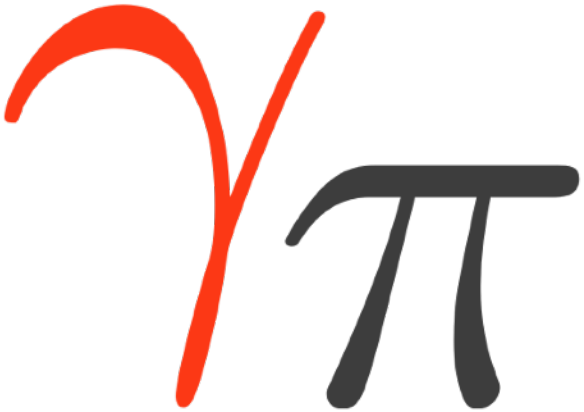


A **Python** package for **gamma-ray** astronomy

Gammapy v0.19

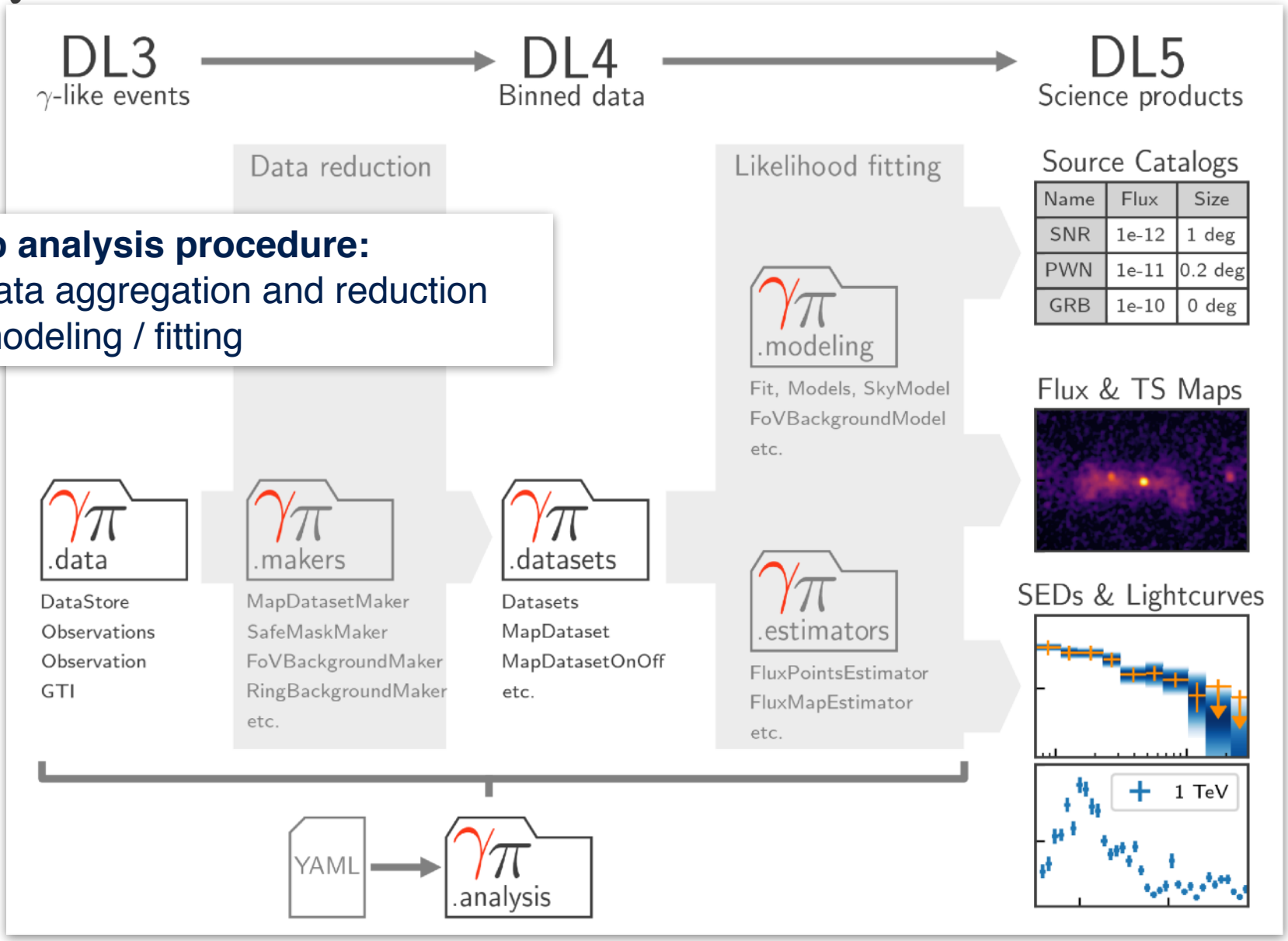
R. Terrier for the dev team

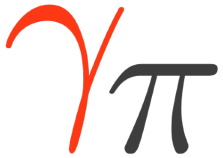
Gammapy user call
Jan *20th, 2022*

Data workflow and package structure

2-step analysis procedure:

- data aggregation and reduction
- modeling / fitting





Installation & setup

- **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
```

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

Installation & setup

- **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**

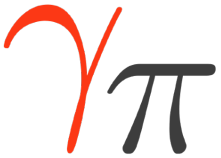
```
gammapy download notebooks --release 0.19
gammapy download datasets
export GAMMAPY_DATA=$PWD/gammapy-datasets
```

Note: mamba might prove a better/faster package manager

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

v0.19 release

- **Released on Nov 22, 2021**
  - 380 merged PRs from 19 contributors
  - See the change-log
- **v0.19 is a preparatory release for v1.0**
  - It contains the proposed user API for v1.0
  - It provides a complete set of functionalities
- We expect very limited backward incompatible API changes before v1.0



Towards v1.0

- **Next release will be the v1.0 release candidate:**
  - Correct remaining issues
  - Documentation polishing still required
  - Timeline: a few months

- We need user feedback to improve incomplete/unclear documentation and correct possible API issues.

- **Should go back to ~6 months release scheme**
  - release branches and bug fixes releases for 1.0
  - new scheme for maintenance and deprecation will be discussed during coding sprint and proposed as a PIG

Towards v1.0

- Next coding sprint (co-working week) will take place next week (Jan 24th - 28th)
  - Main objective is finalization of v1.0
    - Solve open issues (74 remaining)
    - documentation polishing
    - Prepare the release strategy after v1.0

- https://github.com/gammapy/gammapy-meetings/blob/master/coding-sprints/2022-01-Co-Working-Week/README.md

Documentation v0.19

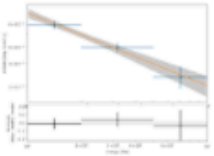
- **Documentation has been largely improved**
- **Gallery of tutorials for better readability**

See docs.gammapy.org

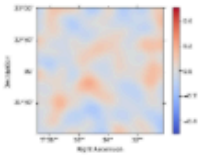
Introduction

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.

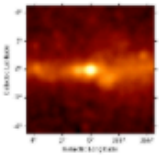
The first tutorial is an overview on how to perform a standard analysis workflow using the high level interface in a configuration-driven approach, whilst the second deals with the same use-case using the low level API and showing what is happening *under-the-hood*. The third tutorial shows a glimpse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



High level interface



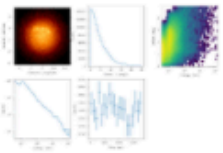
Low level API

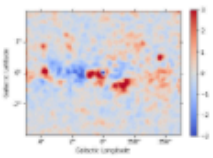


Data structures

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.

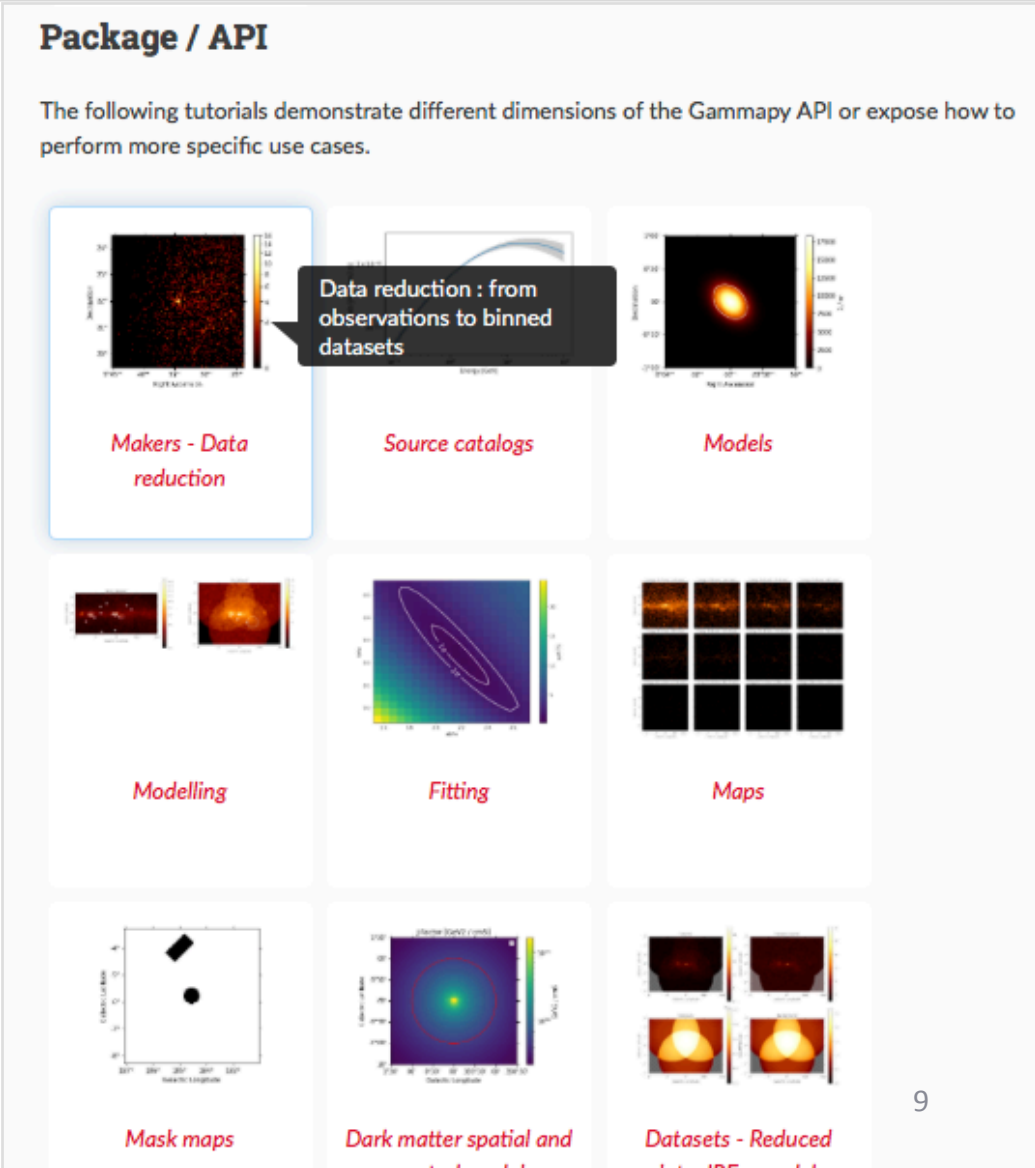




Documentation v0.19

- **Enhanced documentation of base API with dedicated tutorials**

See: [API tutorials]



Documentation : Gammapy recipes

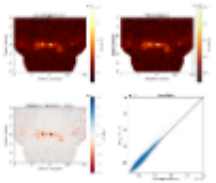
New repository for user contributions

Share tips & tricks, small studies and documented code examples

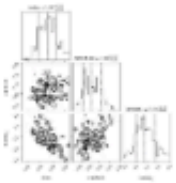
See overview by J.-E. Ruiz

In this webpage you may find a collection of
specific use cases not present in the docume
if you would like to contribute with your own

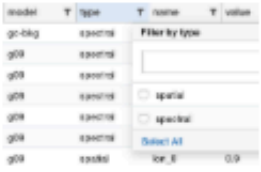
Recipes



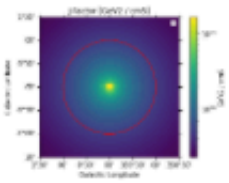
TSMapEstimator vs. ExcessMapEstimator



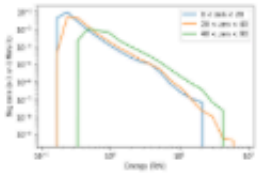
MCMC sampling using the emcee package



Recipe to show the interactively edit the Sky model on the notebook



Dark matter spatial and spectral models



Create a template background model

https://gammapy.github.io/gammapy-recipes
https://github.com/gammapy/gammapy-recipes

Major changes in v0.19

- **Improvements on**
  - `Fit` and `Estimators` API
  - `Models` (region selection, (un)freezing)
  - Cleaned-up `IRF`
  - Better High-level `Analysis` support
  - Model evaluation over regions (extended regions)
  - Integration of extended spatial models
  - Dataset memory handling
  - Now rely on minuit v2.8 and regions v0.5

Major changes in v0.19

- **Improvements on**
  - `Fit` and `Estimators` API
  - `Models` (region selection, (un)freezing)

Main API changes will be presented by A. Sinha

See: Main-API-changes-from-v0.18.2-to-v0.19

  - Integration of extended spatial models
  - Dataset memory handling
  - Now rely on minuit v2.8 and regions v0.5

gammapy.estimators

- **Provide a complete and uniform scheme for high level products from `Estimators`**
  - gadf specifications for DL5 are incomplete
  - Build a uniform and specific data model in gammapy:
    - consistent API for flux points, maps & light curves
    - Internally rely on (N dim) `FluxMap` data structure with possible export to gadf compliant astropy.Table

```
+ table = flux_point.to_table(sed_type="dnde", formatted=True)          # in dnde
+ table = flux_point.to_table(sed_type="flux", formatted=True)            # in flux
```

```
+ table = lc.to_table(format="lightcurve", sed_type="flux")
+ table["time_min", "time_max", "e_min", "e_max", "flux", "flux_err"]
```

gammapy.estimators

- **Provide a complete and uniform scheme for high level products from `Estimators`**
  - gadf specifications for DL5 are incomplete
  - apy:
    ves
    ure
    with possible export to gadf compliant astropy.Table

An overview of the new gammapy.estimators API will be presented by A. Donath

See estimators notebook

```
+ table = flux_point.to_table(sed_type="dnde", formatted=True)          # in dnde
+ table = flux_point.to_table(sed_type="flux", formatted=True)           # in flux
```

```
+ table = lc.to_table(format="lightcurve", sed_type="flux")
+ table["time_min", "time_max", "e_min", "e_max", "flux", "flux_err"]
```

gammapy.estimators

```
lc_maker_1d = LightCurveEstimator(
    energy_edges=[0.4, 0.7, 1.5, 20] * u.TeV,
    source="pks2155",
    time_intervals=time_intervals,
    selection_optional=None,
)
```

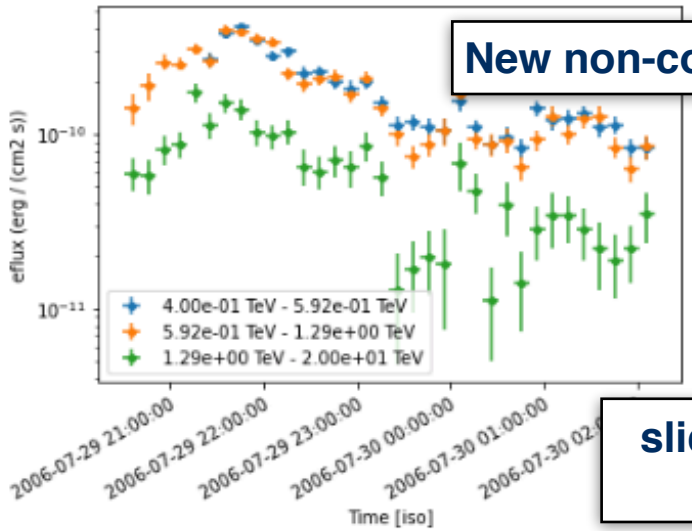
```
%%time
lc_1d = lc_maker_1d.run(datasets)
```

```
CPU times: user 31.2 s, sys: 287 ms, total: 31.5 s
Wall time: 31.9 s
```

Finally we plot the result for the 1D lightcurve:

```
lc_1d.plot(sed_type="eflux", marker="o", axis_name="time")
```

```
<AxesSubplot:xlabel='Time [iso]', ylabel='eflux (erg / (cm2 s))'>
```



Light Curve
RegionMap with 2 non-spatial axes

New non-contiguous time axis

```
lc_1d.eflux.geom.axes["time"]
```

```
TimeMapAxis
-----------

  name           : time
  nbins          : 34
  reference time : 2000-01-01 00:01:04.184
  scale          : tt
  time min.      : 2006-07-29 20:31:05.184
  time max.      : 2006-07-30 02:11:05.184
  total time     : 5.6666666666832235 h
```

```
lc_1d.eflux.geom.axes["energy"]
```

```
MapAxis

      name       : energy
      unit       : 'TeV'
      nbins      : 3
      node type  : edges
      edges min  : 4.0e-01 TeV
      edges max  : 2.0e+01 TeV
      interp     : log
```

slicing along non-spatial axes

```
lc_1d.slice_by_idx({"time":slice(0,15,1)}).dnde
```

```
RegionNDMap

        geom  : RegionGeom
        axes  : ['lon', 'lat', 'energy', 'time']
        shape : (1, 1, 3, 15)
        ndim  : 4
        unit  : 1 / (cm2 s TeV)
        dtype : float64
```

```
lc_1d.dnde
```

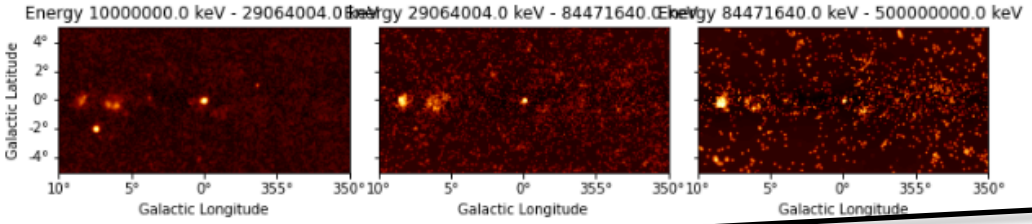
gammapy.estimators

```
%%time
estimator = TSMapEstimator(
    model,
    kernel_width="1 deg",
    energy_edges=[10, 30, 80, 500] * u.GeV,
)
maps = estimator.run(dataset)
```

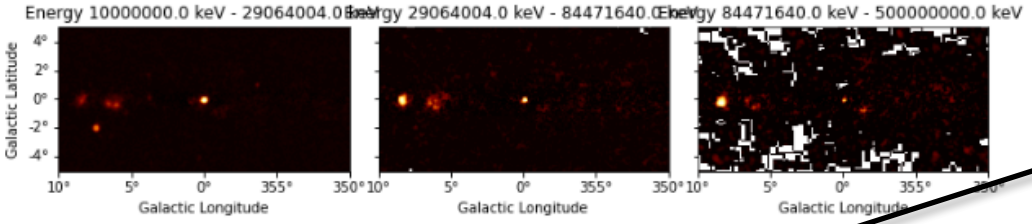
```
CPU times: user 52.8 s, sys: 2.02 s, total: 54.9 s
Wall time: 58.7 s
```

```
maps.write("flux_maps.fits")
```

```
maps.sqrt_ts.plot_grid(ncols=3);
```

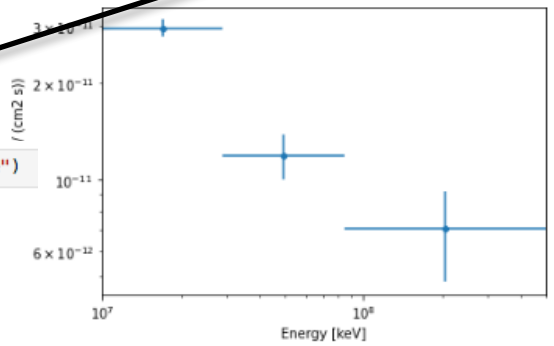



```
maps.flux.plot_grid(ncols=3);
```




```
maps.get_flux_points(SkyCoord(-0.096,-0.05, unit="deg", frame="galactic")).plot(sed_type="e2dnde")
```

```
<AxesSubplot:xlabel='Energy [keV]', ylabel='e2dnde (erg / (cm2 s))'>
```



Flux and TS Map

Direct export of all maps at once

Direct access to flux quantities

Extraction of flux points

gammapy.irf restructuration

```python
from gammapy.irf import IRF, IRF_REGISTRY
from gammapy.irf.io import IRF_DL3_HDU_SPECIFICATION
from gammapy.maps import MapAxis
```

Addition of new IRFs classes simplified

- Generic IRF base class with I/O, interpolation
- IRF registry

```python
class Aeff3D(IRF):
    tag = "aeff_3d"
    required_axes = ["energy_true", "fov_lon", "fov_lat"]
```

```python
IRF_REGISTRY.append(Aeff3D)
IRF_DL3_HDU_SPECIFICATION["aeff_3d"] = {
    "extname": "EFFECTIVE AREA",
    "column_name": "EFFAREA",
    "hduclas2": "EFF_AREA"
}
```

```python
energy = MapAxis.from_energy_bounds(0.01, 100, 5, per_decade=True, name="energy_true", unit="TeV")
fov_lon = MapAxis.from_bounds(-5, 5, 20, unit="deg", name="fov_lon")
fov_lat = MapAxis.from_bounds(-5, 5, 20, unit="deg", name="fov_lat")

meta = {
    'TELESCOP': 'CTA',
    'INSTRUME': 'Southern Array'
}

aeff3D = Aeff3D([energy, fov_lon, fov_lat], data=1, unit="m2", meta=meta)
```

```python
aeff3D.evaluate(energy_true="2 TeV", fov_lon="0.25 deg", fov_lat="8.7 deg")
```

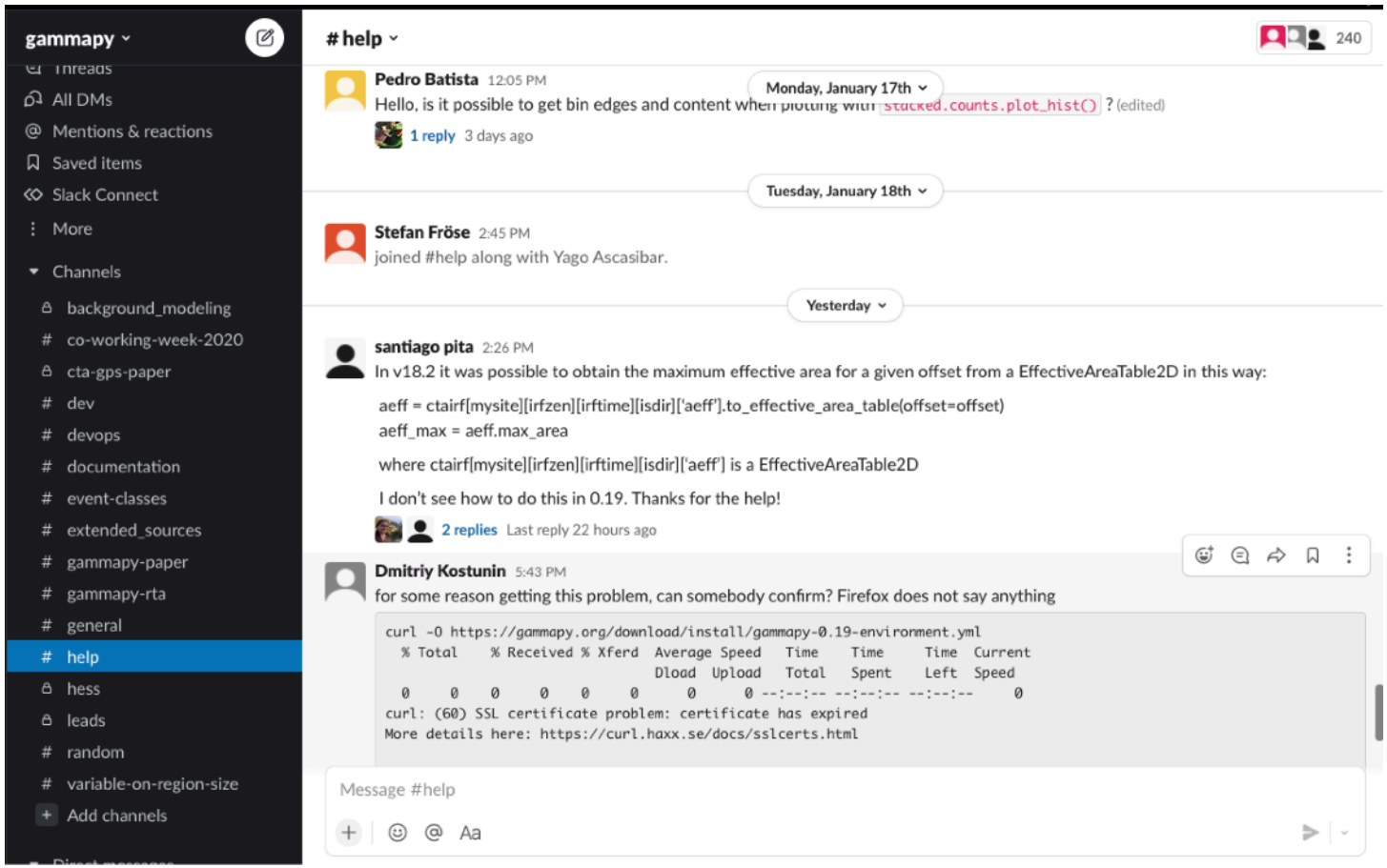
$0\ \mathrm{m^2}$

```python
aeff3D.write("../test_aeff3d.fits", overwrite=True)
```

```python
myi = Aeff3D.read("../test_aeff3d.fits")
```

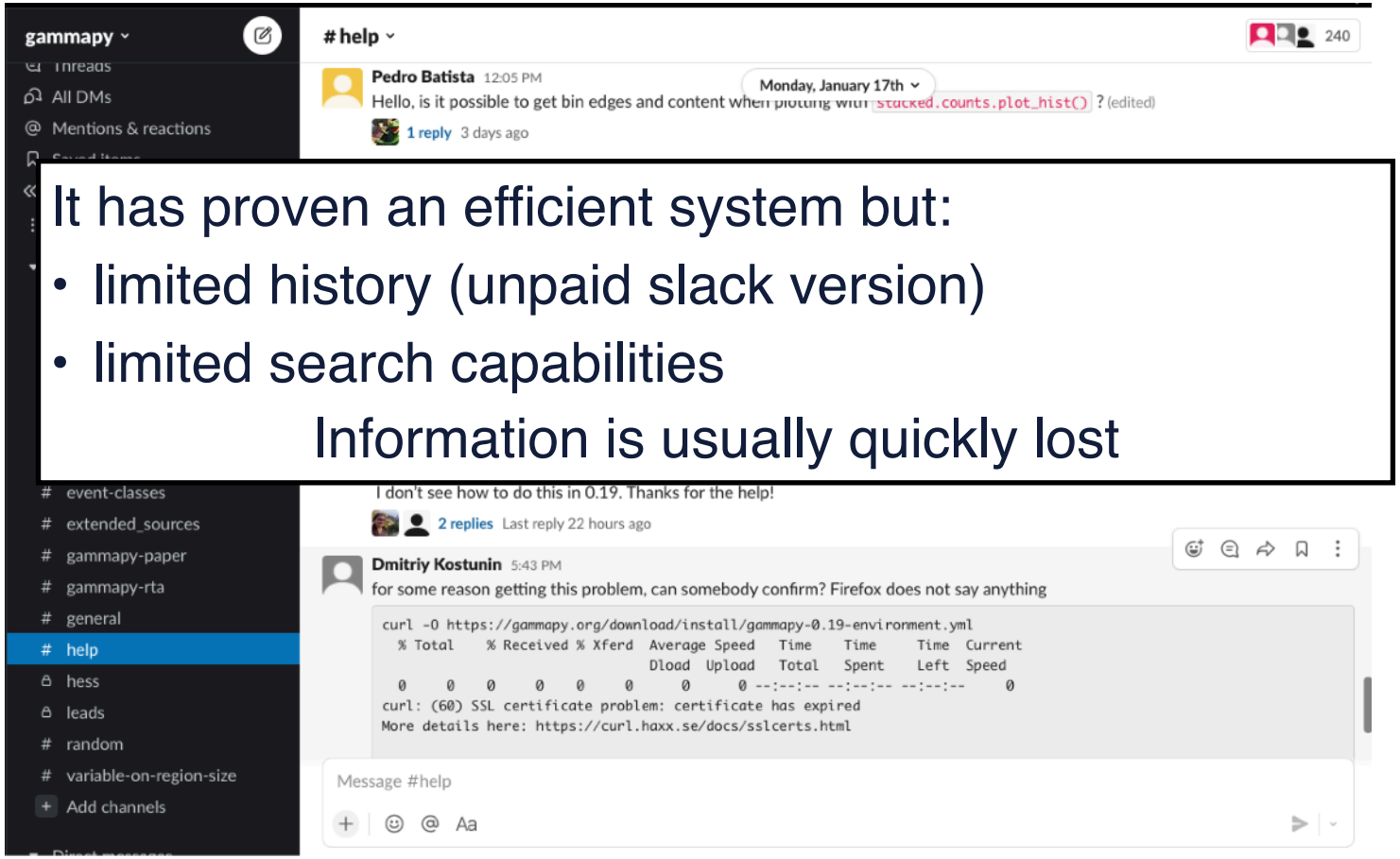
Interaction with community

- Where/How to interact with dev team, provide feedback, get help:
  - gammapy.slack. In particular: #help channel



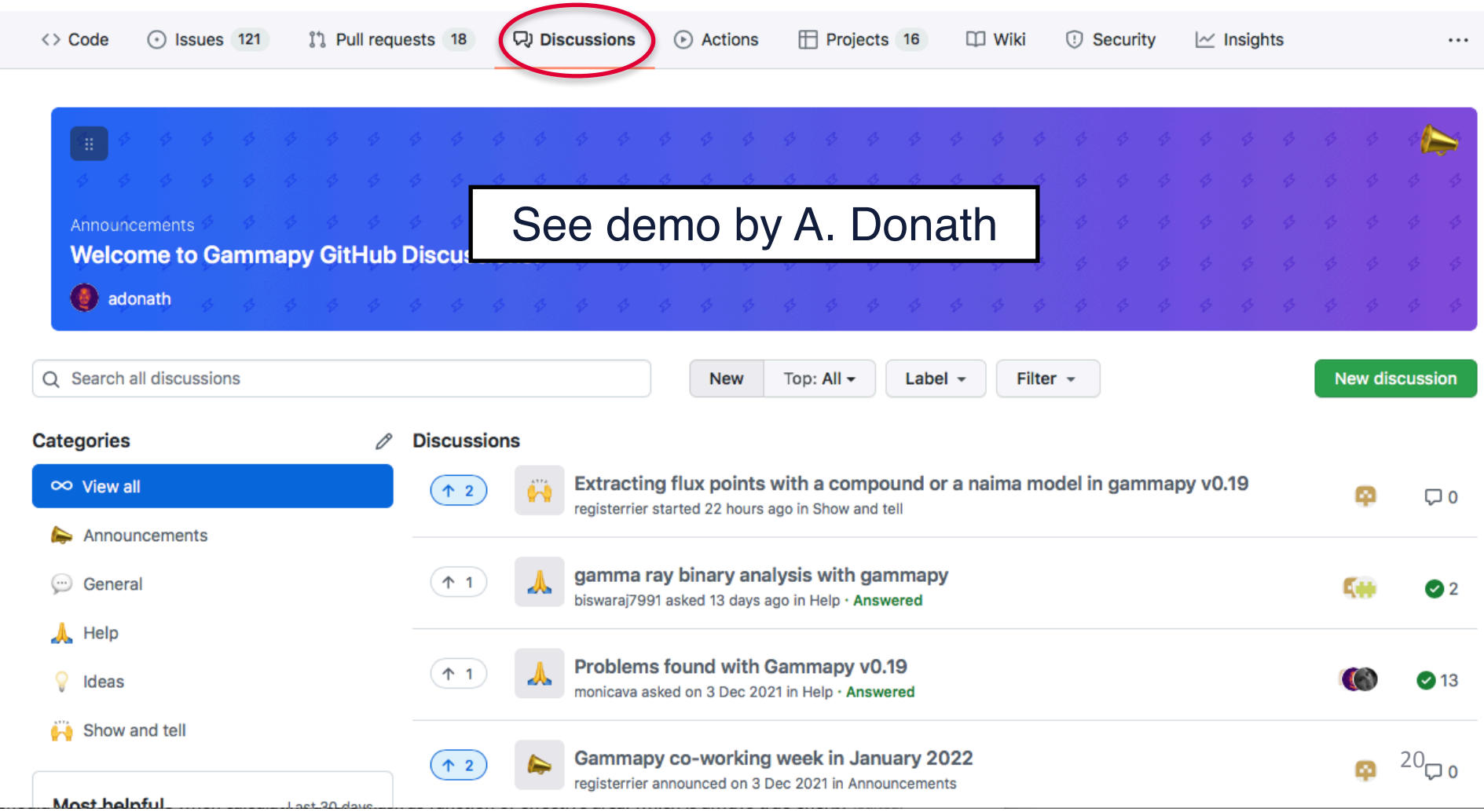
Interaction with community

- Where/How to interact with dev team, provide feedback, get help:
  - gammapy.slack. In particular: #help channel



Interaction with community

Introducing GitHub discussions (searchable, mark answers etc)



Summary

- Gammapy v0.19 contains the proposed API for v1.0
- Documentation, bug-fixes and polishing before 1.0 release
  - Next coding sprint next week
- Test GitHub discussions as a way to interact with users and provide help
- Development >v1.0 will include release branches and bug-fixes releases.