ctools - Action #1297

Get ctbkgcube give bkgcube with a right unit

07/24/2014 06:47 PM - Lu Chia-Chun

Status:	Closed	**Start date:**	07/24/2014
Priority:	Normal	**Due date:**	
Assigned To:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	00-08-00		

Description

The summed bkgcube should be scaled by the total ontime to get the unit (1/MeV/Sq/s) right.

History

#1 - 07/24/2014 06:49 PM - Lu Chia-Chun

- Status changed from New to Pull request

- % Done changed from 0 to 90

Tested. Now ctlike gives reasonable normalization.

Pull request
https://github.com/chiachun/ctools/tree/1297-ctbkgcube-scaling

#2 - 07/24/2014 06:52 PM - Lu Chia-Chun

However, I am not sure it should be scaled by ontime or livetime.

#3 - 07/24/2014 06:52 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> Tested. Now ctlike gives reasonable normalization.
>
> Pull request
> https://github.com/chiachun/ctools/tree/1297-ctbkgcube-scaling

You were too quick. I was just writing the following biggrin.png

Indeed, the expected units are probably counts/MeV/sr/s, but they actually are counts/bin. I guess the code in ctbkgcube.cpp from line 391 on should instead of

```
        // Compute model value for event bin
        double model =
            m_bkgmdl.eval(*(const_cast<const GCTAEventBin*>(bin)), *obs) *
            bin->size();
```

read

```
        // Compute model value for event bin
        double model =
            m_bkgmdl.eval(*(const_cast<const GCTAEventBin*>(bin)), *obs);
```

Can you check if by removing the bin->size() multiplication you get things as expected?

#4 - 07/24/2014 06:55 PM - Knödlseder Jürgen

Just saw you code. You just divided by time, which leaves the issue about the division by MeV and sr. Have you checked that you get the right spectral normalization of your background model?

#5 - 07/24/2014 06:59 PM - Lu Chia-Chun

I didn't get the right norms so I planned to check my exporter. (I am not sure if my exporter gives output with right units...) So now I will try your scheme.

Knödlseder Jürgen wrote:

> Just saw you code. You just divided by time, which leaves the issue about the division by MeV and sr. Have you checked that you get the right spectral normalization of your background model?

#6 - 07/24/2014 07:11 PM - Lu Chia-Chun

after romoving bin->size()
I get norm ~ 0.003 (1/rad/rad)...WHY?

#7 - 07/24/2014 07:12 PM - Lu Chia-Chun

0.003 = 1/deg2rad/deg2rad
Lu Chia-Chun wrote:

> after romoving bin->size()
> I get norm ~ 0.003 (1/rad/rad)...WHY?

#8 - 07/25/2014 12:38 AM - Knödlseder Jürgen

- Status changed from Pull request to Resolved

- Target version set to 00-08-00

- % Done changed from 90 to 100

I did the change I indicated, and even more. In fact, as the background model is in rates (i.e. counts/MeV/s/sr), each observation (or run) has to be weighted by the livetime of each run. For two runs, for example, the background model is

model = (livetime1 * model1 + livetime2 * model2) / (livetime1 + livetime2)

This is now implemented.

Another piece that was missing was the cut of the background model beyond the RoI of a given run. This is important, otherwise you will get a predicted background rate in an area where there are no events due to an RoI cut. I also implemented this in ctbkgcube. I also implemented the same logic in ctbin, GCTAExposure and GCTAMeanPsf to be sure that all stacked cubes are compliant.

To test all this stuff I created a script make_cube_analysis.py that you will find in the ctools/examples folder. You may just go in this folder and type

```
./make_cube_analysis.py
```

to run the script. It will produce a bunch of files, and also log files. It basically will:

- create 4 half hour runs in a four-leaf clover pattern
- stack the events in a cube using ctbin
- compute the exposure and Psf cubes using ctexpcube and ctpsfcube
- compute a background model using ctbkgcube (I use here an analytical model so far)
- run a ctlike analysis on all the data

The source model that I used was the Crab (single point source).

Here for the record the output that I get:

```
2014-07-24T21:52:08: +==================================+
2014-07-24T21:52:08: | Maximum likelihood optimisation |
2014-07-24T21:52:08: +==================================+
2014-07-24T21:52:18:  >Iteration   0: -logL=70840.130, Lambda=1.0e-03
2014-07-24T21:52:28:  >Iteration   1: -logL=70828.547, Lambda=1.0e-03, delta=11.583, max(|grad|)=-20.298646 [Value:7]
2014-07-24T21:52:38:  >Iteration   2: -logL=70828.531, Lambda=1.0e-04, delta=0.016, max(|grad|)=0.040265 [Index:3]
2014-07-24T21:52:48:  >Iteration   3: -logL=70828.531, Lambda=1.0e-05, delta=0.000, max(|grad|)=0.000407 [Index:3]
2014-07-24T21:52:57:
2014-07-24T21:52:57: +=========================================+
2014-07-24T21:52:57: | Maximum likelihood optimization results |
2014-07-24T21:52:57: +=========================================+
2014-07-24T21:52:57: === GOptimizerLM ===
2014-07-24T21:52:57:  Optimized function value ..: 70828.531
2014-07-24T21:52:57:  Absolute precision .....: 1e-06
2014-07-24T21:52:57:  Optimization status .....: converged
2014-07-24T21:52:57:  Number of parameters ......: 9
2014-07-24T21:52:57:  Number of free parameters .: 3
2014-07-24T21:52:57:  Number of iterations ......: 3
2014-07-24T21:52:57:  Lambda .....: 1e-06
2014-07-24T21:52:57:
2014-07-24T21:52:57: +=========================================+
2014-07-24T21:52:57: | Maximum likelihood optimization results |
2014-07-24T21:52:57: +=========================================+
2014-07-24T21:52:57: === GOptimizerLM ===
2014-07-24T21:52:57:  Optimized function value ..: 70828.531
2014-07-24T21:52:57:  Absolute precision .....: 1e-06
2014-07-24T21:52:57:  Optimization status .....: converged
2014-07-24T21:52:57:  Number of parameters ......: 9
2014-07-24T21:52:57:  Number of free parameters .: 3
2014-07-24T21:52:57:  Number of iterations ......: 3
2014-07-24T21:52:57:  Lambda .....: 1e-06
2014-07-24T21:52:57:  Maximum log likelihood ....: -70828.531
2014-07-24T21:52:57:  Observed events  (Nobs) ...: 17666.000
2014-07-24T21:52:57:  Predicted events (Npred) ..: 17666.000 (Nobs - Npred = 5.97629e-07)
2014-07-24T21:52:57: === GModels ===
2014-07-24T21:52:57:  Number of models .....: 2
2014-07-24T21:52:57:  Number of parameters ......: 9
2014-07-24T21:52:57: === GModelSky ===
2014-07-24T21:52:57:  Name .....: Crab
2014-07-24T21:52:57:  Instruments .....: all
2014-07-24T21:52:57:  Instrument scale factors ..: unity
2014-07-24T21:52:57:  Observation identifiers ...: all
2014-07-24T21:52:57:  Model type .....: PointSource
2014-07-24T21:52:57:  Model components .....: "SkyDirFunction" * "PowerLaw" * "Constant"
2014-07-24T21:52:57:  Number of parameters ......: 6
2014-07-24T21:52:57:  Number of spatial par's ...: 2
2014-07-24T21:52:57:   RA .....: 83.6331 [-360,360] deg (fixed,scale=1)
2014-07-24T21:52:57:   DEC .....: 22.0145 [-90,90] deg (fixed,scale=1)
2014-07-24T21:52:57:  Number of spectral par's ..: 3
2014-07-24T21:52:57:   Prefactor .....: 5.74301e-16 +/- 1.55895e-17 [1e-23,1e-13] ph/cm2/s/MeV (free,scale=1e-16,gradient)
2014-07-24T21:52:57:   Index .....: -2.45404 +/- 0.0196043 [-0,-5]  (free,scale=-1,gradient)
2014-07-24T21:52:57:   PivotEnergy .....: 300000 [10000,1e+09] MeV (fixed,scale=1e+06,gradient)
2014-07-24T21:52:57:  Number of temporal par's ..: 1
2014-07-24T21:52:57:   Constant .....: 1 (relative value) (fixed,scale=1,gradient)
2014-07-24T21:52:57: === GCTAModelCubeBackground ===
2014-07-24T21:52:57:  Name .....:
2014-07-24T21:52:57:  Instruments .....: all
2014-07-24T21:52:57:  Instrument scale factors ..: unity
2014-07-24T21:52:57:  Observation identifiers ...: all
2014-07-24T21:52:57:  Model type .....: "MapCubeFunction" * "ConstantValue" * "Constant"
```

2014-07-24T21:52:57: Number of parameters ......: 3
2014-07-24T21:52:57: Number of spatial par's ...: 1
2014-07-24T21:52:57:   Normalization .....: 1 [0.001,1000] (fixed,scale=1,gradient)
2014-07-24T21:52:57: Number of spectral par's ..: 1
2014-07-24T21:52:57:   Value .....: 1.03709 +/- 0.00819384 [0,1000] (free,scale=1,gradient)
2014-07-24T21:52:57: Number of temporal par's ..: 1
2014-07-24T21:52:57:   Constant .....: 1 (relative value) (fixed,scale=1,gradient)

As you see, the fit converges nicely and quickly. Here a comparison between simulated and fitted values:

Parameter	Simulated	Fitted
Prefactor	5.7e-16	5.74301e-16 +/- 1.55895e-17
Index	-2.48	~~2.45404 +/~~ 0.0196043
Value	1.0	1.03709 +/- 0.00819384

Looks all pretty reasonable. The last row is the background normalization, which is basically one.

#9 - 07/25/2014 04:46 PM - Lu Chia-Chun

- Status changed from Resolved to Feedback

- % Done changed from 100 to 90

I get a cube filled with zero after updated because I used an obs.xml with eventlist not processed by ctselect (no roi). In this case, ctbkgcube should throw warning messages or exceptions.

2014-07-25T14:44:18: Bins within RoI .....: 0
2014-07-25T14:44:18: Bins outside RoI .....: 6000
2014-07-25T14:44:18: Background events in cube .: 0
2014-07-25T14:44:18: Cube livetime .....: 1604.94 sec

#10 - 07/25/2014 05:22 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> I get a cube filled with zero after updated because I used an obs.xml with eventlist not processed by ctselect (no roi). In this case, ctbkgcube should throw warning messages or exceptions.

2014-07-25T14:44:18: Bins within RoI .....: 0
2014-07-25T14:44:18: Bins outside RoI .....: 6000
2014-07-25T14:44:18: Background events in cube .: 0
2014-07-25T14:44:18: Cube livetime .....: 1604.94 sec

I agree. I created issue #1298.

#11 - 07/25/2014 05:32 PM - Lu Chia-Chun

I am not sure whether this is a good idea.
It can happen that people want to have frequent changes on ROI for testing. It doesn't make sense to produce new cubes every time when people change ROI, especially for GCTAMeanPsf, which is computationally very expensive. We hope that we only have to calculate it once.

Besides, it happens also very frequently that one wants to change exclusion regions -- there are always regions/sources which we don't know how to do a good modelling. For HESS, whose PSF is at the order of 0.1 deg, we don't have to model all sources in the fov. We can cut out some regions to have a clean map to do modelling on targets in interests.

Does ctbkgcube also exclude pixels in exclusion regions? I don't understand why this is needed since ctlike ignores pixels with negative values in event cube.

I guess it's about background modelling normalization? Then could we proceed as follows?

- make background cube with a fov larger than the ROI.
- ctlike tells GModel that some pixels are switched off.
- GModel re-normalize bkgcube.

Knödlseder Jürgen wrote:

> Another piece that was missing was the cut of the background model beyond the RoI of a given run. This is important, otherwise you will get a predicted background rate in an area where there are no events due to an RoI cut. I also implemented this in ctbkgcube. I also implemented the same logic in ctbin, GCTAExposure and GCTAMeanPsf to be sure that all stacked cubes are compliant.

#12 - 07/26/2014 09:58 AM - Lu Chia-Chun

the size of GEventBin is solidangle() * ewidth().MeV() * ontime();
So, shouldn't we weight the model value by ontime?

Knödlseder Jürgen wrote:

> I did the change I indicated, and even more. In fact, as the background model is in rates (i.e. counts/MeV/s/sr), each observation (or run) has to be weighted by the livetime of each run. For two runs, for example, the background model is

#13 - 07/26/2014 10:10 AM - Lu Chia-Chun

Anyway, I modified my exporter to scale the background by ontime.
Now I get normalization 1. It doesn't matter we use ontime of livetime. we just have to keep it consistent.

#14 - 07/26/2014 11:46 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> I am not sure whether this is a good idea.
> It can happen that people want to have frequent changes on ROI for testing. It doesn't make sense to produce new cubes every time when people change ROI, especially for GCTAMeanPsf, which is computationally very expensive. We hope that we only have to calculate it once.

Does ctpsfcube take a long time for you? I can't even see that it's running, I get results basically immediately.

If you change the RoI, you have to reprocess the event list anyways (using ctselect). This takes more time than the response stuff ...

> Besides, it happens also very frequently that one wants to change exclusion regions -- there are always regions/sources which we don't know how to do a good modelling. For HESS, whose PSF is at the order of 0.1 deg, we don't have to model all sources in the fov. We can cut out some regions to have a clean map to do modelling on targets in interests.
>
> Does ctbkgcube also exclude pixels in exclusion regions? I don't understand why this is needed since ctlike ignores pixels with negative values in event cube.

You do not need exclusion regions for ctbkgcube.

> I guess it's about background modelling normalization? Then could we proceed as follows?
>
> - make background cube with a fov larger than the ROI.
> - ctlike tells GModel that some pixels are switched off.
> - GModel re-normalize bkgcube.

No, it's also (and mainly) about exposure computation. You should not have exposure added into an exposure cube from a region which has been excluded from the event selection. Exposure cube generation needs to know what your event selection region was for each run. The same is true for Psf cube, and also background estimates.

#15 - 07/26/2014 11:49 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> the size of GEventBin is solidangle() * ewidth().MeV() * ontime();
> So, shouldn't we weight the model value by ontime?

It really depends how we define background rates. What is the H.E.S.S. definition? Number of events per livetime or per ontime?

Event bin size is defined per ontime, as the Irf includes the livetime correction.

#16 - 07/26/2014 11:58 PM - Lu Chia-Chun

We have never used absolute background rate in HESS. We can choose the one we like.

Knödlseder Jürgen wrote:

> Lu Chia-Chun wrote:
>
> > the size of GEventBin is solidangle() * ewidth().MeV() * ontime();
> > So, shouldn't we weight the model value by ontime?
>
> It really depends how we define background rates. What is the H.E.S.S. definition? Number of events per livetime or per ontime?
>
> Event bin size is defined per ontime, as the Irf includes the livetime correction.

#17 - 07/27/2014 12:12 AM - Lu Chia-Chun

About PSF cube calculation: It's slow for an observation of several hundreds of runs. Probably I used too many spacial bins. I will check it again tomorrow.

About exposure computation: Now I am confused. We select events in an ROI in sky coordinate, not in instrument coordinate. The ROI is thus always the same for each run and the exposure cube, counts cube, every cube. It's safe to just add all exposure up since the ROI never changes position, Isn't it?

And, ctselect is a tool for unbinned analysis, why do we use it for cube analysis (always a binned analysis)? We should use ctcubemask, which sets pixels outside ROI -1. Then it doesn't matter whether the exposure there is zero or not.

(Too tired, have to go to bed. I'll come back tomorrow morning.)

Knödlseder Jürgen wrote:

> No, it's also (and mainly) about exposure computation. You should not have exposure added into an exposure cube from a region which has been excluded from the event selection. Exposure cube generation needs to know what your event selection region was for each run. The same is true for Psf cube, and also background estimates.

#18 - 07/28/2014 12:26 AM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> About PSF cube calculation: It's slow for an observation of several hundreds of runs. Probably I used too many spacial bins. I will check it again tomorrow.
>
> About exposure computation: Now I am confused. We select events in an ROI in sky coordinate, not in instrument coordinate. The ROI is thus always the same for each run and the exposure cube, counts cube, every cube. It's safe to just add all exposure up since the ROI never changes position, Isn't it?

We probably call different things ROI. For GammaLib, ROI is a circular region around a pointing, typically a few degrees, from which events will be used. That's just the zone of selected events. In a stacked cube analysis, you may want to define a cube that is larger than the ROI of an individual run. Hence for one run, there will be zones in the cube that will not be filled. It is then important that in these zones there is also no exposure and no background.

> And, ctselect is a tool for unbinned analysis, why do we use it for cube analysis (always a binned analysis)? We should use ctcubemask, which sets pixels outside ROI -1. Then it doesn't matter whether the exposure there is zero or not.

See above. It's an issue when your cube is larger than the FoV of a single run. Making pixels does not help here.

#19 - 11/11/2014 11:54 PM - Knödlseder Jürgen

- Status changed from Feedback to Closed

- % Done changed from 90 to 100

- Remaining (hours) set to 0.0