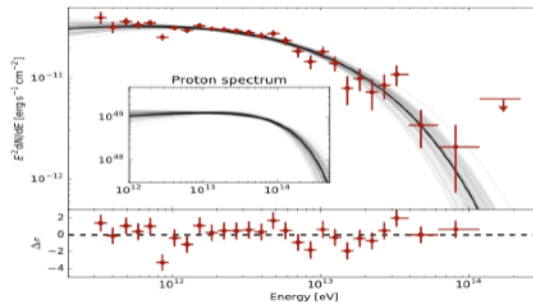
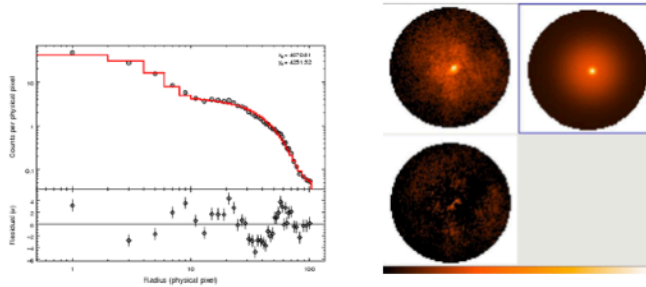
SPHINX



Anaconda

Binary cross-platform package manager.
Install Gammapy and all dependencies on any
Linux & Mac box in \$HOME in 10 min.

Other related packages



```
data:
  evfile : ft1.lst
  scfile : ft2.fits

binning:
  roiwidth : 10.0
  binsz    : 0.1
  binsperdec : 8

from fermipy import GTAnalysis
gta = GTAnalysis('config.yaml')
gta.setup()
gta.fit()
gta.print_roi()
```

SHERPA

- Awesome general modeling and fitting package (similar, but different from astropy.modeling)

NAIMA

- Astropy-affiliated package for non-thermal SED modeling. Fitting using emcee or Sherpa

FERMIPIY

- Fermi-LAT data analysis for humans (using Fermi ScienceTools SWIG Python interface in the background)

Data in Gammapy - Numpy arrays

- One key point I would like to make is that event and pixel data in Gammapy is stored in Numpy arrays
- Algorithms are implemented by calling into existing Numpy, Scipy & Astropy functions that operate on Numpy arrays
- Eventually the computation is executed by existing C / C++ / Fortran code. Uses a simple Python interface, just passing basic types (numbers, strings and array buffers) between Python and C / C++ / Fortran, not Python or C++ objects.
- If needed, Cython is a nice simple option should there be algorithms that can't be written efficiently in Python

```
1  """Make a counts image with Gammapy."""
2  from gammapy.data import EventList
3  from gammapy.image import SkyImage
4  events = EventList.read('events.fits')
5  image = SkyImage.empty(
6      nxpix=400, nypix=400, binsz=0.02,
7      xref=83.6, yref=22.0,
8      coordsys='CEL', proj='TAN',
9  )
10 image.fill_events(events)
11 image.write('counts.fits')
```

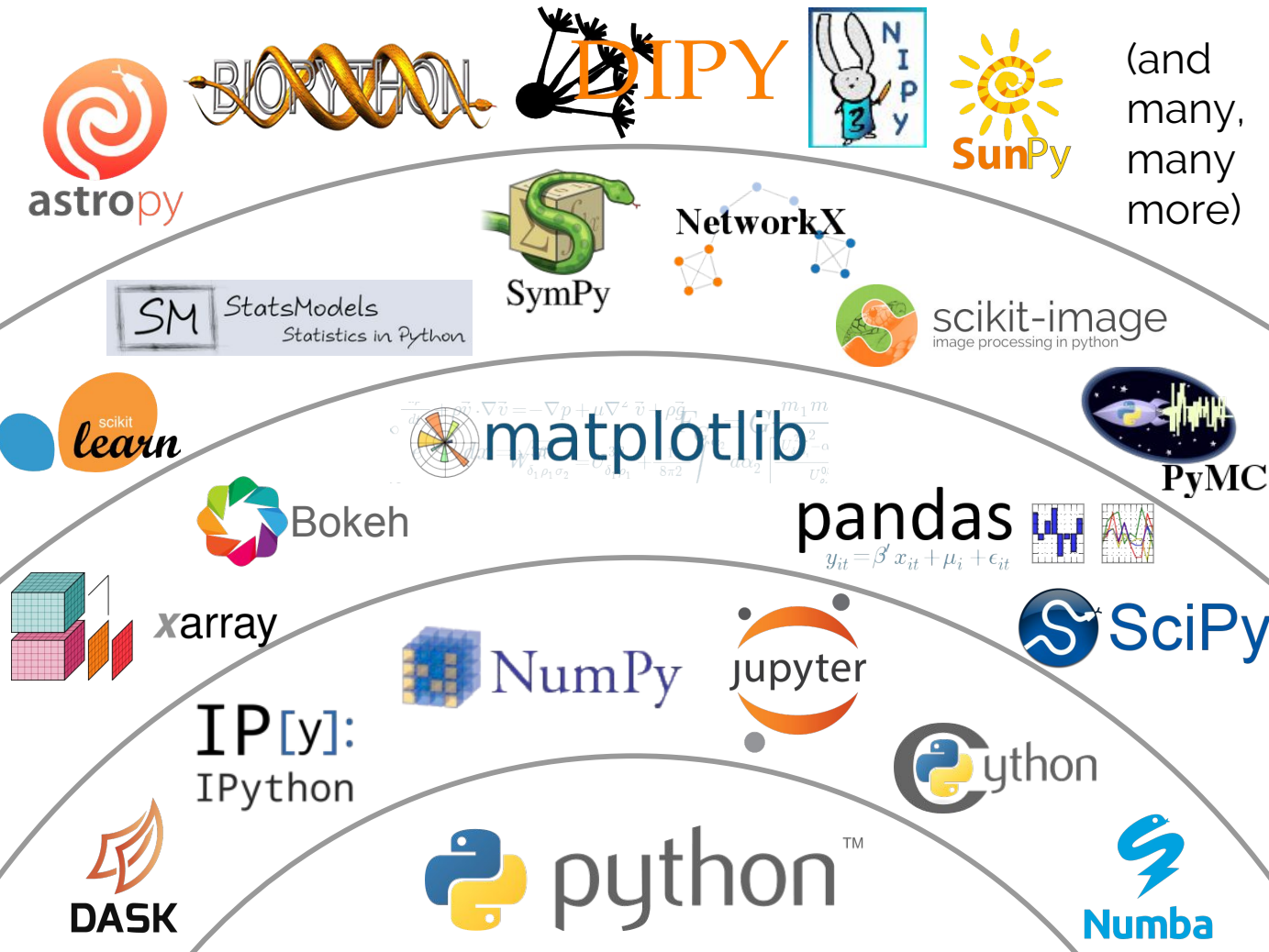
Note that this is different how the Fermi ST / Fermipy and Gammalib / ctools Python interfaces work.

They don't expose event and pixel data as Numpy arrays, and thus it is not easily possible to write custom models (for sky or background) or even IRF models or likelihood functions in Python.

Interoperability with scientific Python stack

Python and Numpy are glue!

Basic philosophy in the scientific Python stack: store data in Numpy arrays, so that they can be easily processed by any package (without a copy) and passed to C / C++ / Fortran codes

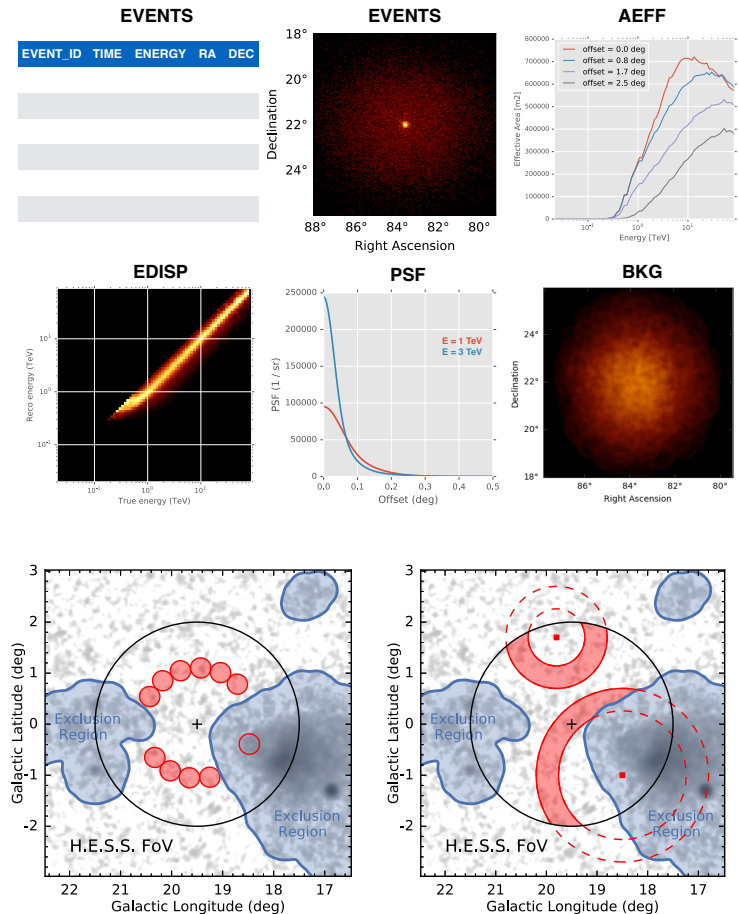


Gammapy features and recent developments



Gammapy features

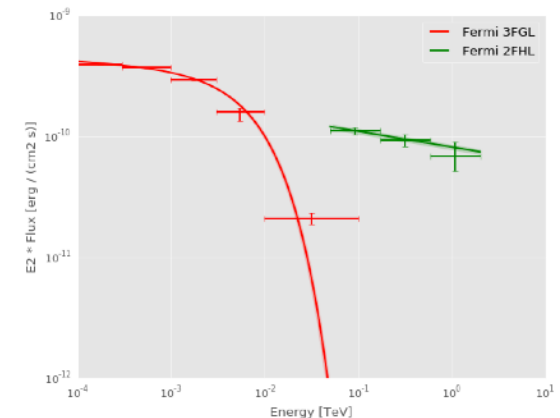
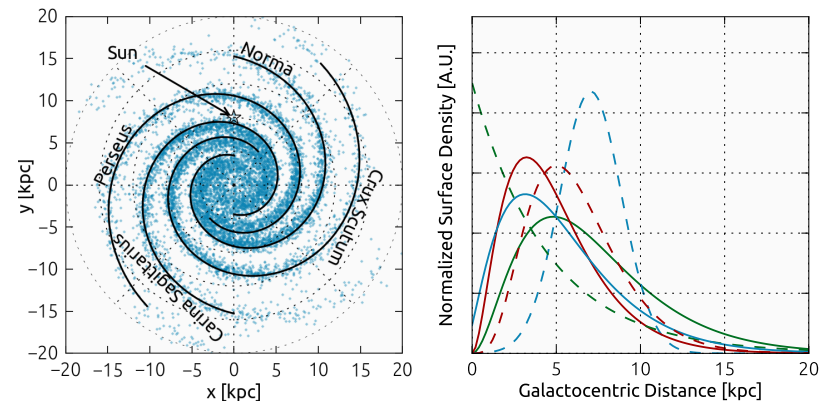
- **gammapy.data & gammapy.irf**
IACT DL3 data handling
- **gammapy.image**
2-dim image analysis
- **gammapy.spectrum**
1-dim region spectral analysis
- **gammapy.background**
*Background modeling methods
(might merge in image, spectrum cube)*
- **gammapy.cube**
3-dim cube analysis (work in progress)
- **gammapy.detect**
Source detection (image-based for now)



Fermi-LAT Galactic plane survey TS image ([tutorial](#))

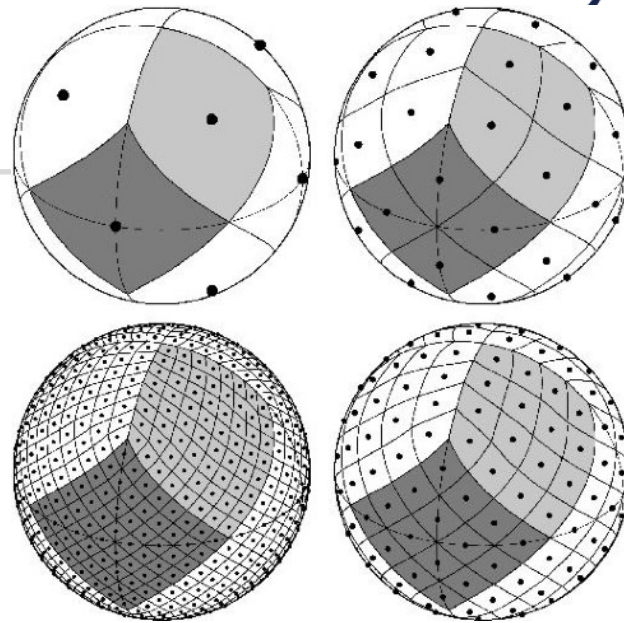
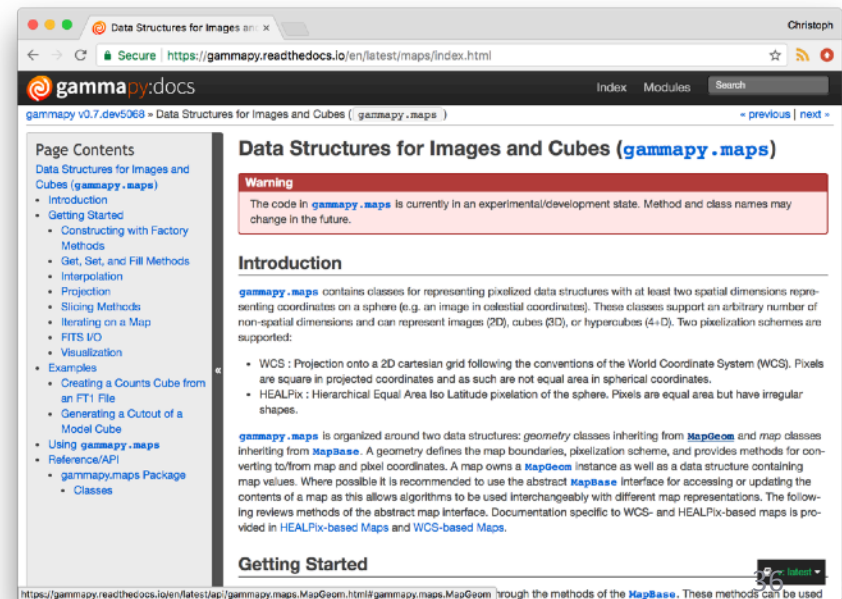
Gammapy features

- **gammapy.stats**
Statistics methods
- **gammapy.time**
Time analysis (not much available yet)
- **gammapy.catalog**
 - Fermi-LAT spectra, lightcurves
 - Next: TeV data ([gamma-cat](#))
- **gammapy.astro**
Some simple models for Galactic sources and source populations
(could go in separate higher-level science package)
- **gammapy.scripts**
Command line interface (CLI) tools for common operations
(not much available yet, see comments on science too user interface in backup slides)



gammapy.maps

- New sub-package `gammapy.maps`, developed by Matthew Wood (Fermi ST, `fermipy`)
- Not used for analysis in Gammapy for analysis yet, but probably this is the basis of analysis in the future.
- Data formats specified, based on experience from Fermi ST, `Fermipy` and pointlike
- Why? Support HEALPix, sparse maps and arbitrary extra axes (n-dim)
- Discussions and some work ongoing with Thomas Robitaille about what belongs in `astropy.healpix` and `astropy.reproject` vs `gammapy.maps`

gammapy v0.7.dev5068 » Data Structures for Images and Cubes (`gammapy.maps`)

Data Structures for Images and Cubes (`gammapy.maps`)

Warning

The code in `gammapy.maps` is currently in an experimental/development state. Method and class names may change in the future.

Introduction

`gammapy.maps` contains classes for representing pixelized data structures with at least two spatial dimensions representing coordinates on a sphere (e.g. an image in celestial coordinates). These classes support an arbitrary number of non-spatial dimensions and can represent images (2D), cubes (3D), or hypercubes (4+D). Two pixelization schemes are supported:

- WCS : Projection onto a 2D cartesian grid following the conventions of the World Coordinate System (WCS). Pixels are square in projected coordinates and as such are not equal area in spherical coordinates.
- HEALPix : Hierarchical Equal Area Iso Latitude pixelation of the sphere. Pixels are equal area but have irregular shapes.

`gammapy.maps` is organized around two data structures: geometry classes inheriting from `MapGeom` and map classes inheriting from `MapBase`. A geometry defines the map boundaries, pixelization scheme, and provides methods for converting to/from map and pixel coordinates. A map owns a `MapGeom` instance as well as a data structure containing map values. Where possible it is recommended to use the abstract `MapBase` interface for accessing or updating the contents of a map as this allows algorithms to be used interchangeably with different map representations. The following reviews methods of the abstract map interface. Documentation specific to WCS- and HEALPix-based maps is provided in HEALPix-based Maps and WCS-based Maps.

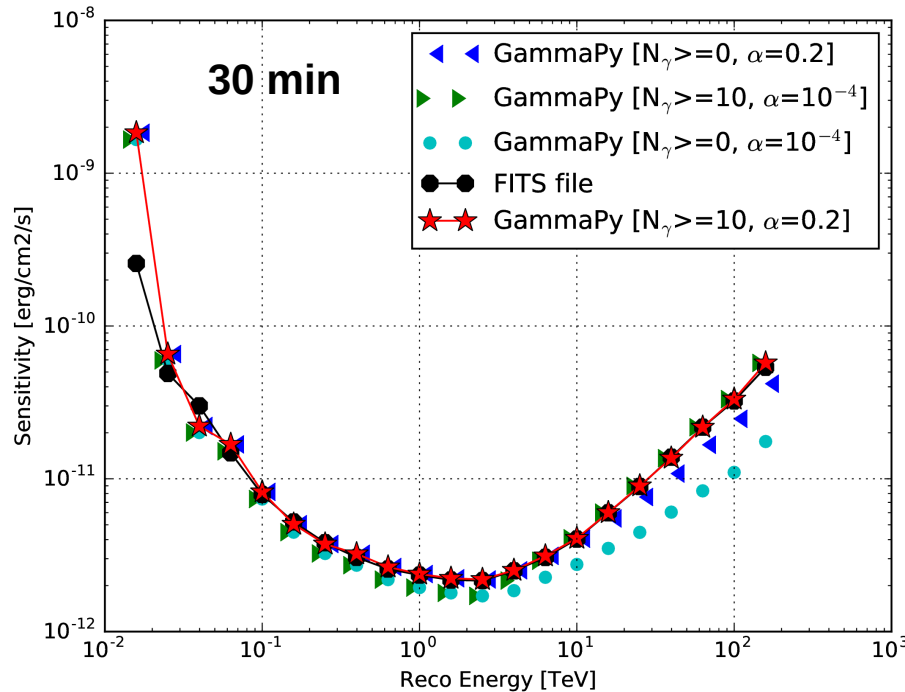
Getting Started

rough the methods of the `MapBase`. These methods can be used



Sensitivity computation

B.Khelifi (& J. Lefaucheur) – 11/05/17



- Reminder of its definition (approved for the TDR publication and thus for the KSPs):
 - $\geq 5\sigma$, $N_\gamma \geq 10$, $N_\gamma \geq 5\% \times N_{\text{bkg}}$, $\alpha = 0.2$
- Use of the GammaPy package (v0.5)
 - Based on the function `spectrum.utils.CountsPredictor`

1) Comparison with the standard definition

In the IRFs, storage of the sensitivity (black)
GammaPy: red star
→ **Good agreement** (<10%), except the lowest energy bin.
Deeper investigation needed (with Gernot)

2) When changing its definition

By relaxing alternatively the constrains on N_γ minimum and α → expected improvements

3) Next steps

Computation at different offsets, for extended sources
Computation for the 3D analysis

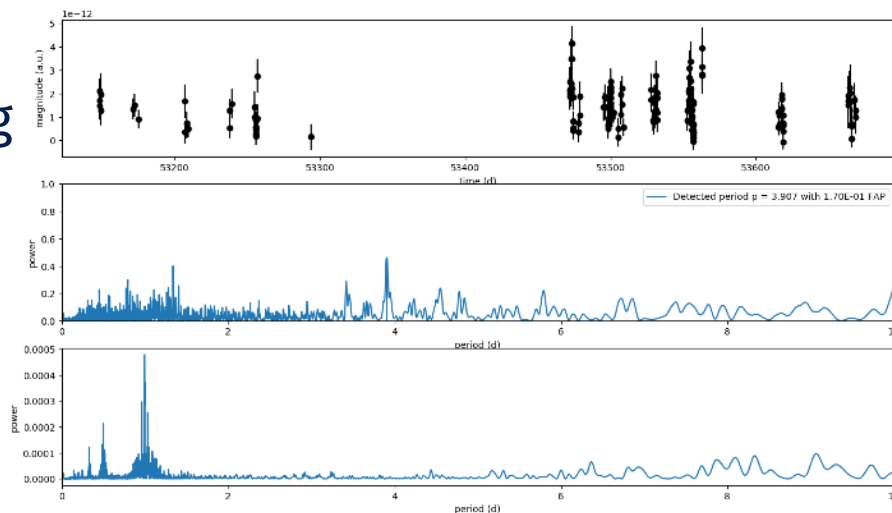


The GammaPy package reproduces well the standard sensitivities

Another recent addition: Lomb-Scargle



- Jake VanderPlas added `astropy.stats.LombScargle`
- Matthias Wegen (DESY Zeuthen) started to add Lomb-Scargle peak significance estimation methods to `gammapy.time`
- Last week saw that Jake is adding similar things to Astropy
- -> it's important to stay in contact and collaborate with the larger astronomy community!

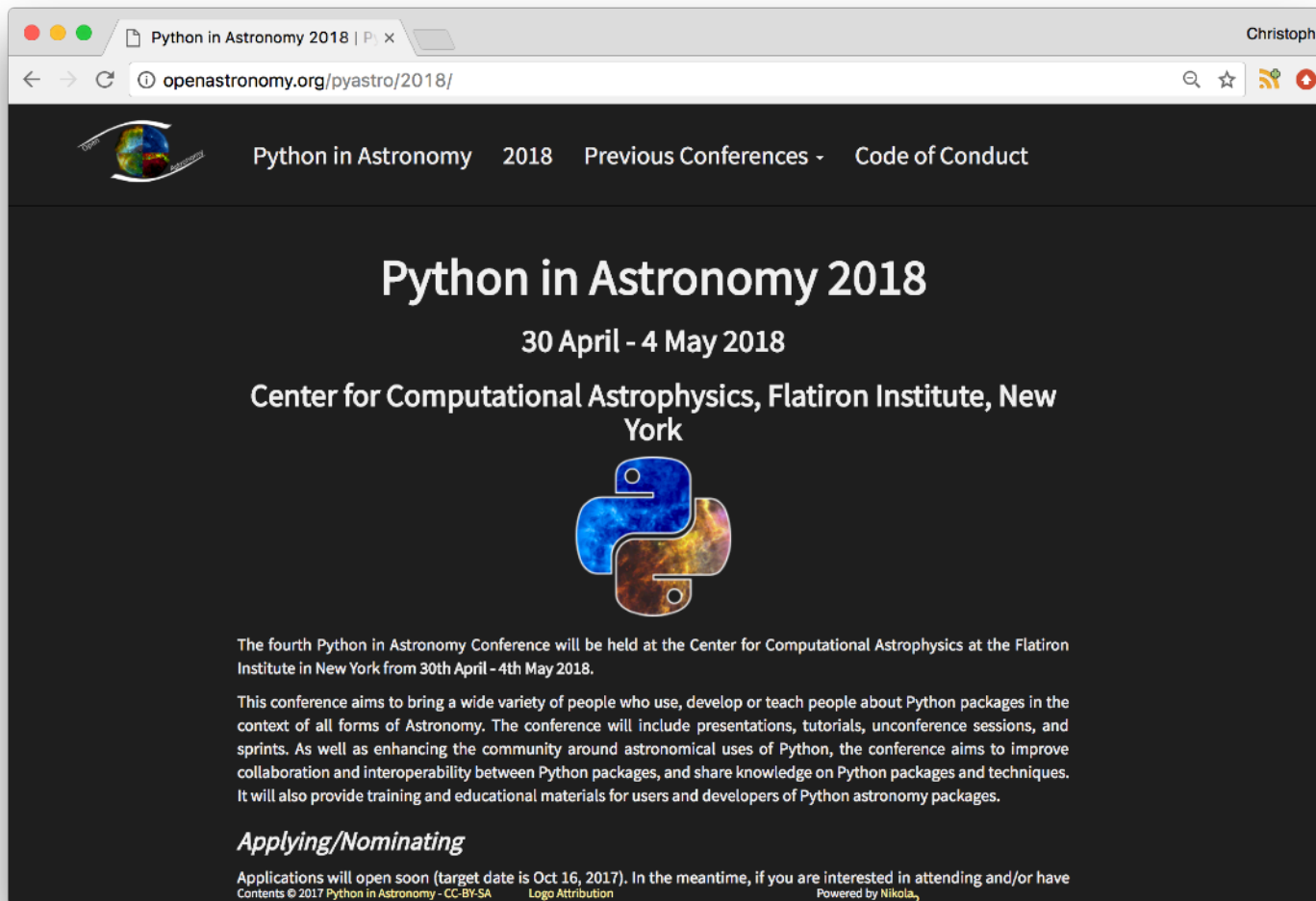


Example using LS 5039 HESS light curve (data from gamma-cat):
<http://docs.gammapy.org/en/latest/time/period.html>

Python in Astronomy yearly conferences



Python in Astronomy 2018 was just announced.
It's a great meeting series, also for beginners, not just for experts!

A screenshot of a web browser displaying the Python in Astronomy 2018 website. The browser's address bar shows the URL 'openastronomy.org/pyastro/2018/'. The website has a dark background with white text. At the top, there is a navigation menu with links for 'Python in Astronomy', '2018', 'Previous Conferences', and 'Code of Conduct'. The main heading is 'Python in Astronomy 2018', followed by the dates '30 April - 4 May 2018' and the location 'Center for Computational Astrophysics, Flatiron Institute, New York'. A Python logo is centered below the text. A paragraph of text describes the conference's goals, and a section titled 'Applying/Nominating' is partially visible at the bottom. The browser's name 'Christoph' is visible in the top right corner.

Python in Astronomy 2018 | P x Christoph


openastronomy.org/pyastro/2018/

Python in Astronomy 2018 Previous Conferences - Code of Conduct

Python in Astronomy 2018

30 April - 4 May 2018

Center for Computational Astrophysics, Flatiron Institute, New York



The fourth Python in Astronomy Conference will be held at the Center for Computational Astrophysics at the Flatiron Institute in New York from 30th April - 4th May 2018.

This conference aims to bring a wide variety of people who use, develop or teach people about Python packages in the context of all forms of Astronomy. The conference will include presentations, tutorials, unconference sessions, and sprints. As well as enhancing the community around astronomical uses of Python, the conference aims to improve collaboration and interoperability between Python packages, and share knowledge on Python packages and techniques. It will also provide training and educational materials for users and developers of Python astronomy packages.

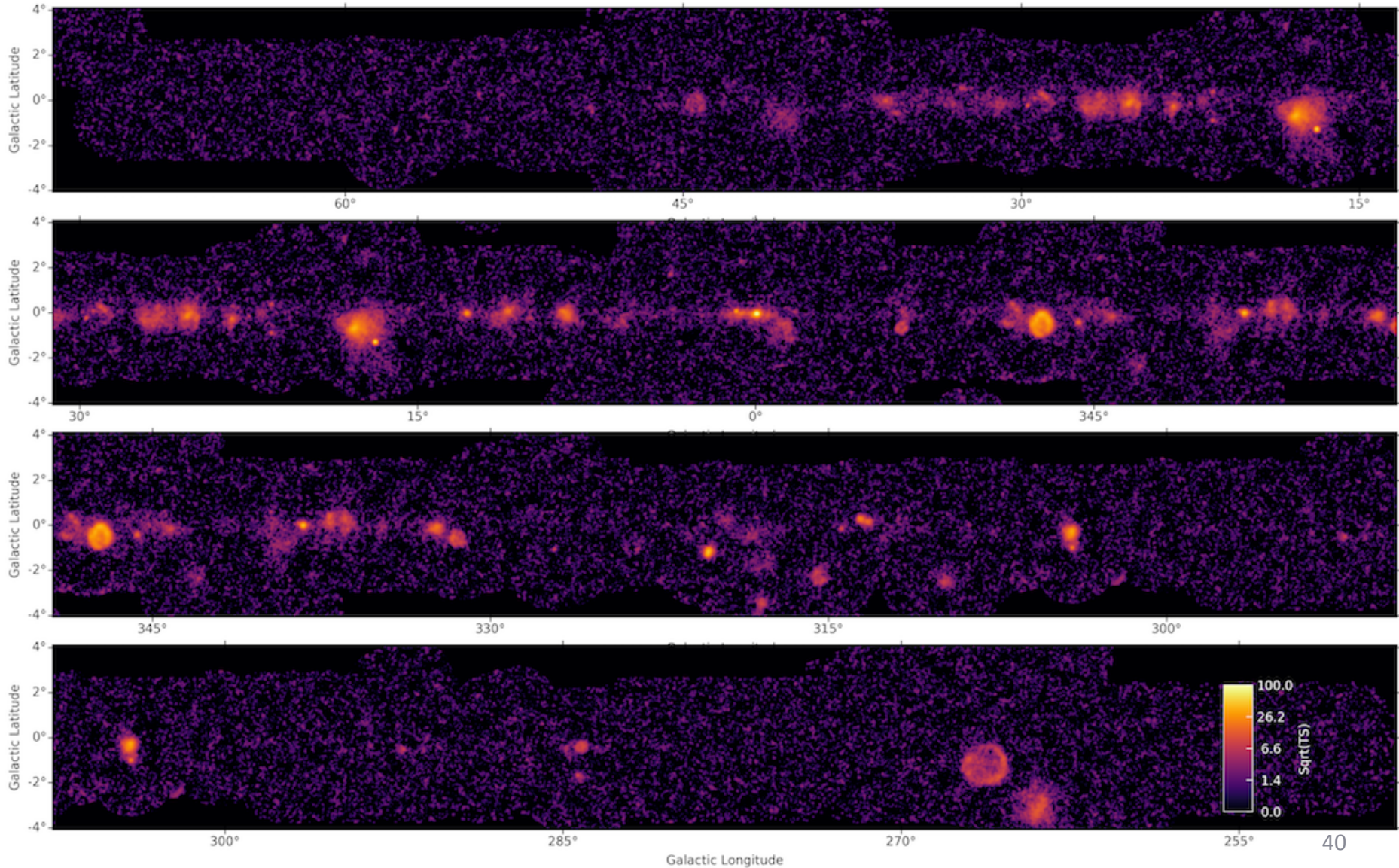
Applying/Nominating

Applications will open soon (target date is Oct 16, 2017). In the meantime, if you are interested in attending and/or have

Contents © 2017 Python in Astronomy - CC-BY-SA Logo Attribution Powered by Nikola

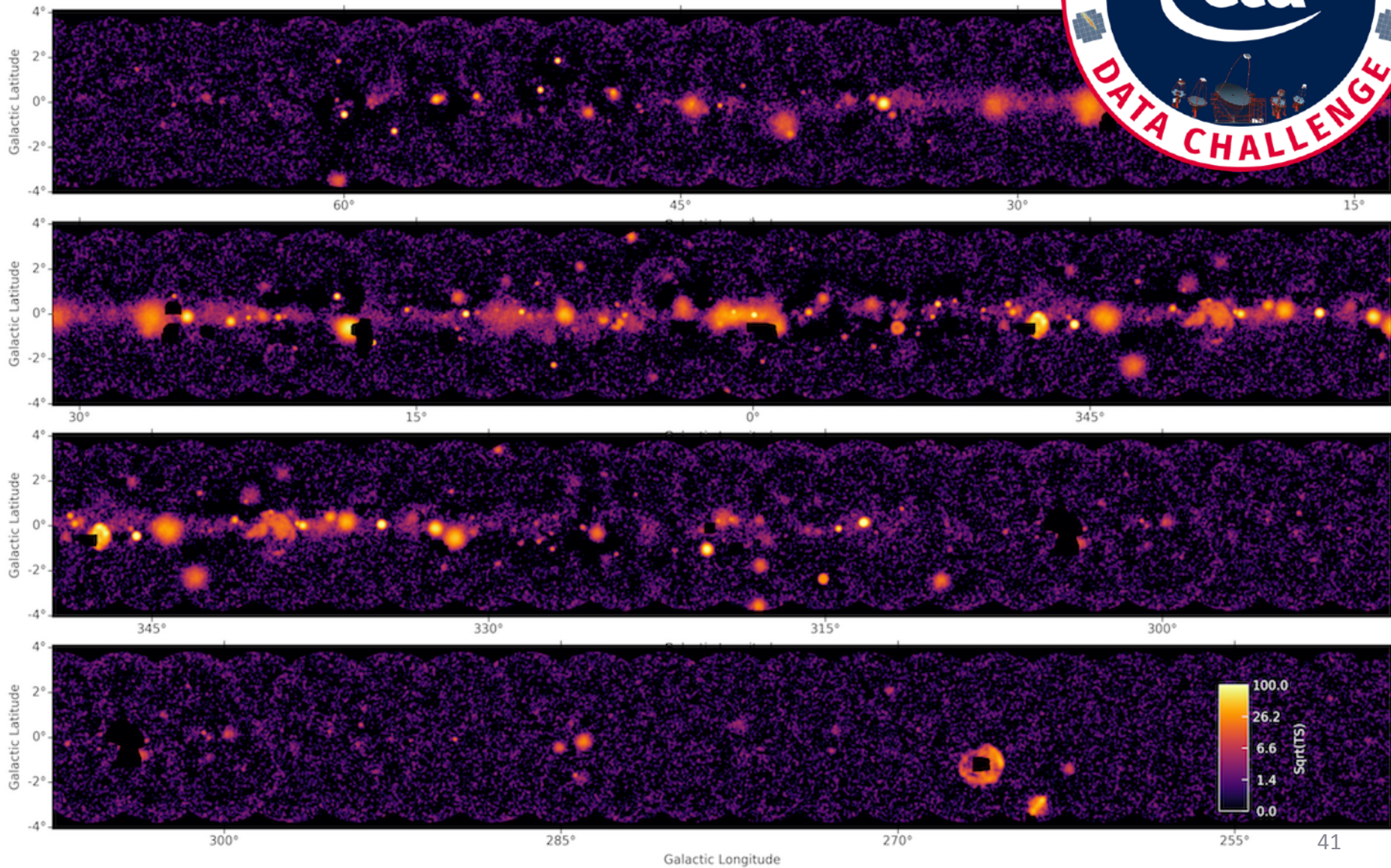
HESS GPS - real TeV sky

This is the official HGPS map, not one re-computed with Gammapy.



CTA 1DC GPS - simulated TeV sky

First quick-look survey map, already outdated, some Gammapy issues fixed last week ...





cherenkov
telescope
array

Gammapy next steps

Some thoughts for Gammapy in 2018



Gammapy next steps

- I don't know what Gammapy will be 5 years from now
- But I think the foundation and philosophy is sound and it's a project worth investing in:
 - a flexible, high-level analysis package for gamma-ray astronomy built on Python, Numpy and Astropy
 - using an open development model (Github)
- For the near future, some key things for Gammapy to focus on:
 - Improve quality & clean up (code, tests, docs)
 - **Rewrite modeling / fitting**
 - Grow the core team and project

Rewrite modeling and fitting



- Currently we mainly use Sherpa for modeling and fitting.
- Sherpa is very nice for what it natively supports: classical 2-dim image and 1-dim spectrum analysis
- Sherpa is very flexible and powerful: Python modelling language, linked parameters, user-defined models in Python, battle-tested optimisers and parameter error estimators
- So far we have mostly tried to wrap or extend Sherpa to our use-cases. This works to a certain extent, we have working examples for 3D analysis or custom likelihood for MWL data (e.g. for radio, X-ray, Fermi-LAT, IACT).
- We could probably go all-in on Sherpa and make it work for all use cases for CTA. But I'm not sure we should, mainly because extending it for our use cases (e.g. energy-dependent PSF, 3D analysis) results in a complex codebase that isn't easy to fully understand and extend for people new to Python / Sherpa. Also, it's not clear if Sherpa will be developed and supported 10 or 20 years from now.

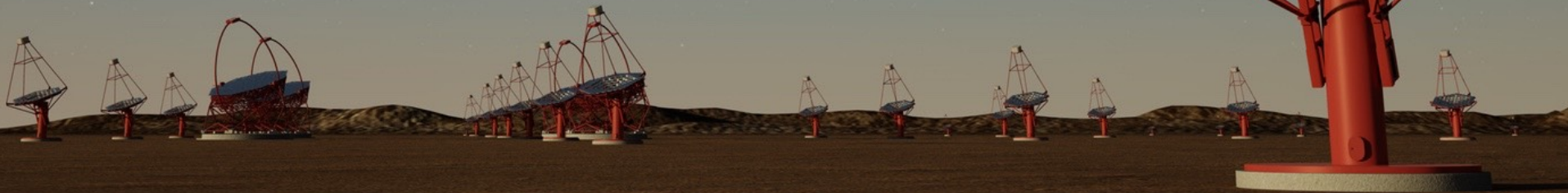
Rewrite modeling and fitting



- So my suggestion for Gammapy is that we continue to implement a simple modeling / likelihood / fitting code, either from scratch (started in `gammapy.utils.modeling`) or using `astropy.modeling` (if we need a flexible Python modeling language and linked parameters).
- The basis is binned analysis using the recently developed `gammapy.maps` by Matthew Wood, which supports efficient analysis of low-count data via multi-resolution and sparse maps (it's clear that this works -> Toby Burnett's pointlike package for Fermi-LAT)
- *To a certain degree it's still not clear what CTA needs (e.g. event types, IRF dependencies), but probably a flexible framework developed now based on our experience from exiting IACTs and Fermi-LAT is sufficient for most use cases, and flexible enough to be extended to all CTA use cases over the next 10 - 20 years.*
- *One nice experiment would be to use Tensorflow (supports all we need, and execution on many CPU / GPU) , or at least one of the Numpy array based automatic differentiation packages, to try efficient gradient-based optimisers. Suggest to do this prototyping outside of Gammapy for now, or in a separate sub-package -> too new / hasn't been proven appropriate for Gammapy (i.e that it works well and is easy to install and use for all).*

Getting started with Gammapy

Install, docs, tutorials



Gammapy installation

- Gammapy works with Python 2.7 and 3.4+ on Linux and Mac
Windows: partly (Sherpa not supported on Windows)
- Stable version:
 - **`pip install gammapy`**
- Binary packages via conda:
 - **`conda install -c conda-forge gammapy`**
- Development version:
 - **`git clone https://github.com/gammapy/gammapy.git`**
`cd gammapy`
`pip install .`
- *If you're not sure how to install Python software, use Anaconda. It's the most-used distribution and usually just works, on Linux, Mac and Windows.*



Gammapy docs



What is Gammapy? — gammapy x Christoph


docs.gammapy.org/en/latest/

gamma^{py}:docs Index Modules Search

gammapy v0.7.dev5068 » next »

Page Contents

- [What is Gammapy?](#)
- [Getting started](#)
- [General documentation](#)
- [The Gammapy toolbox](#)
- [News](#)

 **A Python package for gamma-ray astronomy**

What is Gammapy?

Gammapy is a community-developed, open-source Python package for gamma-ray astronomy.

- Read the [Gammapy documentation](#).
- Ask questions on the [Gammapy mailing list](#).
- Request features, report bugs or contribute on the [Gammapy GitHub page](#).
- Gammapy works with Python 2 and 3, on Linux, Mac OS X and (partly) Windows.

Getting started

Note

Start learning about Gammapy using the [Gammapy tutorial notebooks](#).

Other introductory pages:

To install Gammapy, see the instructions on the [Installation](#) page.

docs.gammapy.org/en/latest/_images/gammapy_banner.png

v: latest 48

Gammapy tutorial notebooks

Notebooks

If you're new to Astro (haven't used `Table`, `SkyCoord` and `Time` much), start here:

- [Astropy introduction for Gammapy users](#)
- [Astropy Hands On \(1st ASTERICS-OBELICS International School\)](#)

For a quick introduction to Gammapy, go here:

- [First steps with Gammapy](#)

Interested to do a first analysis of simulated CTA data?

- [CTA first data challenge \(1DC\) with Gammapy](#)
- [CTA data analysis with Gammapy](#)

To learn how to work with gamma-ray data with Gammapy:

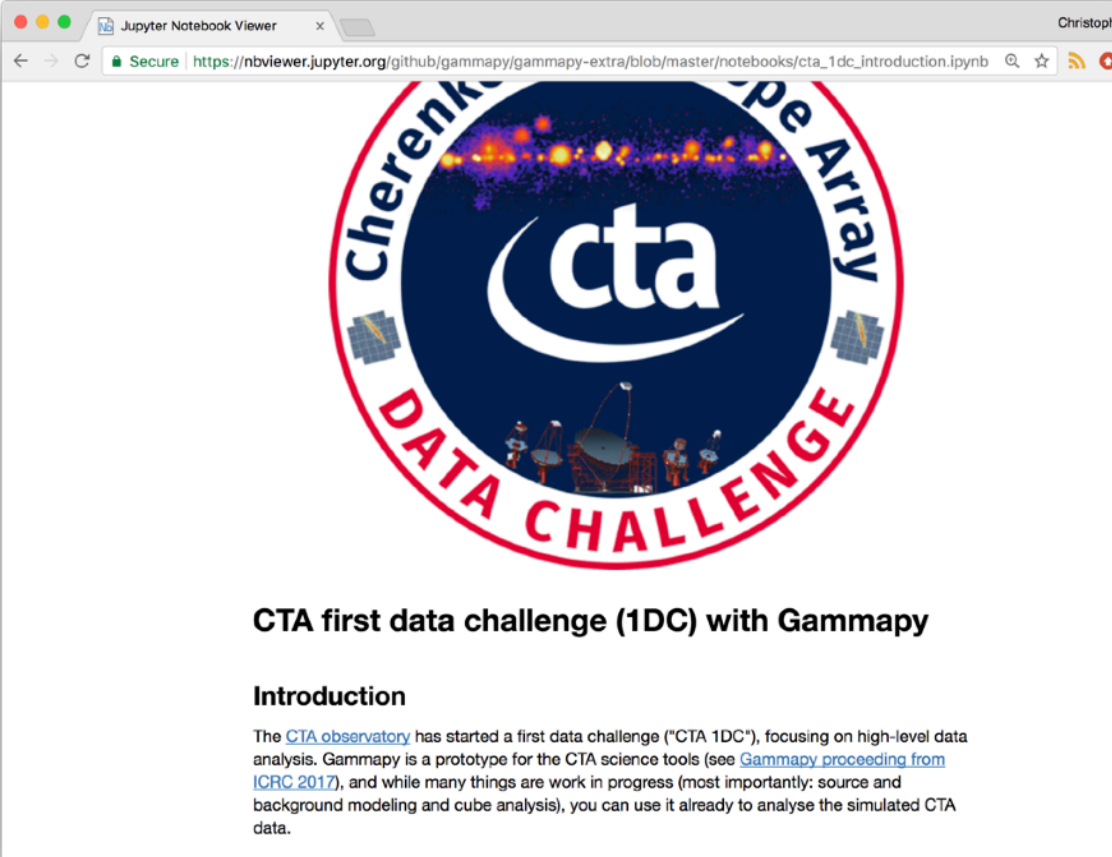
- [IACT DL3 data with Gammapy](#) (H.E.S.S. data example)
- [Fermi-LAT data with Gammapy](#) (Fermi-LAT data example)

Gammapy tutorial for CTA 1DC



First notebook: [cta_1dc_introduction.ipynb](https://nbviewer.jupyter.org/github/gammapy/gammapy-extra/blob/master/notebooks/cta_1dc_introduction.ipynb)

Second notebook: [cta_data_analysis.ipynb](https://nbviewer.jupyter.org/github/gammapy/gammapy-extra/blob/master/notebooks/cta_data_analysis.ipynb)



The screenshot shows a web browser window titled "Jupyter Notebook Viewer" with the URL https://nbviewer.jupyter.org/github/gammapy/gammapy-extra/blob/master/notebooks/cta_1dc_introduction.ipynb. The main content is a circular logo for the "Cherenkov Telescope Array DATA CHALLENGE". The logo features the "cta" logo in the center, surrounded by a red border with the text "Cherenkov Telescope Array" at the top and "DATA CHALLENGE" at the bottom. Below the logo, the text reads "CTA first data challenge (1DC) with Gammapy". Underneath, there is a section titled "Introduction" which states: "The [CTA observatory](#) has started a first data challenge ("CTA 1DC"), focusing on high-level data analysis. Gammapy is a prototype for the CTA science tools (see [Gammapy proceeding from ICRC 2017](#)), and while many things are work in progress (most importantly: source and background modeling and cube analysis), you can use it already to analyse the simulated CTA data." The bottom of the page is partially cut off, showing the start of another sentence: "The main page for CTA 1DC is here: <https://www.cta-observatory.org/cta-1dc/>".

Gammapy references



- Code: <https://github.com/gammapy/gammapy>
- Docs: <http://docs.gammapy.org>
- Tutorials: <https://nbviewer.jupyter.org/github/gammapy/gammapy-extra/blob/master/index.ipynb>

Questions, comments, requests?

- Mailing list: <http://groups.google.com/group/gammapy>
- CTA 1DC: <https://forge.in2p3.fr/projects/data-challenge-1-dc-1/wiki#Sharing-of-analysis-results>

Gammapy tutorial tomorrow

- To prepare for the Gammapy tutorial tomorrow:
 - Install the latest development version of Gammapy
 - Clone the tutorial repository for tomorrow
- If you have any questions or issues, let us know.
(e.g. Bruno, Julien, Régis, me)



cherenkov
telescope
array

Concluding remarks



Concluding remarks

- Python / Numpy / Astropy has become very popular and widely used for astronomical pipelines and science tools
- Gammapy is an Astropy-affiliated package for gamma-ray astronomy. The development philosophy is Python first, build on existing packages, collaborate
- With Gammapy, you can currently do “classical” VHE data analysis (2-dim images & 1-dim spectra), a first working Sherpa-based prototype for 3D analysis exists
- I hope Gammapy will mature from a prototype to become a nice science tool for CTA.
Strengths: simplicity, flexibility, interoperability
- Conclusion: exciting times ahead for CTA and Gammapy!