

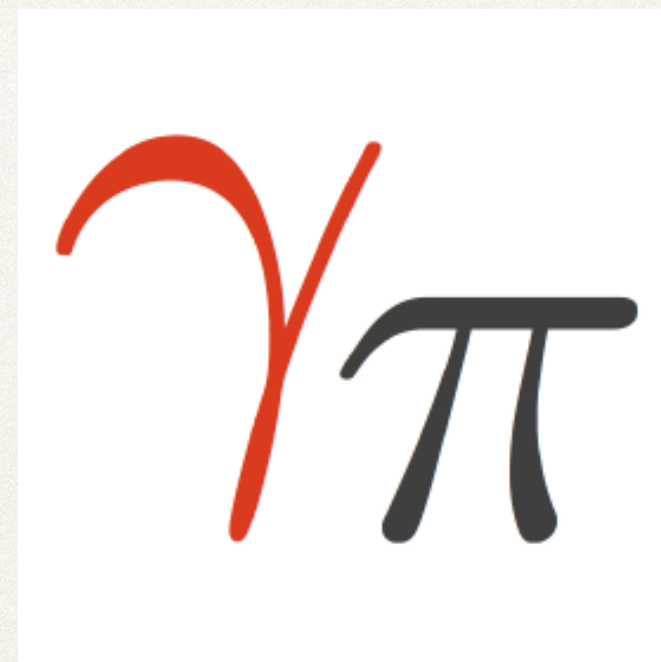
Gammapy: An open source python package for gamma-ray astronomy



Atreyee Sinha
IPARCOS/UCM, Madrid

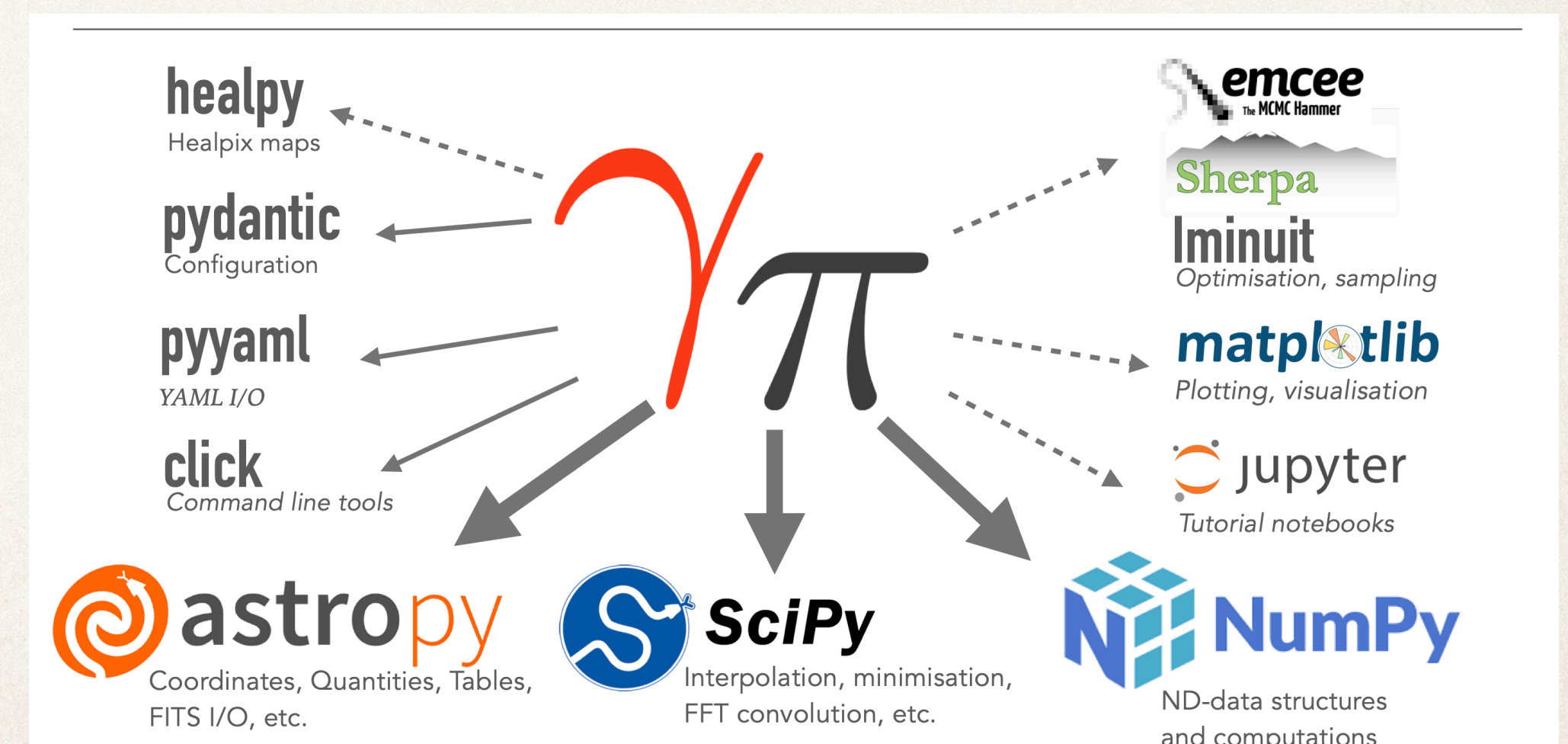
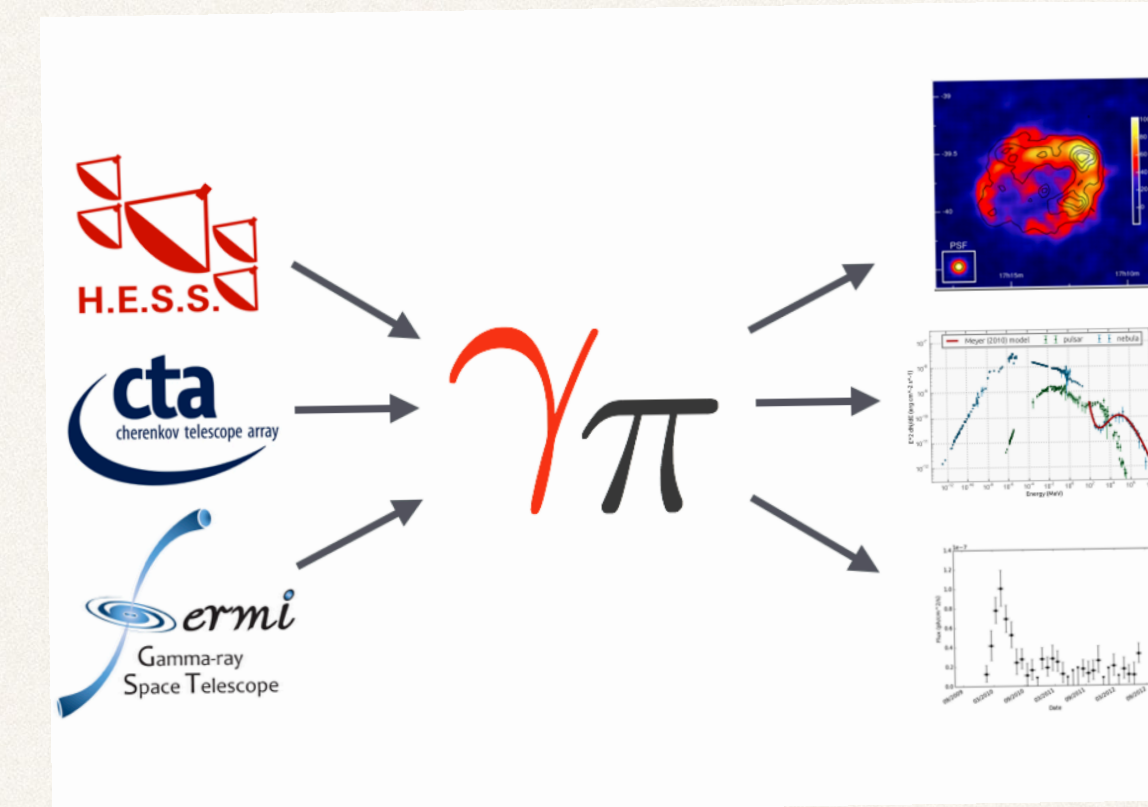


R. Terrier, A. Donath, B. Khelifi, L. Giunti, L. O. Nieto, M. Noethe, C. Nigro et al for the Gammapy dev team



The gammapy concept

- ❖ Flexible, open source, community driven python library
- ❖ Embedded in the python ecosystem
- ❖ Based on common data formats (defined in the GADF) [See: https://gamma-astro-data-formats.readthedocs.io/en/v0.2/](https://gamma-astro-data-formats.readthedocs.io/en/v0.2/)
- ❖ Library for the CTA science tools (and used by many other experiments like MAGIC, H.E.S.S., etc)

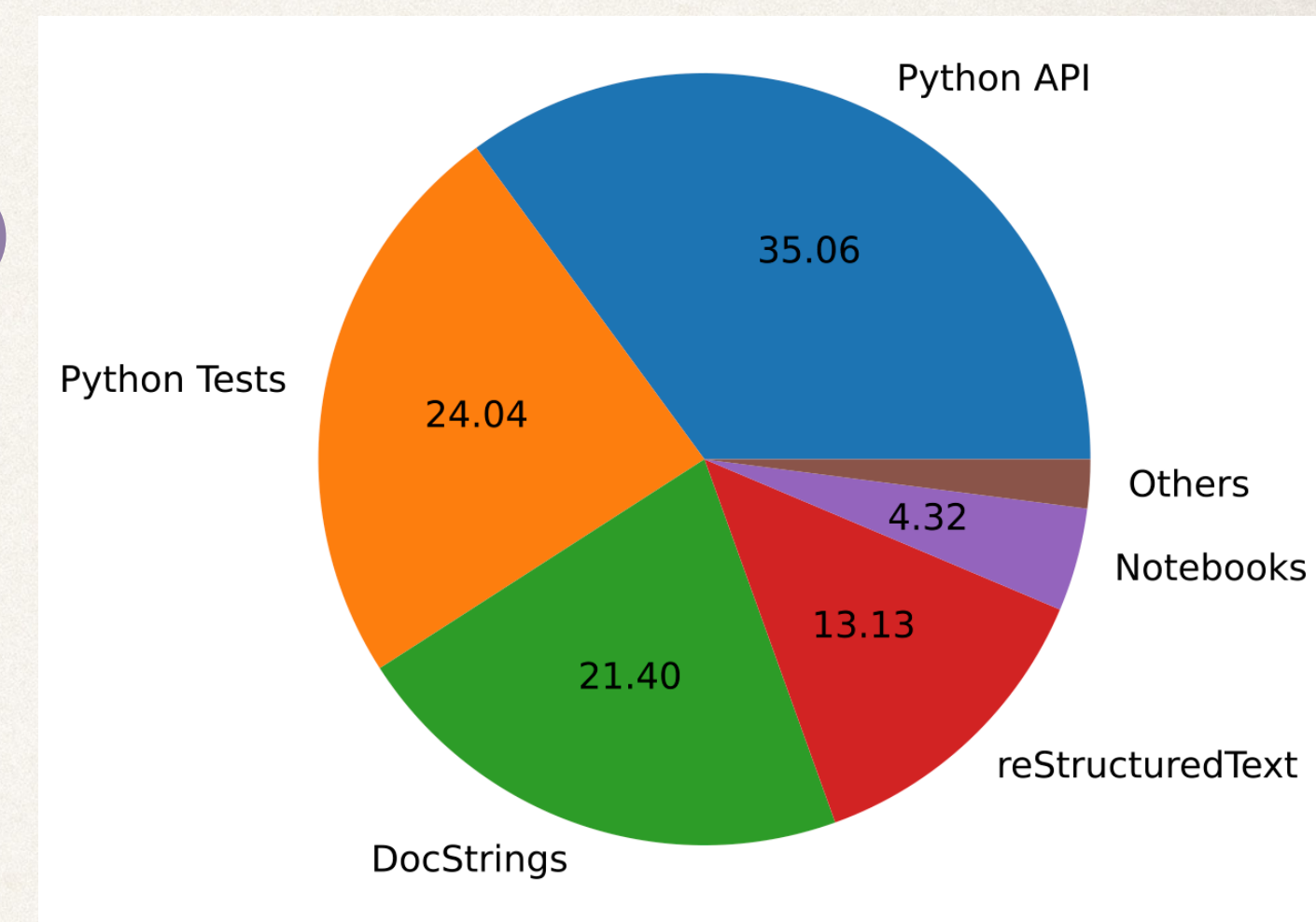
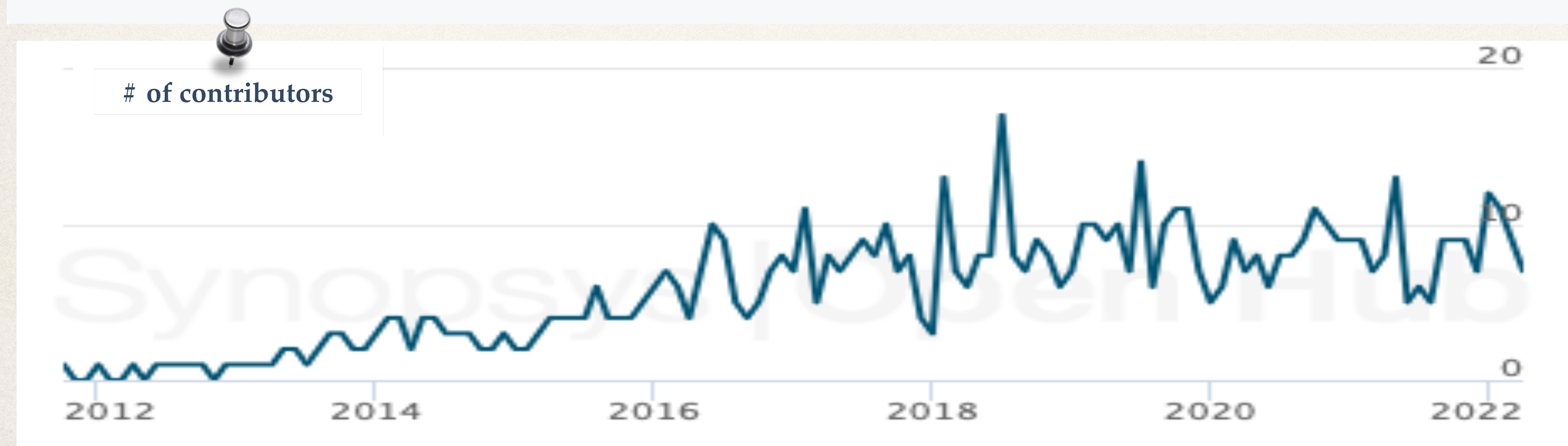
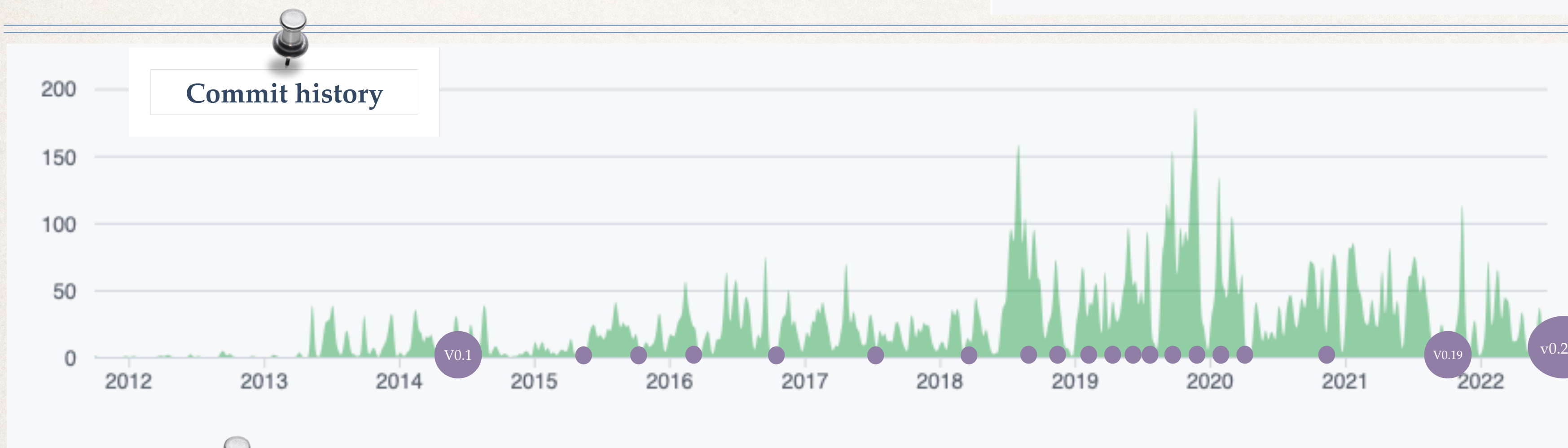


Development history

Total Lines : 93,063
Number of Languages : 3

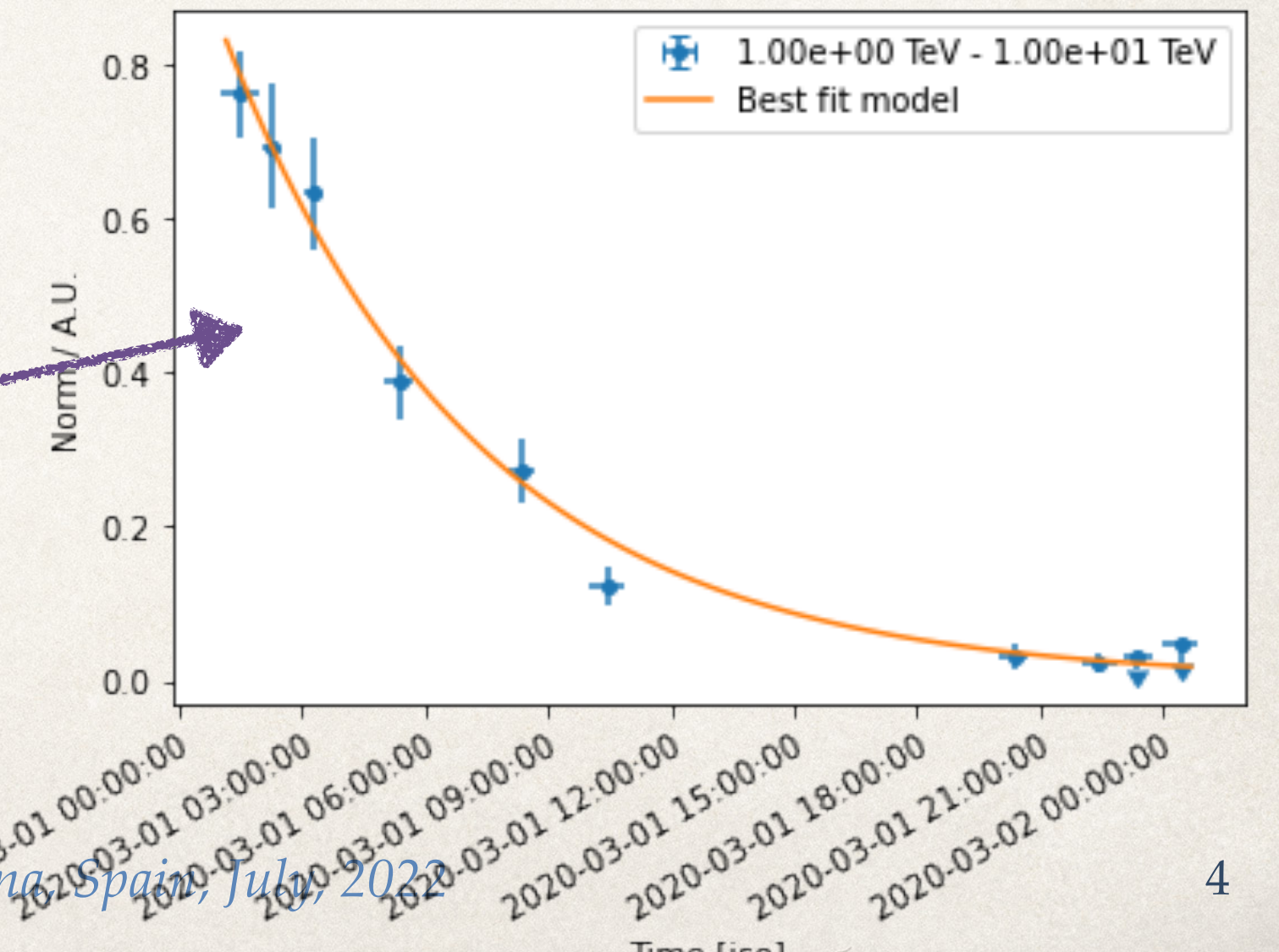
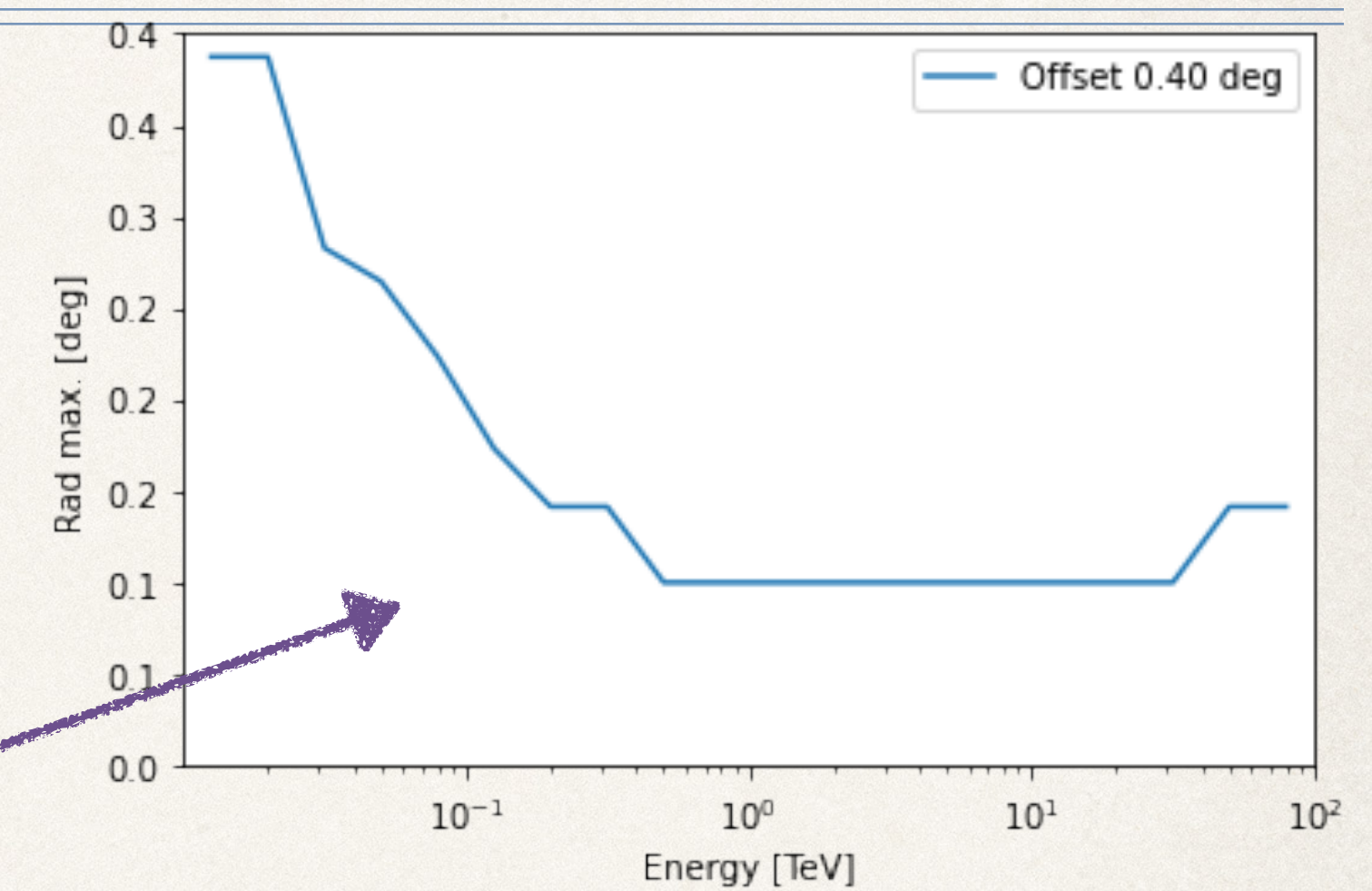
Code Lines : 54,420
Total Comment Lines : 20,155
Total Blank Lines : 18,488

Percent Code Lines : 58.5%
Percent Comment Lines : 21.7%
Percent Blank Lines : 19.9%



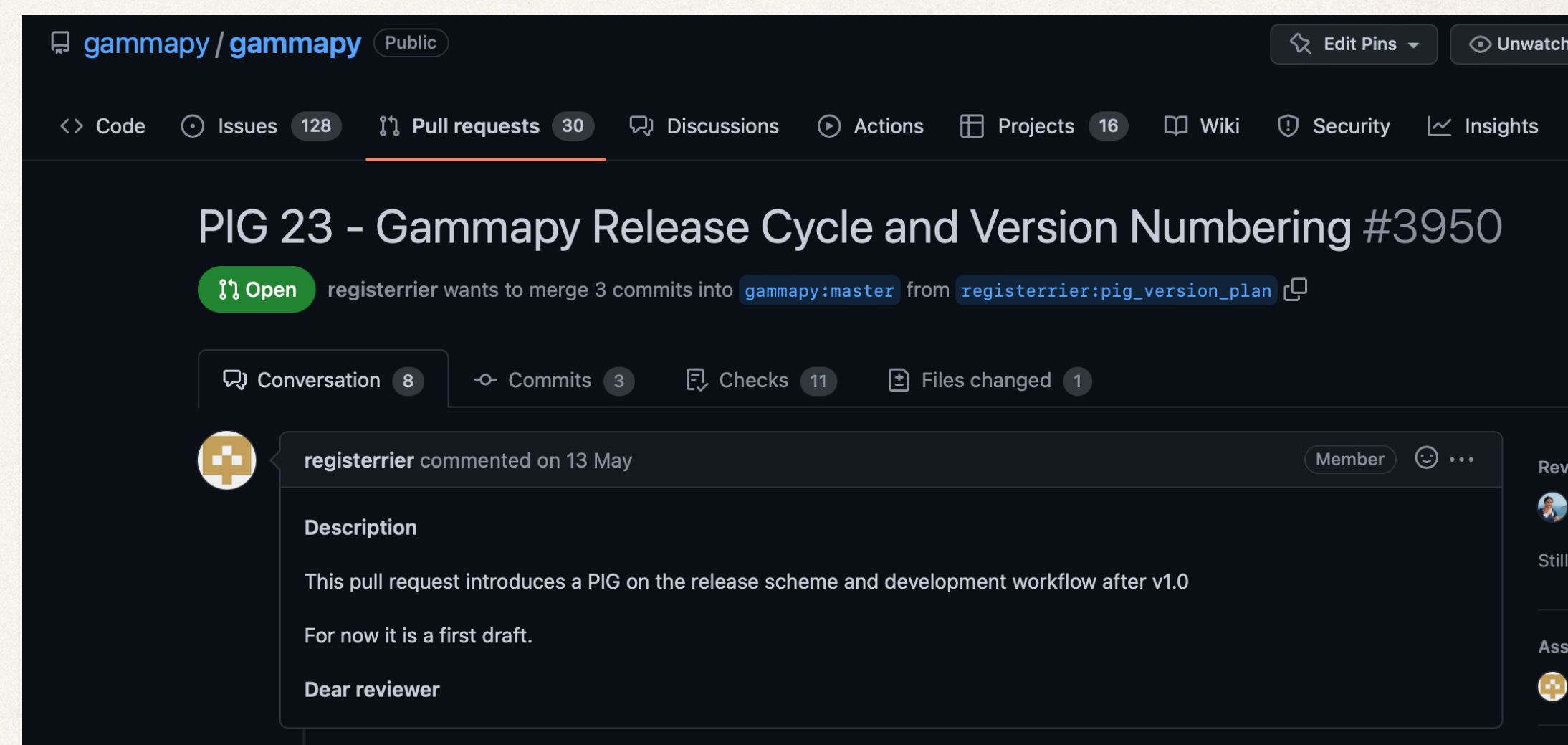
What's new in the latest version?

- ❖ Features:
 - ❖ A new documentation theme
 - ❖ Updated dependencies
 - ❖ Support for energy dependent on-region spectral extraction:
 - ❖ Used by the MAGIC and the LST-1 for point-like analysis
 - ❖ Support for fitting temporal models to light curves
- ❖ Bug fixes...



Towards a long term stable release

- ❖ Present development: Regular feature releases with added functionalities and bug fixes
 - ❖ Not backwards compatible
 - ❖ Urgent cases: Bug fix releases within ~2 week
- ❖ Future plans
 - ❖ A stable API with relevant bug fix releases
 - ❖ Feature releases at regular intervals
 - ❖ Requires planning and dedicated maintenance
 - ❖ **LTS version 1.0** scheduled for release later this year



PIGs: Proposals for
Improvement of Gammapy



Getting started with gammapy

The screenshot shows the Gammapy documentation website. At the top, a dark navigation bar contains the Gammapy logo (a red pi symbol) and a menu with items: Getting started, User guide, Tutorials, API reference, Developer guide, Release notes, and a version dropdown set to 0.20.1. Social media icons for GitHub, Twitter, and a general share icon are also present. Below the navigation bar is a search box labeled 'Search the docs ...'. On the left side, a sidebar menu lists: Gammapy analysis workflow and package structure, How To, Model gallery, Gammapy recipes, and Glossary and references. The main content area is titled 'User guide' and features four large cards: 'Analysis workflow and package structure' (with a box icon), 'How To' (with a chevron icon), 'Model gallery' (with a landscape icon), and 'Gammapy recipes' (with a block letter icon). Each card includes a brief description and a dark button to navigate to the respective page. Callout boxes with arrows point to various elements: 'Installation instructions' points to the navigation bar; 'Search box' points to the search input; 'Using gammapy' points to the 'How To' card; 'Switch between versions' points to the version dropdown; 'Links to get in touch' points to the social media icons; 'Explanations on how the Package works' points to the 'Analysis workflow and package structure' card; and 'User contributed notebooks' points to the 'Gammapy recipes' card.

Installation instructions

Search box

Explanations on how the Package works

See: <https://docs.gammapy.org/>

Using gammapy

Switch between versions

Links to get in touch

User contributed notebooks

How To for quick look ups

- ❖ HowTos for specific use cases
- ❖ Links directly to the code block in the tutorials
- ❖ We welcome your suggestions!

How To

This page contains short “how to” or “frequently asked question” entries for Gammapy. Each entry is for a very specific task, with a short answer, and links to examples and documentation.

If you're new to Gammapy, please check the [Getting started](#) section and the [User guide](#) and have a look at the list of [Tutorials](#). The information below is in addition to those pages, it's not a complete list of how to do everything in Gammapy.

Please give feedback and suggest additions to this page!

+ Spell and pronounce Gammapy

+ Select observations

[Straight to tutorial...](#)

+ Make an on-axis equivalent livetime map

[Straight to tutorial...](#)

+ Check IRFs

[Straight to tutorial...](#)

- Choose units for plotting

Units for plotting are handled with a combination of [matplotlib](#) and [astropy.units](#). The methods `ax.xaxis.set_units()` and `ax.yaxis.set_units()` allow you to define the x and y axis units using [astropy.units](#). Here is a minimal example:

```
import matplotlib.pyplot as plt
from gammapy.estimators import FluxPoints
from astropy import units as u

filename = "$GAMMAPY_DATA/hawc_crab/HAWC19_flux_points.fits"
fp = FluxPoints.read(filename)

ax = plt.subplot()
ax.xaxis.set_units(u.eV)
ax.yaxis.set_units(u.Unit("erg cm-2 s-1"))
fp.plot(ax=ax, sed_type="e2dnde")
```


Detailed workflow in the tutorials

Tutorials

This page lists the Gammapy tutorials that are available as [Jupyter notebooks](#). You can read them here, or execute them using a temporary cloud server in [Binder](#).

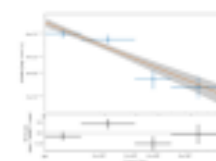
To execute them locally, you have to first install Gammapy locally (see [Installation](#)) and download the tutorial notebooks and example datasets (see [Getting started](#)). Once Gammapy is installed, remember that you can always use `gammapy info` to check your setup.

Gammapy is a Python package built on [Numpy](#) and [Astropy](#), so to use it effectively, you have to learn the basics. Many good free resources are available, e.g. [A Whirlwind tour of Python](#), the [Python data science handbook](#) and the [Astropy Hands-On Tutorial](#).

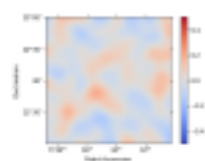
Introduction

The following three tutorials show different ways of how to use Gammapy to perform a complete data analysis, from data selection to data reduction and finally modeling and fitting.

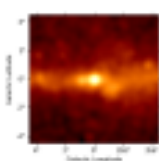
The first tutorial is an overview on how to perform a standard analysis workflow using the high level interface in a configuration-driven approach, whilst the second deals with the same use-case using the low level API and showing what is happening *under-the-hood*. The third tutorial shows a glimpse of how to handle different basic data structures like event lists, source catalogs, sky maps, spectral models and flux points tables.



High level interface



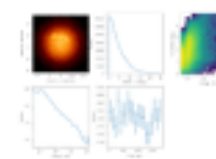
Low level API



Data structures

Data exploration

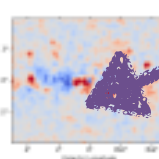
These three tutorials show how to perform data exploration with Gammapy, providing an introduction to the CTA, H.E.S.S. and Fermi-LAT data and instrument response functions (IRFs). You will be able to explore and filter event lists according to different criteria, as well as to get a quick look of the multidimensional IRFs files.



CTA with Gammapy



H.E.S.S. with Gammapy



Fermi-LAT with Gammapy

Data analysis

The following set of tutorials are devoted to data analysis, and grouped according to the specific covered use cases in spectral analysis and flux fitting, image and cube analysis modelling and fitting, as well as time-dependent analysis with light-curves.

Most common analysis recipes

- Spectral fitting
- Source detections
- Temporal Modelling
- 3D FoV analysis
- Simulations

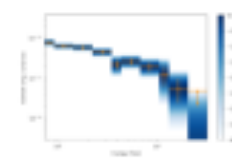
Getting started

- Instrument specific data handling
- Underlying data structures
- A full analysis workflow

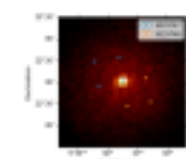
Data analysis

The following set of tutorials are devoted to data analysis, and grouped according to the specific covered use cases in spectral analysis and flux fitting, image and cube analysis modelling and fitting, as well as time-dependent analysis with light-curves.

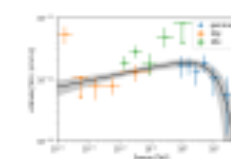
1D Spectral



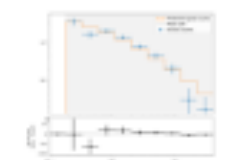
Spectral analysis



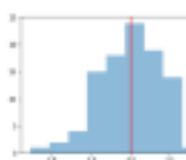
Spectral analysis with energy-dependent directional cuts



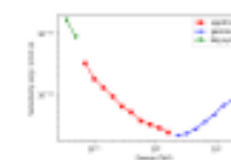
Flux point fitting



Spectral analysis of extended sources

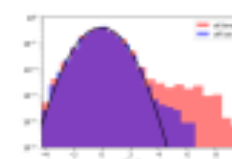


1D spectrum simulation

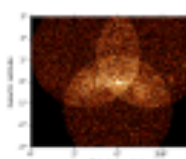


Point source sensitivity

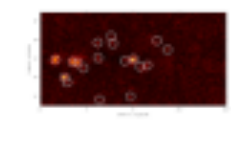
2D Image



Ring background map

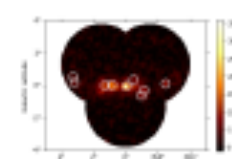


2D map fitting

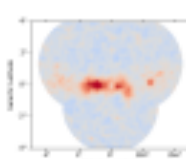


Source detection and significance maps

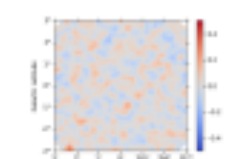
3D Cube



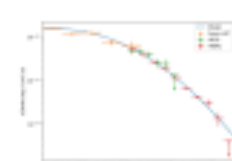
Basic image exploration and fitting



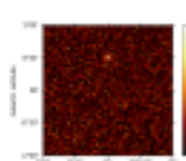
3D detailed analysis



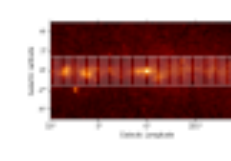
3D map simulation



Multi instrument joint 3D and 1D analysis



Event sampling



Flux Profile Estimation

Learn how to use the general API

- Go beyond the tutorials
- Understand the underlying API
- Exploit Gammapy flexibility!
- Test fit diagnostics
- Add your own models
- —

Package / API

The following tutorials demonstrate different dimensions of the Gammapy API or expose how to perform more specific use cases.

Pulsar analysis

Makers - Data reduction

Source catalogs

Models

Modelling

Fitting

Maps

Mask maps

Dark matter spatial and spectral models

Datasets - Reduced data, IRFs, models

Scripts

For interactive use, IPython and Jupyter are great, and most Gammapy examples use those. However, for long-running, non-interactive tasks like data reduction or survey maps, you might prefer a Python script.

The following example shows how to run Gammapy within a Python script.

- Survey map

Models gallery

- ❖ A variety of inbuilt models in the Models Gallery
- ❖ Spatial, Temporal and Spectral Models
- ❖ Easy to add your own custom models

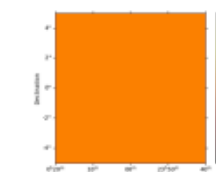
Model gallery

The model gallery provides a visual overview of the available models in Gammapy. In general the models are grouped into the following categories:

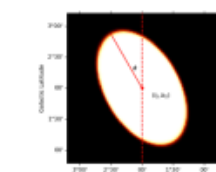
- **SpectralModel**: models to describe spectral shapes of sources
- **SpatialModel**: models to describe spatial shapes (morphologies) of sources
- **TemporalModel**: models to describe temporal flux evolution of sources, such as light and phase curves

The models follow a naming scheme which contains the category as a suffix to the class name.

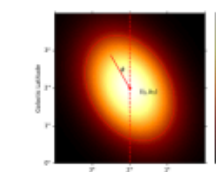
Spatial models



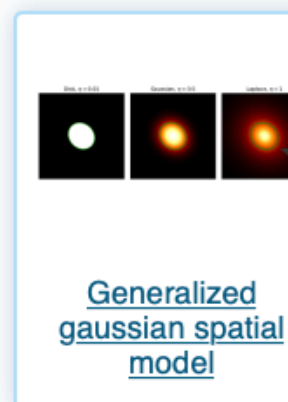
Constant spatial model



Disk spatial model

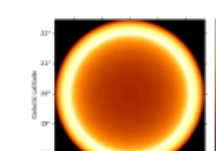


Gaussian spatial model



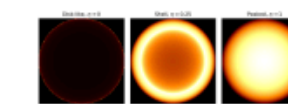
Generalized gaussian spatial model

This is a spatial model parametrising a generalized Gaussian function.

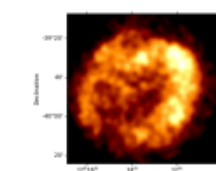


Point spatial model

Shell spatial model

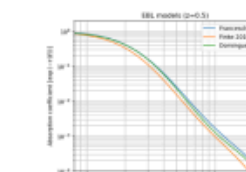


Shell2 spatial model

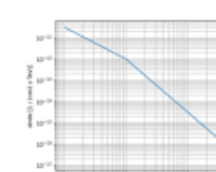


Template spatial model

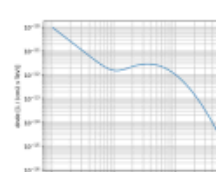
Spectral models



EBL absorption spectral model



Broken power law spectral model



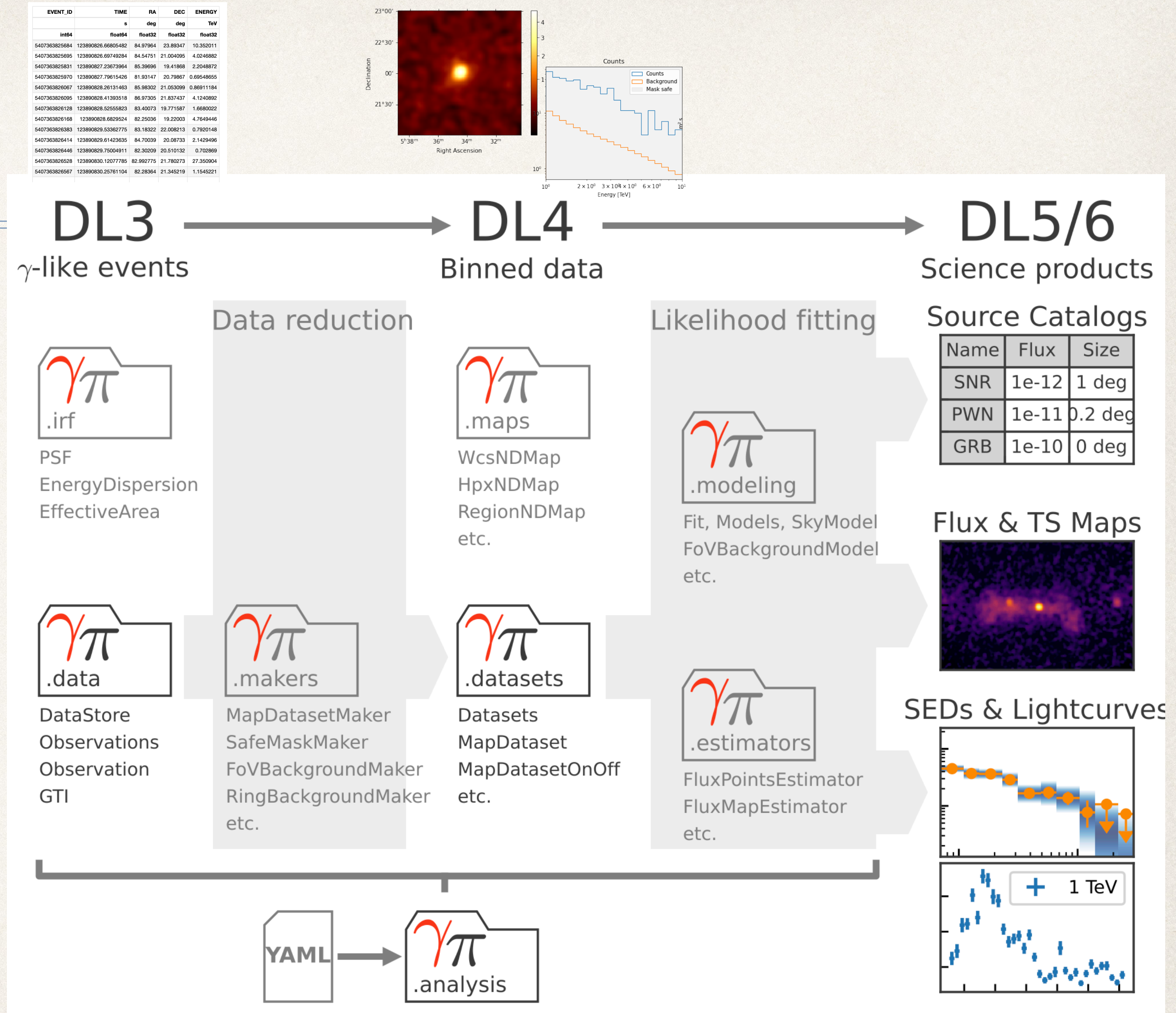
Compound spectral model

Gammapy internal workflow

- ❖ 2 step workflow

- ❖ Data reduction (DL3 - DL4)

- ❖ Modelling and fitting (DL5 - DL5)



Data reduction

DL3 → DL4 → DL5



- Bin events (and IRFs) into n-dim sky maps

- Apply event selections (time, offset, etc)

- Spatial and energy binning

- Generalised case: 3D maps

- Image analysis: cube with one energy bin

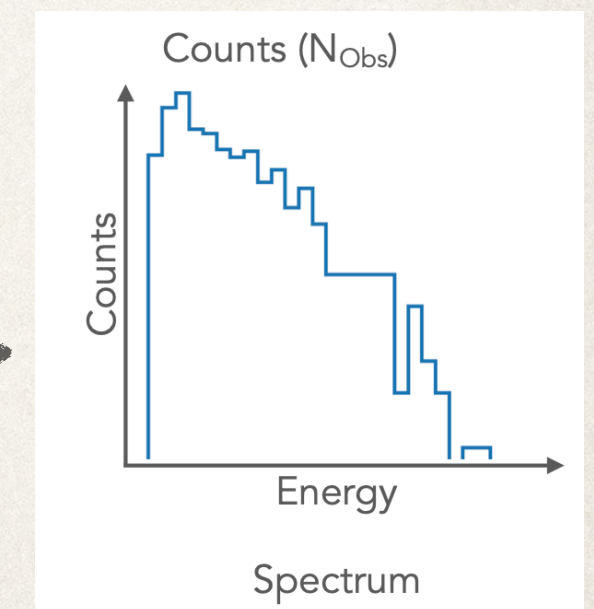
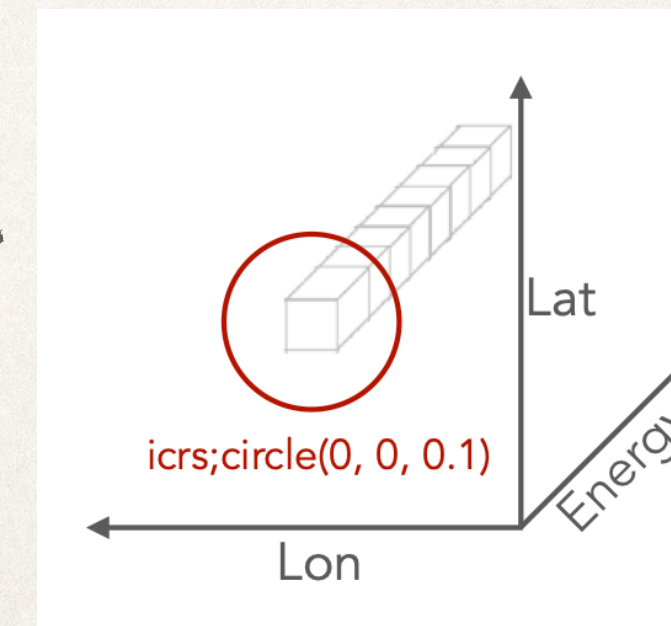
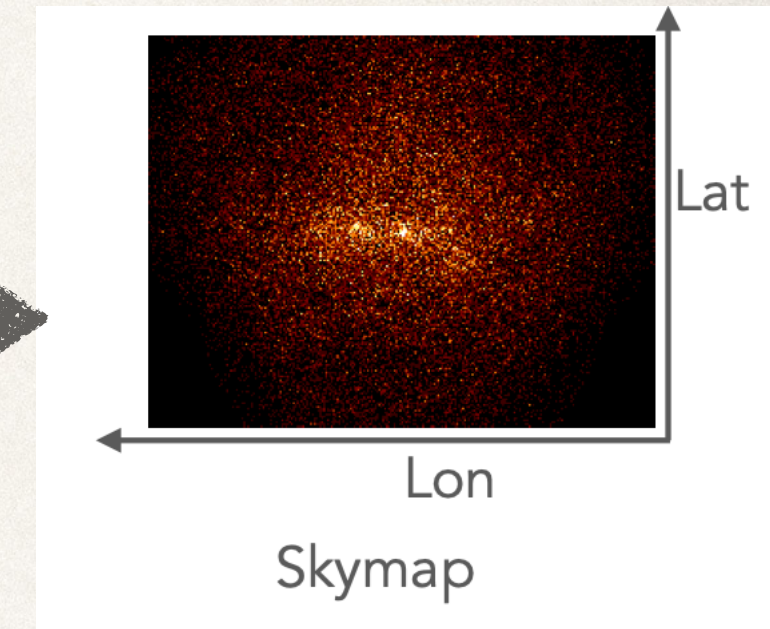
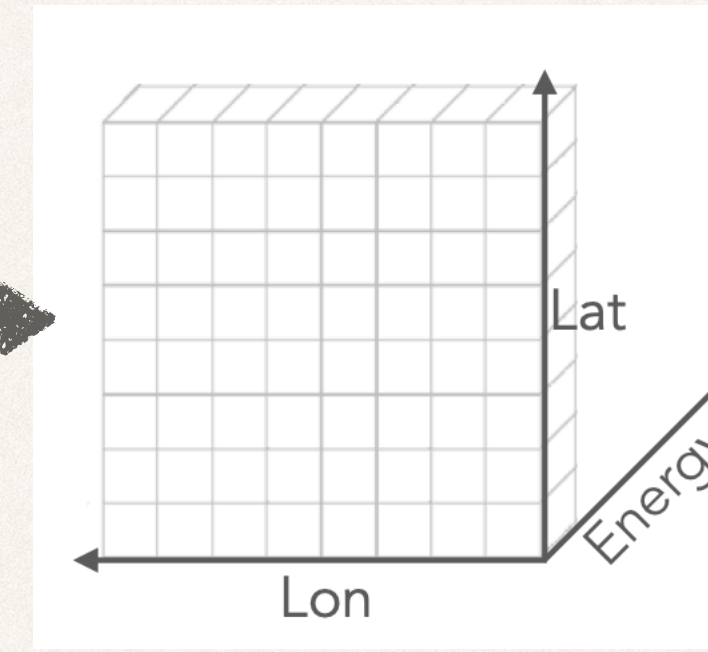
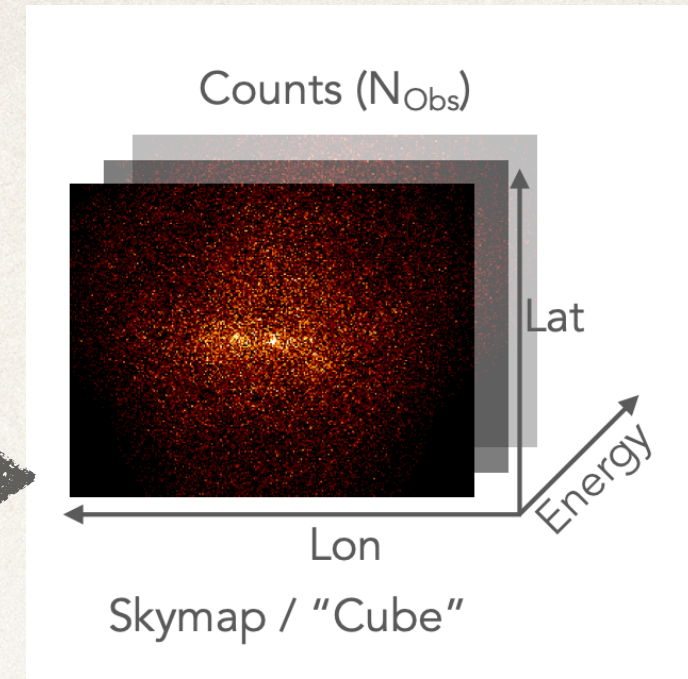
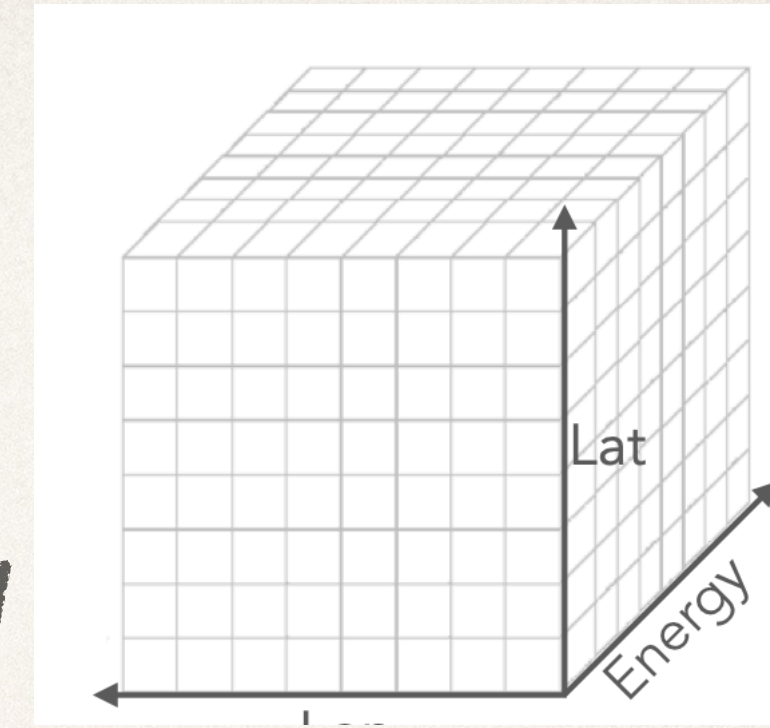
- Spectral analysis: Cube with one spatial bin

EVENT_ID	TIME	RA	DEC	ENERGY
	s	deg	deg	TeV
int64	float64	float32	float32	float32
5407363825684	123890826.66805482	84.97964	23.89347	10.352011
5407363825695	123890826.69749284	84.54751	21.004095	4.0246882
5407363825831	123890827.23673964	85.39696	19.41868	2.2048872
5407363825970	123890827.79615426	81.93147	20.79867	0.69548655
5407363826067	123890828.26131463	85.98302	21.053099	0.86911184
5407363826095	123890828.41393518	86.97305	21.837437	4.1240892
5407363826128	123890828.52555823	83.40073	19.771587	1.6680022
5407363826168	123890828.6829524	82.25036	19.22003	4.7649446
5407363826383	123890829.53362775	83.18322	22.008213	0.7920148
...

3D analysis

Image analysis

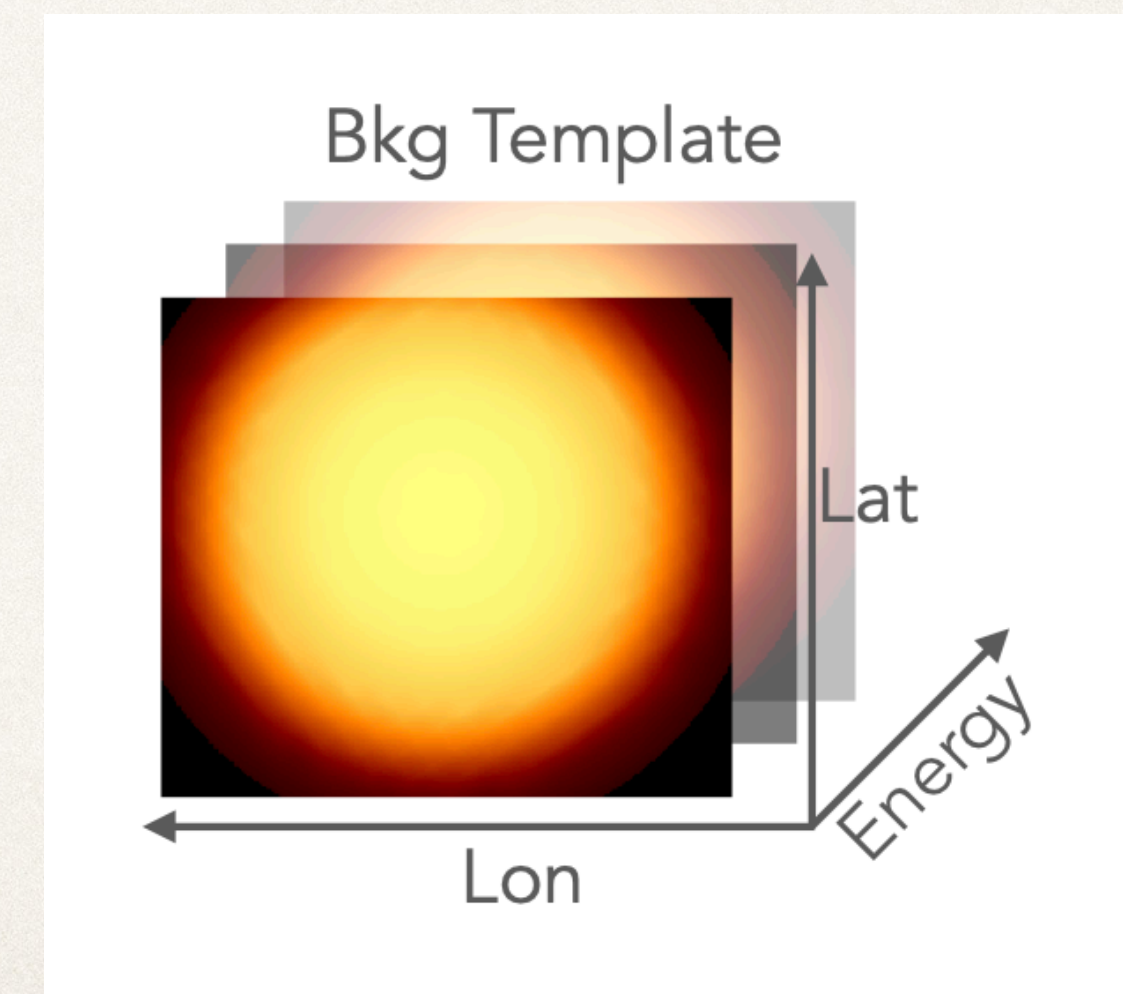
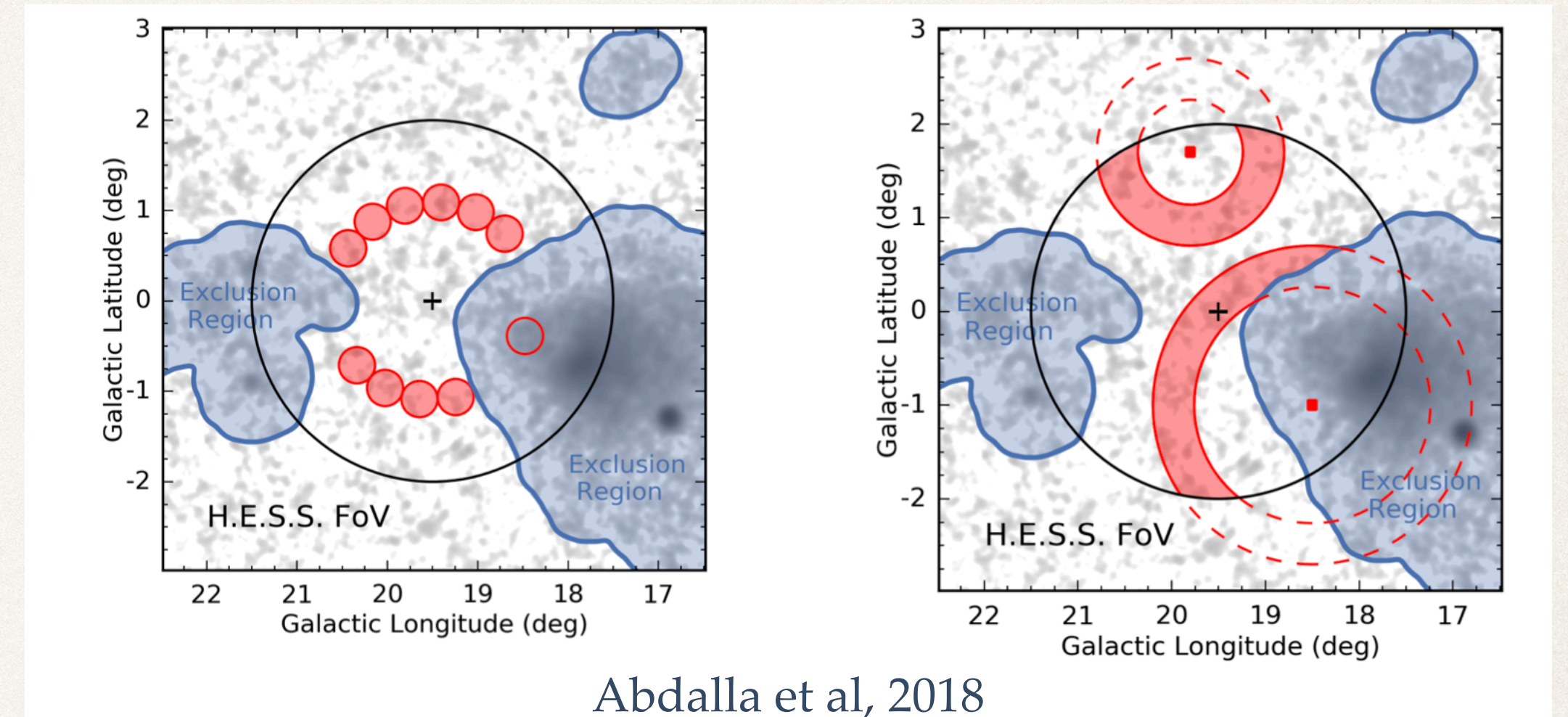
Spectral analysis analysis



Data reduction

DL3 \longrightarrow DL4 \dashrightarrow DL5

- ❖ Choose the background estimation algorithm
 - ❖ Reflected background
 - ❖ Ring background
 - ❖ FoV background to be modelled simultaneously with the source
 - Need full enclosure IRFs, with available background models!
- ❖ Choice of background exclusion regions
- ❖ Appropriate choice of statistics



Data fitting

DL4 \longrightarrow DL5

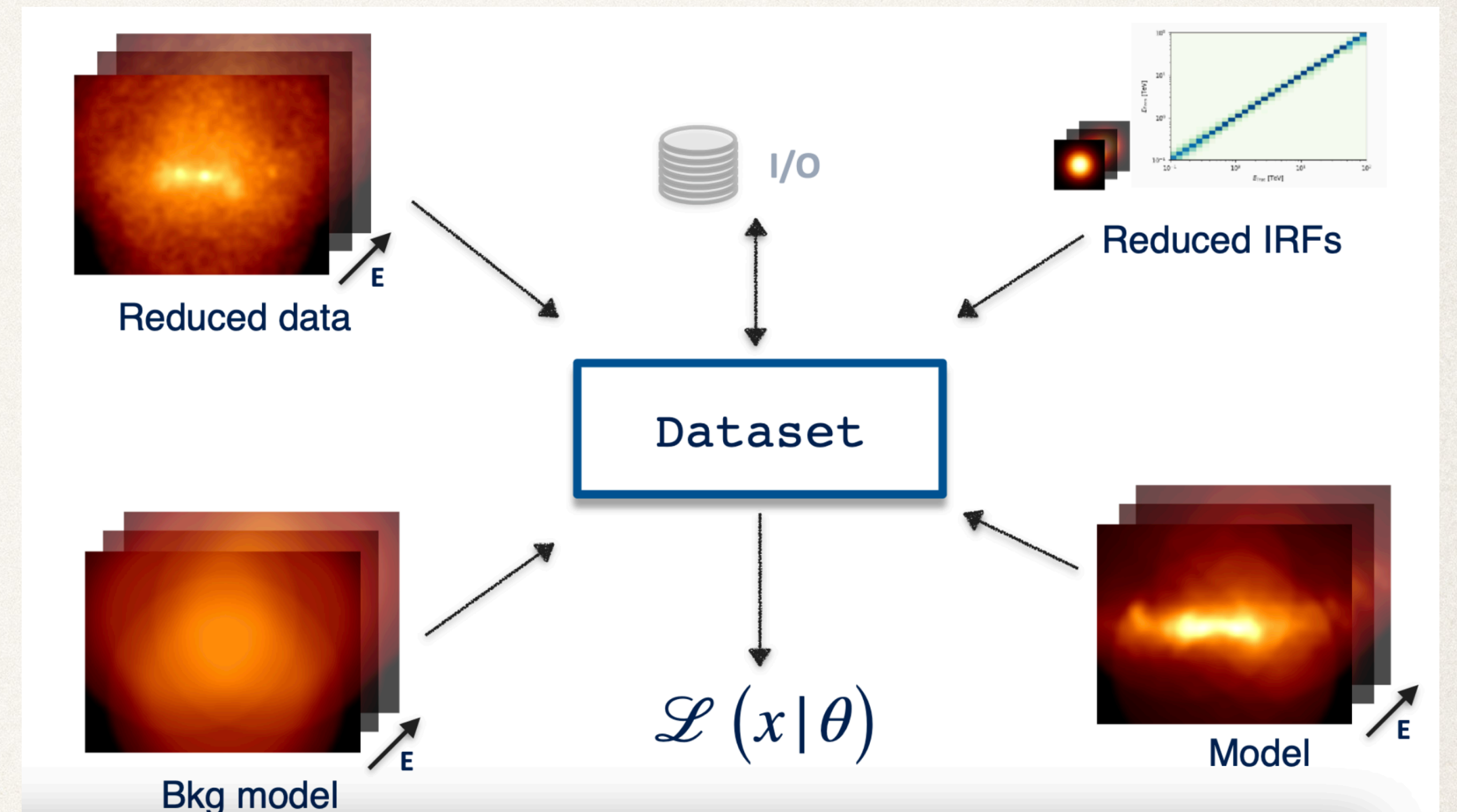
- ❖ Fitting on pre-computed datasets
 - ❖ eg: From HAWC, Fermi-LAT, OGIP files, etc
- ❖ Forward folding with maximum likelihood estimation

$$N_{Pred}(p, E) = N_{bkg}(p, E) + \sum_{src} N_{src}(p, E)$$

Cash: known background

$$TS = -2 \log L = 2 \sum (N * \log N_{pred} - N_{pred})$$

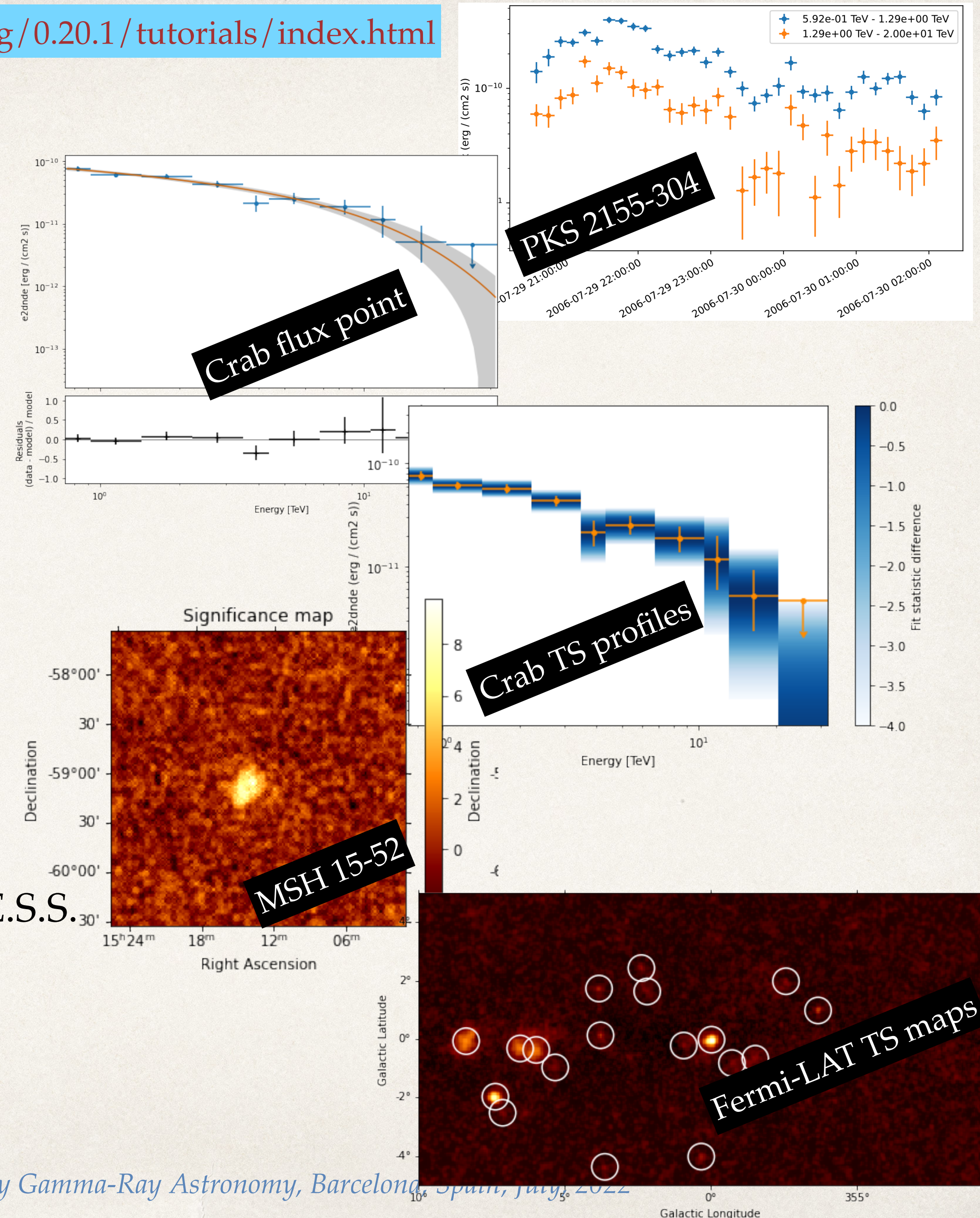
Wstat: counts with measured background



End products: DL5 and DL6

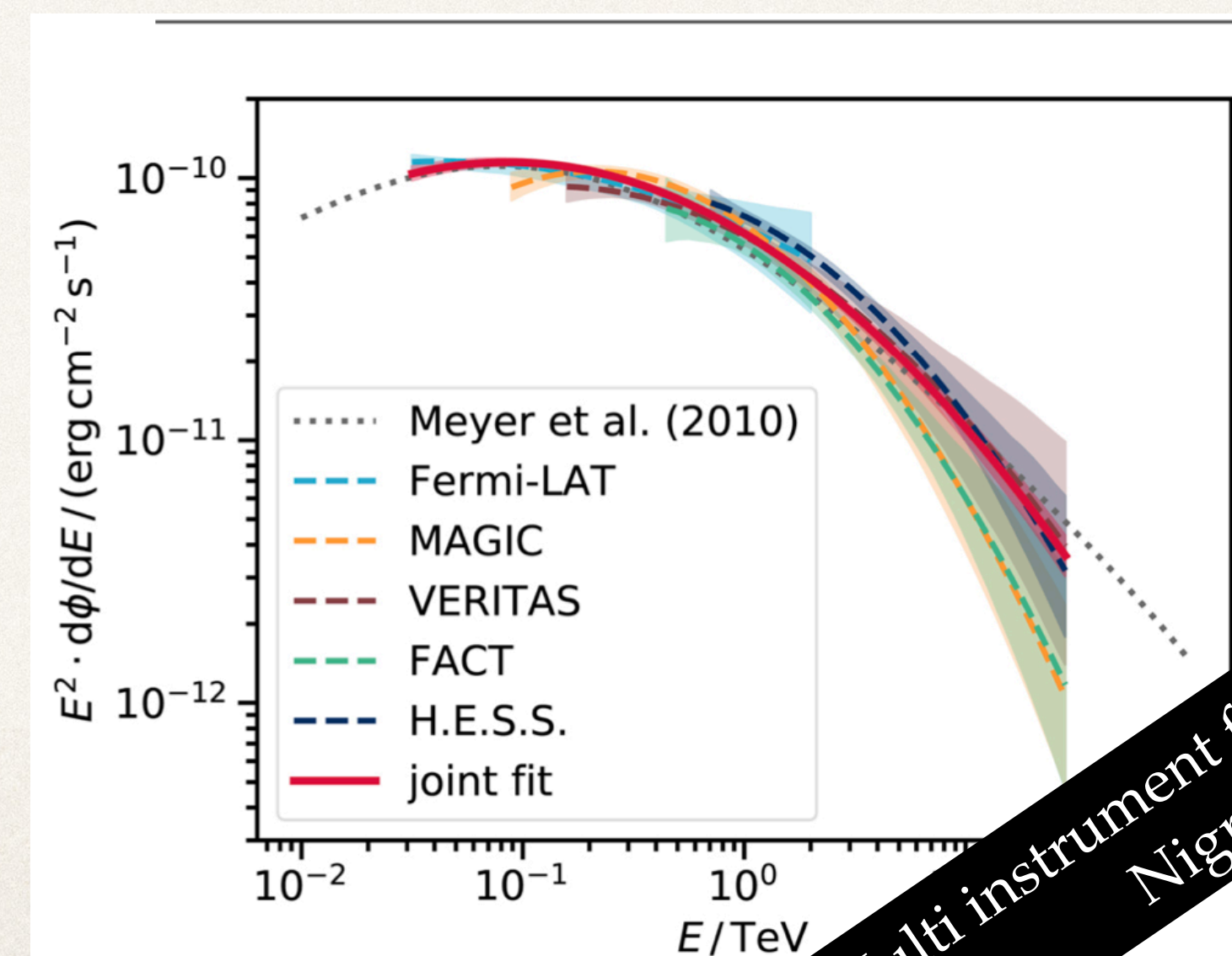
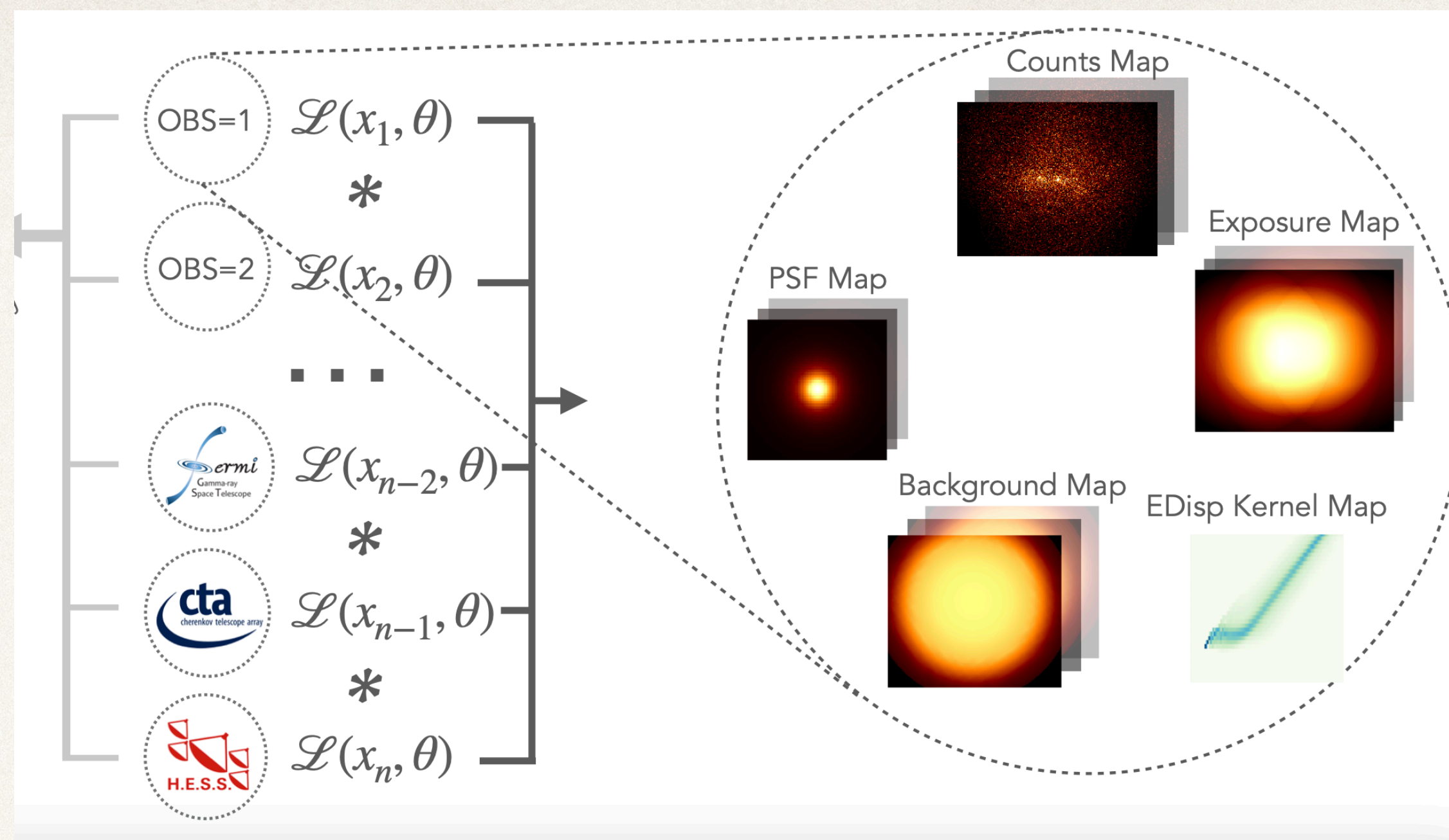
See: <https://docs.gammapy.org/0.20.1/tutorials/index.html>

- ❖ DL5: Flux points, light curves and flux/TS maps
- ❖ Possible to fit DL5 data
 - ❖ Eg: published flux points, lightcurves
 - ❖ Chi2 statistics used
- ❖ DL6: catalogs
 - ❖ Support provide for common catalogs: Fermi 4FGL, H.E.S.S. galactic plane survey, HAWC catalog, etc
 - ❖ Create your own catalogs...



Joint likelihood

- ❖ Simultaneous fitting of various datasets
- ❖ Likelihood evaluated per dataset, individual likelihoods combined to get global likelihood
- ❖ May come from the same or different instruments
- ❖ Possible to combine DL4 and DL5 data

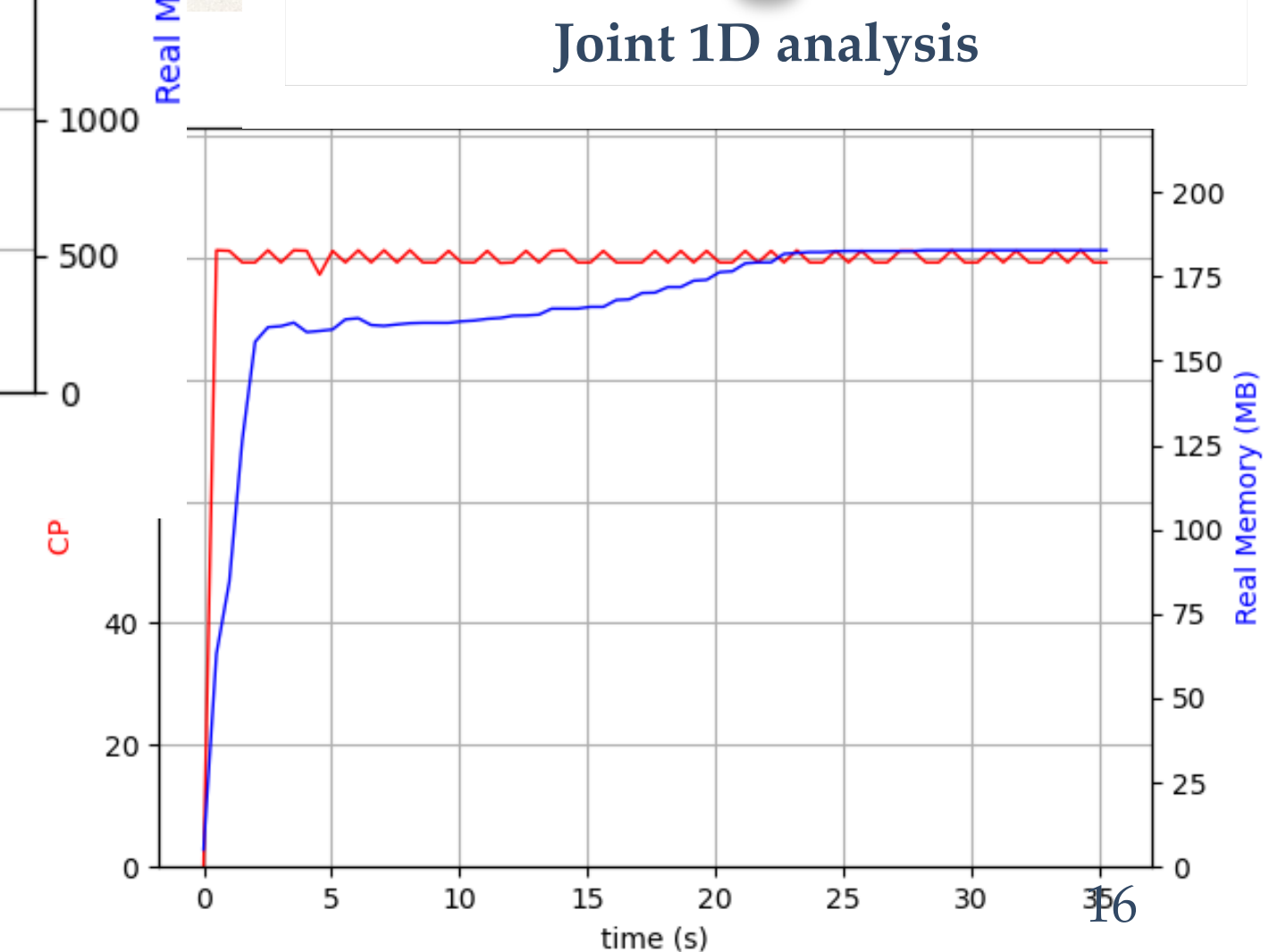
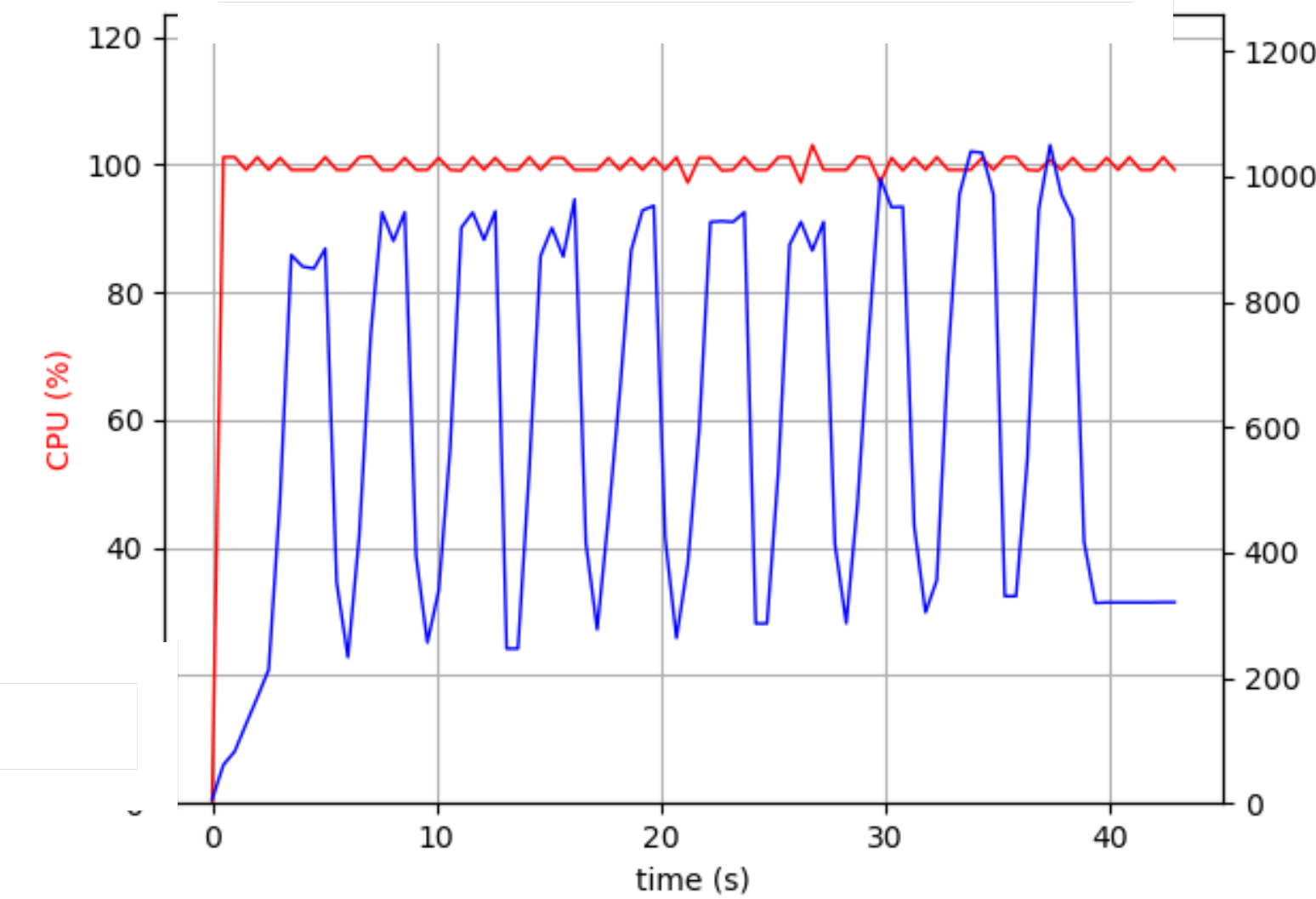
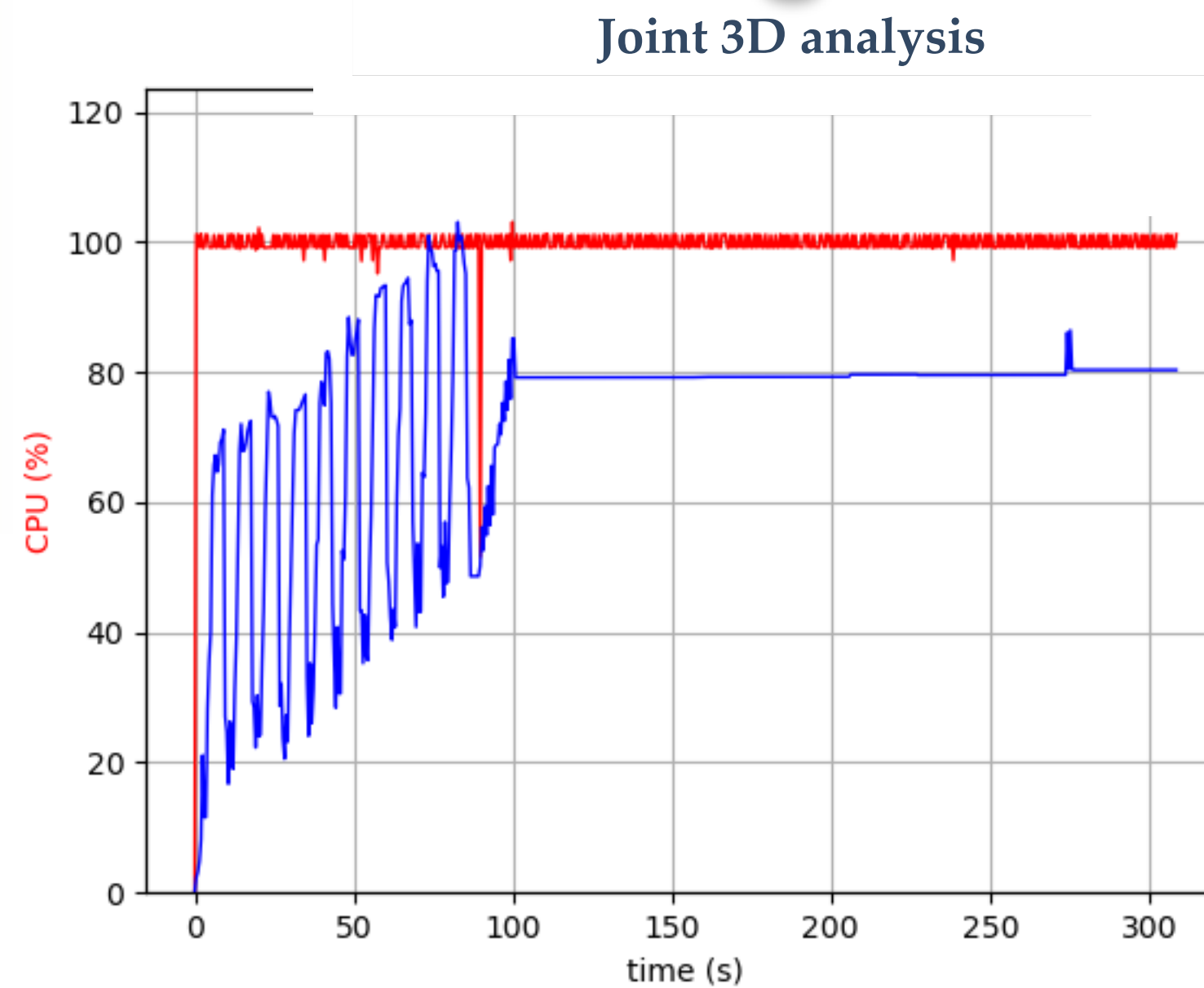
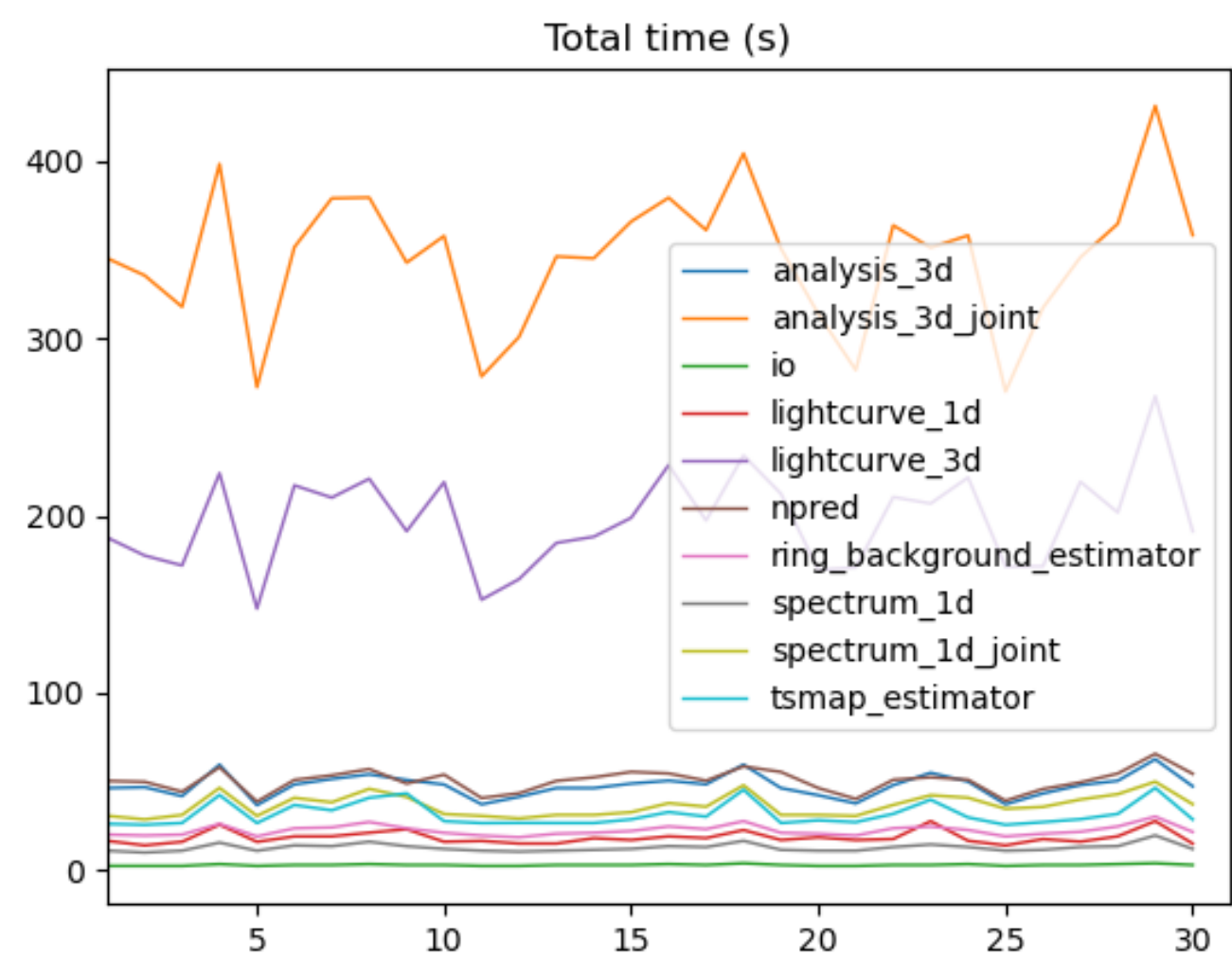


Multi instrument fitting of Crab Nebula
Nigro et al, 2019

See: <https://github.com/gammapy/gammapy-benchmarks/tree/master/benchmarks>

Gammapy benchmarks

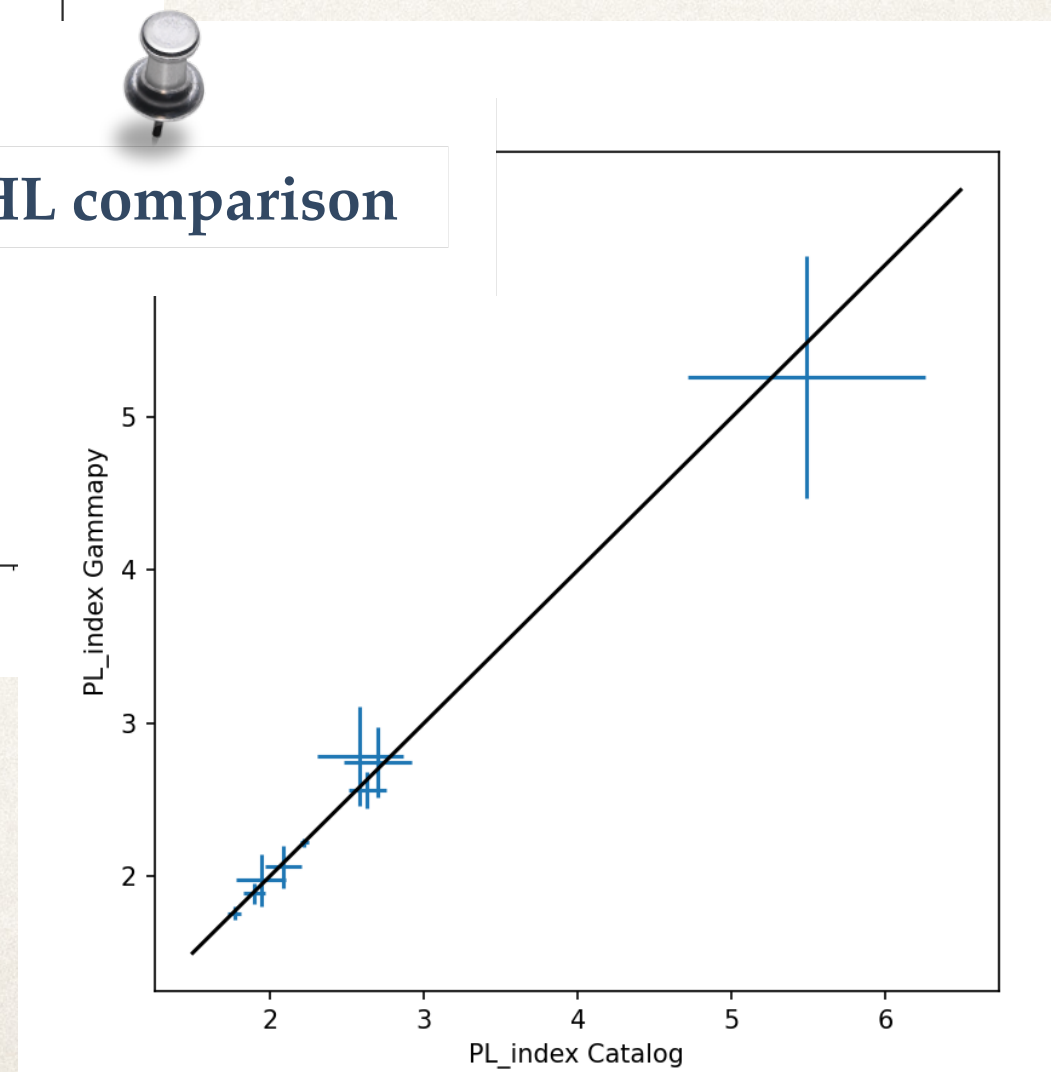
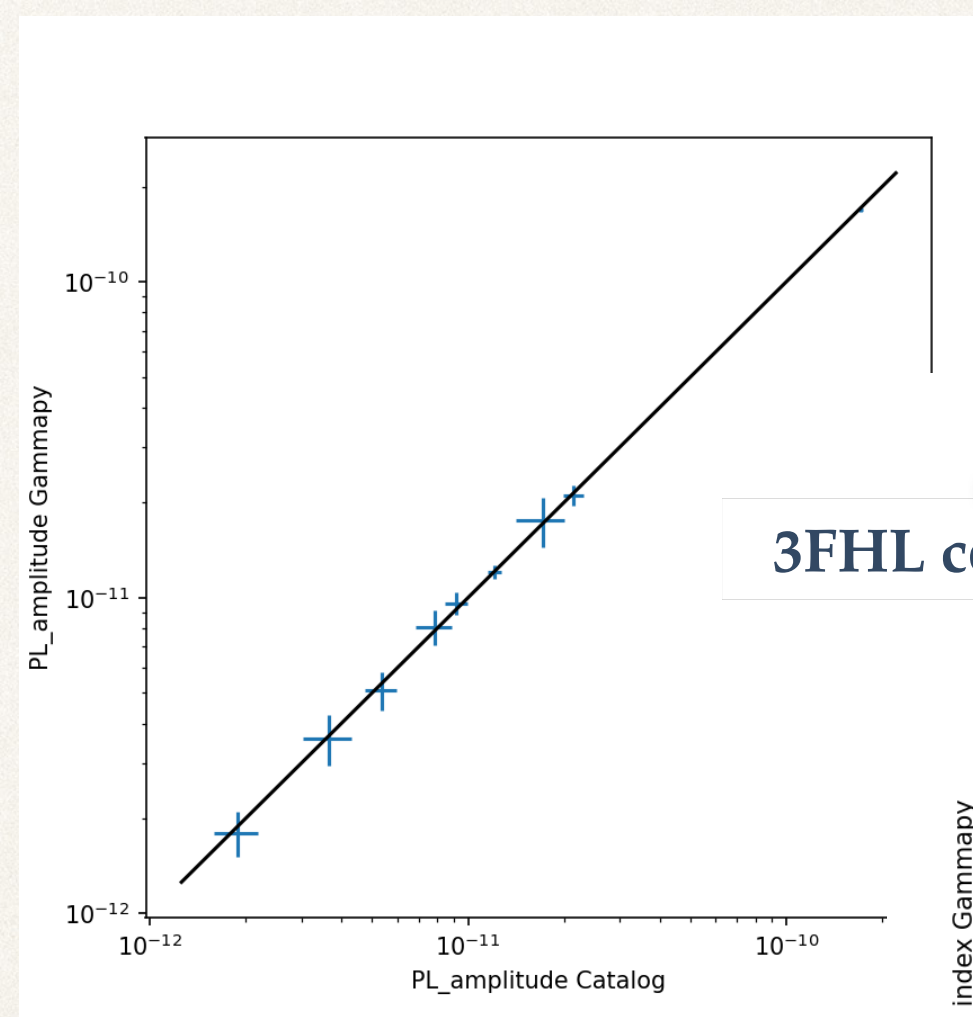
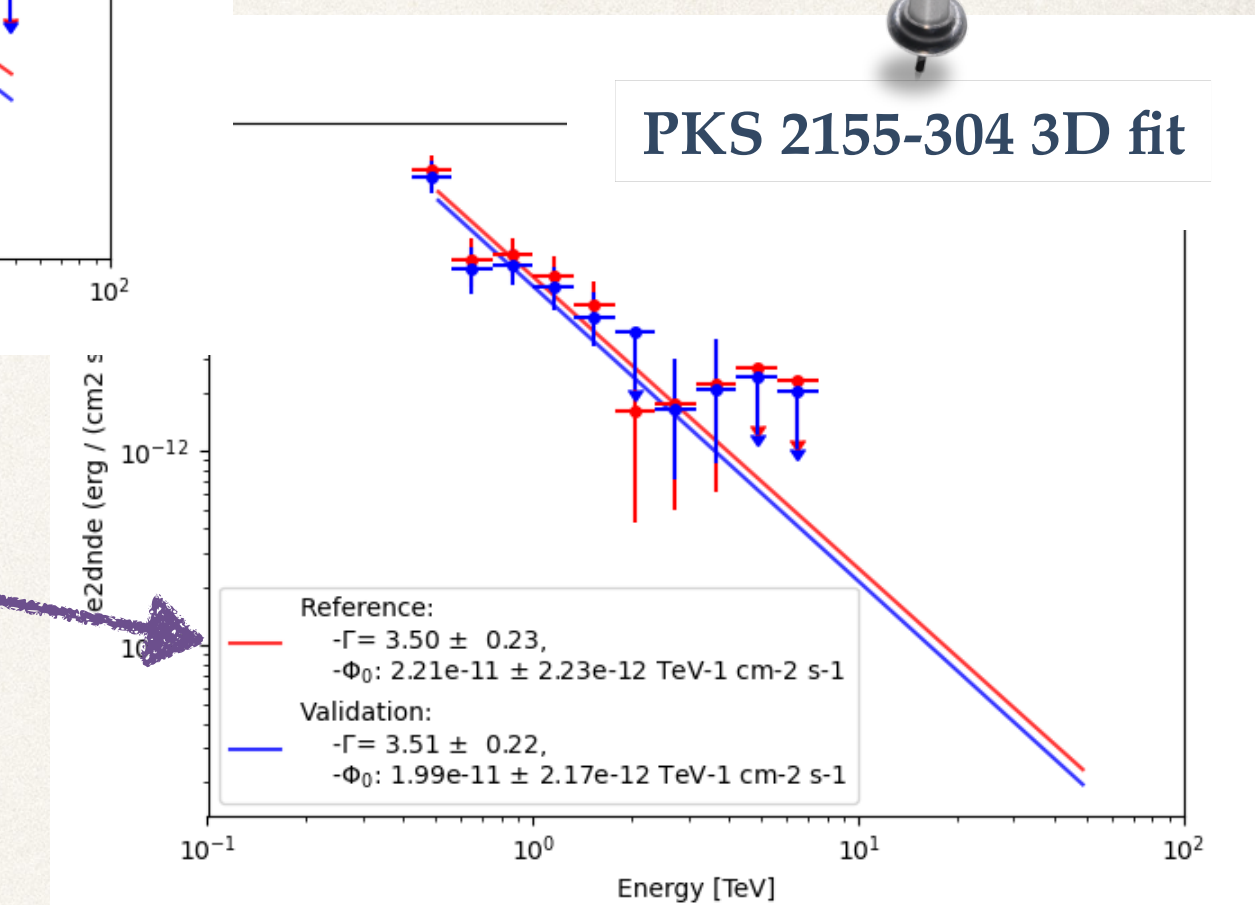
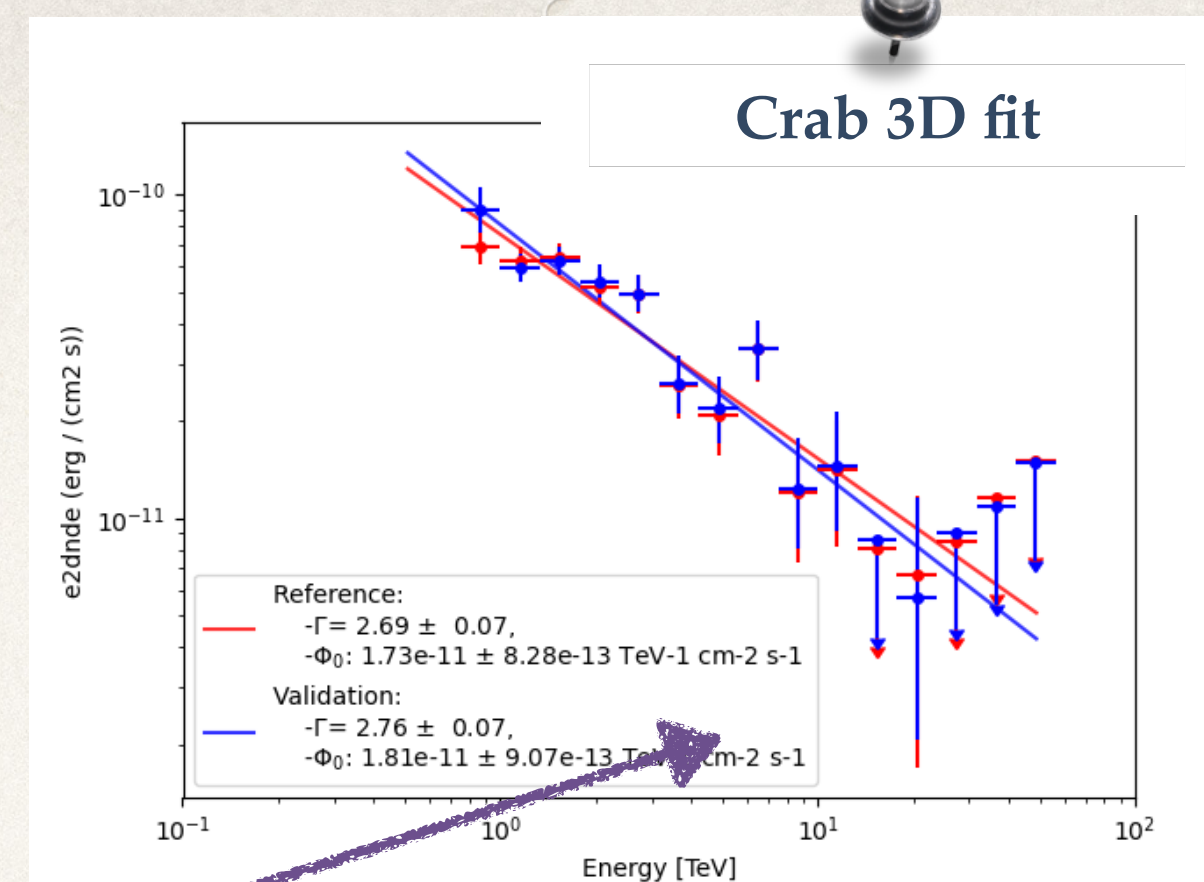
- ❖ Daily automatic monitoring of memory usage and computation time for most common use cases
- ❖ Computation time across different actions
 - ❖ Loading observations
 - ❖ Binning data and IRFs
 - ❖ Fitting
 - ❖ FluxPointEstimations
- ❖ Pinpoint PR significantly affecting the performance



Gammapy validation

- ❖ Analysis of a list of science cases before every gammapy release
 - ❖ H.E.S.S. DL3 DR1 results
 - ❖ CTA DC1 results
 - ❖ Joint Crab validation paper
 - ❖ ...
- ❖ Ensure the stability of results
- ❖ Compare against results from the Fermi ST
 - ❖ Subset of the 3FHL paper

See: <https://github.com/gammapy/gammapy-benchmarks/tree/master/validation>



Get in touch!!!

Gammapy at this
symposium: 11+
contributions

- ❖ Follow the discussions on GitHub: <https://github.com/gammapy/gammapy/discussions>
- ❖ Join us on slack: gammapy.slack.com
- ❖ User calls on zoom
- ❖ Developer calls every Friday afternoon, and regular coding sprints / co-working weeks

Thanks and stay tuned!!!

