Gammapy
high-level interface

Christoph Deil (MPIK)
Feb 5, 2018

Status

- So far, Gammapy is mostly a Python package,
  i.e. a library of functions and classes
- Analysis: write a Python script or Jupyter notebook
- This is very flexible, but pretty low level (user needs to know and import 10 or 20 functions or classes) and leads to a lot of copy & pasting of similar 10 - 100 lines of code for each analysis

```
import astropy.units as u
from gammapy.datasets import gammapy_extra
from gammapy.spectrum import SpectrumObservation, SpectrumFit, models

filename = '$GAMMAPY_EXTRA/datasets/hess-crab4_pha/pha_obs23592.fits'
obs = SpectrumObservation.read(filename)

model = models.PowerLaw(
    index=2 * u.Unit(''),
    amplitude=1e-12*u.Unit('cm-2 s-1 TeV-1'),
    reference=1*u.TeV,
)
fit = SpectrumFit(obs_list=obs, model=model)
fit.fit()
fit.est_errors()
print(fit.result[0])
```

Options

- Common options to make a high-level interface:
  - Config file, e.g. HAP or Fermipy
  - Tool suite, e.g. Fermi ST or ctools
  - High-level analysis Python functions or classes
    that are basically passed config, execute run method, get results
- Each has pros and cons, e.g. flexibility vs simplicity. For IACT analysis there is also the curse of many different analysis methods (1D/2D/3D, stacked?, background method), leading to complexity for any solution (10s of steps with 100s of options) and the question how to organise the functionality in a good way.
- In Gammapy, we have experimented with all of those.
  - Config: gammapy.scripts.SpectrumAnalysisIACT
  - Tool: gammapy image bin
  - Class: gammapy.image.IACTBasicImageEstimator
- In Python, there are ways to write functionality once and have it available via code, configuration, command line (see e.g. python-fire, or also what HAP, ctapipe or ctools are doing) avoiding wrapper code to expose it in different ways.

Plan

- At the moment our focus is still on developing the flexible Python package for Gammapy
- Maybe we should only come back to this in a few months, once Gammapy core functionality is in better shape
- If we make this a topic now, we could
  - Take a detailed look at and discuss options, since some solutions require up-front decision and adding some code to every function and class to achieve configurability, good logging, provenance, in-memory analysis workflows.
  - Continue with gammapy command line interface, just wrap more functionality (e.g. make maps, PSF, …) in the style of the existing “gammapy image bin”. These are good first pull requests.