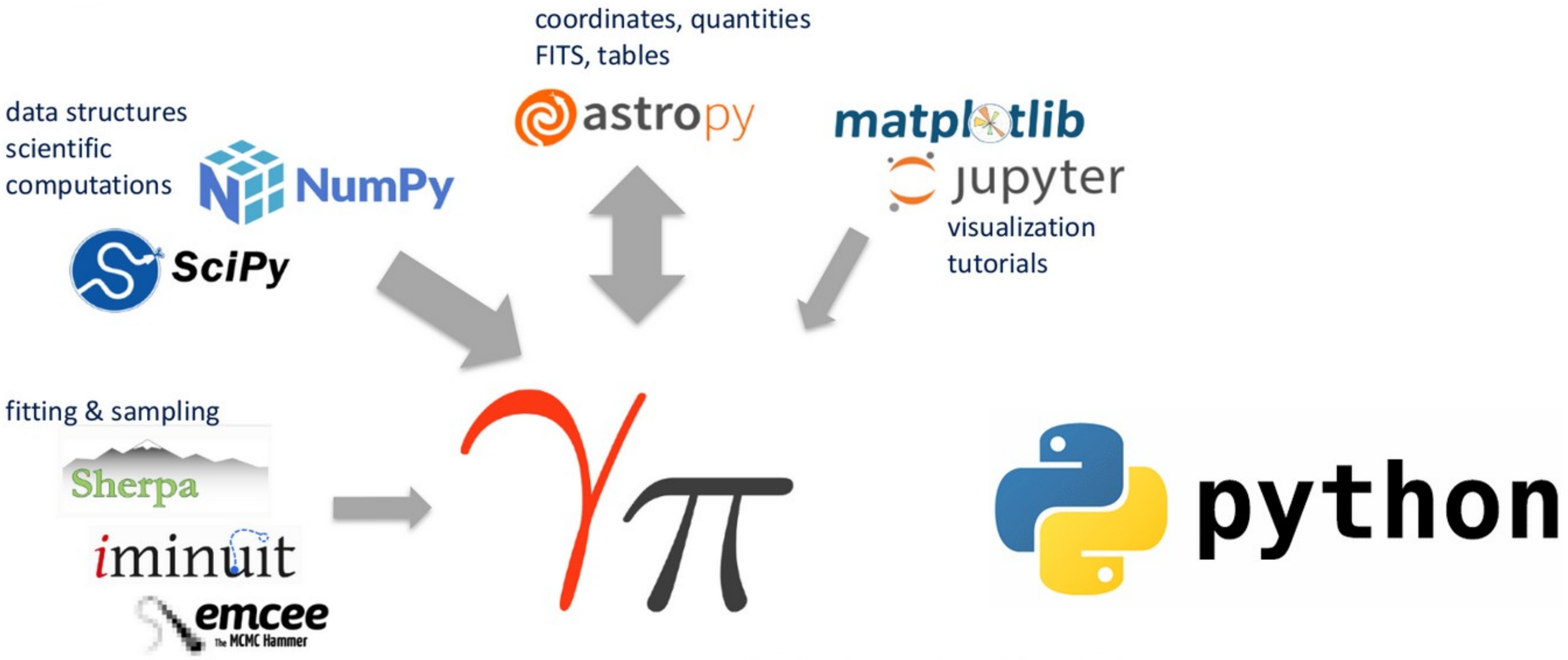
Introduction to Gammapy

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

Sexten School
July 18, 2022

Introduction to Gammapy



data structures scientific computations

coordinates, quantities FITS, tables

visualization tutorials

fitting & sampling

v0.20 released on May 12th

Gammapy: project organisation

Coordination Committee

APC

AIM

l'Observatoire de Paris

MAX-PLANCK-INSTITUT FÜR KERNPHYSIK

DESY

ecap ERLANGEN CENTRE FOR ASTROPARTICLE PHYSICS

UNIVERSIDAD COMPLUTENSE MADRID

IAA INSTITUTO DE ASTROFÍSICA DE ANDALUCÍA

INAF ISTITUTO NAZIONALE DI ASTROFISICA NATIONAL INSTITUTE FOR ASTROPHYSICS

Project Management

B. Khelifi (APC)

C. van Eldik (ECAP)

Lead Development

A. Donath (CfA)

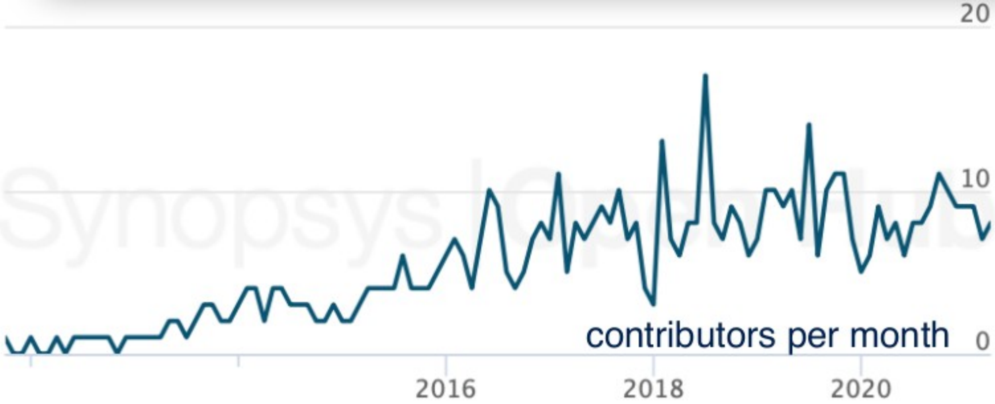
R. Terrier (APC)

Gammapy: developers & contributors

Core development team

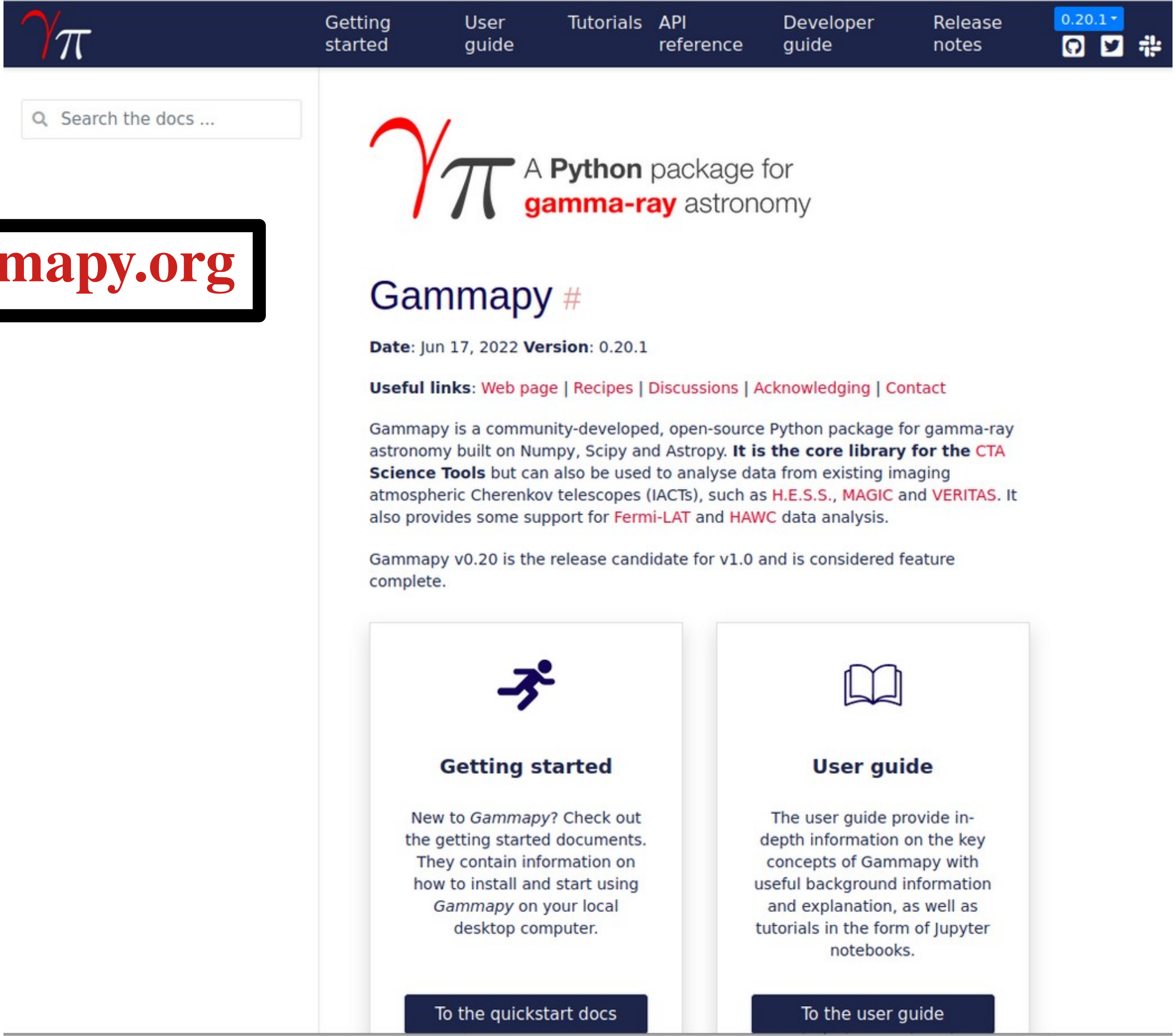
≈ 10 regular core developers

≈ 80 contributors so far



Open development on [GitHub]

docs.gammapy.org



Gammapy

Date: Jun 17, 2022 **Version**: 0.20.1

Useful links: Web page | Recipes | Discussions | Acknowledging | Contact

Gammapy is a community-developed, open-source Python package for gamma-ray astronomy built on Numpy, Scipy and Astropy. **It is the core library for the** CTA **Science Tools** but can also be used to analyse data from existing imaging atmospheric Cherenkov telescopes (IACTs), such as H.E.S.S., MAGIC and VERITAS. It also provides some support for Fermi-LAT and HAWC data analysis.

Gammapy v0.20 is the release candidate for v1.0 and is considered feature complete.

Getting started

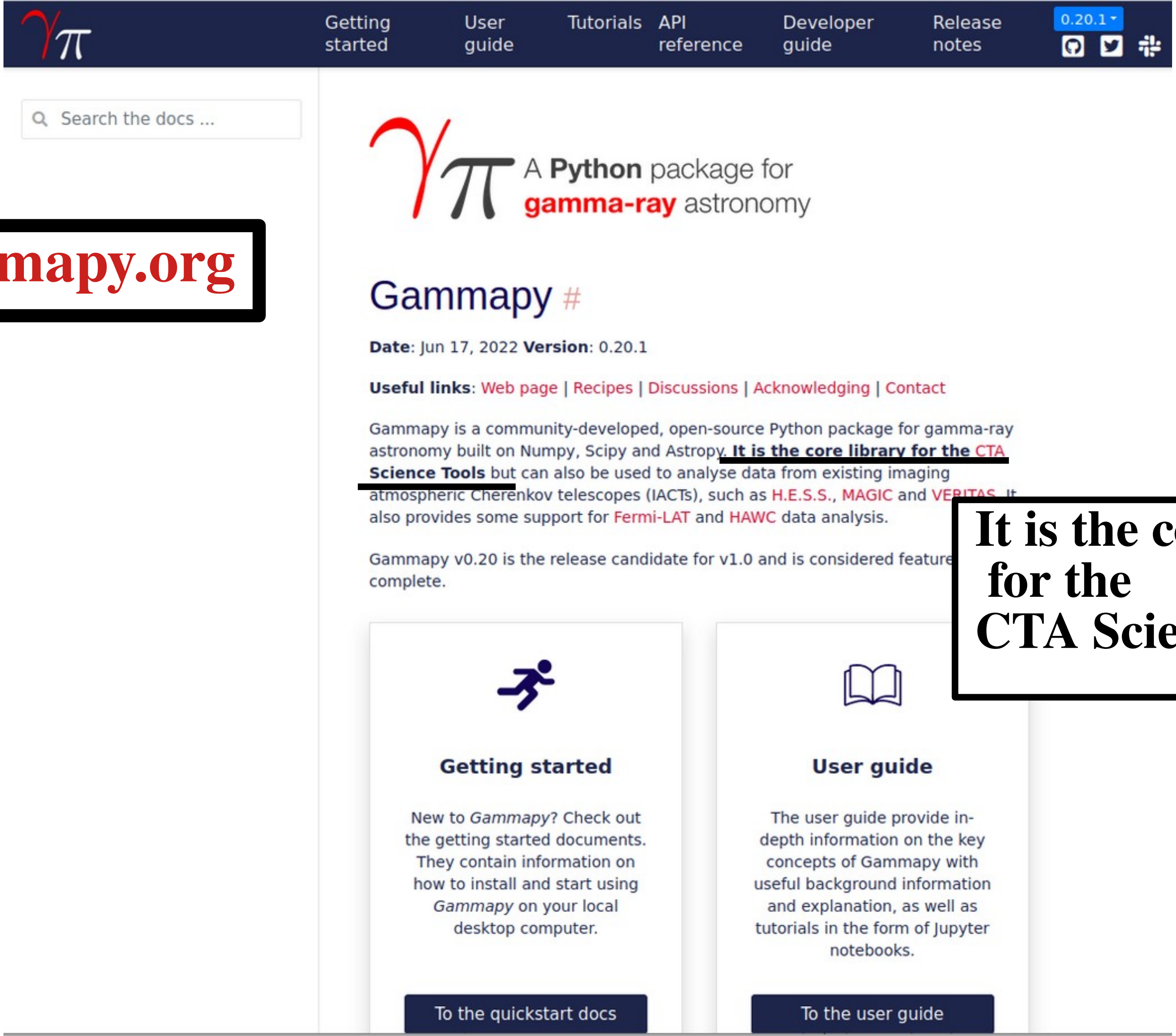
New to *Gammapy*? Check out the getting started documents. They contain information on how to install and start using *Gammapy* on your local desktop computer.

User guide

The user guide provide in-depth information on the key concepts of Gammapy with useful background information and explanation, as well as tutorials in the form of Jupyter notebooks.

cta
cherenkov telescope array

docs.gammapy.org



Gammapy

Date: Jun 17, 2022 **Version**: 0.20.1

Useful links: Web page | Recipes | Discussions | Acknowledging | Contact

Gammapy is a community-developed, open-source Python package for gamma-ray astronomy built on Numpy, Scipy and Astropy. **It is the core library for the** CTA **Science Tools** but can also be used to analyse data from existing imaging atmospheric Cherenkov telescopes (IACTs), such as H.E.S.S., MAGIC and VERITAS. It also provides some support for Fermi-LAT and HAWC data analysis.

Gammapy v0.20 is the release candidate for v1.0 and is considered feature complete.

Getting started

New to *Gammapy*? Check out the getting started documents. They contain information on how to install and start using *Gammapy* on your local desktop computer.

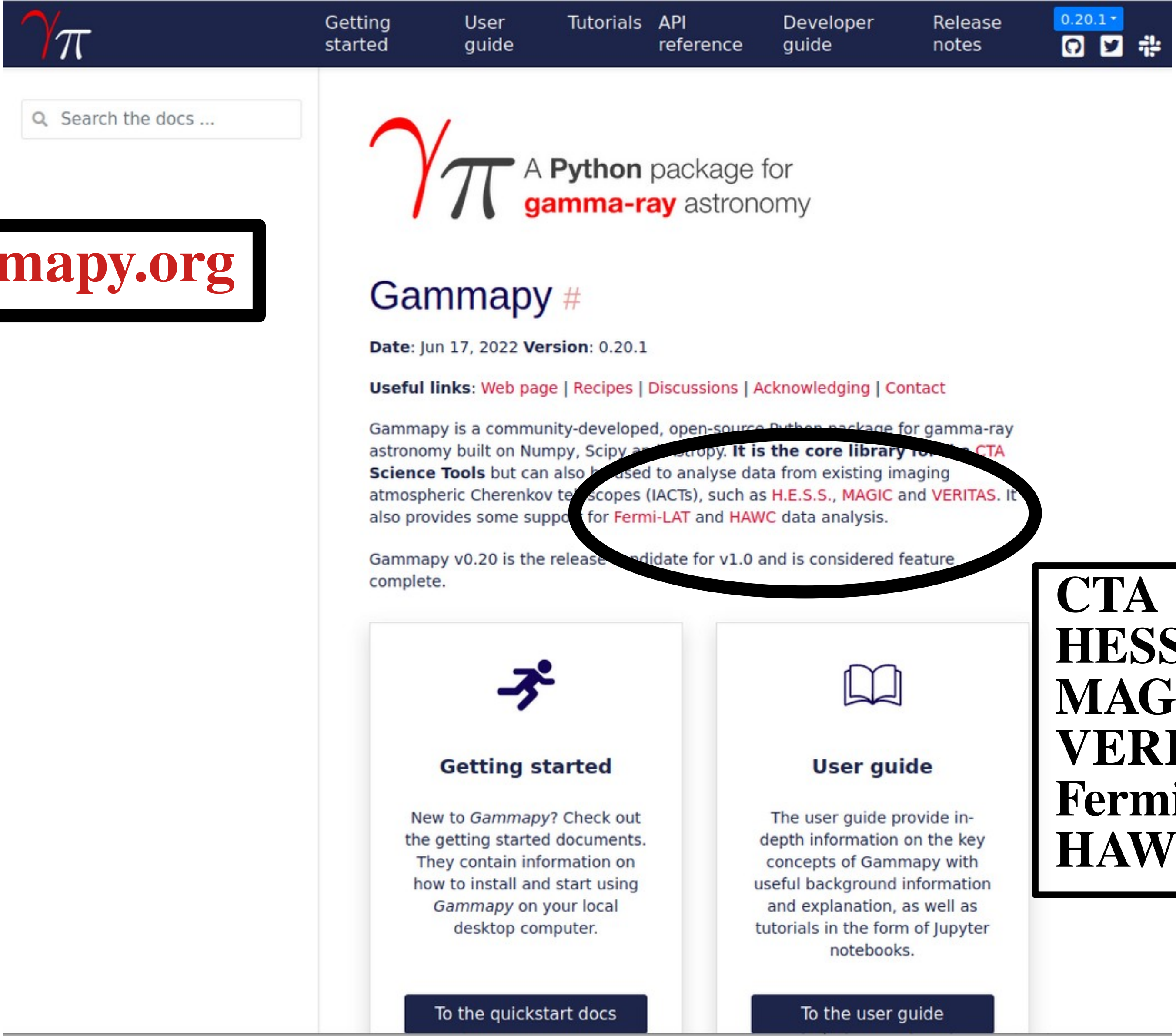
User guide

The user guide provide in-depth information on the key concepts of Gammapy with useful background information and explanation, as well as tutorials in the form of Jupyter notebooks.

It is the core library for the CTA Science Tools

cta
cherenkov telescope array

docs.gammapy.org



Gammapy

Date: Jun 17, 2022 **Version**: 0.20.1

Useful links: Web page | Recipes | Discussions | Acknowledging | Contact

Gammapy is a community-developed, open-source Python package for gamma-ray astronomy built on Numpy, Scipy and Astropy. **It is the core library for the CTA Science Tools** but can also be used to analyse data from existing imaging atmospheric Cherenkov telescopes (IACTs), such as H.E.S.S., MAGIC and VERITAS. It also provides some support for Fermi-LAT and HAWC data analysis.

Gammapy v0.20 is the release candidate for v1.0 and is considered feature complete.

Getting started

New to *Gammapy*? Check out the getting started documents. They contain information on how to install and start using *Gammapy* on your local desktop computer.

To the quickstart docs

User guide

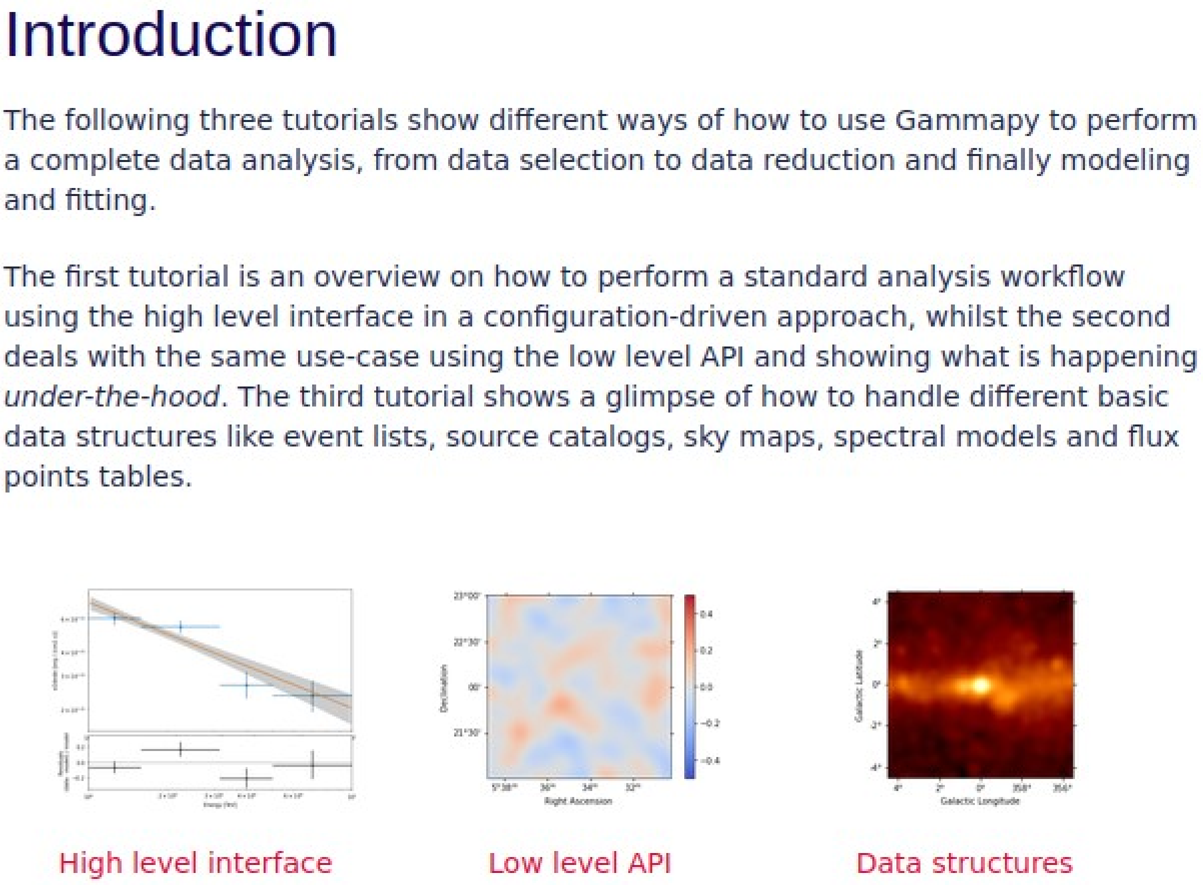
The user guide provide in-depth information on the key concepts of Gammapy with useful background information and explanation, as well as tutorials in the form of Jupyter notebooks.

To the user guide

CTA
HESS
MAGIC
VERITAS
Fermi-LAT
HAWC

docs.gammapy.org

Tutorials to learn simple data analisys recipes:



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

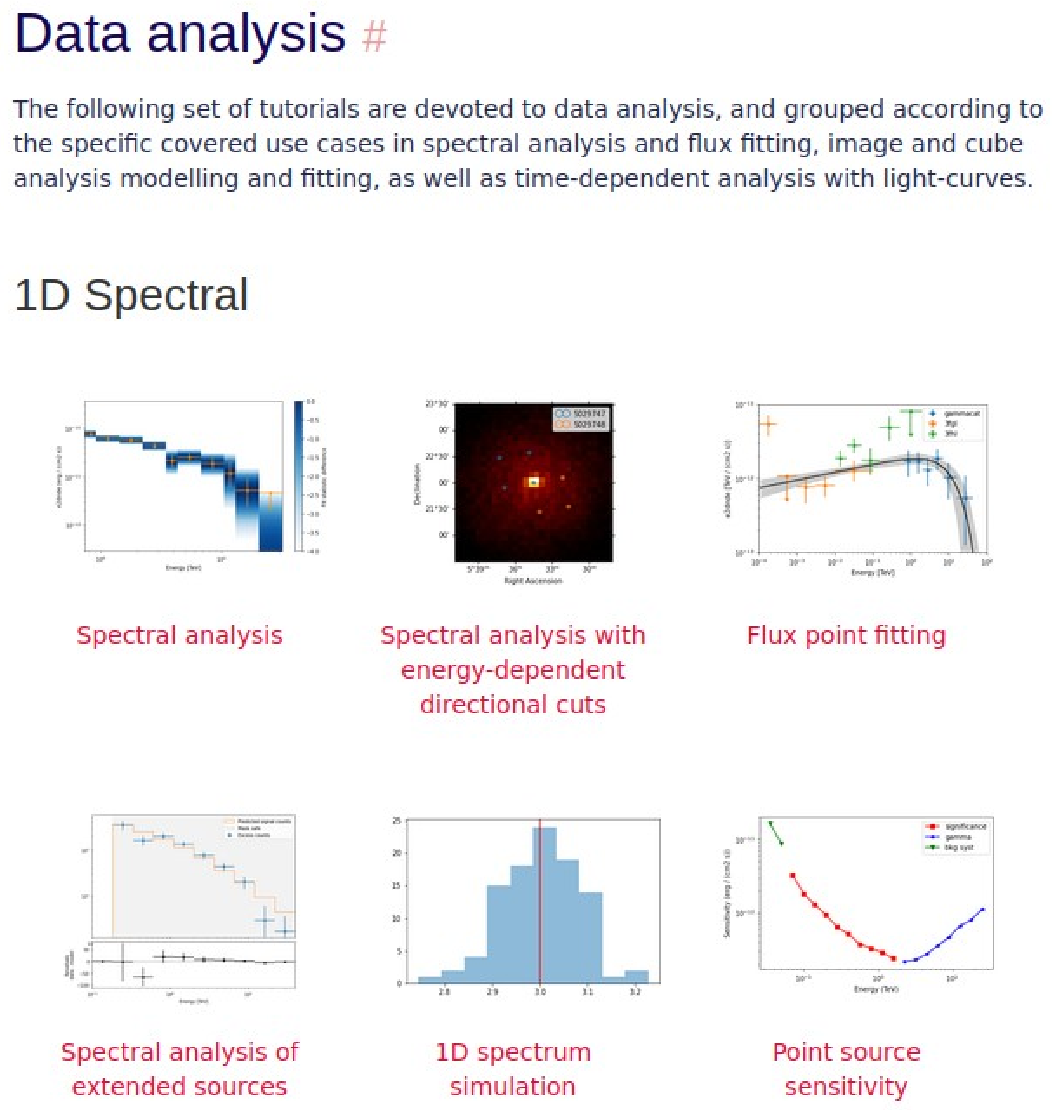
Low level API

Data structures

docs.gammapy.org

Tutorials to learn simple data analisys recipes:

- **spectral analisys**



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

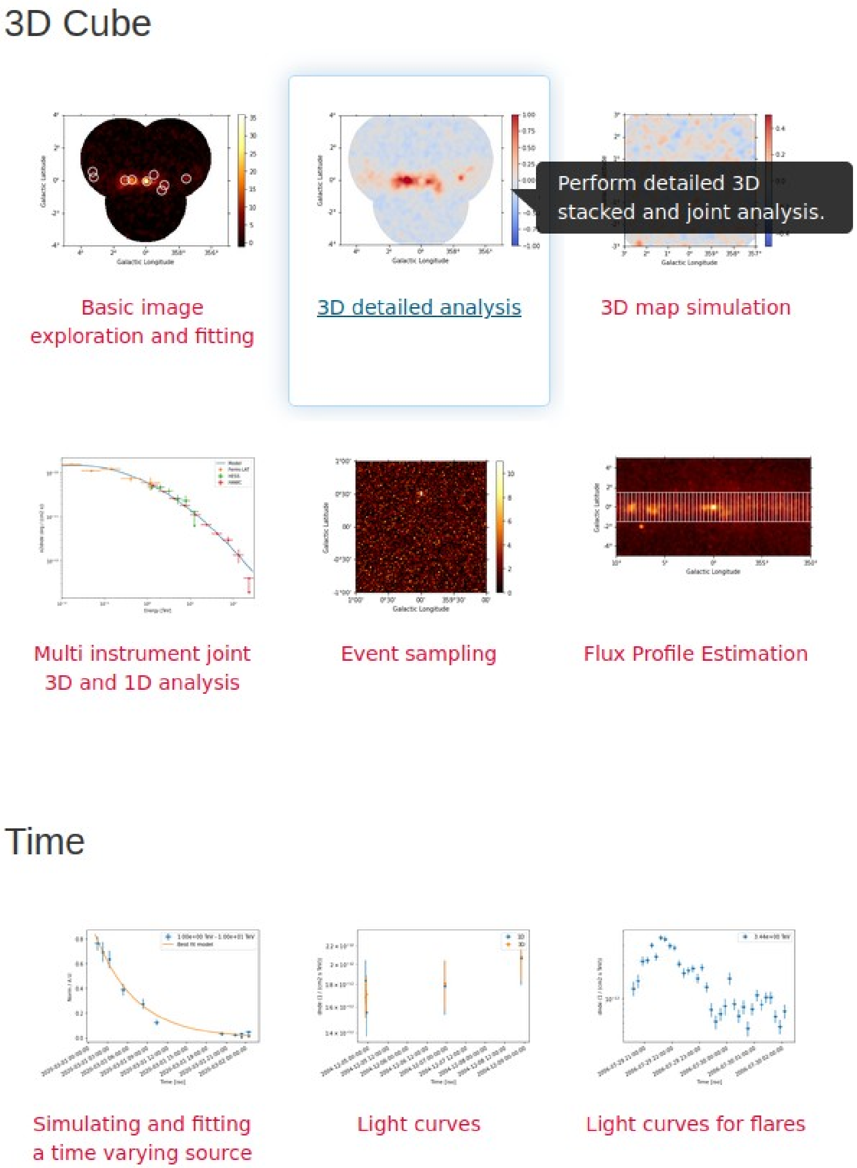
1D spectrum simulation

Point source sensitivity

docs.gammapy.org

Tutorials to learn simple data analisys recipes:

- spectral analisys
- 3D fitting
- light curve extraction
- simulation



3D Cube

Basic image exploration and fitting

3D detailed analysis

Perform detailed 3D stacked and joint analysis.

3D map simulation

Multi instrument joint 3D and 1D analysis

Event sampling

Flux Profile Estimation

Time

Simulating and fitting a time varying source

Light curves

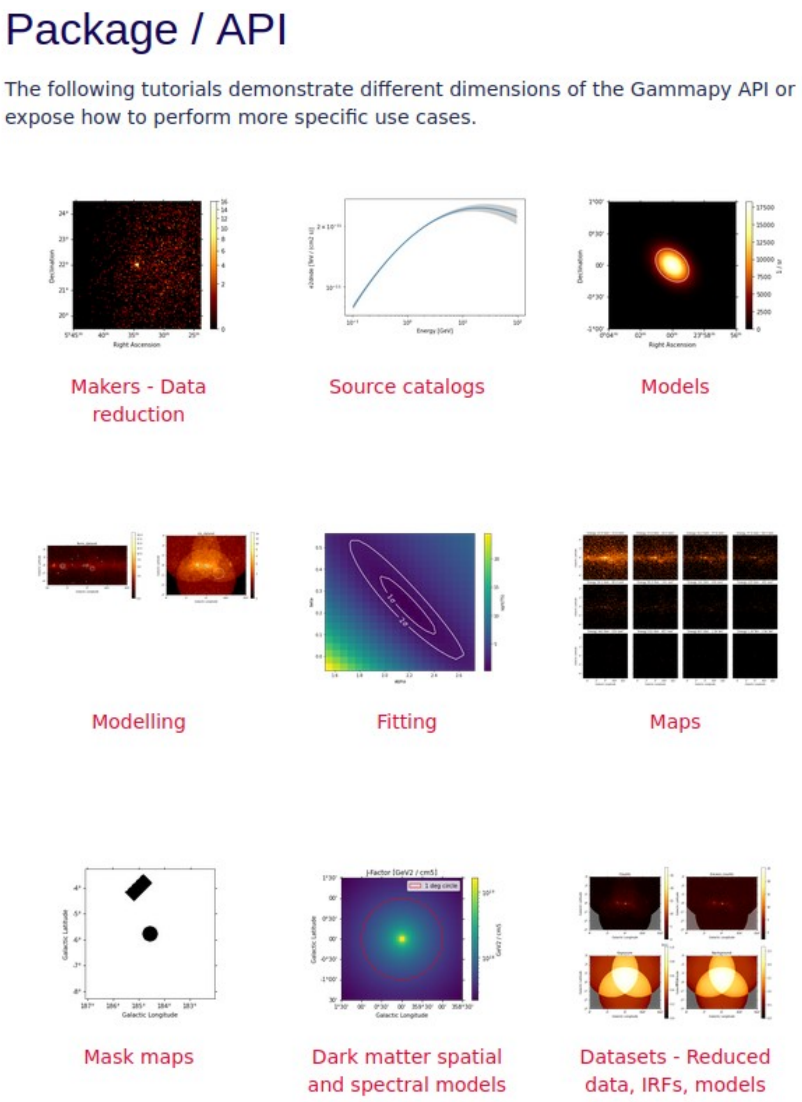
Light curves for flares

cta
cherenkov telescope array

docs.gammapy.org

Learn how to use the general API:

- **go beyond tutorials use cases**
- **exploit Gammapy flexibility**



Package / API

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

Makers - Data reduction

Source catalogs

Models

Modelling

Fitting

Maps

Mask maps

Dark matter spatial and spectral models

Datasets - Reduced data, IRFs, models

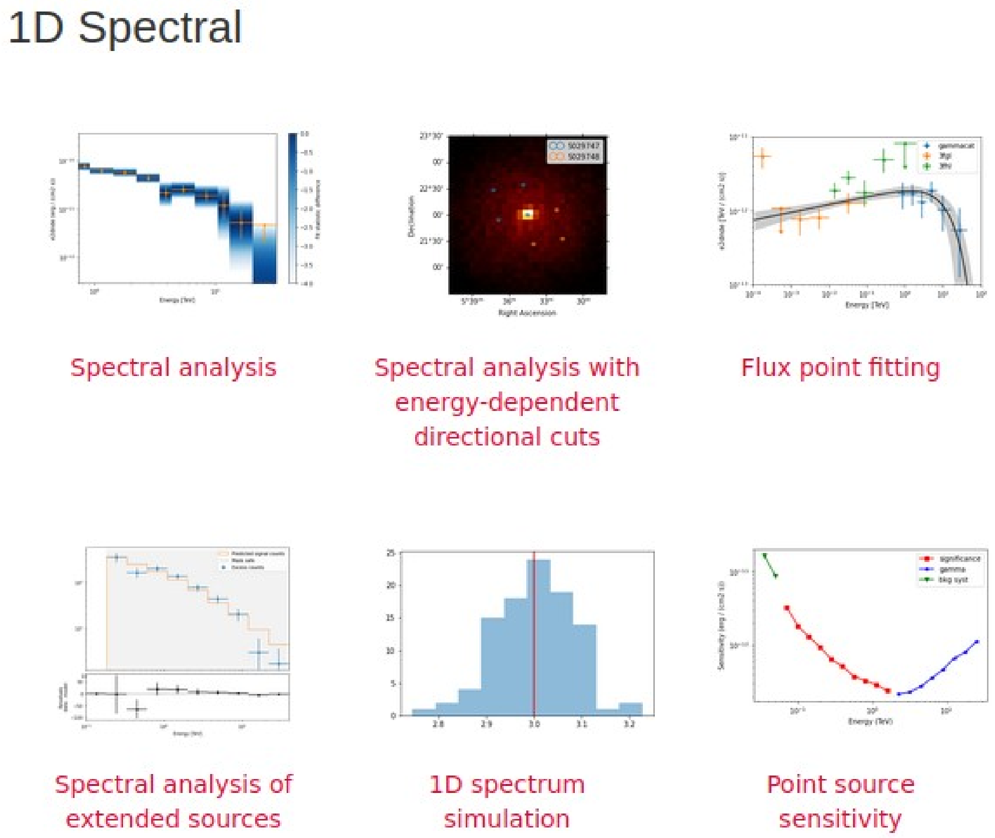
Getting help

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

cta
cherenkov telescope array

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



1D Spectral

Spectral analysis

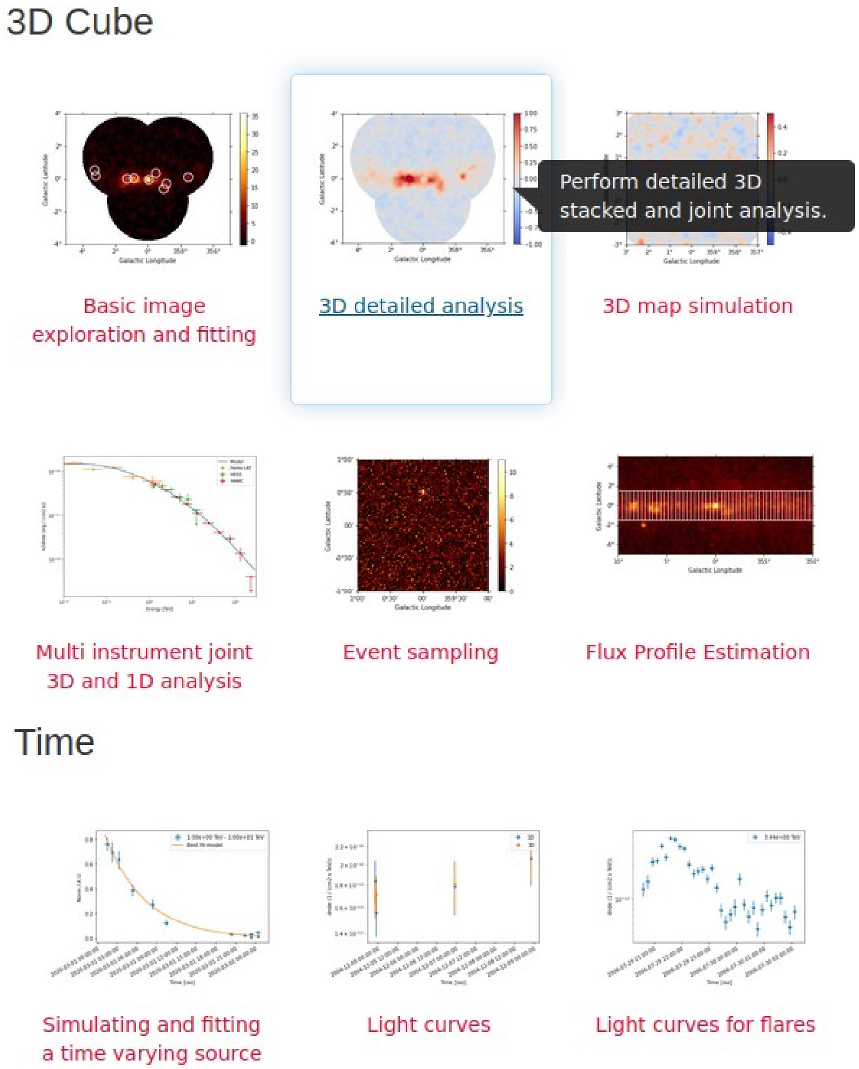
Spectral analysis with energy-dependent directional cuts

Flux point fitting

Spectral analysis of extended sources

1D spectrum simulation

Point source sensitivity



3D Cube

Basic image exploration and fitting

3D detailed analysis

Perform detailed 3D stacked and joint analysis.

3D map simulation

Multi instrument joint 3D and 1D analysis

Event sampling

Flux Profile Estimation

Time

Simulating and fitting a time varying source

Light curves

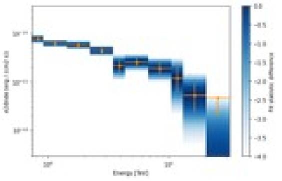
Light curves for flares

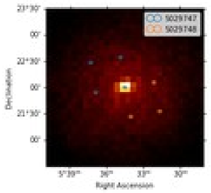
All analysis types follow the same workflow and the same API

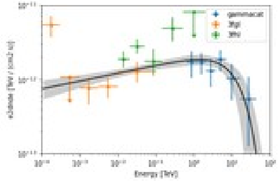
cta cherenkov telescope array

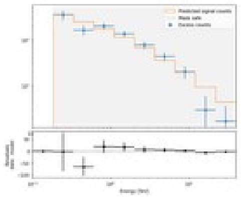
γπ A **Python** package for **gamma-ray** astronomy

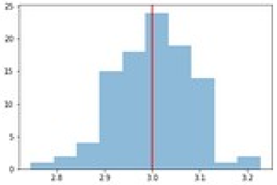
1D Spectral

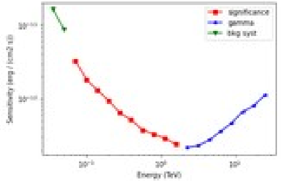

Spectral analysis


Spectral analysis with energy-dependent directional cuts

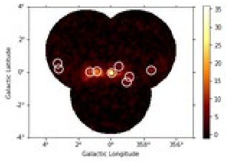

Flux point fitting

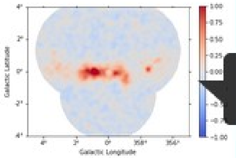

Spectral analysis of extended sources

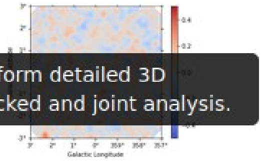

1D spectrum simulation


Point source sensitivity

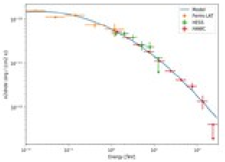
3D Cube

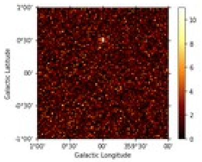

Basic image exploration and fitting

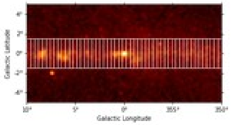

3D detailed analysis


Perform detailed 3D stacked and joint analysis.

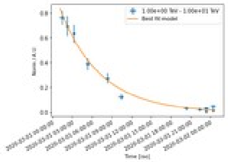
3D map simulation

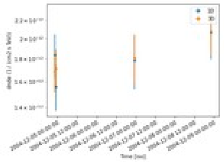

Multi instrument joint 3D and 1D analysis

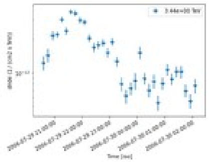

Event sampling


Flux Profile Estimation

Time


Simulating and fitting a time varying source


Light curves

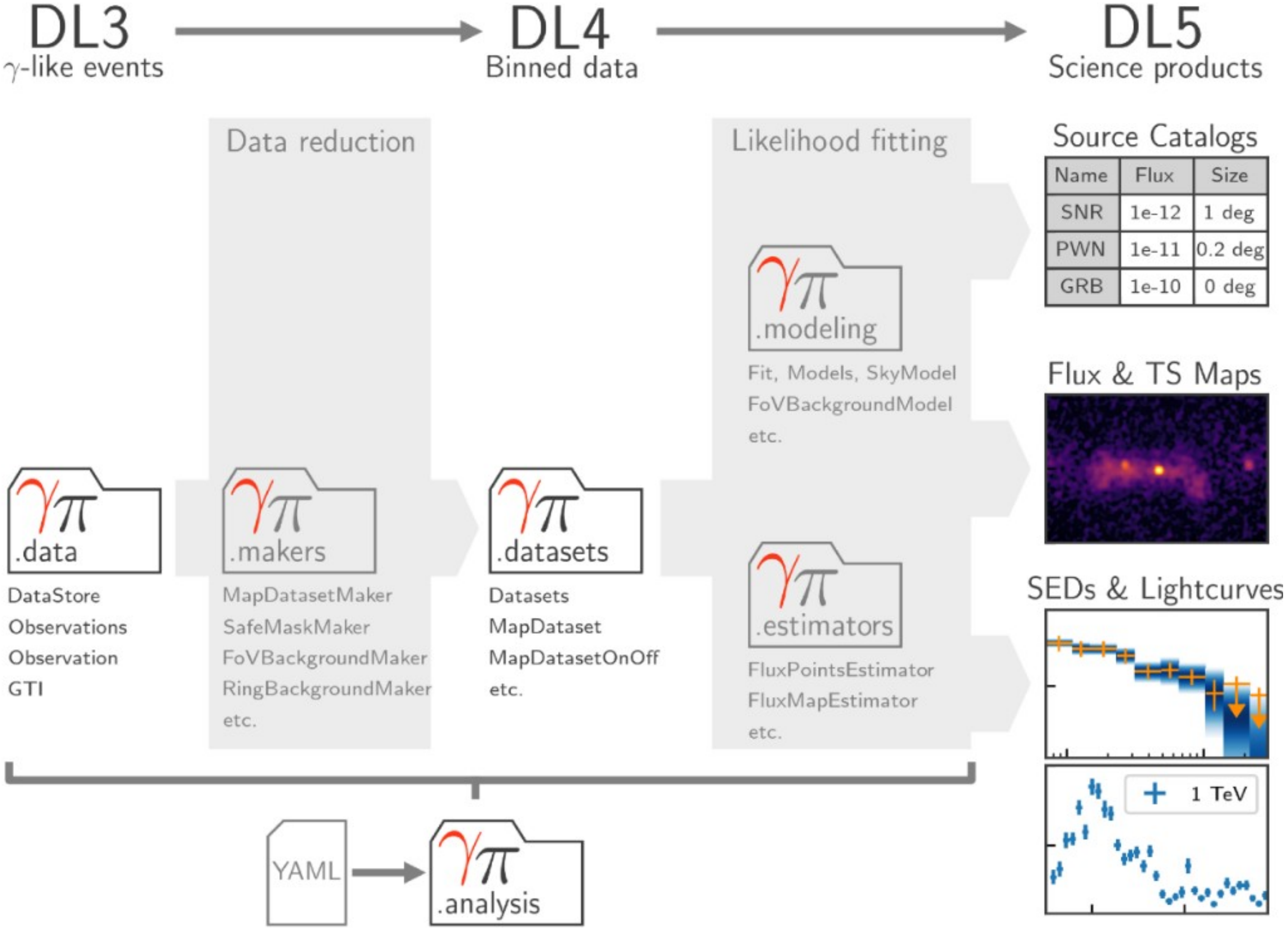

Light curves for flares

All analysis types follow the same workflow and the same API

Data workflow and package structure

Data workflow and package structure

CTA
HESS
MAGIC
VERITAS
Fermi-LAT
HAWC



DL3
γ-like events

DL4
Binned data

DL5
Science products

Data reduction

Likelihood fitting

.data

DataStore
Observations
Observation
GTI

.makers

MapDatasetMaker
SafeMaskMaker
FoVBackgroundMaker
RingBackgroundMaker
etc.

.datasets

Datasets
MapDataset
MapDatasetOnOff
etc.

.modeling

Fit, Models, SkyModel
FoVBackgroundModel
etc.

.estimators

FluxPointsEstimator
FluxMapEstimator
etc.

YAML

.analysis

Source Catalogs

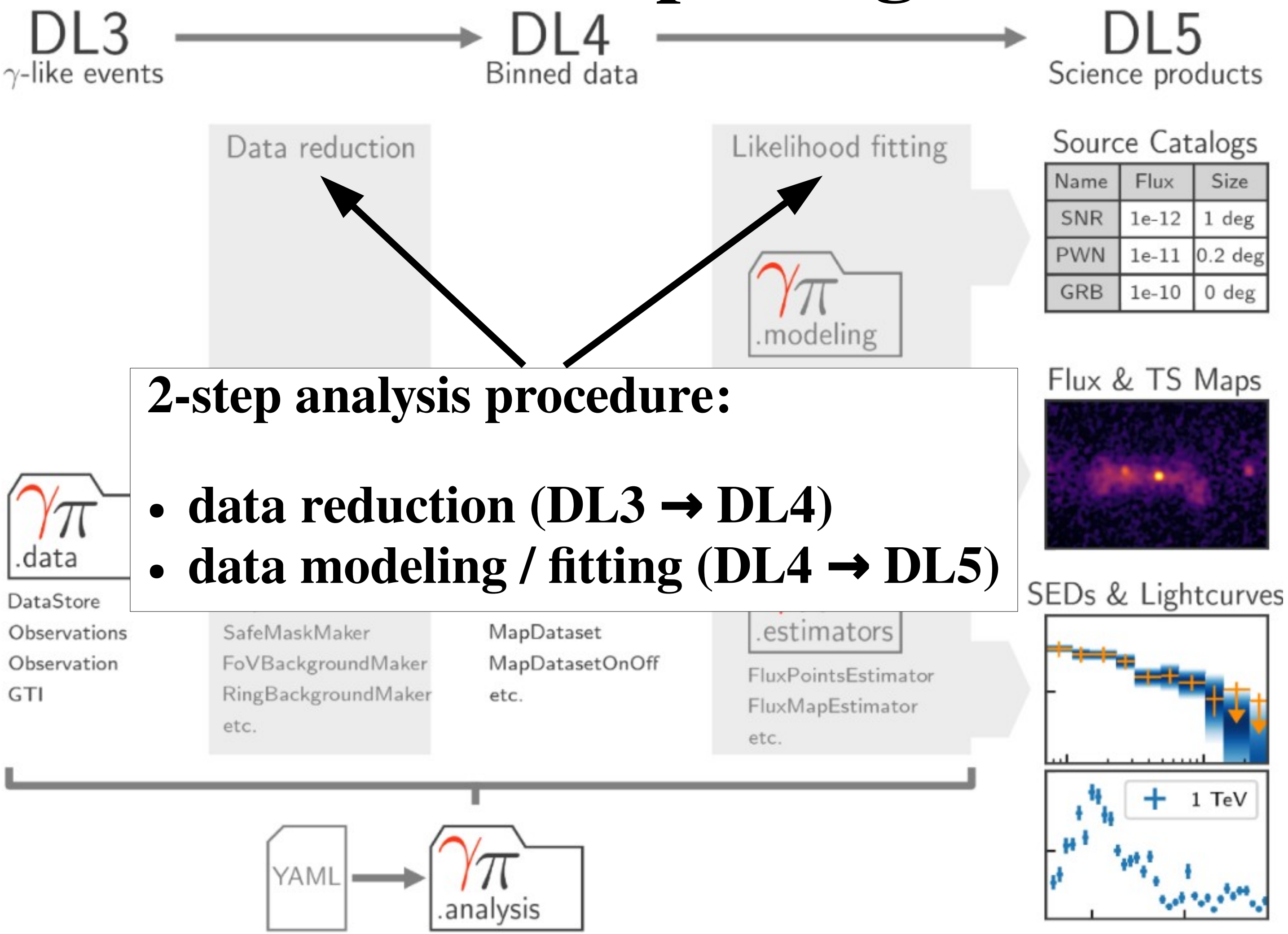
Name	Flux	Size
SNR	1e-12	1 deg
PWN	1e-11	0.2 deg
GRB	1e-10	0 deg

Flux & TS Maps

SEDs & Lightcurves

Data workflow and package structure

CTA
HESS
MAGIC
VERITAS
Fermi-LAT
HAWC



Name	Flux	Size
SNR	1e-12	1 deg
PWN	1e-11	0.2 deg
GRB	1e-10	0 deg

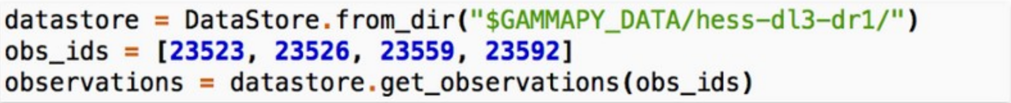
2-step analysis procedure:

- **data reduction (DL3 → DL4)**
- **data modeling / fitting (DL4 → DL5)**

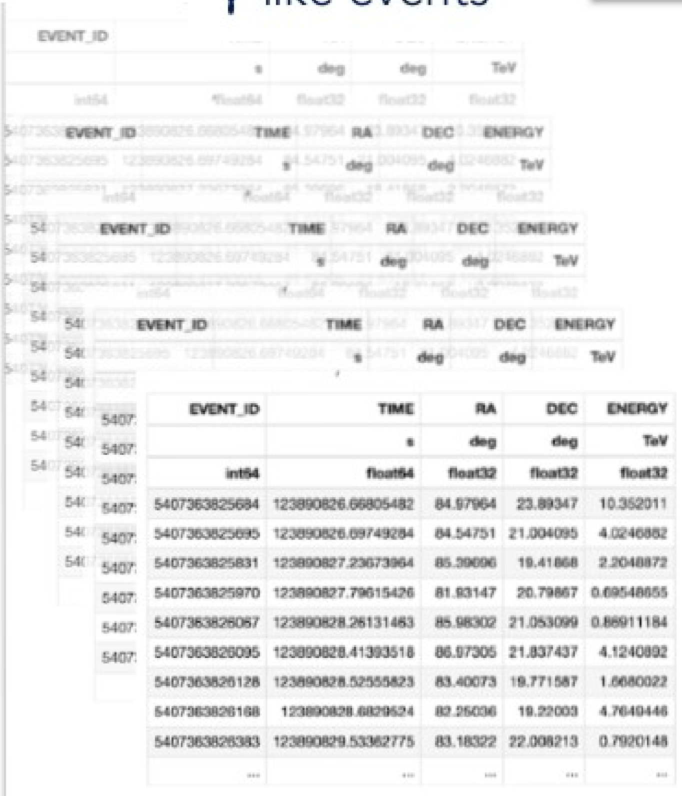
Data reduction (DL3 → DL4)

DL3

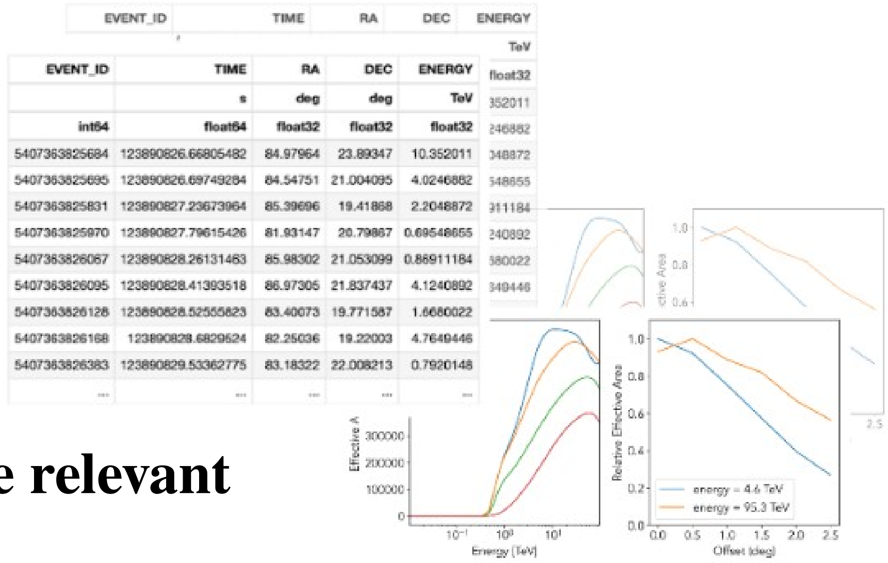
γ-like events



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



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	123890828.6829624	82.25036	19.22003	4.7649446
5407363826383	123890829.53362775	83.18322	22.008213	0.7920148
...	...	...	...	...



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	123890828.6829624	82.25036	19.22003	4.7649446
5407363826383	123890829.53362775	83.18322	22.008213	0.7920148
...	...	...	...	...

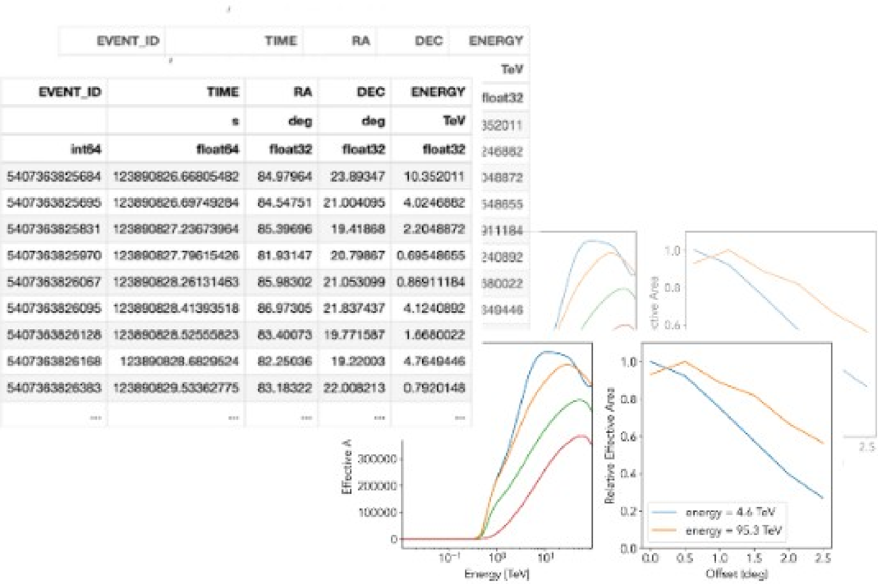
Select and retrieve relevant observations

Data reduction (DL3 → DL4)

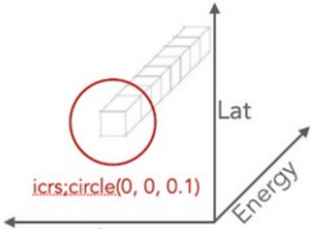
A **Python** package for **gamma-ray** astronomy

Define the reduced dataset geometry:

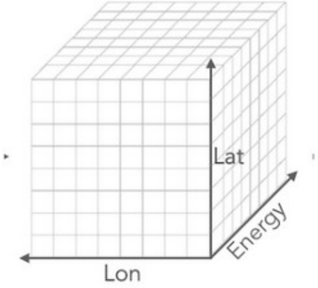
- **Is the analysis 1D (spectral only) or 3D?**
- **Define target binning and projection**



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	123890828.6829624	82.25036	19.22003	4.7649446
5407363826383	123890829.53362775	83.18322	22.008213	0.7920148
...	...	...	...	...



region & energy



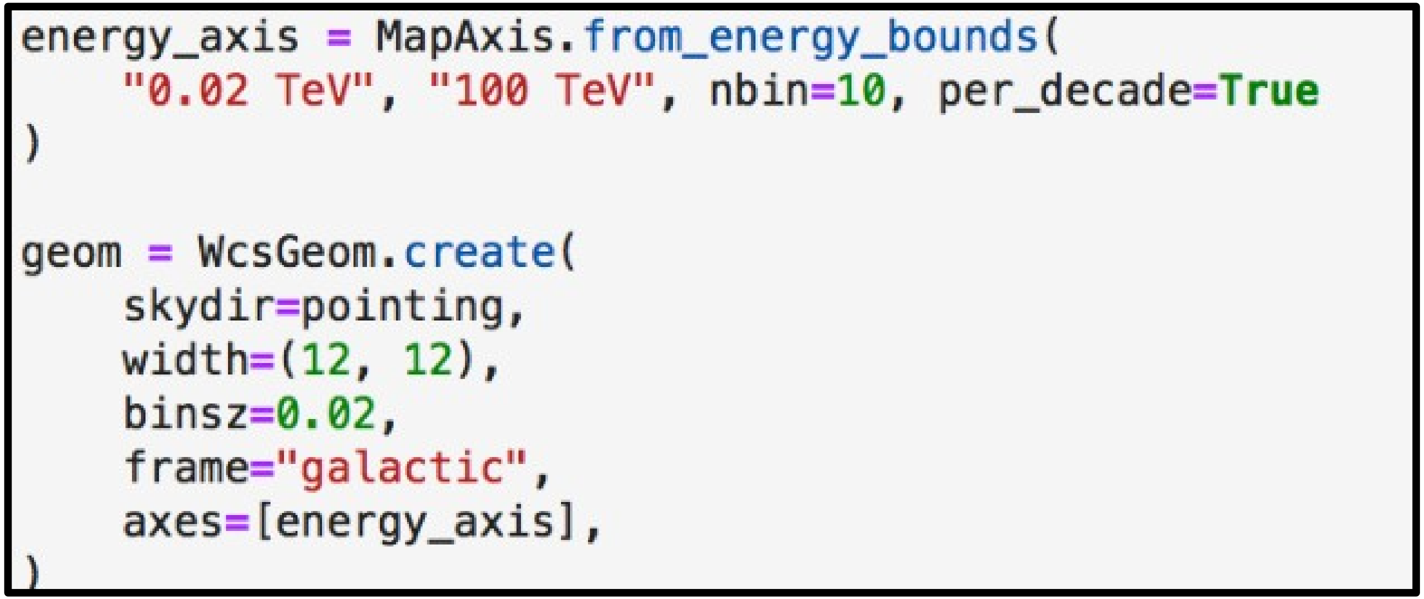
WCS & energy

Data reduction (DL3 → DL4)

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

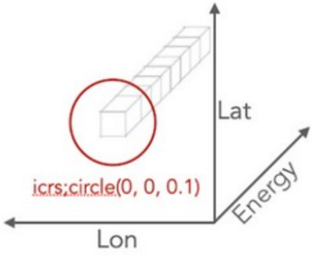
Define the reduced dataset geometry:

- **Is the analysis 1D (spectral only) or 3D?**
- **Define target binning and projection**

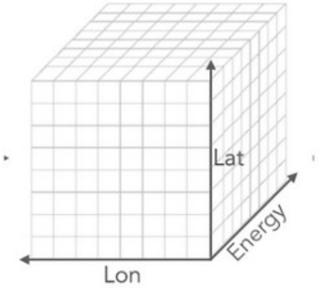


```
energy_axis = MapAxis.from_energy_bounds(
    "0.02 TeV", "100 TeV", nbin=10, per_decade=True
)

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



region & energy



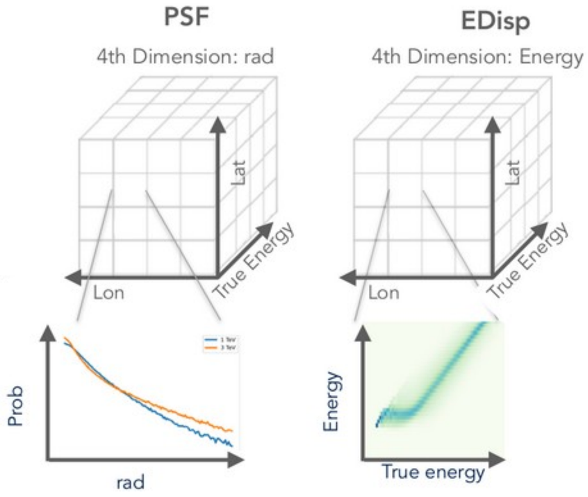
WCS & energy

Data reduction (DL3 → DL4)

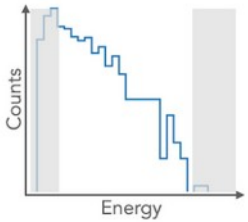
Initialize the data reduction methods:

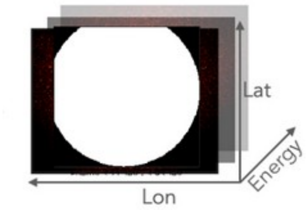
- **Data and IRF projection**
- **Safe mask determination**
- **Background estimation**

DL3 IRFs are reprojected by the DatasetMaker on the target geometry

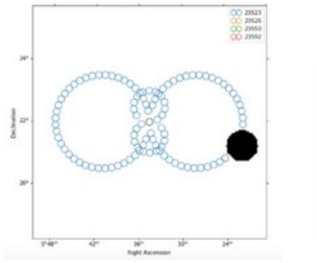


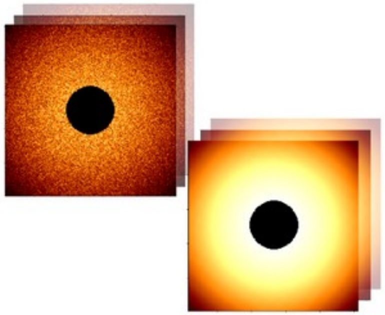
Safe mask



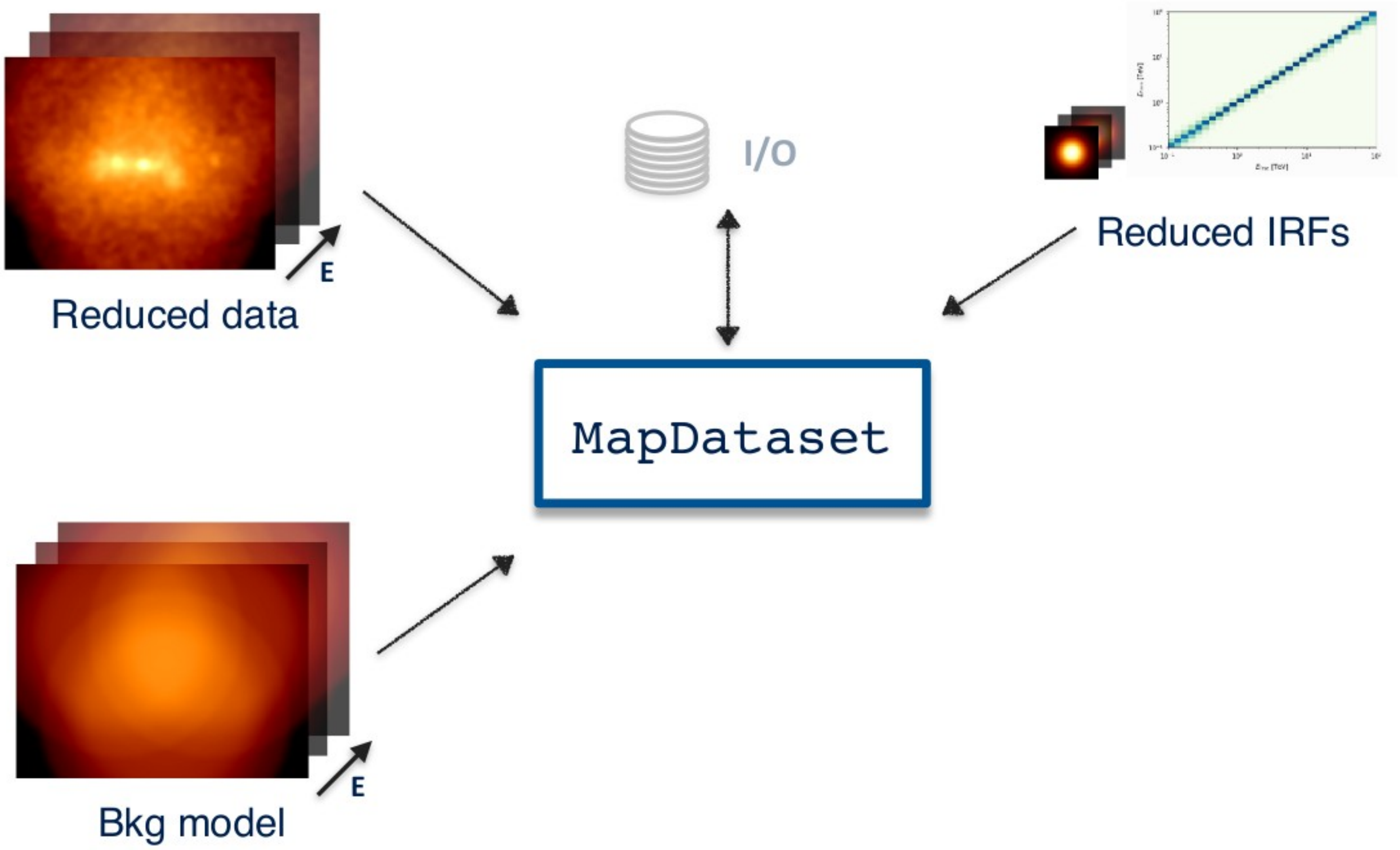


Background estimation





DL4 structure: Datasets



Reduced data

I/O

Reduced IRFs

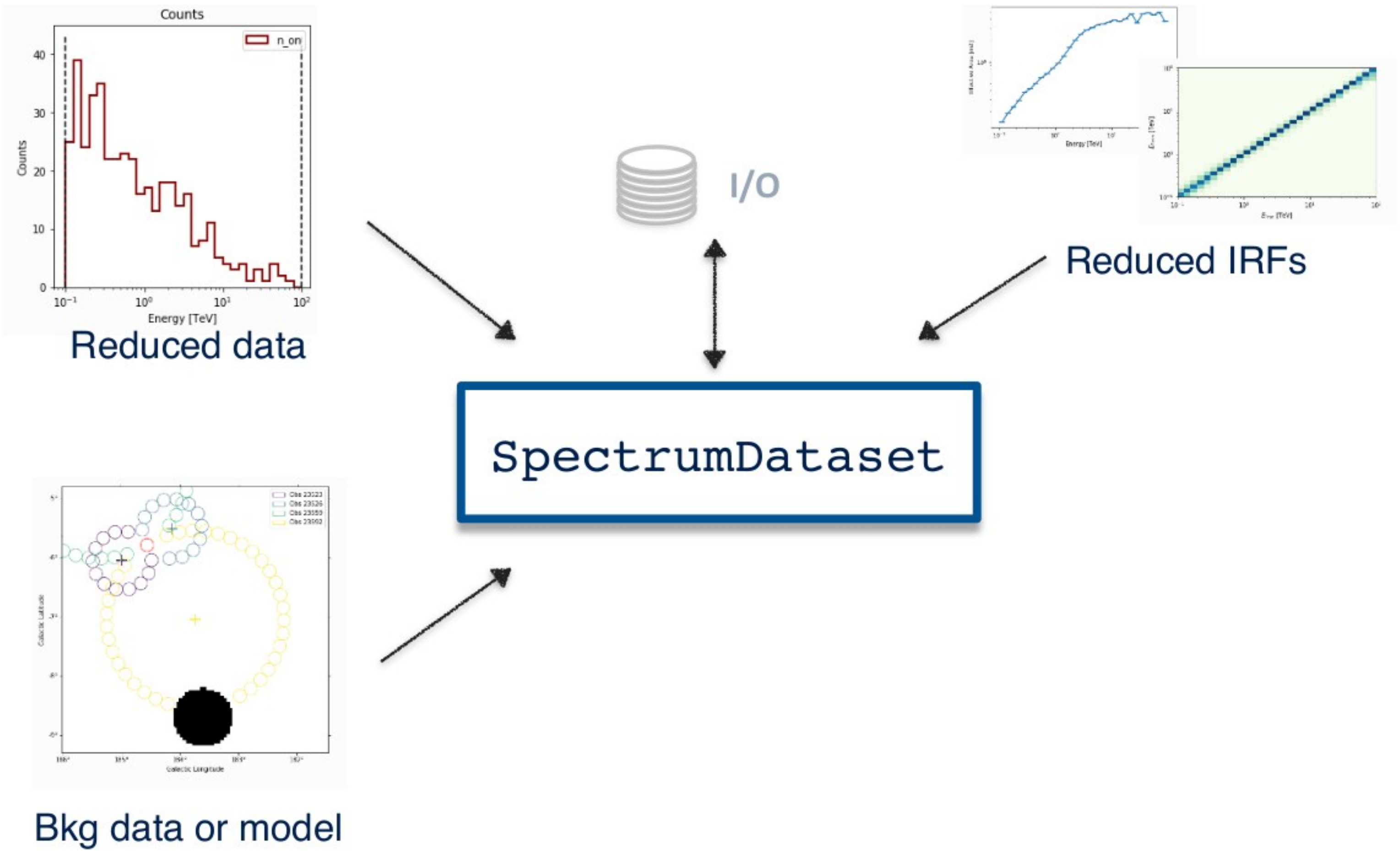
MapDataset

Bkg model

cta cherenkov telescope array

DL4 structure: Datasets

A Python package for gamma-ray astronomy



Reduced data

I/O

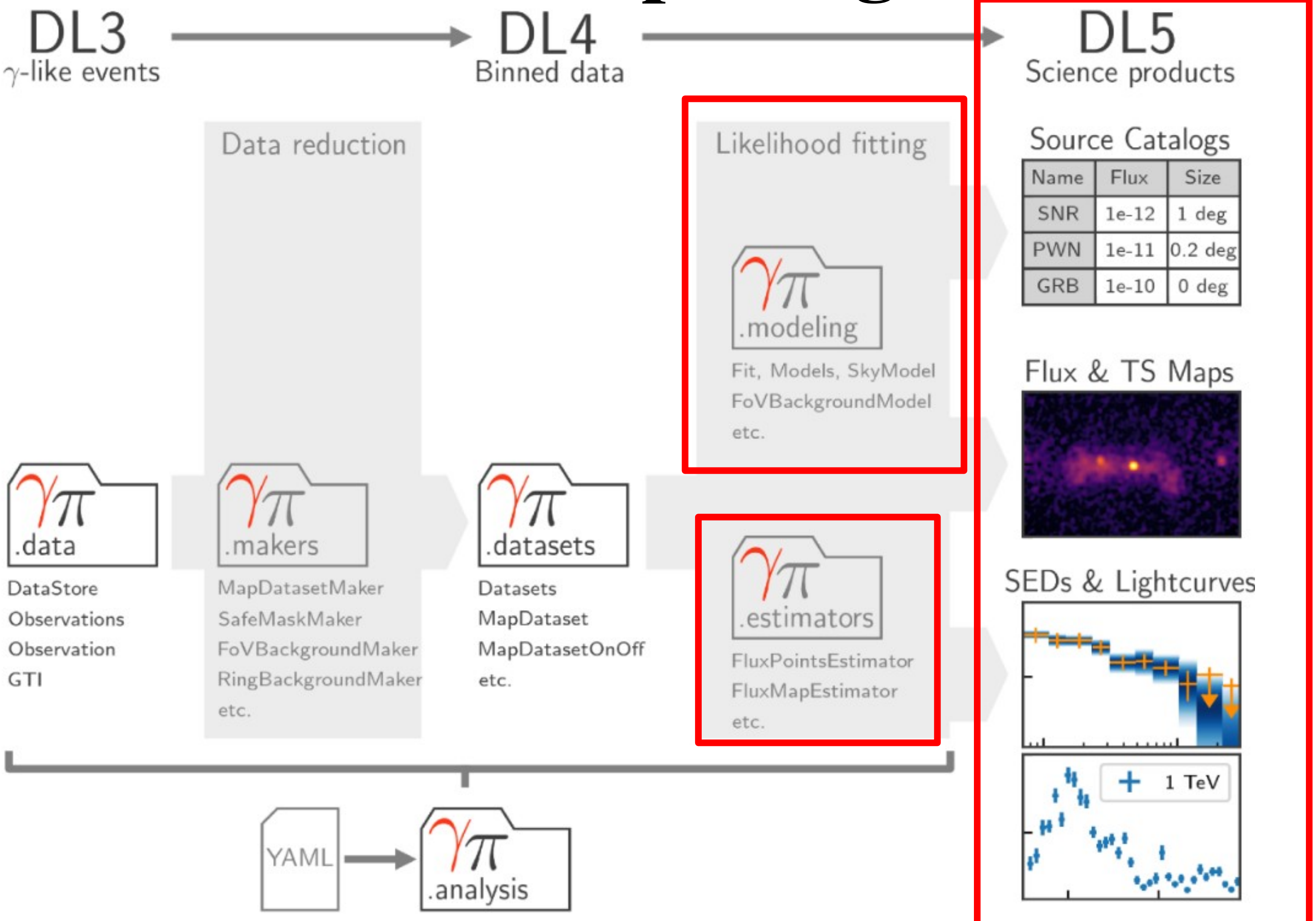
Reduced IRFs

SpectrumDataset

Bkg data or model

Data workflow and package structure

CTA
HESS
MAGIC
VERITAS
Fermi-LAT
HAWC



DL3
γ-like events

DL4
Binned data

DL5
Science products

Data reduction

Likelihood fitting

.data
DataStore
Observations
Observation
GTI

.makers
MapDatasetMaker
SafeMaskMaker
FoVBackgroundMaker
RingBackgroundMaker
etc.

.datasets
Datasets
MapDataset
MapDatasetOnOff
etc.

.modeling
Fit, Models, SkyModel
FoVBackgroundModel
etc.

.estimators
FluxPointsEstimator
FluxMapEstimator
etc.

YAML

.analysis

Source Catalogs

Name	Flux	Size
SNR	1e-12	1 deg
PWN	1e-11	0.2 deg
GRB	1e-10	0 deg

Flux & TS Maps

SEDs & Lightcurves

Modelling & Fitting (DL4 → DL5)

A Python package for gamma-ray astronomy

For modeling and fitting, Gammapy relies on forward-folding:

Measured counts *N* is compared to predicted counts *N*$_{pred}$

$$N_{\mathrm{pred}}(p,E) = \sum_{S} E_{disp}\left[PSF \star \left(expo \times \Phi_S(p_t, E_t)\right)\right] + N_{\mathrm{bkg}}(p,E)$$

Model parameter estimation is performed through maximum likelihood technique.

- **Cash statistics is used for counts data with a known background**

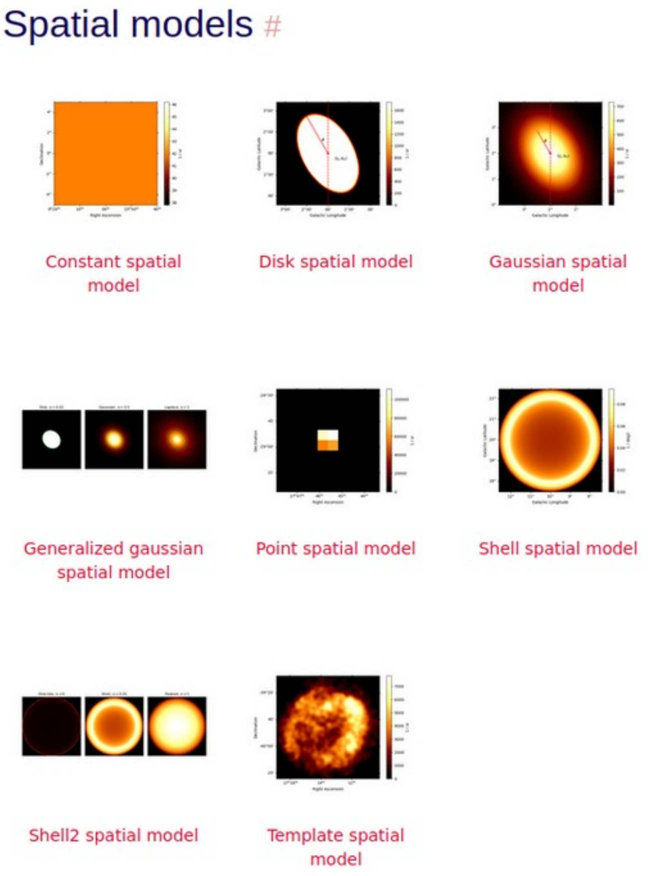
$$TS = -2\log L = 2\sum\left(N\log N_{\mathrm{pred}} - N_{\mathrm{pred}}\right)$$

- **Wstat statistics is used for counts data with a measured background**

cta cherenkov telescope array

Datasets modelling & Fitting

A Python package for gamma-ray astronomy



Spatial models

Constant spatial model

Disk spatial model

Gaussian spatial model

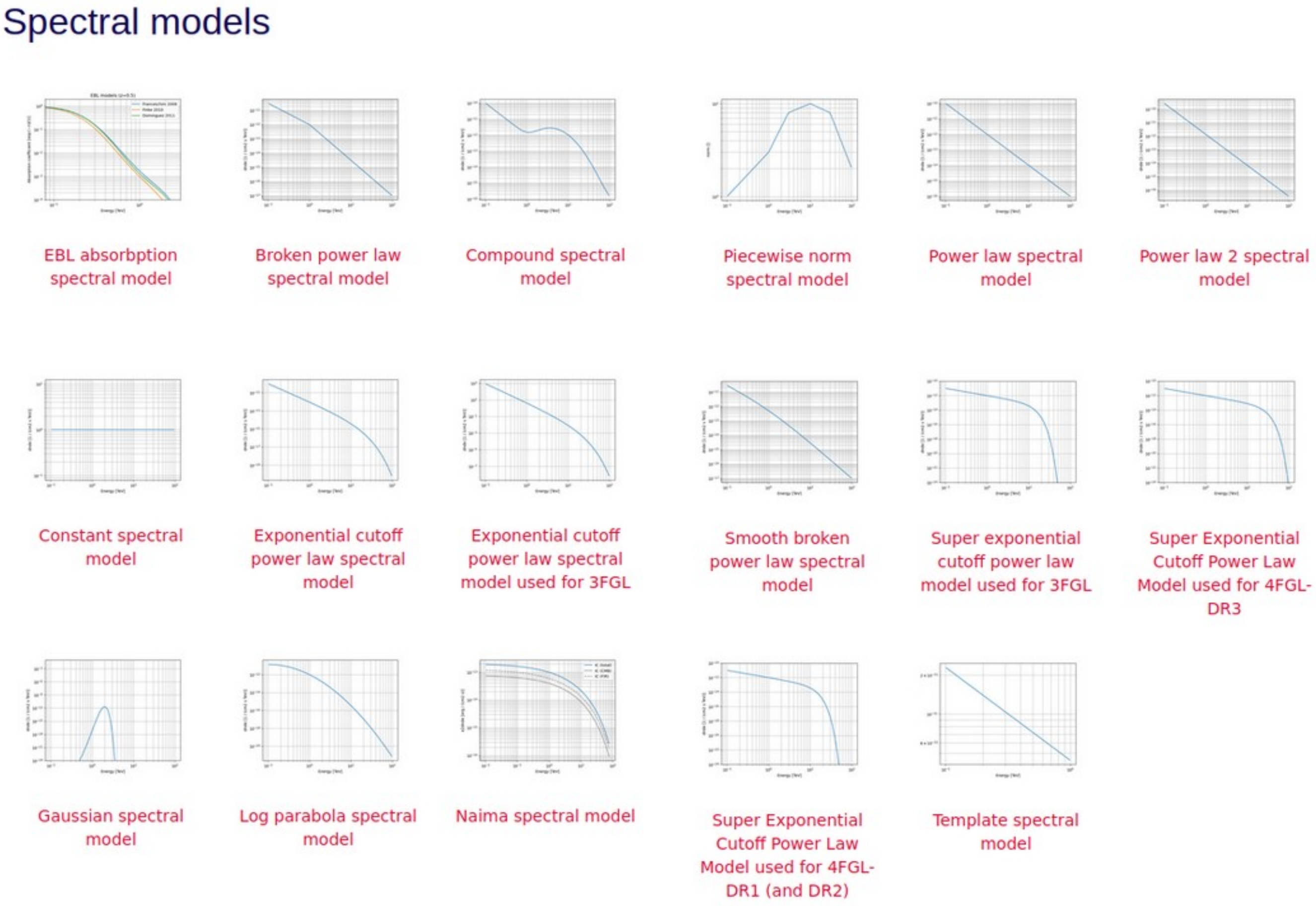
Generalized gaussian spatial model

Point spatial model

Shell spatial model

Shell2 spatial model

Template spatial model



Spectral models

EBL absorbtion spectral model

Broken power law spectral model

Compound spectral model

Piecewise norm spectral model

Power law spectral model

Power law 2 spectral model

Constant spectral model

Exponential cutoff power law spectral model

Exponential cutoff power law spectral model used for 3FGL

Smooth broken power law spectral model

Super exponential cutoff power law model used for 3FGL

Super Exponential Cutoff Power Law Model used for 4FGL-DR3

Gaussian spectral model

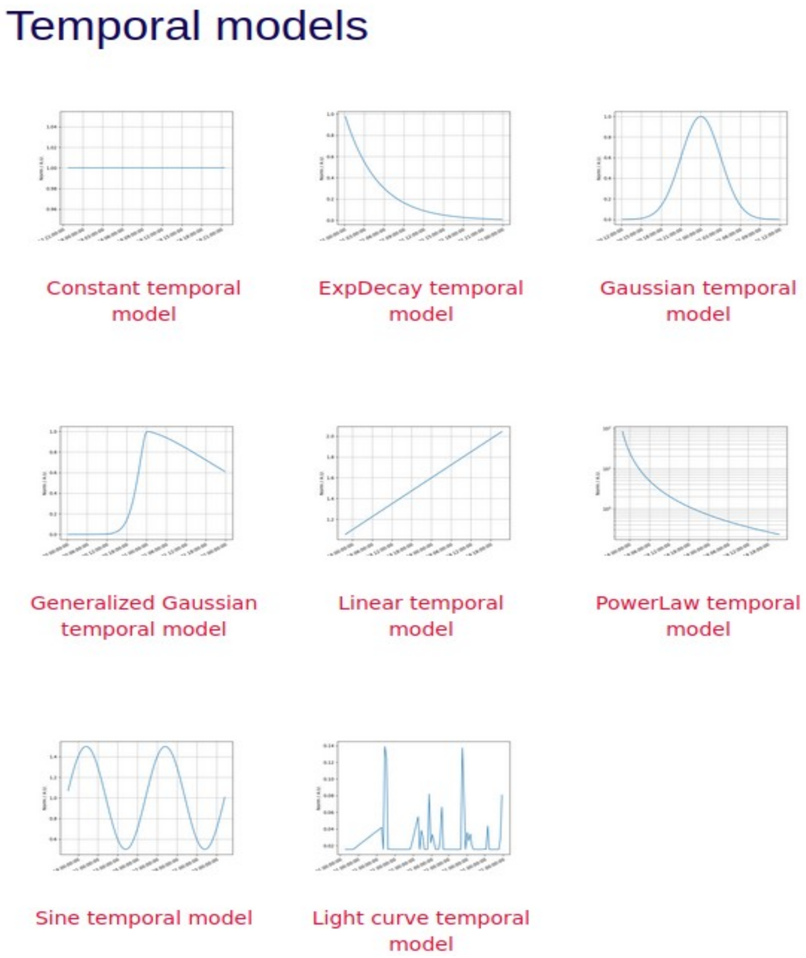
Log parabola spectral model

Naima spectral model

Super Exponential Cutoff Power Law Model used for 4FGL-DR1 (and DR2)

Template spectral model

Datasets modelling & Fitting



Temporal models

Constant temporal model

ExpDecay temporal model

Gaussian temporal model

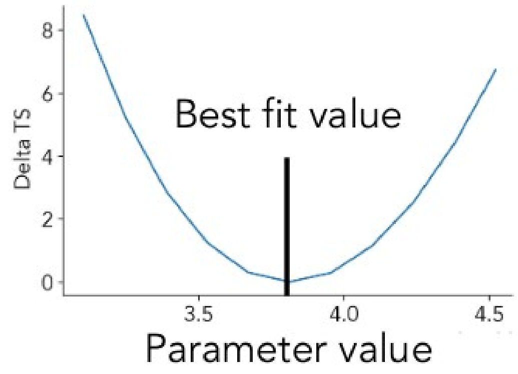
Generalized Gaussian temporal model

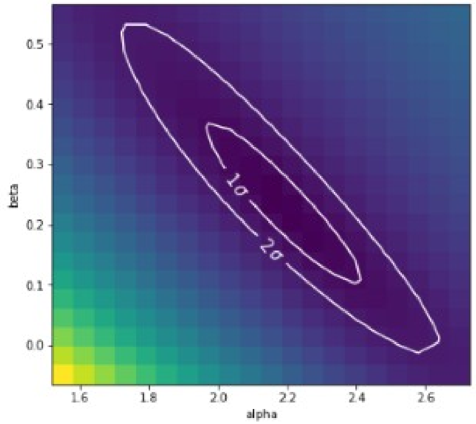
Linear temporal model

PowerLaw temporal model

Sine temporal model

Light curve temporal model

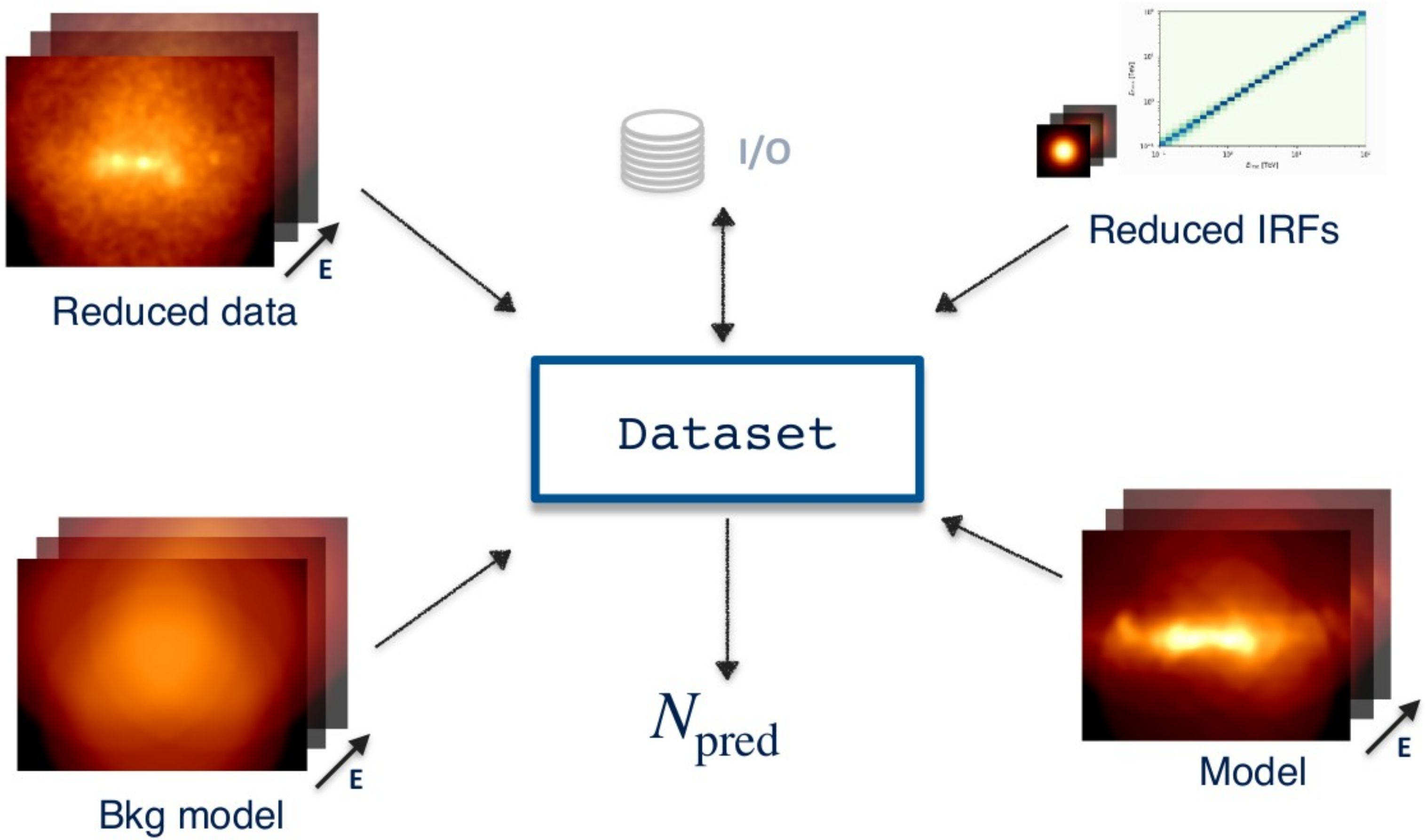




cta cherenkov telescope array

Datasets modelling & Fitting

A Python package for gamma-ray astronomy



Reduced data

I/O

Reduced IRFs

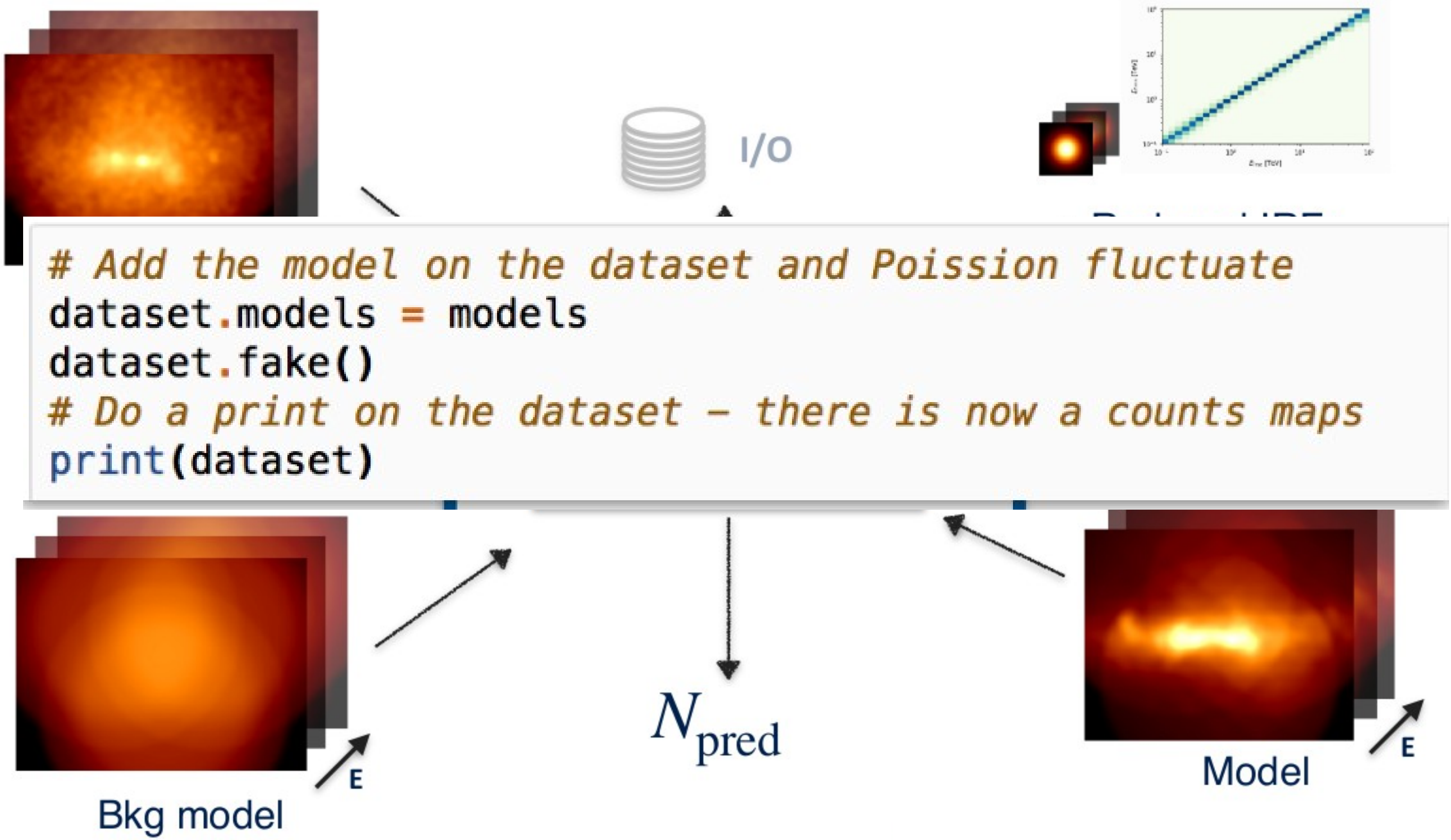
Dataset

N_{pred}

Bkg model

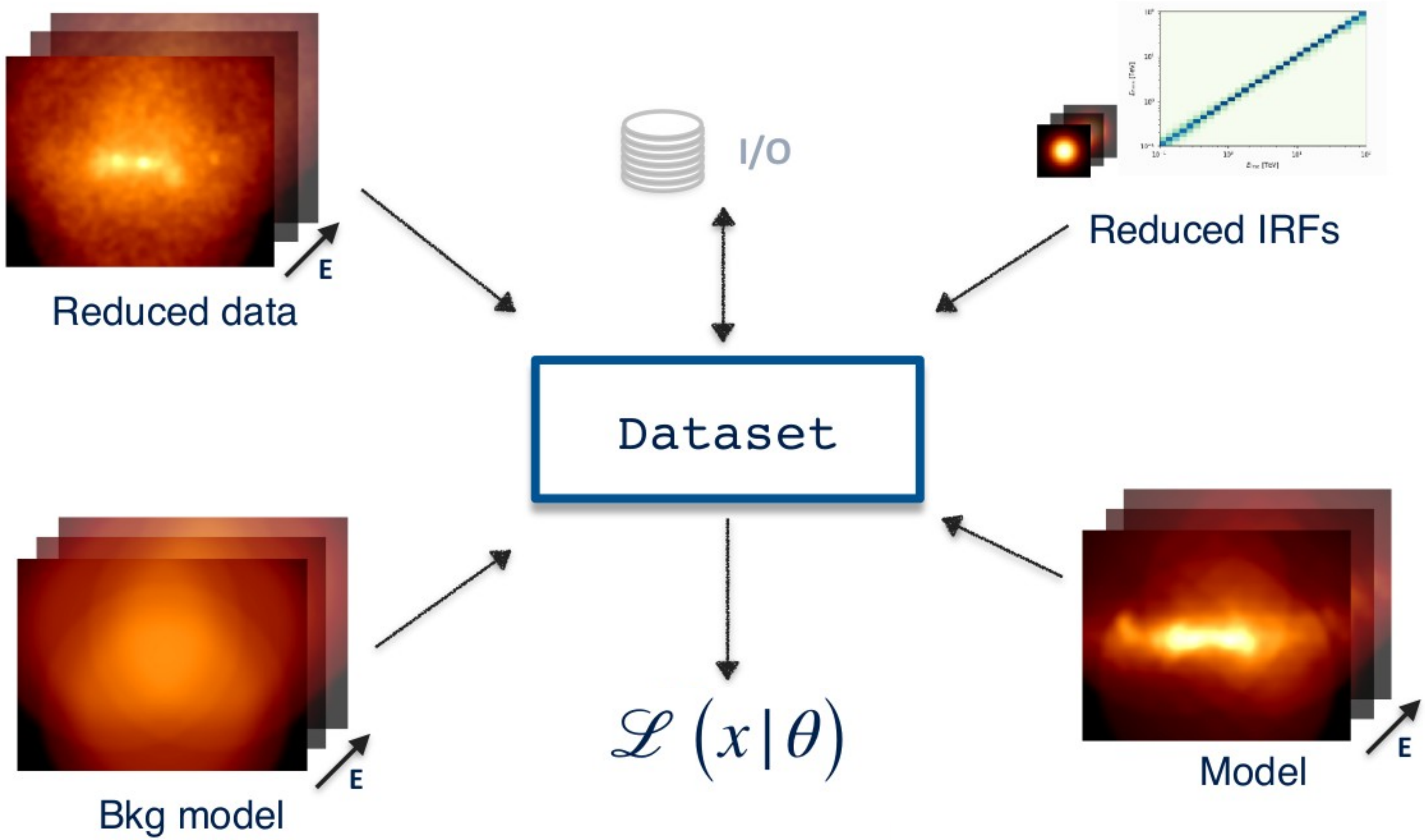
Model

Datasets modelling & Fitting



```
# 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)
```

Datasets modelling & Fitting



Reduced data

I/O

Reduced IRFs

Dataset

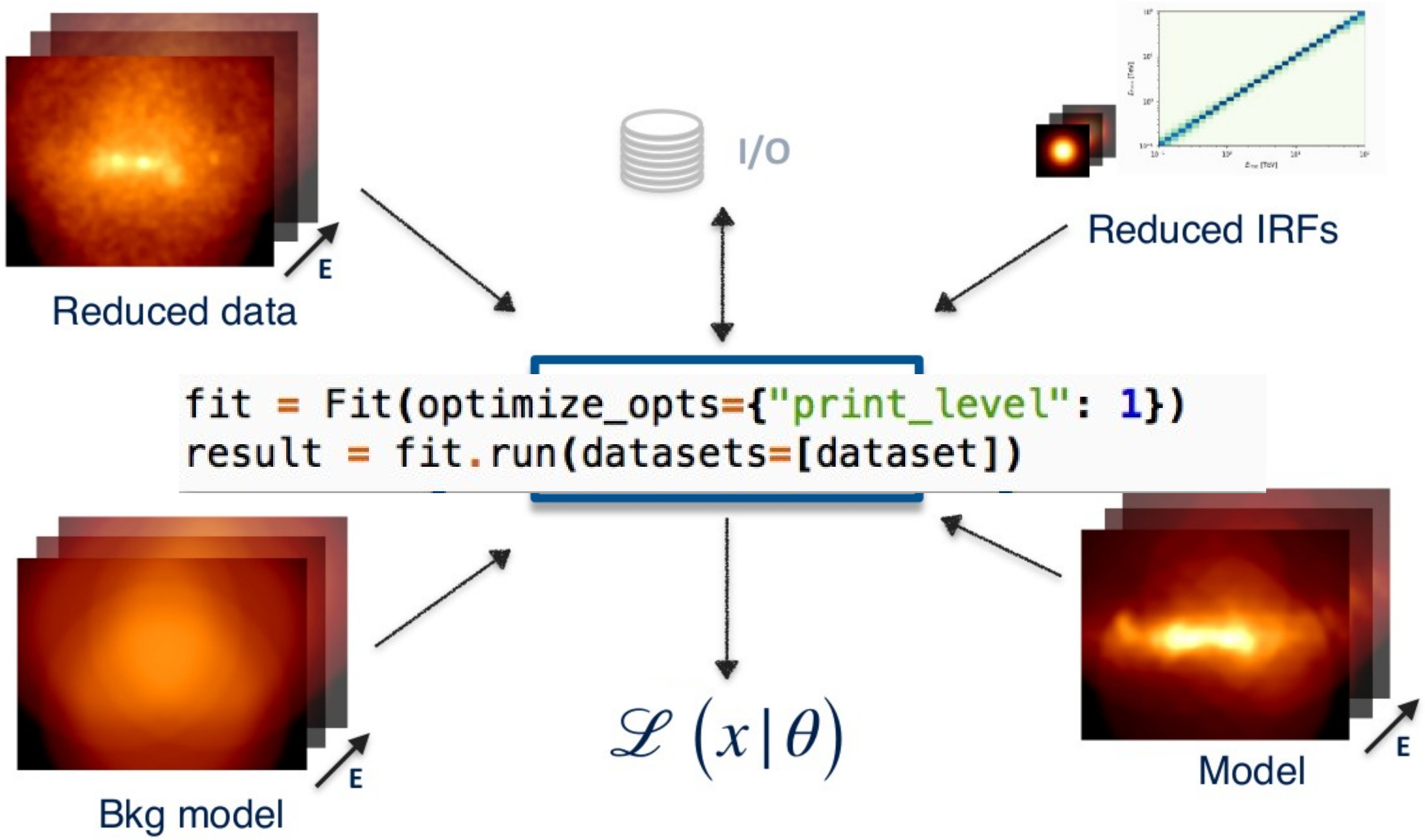
Bkg model

Model

$$\mathscr{L}\left(x|\theta\right)$$

Datasets modelling & Fitting

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

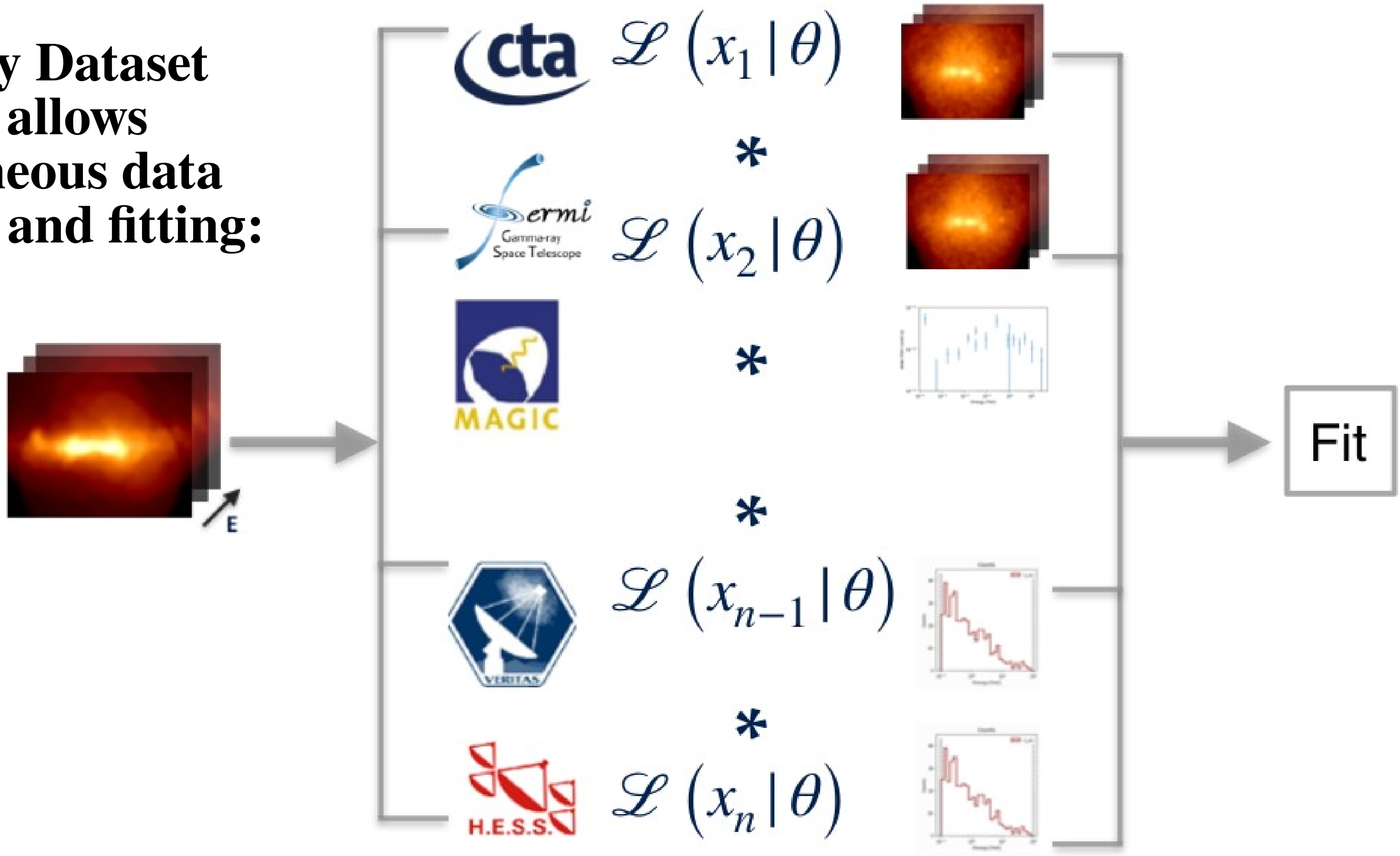


```
fit = Fit(optimize_opts={"print_level": 1})
result = fit.run(datasets=[dataset])
```

Joint fitting (DL4 → DL5)

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

Gammapy Dataset structure allows heterogeneous data modeling and fitting:



$$\mathscr{L}\left(x_1 \mid \theta\right)$$

*

$$\mathscr{L}\left(x_2 \mid \theta\right)$$

*

*

$$\mathscr{L}\left(x_{n-1} \mid \theta\right)$$

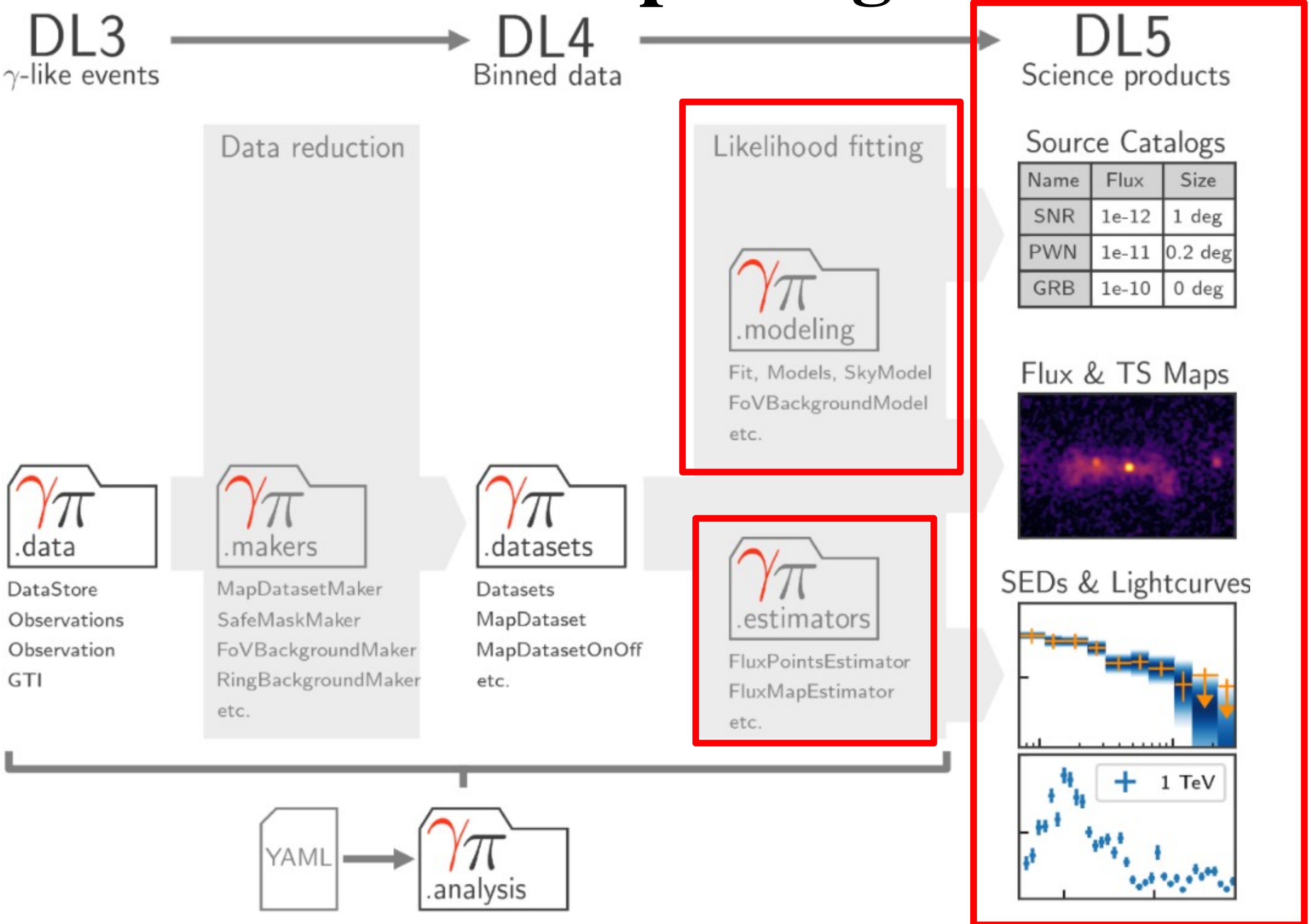
*

$$\mathscr{L}\left(x_n \mid \theta\right)$$

Fit

Data workflow and package structure

CTA
HESS
MAGIC
VERITAS
Fermi-LAT
HAWC



DL3
γ-like events

DL4
Binned data

DL5
Science products

Data reduction

Likelihood fitting

.data

DataStore
Observations
Observation
GTI

.makers

MapDatasetMaker
SafeMaskMaker
FoVBackgroundMaker
RingBackgroundMaker
etc.

.datasets

Datasets
MapDataset
MapDatasetOnOff
etc.

.modeling

Fit, Models, SkyModel
FoVBackgroundModel
etc.

.estimators

FluxPointsEstimator
FluxMapEstimator
etc.

YAML

.analysis

Source Catalogs

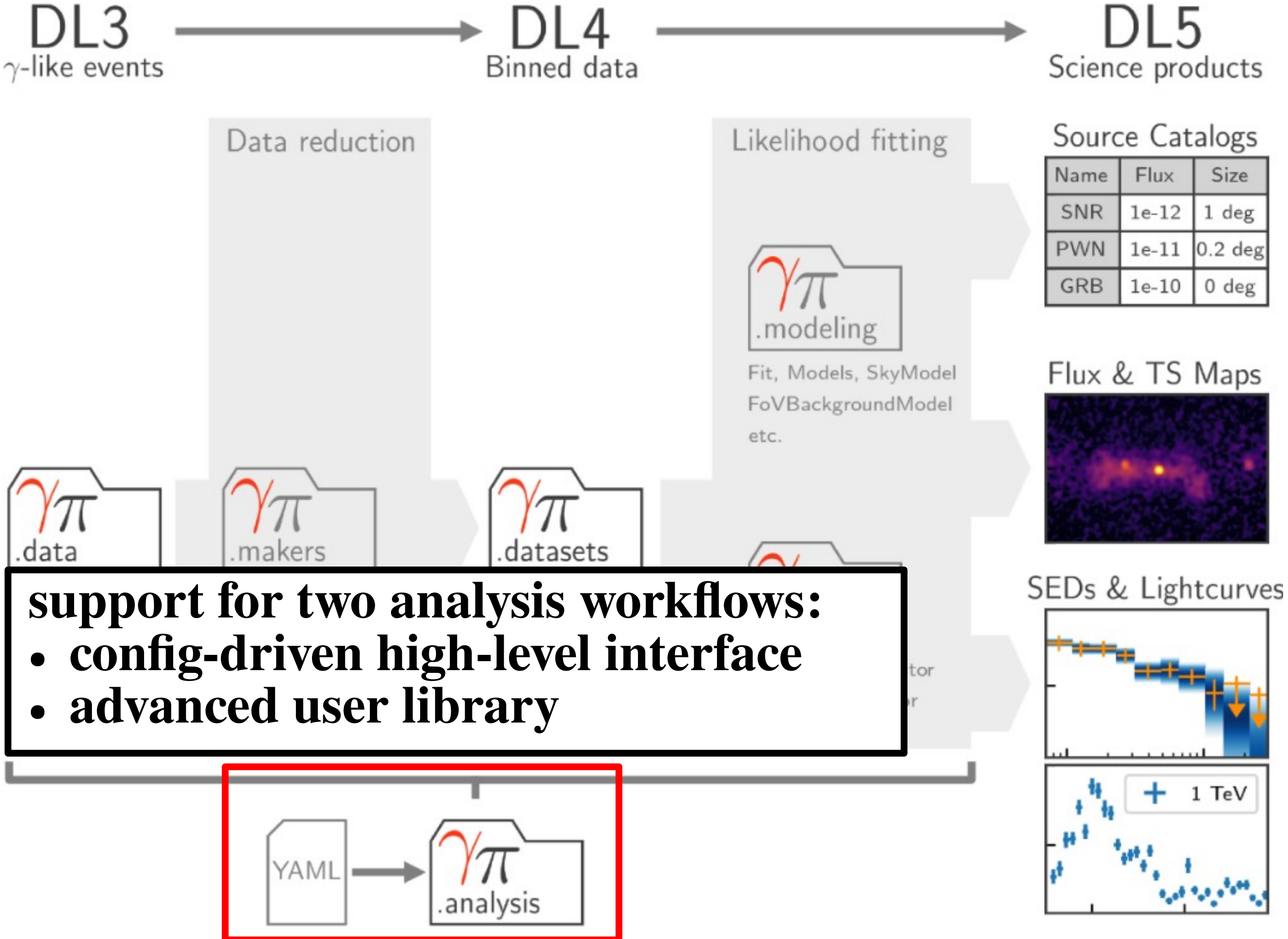
Name	Flux	Size
SNR	1e-12	1 deg
PWN	1e-11	0.2 deg
GRB	1e-10	0 deg

Flux & TS Maps

SEDs & Lightcurves

Data workflow and package structure

CTA
HESS
MAGIC
VERITAS
Fermi-LAT
HAWC

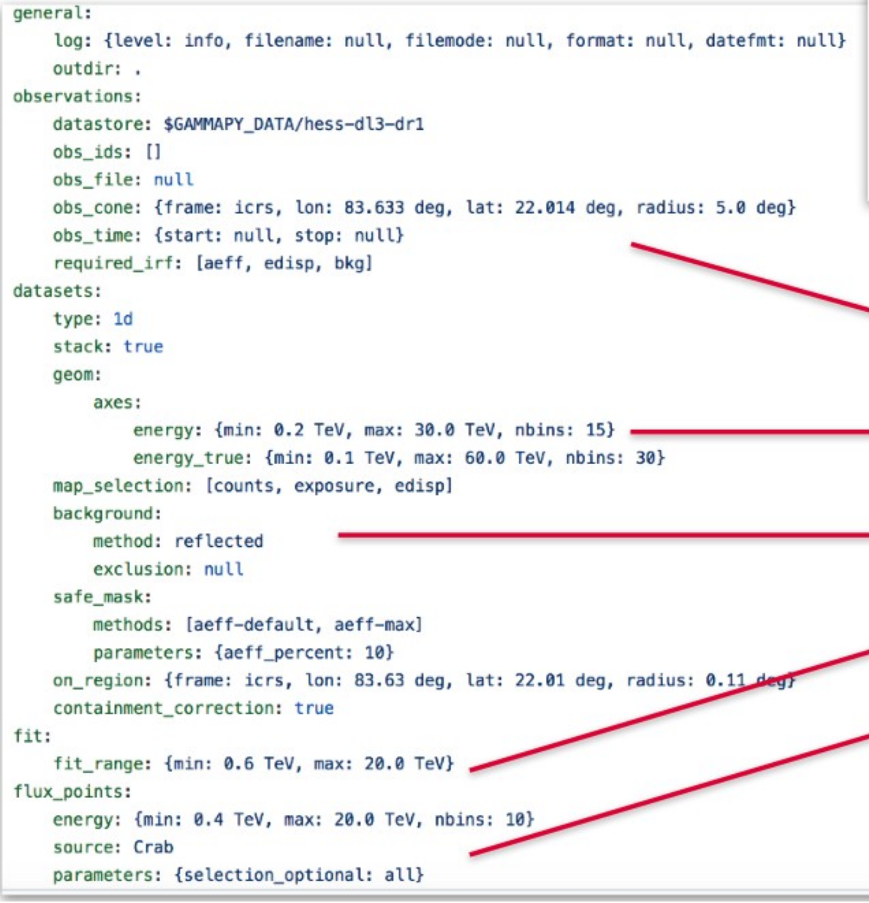


Name	Flux	Size
SNR	1e-12	1 deg
PWN	1e-11	0.2 deg
GRB	1e-10	0 deg

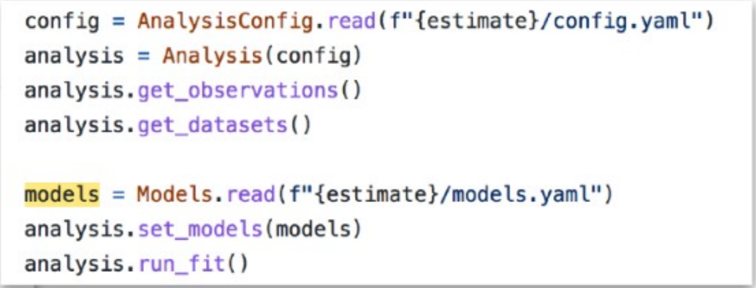
support for two analysis workflows:
- config-driven high-level interface
- advanced user library

Config file drive analisys

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.

What are we going to do these days?

"EVERYTHING"
imgflip.com

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