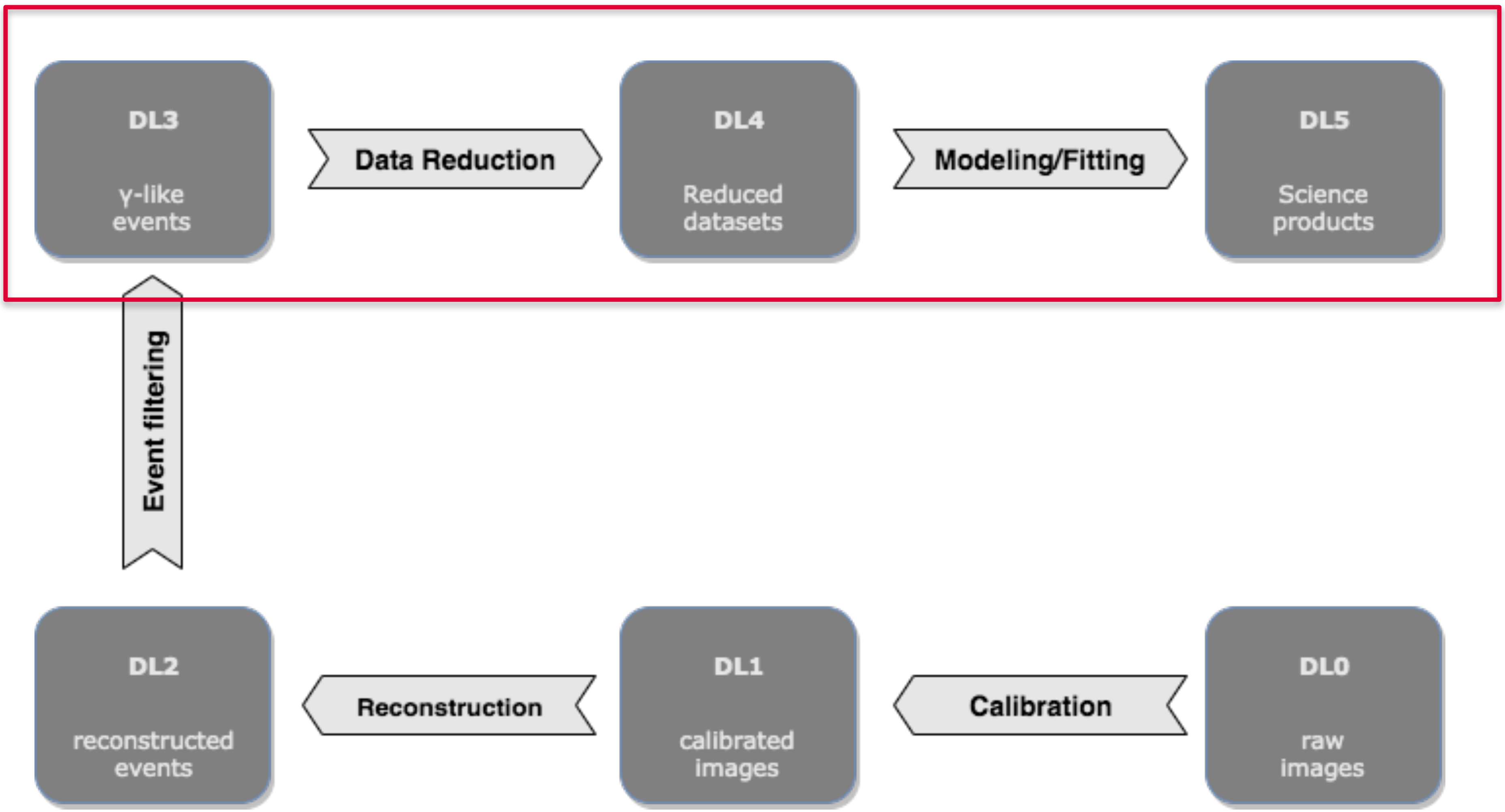
A **Python** package for **gamma-ray** astronomy

Input data formats for gammapy

DL3 and beyond

gammapy user call
October 26, 2020

The data flow concept



Data reduction in gammapy

DL3

γ-like events

Data Reduction

DL4

Reduced datasets

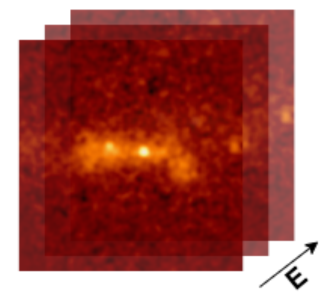
Observation	
	EventList
	EffectiveArea
	EnergyDispersion
	PSF
	GTI

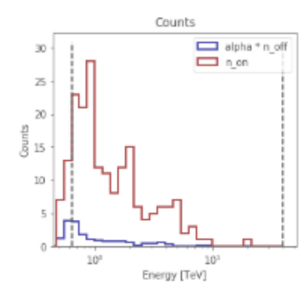
DataStore → ***Observation selection*** → ***Dataset Makers*** → **Datasets**

Observation	
	EventList
	EffectiveArea
	EnergyDispersion
	PSF
	GTI

Filters
- spatial/time selection

Geometry
- Energy bins
- WCS





Data modeling & fitting in gammapy

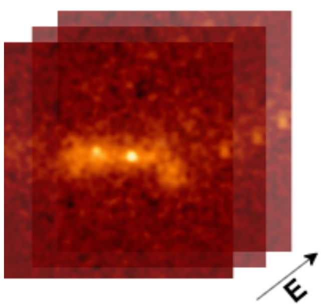
DL4

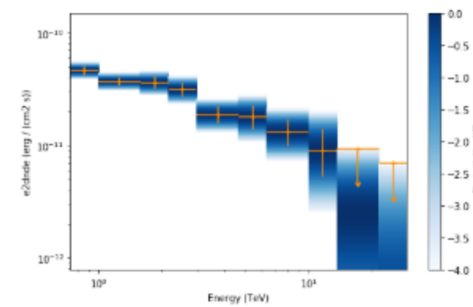
Reduced datasets

Modeling & Fitting

DL5

Science products



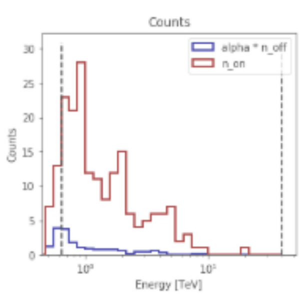


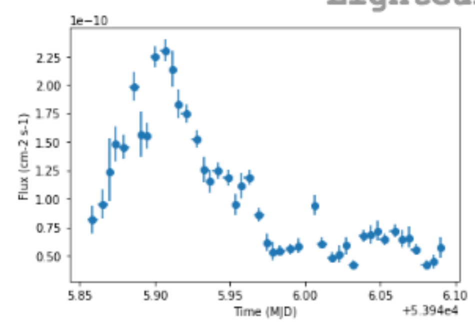
FluxPoints

Datasets

Fit

LightCurve





The DL3 format for IACT data

- For now gammapy follows the definitions provided by the gamma-ray astronomy data format initiative:
  - https://gamma-astro-data-formats.readthedocs.io/en/latest/
  - Started after PyGamma 2015 workshop in MPI-K
  - This will evolve once CTA specifications are defined

- If you want to contribute
  - https://github.com/open-gamma-ray-astro/gamma-astro-data-formats

The DL3 format

- Based on FITS standard
- Follow FITS conventions for time and coordinates
- Information stored in the form binary tables in specific HDUs (Header Data Unit):
  - EVENTS
  - GTI
  - POINTING
  - RESPONSE

- Manipulation in gammapy in CTA and HESS tutorials

EVENTS HDU

- Definition here
- Mandatory columns follow OGIP standard:
  - EVENT_ID, TIME, RA, DEC, ENERGY
- Optional columns
  - EVENT_CLASS, ALT, AZ, etc.
- Mandatory keywords:
  - OBS_ID, ONTIME, DEADC, LIVETIME, ORIGIN, TELESCOP etc.

- Handled in gammapy by the EventList object

GTI HDU

- Definition here
  - Interval of time validity of IRF response associated to events
- Table of START and STOP times in sec
- Reference time defined in header

- Handled by GTI object in gammapy

POINTING HDU

- Preliminary definition [here]
- Table of pointing coordinates with time:
  - TIME, RA_PNT, DEC_PNT
- Keywords: Earth Location, etc.

- Handled in gammapy with `PointingInfo` object

- Not mandatory. For fixed pointing, read the information from the EVENTS header.
  - Handled in gammapy with `FixedPointingInfo` object

RESPONSE: IRFs in DL3 format

- IRFs meant to perform model forward-folding:
  - compute predicted number of counts in detector

$$N(p, E)\mathrm{d}p\mathrm{d}E = t_{\mathrm{obs}} \int_{E_{\mathrm{true}}} \mathrm{d}E_{\mathrm{true}} \int_{p_{\mathrm{true}}} \mathrm{d}p_{\mathrm{true}}\, R(p, E|p_{\mathrm{true}}, E_{\mathrm{true}}) \times \Phi(p_{\mathrm{true}}, E_{\mathrm{true}})$$

- Hypothesis: response can be factored:

$$R(p, E|p_{\mathrm{true}}, E_{\mathrm{true}}) = A_{\mathrm{eff}}(p_{\mathrm{true}}, E_{\mathrm{true}}) \times PSF(p|p_{\mathrm{true}}, E_{\mathrm{true}}) \times E_{\mathrm{disp}}(E|p_{\mathrm{true}}, E_{\mathrm{true}}),$$

- All IRFs are functions of true photon energy (except background)

RESPONSE: IRFs in DL3 format

- 4 main components:
  - AEFF, EDISP, PSF and BACKGROUND
  - Binary tables with energy, FoV coordinate and IRF columns

- 2 main types of IRFs:
  - Full enclosure IRFs
    - For extended sources and 3D analyses
    - For now, a number of
  - Pointlike IRFs
    - obtained after cut in offset w.r.t. expected source position (RAD_MAX) e.g. for events within a ON integration region.

Effective area

- For now, AEFF2D assumes radially symmetric response over the FoV.
- It contains 3 columns:
  - ENERGY_LO, ENERG_HI in TeV
  - THETA_LO, THETA_HI in deg
  - EFFAREA in m^2

- Validity thresholds can be exported to header keywords

- In gammapy, handled with `EffectiveAreaTable2D`

Energy dispersion

- pdf of migration E/E_{true} as a function of true energy and FoV position
- EDISP_2D assumes radially symmetric response over FoV
- It contains 4 columns:
  - ENERGY_LO, ENERG_HI in TeV
  - MIGRA_LO, MIGRA_HI dimensionless
  - THETA_LO, THETA_HI in deg
  - EFFAREA in m^2

- Handling in gammapy with `EnergyDispersion2D`

PSF

- For now, only isotropic PSF with radially symmetric response over the FoV are defined.
- Stored either in the form of a table or of predefined functional forms.
  - PSF_TABLE
  - PSF_3GAUSS
  - PSF_KING

- In gammapy, handling with `PSF3D`, `EnergyDependentMultiGaussPSF`, `PSFKing`
  - Internally rely only on `PSF3D`

Background

- Provides the *differential* background flux brightness as a function of *reconstructed* energy and FoV coordinates.
  - in TeV^{-1} s^{-1} sr^{-1}
  - multiplied by ON time (not dead time corrected)
- Required for most analyses in gammapy
- Defined here:
  - Radially symmetric: BKG_2D
  - As a function of DET_X, DET_Y: BKG_3D
- Handling in gammapy with Background2D, Background3D

Data storage: Index files

- Each EVENTS HDU is connected to the relevants HDUs with an index file
  - HDU index table. Proposed definition here.
  - Provides location of each HDU from base directory

OBS_ID	HDU_TYPE	HDU_CLASS	FILE_DIR	FILE_NAME	HDU_NAME	SIZE
int64	bytes6	bytes9	bytes4	bytes34	bytes6	int64
20136	aeff	aeff_2d	data	hess_dl3_dr1_obs_id_020136.fits.gz	aeff	11520
20136	bkg	bkg_3d	data	hess_dl3_dr1_obs_id_020136.fits.gz	bkg	207360
20136	edisp	edisp_2d	data	hess_dl3_dr1_obs_id_020136.fits.gz	edisp	377280
20136	events	events	data	hess_dl3_dr1_obs_id_020136.fits.gz	events	414720
20136	gti	gti	data	hess_dl3_dr1_obs_id_020136.fits.gz	gti	5760
20136	psf	psf_table	data	hess_dl3_dr1_obs_id_020136.fits.gz	psf	118080
20137	aeff	aeff_2d	data	hess_dl3_dr1_obs_id_020137.fits.gz	aeff	11520
20137	bkg	bkg_3d	data	hess_dl3_dr1_obs_id_020137.fits.gz	bkg	207360
20137	edisp	edisp_2d	data	hess_dl3_dr1_obs_id_020137.fits.gz	edisp	377280
20137	events	events	data	hess_dl3_dr1_obs_id_020137.fits.gz	events	216000

  - Handled with HDUIndexTable object

Data storage: index files

- The observation index provides information of meta data about each observation run: e.g. pointing in the sky, duration, number of events, etc

OBS_ID	RA_PNT	DEC_PNT	GLON_PNT	GLAT_PNT	ZEN_PNT	ALT_PNT	AZ_PNT	OBJECT	RA_OBJ	DEC_OBJ	OFFSET_OBJ
	deg	deg	deg	deg	deg	deg	deg		deg	deg	deg
int64	float32	float32	float32	float32	float32	float32	float32	bytes18	float32	float32	float32
20136	228.6125	-58.771667	320.56754	-0.8857012	38.512962	51.487038	195.73102	MSH15-52	228.6125	-59.271667	0.5
20137	228.6125	-59.771667	320.04724	-1.7397733	40.21616	49.78384	199.6482	MSH15-52	228.6125	-59.271667	0.5
20151	228.6125	-58.771667	320.56754	-0.8857012	37.164658	52.835342	190.97171	custom	228.6125	-59.271667	0.5
20275	187.27792	2.552389	289.7155	64.849686	36.18243	53.81757	49.144917	3C 273	187.27792	2.052389	0.5
20282	228.6125	-58.771667	320.56754	-0.8857012	37.13134	52.86866	169.21602	MSH 15-5-02	228.6125	-59.271667	0.5
20283	228.6125	-59.771667	320.04724	-1.7397733	36.221436	53.778564	175.77263	MSH 15-5-02	228.6125	-59.271667	0.5

- Handled with ObservationTable object

Importing data at DL4

- DL3 format might be relevant in all cases
- Data reduction might be performed by instrument specific software and gammapy can be used for modeling and fitting
- No definition of DL4 format so far.

- gammapy uses the `Dataset` concept for modeling/fitting of reduced data and IRFs
  - Based on the SkyMap data format of gadf (see here)
  - I/O possible with OGIP standard for 1D spectra

Importing data at DL4: `Dataset`

- SpectrumDataset contains 1D structures for:
    - counts
    - background model
    - exposure
    - energy dispersion
- MapDataset contains 3D structures for:
    - counts
    - background model
    - exposure
    - energy dispersion
    - PSF
- ON/OFF versions exist as well.

Importing data at DL4

```python
counts = Map.read(
    "$GAMMAPY_DATA/fermi-3fhl-gc/fermi-3fhl-gc-counts-cube.fits.gz"
)
background = Map.read(
    "$GAMMAPY_DATA/fermi-3fhl-gc/fermi-3fhl-gc-background-cube.fits.gz"
)
background = BackgroundModel(background, datasets_names=["fermi-3fhl-gc"])

exposure = Map.read(
    "$GAMMAPY_DATA/fermi-3fhl-gc/fermi-3fhl-gc-exposure-cube.fits.gz"
)
# unit is not properly stored on the file. We add it manually
exposure.unit = "cm2s"

psf = EnergyDependentTablePSF.read(
    "$GAMMAPY_DATA/fermi-3fhl-gc/fermi-3fhl-gc-psf-cube.fits.gz"
)
psfmap = PSFMap.from_energy_dependent_table_psf(psf)

edisp = EDispKernelMap.from_diagonal_response(
    energy_axis=counts.geom.axes["energy"],
    energy_axis_true=exposure.geom.axes["energy_true"],
)

dataset = MapDataset(
    counts=counts,
    models=[background],
    exposure=exposure,
    psf=psfmap,
    name="fermi-3fhl-gc",
    edisp=edisp,
)
```

Example from source detection tutorial

Conclusions

- DL3 format provides a convenient open description of gamma-like data
  - An evolving format
  - Specifications given in gamma-astro-data-format are supported in gammapy

- Reduced data and IRF can also be imported to perform modeling and fitting with gammapy.
  - see tutorial on joint fitting of HESS, Fermi and HAWC