Astropy, Sherpa and Gammapy

Axel Donath

PyGamma15 Workshop – Nov. 16th 2015

astropy

A Community Python Library for Astronomy

What is the Astropy Project?

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015

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

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015

Core data structures and transformations

- Constants (**`astropy.constants`**)
- Units and Quantities (**`astropy.units`**)
- N-dimensional datasets (**`astropy.nddata`**)
- Data Tables (**`astropy.table`**)
- Time and Dates (**`astropy.time`**)
- Astronomical Coordinate Systems (**`astropy.coordinates`**)
- World Coordinate System (**`astropy.wcs`**)
- Models and Fitting (**`astropy.modeling`**)
- Analytic Functions (**`astropy.analytic_functions`**)

Connecting up: Files and I/O

- Unified file read/write interface
- FITS File handling (**`astropy.io.fits`**)
- ASCII Tables (**`astropy.io.ascii`**)
- VOTable XML handling (**`astropy.io.votable`**)
- Miscellaneous Input/Output (**`astropy.io.misc`**)

Astronomy computations and utilities

- Convolution and filtering (**`astropy.convolution`**)
- Data Visualization (**`astropy.visualization`**)
- Cosmological Calculations (**`astropy.cosmology`**)
- Astrostatistics Tools (**`astropy.stats`**)
- Virtual Observatory Access (**`astropy.vo`**)

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015

Astropy Core Package

4 major public releases (first release February 2013)

Latest stable version: **v1.0.6** (released 22nd October 2015)

Over **150** individual contributors so far!

Almost **14,000** commits (as of 1st November 2015)

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015

Project Coordinators: Perry Greenfield (STScI), Thomas Robitaille (MPIA), Erik Tollerud (Yale)

Developers/Contributors for core package (as of 1st Nov 2015):

- Ryan Abernathey
- Shailesh Ahuja
- Tom Aldcroft
- Anne Archibald
- Cristian Ardelean
- Matteo Bachetti
- Kyle Barbary
- Geert Barentsen
- Pauline Barmby
- Paul Barrett
- Andreas Baumbach
- Chris Beaumont
- Daniel Bell
- Kristin Berry
- Francesco Biscani
- Thompson Le Blanc
- Christopher Bonnett
- Joseph Jon Booker
- Médéric Boquien
- Azalee Bostroem
- Matthew Bourque
- Larry Bradley
- Gustavo Bragança
- Erik M. Bray
- Eli Bressert
- Hannes Breytenbach
- Hugo Buddelmeijer
- Doug Burke
- Mihai Cara
- Patti Carroll
- Mabry Cervin
- Pritish Chakraborty
- Alex Conley
- Jean Connelly
- Simon Conseil
- Ryan Cooke
- Yannick Copin
- Matthew Craig
- Steven Crawford
- Neil Crighton
- Kelle Cruz
- Daniel Datsev
- Matt Davis
- Christoph Deil
- Nadia Dencheva
- Jörg Dietrich
- Axel Donath
- Michael Droettboom
- Zach Edwards
- Jonathan Eisenhamer
- Thomas Erben
- Henry Ferguson
- Jonathan Foster
- Ryan Fox
- Lehman Garrison
- Simon Gibbons
- Adam Ginsburg
- Christoph Gohlke
- Danny Goldstein
- Perry Greenfield
- Dylan Gregersen
- Austen Groener
- Frédéric Grollier
- Karan Grover
- Kevin Gullikson
- Hans Moritz Günther
- Alex Hagen
- Paul Hirst
- Moataz Hisham
- Michael Hoenig
- Emma Hogan
- Derek Homeier
- Chris Hanley
- JC Hsu
- Anthony Horton
- Eric Jeschke
- Joseph Jon Booker
- Sarah Kendrew
- Marten van Kerkwijk
- Wolfgang Kerzendorf
- Lennard Kiehl
- Rashid Khan
- Dominik Klaes
- Kacper Kowalik
- Roban Hultman Kramer
- Arne de Laat
- Antony Lee
- Simon Liedtke
- Pey Lian Lim
- Joseph Long
- Joe Lyman
- Vinayak Mehta
- Aaron Meisner
- Serge Montagnac
- José Sabater Montes
- Brett Morris
- Michael Mueller
- Stuart Mumford
- Demitri Muna
- Prasanth Nair
- Bogdan Nicula
- Asra Nizami
- Joe Philip Ninan
- Bryce Nordgren
- Miruna Oprescu
- Carl Osterwisch
- Luigi Paioro
- Asish Panda
- Madhura Parikh
- Neil Parley
- Sergio Pascual
- Rohit Patil
- David Perez-Suarez
- Ray Plante
- Orion Poplawski
- Adrian Price-Whelan
- J. Xavier Prochaska
- David Pérez-Suárez
- Tanuj Rastogi
- Thomas Robitaille
- Juan Luis Cano Rodríguez
- Evert Rol
- Alex Rudy
- Joseph Ryan
- Eloy Salinas
- Gerrit Schellenberger
- David Shiga
- Albert Y. Shih
- David Shupe
- Jonathan Sick
- Leo Singer
- Brigitta Sipocz
- Shivan Sornarajah
- Shantanu Srivastava
- Ole Streicher
- Matej Stuchlik
- Bernardo Sulzbach
- James Taylor
- Jeff Taylor
- Kirill Tchernyshyov
- Víctor Terrón
- Scott Thomas
- Erik Tollerud
- James Turner
- Jake VanderPlas
- Miguel de Val-Borro
- Jonathan Whitmore
- Julien Woillez
- Lisa Walter
- Benjamin Alan Weaver
- Jonathan Whitmore
- Julien Woillez
- Víctor Zabalza

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015

Astropy-affiliated packages

Image reprojection

Publication-quality image plotting

Machine learning

Photometry

Interface to many web services/archives

Gamma-ray data analysis

CCD image reduction

Interfaces to ds9

Spectroscopic analysis

'Big' spectral cube analysis (e.g. ALMA, etc.)

Spectral cube slicing

<your package here!>

etc.

From T.Robitaille, "Astropy and the Open Source Revolution in Astronomy", Talk @ .Astronomy 2015


Sherpa

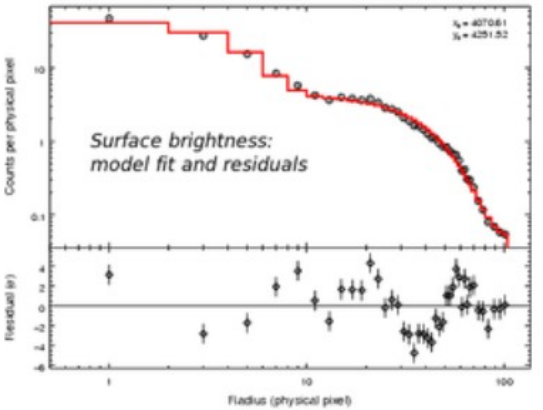
What is Sherpa?

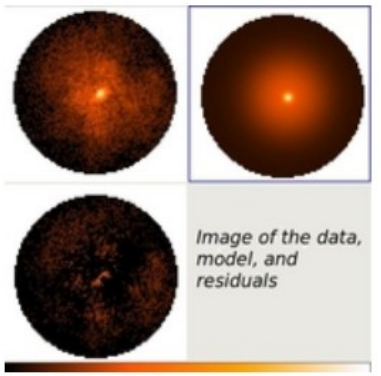
- Ciao's modelling and fitting package

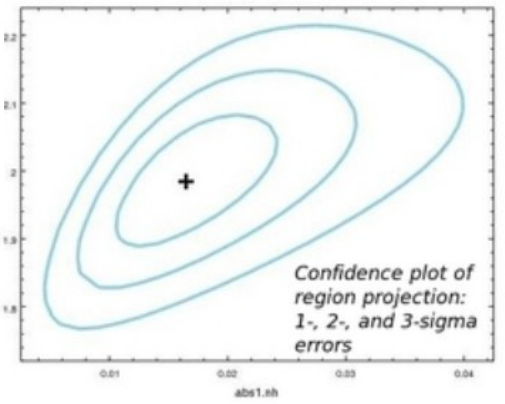
CHANDRA
X-RAY OBSERVATORY

Sherpa lets you:

- *fit 1-D data sets (simultaneously or individually), including: spectra, surface brightness profiles, light curves, general ASCII arrays;*
- *fit 2-D images/surfaces in the Poisson/Gaussian regime;*
- *access the internal data arrays;*
- *build complex model expressions;*
- *import and use your own models;*
- *choose appropriate statistics for modeling Poisson or Gaussian data;*
- *import new statistics, with priors if required by analysis;*
- *visualize a parameter space with simulations or using 1-D/2-D cuts of the parameter space;*
- *calculate confidence levels on the best-fit model parameters;*
- *choose a robust optimization method for the fit: Levenberg-Marquardt, Nelder-Mead Simplex or Monte Carlo/Differential Evolution;*
- ...







Excerpt from http://cxc.harvard.edu/sherpa/

Sherpa API

High level session API:

```
sherpa> load_image("img.fits")
sherpa> show_data()
sherpa>
sherpa> contour_data()
sherpa> print_window("contour_plot")
sherpa>
sherpa> load_table_model("emap", "expmap.fits")
sherpa> print(emap)
sherpa>
sherpa> set_model(beta2d.b1*emap)
sherpa> show_model()
sherpa>
sherpa> b1.r0 = 30
sherpa> b1.xpos = 40
sherpa> b1.ypos = 40
sherpa> b1.ellip = 0.3
sherpa> b1.theta = 5
sherpa> b1.ampl = 3.0
sherpa> b1.alpha = 1.5
sherpa>
sherpa> thaw(b1.ellip, b1.theta)
sherpa> freeze(emap.ampl)
```

Very nicely documented, with lots of examples, tutorials etc.
→ good for users!

Class API:

```
53
54  def fit_powerlaw_sherpa(x, y, xmin=FLUX_MIN, xmax=FLUX_MAX):
55      """Fit powerlaw using the sherpa."""
56      from sherpa.data import Data1DInt
57      from sherpa.models import PowLaw1D
58      from sherpa.stats import Cash
59      from sherpa.optmethods import LevMar
60      from sherpa.fit import Fit
61      from sherpa.estmethods import Covariance
62      data = Data1DInt('Dataset 1', x[:-1], x[1:], y)
63      data.notice(xmin, xmax)
64      pl = PowLaw1D('pl_sherpa')
65      pl.ref.val = xmin
66
67      f = Fit(data, pl, Cash(), LevMar(), Covariance())
68      f.fit()
69
70      # estimate errors
71      errors = f.est_errors()
72      return pl, errors.extra_output
73
```

Almost not documented
→ not so good for developers, but...

…good news from Sherpa

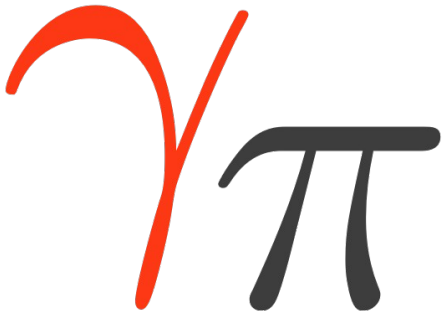
NEW On **April 20, 2015** Sherpa became an Open Source project with the Sherpa 4.7 source code repository placed on GitHub. The complete tar files are available for download as well as the full project repository which can be 'cloned'. The Sherpa Project welcomes contributions from the users via GitHub. Check the details on the Sherpa for Python web page.

Sherpa on GitHub: https://github.com/sherpa/sherpa

- Sherpa is now open source, even open developed, contributions via GitHub are welcomed…

Open issues (being worked on by the developers…):

- Sherpa not yet compatible with Python 3
- Class API not yet documented



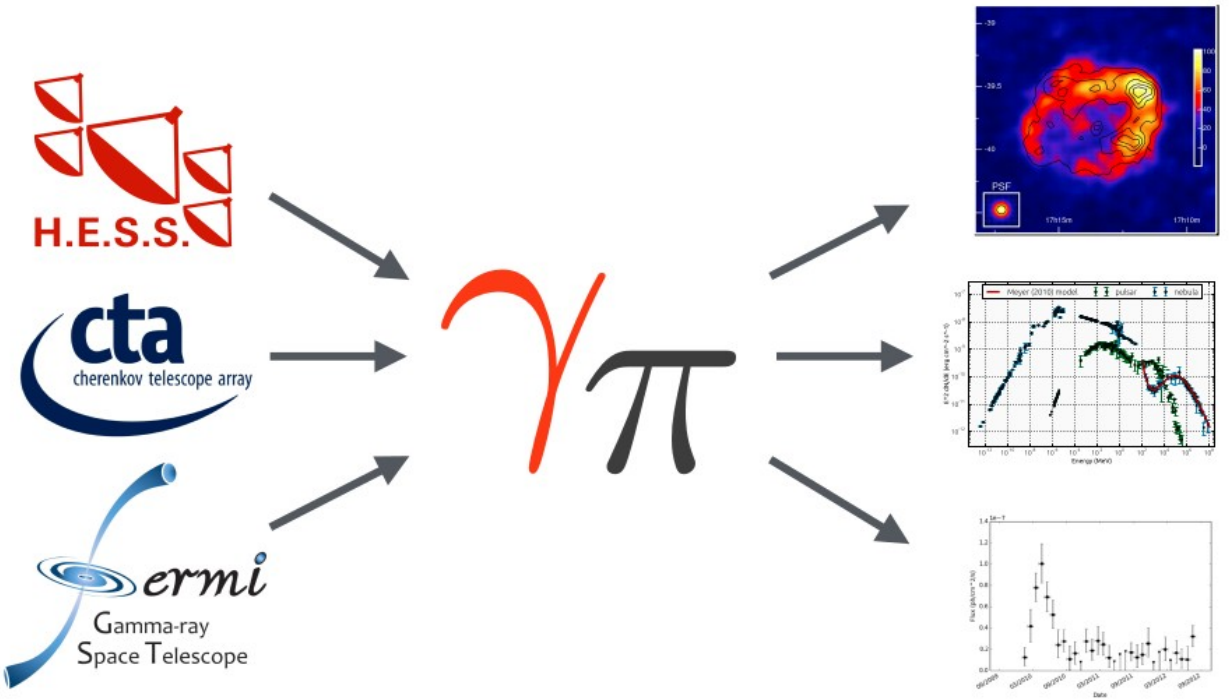
A **Python** package for
gamma-ray astronomy

What is Gammapy?

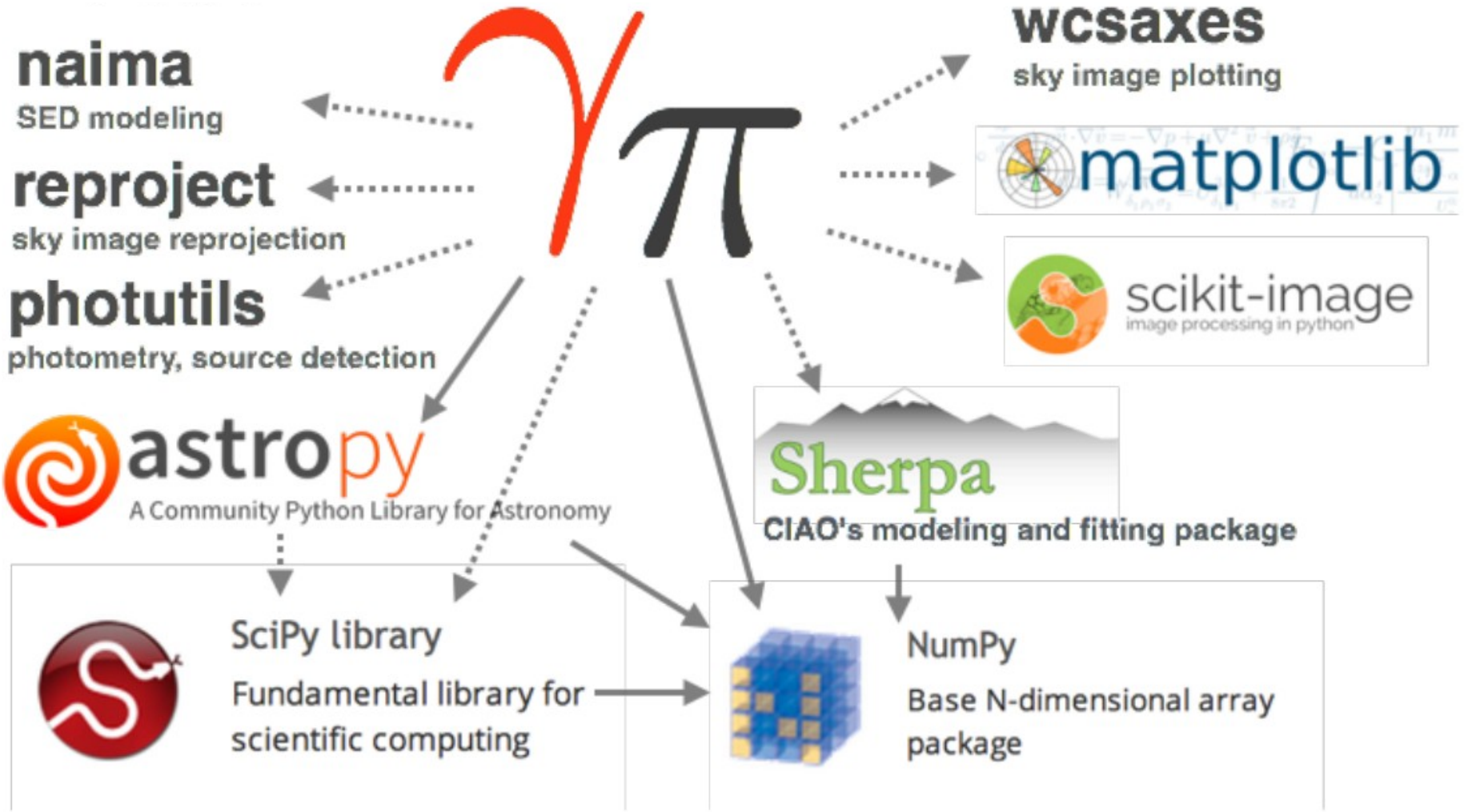
- Code: https://github.com/gammapy/gammapy
- Docs: https://gammapy.readthedocs.org/

Gammapy is an open source (BSD licensed) gamma-ray astronomy Python package.

It is an in-development affiliated package of Astropy that builds on the core scientific Python stack to provide tools to simulate and analyse the gamma-ray sky for telescopes such as CTA, H.E.S.S., and Fermi.



Gammapy dependencies



Gammapy development

Using the open-source and Python development tools ….

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

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

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

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

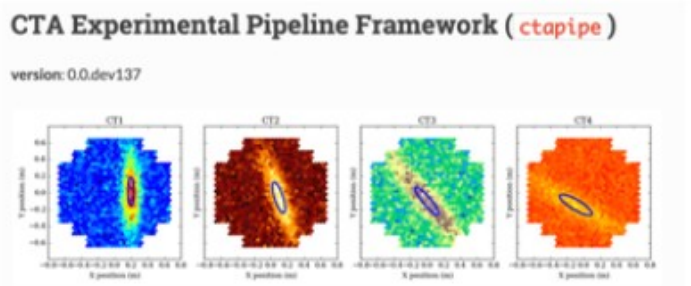
Binary cross-platform package manager.
Install Gammapy and all dependencies on any
Linux & Mac box in $HOME in 10 min.

Gammapy scope

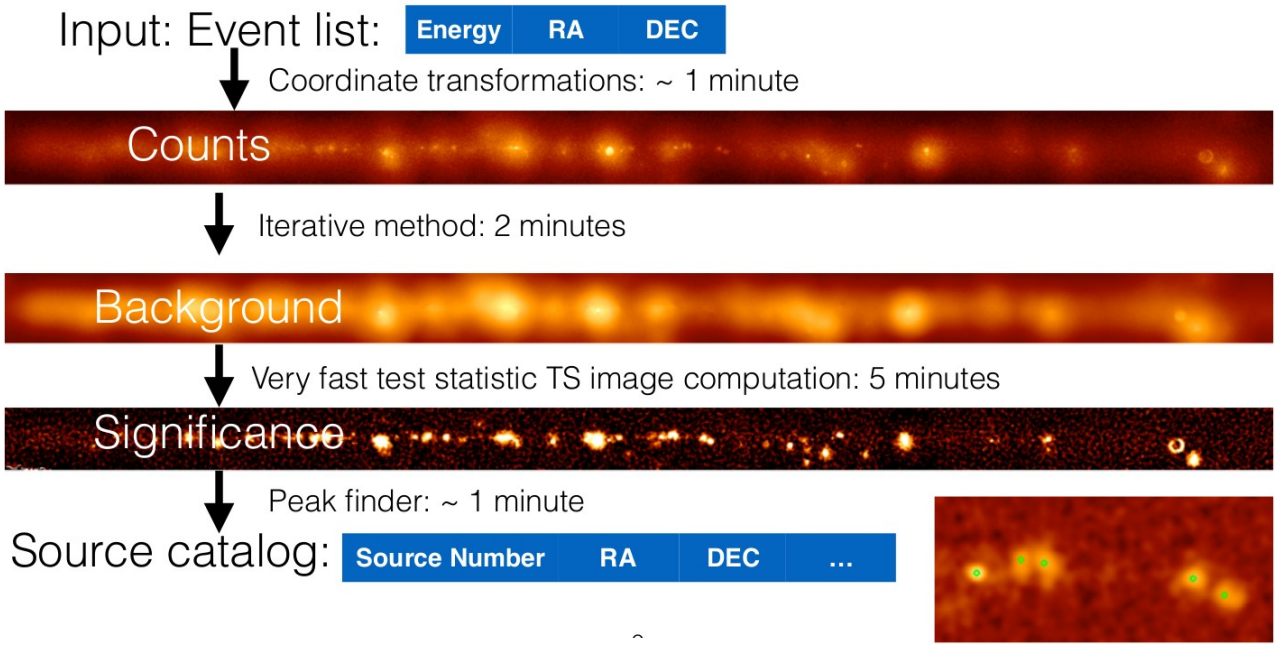
- Gammapy started as a repo where Axel an I share scripts for H.E.S.S. image analysis (source detection and morphology fitting in HGPS)
- Recently most work was on H.E.S.S. data, IRF, observation handling, background modeling and 1D spectral analysis.
- Future developments depend on who wants and has time to contribute. (Most people develop their own scripts or contribute to the internal H.E.S.S. software or other projects).

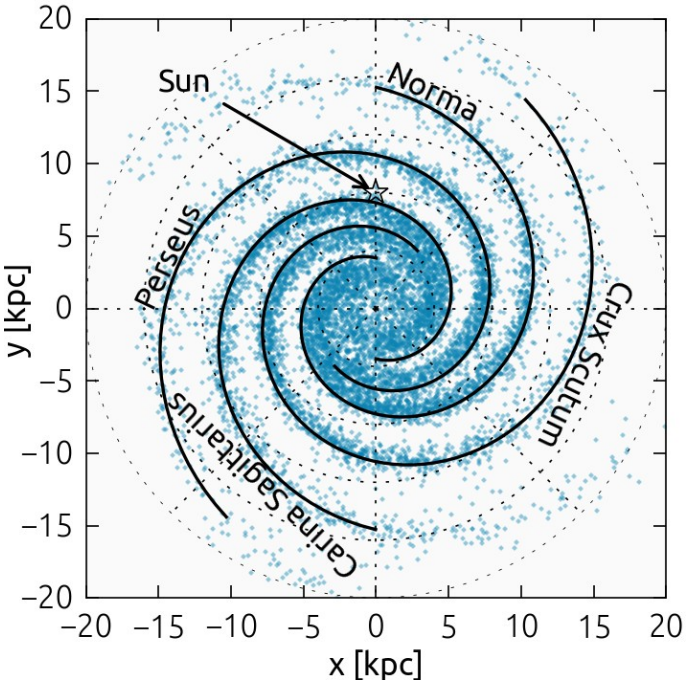
- Command line tools (`gammapy.scripts`)
- Astrophysical source and population models (`gammapy.astro`)
- Background estimation and modeling (`gammapy.background`)
- Catalog (`gammapy.catalog`)
- Data classes (`gammapy.data`)
- Access datasets (`gammapy.datasets`)
- Source detection tools (`gammapy.detect`)
- Image processing and analysis tools (`gammapy.image`)
- Instrument response function (IRF) functionality (`gammapy.irf`)
- Morphology and PSF methods (`gammapy.morphology`)
- Observation handling (`gammapy.obs`)
- Spectrum estimation and modeling (`gammapy.spectrum`)
- Statistics tools (`gammapy.stats`)
- Time handling and analysis (`gammapy.time`)
- Utility functions and classes (`gammapy.utils`)

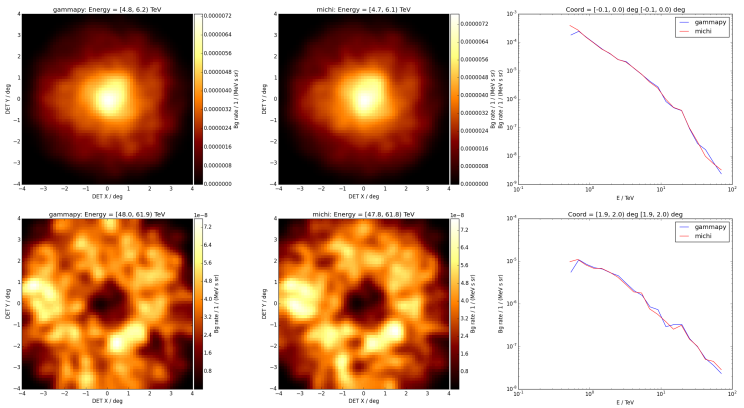
(gammapy.shower has been removed, because there's now ctapipe.)

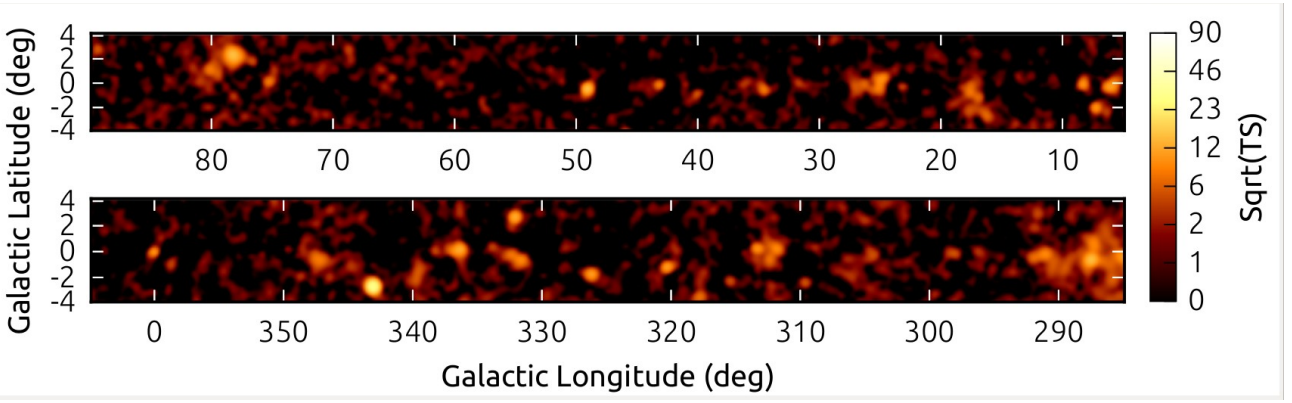


Analysis examples









More examples will be shown in the Gammapy tutorial Tuesday 4 pm!

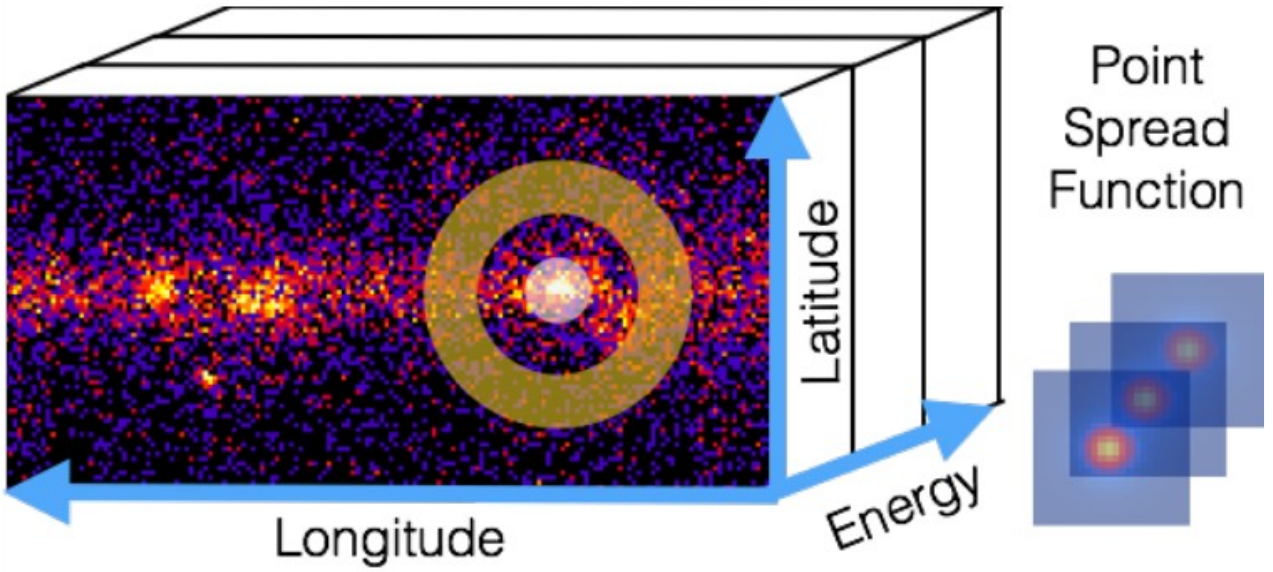
Next steps for Gammapy

- Polish and document how to use the existing 2D image and 1D spectrum analysis tools

- Implement 3D likelihood fitting (We're in contact with the Sherpa developers...)

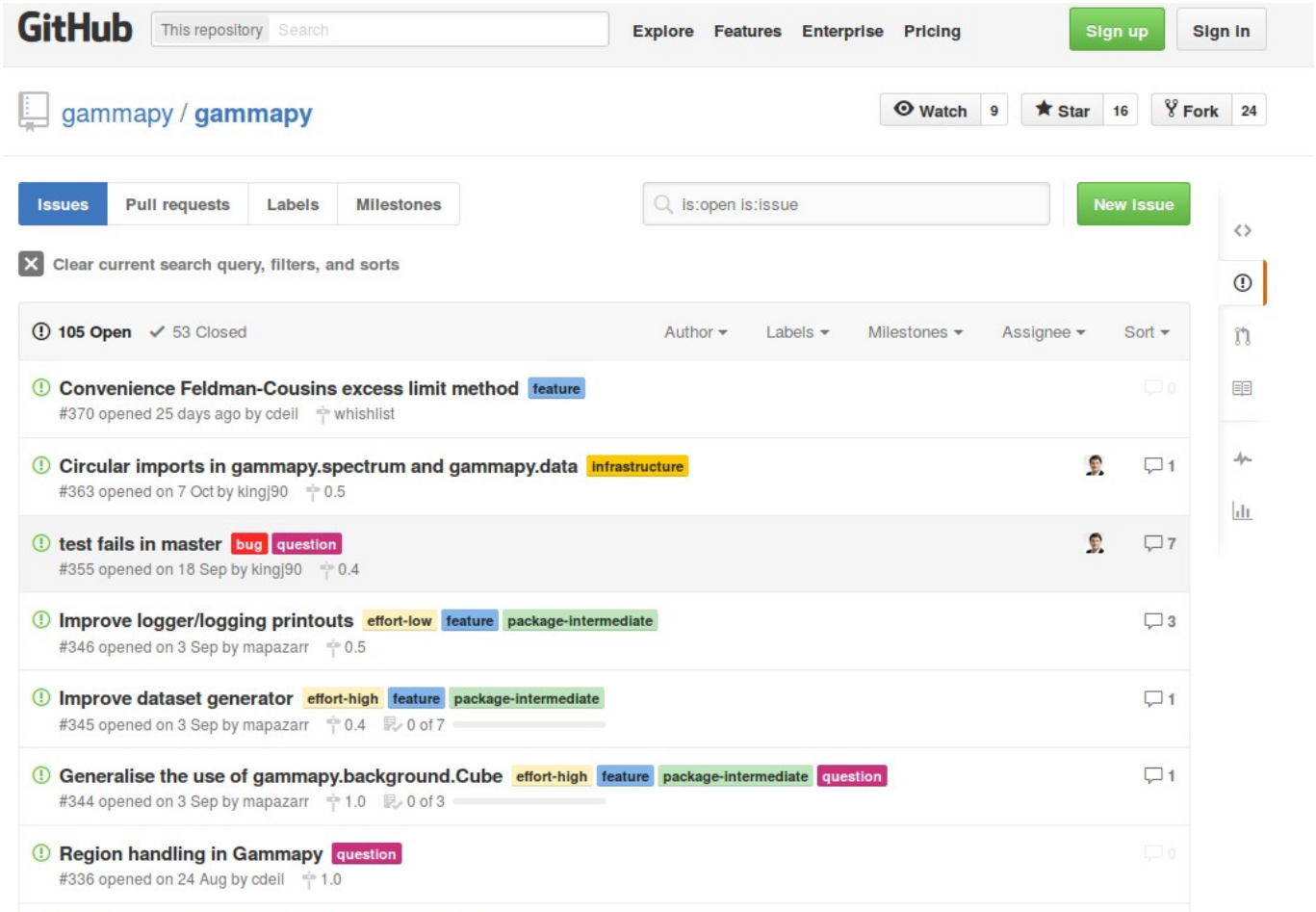
- Enable joint analysis e.g. Fermi – H.E.S.S.

- Integrate Naima SED modeling



.

Contributing to Gammapy



There are a lot of possibilities to contribute...and this week is a great opportunity to start! Just talk to us, any contribution is welcome!