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

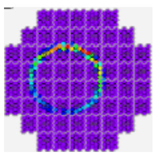
VHE data analysis using Open Source Packages workshop

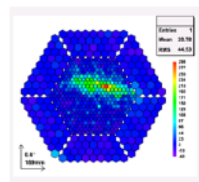
Feb 2th 2022

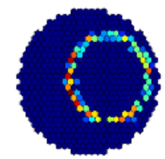
Régis Terrier (APC), Atreyee Sinha (UCM)

VHE analysis: formats and tools

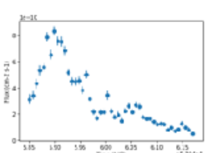
- All VHE gamma-ray instruments have their own proprietary formats and tools making joint analyses impossible

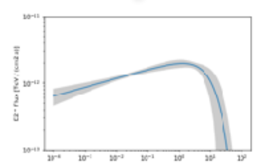


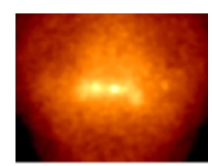




Raw data







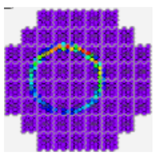
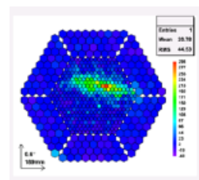
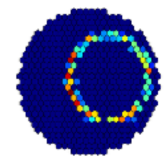
High level science products

How to compare:

- instrument-based assumptions on physical spectrum?
- inter-instrument systematics effects?
- treatment of low statistics?

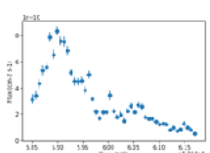
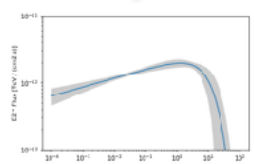
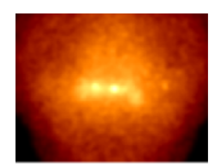
VHE analysis: formats and tools

- All VHE gamma-ray instruments have their own proprietary formats and tools making joint analyses impossible

Raw data

VHE analysis needs common open *data formats* and common open *tools*

High level science products

How to compare:

- instrument-based assumptions on physical spectrum?
- inter-instrument systematics effects?
- treatment of low statistics?

The data flow

Separating instrument specific data treatment from common use cases and methods

Reconstruction pipeline
(instrument specific)

Science Tools (general users)

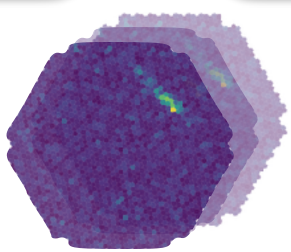
DL 0-1
raw - calibrated

DL 2
reconstructed

DL 3
filtered γ-like events

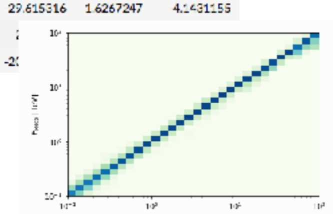
DL 4
science

DL 5
high-level

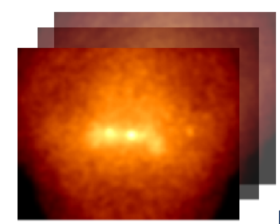


camera data

ENERGY	RA	DEC	L	B
MeV	deg	deg	deg	deg
float32	float32	float32	float32	float32
12186.642	260.45935	-33.553337	353.36273	1.7538676
25196.598	261.37506	-34.395004	353.09607	0.6520652
15621.498	259.56973	-33.409416	353.05673	2.4450684
12816.32	279.95883	-25.340391	6.45856	-4.0548873
18988.387	260.8568	-36.355804	351.23734	-0.101912394
11610.23	266.15518	-26.224436	2.1986027	1.6034819
13960.802	271.44742	-29.615316	1.6267247	4.1431155
10477.372	266.3981	[illegible]		
13030.88	271.70128	[illegible]		



γ-like event lists
IRFs



maps,
spectra,
light curves

The gammapy concept

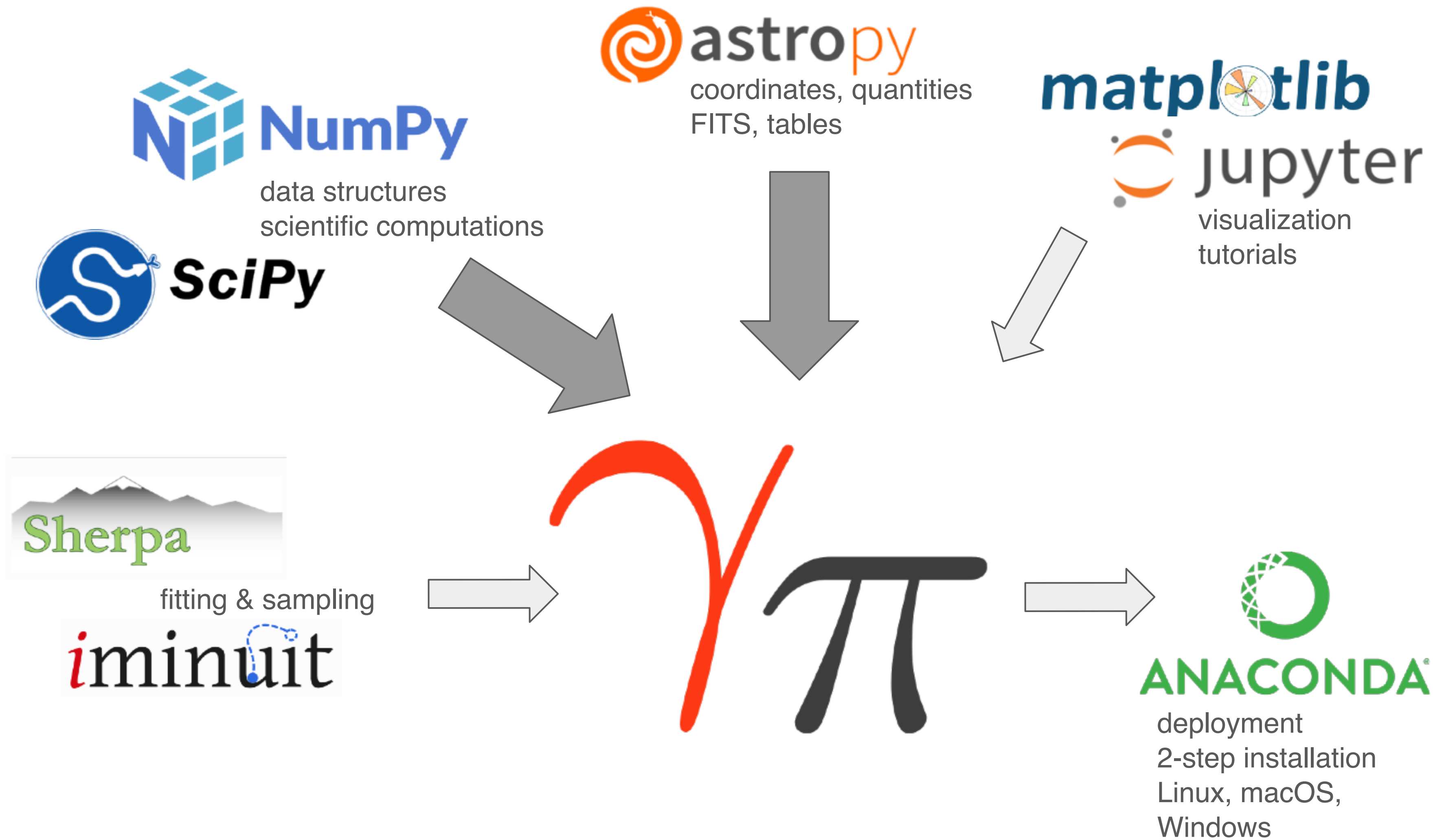
A python package for high-level γ-ray astronomy based on common data formats

A flexible, open-source, community driven, python library

+

The library for CTA science tools

Gammapy in the Python ecosystem



Getting the software

- **Recommended gammapy installation**

```
curl -O https://gammapy.org/download/install/gammapy-0.19-
environment.yml
conda env create -f gammapy-0.19-environment.yml
conda activate gammapy-0.19
```

- **Download tutorials & associated data**

```
gammapy download notebooks --release 0.19
gammapy download datasets
export GAMMAPY_DATA=$PWD/gammapy-datasets
```

Note: mamba might prove a better/faster package manager

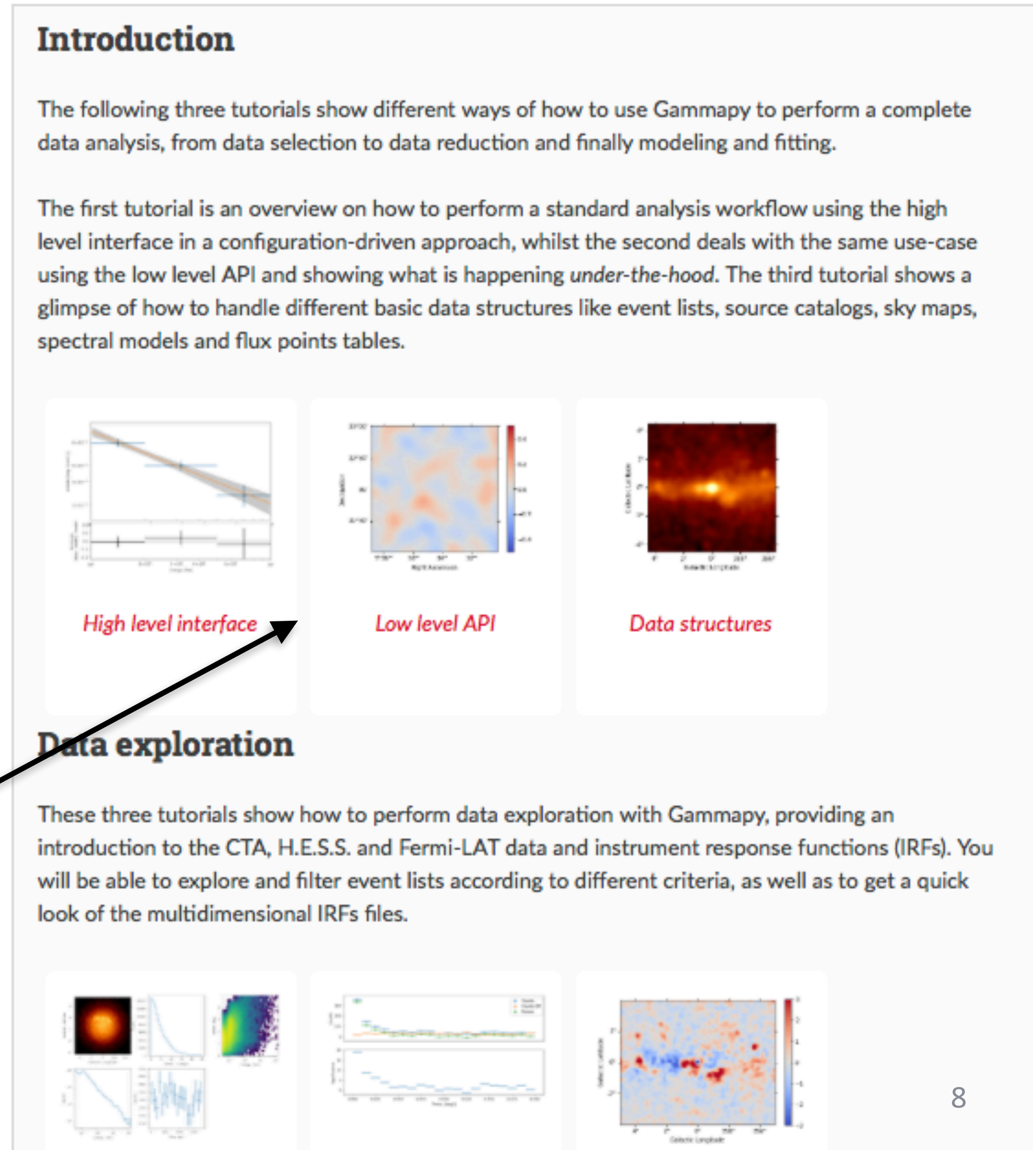
See: https://docs.gammapy.org/0.19/install/index.html

Getting started: documentation

See docs.gammapy.org

Tutorials to learn simple data analysis recipes

- spectral analysis
- lightcurve extraction
- 3D fitting
- simulation



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*

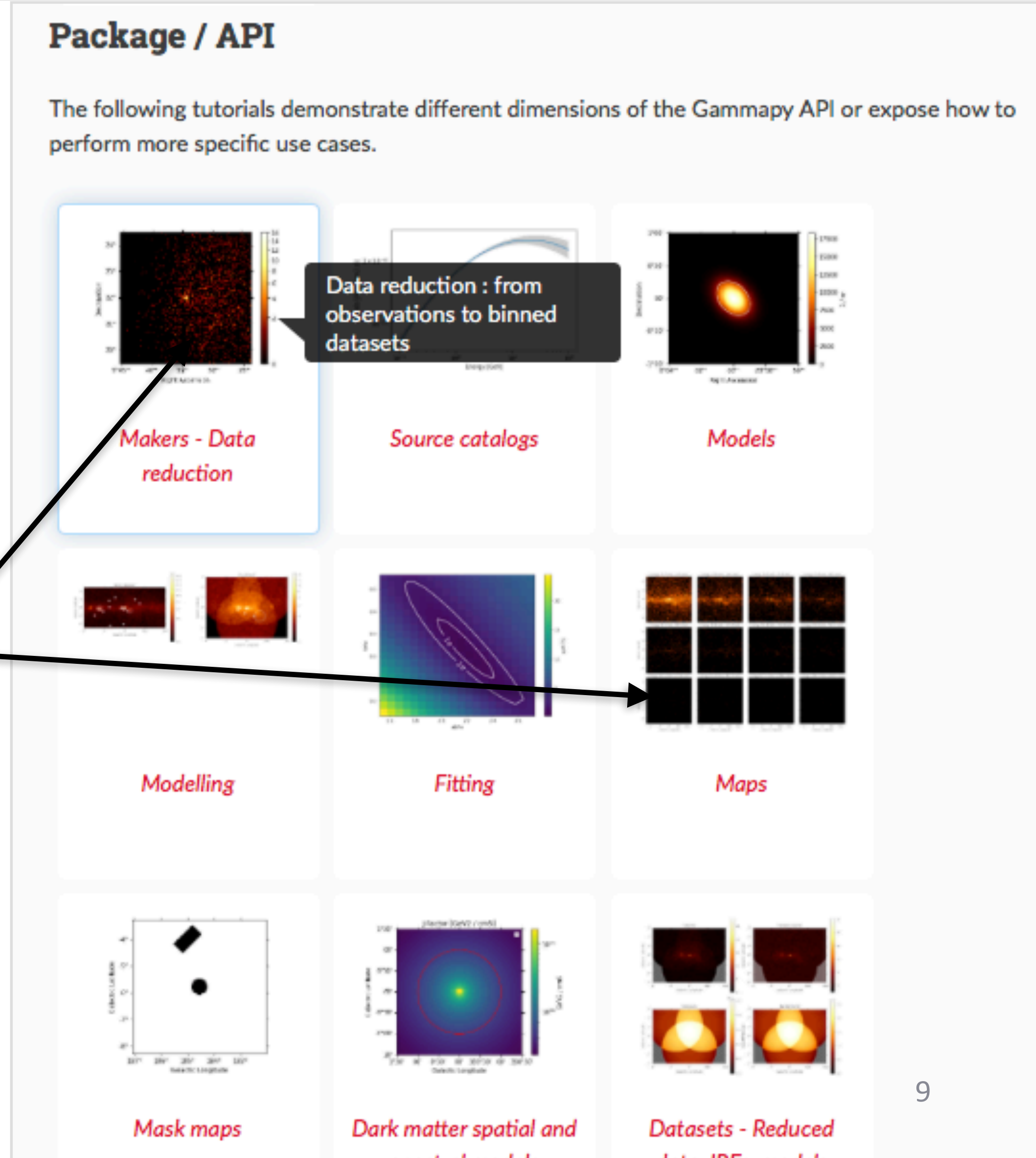
Data exploration

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.

Getting started: documentation

Learn how to use the general API

- go beyond tutorials use cases
- exploit Gammapy flexibility



Getting help

- Where/How to interact with dev team and experienced users, provide feedback, get help:

  - gammapy.slack.
    - In particular: #help channel

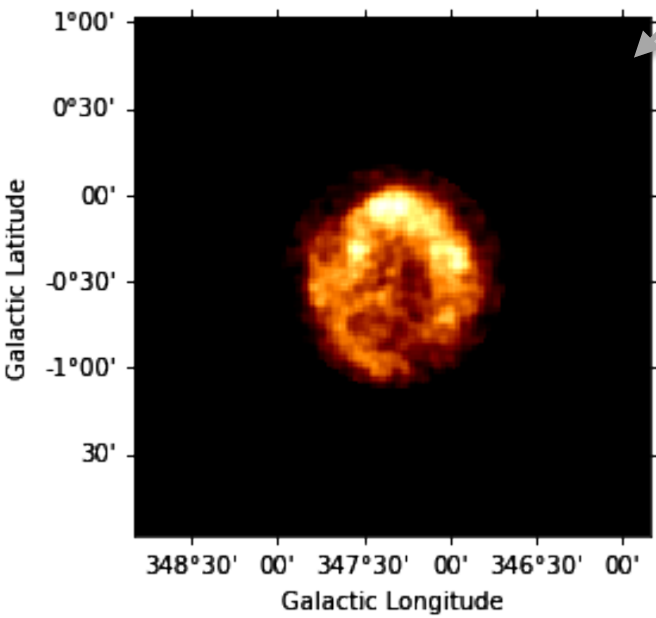
  - [GitHub discussions]
    - help category

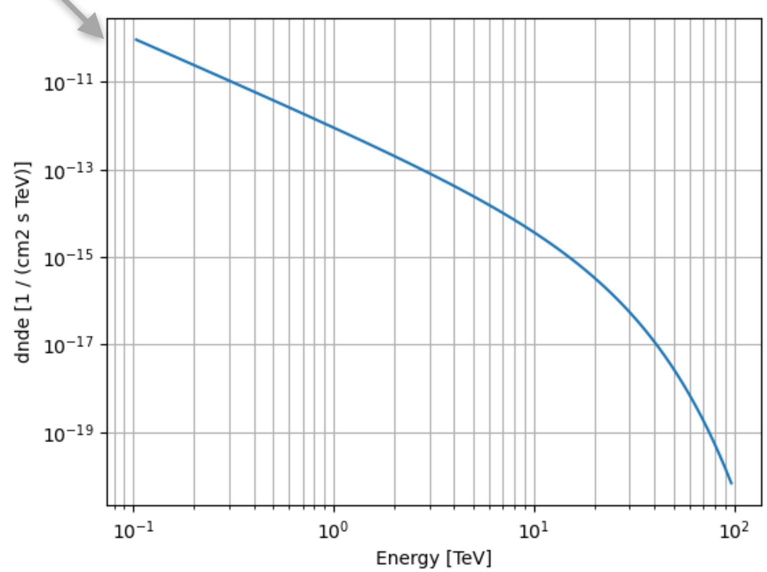
  - [GitHub issues] to report bugs or feature requests

Modeling the expected number of detected photons

- Assume a source *S* emits gamma-ray photons.
- Its emission is represented by sky model $\Phi(p_t, E_t)$ with p_t the photon position in the sky and E_t its energy
- We want to determine the model parameters that best reproduce the measured data.

$$\Phi(p_t, E_t)$$





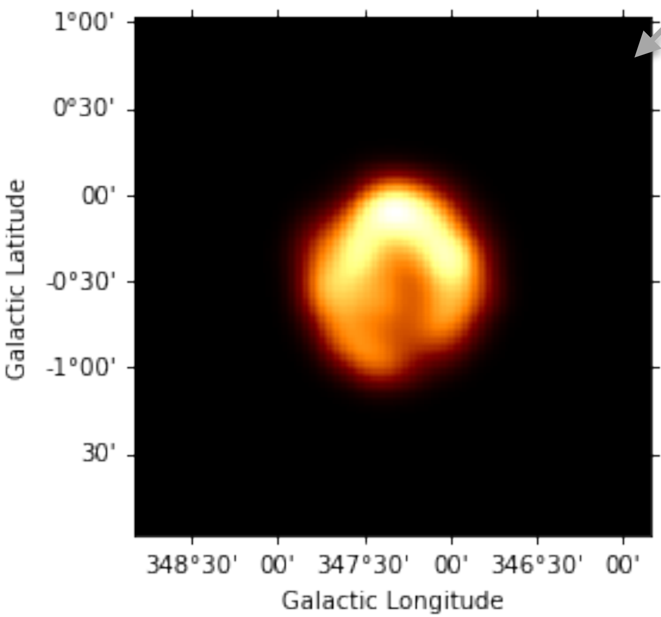
Modeling the expected number of detected photons

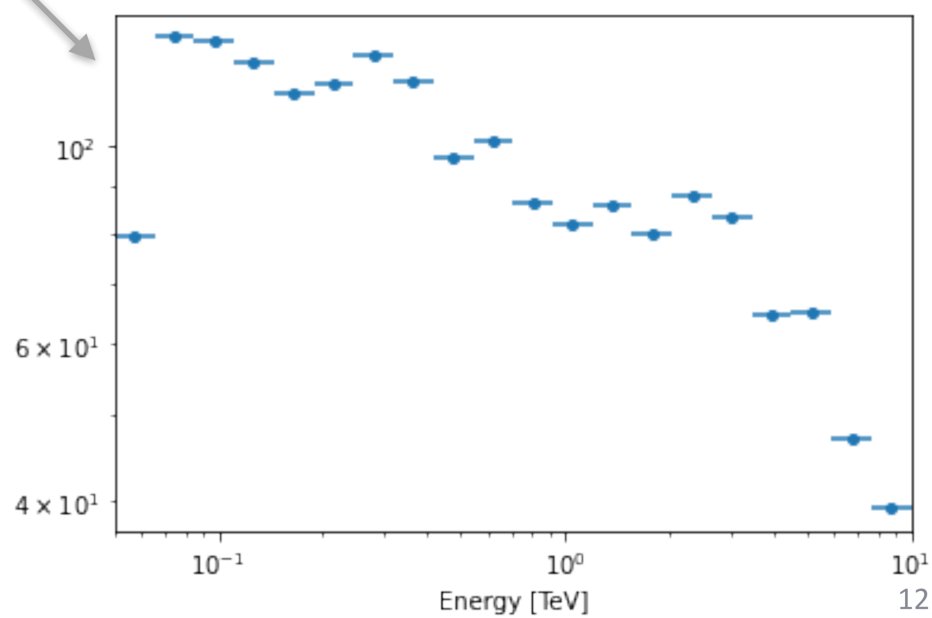
- The number of observed photons from source *S* is

$$N(p,E)dpdE = t_{obs}\int_{E_t} dE_t \int_{p_t} dp_t\ R(p,E\,|\,p_t,E_t) \times\ \Phi(p_t,E_t)$$

where $R(p,E\,|\,p_t,E_t)$ is the instrument response

$N(p,E)$





The Instrument Response Functions (IRFs)

- We assume that the instrument response can be simplified as the product of:

$$R(p, E \,|\, p_t, E_t) = A_{\text{eff}}(p_t, E_t) \times PSF(p \,|\, p_t, E_t) \times E_{\text{disp}}(E \,|\, p_t, E_t)$$

with :

- $A_{\text{eff}}(p_t, E_t)$ the effective collection area in m^2
- $PSF(p \,|\, p_t, E_t)$ the point spread function in sr^{-1}. It is the density function of the probability to detect a photon emitted at p_true at position p.
- $E_{\text{disp}}(E \,|\, p_t, E_t)$ the energy dispersion in TeV^{-1}. Probability to detect photon emitted at True at energy E.

The Instrument Response Functions (IRFs)

- Measured events do not only contain genuine photons but also residual charged cosmic-ray background:

$$N(p,E) = \sum_S N_S(p,E) + N_{bkg}(p,E)$$

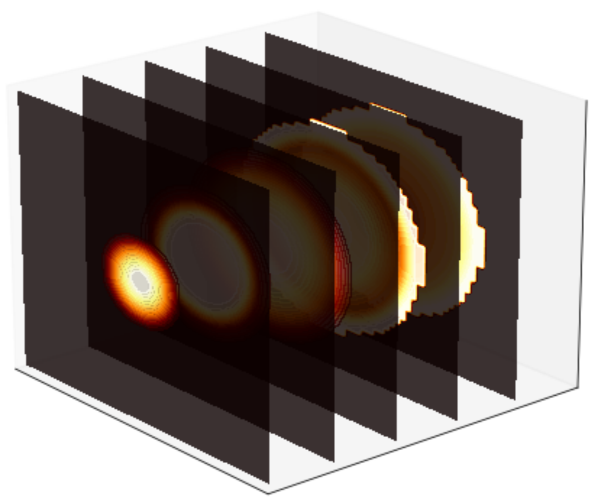
with :

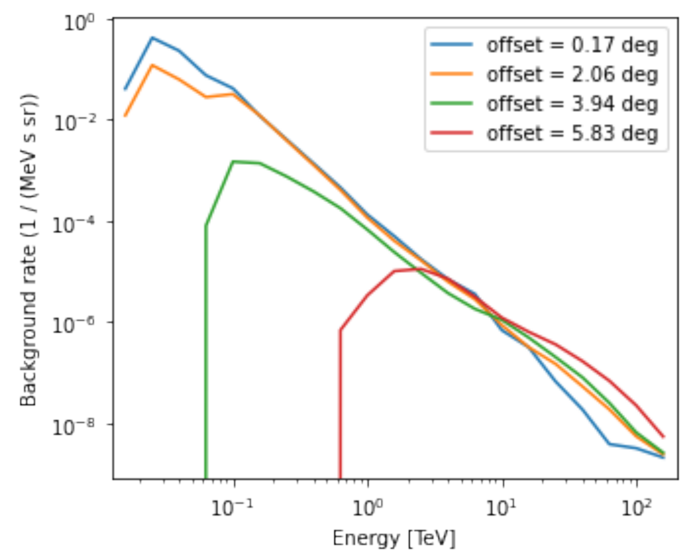
- $N_S(p,E)$ the number of predicted photons from source S
- $N_{bkg}(p,E)$ the number of background events

- The residual CR background must be modeled along the sky model:
  - can be estimated from OFF data
  - can be described by a model $N_{\rm bkg}(p,E) = BKG(p,E) \times t_{\rm obs}$

The background model IRF

- $BKG(p, E)$ is the 3D background model in $s^{-1}sr^{-1}TeV^{-1}$
- The model can be built from simulations of atmospheric showers or from a large set of empty field observations taken in similar conditions. ***It is subject to non-negligible uncertainties***.
- Note that the background is highly sensitive to observing conditions such as zenith angle, optical efficiency of the system, atmospheric transparency etc.

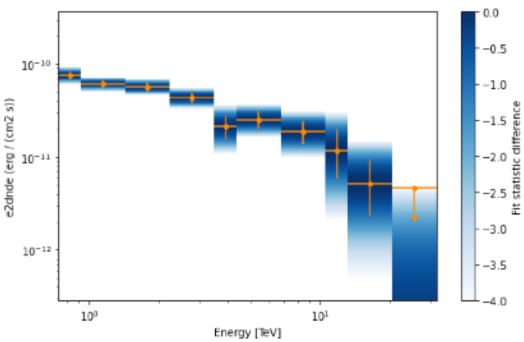




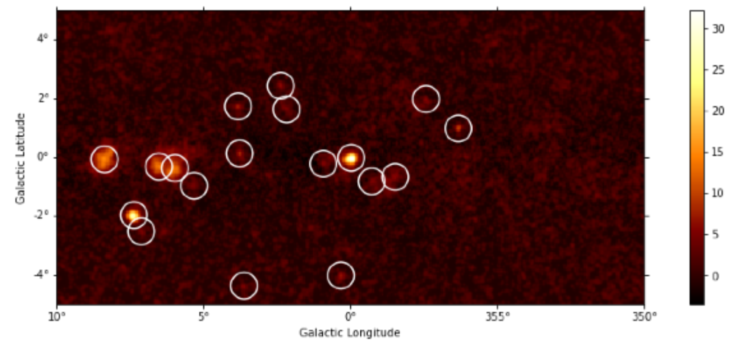
Exploring DL3 with gammapy

- The g.a.d.f. v0.2 format relies on several FITS HDUs:
  - EVENTS: table of gamma-like events measured parameters
  - GTI: Interval of time associated to events
  - POINTING: Telescope pointing info
  - AEFF : Effective area table (true energy, FoV offset)
  - EDISP : Energy dispersion (true energy, FoV offset)
  - PSF: isotropic PSF (true energy, FoV offset)
  - BACKGROUND: (energy, FoV lon, FoV lat)
- A general HDU table connects everything
- See data exploration tutorial

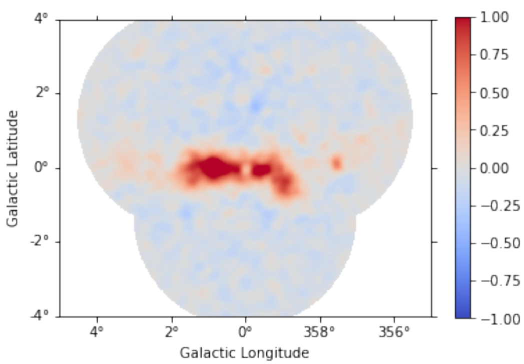
Typical analysis use cases



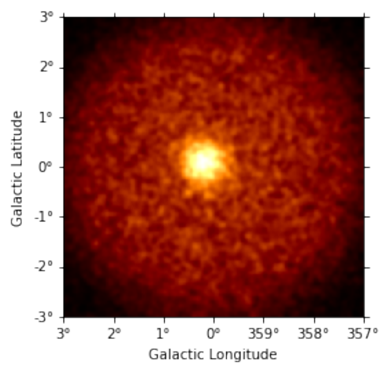
[1D spectral analysis]



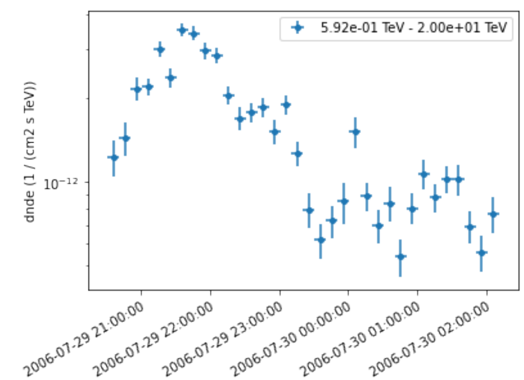
[Source detection]



[3D analysis]



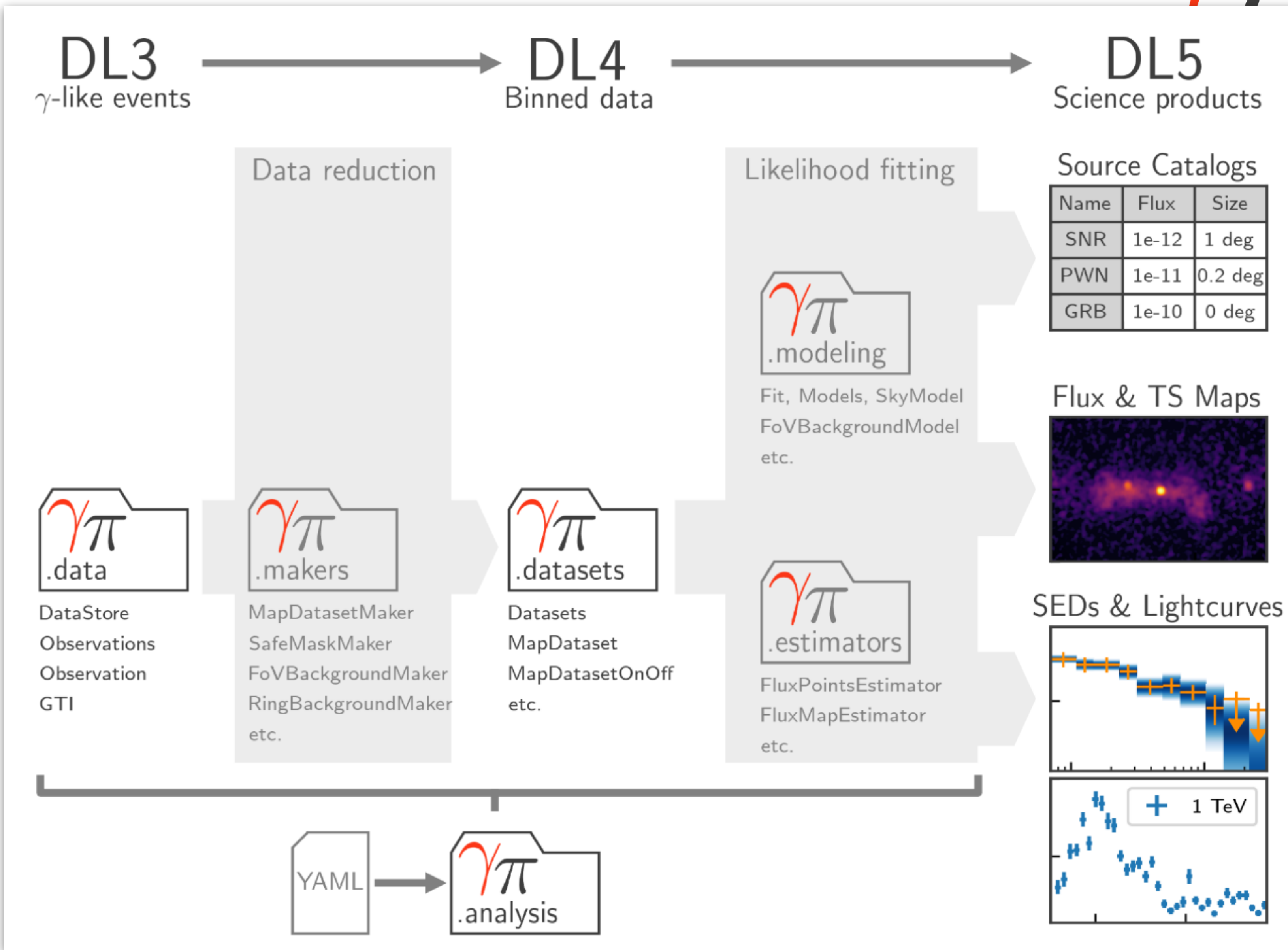
[observation simulation]



[light-curve extraction]

All analysis types follow the same workflow and the same API

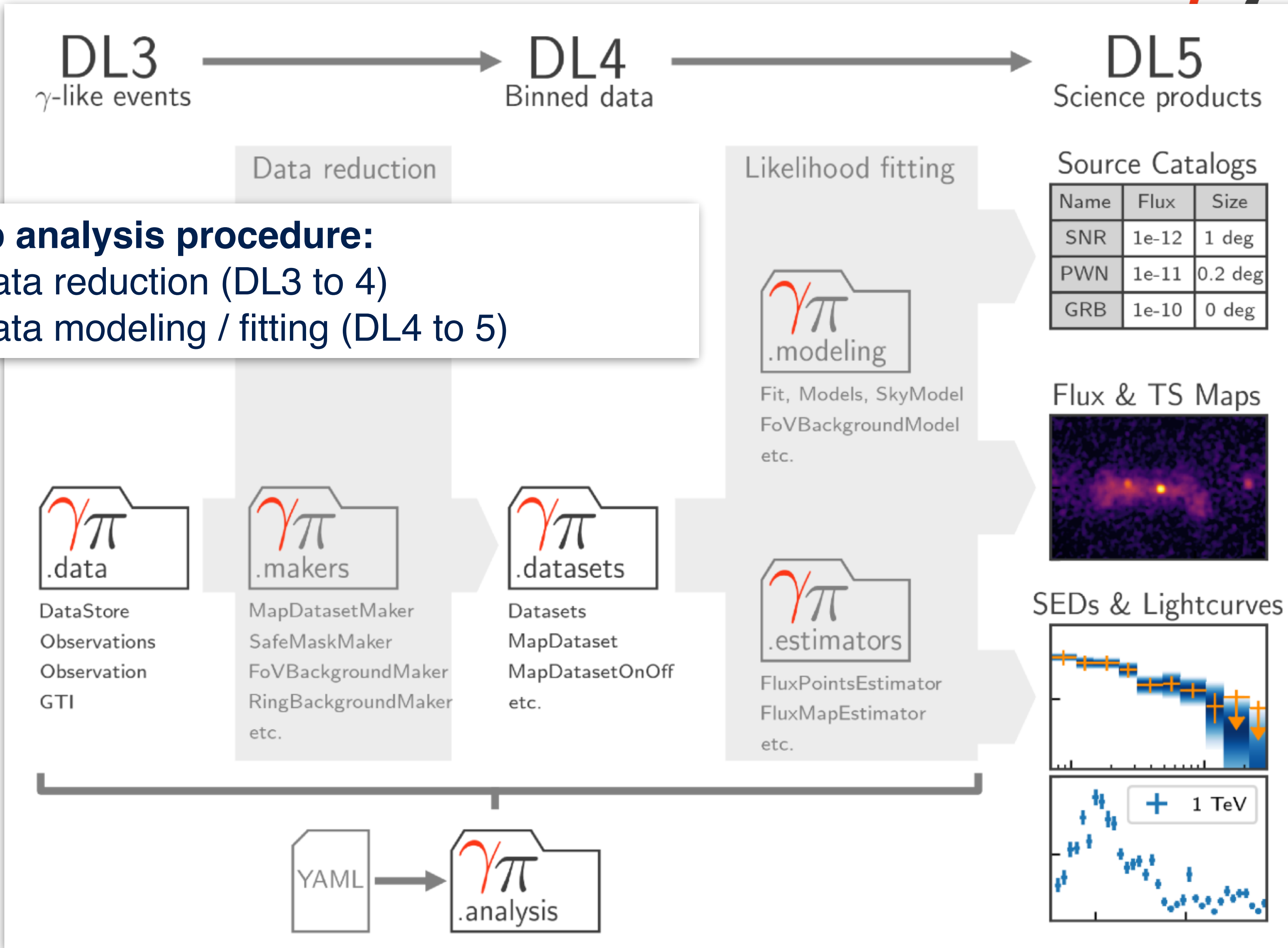
Data workflow and package structure



Data workflow and package structure

2-step analysis procedure:

- data reduction (DL3 to 4)
- data modeling / fitting (DL4 to 5)

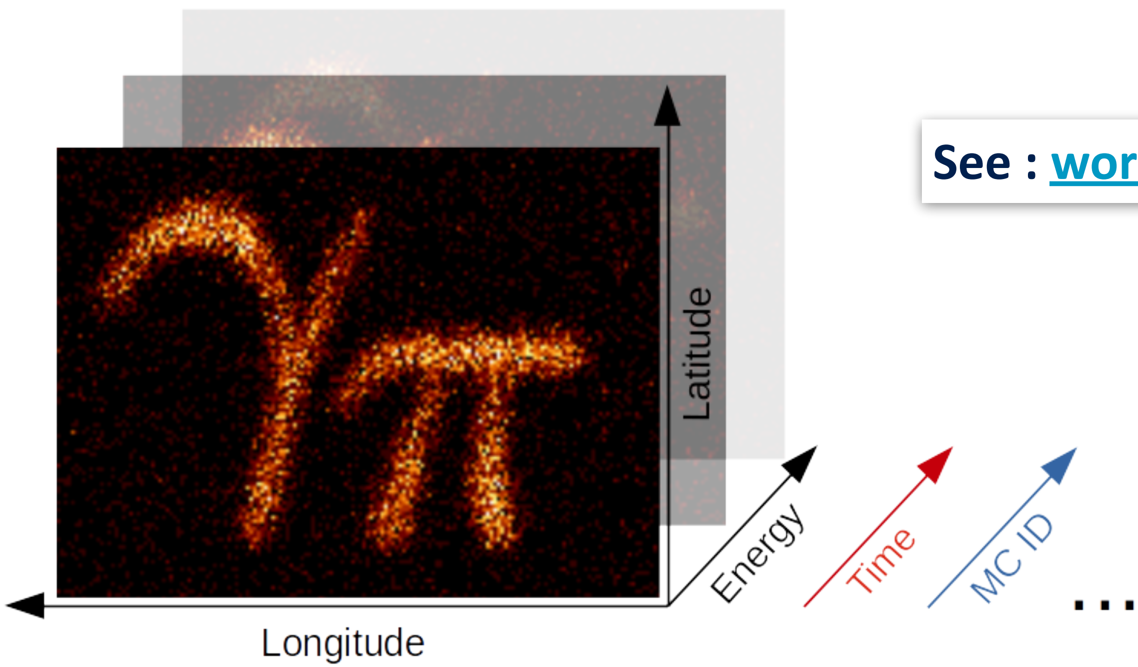


Data reduction

1. Select and retrieve relevant observations from the data store
2. Define the reduced dataset geometry
   - Is the analysis 1D (spectral only) or 3D?
   - Define target binning and projection
3. Initialize the data reduction methods (makers)
   - Data and IRF projection
   - Background estimation
   - Safe Mask determination
4. Loop over selected observations
   - Apply makers to produce reduced datasets
   - Optionally combine them (stacking)

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

Data reduction

1. Select and retrieve relevant observations from the data store
2. Define the reduced dataset geometry
   - Is the analysis 1D (spectral only) or 3D?
   - Define target binning and projection
3. Initialize the data reduction methods (makers)
   - Data and IRF projection
   - Background estimation
   - Safe Mask determination
4. Loop over selected observations
   - Apply makers to produce reduced datasets
   - Optionally combine them (stacking)

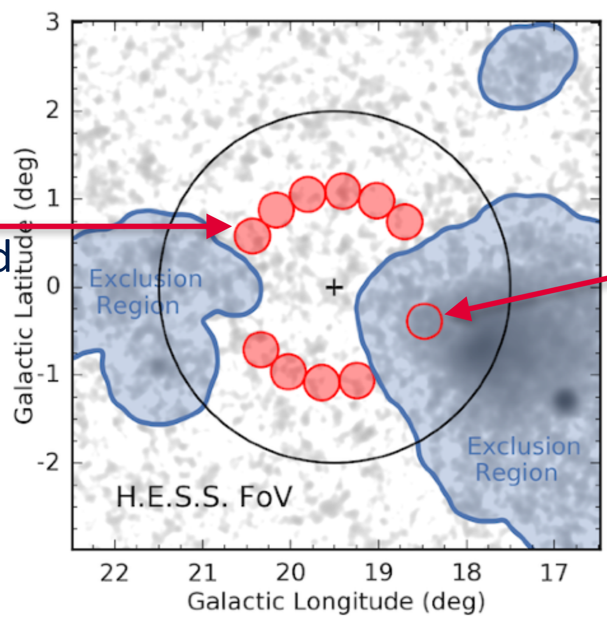
Estimating the background from the data

- To reduce systematic uncertainties, $BKG(p, E)$ is usually corrected on the observed data themselves.
  - Field of View (FoV) background estimation
    - $BKG(p, E)$ is normalized in regions of the observed FoV assumed to be deprived of gamma-ray signal

Measuring the background from the data

- To further reduce systematics, 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 background

OFF regions
containing only background

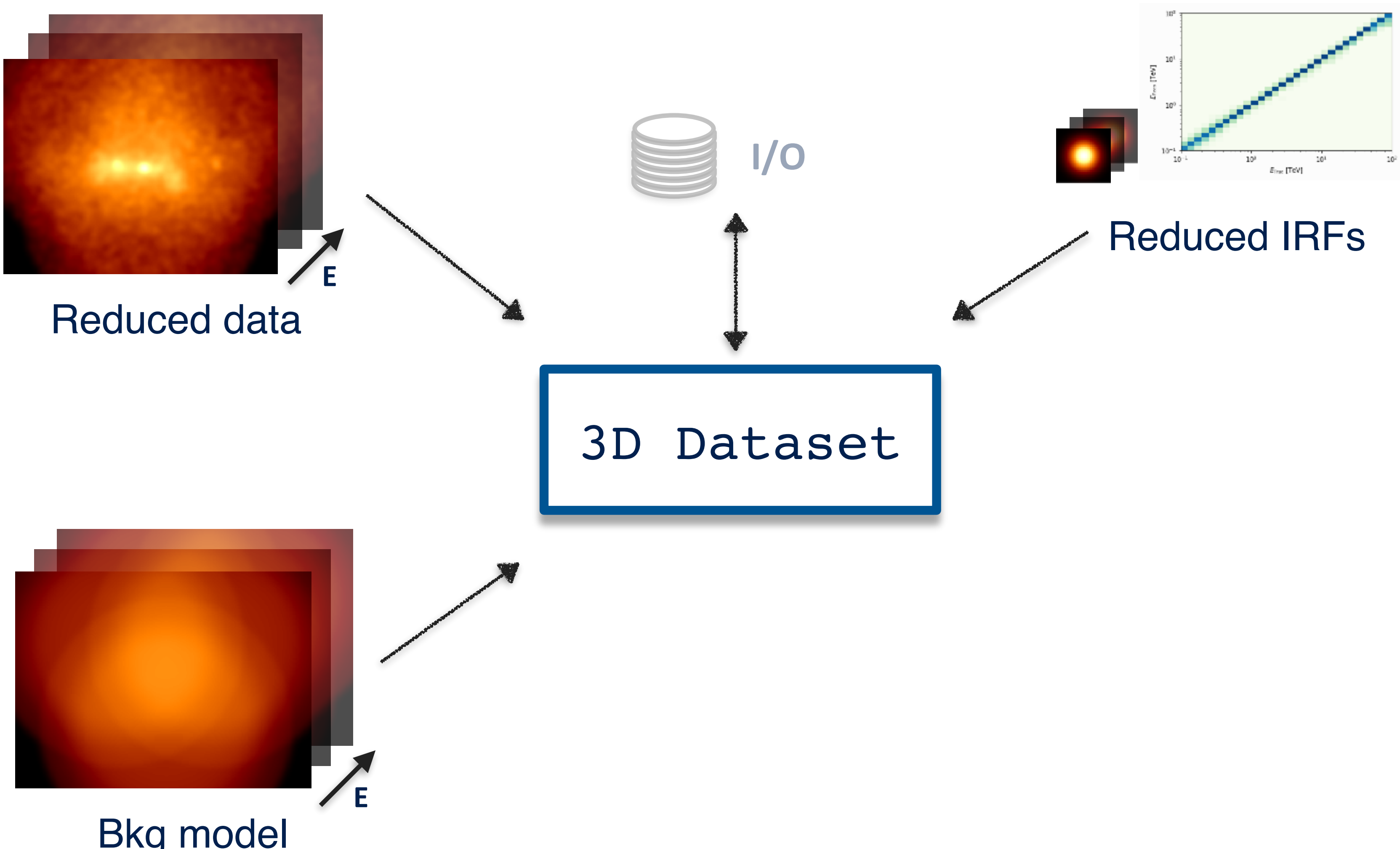


ON region
containing signal and background

Data reduction

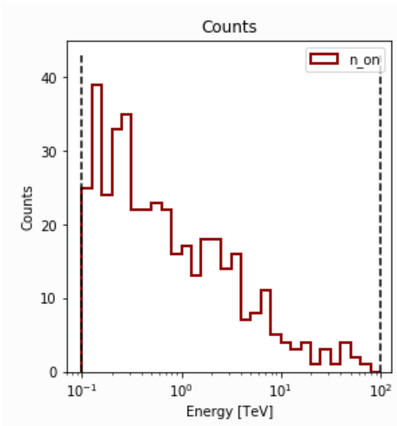
1. Select and retrieve relevant observations from the data store
2. Define the reduced dataset geometry
    - Is the analysis 1D (spectral only) or 3D?
    - Define target binning and projection
3. Initialize the data reduction methods (makers)
    - Data and IRF projection
    - Background estimation
    - Safe Mask determination
4. Loop over selected observations
    - Apply makers to produce reduced datasets
    - Optionally combine them (stacking)

DL4 structures: Datasets



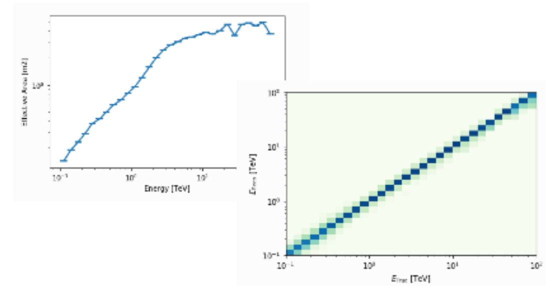
see: [Dataset API tutorial]

DL4 structures: Datasets



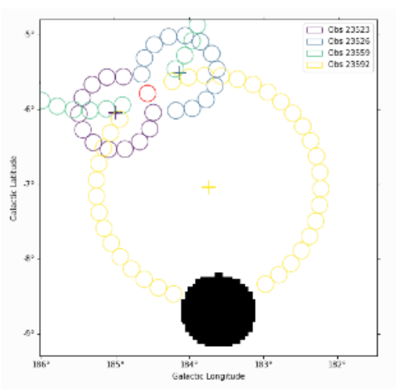
Reduced data

I/O



Reduced IRFs

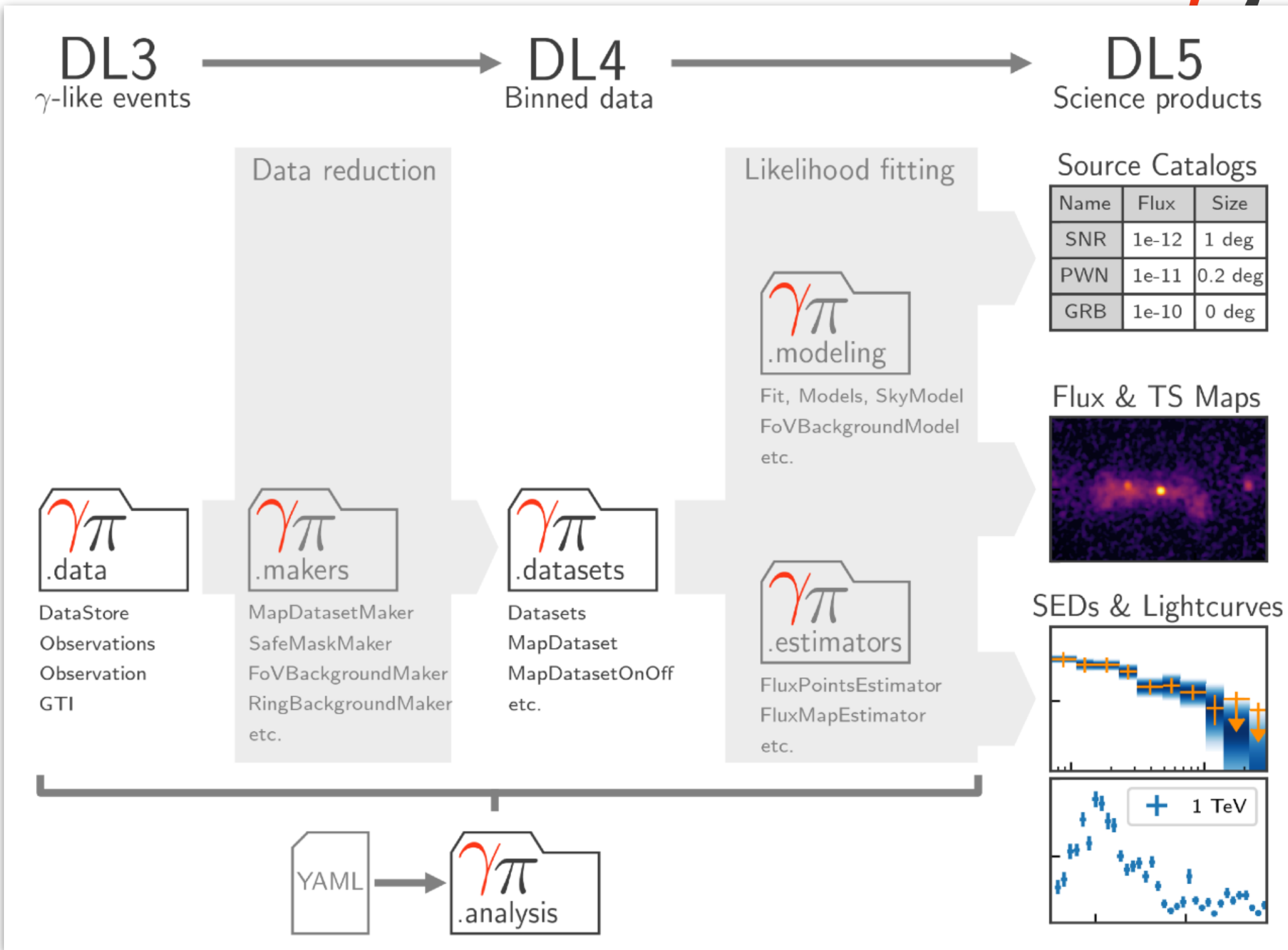
1D Dataset



Bkg data or model

see: Dataset API tutorial

Data workflow and package structure



Modeling and fitting

- For modeling and fitting, Gammapy relies on ***forward-folding***:
  - the number of measured counts N is compared to the total predicted number of counts N_{pred}

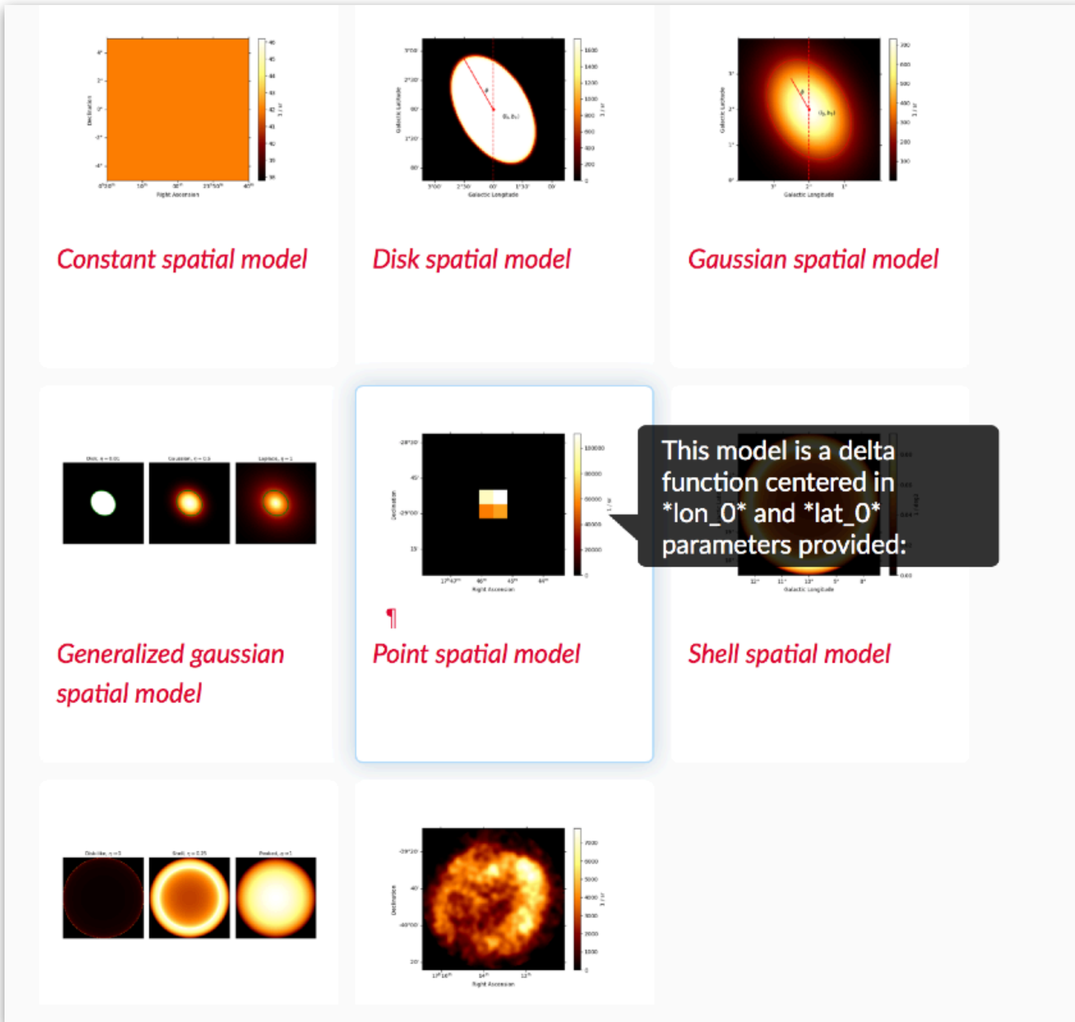
$$N_{\mathrm{pred}}(p, E) = \sum_{S} N_S(p, E) + N_{\mathrm{bkg}}(p, E)$$

- Model parameter estimation is performed through maximum likelihood technique.
  - [Cash statistics]() is used for counts data with a known background

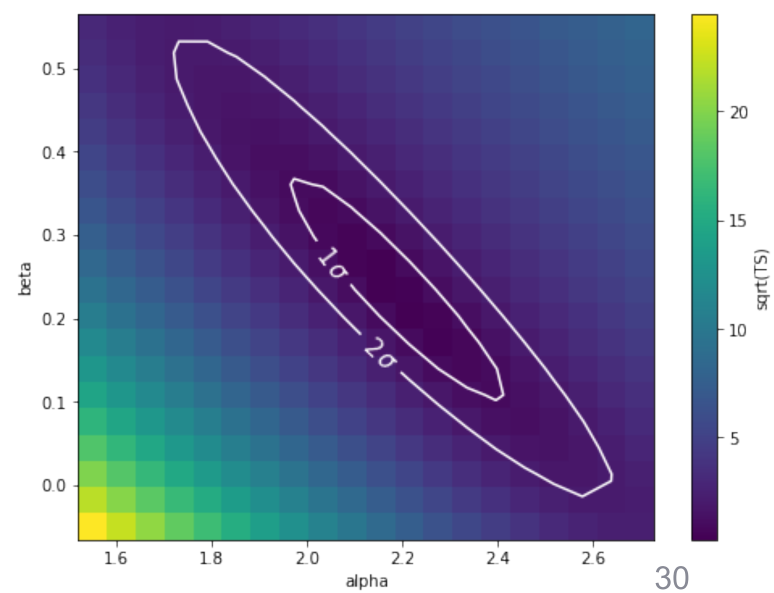
$$TS = -2 \log L = 2 \sum \left(N \log N_{\mathrm{pred}} - N_{\mathrm{pred}} \right)$$

  - [Wstat statistics]() is used for counts data with a measured background

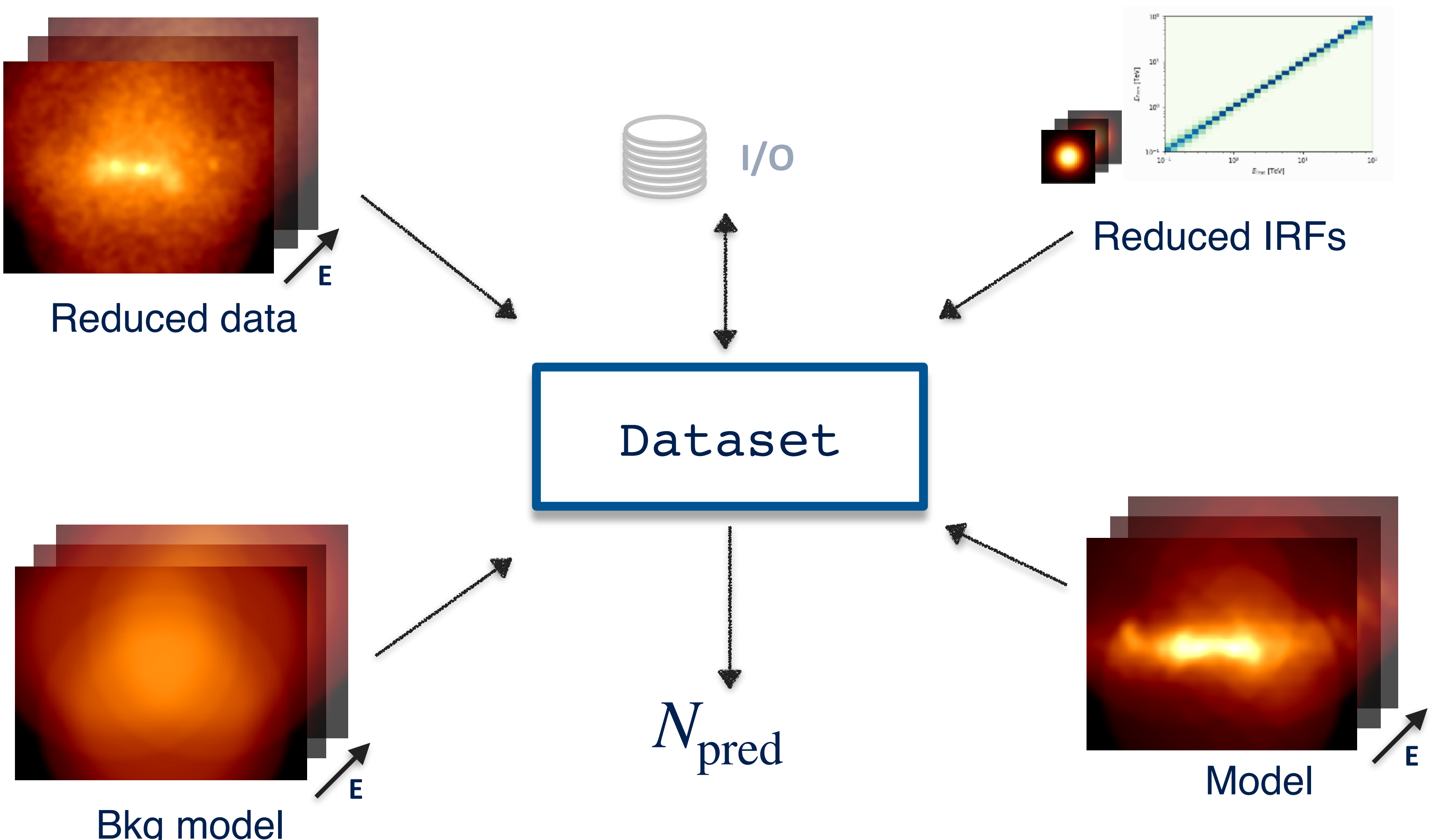
Datasets modeling and fitting



A library of models and a Fitting interface

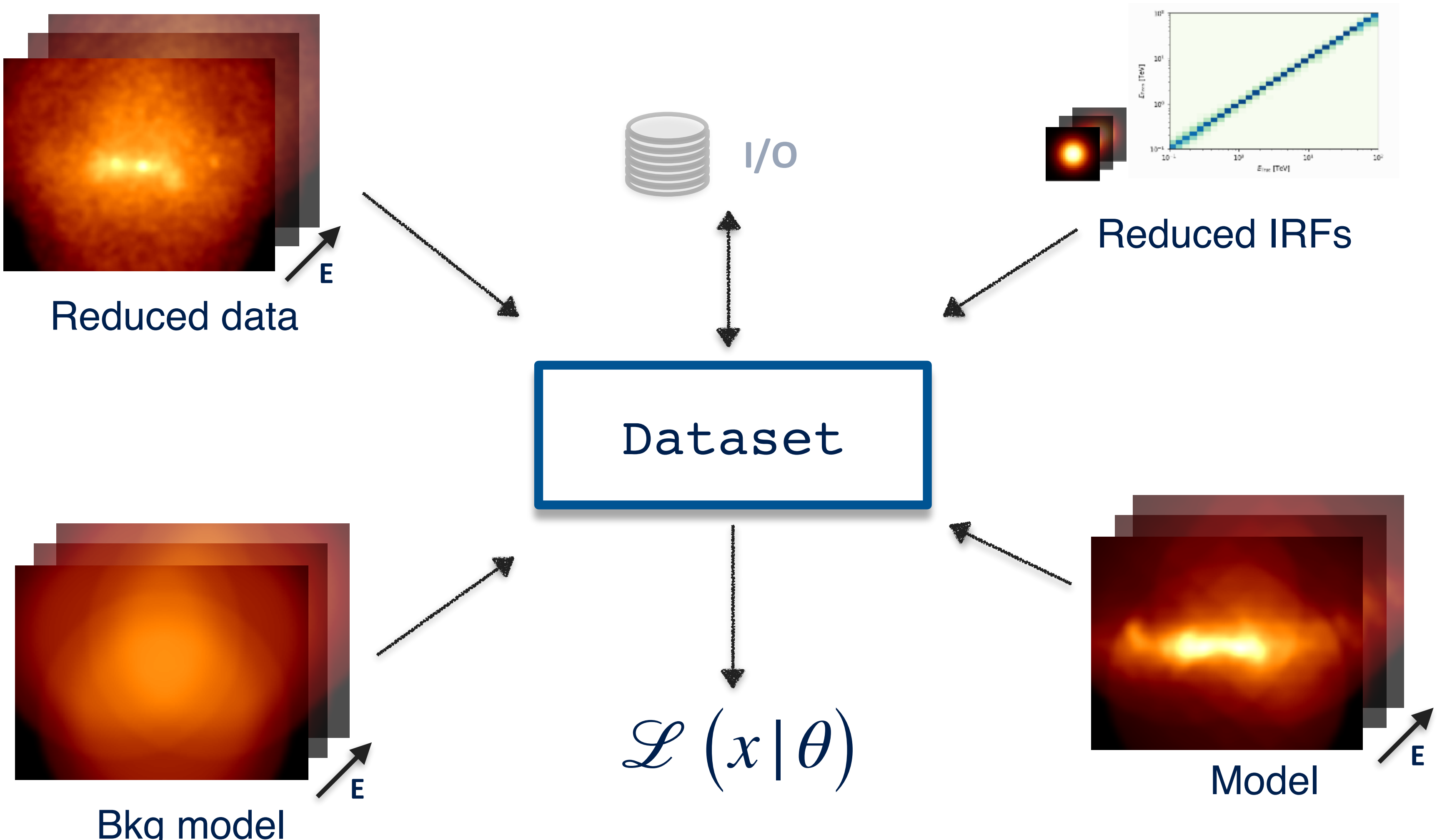


Datasets modeling and fitting



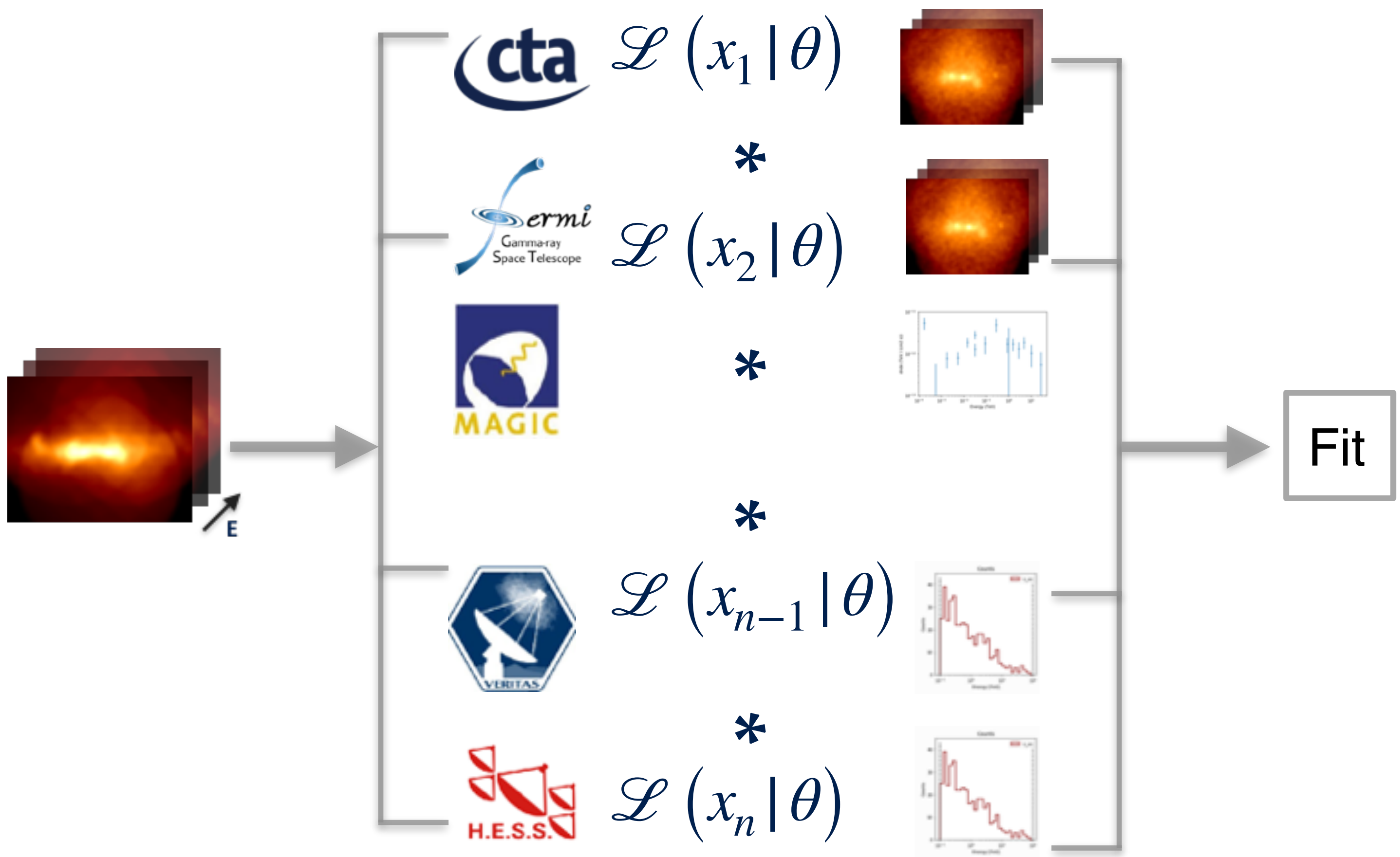
see:[Dataset simulation tutorial]

Datasets modeling and fitting



see: [Dataset fitting tutorial]

Multi-instrument modeling and fitting

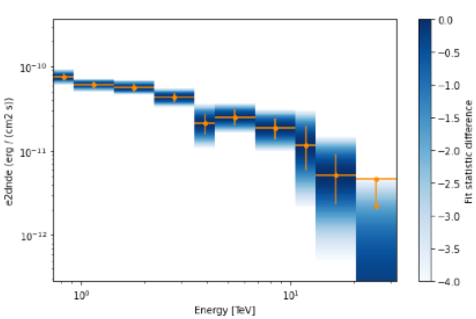


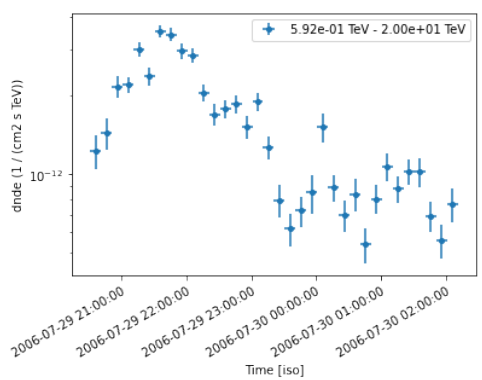
Gammapy Dataset structure allows heterogeneous data modeling and fitting:

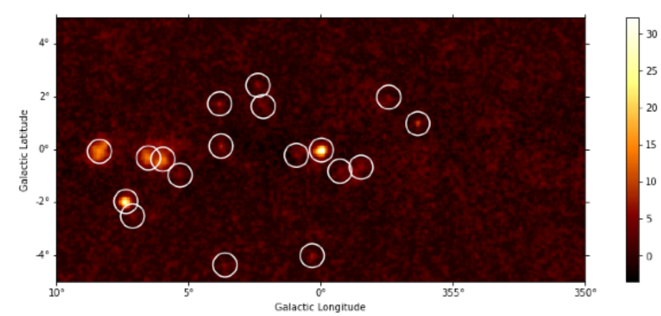
- See [joint fit tutorial]

DL5 products: 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







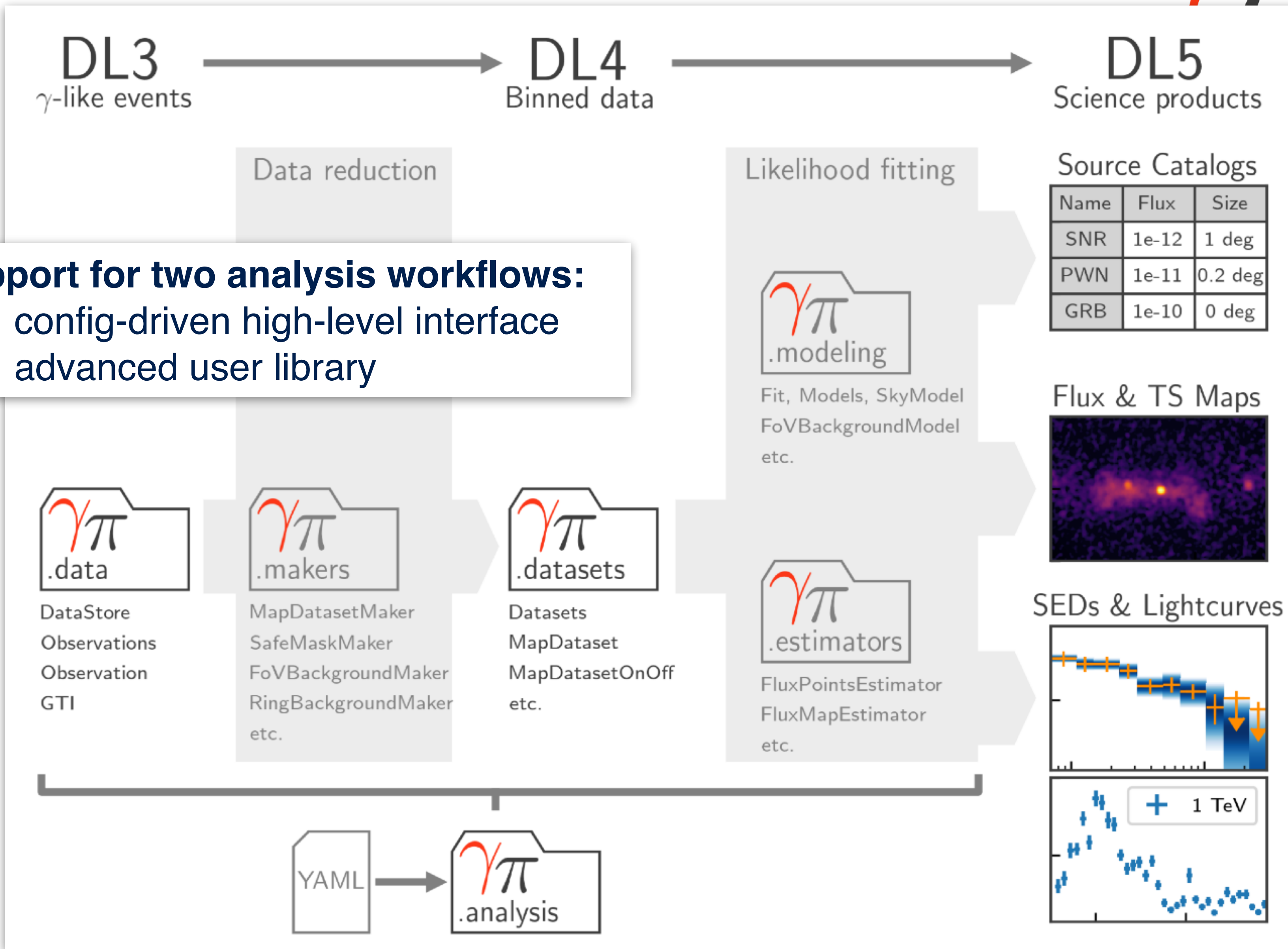
Estimating statistical significance

- Estimate whether H_1 is statistically preferred over the reference H_0
- It is possible to compare the two *nested models* (i.e. H_0 is a subset of H_1) with the maximum likelihood ratio test
  - $\Delta TS = TS_1 - TS_0$ follows asymptotically a χ^2 with n degrees of freedom
    - allows to determine p-value of e.g. a source component
  - with 1 degree of freedom $\sqrt{\Delta TS}$ gives a statistical significance as a number of « gaussian sigma »

Data workflow and package structure

support for two analysis workflows:

- config-driven high-level interface
- advanced user library



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

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]

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]

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]

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.

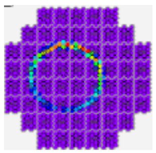
See [High Level Interface tutorial]

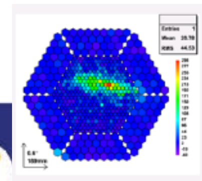
Questions & Comments

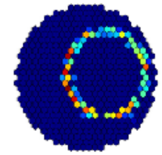
- Where/How to interact with dev team and experienced users, provide feedback, get help:
  - gammapy.slack.
    - In particular: #help channel
  - [GitHub discussions]
    - help category
  - [GitHub issues] to report bugs or feature requests

Thank you!

An example of joint analysis



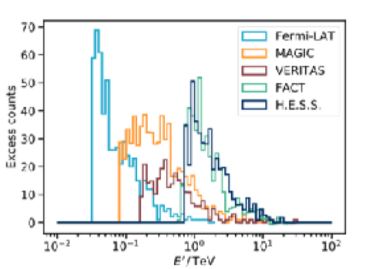


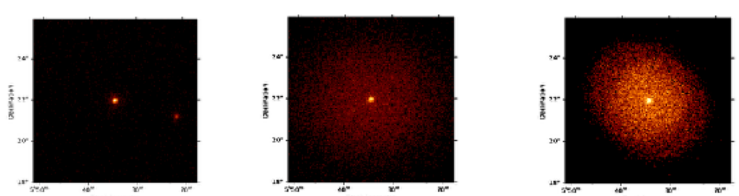


Raw data

Dataset	T_{obs}	E_{min} (TeV)	E_{max} (TeV)	N_{on}	N_{bkg}	R_{on} (deg)
Fermi-LAT	~7 yr	0.03	2	578	1.2	0.30
MAGIC	0.66 h	0.08	30	784	129.9	0.14
VERITAS	0.67 h	0.16	30	289	13.7	0.10
FACT	10.33 h	0.45	30	691	272.8	0.17
H.E.S.S.	1.87 h	0.71	30	459	27.5	0.11

DL3 gadf

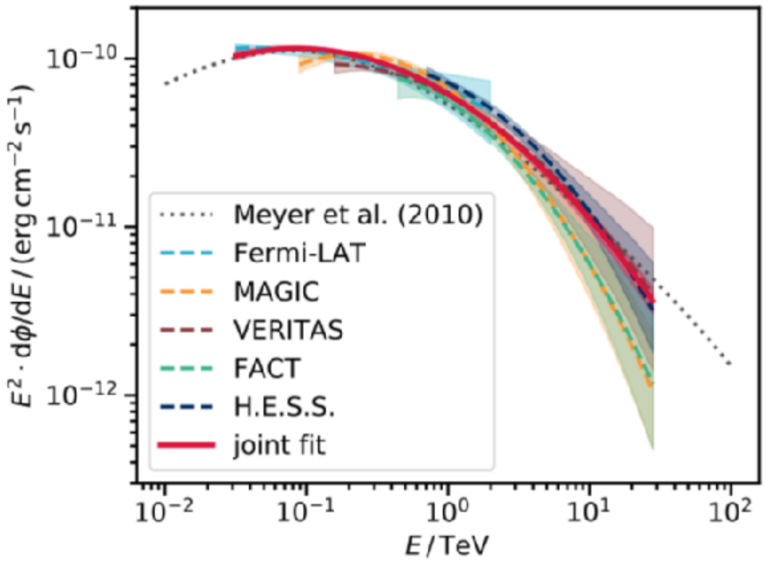


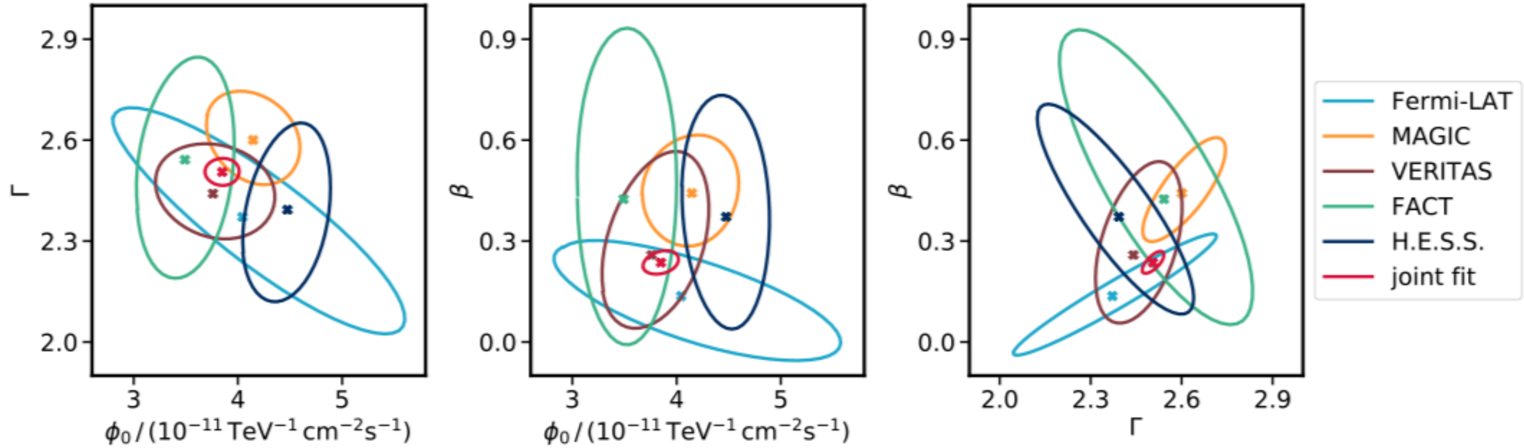


High level
science products

An example of joint analysis

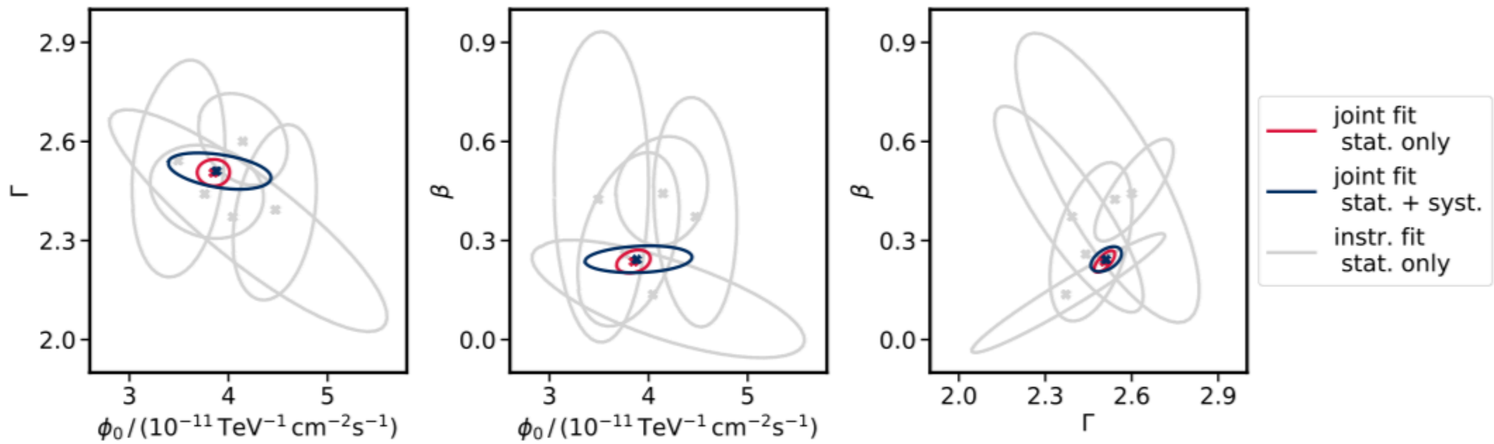
- joint point-like analysis
- log-parabola fit using ON-OFF likelihood





An example of joint analysis

- Can perform inter-calibration studies to evaluate systematics:
  - e.g. uncertainties on energy scale



- Can perform spectral fits on the parent particle population