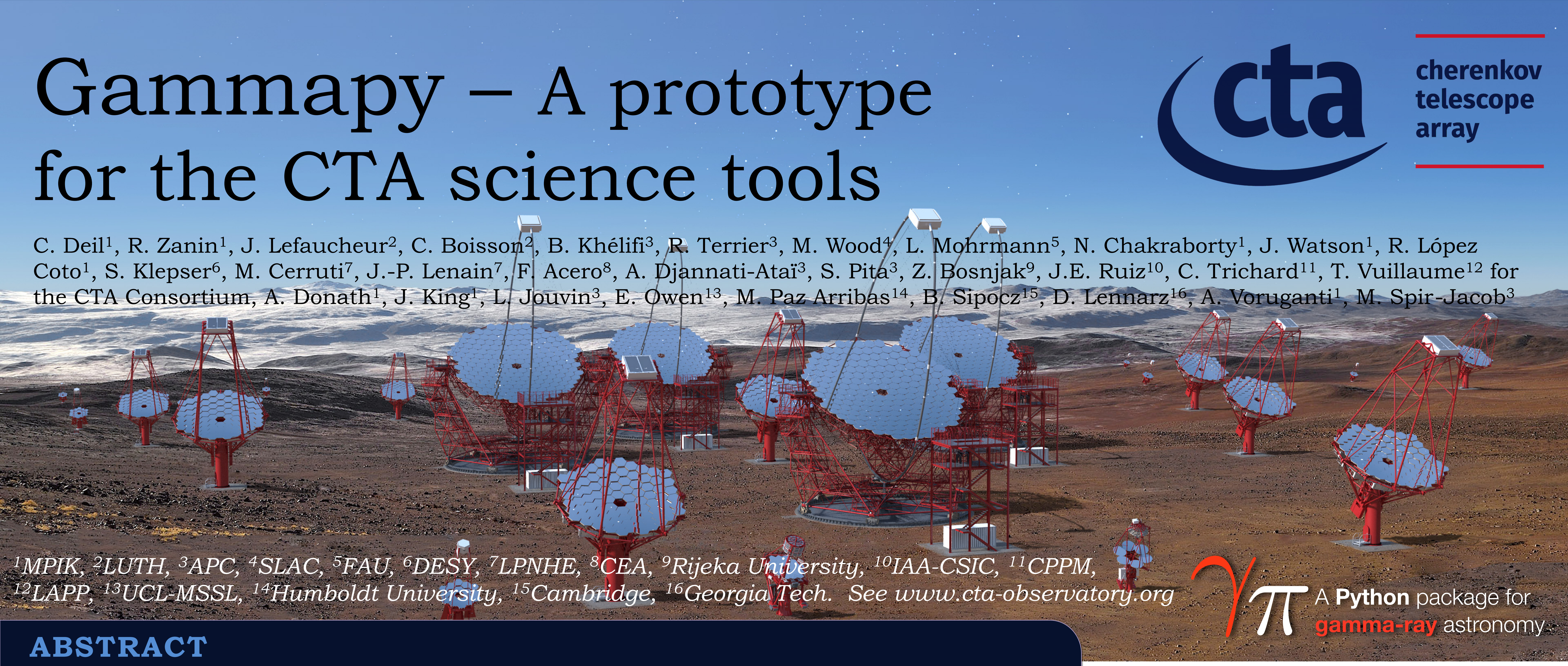

Gammapy – A prototype for the CTA science tools

C. Deil[1], R. Zanin[1], J. Lefaucheur[2], C. Boisson[2], B. Khélifi[3], R. Terrier[3], M. Wood[4], L. Mohrmann[5], N. Chakraborty[1], J. Watson[1], R. López Coto[1], S. Klepser[6], M. Cerruti[7], J.-P. Lenain[7], F. Acero[8], A. Djannati-Ataï[3], S. Pita[3], Z. Bosnjak[9], J.E. Ruiz[10], C. Trichard[11], T. Vuillaume[12] for the CTA Consortium, A. Donath[1], J. King[1], L. Jouvin[3], E. Owen[13], M. Paz Arribas[14], B. Sipocz[15], D. Lennarz[16], A. Voruganti[1], M. Spir-Jacob[3]

[1]MPIK, [2]LUTH, [3]APC, [4]SLAC, [5]FAU, [6]DESY, [7]LPNHE, [8]CEA, [9]Rijeka University, [10]IAA-CSIC, [11]CPPM, [12]LAPP, [13]UCL-MSSL, [14]Humboldt University, [15]Cambridge, [16]Georgia Tech. See www.cta-observatory.org

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

ABSTRACT

Gammapy is a Python package for gamma-ray astronomy, built on Numpy and Astropy, and a prototype for the Cherenkov Telescope Array (CTA) science tools.
Here we give an introduction to Gammapy and show an application example how to create a sky image using simulated CTA event data and instrument response functions.
For further information, visit docs.gammapy.org and follow the links to "tutorials" and "CTA".

Supported by:

CTA science tools

The Cherenkov Telescope Array (CTA) will observe the sky in very-high-energy gamma-ray light soon. All astronomers will have access to CTA high-level data, as well as CTA science tools (ST) software. The ST can be used for example to generate sky images and to measure source properties such as morphology, spectra and light curves, using event lists as well as instrument response function (IRF) and auxiliary information as input.

Gammapy

Gammapy is a prototype for the CTA ST, built on the scientific Python stack and Astropy, optionally using Sherpa or other packages for modeling and fitting (see Figure 1).

Our initial focus was to implement the "classical TeV analysis", i.e. using 2-dimensional sky images for source detection and morphology characterisation, followed by spectral analysis for a given source region. A 3-dimensional analysis with a simultaneous spatial and spectral models of the gamma-ray emission, as well as background is in development.

Further developments and verification using data from existing Cherenkov telescope arrays such as H.E.S.S. and MAGIC, as well as simulated CTA data is ongoing.

CTA applications

This poster and proceeding is focused on the Gammapy software package. We show an example of a sky image and spectrum computed with Gammapy in Figures 3 and 4.

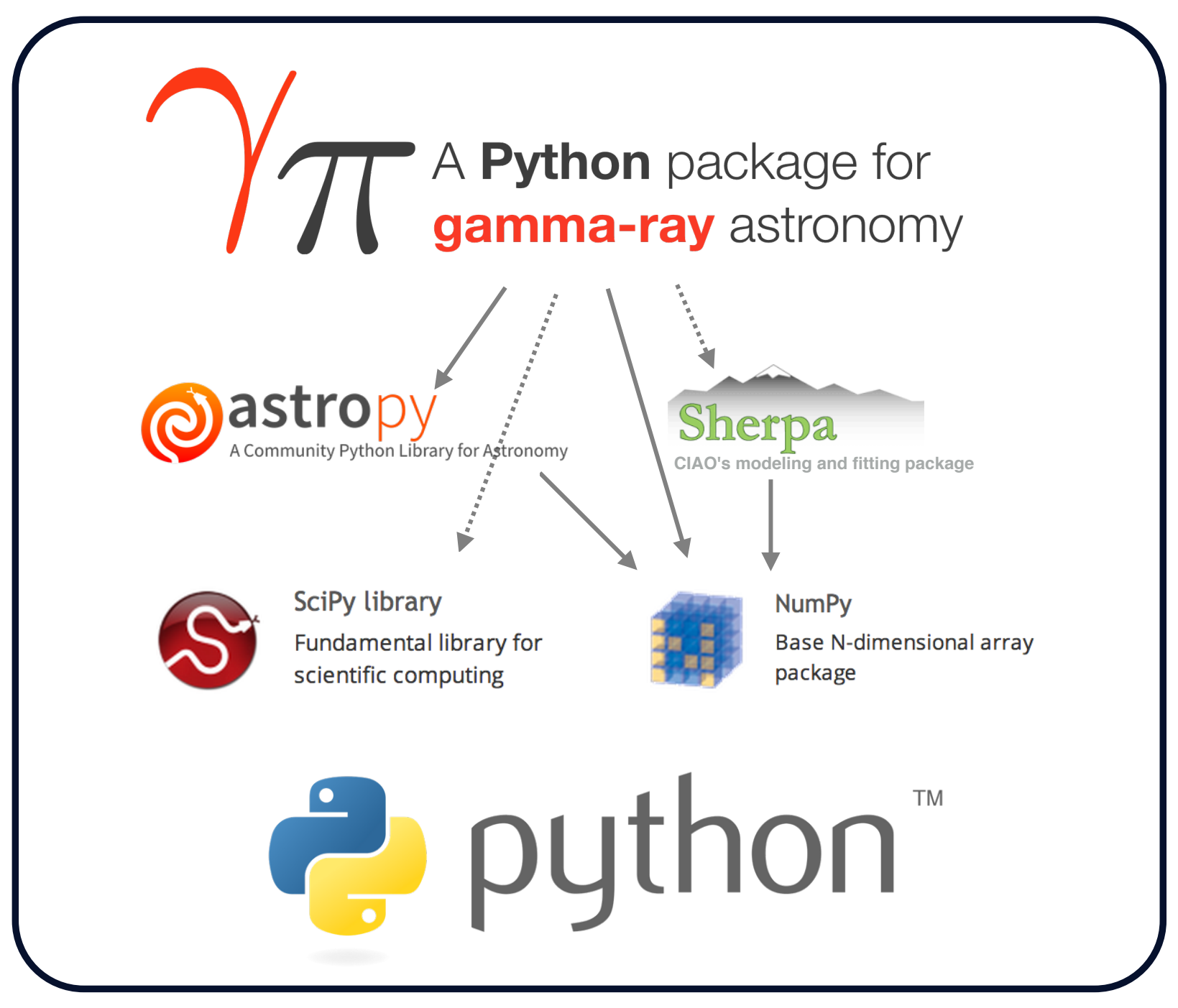
A Python package for gamma-ray astronomy
astropy
A Community Python Library for Astronomy
Sherpa
CIAO's modeling and fitting package
SciPy library
Fundamental library for scientific computing
NumPy
Base N-dimensional array package
python

Fig. 1: Gammapy approach: build on the scientific Python stack; use and collaborate with existing astro packages

```
"""Make a counts image with Gammapy."""
from gammapy.data import EventList
from gammapy.image import SkyImage
events = EventList.read('events.fits')
image = SkyImage.empty(
  nxpix=400, nypix=400, binsz=0.02,
  xref=83.6, yref=22.0,
  coordsys='CEL', proj='TAN',
)
image.fill_events(events)
image.write('counts.fits')
```

Fig. 2: Gammapy code example: efficiently work with events and pixels from Python, by always storing data in Numpy arrays

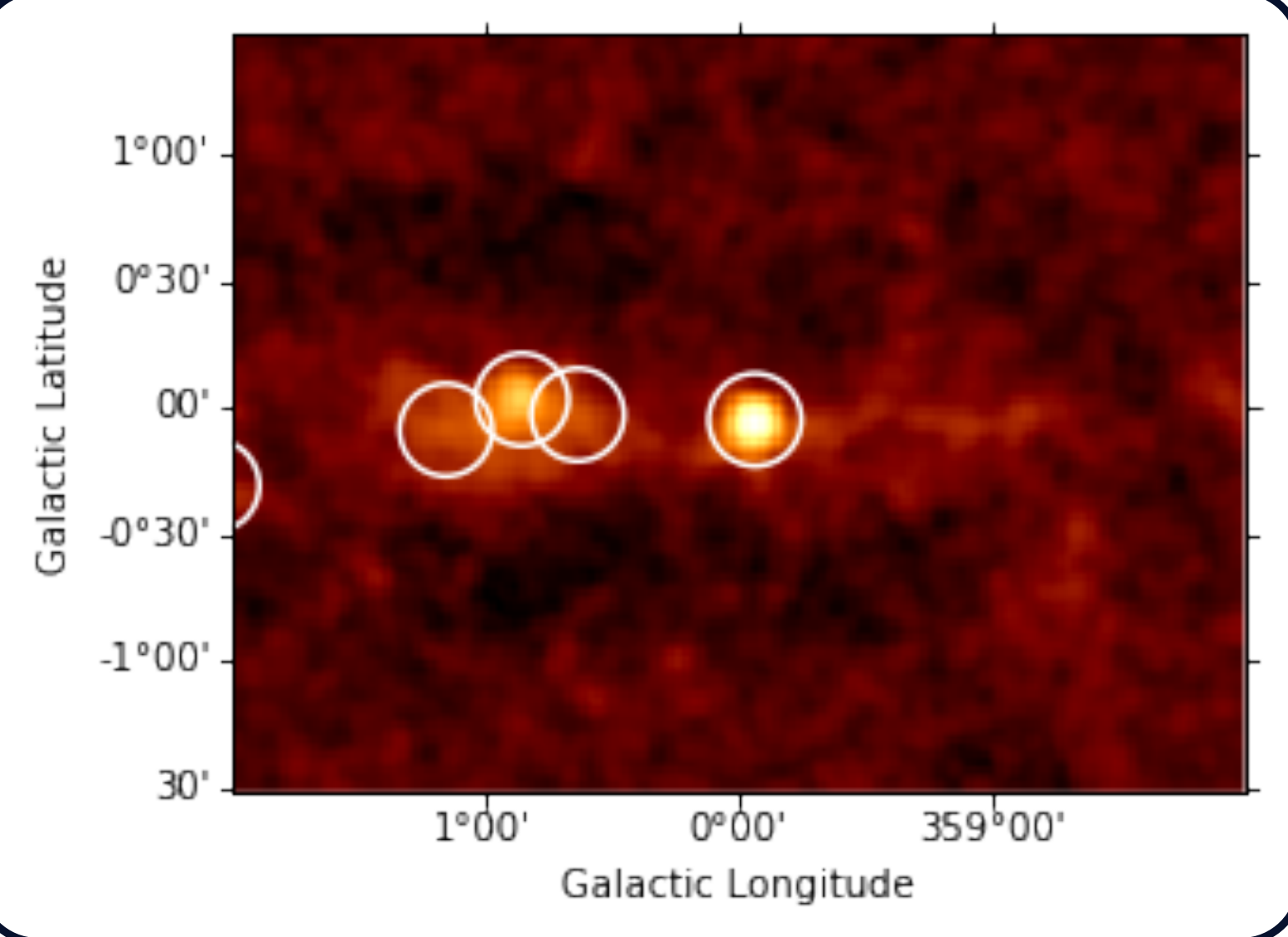

Fig. 3: Significance image computed with Gammapy. Galactic centre region using 1.5 hours of simulated data with CTA South.

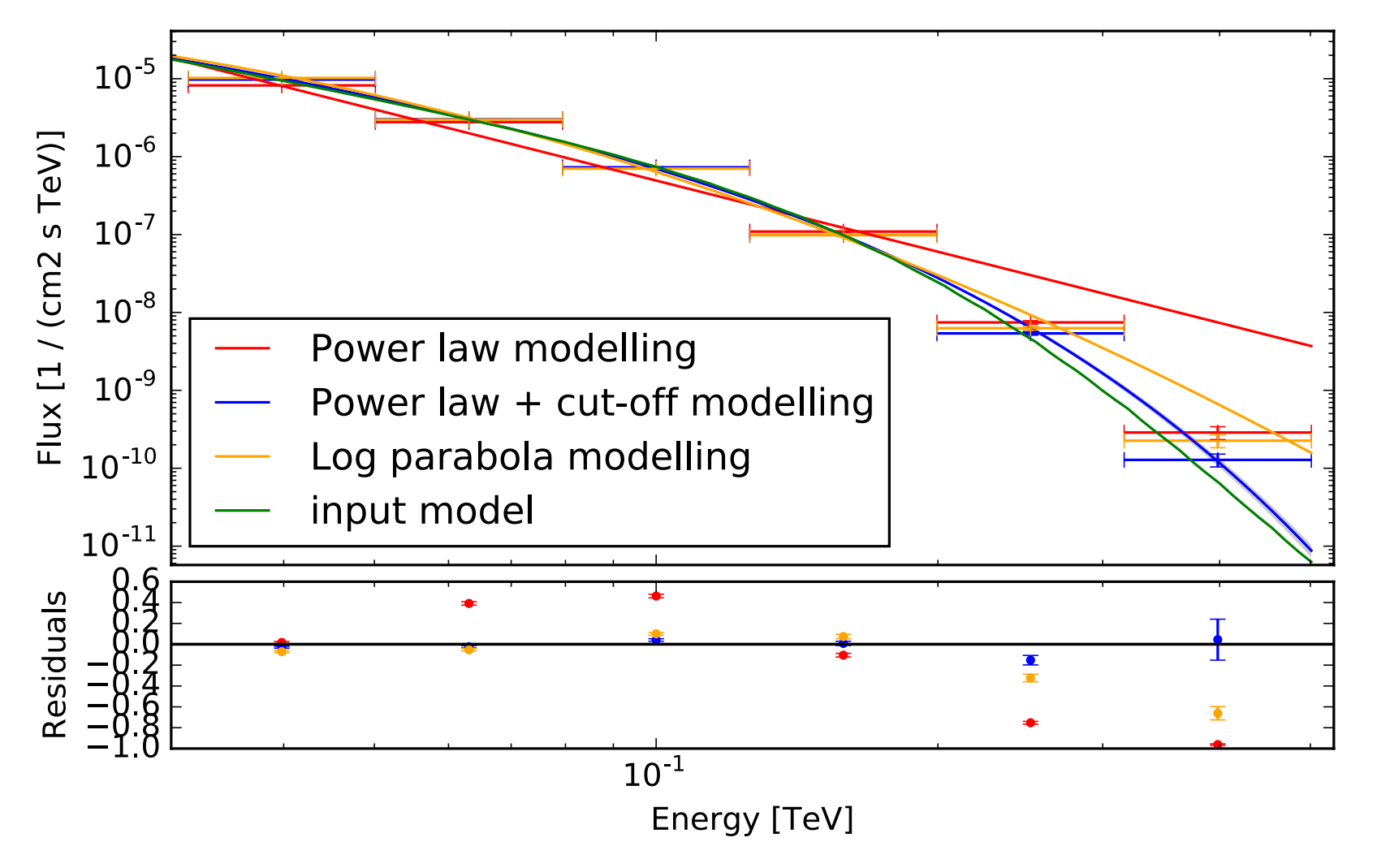

Fig. 4: Spectrum simulated and analysed with Gammapy. Example of a very short exposure (100 seconds) with CTA of a very bright source.

Getting started

Visit docs.gammapy.org and browse the tutorial Jupyter notebooks to get an overview of what Gammapy can do.

To install Gammapy, use one of

```
pip install gammapy
conda install -c astropy gammapy
```

try out some of the tutorials locally, and then adapt them to your application and use cases.

Note that Gammapy is under very active development. If you have any questions, feature requests or find an issue, please contact the Gammapy mailing list or issue tracker.

For further CTA application examples using Gammapy, see the following poster here at ICRC 2017:
R. Zanin (Galactic plane), C. Trichard (PeVatrons), J. Lefaucheur (extra-galactic sources)

Behind the scenes ...

Figure 2 shows a Gammapy code example that generates a counts image from an event list.

The key point here is that all data are stored in Numpy arrays and processed efficiently via calls into existing C extensions in Numpy and Astropy. For instance, in the example, the EventList and SkyImage objects store coordinates and pixel data as Numpy arrays, respectively. Data processing routines such as image.fill(events) are based on Numpy histograms and calls into the CFITSIO and WCSLib C libraries.

Conclusions

In the past two years, we have developed Gammapy as an open-source analysis package for existing gamma-ray telescope and as a prototype for the CTA science tools.

We find that the Gammapy approach, to build on the powerful and well-tested Python packages Numpy and Astropy, brings large benefits: A small codebase that is focused on gamma-ray astronomy in a single high-level language is easy to understand and maintain. It is also easy to modify and extend as new use cases arise, which is important for CTA, since it can be expected that the modeling of the instrument, background and astrophysical emission, as well as the analysis method in general will evolve and improve over the next decade.

Last but not least, the Gammapy approach is inherently collaborative, sharing development effort as well as know-how with the larger astronomical community, that to a large degree already has adopted Numpy and Astropy as the basis for astronomical analysis codes in the past 5 years.

ACKNOWLEDGEMENTS

We gratefully acknowledge support from the agencies and organizations under *Funding Agencies* at www.cta-observatory.org