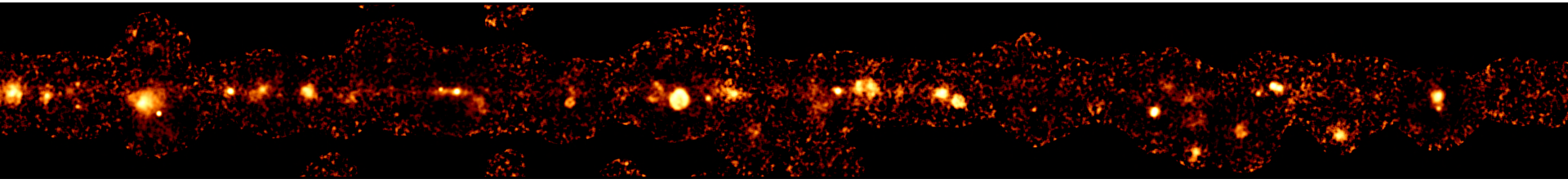
A **Python** package for
gamma-ray astronomy

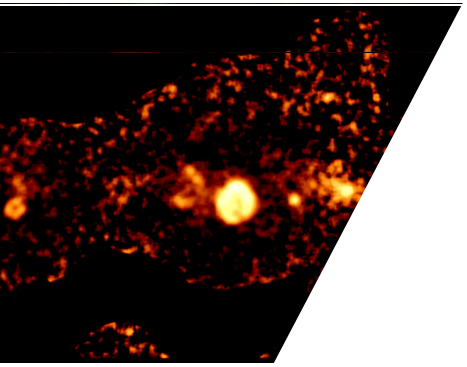
Gammapy overview

Bruno Khélifi, for the Gammapy Team
APC (CNRS/IN2P3, Université Paris Cité)

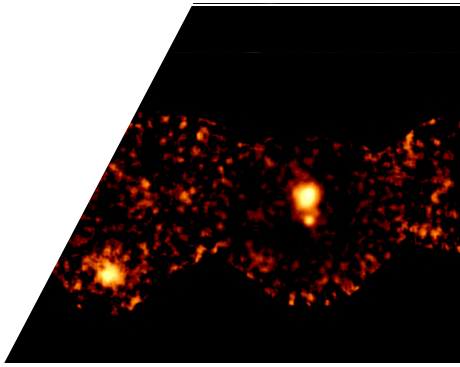


About me

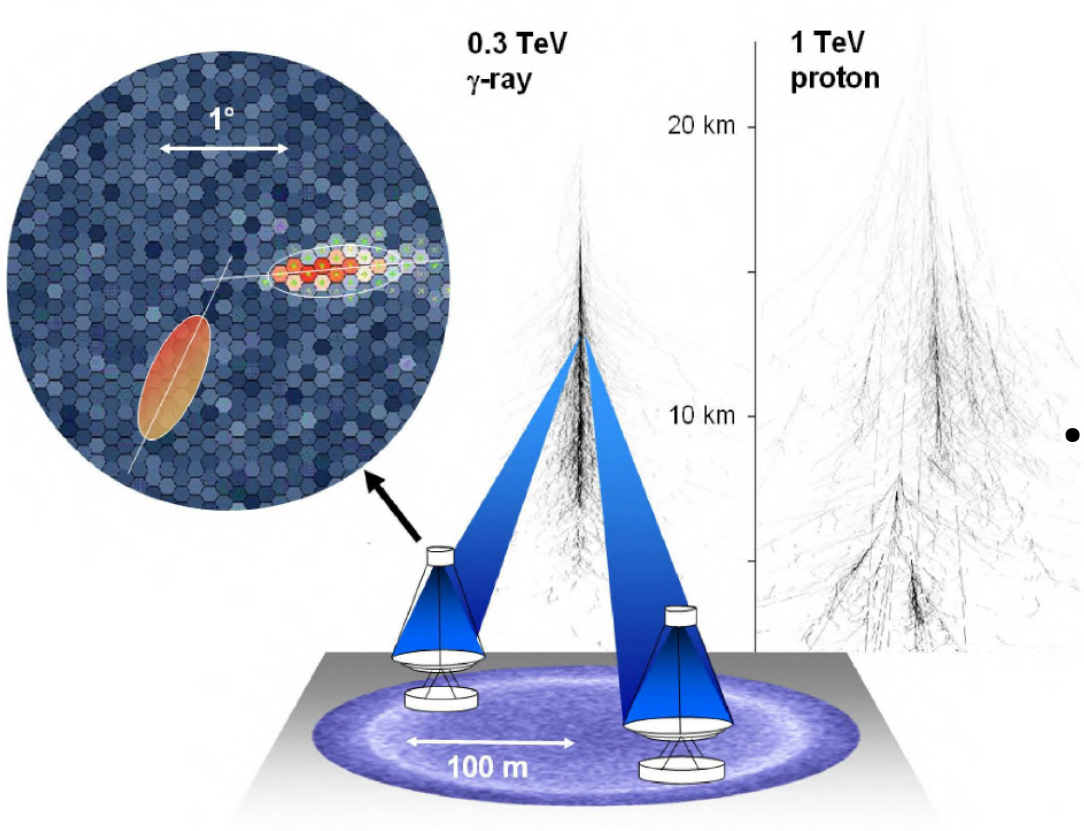
- Researcher at the laboratory AstroParticule and Cosmology (APC, Paris) – 0000-0001-6876-5577
- Gamma-ray astrophysicist (galactic sources: PWN, PeVatron), software developer (C++, Python) and data formatting
- Member of the H.E.S.S. collaboration and the CTA consortium, supporting scientist of SWGO
- Responsible of one of the 3 official pipelines of the H.E.S.S. low-level analysis (HAP-Fr), Project Manager of Gammapy (core library of the CTA Science Analysis Tool)
- Convener of the Very-high-energy Open Data Format (VODF) initiative
- Involved in the Open Science movement: SoftWare Heritage, IVOA, referent in my university and in IN2P3, etc



Reminder on the Imaging Atmospheric Cherenkov technique



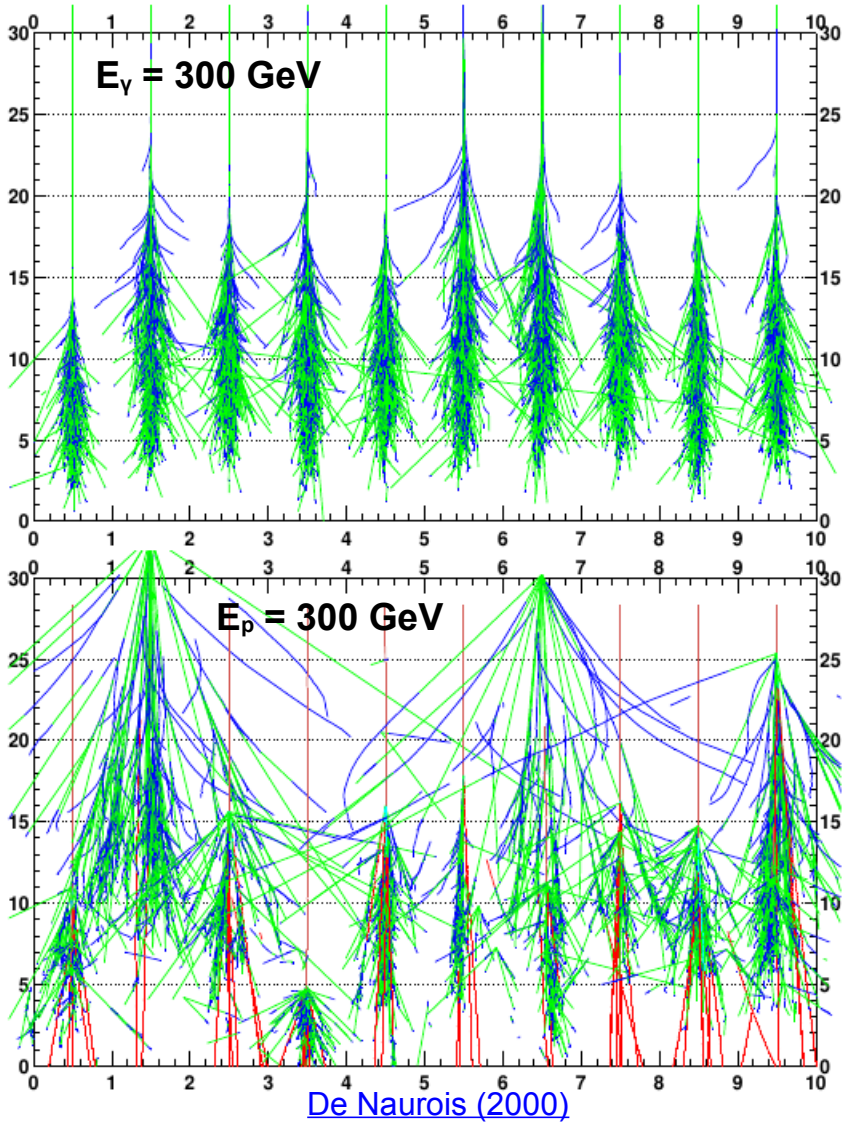
The detection technique



Hofmann, Hinton (2009)

- γ-ray shower for TeV energies
  - $H_{max} \simeq 10$km
  - Length: 10/15 km
  - Diameter: 40/100 m
  - Typical Cherenkov angle: 0.1°
- Proton shower for TeV energies
  - Inelastic collisions lead to much wider showers and much more sub-showers

The detection technique

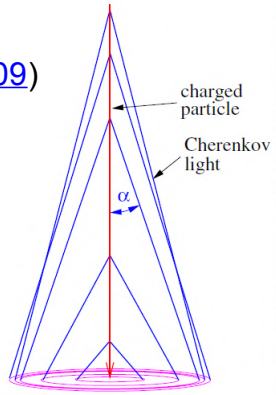


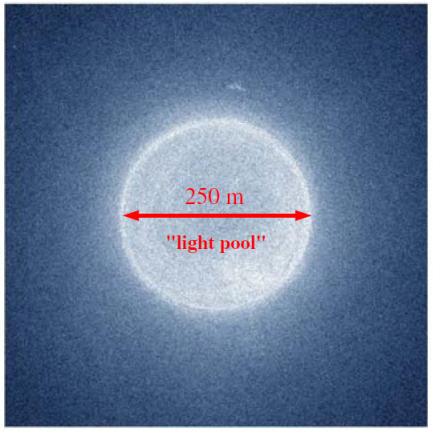
De Naurois (2000)

- γ-ray shower for TeV energies
  - $H_{max} \simeq$ 10km
  - Length: 10/15 km
  - Diameter: 40/100 m
  - Typical Cherenkov angle: 0.1°
- Proton shower for TeV energies
  - Inelastic collisions lead to much wider showers and much more sub-showers

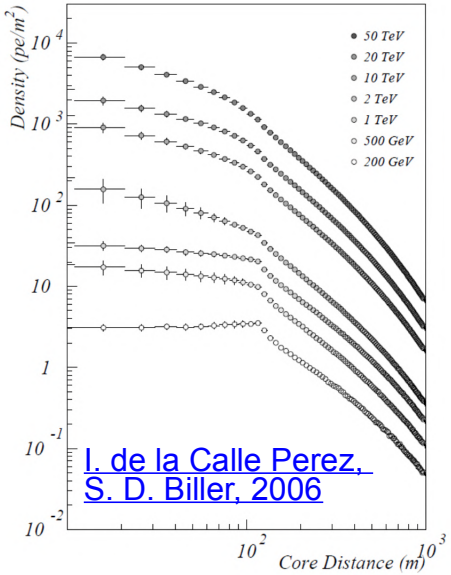
The detection technique

1TeV γ-ray
(Völk & Bernlöhr, 2009)

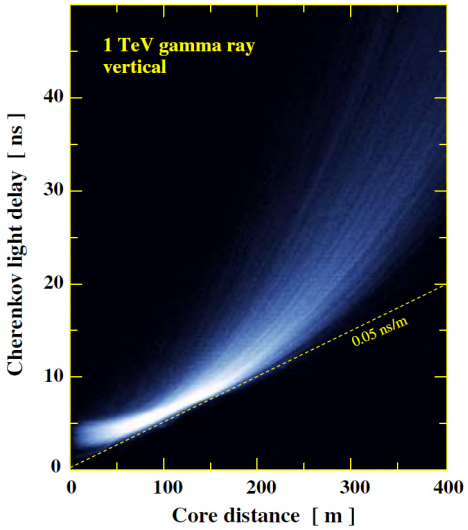




- γ-ray shower for TeV energies
  - $H_{max} \simeq 10$km
  - Length: 10/15 km
  - Diameter: 40/100 m
  - Typical Cherenkov angle: 0.1° (depend on h)
- Cherenkov light pool of TeV γ-rays
  - 300-600 nm
  - 100/150m depending of the instrument altitude
  - ~50 pe/m²
  - Arrival time: within ~2ns

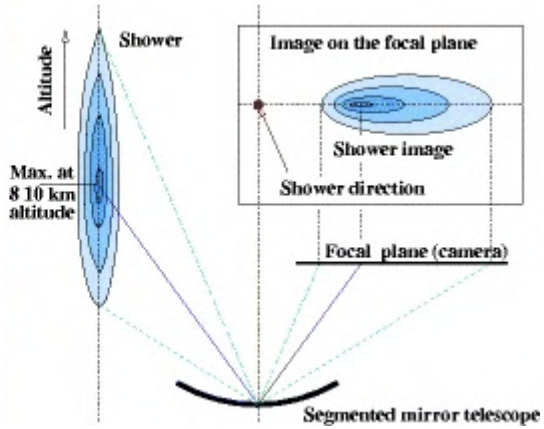


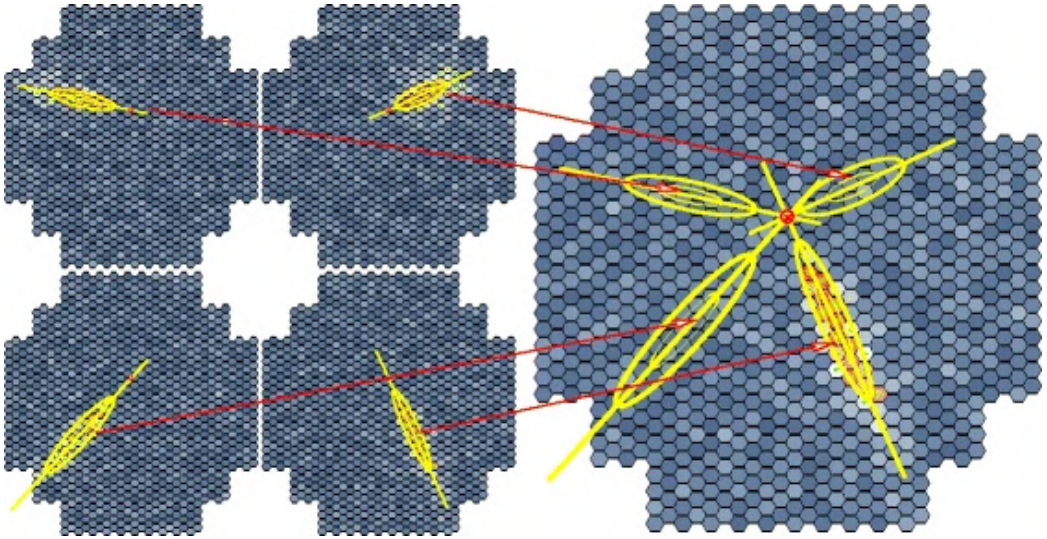
I. de la Calle Perez, S. D. Biller, 2006



Völk & Bernlöhr, 2009

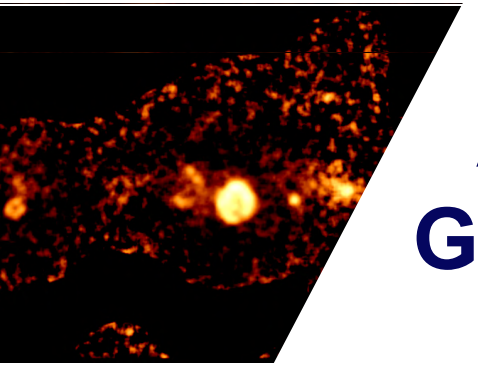
The detection technique



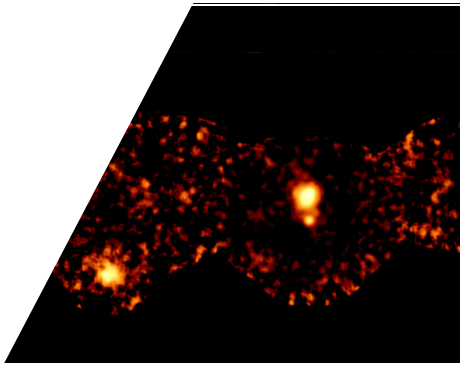


- Cherenkov images of the atmospheric showers by cameras

- Low level analysis
  - Per camera: calibration (ADC → pe) and cleaning
  - Reconstruction of the shower parameters: arrival direction, energy + shower parameters
  - Discrimination γ/hadron



A short introduction to Gamma-ray Data Analysis



From **A. Donath** (CfA), Lead Developer of Gammapy
Presentation made during the Scipy 2023 conference

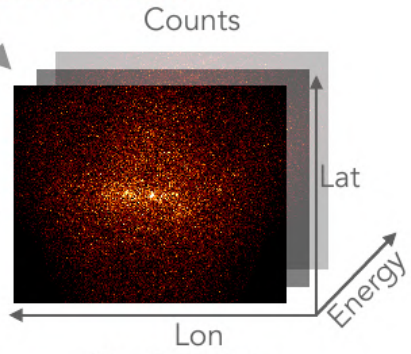
Binned Poisson Log-Likelihood

List of gamma-like events…

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

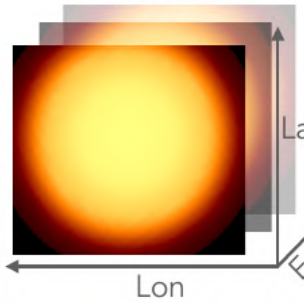
…binned into…

$N_{Obs} =$



Bkg Template

$N_{Bkg} =$



$\cdot\, n_0 \cdot \left(\frac{E}{E_0}\right)^{-\Gamma}$

"Cash statistics": summed over all "bins"

$$\mathscr{C} = 2 \sum_i N^i_{Pred} - N^i_{Obs} \cdot \log N^i_{Pred}$$

$$N_{Pred} = N_{Bkg} + \sum_{Src} N_{Pred,Src}$$

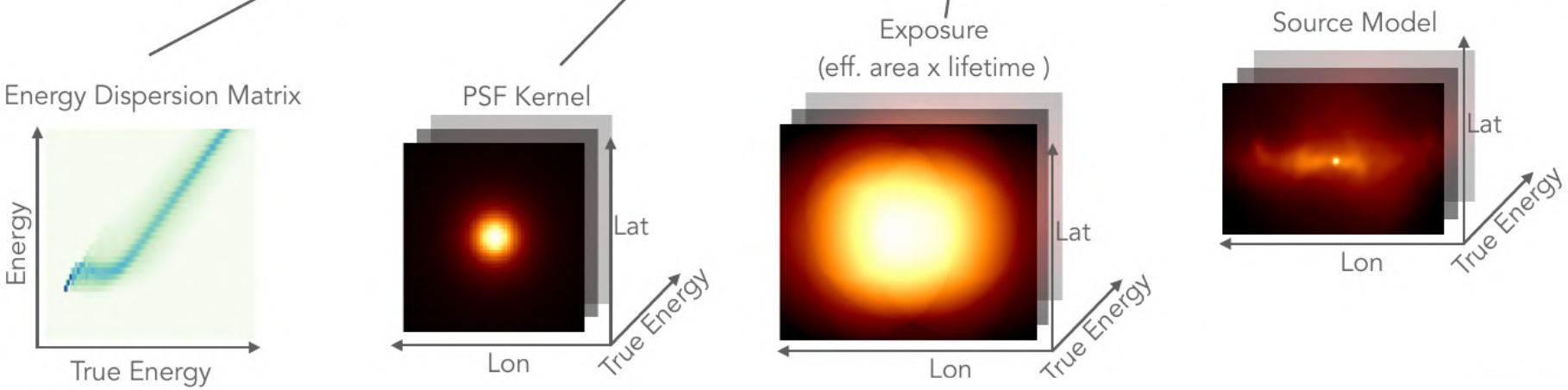
- Predicted counts are **computed per model component** ("source / object") and summed
- A **"global" background model** template with "correction parameters" is added

Data Model / Instrument Reponse

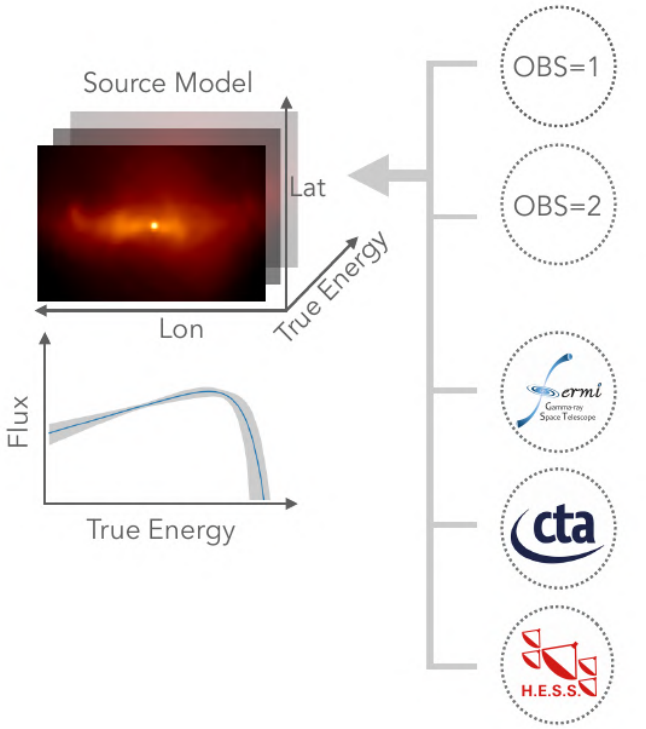
An analytical source model or template is **"forward folded" through the instrument response function (IRF)** to predict the measured number of counts…

$$N_{Pred,Src} = \mathrm{EDISP}_{Src}(\mathrm{PSF}_{Src}(\mathscr{E}_{Src} \cdot f_{Src}))$$

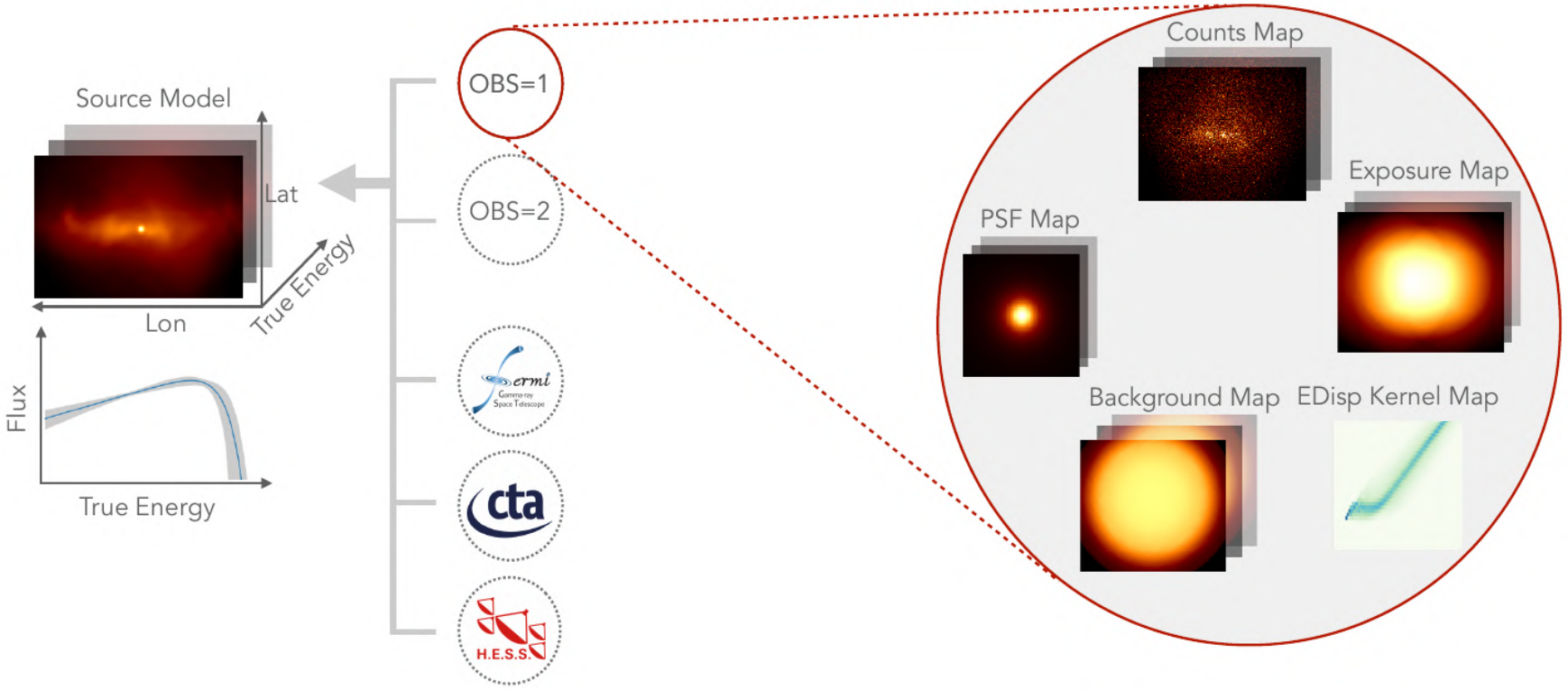
$$f_{Src} = f_{Spectral}(E) \cdot f_{Spatial}(E, l, b) \cdot f_{Temporal}(t)$$



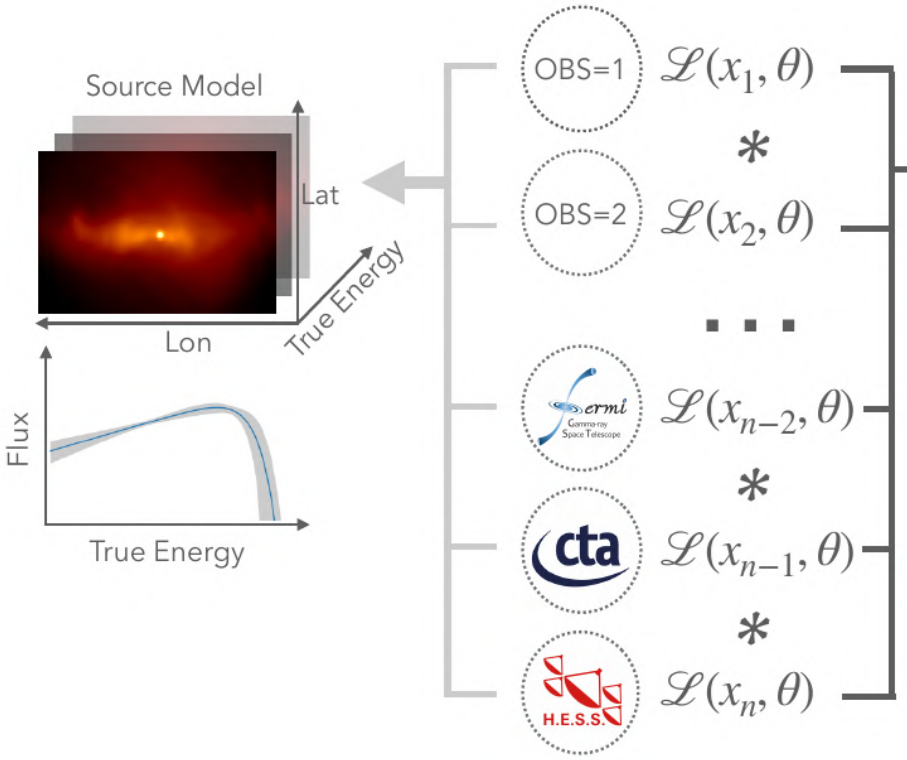
Joint Likelihood



Joint Likelihood



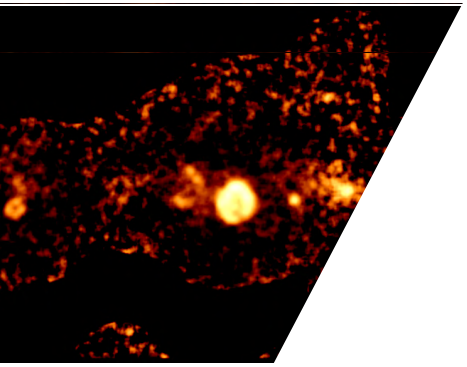
Joint Likelihood



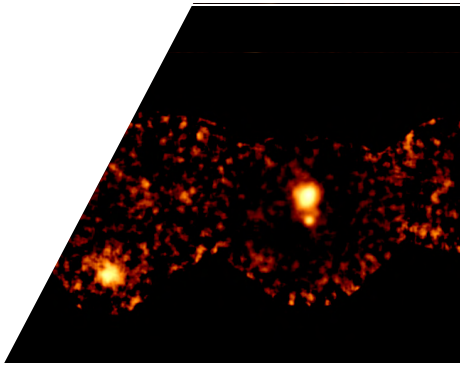
$$\mathscr{L}(x_{1\ldots n}, \theta)$$

$$\hat{\theta} = \underset{\theta \in D}{\operatorname{argmax}}\ \mathscr{L}(x_{1\ldots n}, \theta)$$

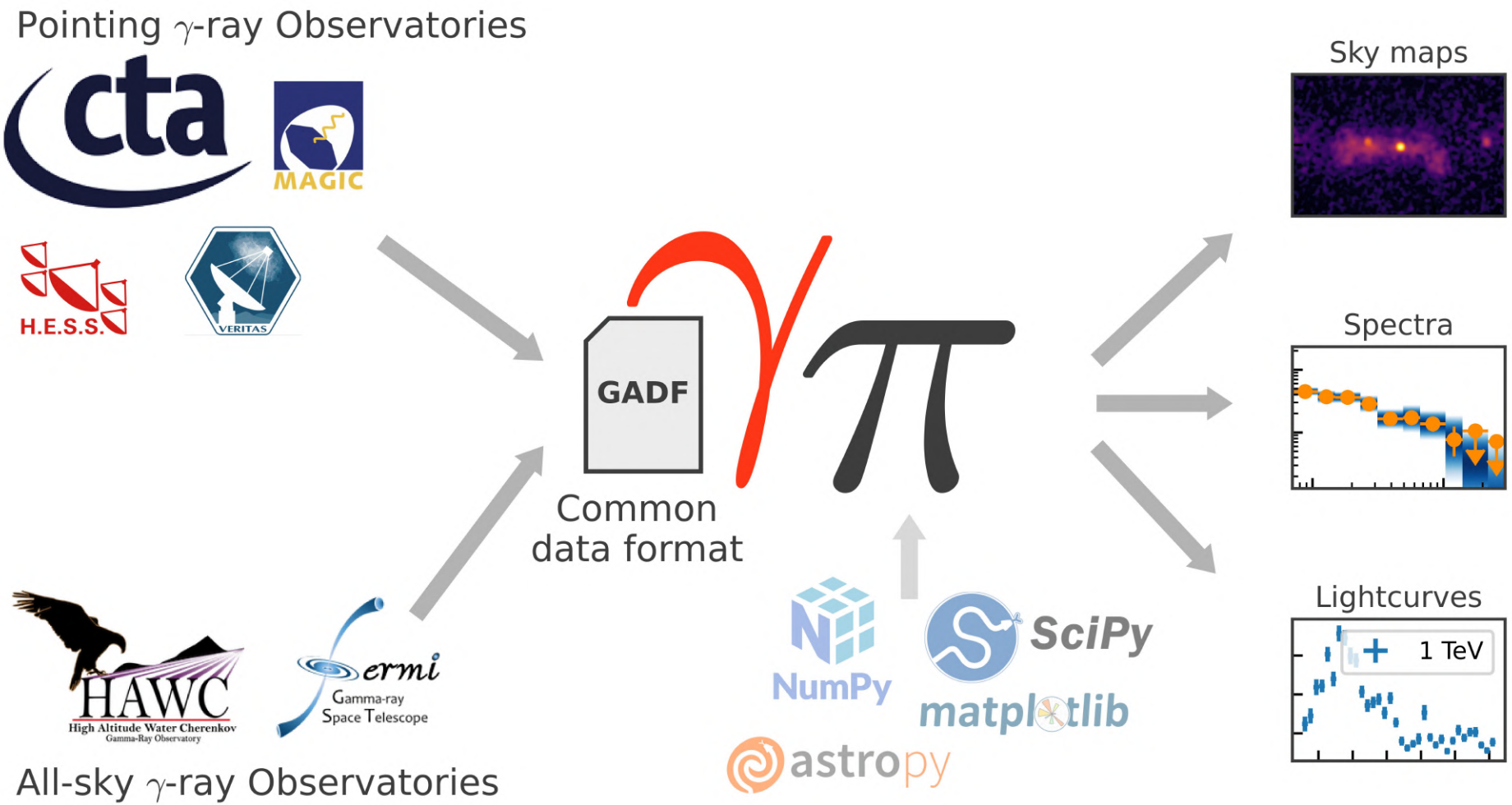
Best fit parameters and errors are reported as measurements in publications, including e.g., **total flux, spectral indices** and morphological parameters such as **size, position etc.**



The Gammapy package

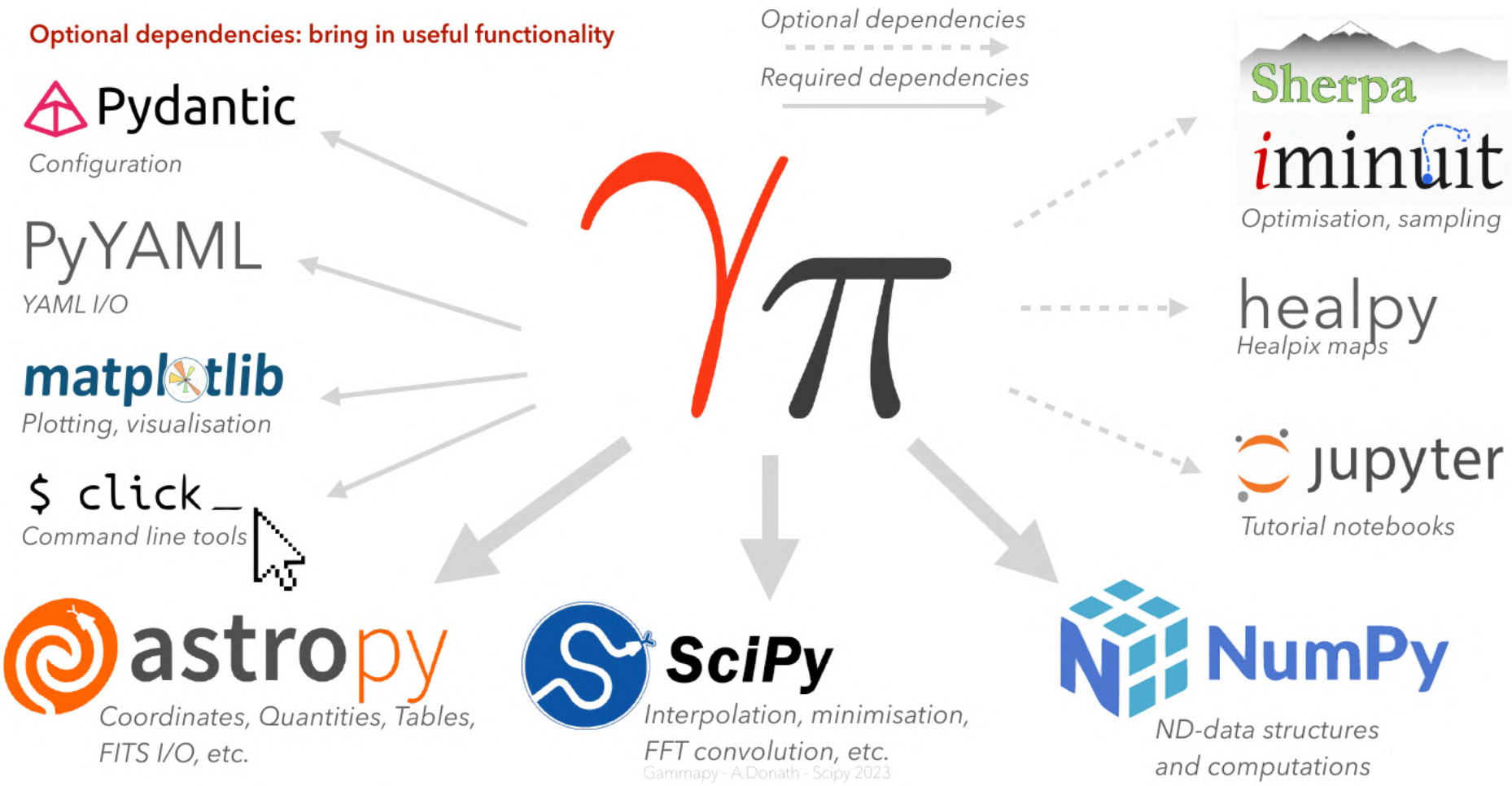


Main concept of Gammapy

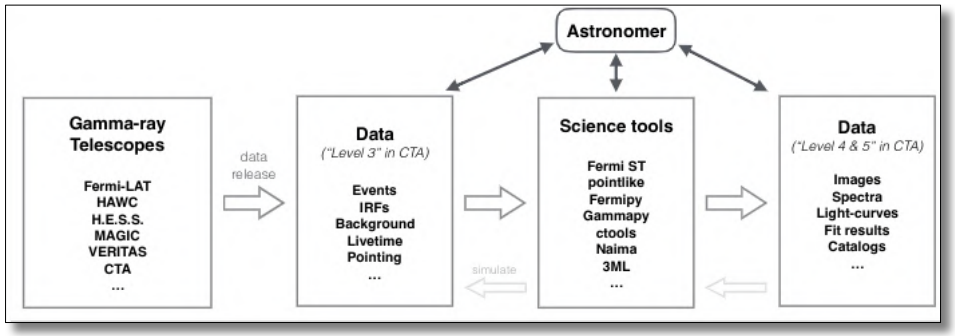


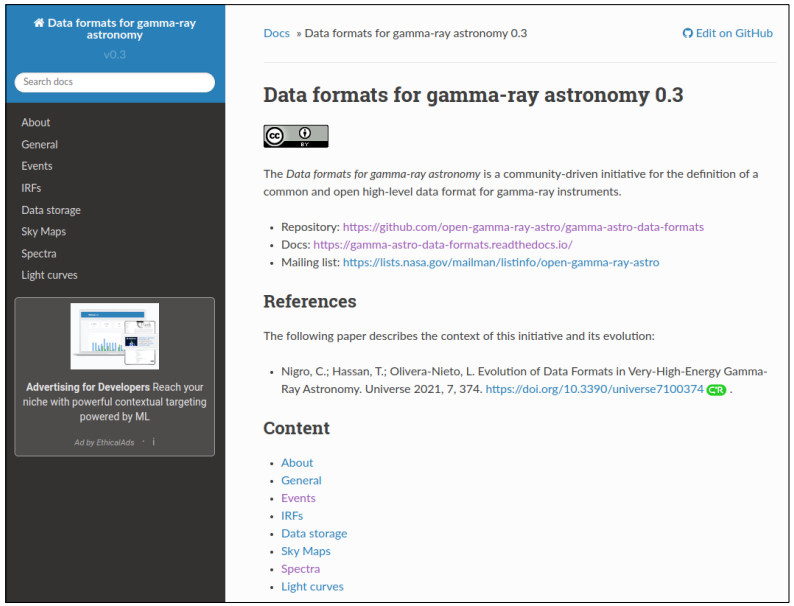
V1.2rc1 released on Feb., 16th

Gammapy dependencies



Read/Input format

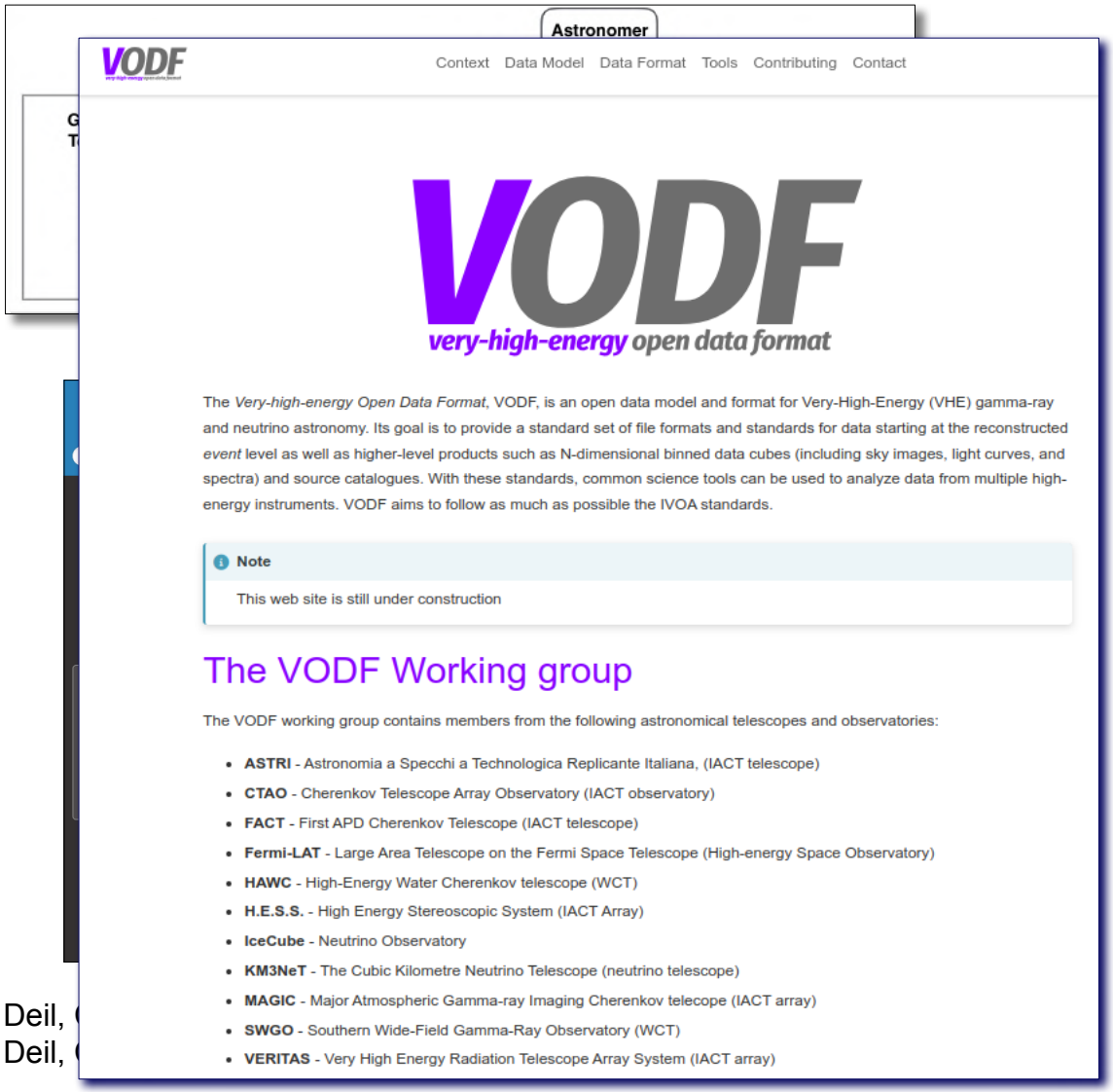




- The TeV γ-ray community agrees to use an open common format: Gamma Astro Data Format (GADF)
  - Creation within an open initiative
- Define mainly 2 levels of data:
  - The DL3
  - The DL5 (partially)
- Format validated for Cherenkov telescopes and for Water detectors

Deil, C., *et al.*, Proc. of `Gamma 2016', DOI: 10.1063/1.4969003
Deil, C., *et al.*, Zenodo DOI: 10.5281/zenodo.1409830

Read/Input format

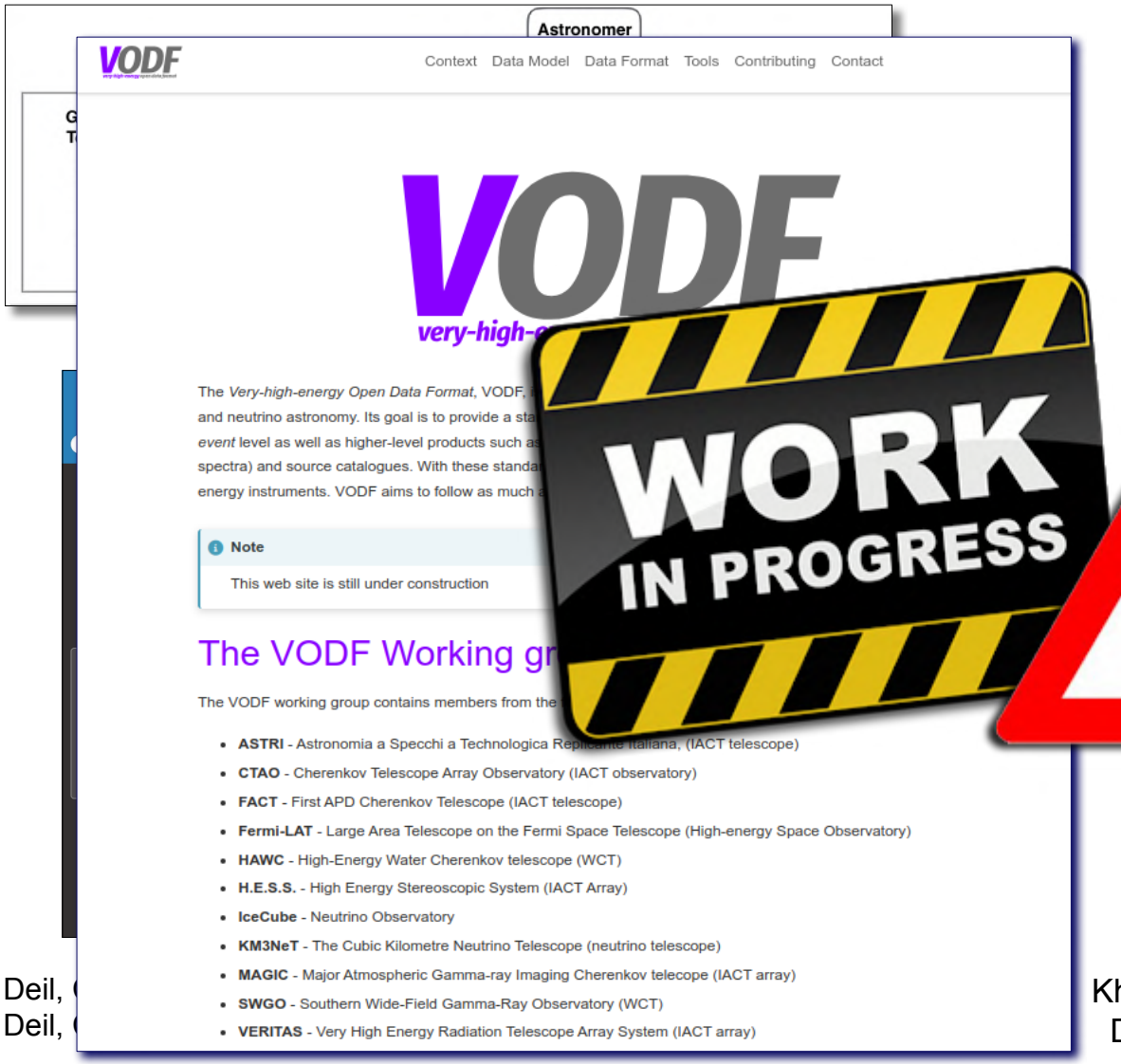


- The HE astroparticle community created an open initiative to create an open format for gamma and neutrino VHE data: Very-high-energy Open Data Format (VODF)
  - 11 major experiments are on-board
- Aim to define 4 levels of data: DL3, DL4, DL5, DL5
  - FAIR compliant
  - Following IVOA recommendations

Deil,
Deil,

Khélifi, B., *et al.*, Proc. of the 38th ICRC, DOI: 10.48550/arXiv.2308.13385

Read/Input format

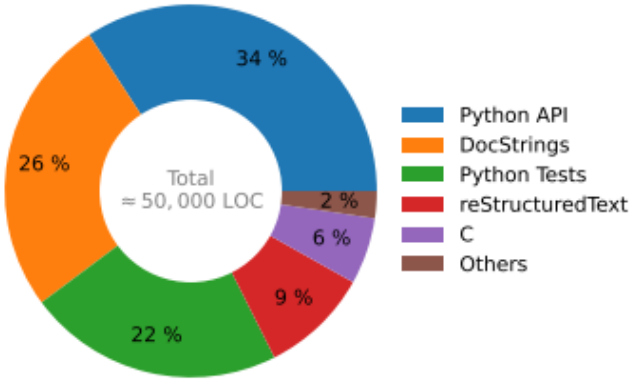


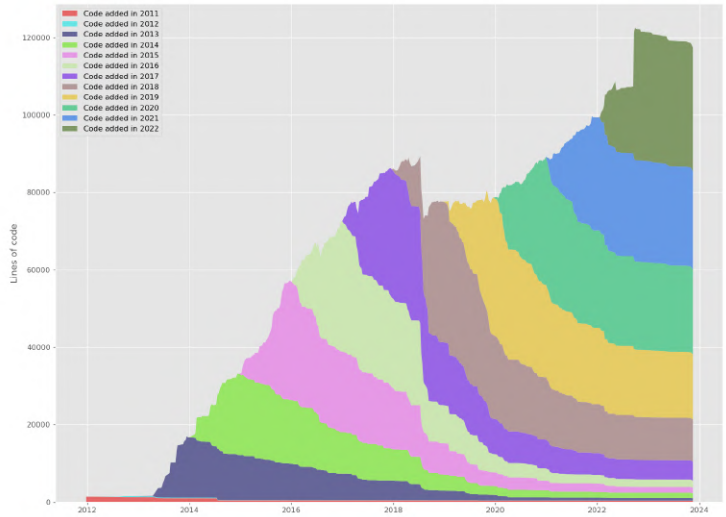
- The HE astroparticle community created an open initiative to create an open format for gamma and neutrino HE data:
  - -high-energy Open Format (VODF)
  - major experiments -board
  - ne 4 levels of 4, DL5, DL5
  - FAIR compliant
  - Following IVOA recommendations

Deil,
Deil,

Khélifi, B., *et al.*, Proc. of the 38th ICRC, DOI: 10.48550/arXiv.2308.13385

Some statistics

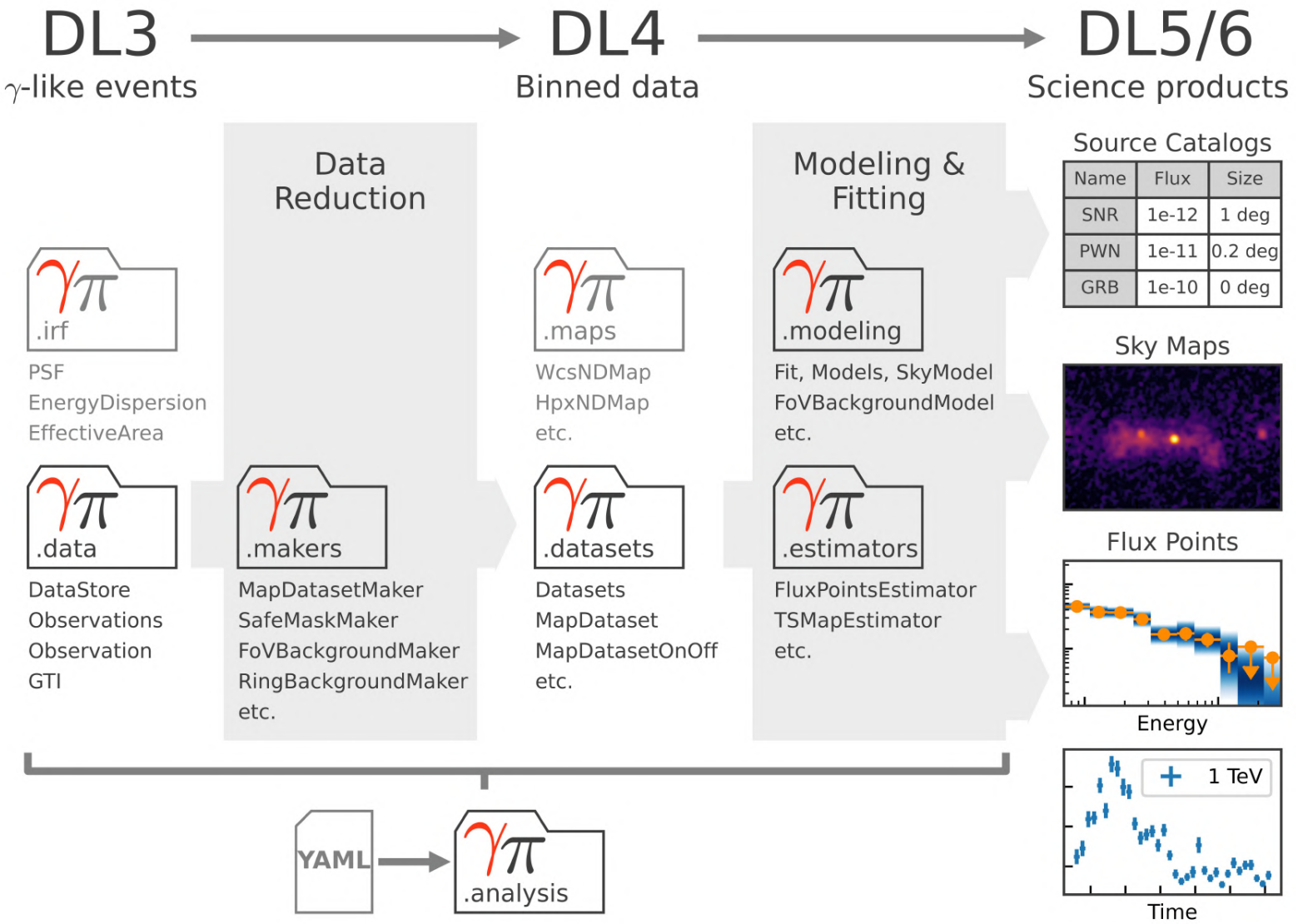


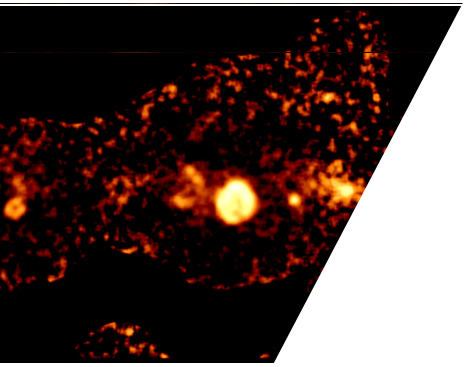


- We have **achieved a healthy distribution of approximately 1/3 code, 1/3 docs and 1/4 tests**

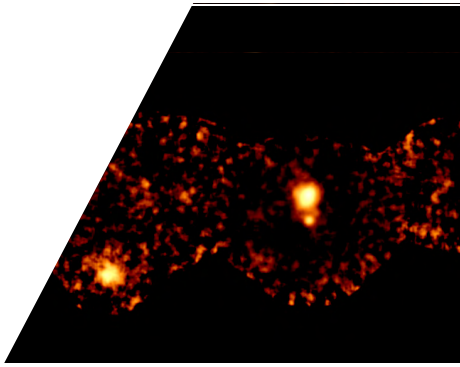
- Since its birth, in Aug. 19th 2013, the team made 33 tags, 24 releases including one long-term supported (LTS v1.0) and 10 Zenodo deposit

Data workflow and package structure





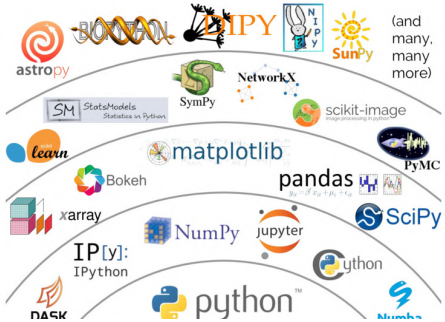
The organization of the Gammapy project



Open Python research software

- All the library and its associated repositories are freely accessible
- Contributions to the code are open to everyone
  - The Gammapy community is wide and diversified
- Written in Python, it benefits from a large ecosystem
  - The project is linked with many other software projects

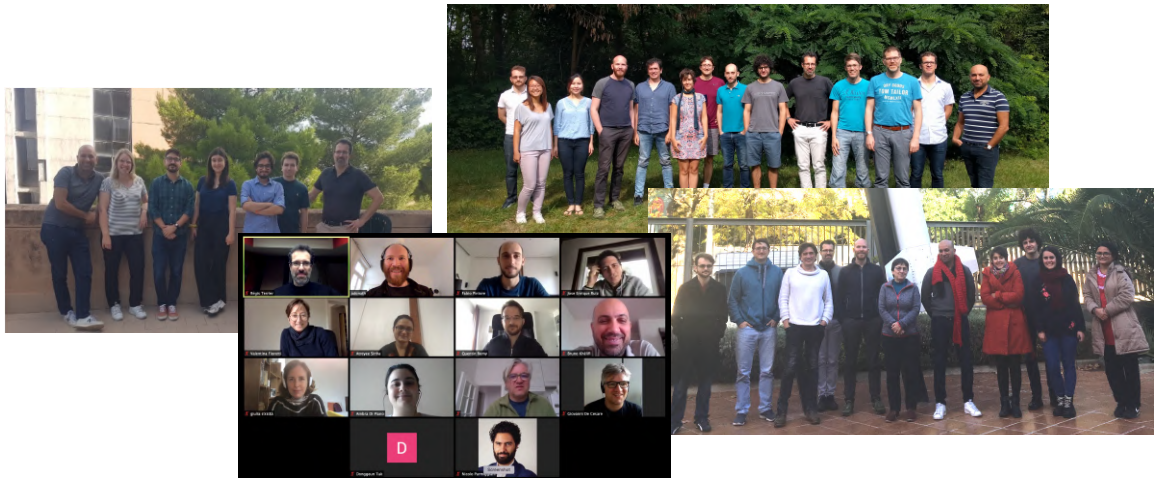
Python in science today



- Although without legal structure, the project is supported by many laboratories, institutes and university
- And as the core library of the CTA Science Analysis Tool, it is widely used by students and researchers
  - An active user support is in place

Well structured project

The teams of developers



Coordination Committee

Project Management

B. Khélifi
(APC)

C. van Eldik
(ECAP)

Lead Development

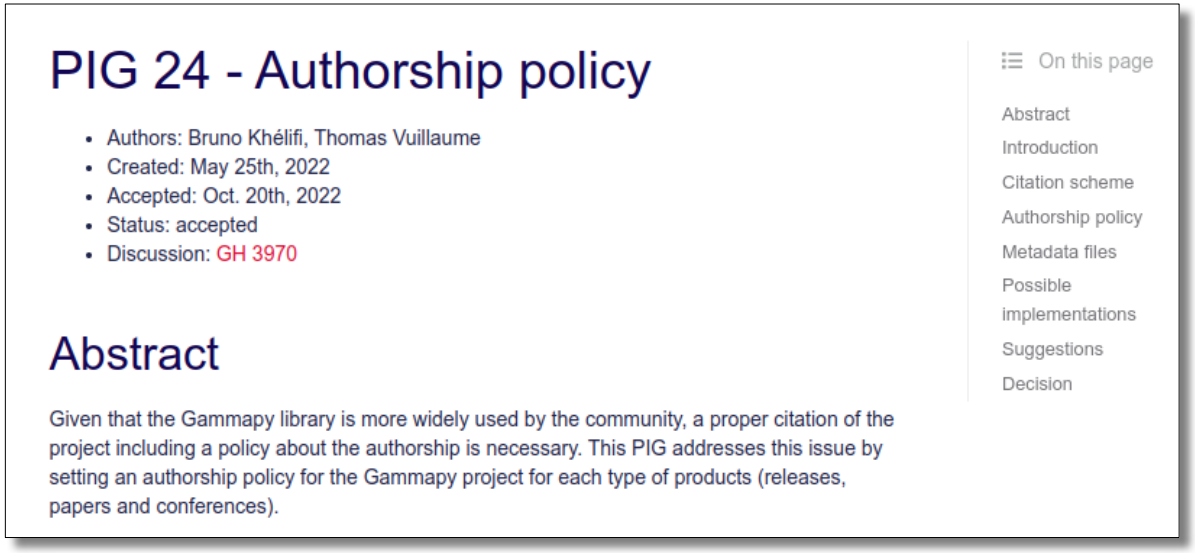
R. Terrier
(APC)

A. Donath
(CfA)

Development scheme

- Development repository:
  https://github.com/gammapy/
  - GitHub actions to run the continuous integration
  - Pytest used for testing
  - Black, isort and Flake 8 to respect Python standards
- Documentation: https://docs.gammapy.org/
- Contributions
  - Code, test functions and documentation
  - Pull request reviews by at least 2 persons
- Developer call each Friday at 2pm (CET)

Recognition and valorization



PIG 24 - Authorship policy

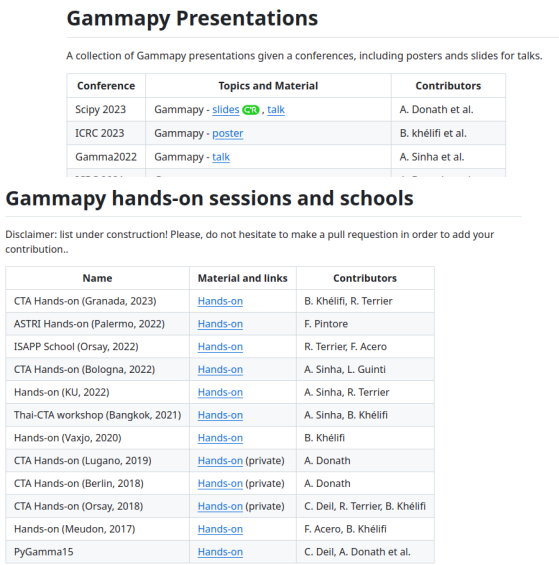
- Authors: Bruno Khélifi, Thomas Vuillaume
- Created: May 25th, 2022
- Accepted: Oct. 20th, 2022
- Status: accepted
- Discussion: GH 3970

On this page

Abstract
Introduction
Citation scheme
Authorship policy
Metadata files
Possible implementations
Suggestions
Decision

Abstract

Given that the Gammapy library is more widely used by the community, a proper citation of the project including a policy about the authorship is necessary. This PIG addresses this issue by setting an authorship policy for the Gammapy project for each type of products (releases, papers and conferences).



Gammapy Presentations

A collection of Gammapy presentations given a conferences, including posters ands slides for talks.

Conference	Topics and Material	Contributors
Scipy 2023	Gammapy - slides, talk	A. Donath et al.
ICRC 2023	Gammapy - poster	B. khélifi et al.
Gamma2022	Gammapy - talk	A. Sinha et al.

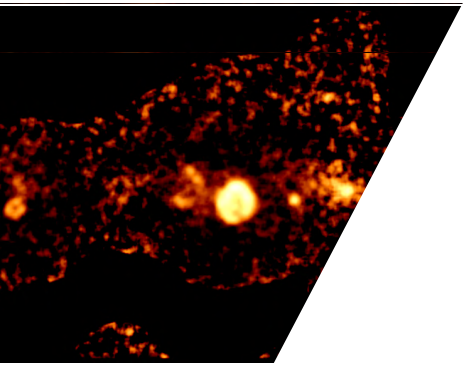
Gammapy hands-on sessions and schools

Disclaimer: list under construction! Please, do not hesitate to make a pull requestion in order to add your contribution..

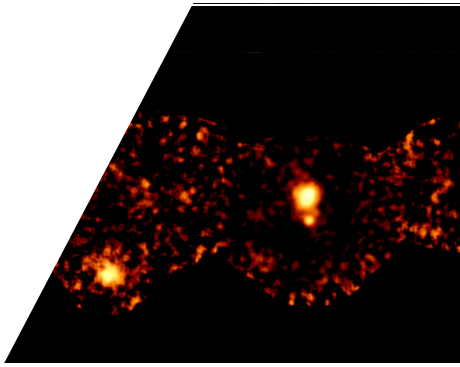
Name	Material and links	Contributors
CTA Hands-on (Granada, 2023)	Hands-on	B. Khélifi, R. Terrier
ASTRI Hands-on (Palermo, 2022)	Hands-on	F. Pintore
ISAPP School (Orsay, 2022)	Hands-on	R. Terrier, F. Acero
CTA Hands-on (Bologna, 2022)	Hands-on	A. Sinha, L. Guinti
Hands-on (KU, 2022)	Hands-on	A. Sinha, R. Terrier
Thai-CTA workshop (Bangkok, 2021)	Hands-on	A. Sinha, B. Khélifi
Hands-on (Vaxjo, 2020)	Hands-on	B. Khélifi
CTA Hands-on (Lugano, 2019)	Hands-on (private)	A. Donath
CTA Hands-on (Berlin, 2018)	Hands-on (private)	A. Donath
CTA Hands-on (Orsay, 2018)	Hands-on (private)	C. Deil, R. Terrier, B. Khélifi
Hands-on (Meudon, 2017)	Hands-on	F. Acero, B. Khélifi
PyGamma15	Hands-on	C. Deil, A. Donath et al.

- Each Gammapy release is an official publication
  - The SWH/ZenodoDOI has an author list
- Technical papers on developments are very encouraged
- Each presentation, hands-on session, school is promoted

Following the Open Science recommendation about academic evaluation

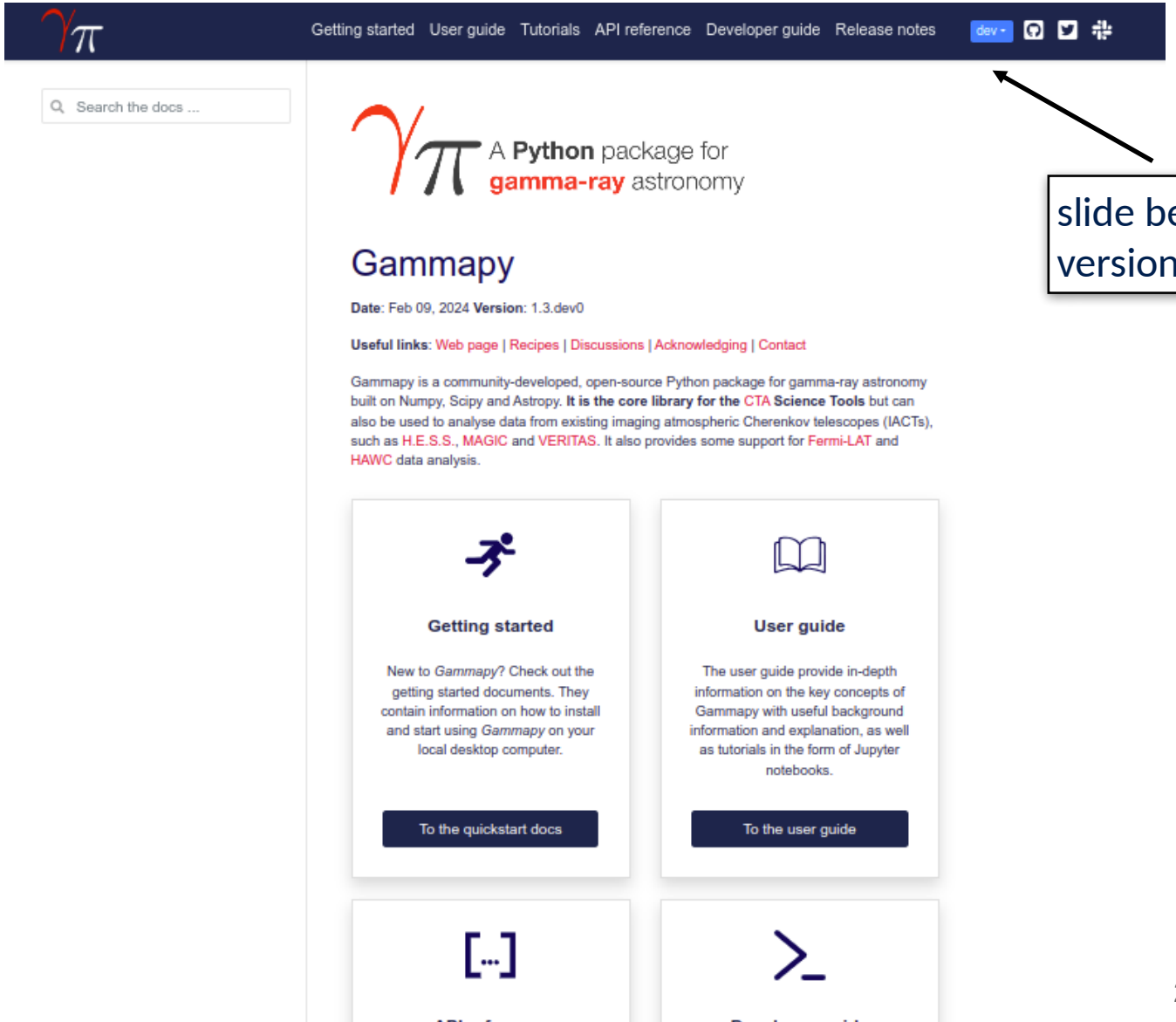


The documentation of Gammapy



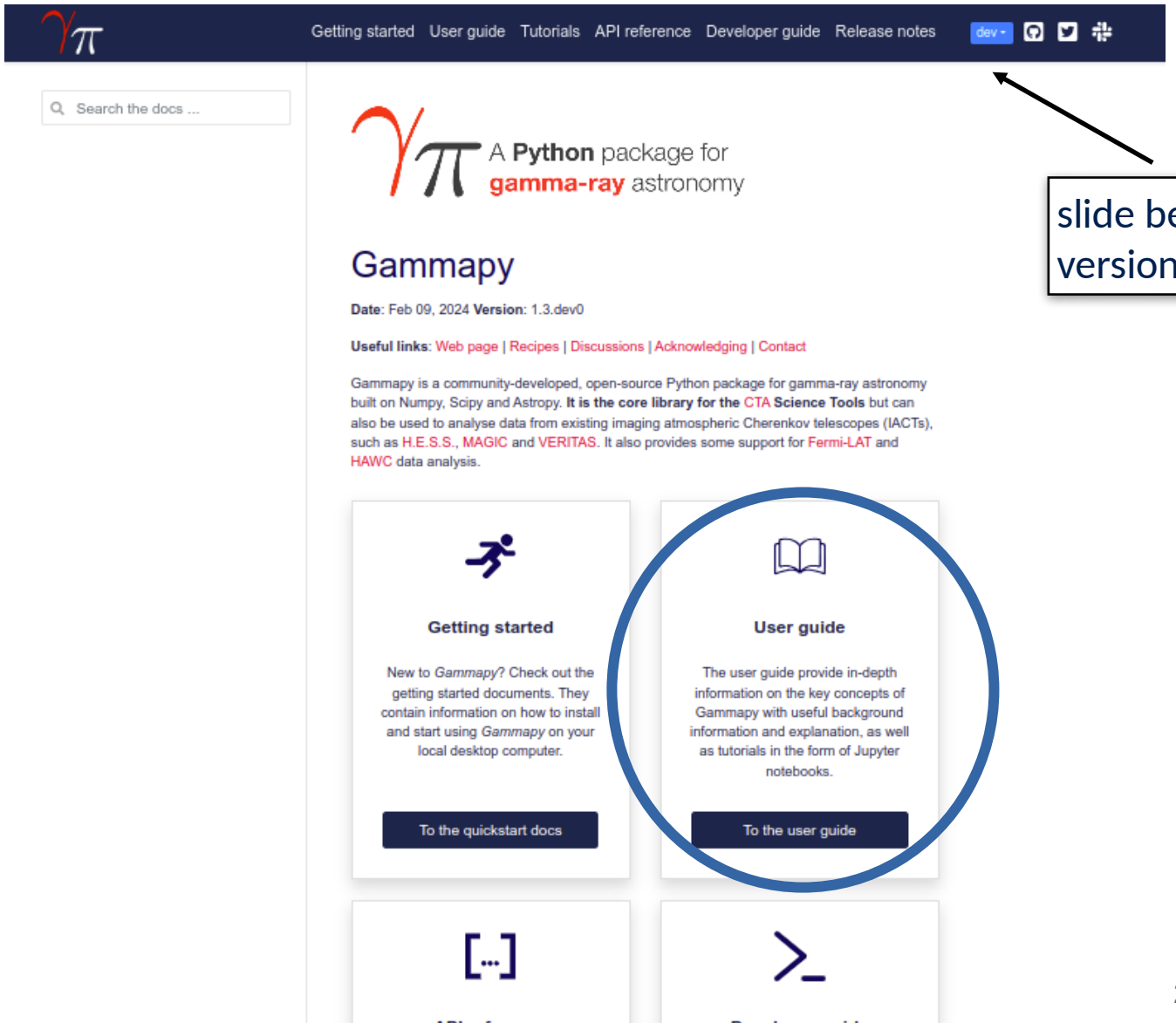
See docs.gammapy.org

Getting started: documentation



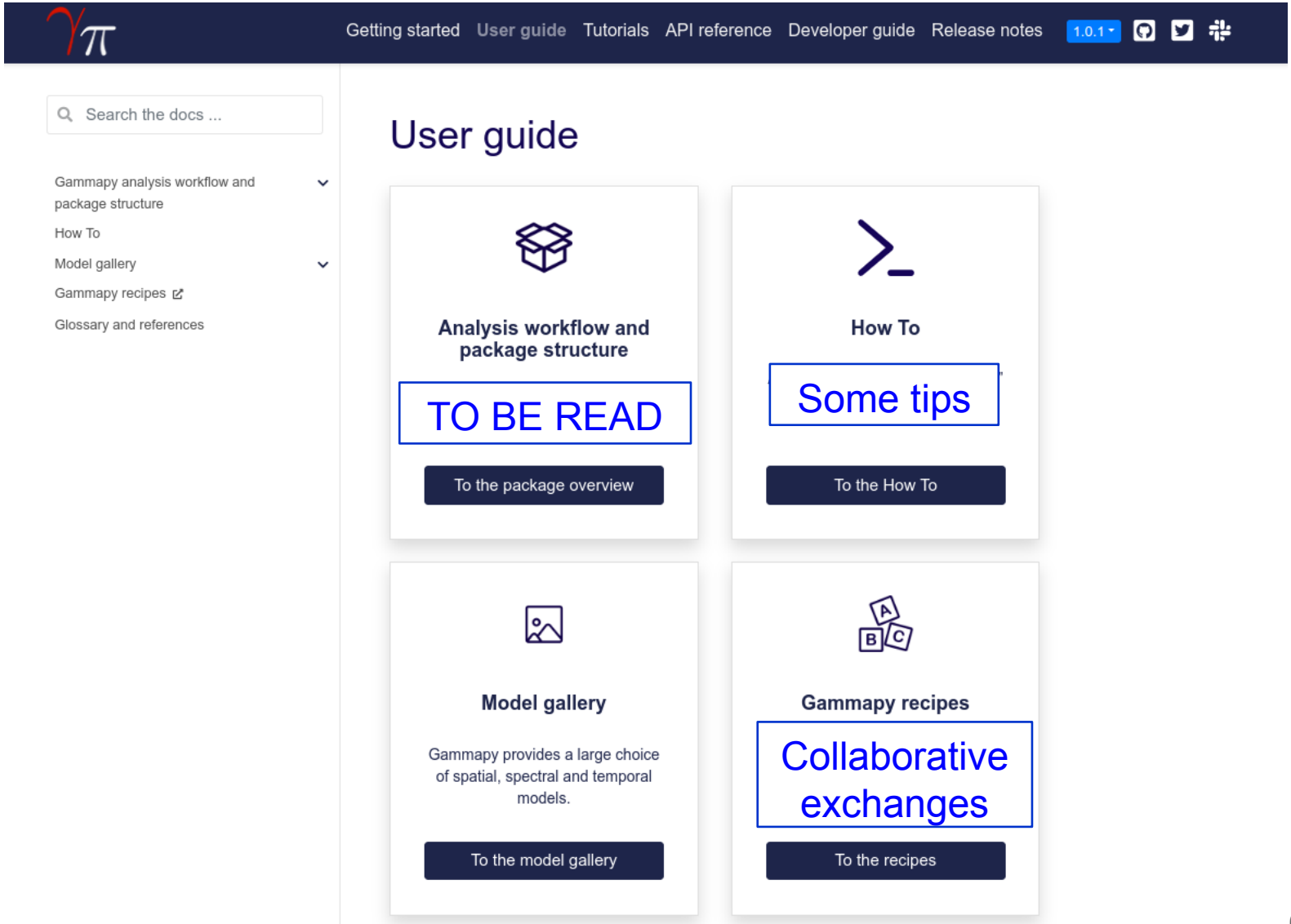
slide between versions

Getting started: documentation

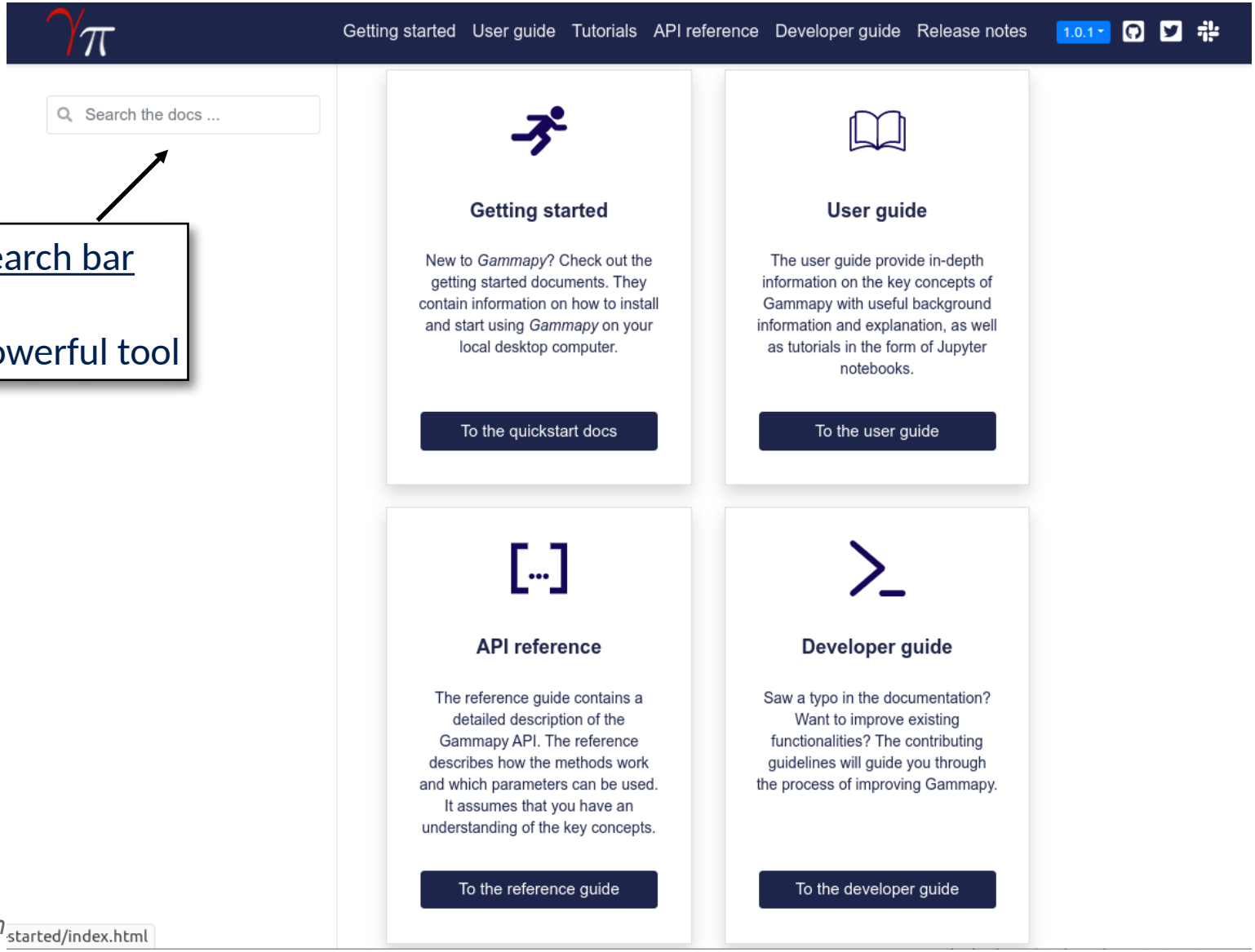


slide between versions

Getting started: documentation

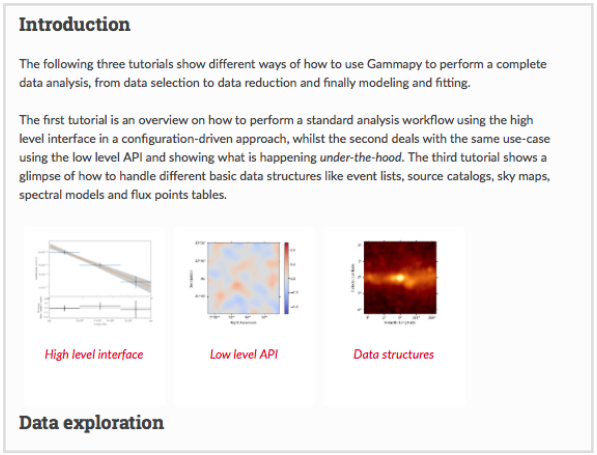


Getting started: documentation

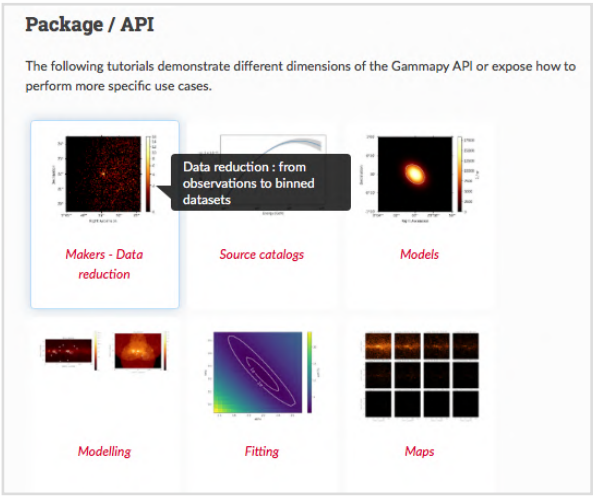


Getting started: documentation

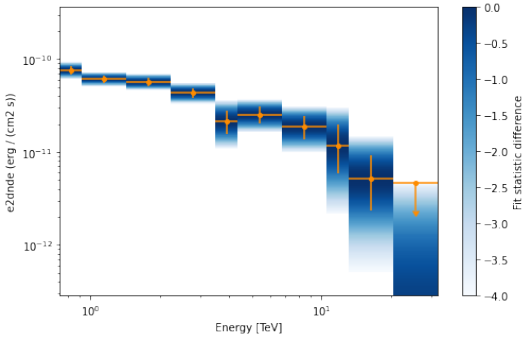
- Learning with examples: the Tutorials



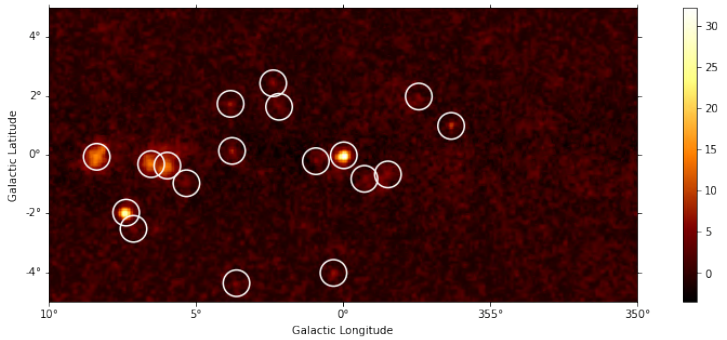
- More in depth: the API description



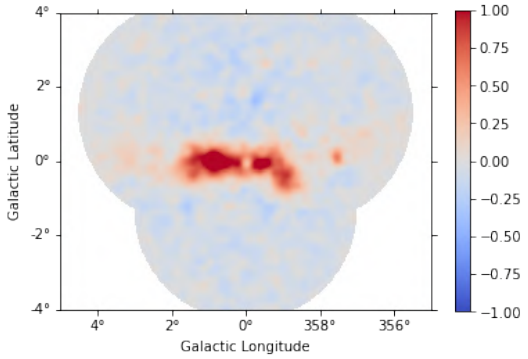
Typical analysis use cases



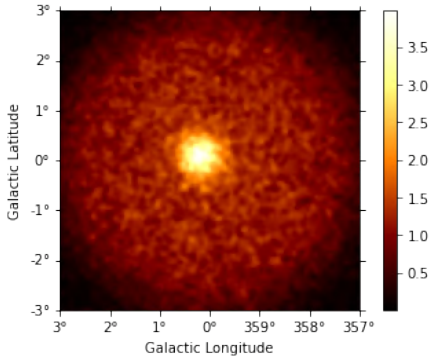
[1D spectral analysis]



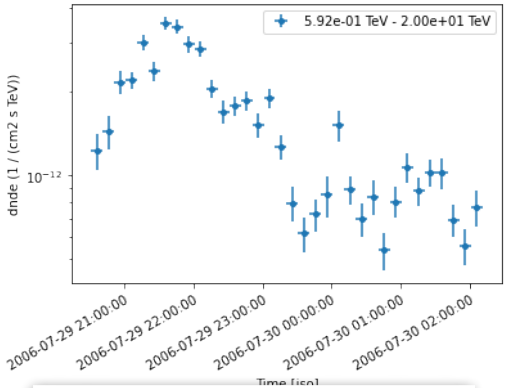
[Source detection]



[3D analysis]



[Observation simulation]

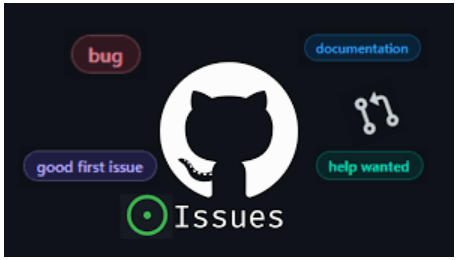


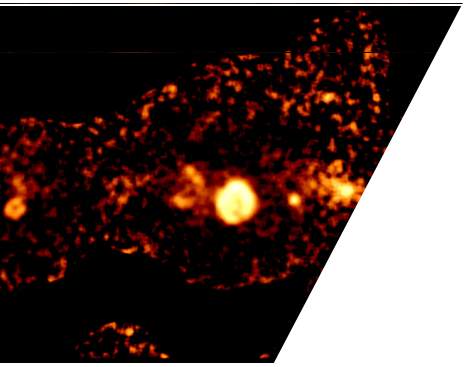
[Light-curve extraction]

All analysis types follow the same workflow and the same API

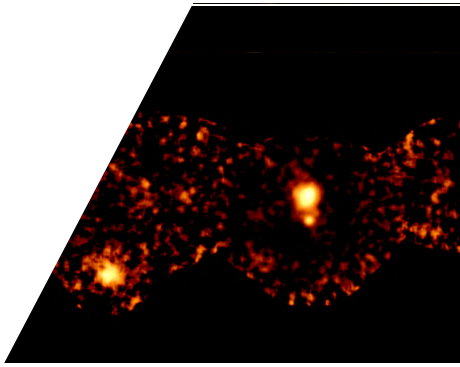
Getting help

- Where/How to interact with dev team and experienced users, provide feedback, get help:
  - gammapy.slack
    - In particular: #help channel
  - GitHub discussions
    - help category
  - GitHub issues to report bugs or feature requests

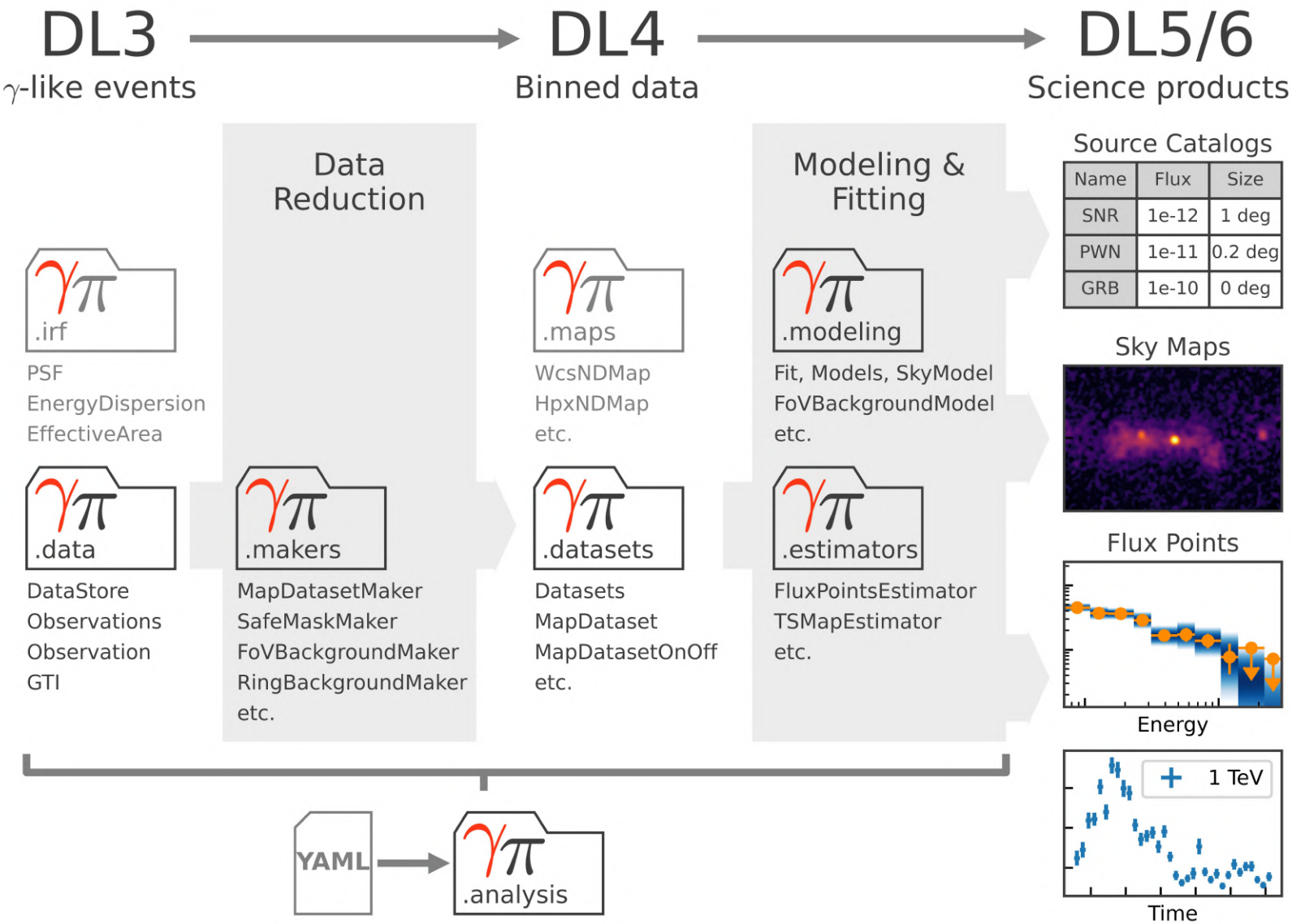




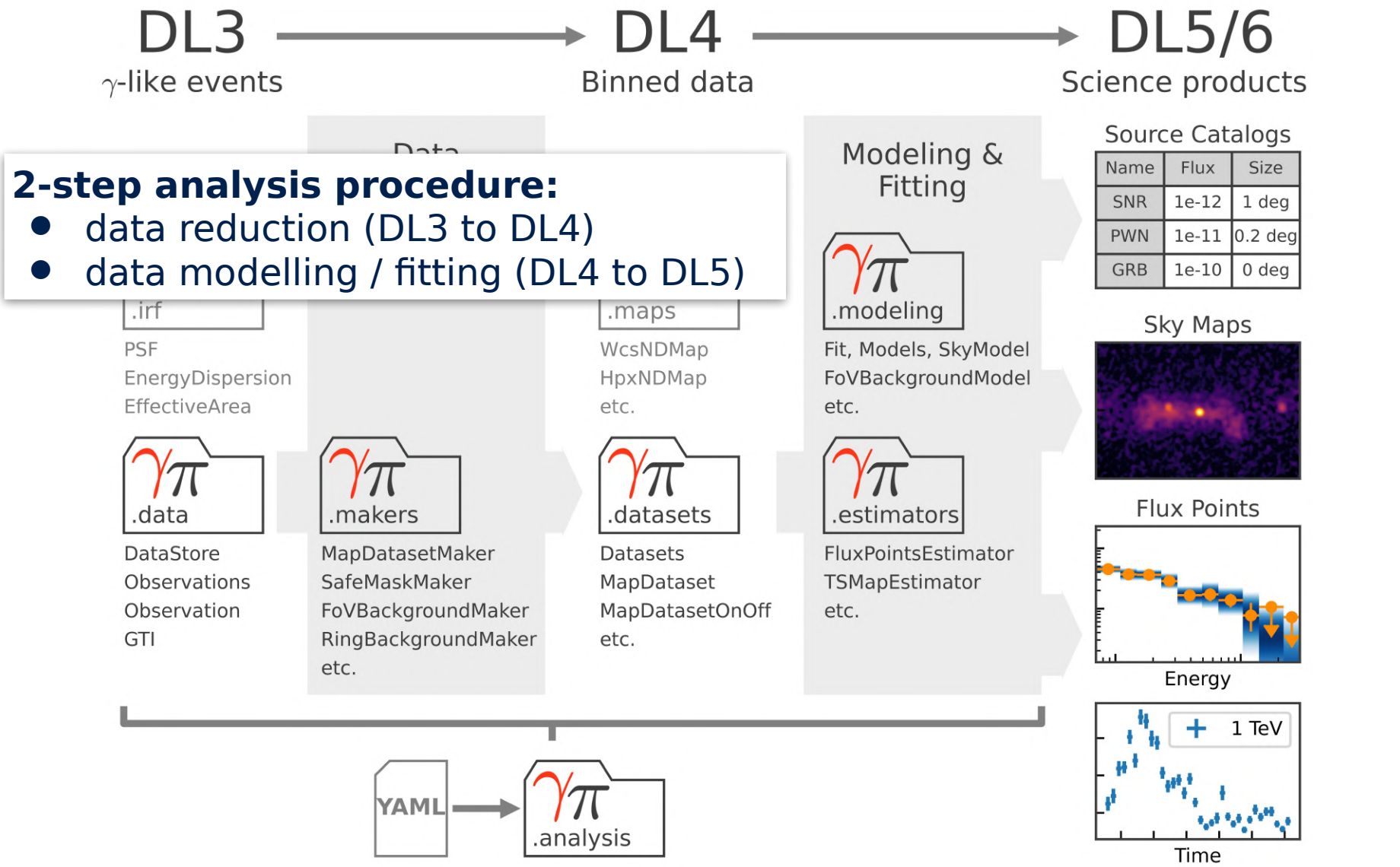
The Gammapy workflow



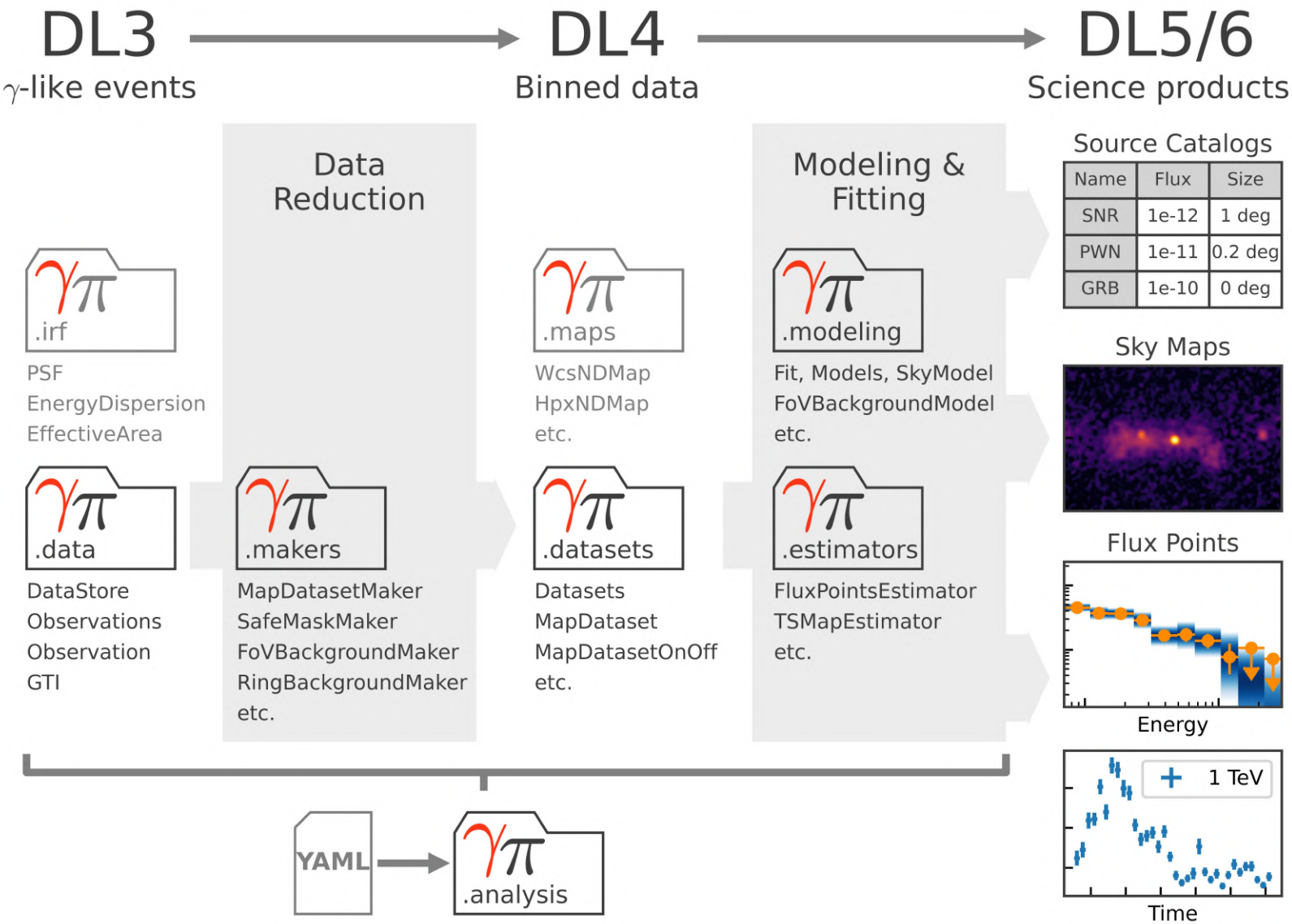
Data workflow and package structure

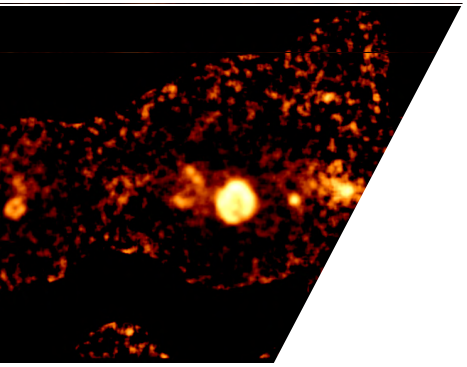


Data workflow and package structure

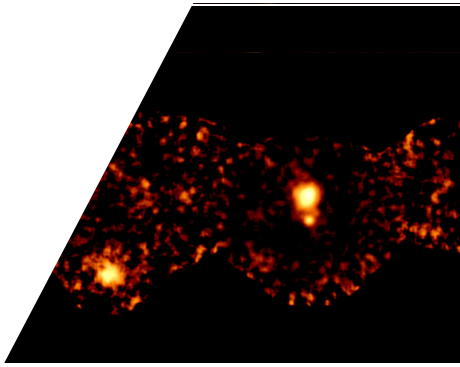


Data workflow and package structure





From DL3 to DL4



The basic analysis steps

DL3 to DL4

1. Select and retrieve observations
   - `Datastore` → list of `Observation`

See the backup slides...

DL4 to DL5

The basic analysis steps

DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
   - Is the analysis 1D or 3D?
   - Define target binning and projection

See the backup slides...

DL4 to DL5

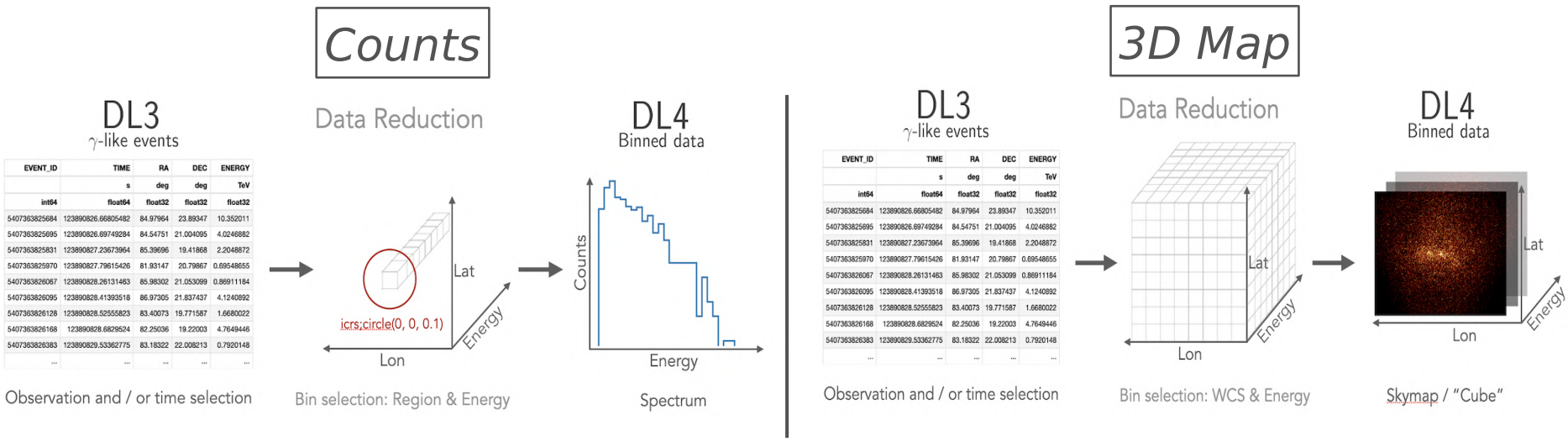
The basic analysis steps

DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
   - Data and IRFs projection
   - Safe Mask determination
   - Background estimation

DL4 to DL5

Data and IRF re-projection



SpectrumDatasetMaker

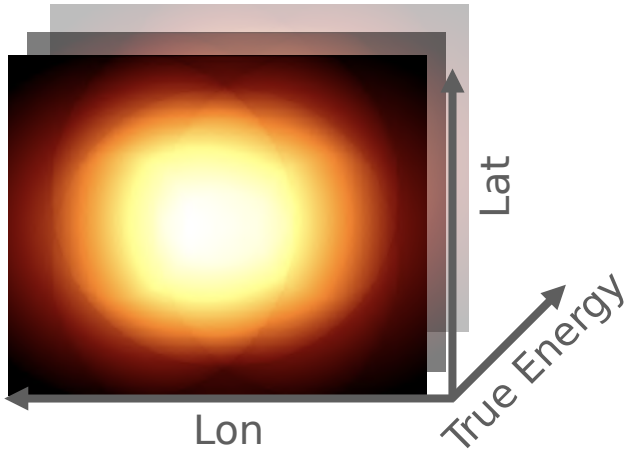
MapDatasetMaker

```
empty = MapDataset.create(
    geom,
    energy_axis_true=energy_true,
    name="my-dataset")
maker = MapDatasetMaker(selection=["exposure", "background", "psf", "edisp"])
dataset = maker.run(empty, observation)
```

Reduced IRFs

- DL3 IRFs are reprojected by the `DatasetMaker` on the target geometry

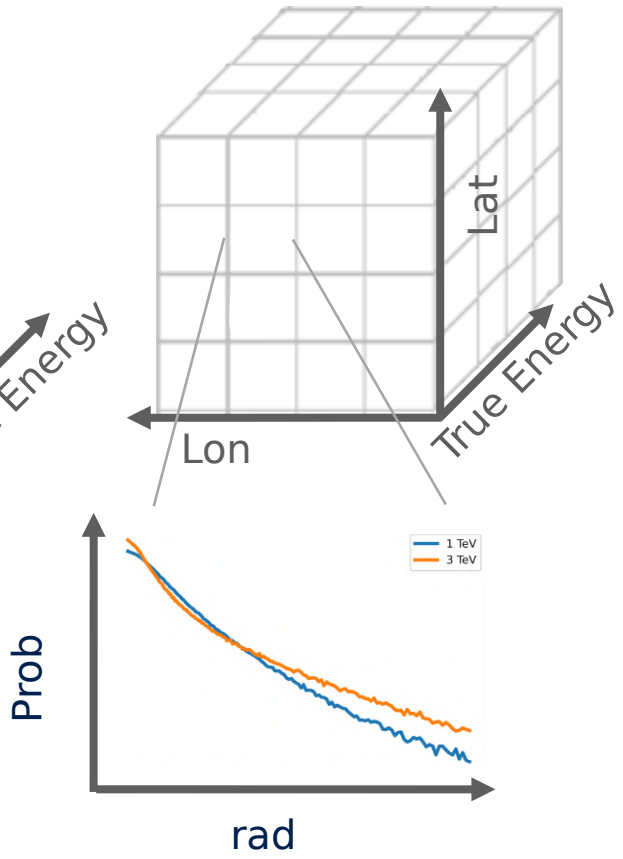
Exposure



`PSFMap` / `EDispKernelMap`

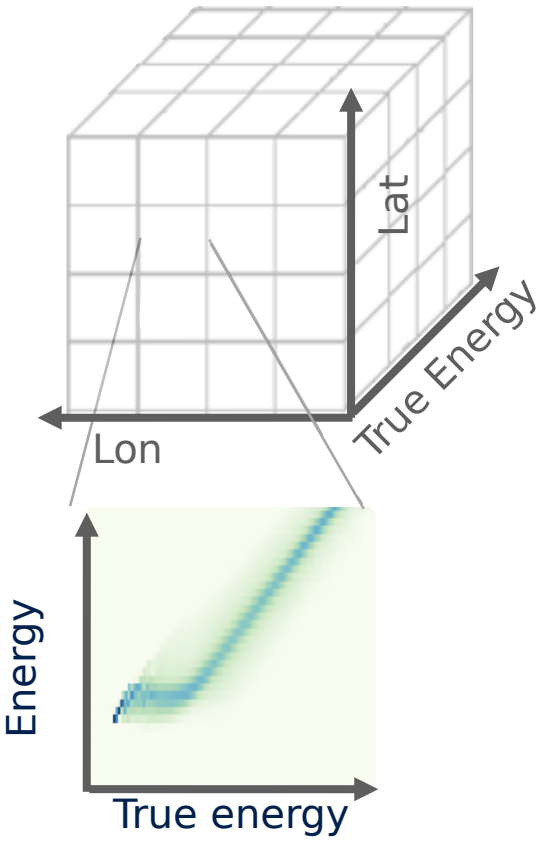
PSF

4th Dimension: rad



EDisp

4th Dimension: Energy



The basic analysis steps

DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
   - Data and IRFs projection
   - Safe Mask determination
   - Background estimation

DL4 to DL5

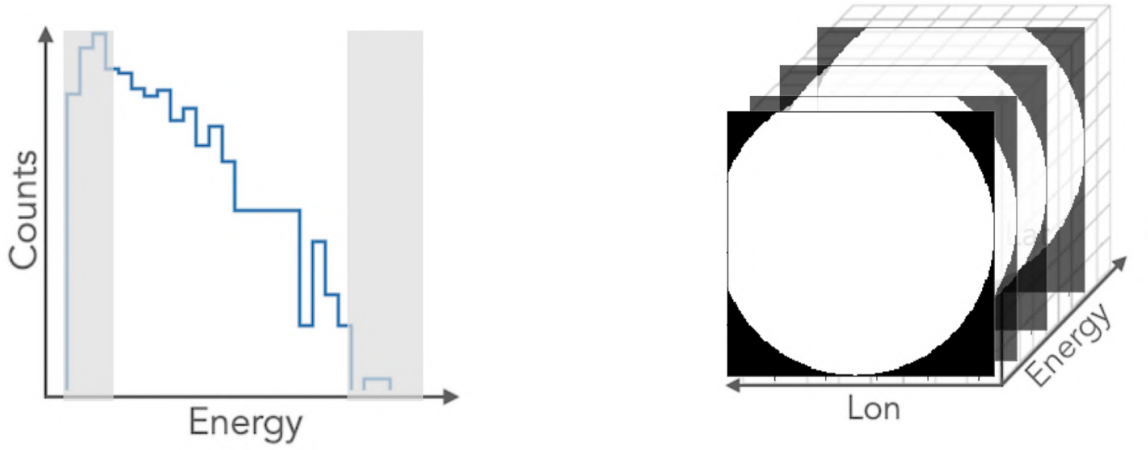
SafeMask concept

It allows to restrict the analysis bins because of any of these reasons:

- restriction the phase space (statistics, sources)
- validity range of the IRFs (systematics)
- scaling of the bkg model to the data

SafeMaskMaker

SafeMask



The basic analysis steps

DL3 to DL4

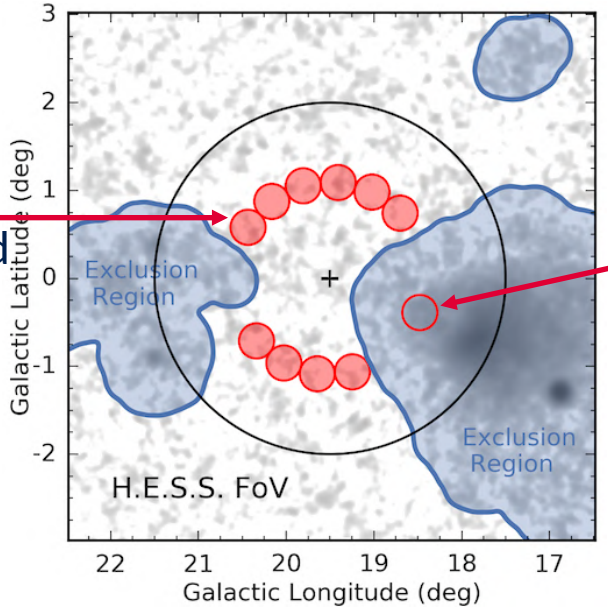
1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
   - Data and IRFs projection
   - Safe Mask determination
   - Background estimation

DL4 to DL5

Correcting background from data

- To further reduce systematics or for the 1D analysis (spectrum), the background is sometimes measured directly in the data, e.g. in regions of the FoV where the background is assumed to be identical
  - Common approach used for 1D spectral analysis
  - e.g. reflected regions or wobble regions background

OFF regions
containing only background



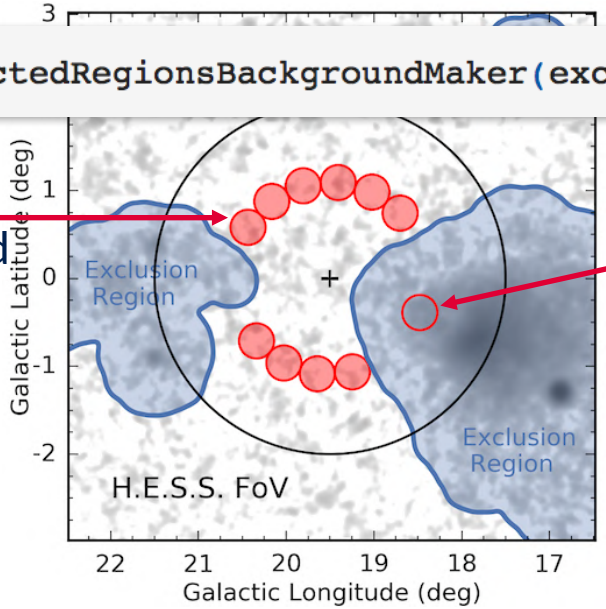
ON region
containing signal and background

Correcting background from data

- To further reduce systematics or for the 1D analysis (spectrum), the background is sometimes measured directly in the data, e.g. in regions of the FoV where the background is assumed to be identical
  - Common approach used for 1D spectral analysis
  - e.g. reflected regions or wobble regions background

```
bkg_maker = ReflectedRegionsBackgroundMaker(exclusion_mask=exclusion_mask)
```

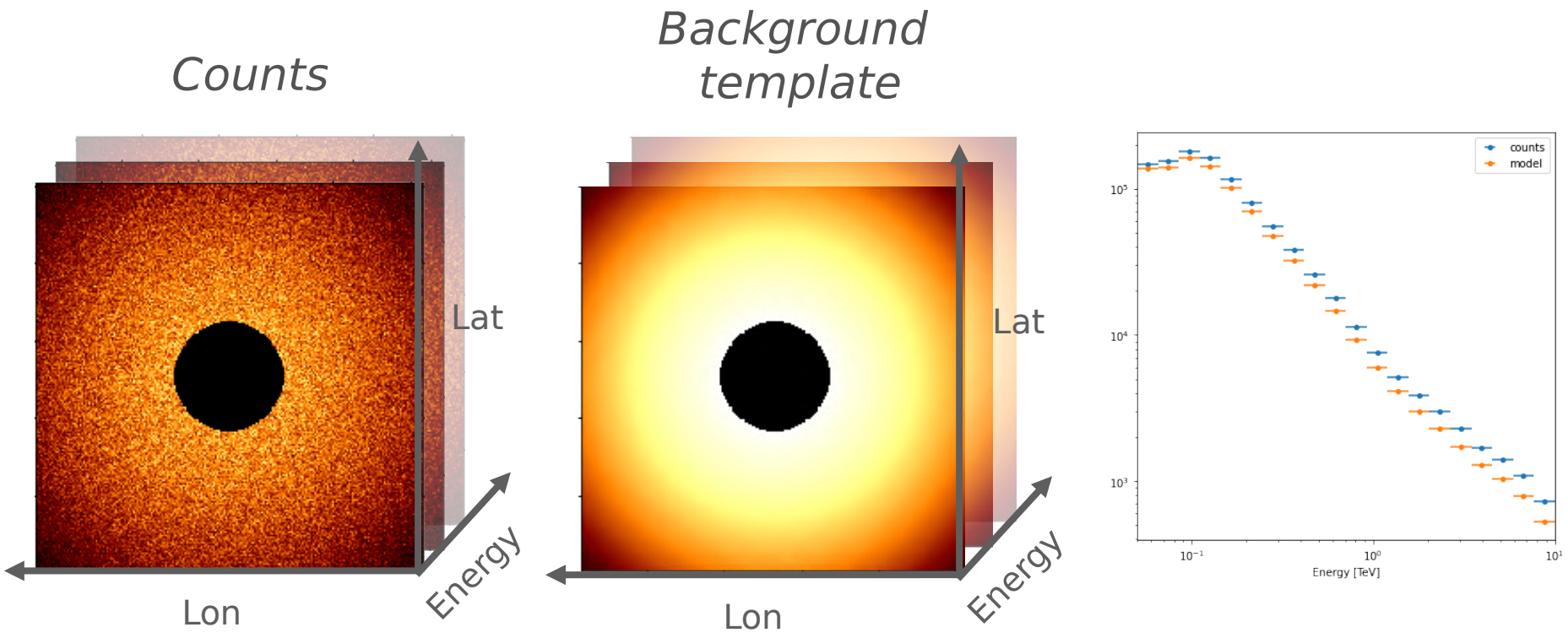
OFF regions
containing only background



ON region
containing signal and background

Correcting background from data

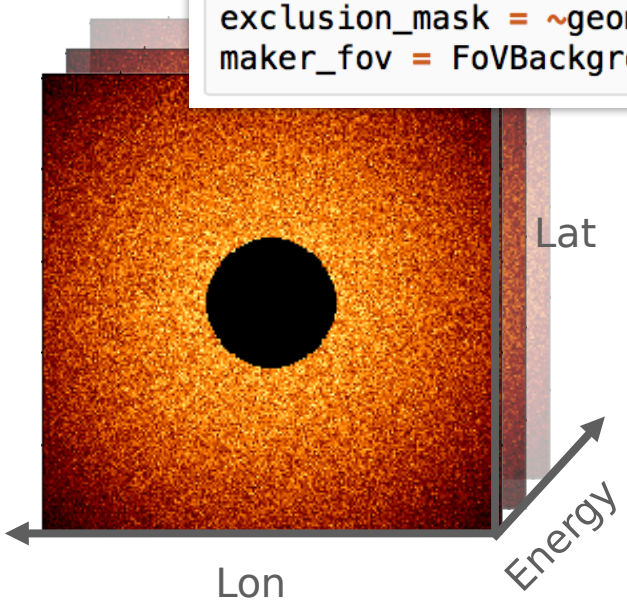
- *BKG(**p**, E)* is usually corrected on the observed data themselves.
  - Field of View (FoV) background estimation: `FoVBackgroundModel` adjusted with the `FoVBackgroundMaker`
    - *BKG(p, E)* is normalized in regions devoid of signal

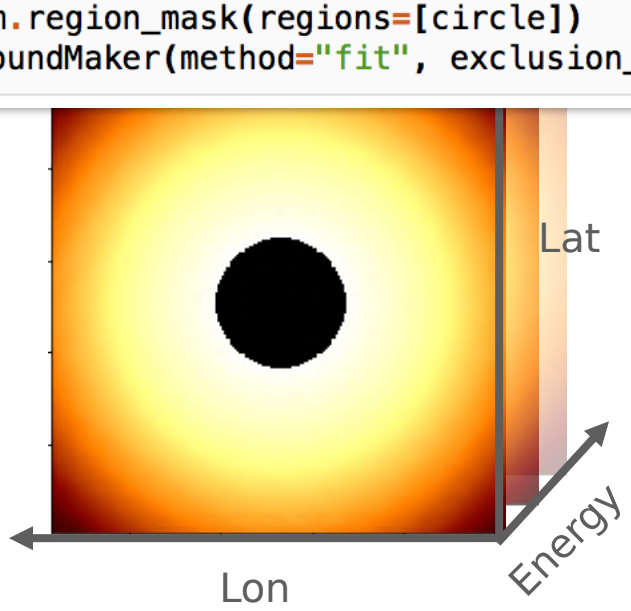


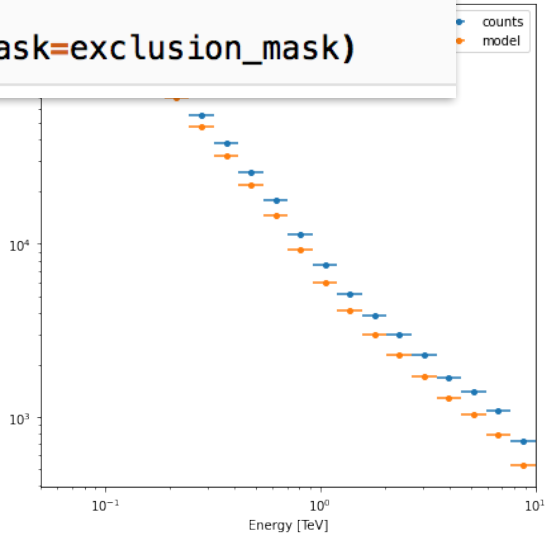
Correcting background from data

- *BKG(p, E)* is usually corrected on the observed data themselves.
  - Field of View (FoV) background estimation
    - *BKG(p, E)* is normalized in regions devoid of signal

```
circle = CircleSkyRegion(
    center=SkyCoord("83.63 deg", "22.14 deg"), radius=0.2 * u.deg
)
exclusion_mask = ~geom.region_mask(regions=[circle])
maker_fov = FoVBackgroundMaker(method="fit", exclusion_mask=exclusion_mask)
```







The basic analysis steps

DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
4. Loop over selected observations
   - Apply makers to produce reduced datasets
   - Combine them for stacked or joint analysis

DL4 to DL5

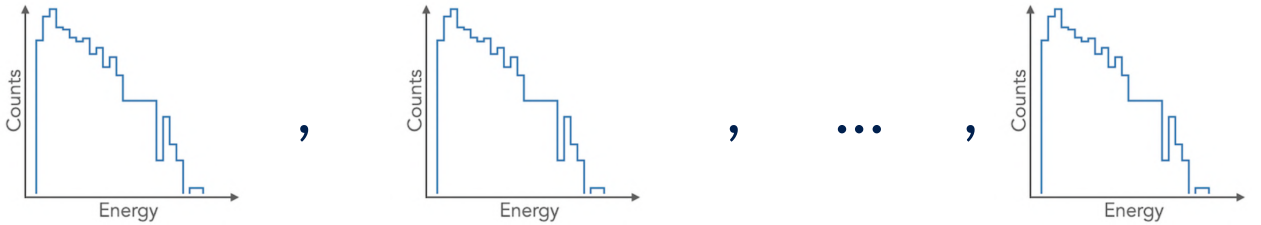
Loop over observations

- Apply makers to produce reduced datasets
- Combine them for stacked or joint analysis

Obs 1 + Obs 2 + ... + Obs n = Stacked Dataset

Dataset 1 Dataset 2 Dataset n

[Obs 1 , Obs 2 , ... , Obs n] = Datasets



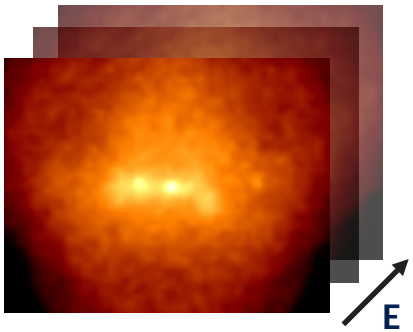
Dataset 1 Dataset 2 Dataset n

The basic analysis steps: overview of the Data Reduction

DL3 to DL4

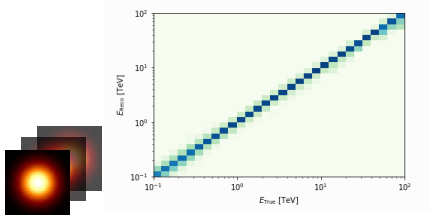
1. Select and retrive observations
   - `Datastore` → list of `Observation`
2. Define the reduced dataset geometry
   - Is the analysis 1D or 3D?
   - Define target binning and projection
3. Initialize the data reduction methods (makers)
   - Data and IRF projection
   - Safe Mask determination
   - Background estimation
4. Loop over selected observations
   - Apply makers to produce reduced datasets
   - Combine them for stacked or joint analysis

DL4 structures: Datasets



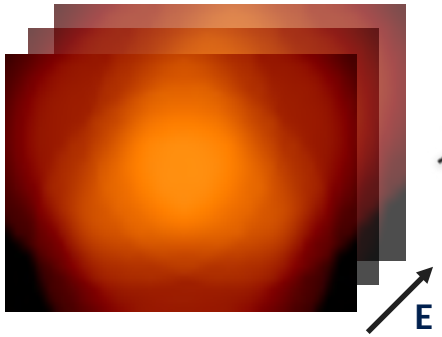
Reduced data

I/O



Reduced IRFs

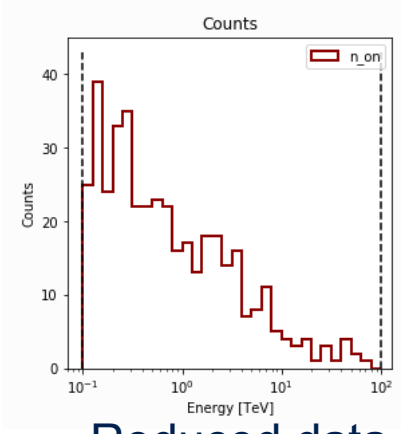
MapDataset



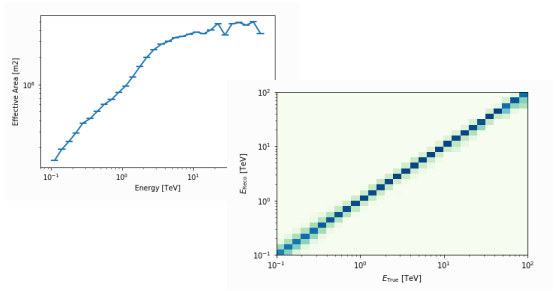
Bkg model

see: Dataset API tutorial

DL4 structures: Datasets



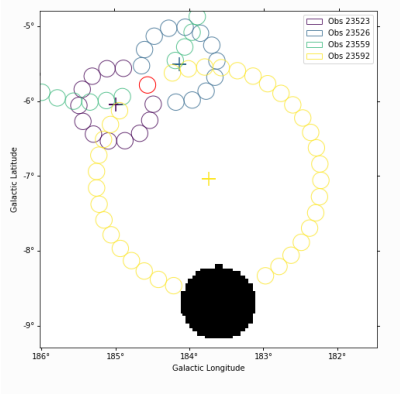
Reduced data



Reduced IRFs

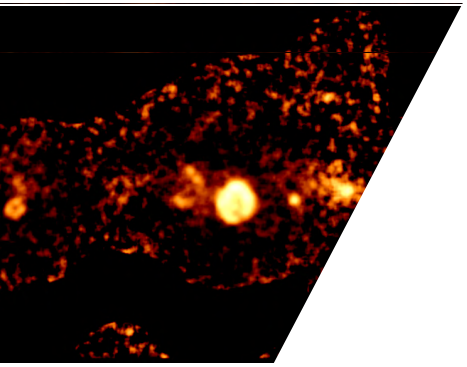
I/O

SpectrumDataset

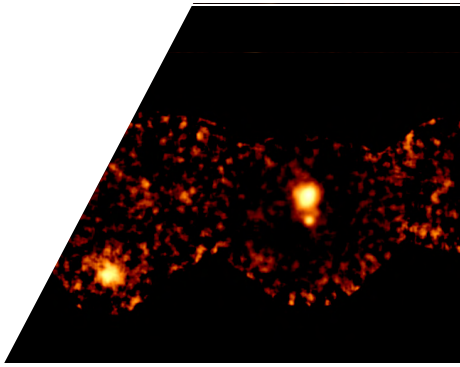


Bkg data or model

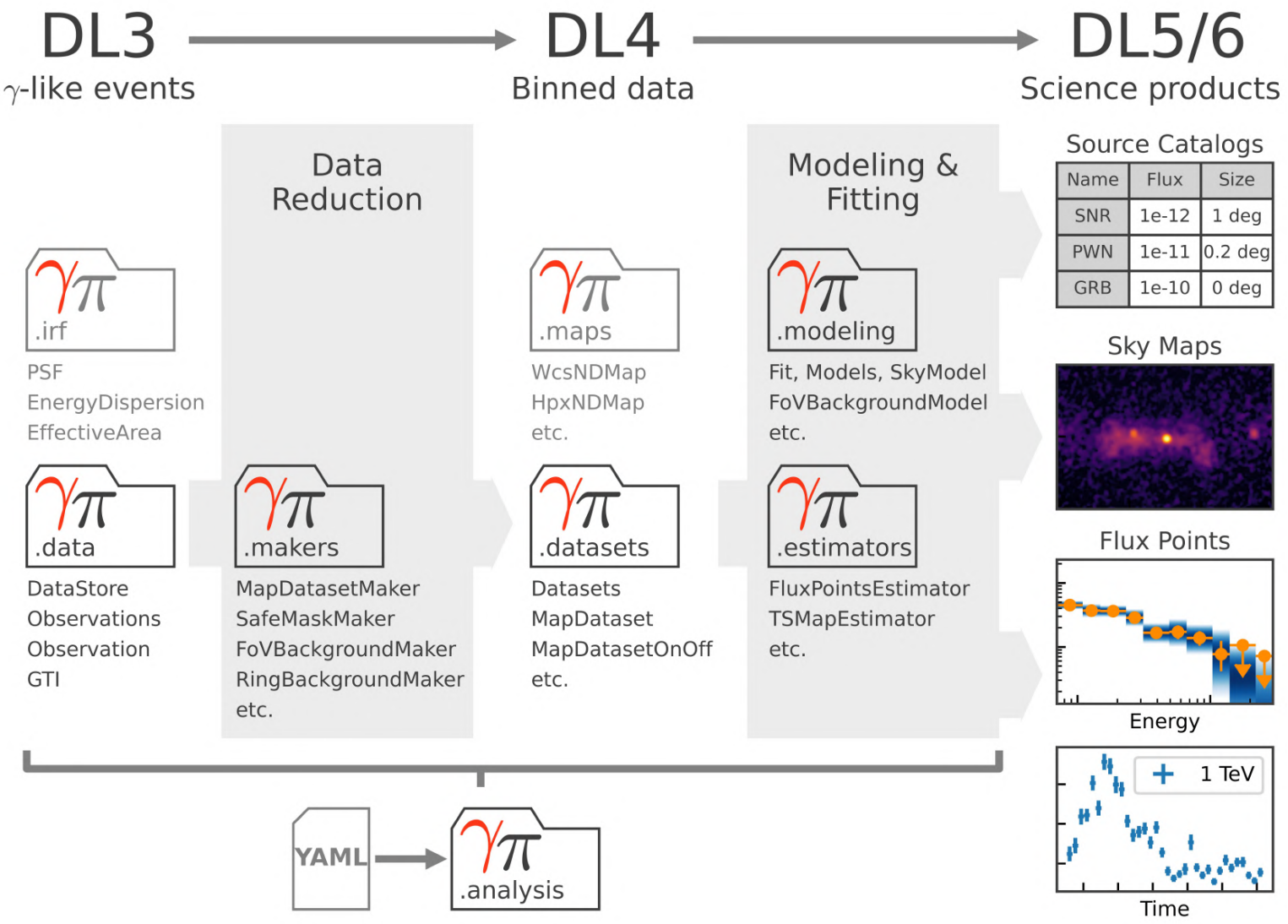
see: Dataset API tutorial



From DL4 to DL5



Data workflow and package structure



DL4 to DL5: Modelling and fitting

- For modelling and fitting, Gammapy relies on ***forward-folding***:
  - Measured counts is compared to predicted counts

- Model parameter estimation is performed through maximum likelihood technique:
  - Cash statistics is used for counts data with a known background
  - Wstat statistics is used for counts data with a measured background

The basic analysis steps

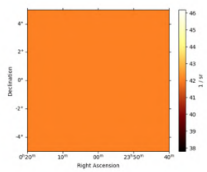
DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
4. Loop over selected observations

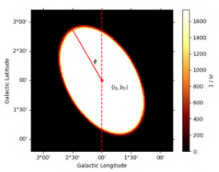
DL4 to DL5

1. Modelling
   - Define your model(s)
     - For the 3D analysis, add a final `FoVBackgroundModel`
   - Associate them/it to the correct dataset (or several)

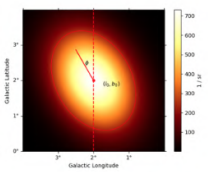
Datasets modelling and fitting



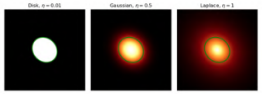
Constant spatial model



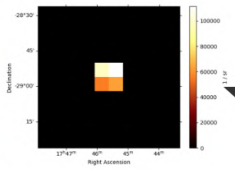
Disk spatial model



Gaussian spatial model



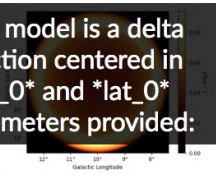
Generalized gaussian spatial model



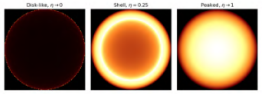
This model is a delta function centered in *lon_0* and *lat_0* parameters provided:

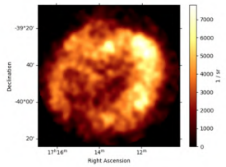
¶

Point spatial model

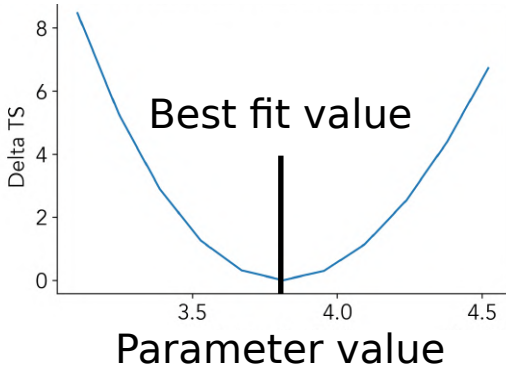


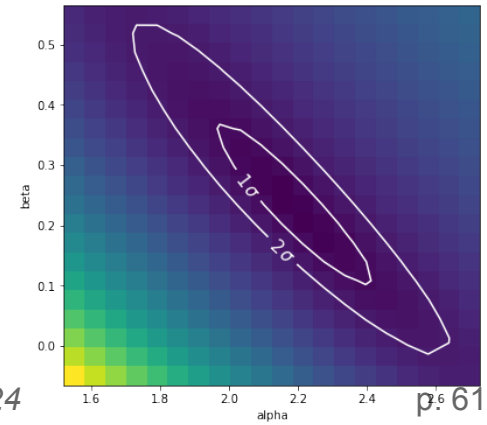
Shell spatial model



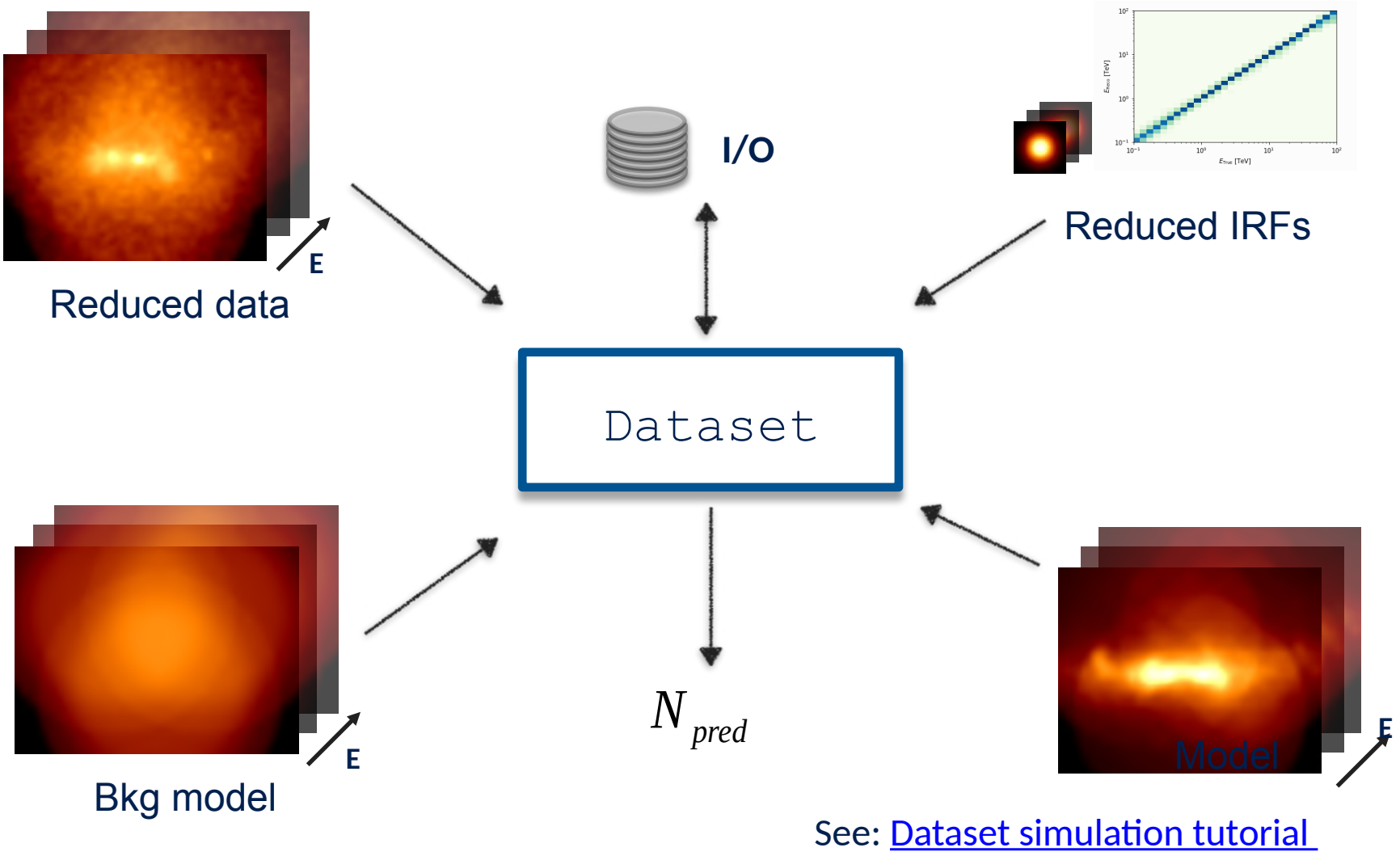


A [library of models](#) and a [Fitting](#) interface



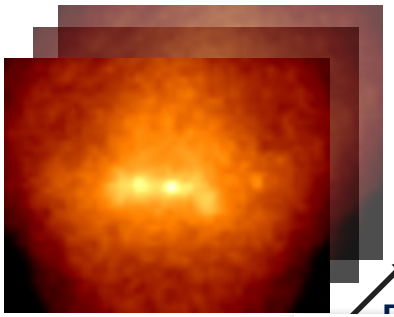


Datasets modelling and fitting



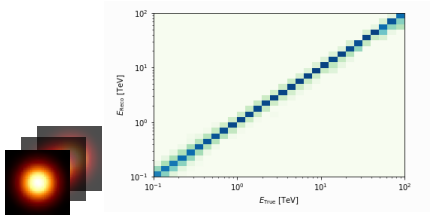
See: Dataset simulation tutorial

Datasets modelling and fitting



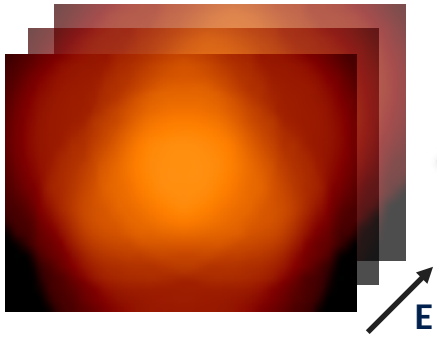
Reduced d

I/O



Reduced IRFs

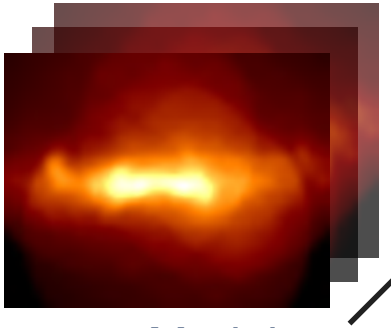
```
# Add the model on the dataset and Poission fluctuate
dataset.models = models
dataset.fake()
# Do a print on the dataset - there is now a counts maps
print(dataset)
```



E

Bkg model

N_{pred}

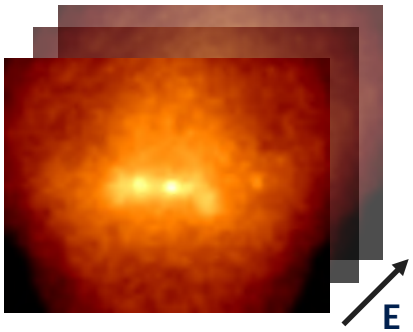


E

Model

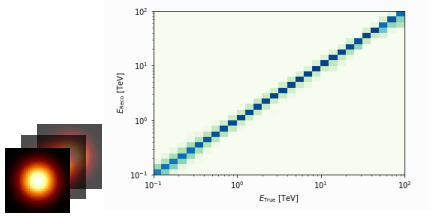
See: Dataset simulation tutorial

Datasets modelling and fitting



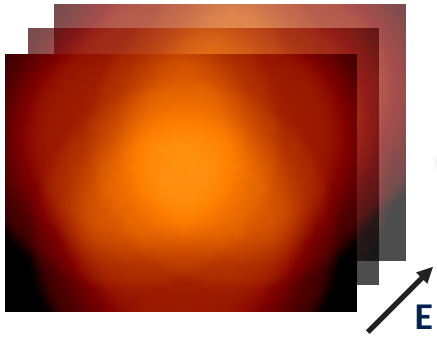
Reduced data

I/O



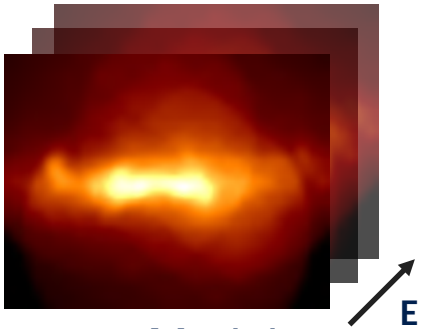
Reduced IRFs

Dataset



Bkg model

$\mathcal{L}(x \mid \theta)$



Model

See: Dataset simulation tutorial

The basic analysis steps

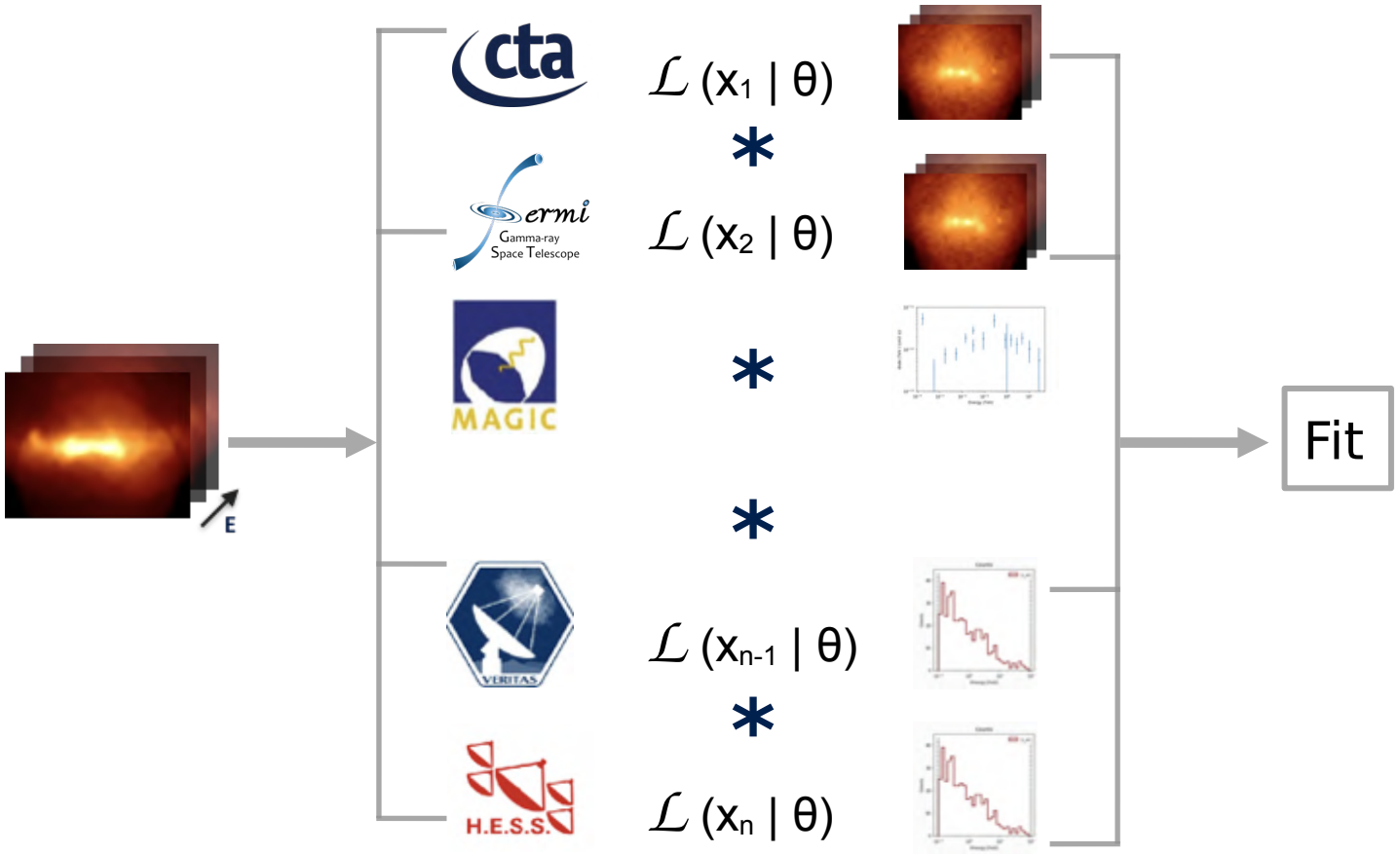
DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
4. Loop over selected observations

DL4 to DL5

1. Modelling
2. Do the fit
   - Choose your minimization parameters (optional)
   - Make the control plots, compute significance

DL4 to DL5 : Joint fitting



Gammapy Dataset structure allows heterogeneous data modelling and fitting: see joint fit tutorial

The basic analysis steps

DL3 to DL4

1. Select and retrieve observations
2. Define the reduced dataset geometry
3. Initialize the data reduction methods (makers)
4. Loop over selected observations

DL4 to DL5

1. Modelling
2. Do the fit
3. Run the DL5 estimators (estimators)
   - Initialisation of the geometry
   - Creation of the estimator(s)
   - Run them of the dataset(s)

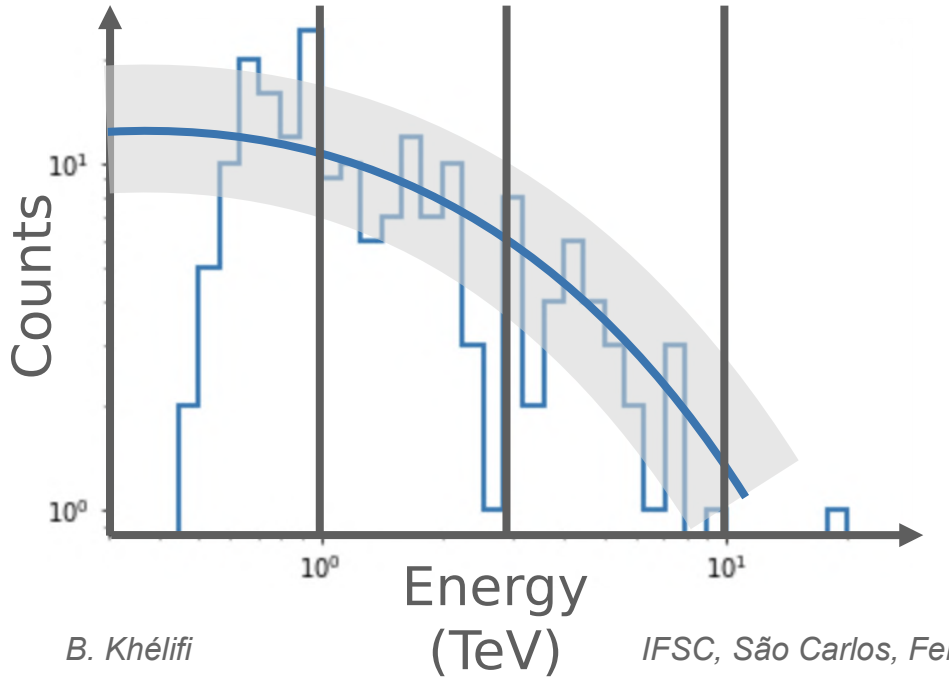
The basic analysis steps: Overview of the modelling and fitting

DL4 to DL5

1. Modelling
   - Define your model(s)
     - For the 3D analysis, add a final `FoVBackgroundModel`
   - Associate them/it to the correct dataset (or several)
2. Do the fit
   - Choose your minimization parameters (optional)
   - Make the control plots, compute significance
3. Run the DL5 estimators (estimators)
   - Initialization of the geometry
   - Creation of the estimator(s)
   - Run it/them on dataset(s)

DL5: estimating fluxes

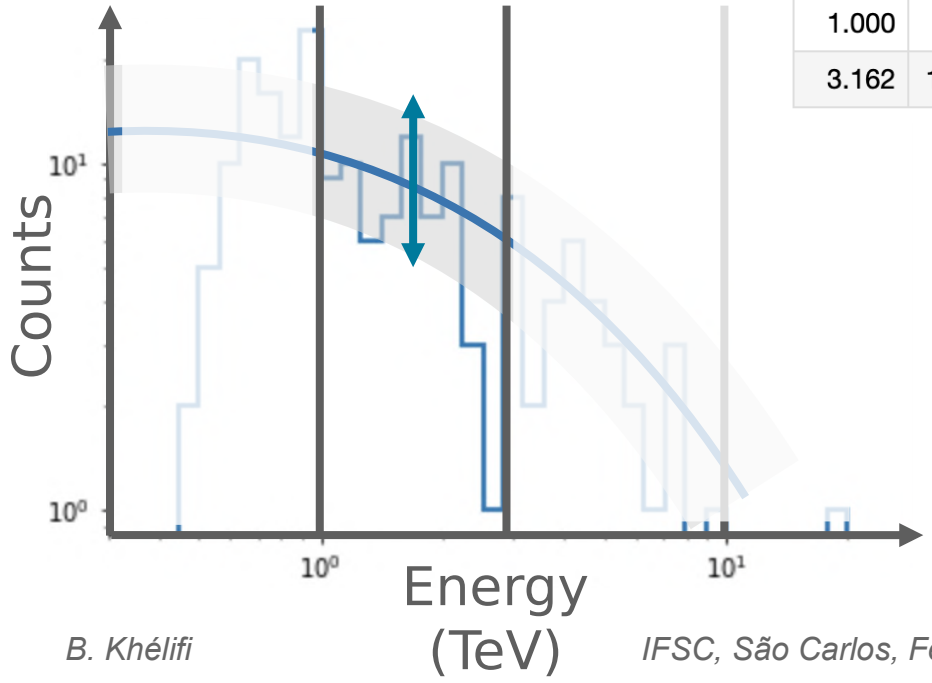
- Gammapy provides a set of estimator objects which create DL5 data products based on a model assigned to one or more datasets.



DL5: estimating fluxes

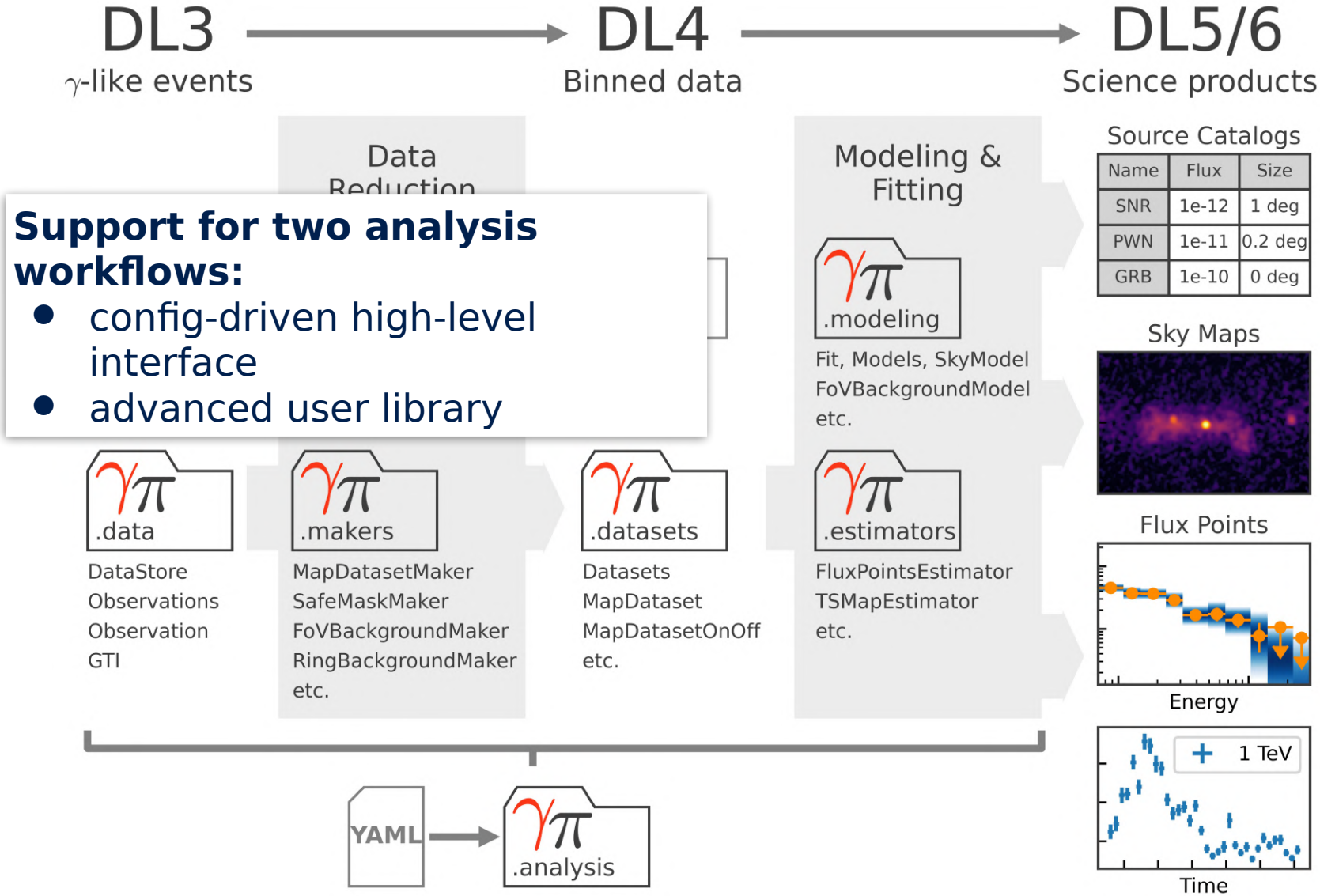
- Gammapy provides a set of estimator objects which create DL5 data products based on a model assigned to one or more datasets.

e_min	e_max	ref_flux	ref_eflux	norm	norm_err	norm_ul
TeV	TeV	1 / (cm2 s)	TeV / (cm2 s)			
float64	float64	float64	float64	float64	float64	float64
0.300	1.000	1.699e-10	8.184e-11	nan	nan	nan
1.000	3.162	2.433e-11	3.845e-11	1.032	0.058	1.152
3.162	10.000	3.847e-12	1.923e-11	0.879	0.103	1.099



`FluxPointsEstimator`
`LightCurveEstimator`
`TSMapEstimator`
`ExcessMapEstimator`

Data workflow and package structure



Config-file driven analysis

The YAML configuration file

```yaml
general:
    log: {level: info, filename: null, filemode: null, format: null, datefmt: null}
    outdir: .
observations:
    datastore: $GAMMAPY_DATA/hess-dl3-dr1
    obs_ids: []
    obs_file: null
    obs_cone: {frame: icrs, lon: 83.633 deg, lat: 22.014 deg, radius: 5.0 deg}
    obs_time: {start: null, stop: null}
    required_irf: [aeff, edisp, bkg]
datasets:
    type: 1d
    stack: true
    geom:
        axes:
            energy: {min: 0.2 TeV, max: 30.0 TeV, nbins: 15}
            energy_true: {min: 0.1 TeV, max: 60.0 TeV, nbins: 30}
    map_selection: [counts, exposure, edisp]
    background:
        method: reflected
        exclusion: null
    safe_mask:
        methods: [aeff-default, aeff-max]
        parameters: {aeff_percent: 10}
    on_region: {frame: icrs, lon: 83.63 deg, lat: 22.01 deg, radius: 0.11 deg}
    containment_correction: true
fit:
    fit_range: {min: 0.6 TeV, max: 20.0 TeV}
flux_points:
    energy: {min: 0.4 TeV, max: 20.0 TeV, nbins: 10}
    source: Crab
    parameters: {selection_optional: all}
```

```python
config = AnalysisConfig.read(f"{estimate}/config.yaml")
analysis = Analysis(config)
analysis.get_observations()
analysis.get_datasets()

models = Models.read(f"{estimate}/models.yaml")
analysis.set_models(models)
analysis.run_fit()
```

Select observations

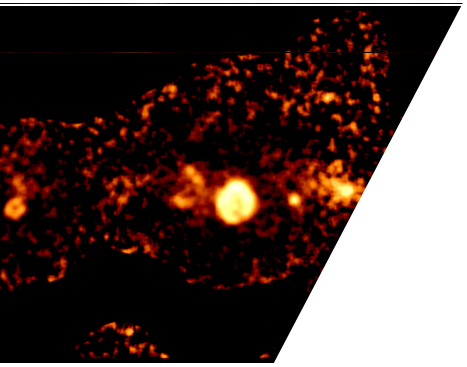
Define target Dataset geometry

Define data reduction methods

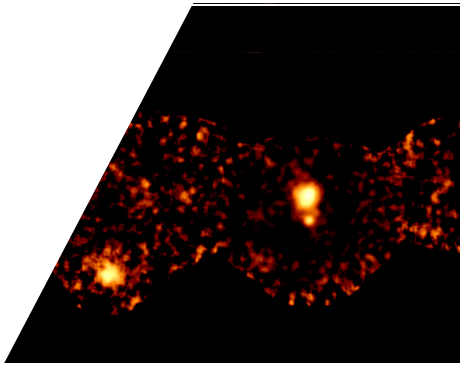
Define Fit configuration

Define high level estimators config.

See High Level Interface tutorial



Next sessions...



Next sessions: hands-on

We have prepared three specific analysis tutorials:

I. Basics of the analysis: 1D (spectrum) and 3D

II. Realization of simulations: impact of the statistics on results

III. Study of instrument systematics

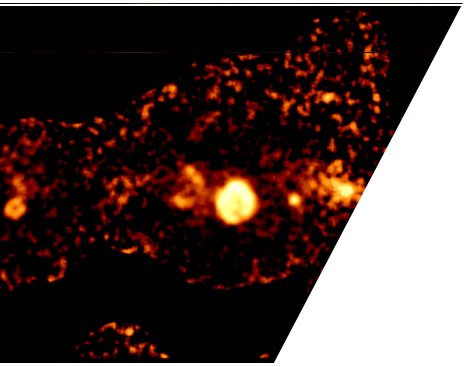
Conduct of the sessions:

Tutorial as a demo (by me) → Exercise (for you)

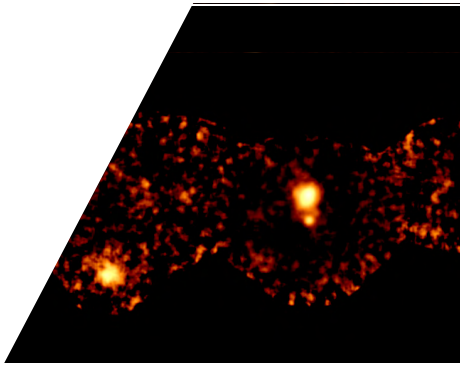
You can retrieve them with:

```
git clone https://github.com/bkhelifi/Brazil_2024.git
```

Contact point: khelifi@in2p3.fr



Backup slides



Getting the software

- **Recommended gammapy installation**

```
curl -O https://gammapy.org/download/install/gammapy-1.1-environment.yml
conda env create -f gammapy-1.1-environment.yml
conda activate gammapy-1.1
```

- **Download tutorials & associated data**

```
gammapy download notebooks
gammapy download datasets
export GAMMAPY_DATA=$PWD/gammapy-datasets/1.1
```

Note: mamba might prove a better/faster package manager

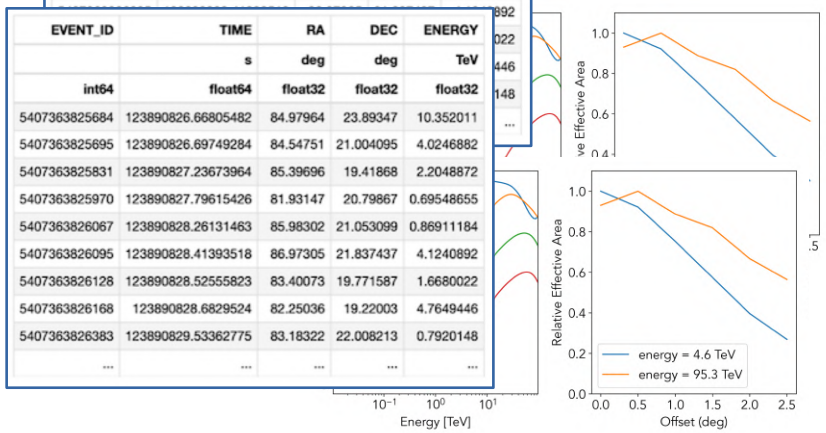
See: https://docs.gammapy.org/1.1/getting-started/index.html#quickstart-setup

DL3 to DL4: data reduction

DL3
γ-like events

1. Select and retrieve relevant observations

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



DataStore

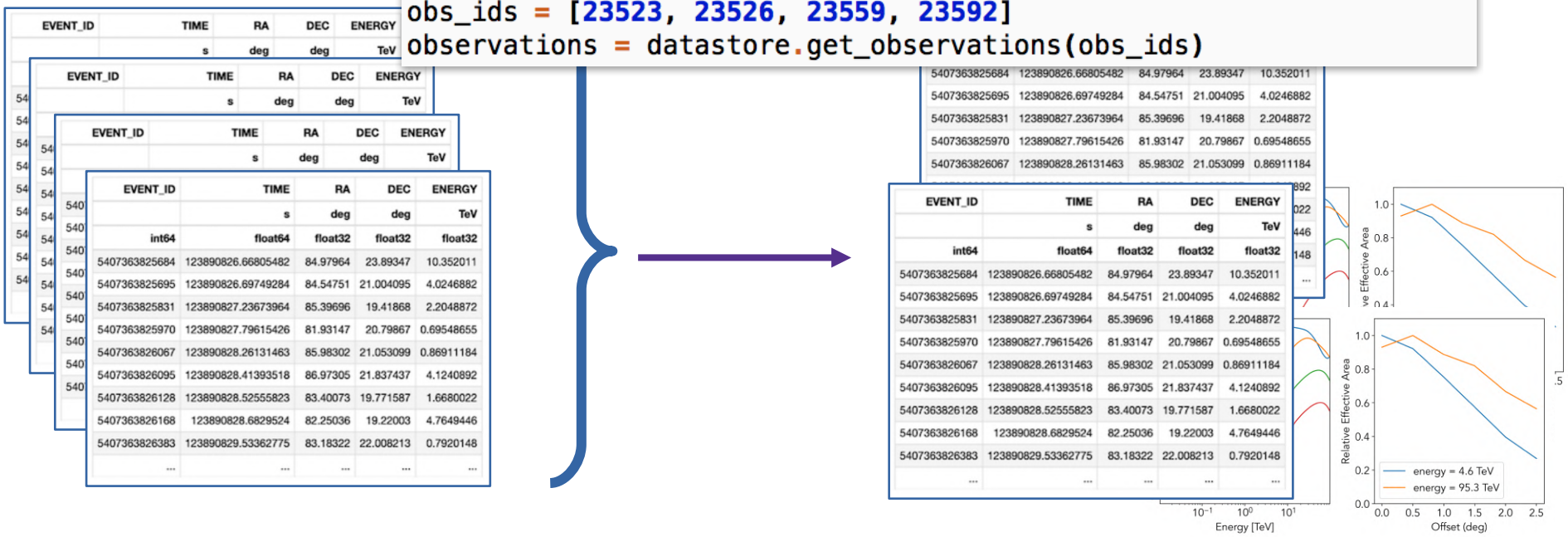
Observation / Observations

DL3 to DL4: data reduction

DL3
γ-like events

1. Select and retrieve relevant observations

```
datastore = DataStore.from_dir("$GAMMAPY_DATA/hess-dl3-dr1/")
obs_ids = [23523, 23526, 23559, 23592]
observations = datastore.get_observations(obs_ids)
```



DataStore

Observation / Observations

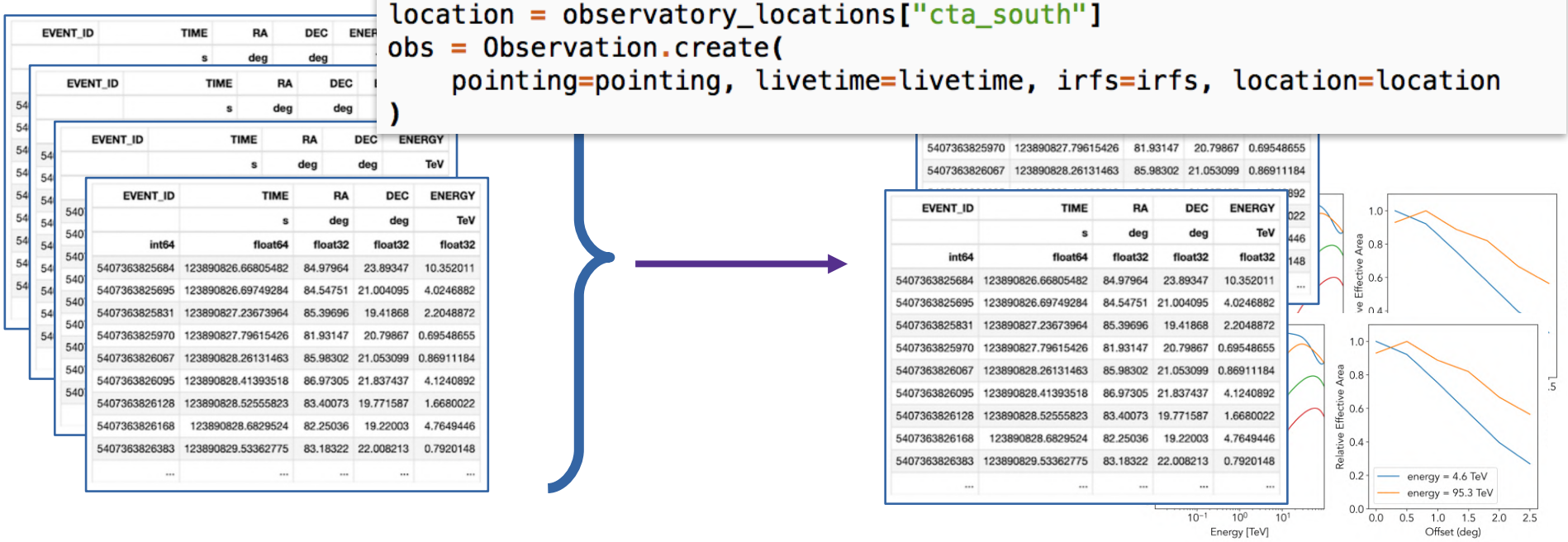
DL3 to DL4: data reduction

DL3

γ-like events

1. Select and retrieve relevant observations

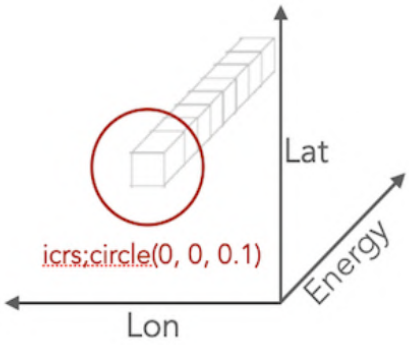
```
# Create an in-memory observation
location = observatory_locations["cta_south"]
obs = Observation.create(
    pointing=pointing, livetime=livetime, irfs=irfs, location=location
)
```



DataStore

Observation / Observations

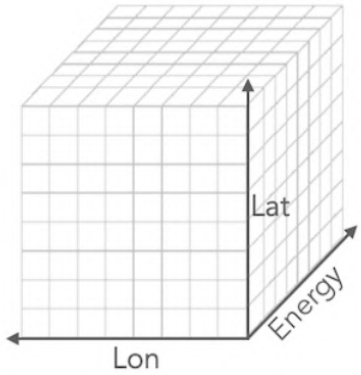
DL3 to DL4: data reduction



region & energy

1. Select and retrieve relevant observations

2. Define the reduced dataset geometry
   - Is the analysis 1D (spectral only) or 3D?
   - Define target binning and projection

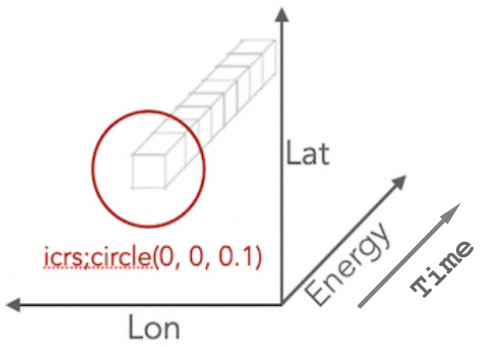


WCS & energy

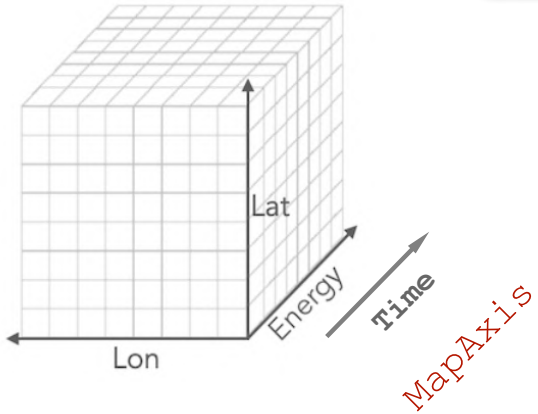
Geometry : multidimensional maps

- Gammapy maps represent data on the sky with non-spatial dimensions (in particular energy)
  - World Coord. System (WCS) for 3D analyses (lon, lat, E)
  - Region geometry for 1D analysis

See : working with maps



`RegionGeom` / `RegionNDMap`



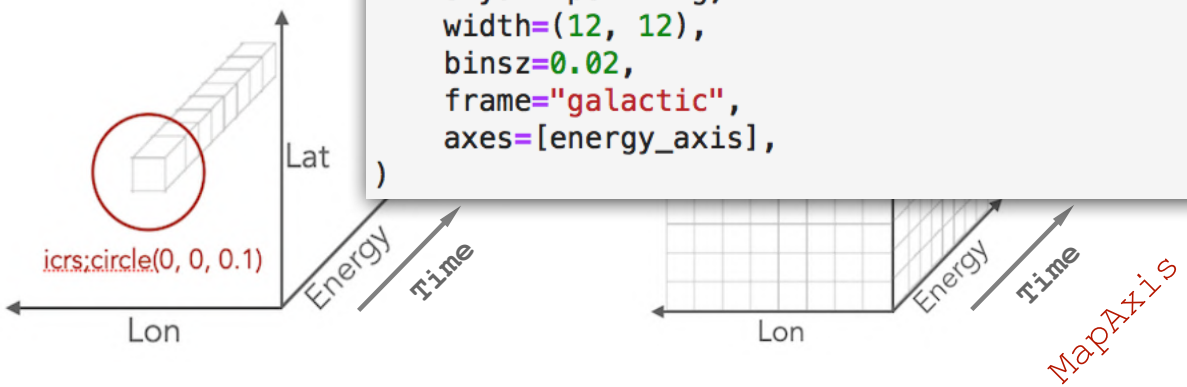
`WcsGeom` / `WcsNDMap`

Geometry : multidimensional maps

- Gammapy maps represent data on the sky with non-spatial dimensions (in particular energy)
  - World Coord. System (WCS) for 3D analyses (lon, lat, E)
  - Region geometry for 1D analysis

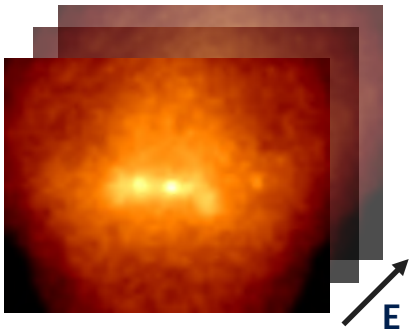
```
energy_axis = MapAxis.from_energy_bounds(
    "0.02 TeV", "100 TeV", nbin=10, per_decade=True
)

geom = WcsGeom.create(
    skydir=pointing,
    width=(12, 12),
    binsz=0.02,
    frame="galactic",
    axes=[energy_axis],
)
```

king with maps



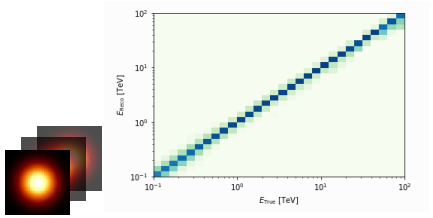
`RegionGeom` / `RegionNDMap` `WcsGeom` / `WcsNDMap`

Datasets modelling and fitting



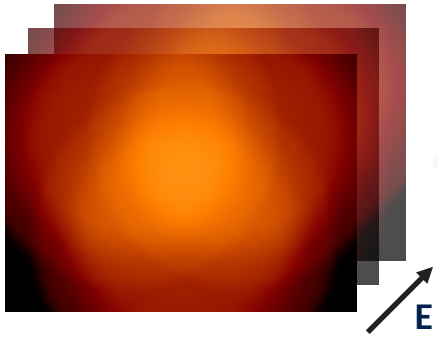
Reduced data

I/O



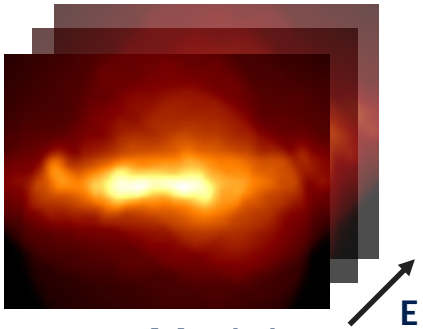
Reduced IRFs

```
fit = Fit(optimize_opts={"print_level": 1})
result = fit.run(datasets=[dataset])
```



Bkg model

$\mathcal{L}(x \mid \theta)$



Model

see: Dataset fitting tutorial

DL4 to DL5: estimating fluxes

- Gammapy provides a set of estimator objects which create DL5 data products based on a model assigned to one or more datasets.
  - Once a proper model is determined
  - In predefined energy intervals, estimators compute:
    - fluxes errors and associated significance
    - fit statistic scan etc.
  - They can produce flux points, light curves, flux maps