# GammaLib - Change request #2674

## Application of thresholds in the on/off analysis case including energy dispersion

09/05/2018 03:40 PM - Specovius Andreas

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 09/05/2018 |
| **Priority:** | Urgent | **Due date:** | |
| **Assigned To:** | Knödlseder Jürgen | **% Done:** | 100% |
| **Category:** | | **Estimated time:** | 0.00 hour |
| **Target version:** | 1.6.0 | | |

**Description**

## Introduction

The predicted flux in reconstructed energy is defined via the product of model flux, effective area and livetime.

Including energy dispersion means that the predicted flux in reconstructed energy has to be derived via summing over the flux predicted in each true energy bin weighted with the energy dispersion relating true and reconstructed energy.

Including energy threshold means that, on the one hand, the observed counts spectra should be cutted below the threshold and, on the other hand, it should be ensured that the predicted counts spectra in reconstructed energy should contain no counts below threshold.
The easiest way to deal with the latter is to modify the energy dispersion so that a redistribution of true energies to reconstructed energies below threshold is prevented.

In cta on/off the application of the energy threshold is mainly done in GCTAOnOffObservation::apply_ebounds().

## My frame

Joint on/off analysis of H.E.S.S. data using the latest ctools dev (Aug03) / gammapy dev (Aug04) versions.

## The problem

Including energy dispersion, you will aggree that model flux for true energies below the threshold has a nonzero probability to be reconstructed above threshold.
In order to ensure that this flux is considered, an effective area is needed for computation that features reasonable values over the whole energy range, including regimes beyond the threshold.

Obviously in ctools the effective area is set to zero beyond the threshold.
(See GCTAAeff2D::operator() which basically applies the energy threshold via an energy bounds check.)

That reduces the number of events predicted in the reconstructed energy range in the vicinity of the threshold and affects the model fitting.

## Validation

Removing the check for the energy bounds in GCTAAeff2D::operator() (which of course is just a temporary kludge for validation and not a solution at all!) basically leaves the effective area like it is and does not set it to zero beyond threshold.

A test with H.E.S.S. PDR data on pks2155 now results in spectral parameters which are very good comparable to what we obtain with the internal H.E.S.S. analysis tool and with gammapy:

```
                   | ctools: native       | ctools: Aeff without threshold check | H.E.S.S. analysis tool | gammapy
-------------------+----------------------+--------------------------------------+------------------------+----------------------
Norm [/TeV /s /cm2] | 2.236 +- 0.2421 e-11 | 1.816 +- 0.1906 e-11                 | 1.804 +- 0.2096 e-11   | 1.718 +- 0.1814 e-11
Index [1]          | 3.758 +- 0.265       | 3.431 +- 0.231                       | 3.464 +- 0.2389        | 3.422 +- 0.2096
```

**History**

**#1 - 09/07/2018 04:57 PM - Knödlseder Jürgen**

*- Assigned To set to Knödlseder Jürgen*

*- Target version set to 1.6.0*

The definition of the validity range of the effective area is a lurking and unsolved issue since some time.

I fully agree that energies below the reconstructed energy threshold should be taken into account. I infer from your message that you use the LO_THRES and HI_THRES keywords in the effective area, which in your case are reconstructed and not true energies. Is this correct?

**#2 - 09/10/2018 10:45 AM - Knödlseder Jürgen**

*- Status changed from New to In Progress*

*- % Done changed from 0 to 50*

It seems that it sufficient to not set the lower and upper energy threshold from the LO_THRES and HI_THRES keywords in the effective area, since the response class method GCTAResponseIrf::load_aeff extracts these keywords for usage in ctselect.

I therefore will remove the setting of the effective area boundaries from these keywords.

**#3 - 09/10/2018 10:50 AM - Mohrmann Lars**

I guess the problem is that the lower threshold we currently store in the effective area file is not strictly giving the validity range, but simply marks the (true) energy at which the energy bias exceeds a certain value. If we now want to apply an energy threshold in the analysis at a similar value, a problem occurs: two thresholds are applied when computing the predicted counts, leading to incorrect results.

In short: we'd need to access the effective area table below the energy threshold that is stored in the file. Is that possible/foreseen at all in ctools? Could an option for this be added?

P.S. Just saw your second comment. This might solve our problem.

**#4 - 09/10/2018 01:15 PM - Specovius Andreas**

user#3 wrote:

> I therefore will remove the setting of the effective area boundaries from these keywords.

m_logE_min will then be initialized only from the axis range no longer from hdu keys. This will indeed allow GCTAAeff2D::operator() to return nonzero values below threshold. I think this solves the issue.

**#5 - 09/10/2018 01:15 PM - Specovius Andreas**

I scrolled through the code and noticed in GCTAAeff2D::set_boundaries() that neng, emin and emax are initialized using the axis index m_inx_aeff. Shouldn't this be m_inx_energy?

**#6 - 09/10/2018 03:19 PM - Knödlseder Jürgen**

*- Status changed from In Progress to Feedback*

*- % Done changed from 50 to 100*

Code is merged into devel (change only in GammaLib). Could you please check whether the results are as you expect?

**#7 - 09/10/2018 03:21 PM - Knödlseder Jürgen**

user#291 wrote:

> I scrolled through the code and noticed in GCTAAeff2D::set_boundaries() that neng, emin and emax are initialized using the axis index m_inx_aeff. Shouldn't this be m_inx_energy?

Ooups. Indeed, this looks like a bug. Will fix.

**#8 - 09/10/2018 04:19 PM - Knödlseder Jürgen**

Bug is fixed and merged into devel.

I was wondering why the bug did not create any obvious problem. The reason is simple: for all IRF files that I have used so far m_inx_aeff=m_inx_energy=0.

I'm waiting for a final check from your side before closing the issue.

**#9 - 09/11/2018 08:55 AM - Specovius Andreas**

user#3 wrote:

> I was wondering why the bug did not create any obvious problem. The reason is simple: for all IRF files that I have used so far m_inx_aeff=m_inx_energy=0.

Same for us!

user#3 wrote:

> I'm waiting for a final check from your side before closing the issue.

As effective area now returns the expected values, from my side the issue is closed. Great!

**#10 - 09/11/2018 09:40 AM - Knödlseder Jürgen**

*- Status changed from Feedback to Closed*