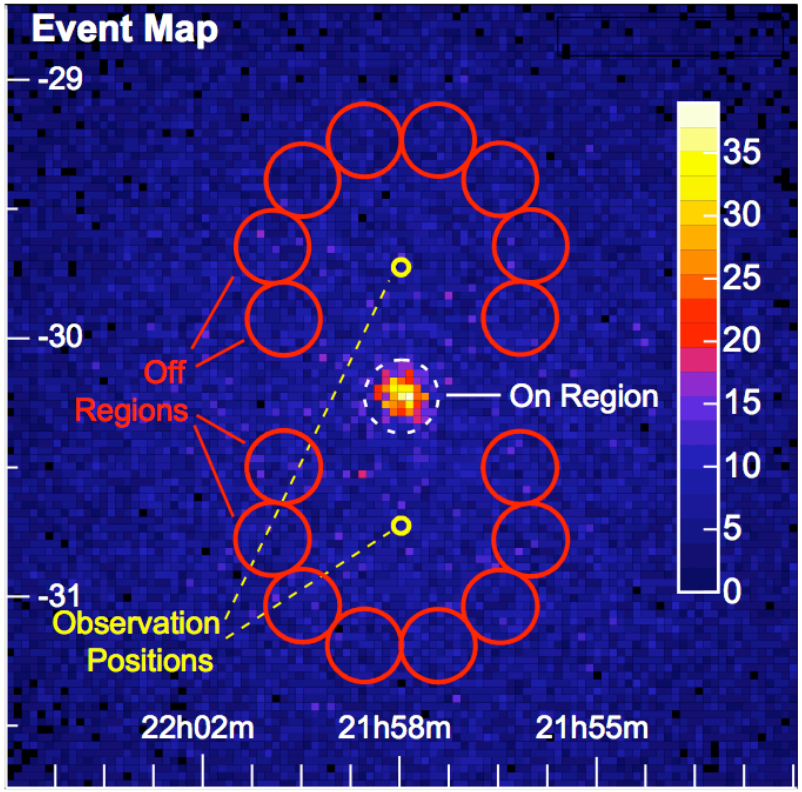
Implementing the traditional analysis methods of TeV data in Gammalib/ctools

Pierrick MARTIN, Jürgen Knödlseder (IRAP)
Anneli Schulz, Michael Mayer (DESY)
Christop Deil, Axel Donath (MPIK)

Objectives

To begin with

- Make a spectrum for a point source
- Observations taken in wobble mode
- Determine flux points and significance
- Determine best single power-law fit
- Background estimation : reflected regions (Berge et al. 2007)



Further steps

- Excess and significance maps
- More sophisticated background estimation methods, exclusion regions
- (and more simple background estimation method: ON-OFF in different pointings ?)
- Extended emissions
- …

Technical implementation

Why this does not easily fit in the current Gammalib approach

- It is assumed that any sky model GModelSky is convolved with the IRF
- The likelihood function is predefined in the Gobservation class (this will change)
- Other potential problems like interfaces, unaccessible functionalities,…

The way ahead : dedicated implementation

- Based on dedicated classes
- Pathfinder to identify needs, problems, and get to know gammalib better
- Merging with the current approach/classes may come later
- Refactorizing is always necessary so better start now

Technical implementation

Approach: Test-driven development

- See draft of final python script https://cta-redmine.irap.omp.eu/issues/570
- Gives the main steps and the kind of objects/functionalities we would like to have

Main steps (to be completed…?)

- Simulate / load data
- Define ON region
- Define OFF regions
- Set energy range/binning
- Make ON and OFF counts spectra
- Compute energy-dependent exposure in ON region
- Compute energy-dependent acceptance in ON and OFF regions
- Perform fitting and compute significances

Technical implementation

Sky region class

- GSkyRegions : sky regions container
    - GSkyRegion : sky region interface
        - GSkyRegionCircle : implementation of circular sky region
        - GSkyRegionRing : implementation or ring sky region

Functionalities to be implemented/discussed

- Data/instrument-specific functionalities : integrate events, exposure, acceptance,…
- Region definition from pointing characteristics: make N reflected regions for pointing p…
- Or keep it to the bare minimum and handle all that elsewhere…
- Format for region saving, start with XML defined by us ?

Technical implementation

ON-OFF fitter class

- GOnOffFitter (or any other name)
- At the moment, where everything will be packed
    - GOnOffFitter.prepare_off(obs,on,pars) … or handle that out (but OFF tied to pointings)
    - GOnOffFitter.make_spectra(on,off,ebds)
    - GOnOffFitter.compute_exposure(obs,irf,ebds) … or handled by GSkyRegion
    - GOnOffFitter.compute_acceptance(obs,irf,ebds) … or handled by GSkyRegion
    - GOnOffFitter.optimize(opt,model)

Functionalities to be implemented/discussed

- PSF integration over ON region to correct effective area
- Acceptance calculation ? Can be done from obs.models() for simulated data…
- What about energy dispersion ?
- Optimization function to be defined as a GOptimizerFunction ?
- Li&Ma-1983 arithmetics

Work ahead

At the moment

- « classical-analysis » branch on the git repository
- GSkyRegions, GSkyRegion,GSkyRegionCircle header files (in my working copy)
- End-user final script draft

Soon

- Decide GSkyRegion functionalities
- Clarify GOnOffFitter structure (what goes in, what goes out or derive from other classes)

And then…

- Who does what ? Use backlog after redefinition of existing tasks
- Identify test steps in the development
- Identify a final test case (for comparison with a HESS analysis pipeline)
- Be ready for next CTA consortium meeting ?