ctools - Action #1268

Create ctbackcube tool

07/11/2014 02:14 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	07/11/2014
Priority:	Normal	**Due date:**	
Assigned To:	Lu Chia-Chun	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	3rd coding sprint		

Description

This tool should create an background cube, analogous to the ctexpcube tool for average exposure cubes.

History

#1 - 07/11/2014 09:08 PM - Lu Chia-Chun

- Status changed from New to Feedback

As what we discussed, ctmodel should do the work.

#2 - 07/12/2014 10:34 PM - Knödlseder Jürgen

- Status changed from Feedback to Rejected

- Remaining (hours) set to 0.0

Indeed, so we reject this action.

#3 - 07/21/2014 04:58 PM - Knödlseder Jürgen

- Status changed from Rejected to New

- Estimated time set to 0.00

Following some discussion (that I reproduce here), we decided that we should implement ctbkgcube:

Chia-Chun:

> Dear Juergen,
>
> I decide to write a new tool, ctbkgcube, to merge background models since I don't know how to do it without changing the currect ctmodel stucture.
>
> The philosophy is similar to ctexpcube ctpsfcube so easier for me to implement it.
>
> Cheers,
> Chia-Chun

Jürgen:

> Ok, fine for me. I was in fact also re-considering this, as ctmodel is also for source models, and there it won't use the expcube and psfcube structures but just the original IRFs.
>
> I was wondering whether there are background specific things in ctbkgcube, or could it also be a ctmodcube (for "model") that can also produce cubes based on source models?
>
> The question then is how ctmodel and ctmodcube differ in the end ...
>
> But maybe just go ahead with ctbkgcube. We can always re-structure things when they become clearer in the future.
>
> I was also thinking about adding a support module to ctools, as we have now the same bunch of code in several tools (for example the routine to

get the energy boundaries from user parameters). So I was anyways planning to re-organize things later ...

Cheers,
Jürgen

Chia-Chun:

I almost finished.

The difference I see:

- difficulty in I/O, for ctmodel user either give an obs or a cntmap, but for ctbkgcube we need both, and we usually want different binning of cntmap. We have to make a lot of conditional loop if we merge ctmodel and cntmap.

- ctbkgcube gives skymap as output and ctmodel gives eventcube.

Jürgen:

What do you mean here? ctbkgcube will give a 2D skymap, or a 3D map (skymap as function of energy)?

Chia-Chun:

I can't finish ctbkgcube because I don't know how to handle GCTAEventBin, which is needed for model evaluation. I intended to use GCTAPointing to do coordinate transformation to converge a skymap pixel to a GCTAEventBin, but there is no way to set GCTAEventBin content...

My half-finished code is pushed
https://github.com/chiachun/ctools/tree/ctbkgcube

I can finish and test it if you tell me how to do it. Thanks!

BTW, line 352

bin->counts(bin->counts() + model);
should be map(pixel,iebin) =map(pixel,iebin) + model

I forgot to change it.

#4 - 07/21/2014 05:01 PM - Knödlseder Jürgen

I got your tree. Just one comment (I've seen the same yesterday with ctpsfcube): don't put the binary under version control of git. The binary will be produce from the source code, but should not be part of the repo (as it is platform dependent).

#5 - 07/21/2014 05:05 PM - Lu Chia-Chun

sorry. It's a mistake. sad.png

Knödlseder Jürgen wrote:

> I got your tree. Just one comment (I've seen the same yesterday with ctpsfcube): don't put the binary under version control of git. The binary will be produce from the source code, but should not be part of the repo (as it is platform dependent).

#6 - 07/21/2014 05:06 PM - Lu Chia-Chun

> Chia-Chun: ctbkgcube gives skymap as output and ctmodel > gives eventcube.
> Juergen: What do you mean here? ctbkgcube will give a 2D > skymap, or a 3D map (skymap as function of energy)?

Chia-Chun: My understanding is that EventCubes have more information such as GTI. It doesn't make sense to have this for a model cube. Combing ctmodel and ctbkgcube together is possible, but we have to add several condition loops, which make the code hard to read.

#7 - 07/21/2014 05:09 PM - Knödlseder Jürgen

Going back to your question of how to implement things, here what I would do:

- replicate the code ctbin::init_cube to initialise a GSkymap object
- use code like that found in ctbin::cube() to create a GCTAEventCube object from the GSkymap object
- now loop over the event bins, like in ctmodel::model_map to evaluate the model and set the model value
- then save the GCTAEventCube (you may simply call the GCTAEventCube.save() method)

Does this sound reasonable?

#8 - 07/21/2014 05:10 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> My understanding is that EventCubes have more information such as GTI. It doesn't make sense to have this for a model cube. Combing ctmodel and ctbkgcube together is possible, but we have to add several condition loops, which make the code hard to read.

Indeed, they have GTIs, but does it harm to have this information in a FITS file? Construction would be very similar compared to ctbin::fill_cube (you use m_gti.extend(events->gti()); to append the GTIs of a specific observation to the list. That just costs you a single line of code.

You would then need to update the GTIs in the event cube at the end (as you won't have this information upon the initial construction, unless you want a separate loop over all observations to get this information).

#9 - 07/21/2014 05:27 PM - Lu Chia-Chun

> Indeed, they have GTIs, but does it harm to have this information in a FITS file? Construction

I don't object, but if we really want to have GTI in background cube, we'd better make the files consistent. It's strange that background cubes without ctmodel processing don't have GTI while those after ctmodel processing do.

#10 - 07/21/2014 05:38 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> > Indeed, they have GTIs, but does it harm to have this information in a FITS file? Construction
>
> I don't object, but if we really want to have GTI in background cube, we'd better make the files consistent. It's strange that background cubes without ctmodel processing don't have GTI while those after ctmodel processing do.

Now I'm lost. I though we won't use ctmodel anymore for background cube processing, and that you create ctbkgmodel specifically for writing background cubes.

In any case, it won't be difficult either to just save the GSkymap object, and not the entire GCTAEventCube object. The logic would still be the same. You would create a GCTAEventCube first, loop over it's bins, but then just save the sky map (either the one in the cube, or another one that you create outside; but the latter would double the memory requirements, hence you'd like to use the GCTAEventCube skymap ...)

#11 - 07/21/2014 07:42 PM - Lu Chia-Chun

- I vote for using ctbackcube because I don't like multi-condition loops.
- The steps I take in ctbackcube is

1. create a skymap with user-defined bins
2. go to each pixel and calculate the model value for that position.

The difficulty I have is:
GModels takes GCTAEventBin as an argument.
skymap is in the sky coordinate but GCTAEventBin has to be in instrument coordinate.
I want to use GCTAPointing to transform skymap sky direction to instrument direction and then create a GCTAEventBin by the instrument coordinate, but I don't know how to do it. GCTAEventBin doesn't allow setting directions.

Another possibility is to write a new GCTAEventCube constructor,
GCTAEventCube::GCTAEventCube(skymap, ebound, gti, pointing) and do coordinate transformation in GCTAEventCube.

#12 - 07/21/2014 08:18 PM - Knödlseder Jürgen

As said above, the way how to implement this is the following:

- replicate the code ctbin::init_cube to initialise a GSkymap object
- use code like that found in ctbin::cube() to create a GCTAEventCube object from the GSkymap object
- now loop over the event bins, like in ctmodel::model_map to evaluate the model and set the model value
- then save the GCTAEventCube (you may simply call the GCTAEventCube.save() method)

So you create a GCTAEventCube object which then allows you looping over the events, without any need to pass through pointings, etc. If you're stuck with that, just commit the latest code and I can take over implementing this part.

#13 - 07/22/2014 06:40 PM - Lu Chia-Chun

I've used GCTAEventCube to do the loop similar to model_map(). Now the loop seems to do model evaluation right, but the saved event cube or skymap is still empty. Don't know what's wrong...
My newest code is here:

https://github.com/chiachun/ctools/tree/ctbkgcube-2

Thanks!

#14 - 07/22/2014 06:55 PM - Knödlseder Jürgen

I'll give it a look later ...

#15 - 07/22/2014 10:40 PM - Knödlseder Jürgen

- Status changed from New to Resolved

- % Done changed from 0 to 100

Here you are. Your tool is in devel and working.

#16 - 07/22/2014 10:55 PM - Lu Chia-Chun

Thank you ! biggrin.png
I am checking the code to find out what I did wrong...
Knödlseder Jürgen wrote:

> Here you are. Your tool is in devel and working.

#17 - 07/22/2014 11:06 PM - Lu Chia-Chun

It looks like I missed this line....
m_bkgcube.gti(obs->events()->gti());

Is this the reason why my code didn't work?
And I am surprised to find that GCTAEventCube doesn't have to be a member of a specific observation!

#18 - 07/22/2014 11:37 PM - Knödlseder Jürgen

Lu Chia-Chun wrote:

> It looks like I missed this line....
> m_bkgcube.gti(obs->events()->gti());
>
> Is this the reason why my code didn't work?

Possibly, but not sure. I don't think that the times are used at the end.

> And I am surprised to find that GCTAEventCube doesn't have to be a member of a specific observation!

In principle, every GammaLib class can live alone. Even GCTAEventBin!

#19 - 07/25/2014 01:29 AM - Knödlseder Jürgen

- Status changed from Resolved to Closed