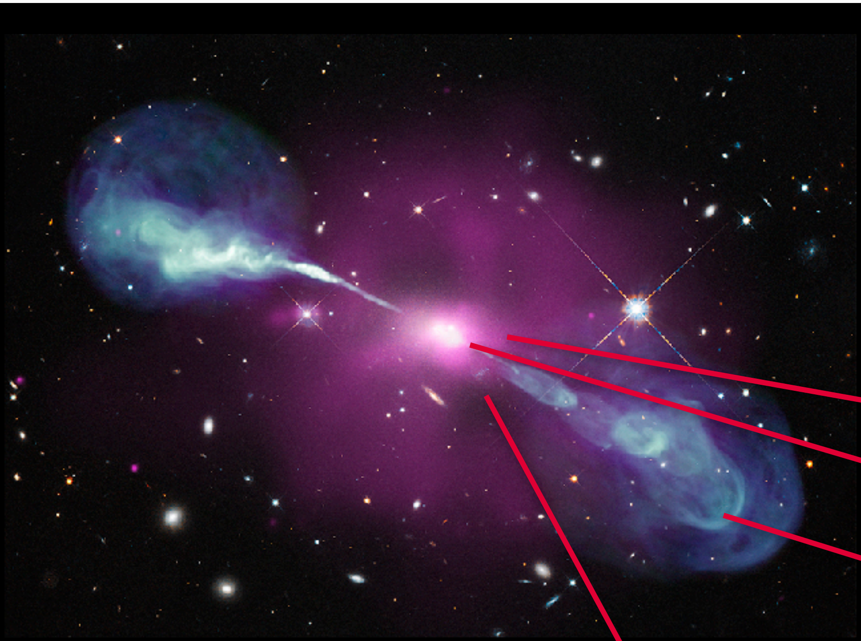
γπ A **Python** package for **gamma-ray** astronomy

Ecole du LabEx UnivEarthS
Opening Science

Dec 15th 2020

Régis Terrier, Laboratoire APC

γ-ray and multi-messenger astronomy



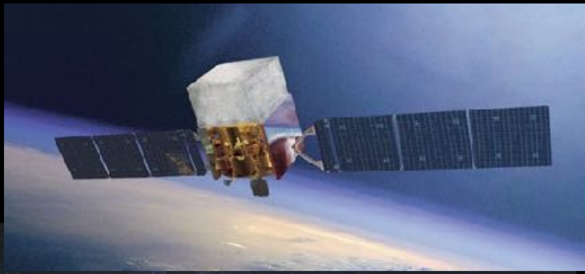
Electromagnetic signal

radio

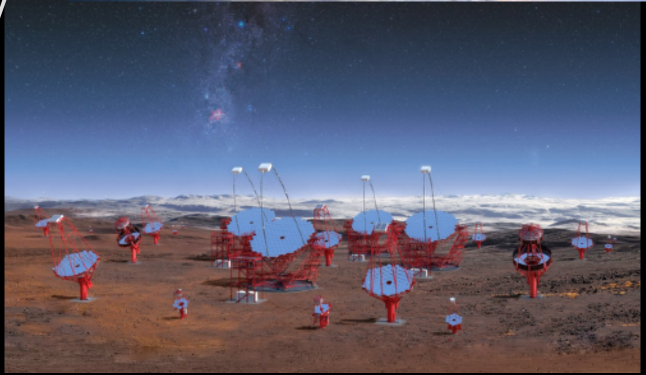
optical/IR
$10^{-2\text{—}0}$ eV

X-rays
$10^{3\text{–}5}$ eV

HE γ-rays
$10^{7\text{–}11}$ eV

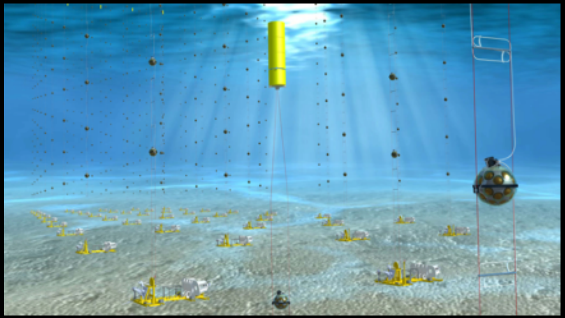


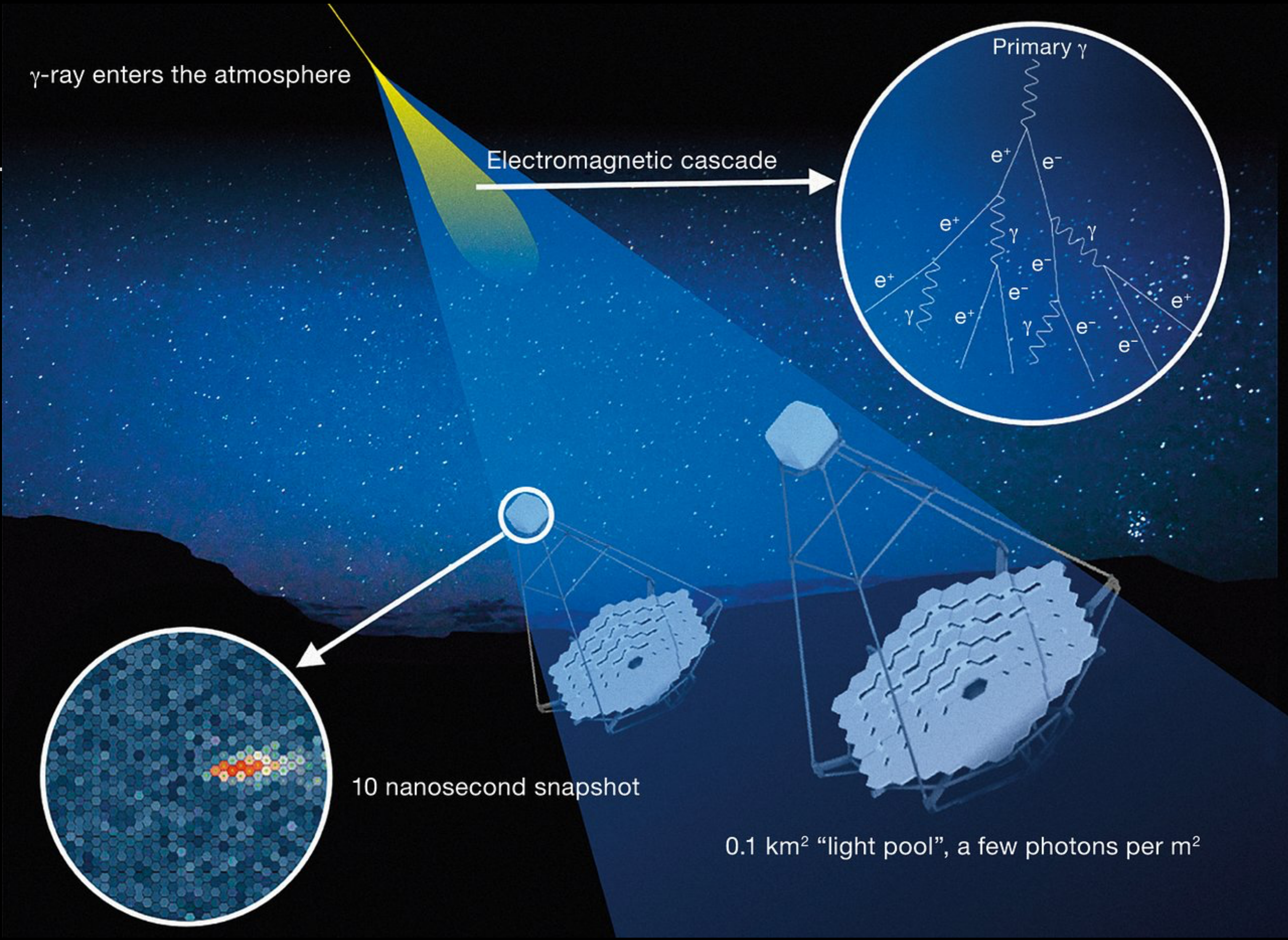
VHE γ-rays
$10^{11\text{–}15}$ eV



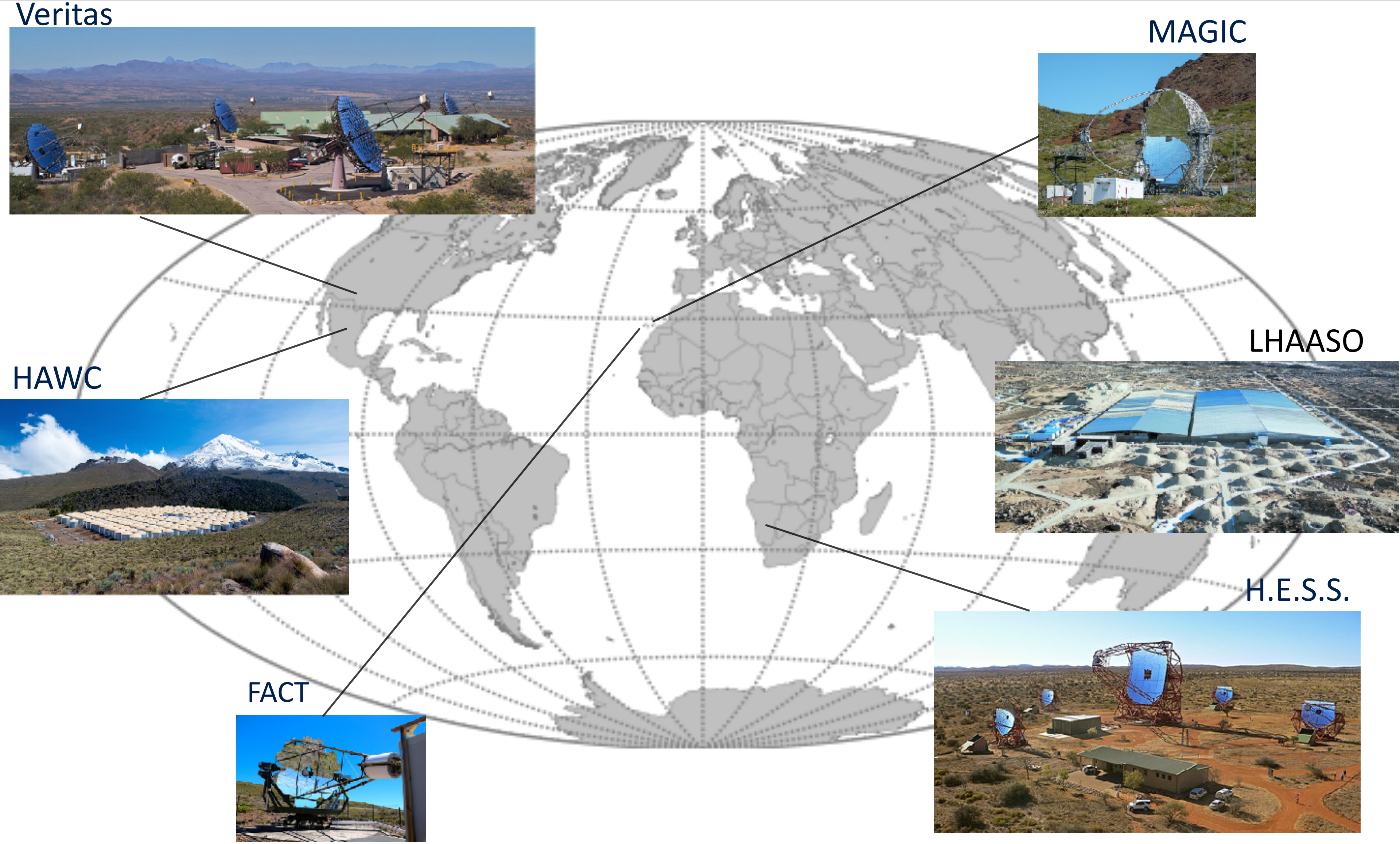
Neutrino signal

VHE neutrinos
$10^{11\text{–}}$ eV




γ-ray enters the atmosphere
Electromagnetic cascade
Primary γ
e^+ e^-
e^+ γ γ
e^+ e^- e^- e^+
γ e^+ γ e^- e^-
10 nanosecond snapshot
0.1 km^2 “light pool”, a few photons per m^2

Ground based γ-ray astronomy today



Veritas

MAGIC

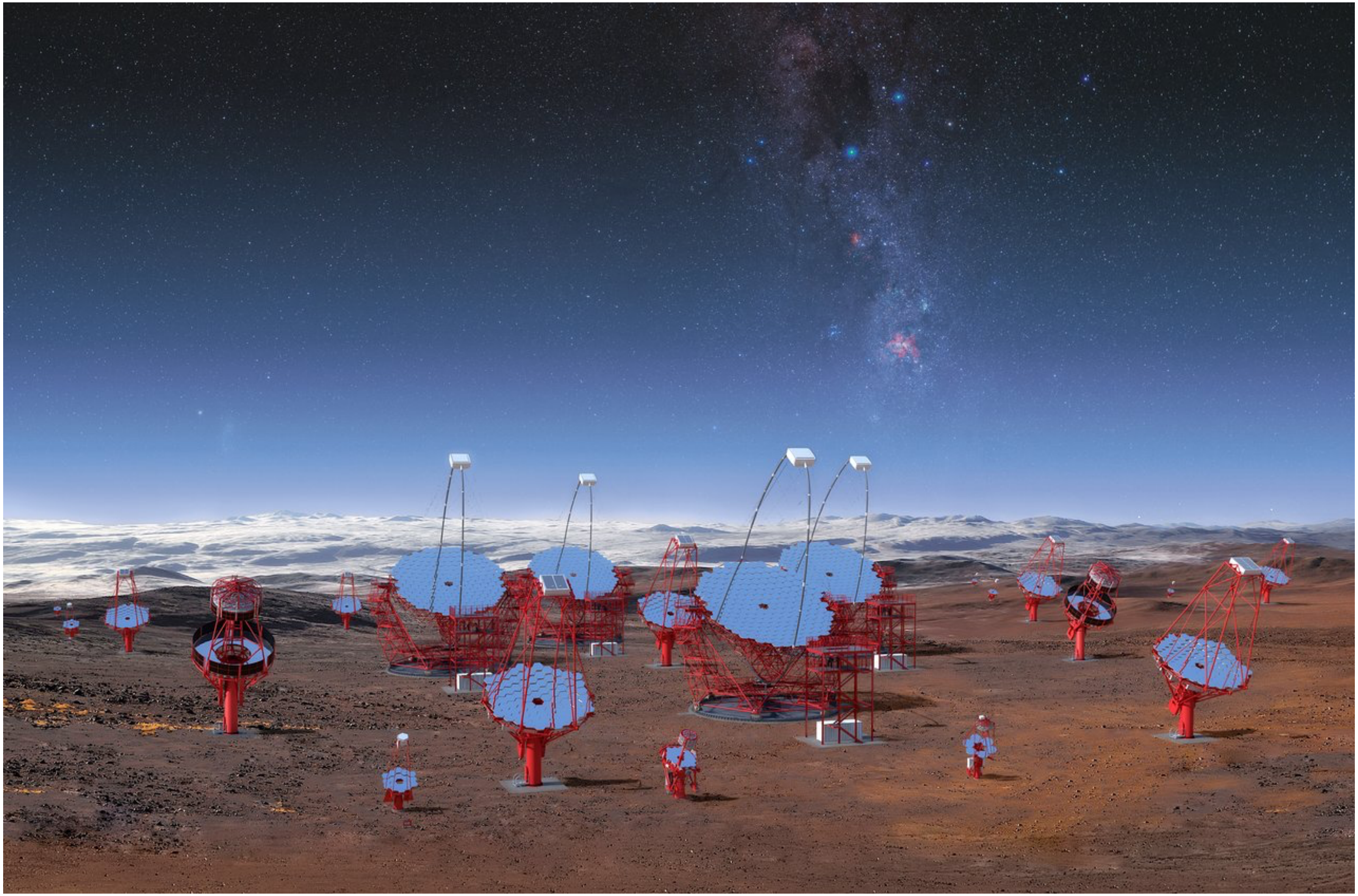
HAWC

LHAASO

H.E.S.S.

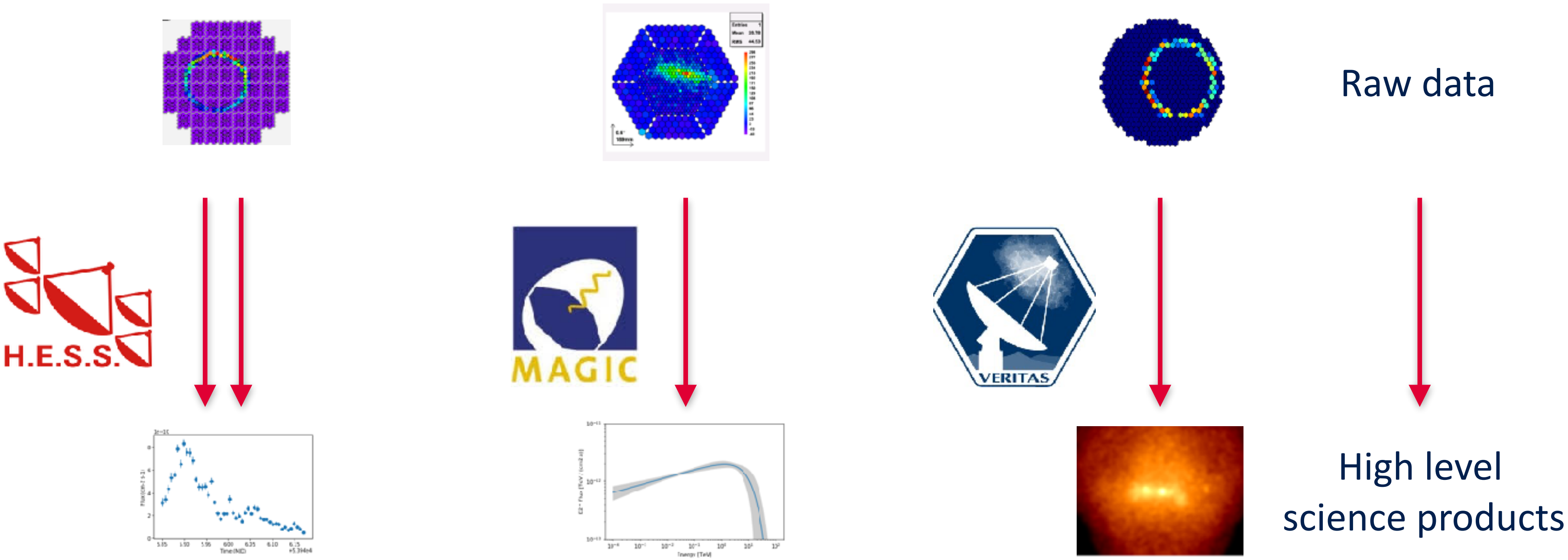
FACT

Ground based γ-ray astronomy: CTA



Proprietary formats and tools

- All instruments have their own proprietary formats and tools
  - usually based on the open software ROOT from CERN
- Common joint analyses are impossible



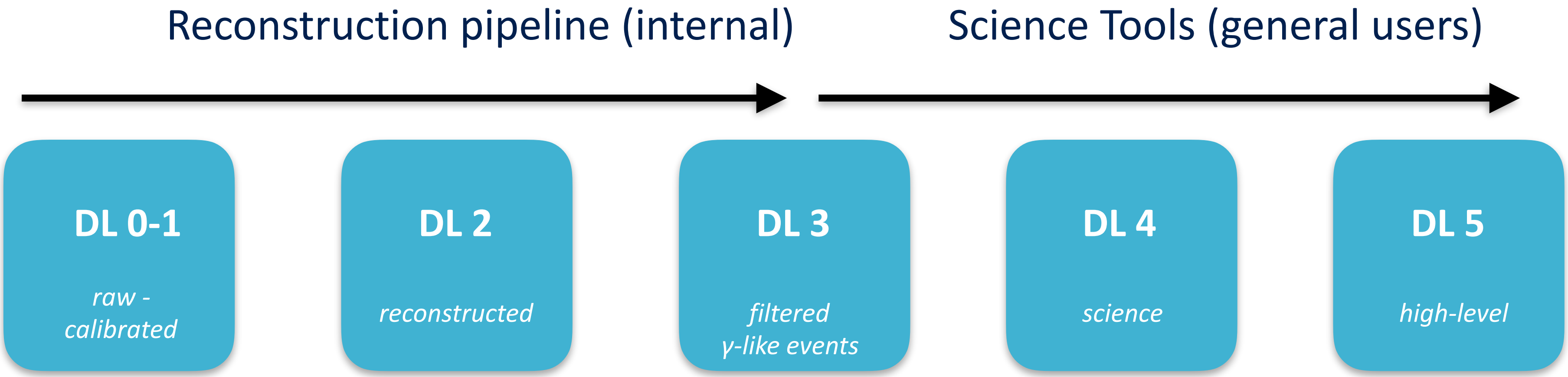
Raw data

High level science products

Interoperability & reusability require
common open *data formats* and common open *tools*

The data flow

Separating instrument specific data treatment from common use cases and methods



Reconstruction pipeline (internal)

Science Tools (general users)

DL 0-1

raw - calibrated

DL 2

reconstructed

DL 3

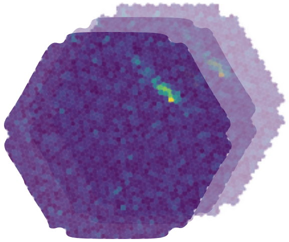
filtered γ-like events

DL 4

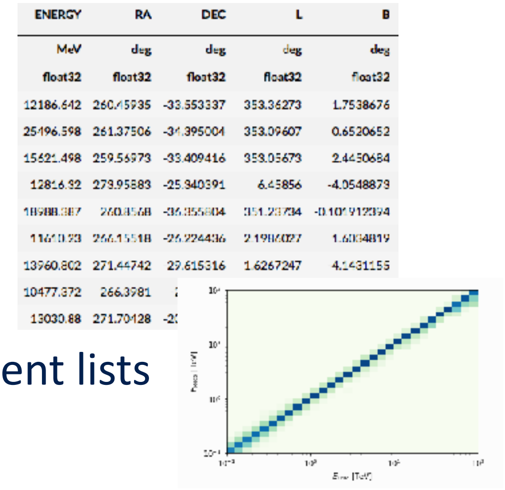
science

DL 5

high-level

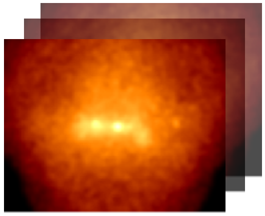


camera data



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
25496.598	261.37506	-34.395004	353.09607	0.6520652
15621.498	259.56973	-33.409416	353.05673	2.4450684
12816.32	273.95883	-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	2		
13030.88	271.70128	-2		

γ-like event lists
IRFs



maps,
spectra,
light curves

The gammapy concept

A python package for γ-ray astronomy based on [common data formats]

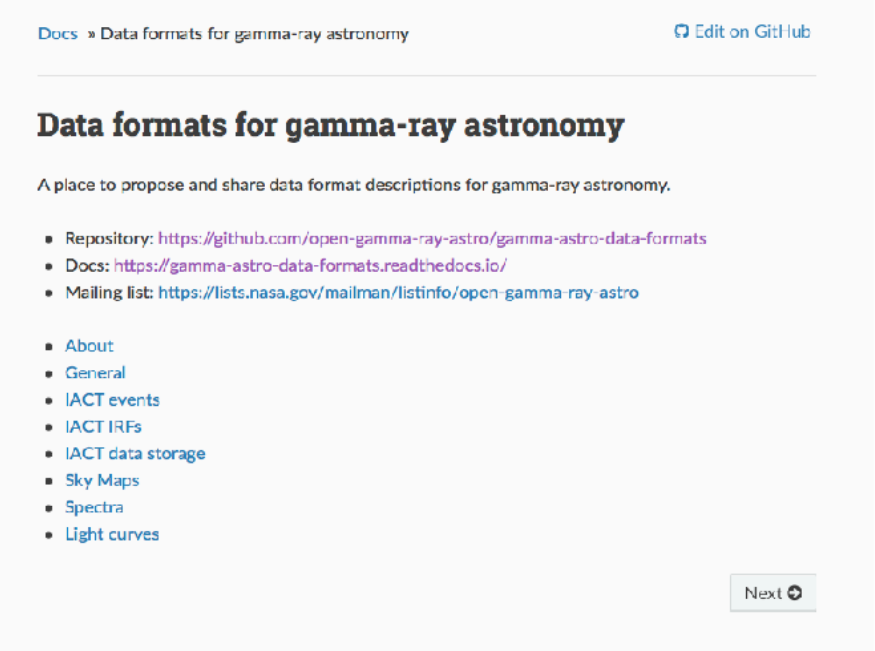
A flexible, open-source, community driven, python library

+

A prototype for CTA science tools

Towards common data formats

- A community initiative to provide open data formats for γ-ray astronomy
  - g.a.d.f. (gamma-ray astro data formats)
  - members of Fermi-LAT and ground based instruments
  - a prototype format already used for data productions based on FITS



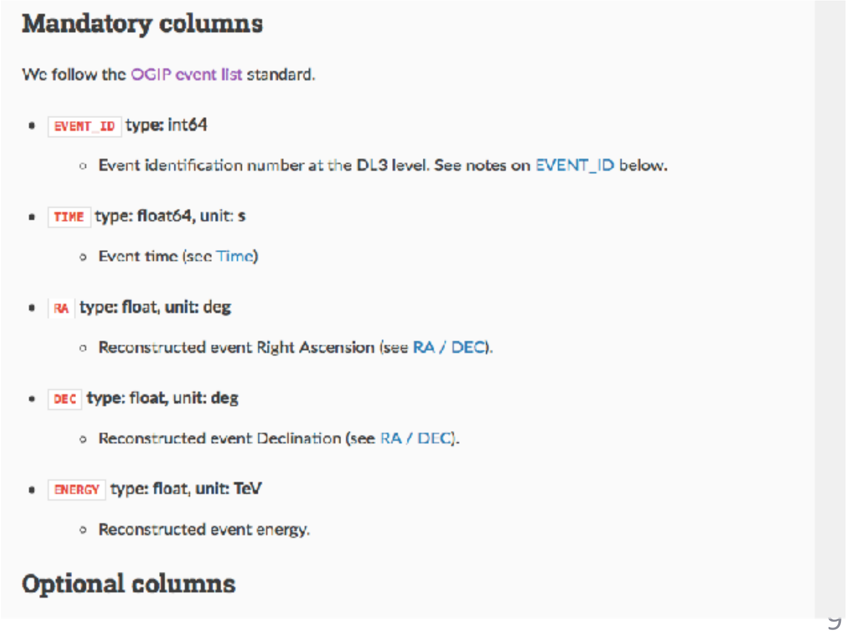
Docs » Data formats for gamma-ray astronomy

Data formats for gamma-ray astronomy

A place to propose and share data format descriptions for gamma-ray astronomy.

- Repository: https://github.com/open-gamma-ray-astro/gamma-astro-data-formats
- Docs: https://gamma-astro-data-formats.readthedocs.io/
- Mailing list: https://lists.nasa.gov/mailman/listinfo/open-gamma-ray-astro

- About
- General
- IACT events
- IACT IRFs
- IACT data storage
- Sky Maps
- Spectra
- Light curves



Mandatory columns

We follow the OGIP event list standard.

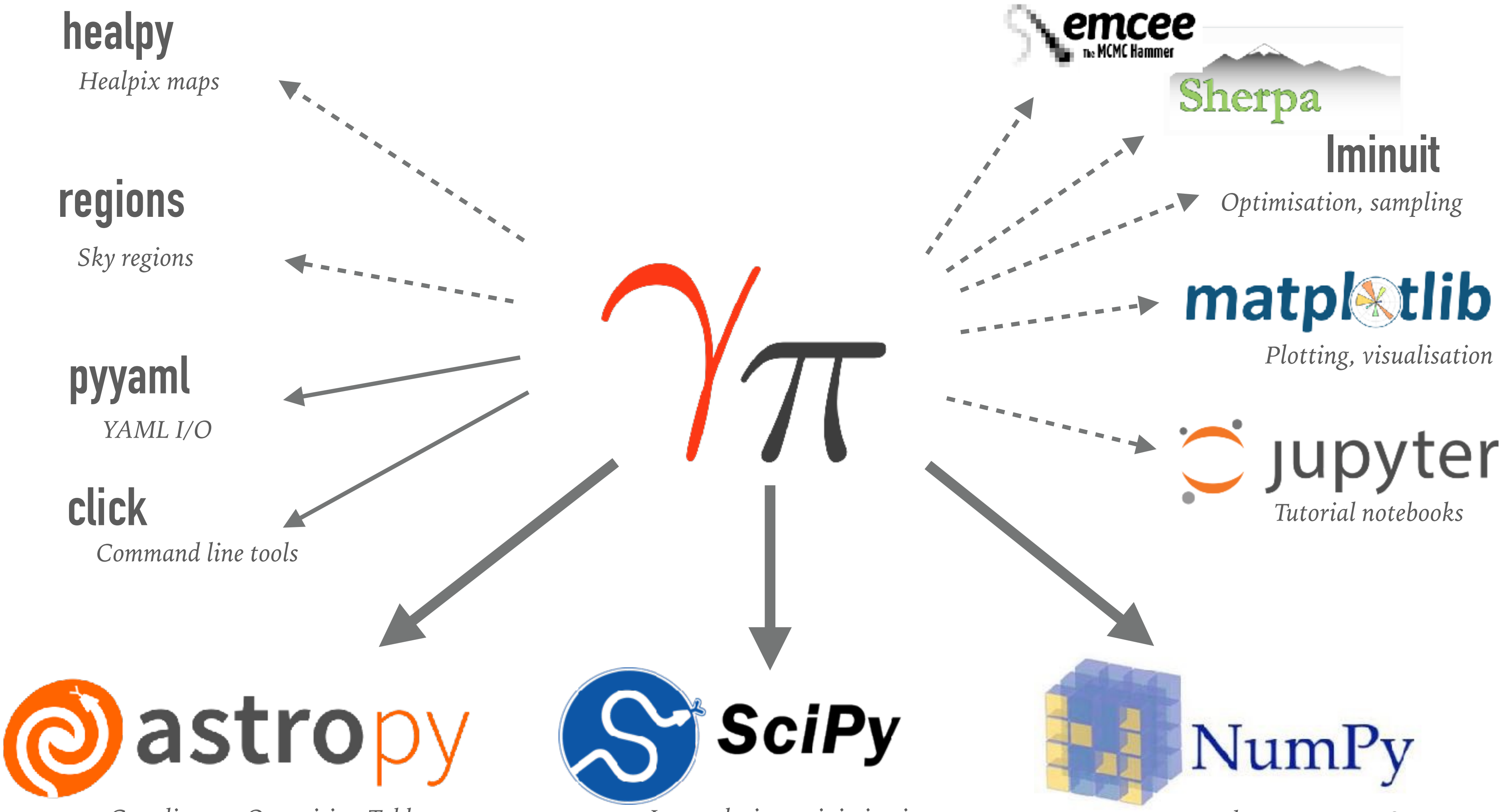
- `EVENT_ID` type: int64
  - Event identification number at the DL3 level. See notes on EVENT_ID below.
- `TIME` type: float64, unit: s
  - Event time (see Time)
- `RA` type: float, unit: deg
  - Reconstructed event Right Ascension (see RA / DEC).
- `DEC` type: float, unit: deg
  - Reconstructed event Declination (see RA / DEC).
- `ENERGY` type: float, unit: TeV
  - Reconstructed event energy.

Optional columns

Gammapy overview

Required dependencies ⟶

Optional dependencies ⇢



healpy

Healpix maps

regions

Sky regions

pyyaml

YAML I/O

click

Command line tools

emcee
The MCMC Hammer

Sherpa

iminuit

Optimisation, sampling

matplotlib

Plotting, visualisation

jupyter

Tutorial notebooks

astropy

Coordinates, Quantities, Tables, FITS I/O, etc.

SciPy

Interpolation, minimisation, FFT convolution, etc.

NumPy

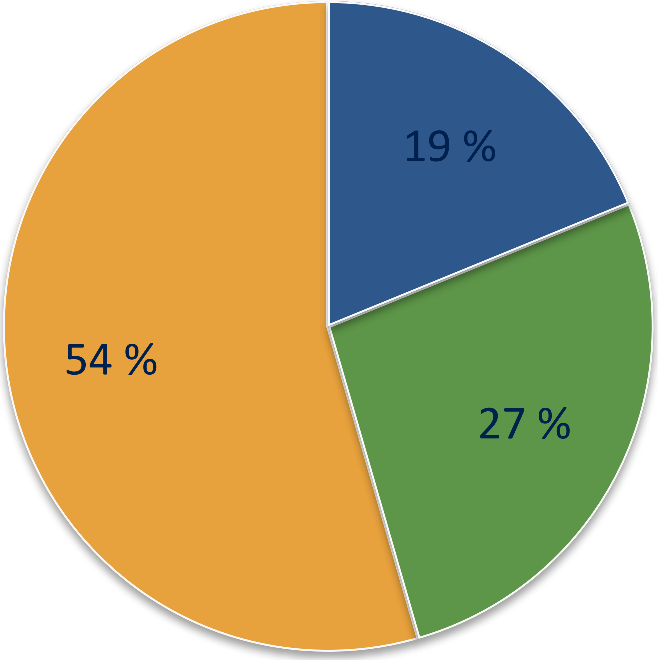
ND-data structures & computations

Overview: gammapy code

40000 lines
of python code

1500 unit & high level tests
93% coverage

pytest



documentation
& tutorials

jupyter

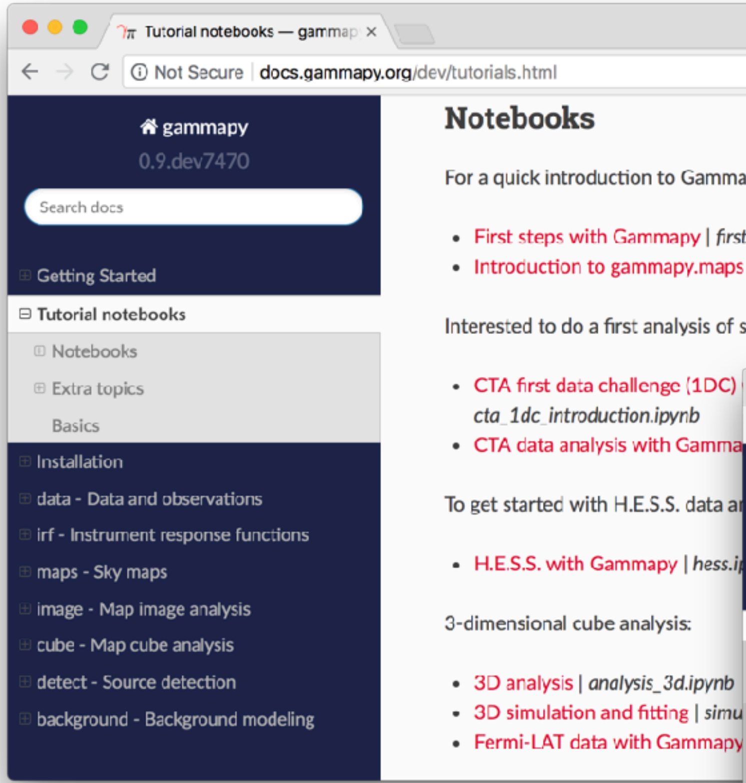
SPHINX
Python Documentation Generator

Hosted and openly developed on GitHub

Distribution with conda, pip

Documentation

https://docs.gammapy.org/


Tutorial notebooks — gammap ×
Christoph
Not Secure | docs.gammapy.org/dev/tutorials.html
gammapy
0.9.dev7470
Search docs
Getting Started
Tutorial notebooks
Notebooks
Extra topics
Basics
Installation
data - Data and observations
irf - Instrument response functions
maps - Sky maps
image - Map image analysis
cube - Map cube analysis
detect - Source detection
background - Background modeling

Notebooks

For a quick introduction to Gammapy, go here:

- First steps with Gammapy | *first_steps.ipynb*
- Introduction to gammapy.maps | *intro_maps.ipynb*

Interested to do a first analysis of simulated CTA data?

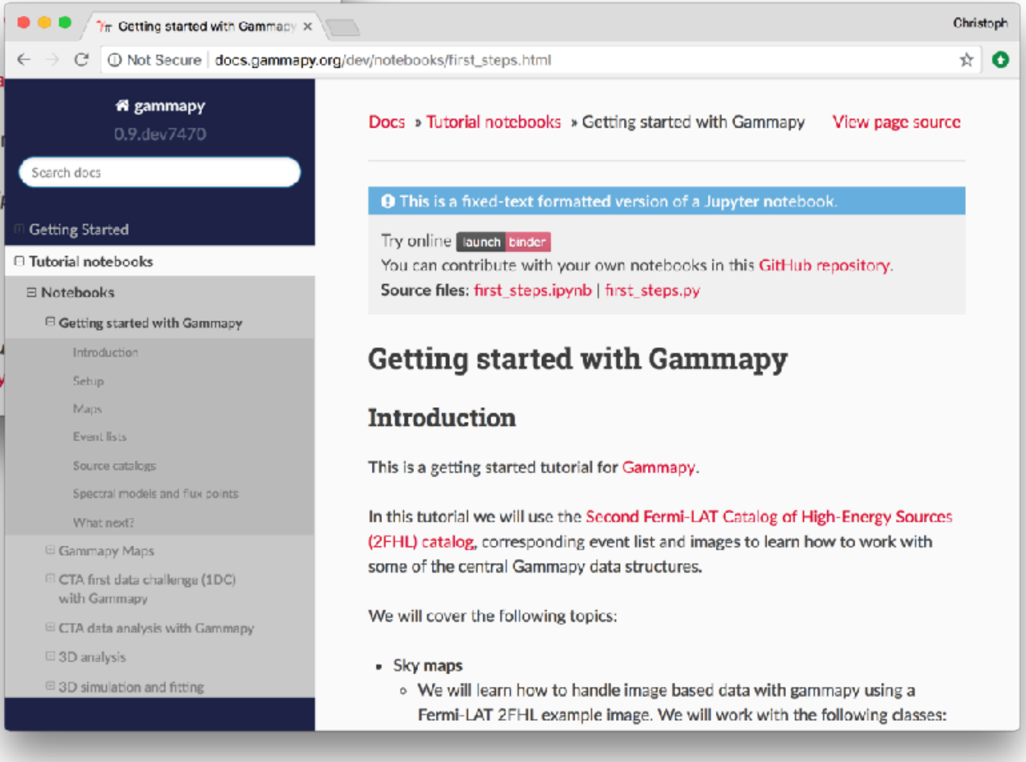
- CTA first data challenge (1DC) *cta_1dc_introduction.ipynb*
- CTA data analysis with Gamma

To get started with H.E.S.S. data a

- H.E.S.S. with Gammapy | *hess.i*

3-dimensional cube analysis:

- 3D analysis | *analysis_3d.ipynb*
- 3D simulation and fitting | *simu*
- Fermi-LAT data with Gammapy


Getting started with Gammapy ×
Christoph
Not Secure | docs.gammapy.org/dev/notebooks/first_steps.html
gammapy
0.9.dev7470
Search docs
Getting Started
Tutorial notebooks
Notebooks
Getting started with Gammapy
Introduction
Setup
Maps
Event lists
Source catalogs
Spectral models and flux points
What next?
Gammapy Maps
CTA first data challenge (1DC) with Gammapy
CTA data analysis with Gammapy
3D analysis
3D simulation and fitting
Docs » Tutorial notebooks » Getting started with Gammapy
View page source

This is a fixed-text formatted version of a Jupyter notebook.

Try online launch binder
You can contribute with your own notebooks in this GitHub repository.
Source files: first_steps.ipynb | first_steps.py

Getting started with Gammapy

Introduction

This is a getting started tutorial for Gammapy.

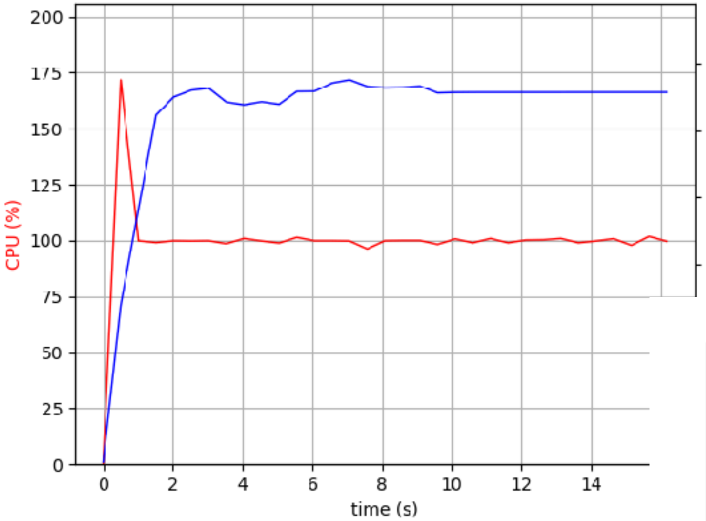
In this tutorial we will use the Second Fermi-LAT Catalog of High-Energy Sources (2FHL) catalog, corresponding event list and images to learn how to work with some of the central Gammapy data structures.

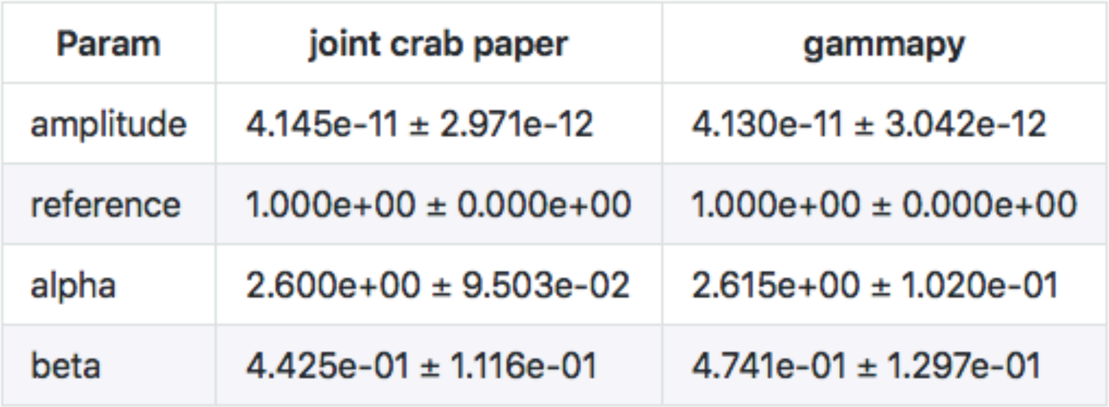
We will cover the following topics:

- Sky maps
  - We will learn how to handle image based data with gammapy using a Fermi-LAT 2FHL example image. We will work with the following classes:

Validation and benchmarks

- Regular monitoring of performances High level science validation tests comparing to established results

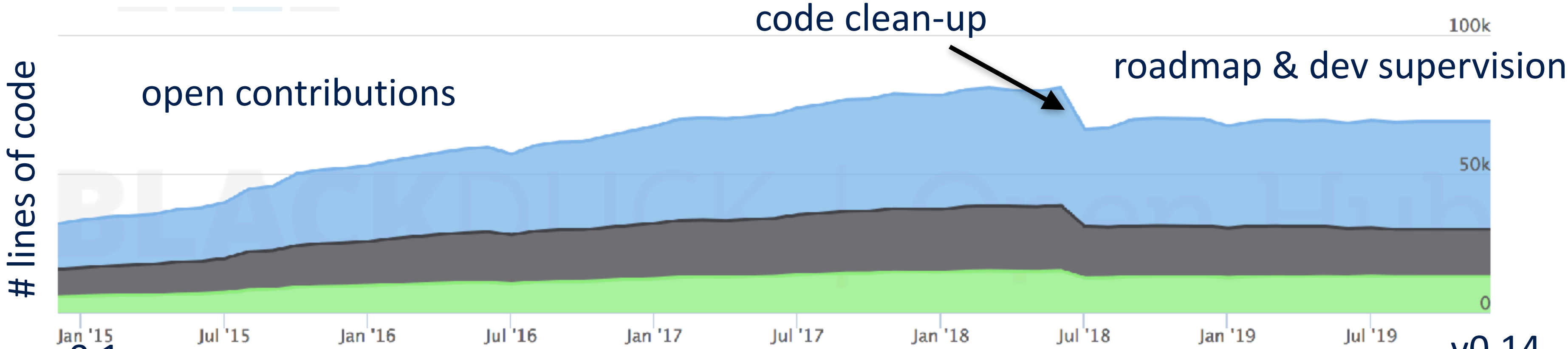




Param	joint crab paper	gammapy
amplitude	4.145e-11 ± 2.971e-12	4.130e-11 ± 3.042e-12
reference	1.000e+00 ± 0.000e+00	1.000e+00 ± 0.000e+00
alpha	2.600e+00 ± 9.503e-02	2.615e+00 ± 1.020e-01
beta	4.425e-01 ± 1.116e-01	4.741e-01 ± 1.297e-01

https://github.com/gammapy/gammapy-benchmarks/tree/master

Activity & project growth



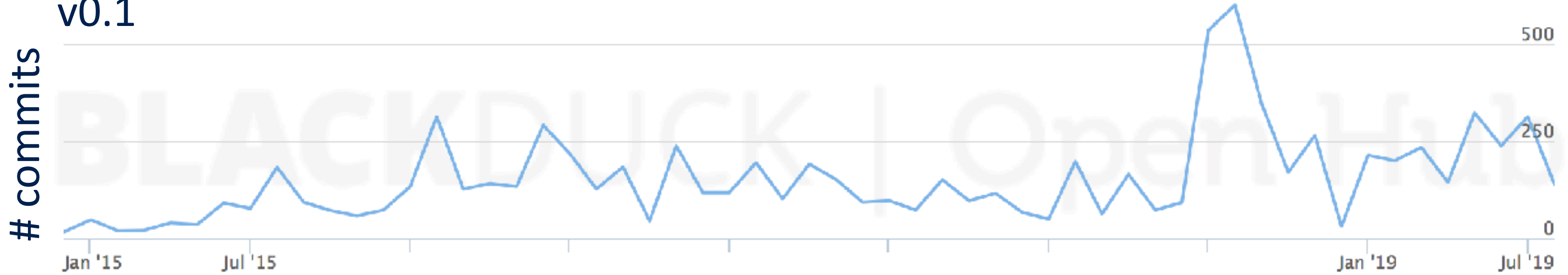
open contributions

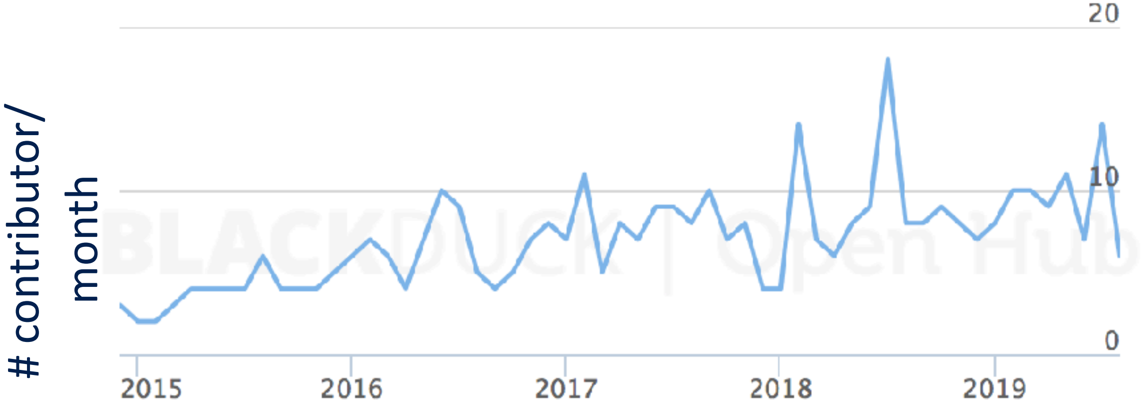
code clean-up

roadmap & dev supervision

v0.1

v0.14





stable base of
8-10 dev over
last 2.5 yrs

Gammapy project organisation

Coordination committee

APC

MAX-PLANCK-INSTITUT FÜR KERNPHYSIK

ecap

l'Observatoire de Paris

AIM

DESY

INSTITUTO de ASTROFISICA de ANDALUCIA
CSIC
I·A·A

UNIVERSIDAD COMPLUTENSE MADRID

Project managers

APC & ECAP

Lead developers

MPI-K & APC

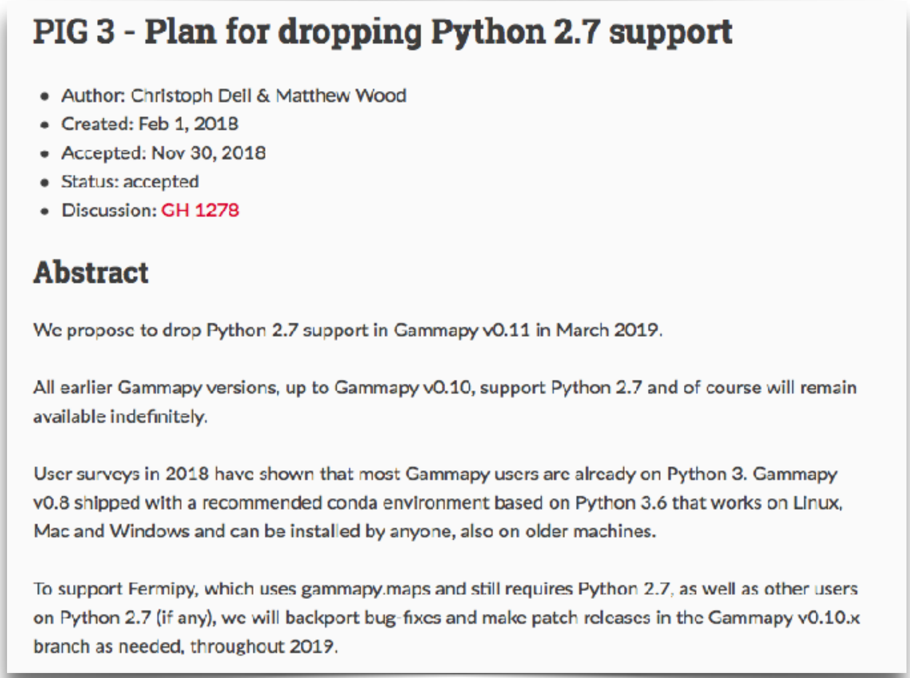
Developers

About 10 regular contributors
More than 80 contributors in total

More formal structure of the project introduced end 2018 to ensure development planning and improve interactions with experiments/observatories

Planning development : PIGs

- Proposals for Improvement of Gammapy (see PEP, APE etc)



PIG 3 - Plan for dropping Python 2.7 support

- Author: Christoph Deil & Matthew Wood
- Created: Feb 1, 2018
- Accepted: Nov 30, 2018
- Status: accepted
- Discussion: GH 1278

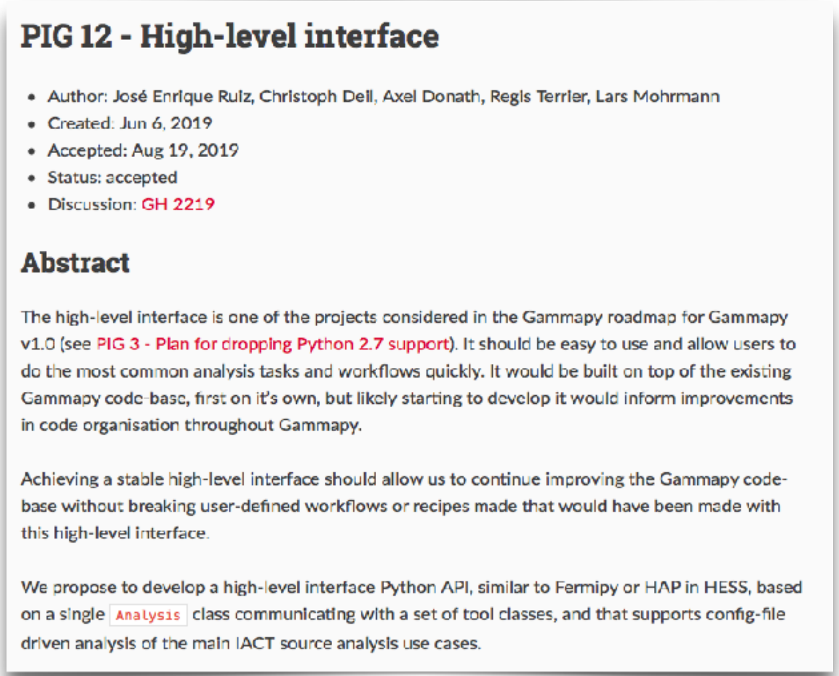
Abstract

We propose to drop Python 2.7 support in Gammapy v0.11 in March 2019.

All earlier Gammapy versions, up to Gammapy v0.10, support Python 2.7 and of course will remain available indefinitely.

User surveys in 2018 have shown that most Gammapy users are already on Python 3. Gammapy v0.8 shipped with a recommended conda environment based on Python 3.6 that works on Linux, Mac and Windows and can be installed by anyone, also on older machines.

To support Fermipy, which uses gammapy.maps and still requires Python 2.7, as well as other users on Python 2.7 (if any), we will backport bug-fixes and make patch releases in the Gammapy v0.10.x branch as needed, throughout 2019.



PIG 12 - High-level interface

- Author: José Enrique Ruiz, Christoph Deil, Axel Donath, Regis Terrier, Lars Mohrmann
- Created: Jun 6, 2019
- Accepted: Aug 19, 2019
- Status: accepted
- Discussion: GH 2219

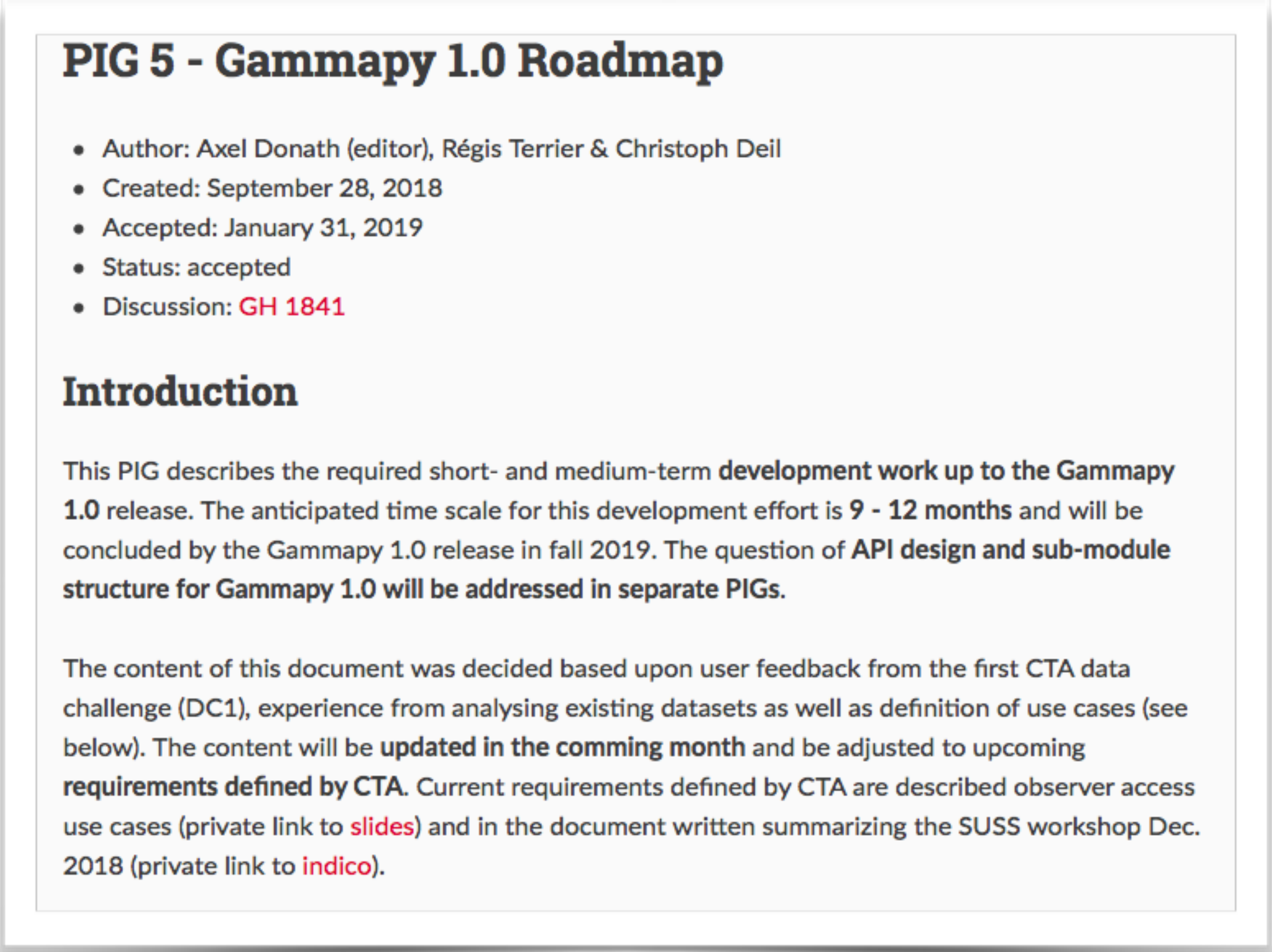
Abstract

The high-level interface is one of the projects considered in the Gammapy roadmap for Gammapy v1.0 (see PIG 3 - Plan for dropping Python 2.7 support). It should be easy to use and allow users to do the most common analysis tasks and workflows quickly. It would be built on top of the existing Gammapy code-base, first on it's own, but likely starting to develop it would inform improvements in code organisation throughout Gammapy.

Achieving a stable high-level interface should allow us to continue improving the Gammapy code-base without breaking user-defined workflows or recipes made that would have been made with this high-level interface.

We propose to develop a high-level interface Python API, similar to Fermipy or HAP in HESS, based on a single `Analysis` class communicating with a set of tool classes, and that supports config-file driven analysis of the main IACT source analysis use cases.

Gammapy 1.0 Roadmap



PIG 5 - Gammapy 1.0 Roadmap

- Author: Axel Donath (editor), Régis Terrier & Christoph Deil
- Created: September 28, 2018
- Accepted: January 31, 2019
- Status: accepted
- Discussion: GH 1841

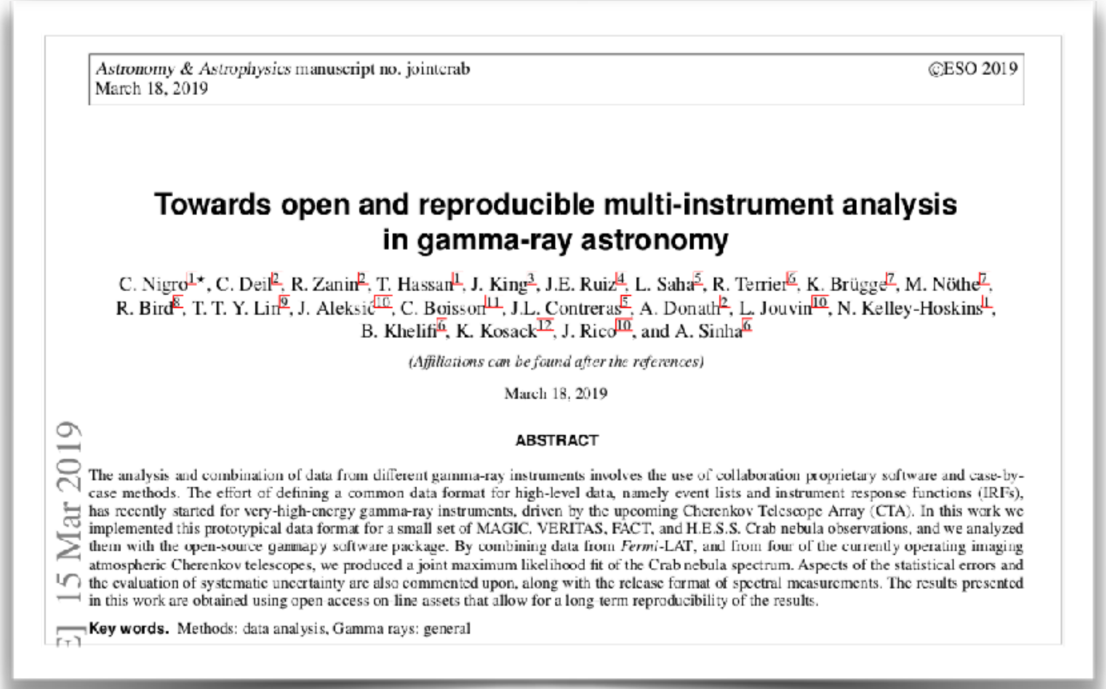
Introduction

This PIG describes the required short- and medium-term **development work up to the Gammapy 1.0** release. The anticipated time scale for this development effort is **9 - 12 months** and will be concluded by the Gammapy 1.0 release in fall 2019. The question of **API design and sub-module structure for Gammapy 1.0 will be addressed in separate PIGs.**

The content of this document was decided based upon user feedback from the first CTA data challenge (DC1), experience from analysing existing datasets as well as definition of use cases (see below). The content will be **updated in the comming month** and be adjusted to upcoming **requirements defined by CTA.** Current requirements defined by CTA are described observer access use cases (private link to slides) and in the document written summarizing the SUSS workshop Dec. 2018 (private link to indico).

joint-crab paper

- Fully reproducible multi-instrument gamma-ray analysis
- Online material: https://github.com/open-gamma-ray-astro/joint-crab/
- All data, scripts and notebooks provided. Reproducibility:
  - conda distribution for local execution or binder environment
  - docker image provided for longer term reproducibility



Astronomy & Astrophysics manuscript no. jointcrab
March 18, 2019

©ESO 2019

Towards open and reproducible multi-instrument analysis in gamma-ray astronomy

C. Nigro[1*], C. Deil[2], R. Zanin[2], T. Hassan[1], J. King[3], J.E. Ruiz[4], L. Saha[5], R. Terrier[6], K. Brügge[7], M. Nöthe[7], R. Bird[8], T. T. Y. Lin[9], J. Aleksić[10], C. Boisson[11], J.L. Contreras[5], A. Donath[2], L. Jouvin[10], N. Kelley-Hoskins[1], B. Khelifi[6], K. Kosack[12], J. Rico[10], and A. Sinha[6]

(Affiliations can be found after the references)

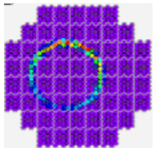
March 18, 2019

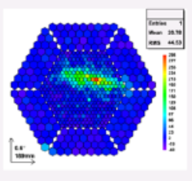
ABSTRACT

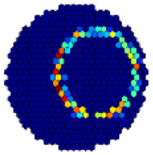
The analysis and combination of data from different gamma-ray instruments involves the use of collaboration proprietary software and case-by-case methods. The effort of defining a common data format for high-level data, namely event lists and instrument response functions (IRFs), has recently started for very-high-energy gamma-ray instruments, driven by the upcoming Cherenkov Telescope Array (CTA). In this work we implemented this prototypical data format for a small set of MAGIC, VERITAS, FACT, and H.E.S.S. Crab nebula observations, and we analyzed them with the open-source gammapy software package. By combining data from *Fermi*-LAT, and from four of the currently operating imaging atmospheric Cherenkov telescopes, we produced a joint maximum likelihood fit of the Crab nebula spectrum. Aspects of the statistical errors and the evaluation of systematic uncertainty are also commented upon, along with the release format of spectral measurements. The results presented in this work are obtained using open access on-line assets that allow for a long-term reproducibility of the results.

Key words. Methods: data analysis, Gamma rays: general

joint-crab paper



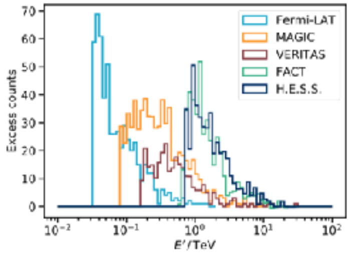


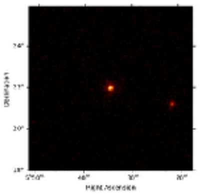


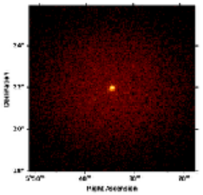
Raw data

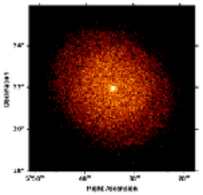
Dataset	T_{obs}	E_{min} (TeV)	E_{max} (TeV)	N_{on}	N_{bkg}	R_{on} (deg)
Fermi-LAT	~7 yr	0.03	2	578	1.2	0.30
MAGIC	0.66 h	0.08	30	784	129.9	0.14
VERITAS	0.67 h	0.16	30	289	13.7	0.10
FACT	10.33 h	0.45	30	691	272.8	0.17
H.E.S.S.	1.87 h	0.71	30	459	27.5	0.11

DL3 gadf





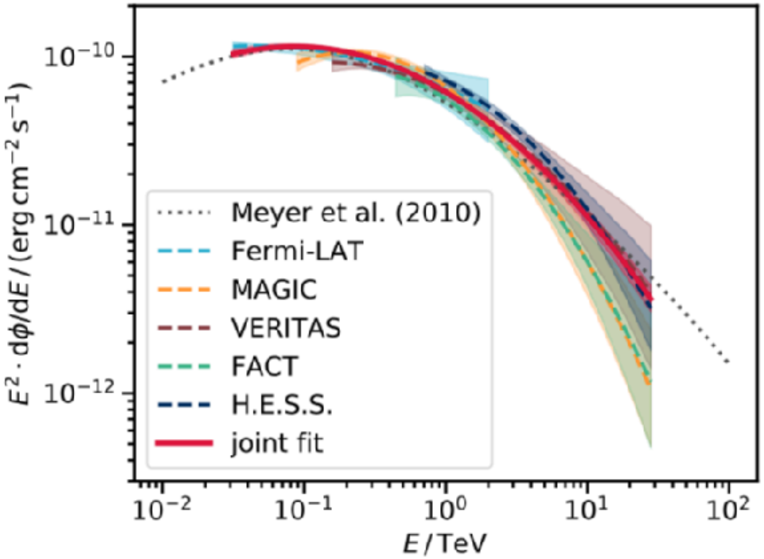




High level science products

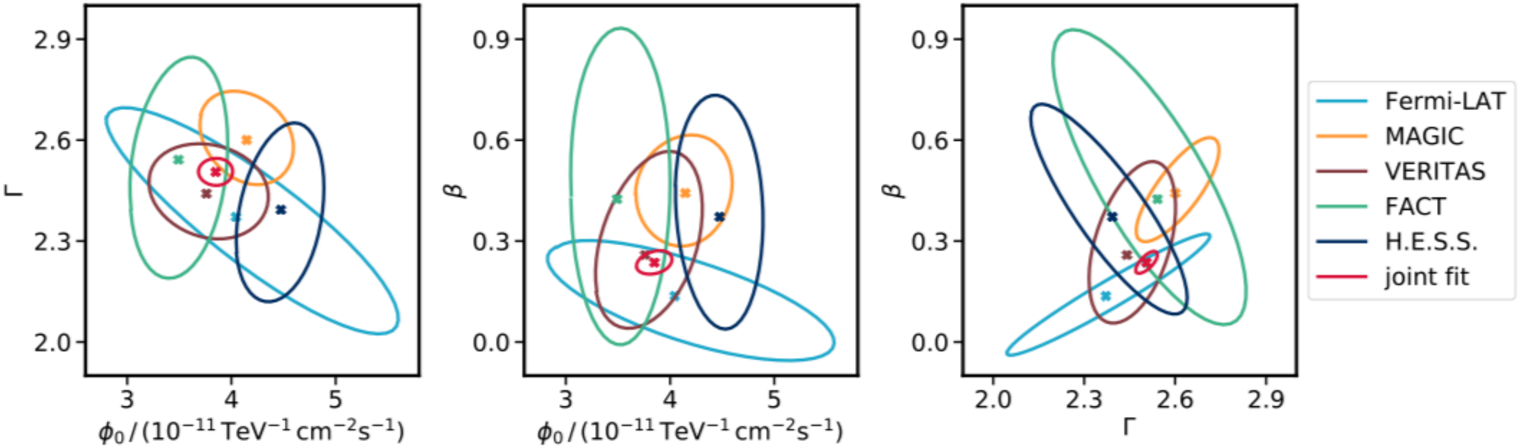
joint-crab paper

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



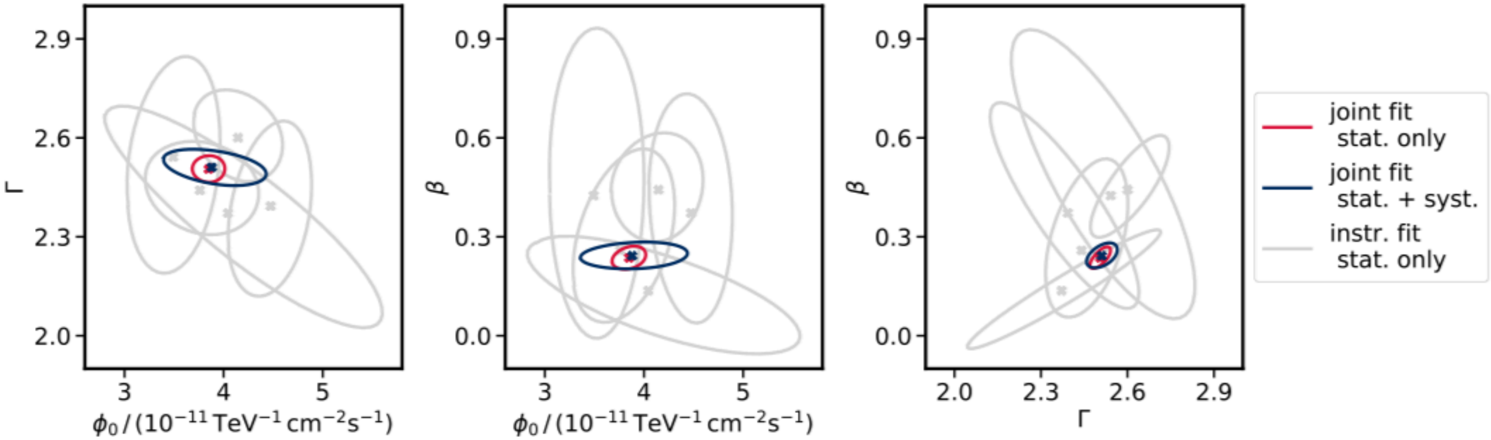
Explore results:

https://github.com/open-gamma-ray-astro/joint-crab/blob/master/2_results.ipynb



joint-crab paper

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



See notebook:
https://github.com/open-gamma-ray-astro/joint-crab/blob/master/3_systematics.ipynb

- Can perform spectral fits on the parent particle population

See notebook:
https://github.com/open-gamma-ray-astro/joint-crab/blob/master/4_naima.ipynb

Summary

- Common & open data formats are prototyped within the VHE γ-ray community
- Gammapy is a python package for γ-ray astronomy:
  - open source & community driven.
  - developed on GitHub
  - relies on broadly used open source libraries
  - light-weight, easily adaptable
- Gammapy already allows for reproducible, multi-instrument analysis, see e.g. joint-crab effort
  - supports all data in g.a.d.f. DL3 format
  - HAWC data analysis support is coming
  - prototype tools for Km3Net neutrino data analysis currently developed