ctools - Change request #1566

Make csspec behavior consistent with parameter interface

11/03/2015 10:53 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	11/03/2015
Priority:	High	**Due date:**	
Assigned To:	Mayer Michael	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

While writing the csspec user documentation, I think I discovered an inconsistency between the script and the parameter interface. From the parameter interface, I would have guessed that I can either provide an event list, a counts cube or an observation definition file. Specifically, for the counts cube I would have expected from the expcube, psfcube, and bkgcube parameters that the script works also on binned data.

However, I see that script toggles between unbinned and binned analysis using the binned parameter, hence in principle I could provide a counts cube and specify unbinned analysis. This obviously will not work as in unbinned analysis the ctselect tools in invoked.

My proposal would be to have csspec either work on event lists or binned data, and to remove the internal binning from the script (this should be done externally as this provides more flexibility; for example, regions of a counts cube could be masked out). The script would then not need the binned, nebins, coordsys, proj, xref, yref, nxpix, nypix, binsze and anumbins parameters as no internal binning will be done.

The only thing that has to be decided would then be how to select the energy binning. One could imagine that the script will always use the binning of the counts cube, and only use a multiplicity factor derived from the enumbins parameter (use for example always 2 of the energy bins, or always 3). This would keep the possibility of the user influencing the binning, but he could not specify the exact number of energy bins. Also emin and emax could be used as (loose) thresholds.

History

#1 - 11/04/2015 03:21 PM - Mayer Michael

You are right, there is an inconsistency in the parameters. I recall when writing the code, I wanted to implement that an exception is thrown if a binned observation is given. Don't know why but apparently I forgot this.
Your change request sounds great, but I am still a bit uncertain, how to achieve an internal binning from a binned observation.
I just want to outline the different needed use cases from my point of view.

1. Provide unbinned observations and do an unbinned analyses in each energy bin (as is working now)
2. Provide an unbinned observation and run binned analysis in each energy bin (as is working now). This might be important if the unbinned analysis takes much too long (even in a small energy bin)
3. Provide a binned observation: I am not sure how to run an analysis in this case as an internal rebinning is not possible (only by merging bins, is that what you mean?). But if we assume a spectral shape in each energy bin, we would probably need more than 1 energy bin per spectrum point, right?

If I understand correctly, your plan is to remove 2) and implement 3) instead, right? I agree this would very much simplify the parameter interface. The only concern I have is that 2) will be important for large data sets (e.g. in Fermi-LAT this is done quite frequently).

#2 - 11/04/2015 03:31 PM - Knödlseder Jürgen

Mayer Michael wrote:

> You are right, there is an inconsistency in the parameters. I recall when writing the code, I wanted to implement that an exception is thrown if a binned observation is given. Don't know why but apparently I forgot this.
> Your change request sounds great, but I am still a bit uncertain, how to achieve an internal binning from a binned observation.
> I just want to outline the different needed use cases from my point of view.
>
> 1. Provide unbinned observations and do an unbinned analyses in each energy bin (as is working now)
> 2. Provide an unbinned observation and run binned analysis in each energy bin (as is working now). This might be important if the unbinned analysis takes much too long (even in a small energy bin)
> 3. Provide a binned observation: I am not sure how to run an analysis in this case as an internal rebinning is not possible (only by merging bins, is that what you mean?). But if we assume a spectral shape in each energy bin, we would probably need more than 1 energy bin per spectrum point, right?

I won't merge bins. You can blank out bins from a counts cube by setting their values to negative values (see what ctcubemask does). So we can walk over all layers of a cube, blank out all layers other then the selected layer, and do the fit.

One could generalize this to using always 2 layers, 3 layers, etc. dependent on what is provided by the enumbins parameter.

> If I understand correctly, your plan is to remove 2) and implement 3) instead, right? I agree this would very much simplify the parameter interface. The only concern I have is that 2) will be important for large data sets (e.g. in Fermi-LAT this is done quite frequently).

Yes, as 3 gives you more control than 2 (in 2 things are done internally, hence you cannot for example blank out specific regions in a counts cube).

#3 - 11/04/2015 03:32 PM - Knödlseder Jürgen

Knödlseder Jürgen wrote:

> Mayer Michael wrote:
>
> > You are right, there is an inconsistency in the parameters. I recall when writing the code, I wanted to implement that an exception is thrown if a binned observation is given. Don't know why but apparently I forgot this.
> > Your change request sounds great, but I am still a bit uncertain, how to achieve an internal binning from a binned observation.
> > I just want to outline the different needed use cases from my point of view.
> >
> > 1. Provide unbinned observations and do an unbinned analyses in each energy bin (as is working now)
> > 2. Provide an unbinned observation and run binned analysis in each energy bin (as is working now). This might be important if the unbinned analysis takes much too long (even in a small energy bin)
> > 3. Provide a binned observation: I am not sure how to run an analysis in this case as an internal rebinning is not possible (only by merging bins, is that what you mean?). But if we assume a spectral shape in each energy bin, we would probably need more than 1 energy bin per spectrum point, right?

> I won't merge bins. You can blank out bins from a counts cube by setting their values to negative values (see what ctcubemask does). So we can walk over all layers of a cube, blank out all layers other then the selected layer, and do the fit.

But I admit that this is probably time consuming, hence we could also think about extracting bins.

> One could generalize this to using always 2 layers, 3 layers, etc. dependent on what is provided by the enumbins parameter.

> > If I understand correctly, your plan is to remove 2) and implement 3) instead, right? I agree this would very much simplify the parameter interface. The only concern I have is that 2) will be important for large data sets (e.g. in Fermi-LAT this is done quite frequently).

> Yes, as 3 gives you more control than 2 (in 2 things are done internally, hence you cannot for example blank out specific regions in a counts cube).

#4 - 11/04/2015 04:39 PM - Mayer Michael

> I won't merge bins. You can blank out bins from a counts cube by setting their values to negative values (see what ctcubemask does). So we can walk over all layers of a cube, blank out all layers other then the selected layer, and do the fit.

Now I understand. I wasn't aware ctcubemask can also make an energy selection (I thought it works only spatial.

> But I admit that this is probably time consuming, hence we could also think about extracting bins.

One would need to check how long ctcubemask runs on a counts cube (but I guess it is not longer than ctselect on an observation definition file with a lot of event lists)

> One could generalize this to using always 2 layers, 3 layers, etc. dependent on what is provided by the enumbins parameter.

Ok, so if the user provides a counts cube with 20 energy bins and wants to create 25 spectral points, we would run into troubles of course. On the other hand, 20 energy bins would only allow for 20 (or 10 or 5) spectral points without losing information, right? We would have to think how to handle cases like 20 energy bins in the cube and the user wants to use 3 layers (do we create 6 spectral points and omit the 2 last energy bins in the cube?).

Nevertheless, I am a big fan of making the spectral points binning dependent on the binning of the input counts cube.

#5 - 11/04/2015 06:41 PM - Knödlseder Jürgen

Mayer Michael wrote:

> One could generalize this to using always 2 layers, 3 layers, etc. dependent on what is provided by the enumbins parameter.

> Ok, so if the user provides a counts cube with 20 energy bins and wants to create 25 spectral points, we would run into troubles of course. On the other hand, 20 energy bins would only allow for 20 (or 10 or 5) spectral points without losing information, right? We would have to think how to handle cases like 20 energy bins in the cube and the user wants to use 3 layers (do we create 6 spectral points and omit the 2 last energy bins in the cube?).

Right, the translation can only be done approximate. The finest binning is driven by the binning of the counts cube, the number of layers used will be computed using

int n_layers = n_bins_in_counts_cube / enumbins;

hence until enumbins is half of n_bins_in_counts_cube the n_layers stays one.

You are right that we have to define the behavior for the case where the division has a remainder. I probably would just cut off the thing at the end (i.e. use for example only 1 bin even if the division above would suggest 3).

#6 - 11/05/2015 09:45 AM - Mayer Michael

Ok that sounds reasonable. Accordingly, if the user really wants a finer binning, the data products (countcube, psf, bkg, exp) need to be precomputed with a finer energy binning, too. This makes sense to me.

#7 - 11/06/2015 02:15 PM - Mayer Michael

I have implemented the change you proposed on branch *1566-csspec-update-for-binned-mode*.

A slight modification to ctcubemask was necessary to allow for undefined parameters, i.e. making only an energy selection and no RoI selection. This is analogous to ctselect.

I also created a test case for the binned mode. Note that I had to change the header of test/test_cscripts.sh.in: it now only deletes .fits files that were created from the cscripts. The reason is that binned csspec needs input cubes that are not available unless the ctools tests were run beforehand. We might think of permanently adding such cubes to $CTOOLS/test/data?

#8 - 11/06/2015 02:15 PM - Mayer Michael

- Status changed from New to Pull request

- % Done changed from 0 to 100

#9 - 11/06/2015 02:15 PM - Mayer Michael

- Assigned To set to Mayer Michael

#10 - 11/06/2015 02:49 PM - Knödlseder Jürgen

Mayer Michael wrote:

> I have implemented the change you proposed on branch *1566-csspec-update-for-binned-mode*.
>
> A slight modification to ctcubemask was necessary to allow for undefined parameters, i.e. making only an energy selection and no RoI selection. This is analogous to ctselect.
>
> I also created a test case for the binned mode. Note that I had to change the header of test/test_cscripts.sh.in: it now only deletes .fits files that were created from the cscripts. The reason is that binned csspec needs input cubes that are not available unless the ctools tests were run beforehand. We might think of permanently adding such cubes to $CTOOLS/test/data?

Indeed, I'd prefer to add test data as otherwise one test depends on another and when something breaks we don't know who is the actual culprit.

#11 - 11/06/2015 02:53 PM - Mayer Michael

> Indeed, I'd prefer to add test data as otherwise one test depends on another and when something breaks we don't know who is the actual culprit.

We would need a counts cube, exposure cube, psf cube and background cube and a corresponding model XML file. Ideally the cubes would have a rather coarse binning to speed up the unit tests.

#12 - 11/06/2015 03:03 PM - Knödlseder Jürgen

Mayer Michael wrote:

> > Indeed, I'd prefer to add test data as otherwise one test depends on another and when something breaks we don't know who is the actual culprit.
>
> We would need a counts cube, exposure cube, psf cube and background cube and a corresponding model XML file. Ideally the cubes would have a rather coarse binning to speed up the unit tests.

There are these cubes in gammalib/inst/cta/test/data but I have not checked if the cubes are adapted. Also the exposure cube is a bit large (I probably should try to reduce the size). We should try to make cubes as small as possible as we only want to test the functionality.

#13 - 11/06/2015 03:08 PM - Mayer Michael

The cubes which come out of the unit tests are reasonable in size. After running test_ctools.sh there are files called cntcube2.fits, expcube2.fits, psfcube2.fits, bkgcube2.fits, bkgcube2.xml. The cubes are in the 60KB region and seem to be suitable.

#14 - 11/06/2015 03:15 PM - Knödlseder Jürgen

Mayer Michael wrote:

> The cubes which come out of the unit tests are reasonable in size. After running test_ctools.sh there are files called cntcube2.fits, expcube2.fits, psfcube2.fits, bkgcube2.fits, bkgcube2.xml. The cubes are in the 60KB region and seem to be suitable.

Indeed. I propose to rename and gzip the files necessary and add them to the test/data folder.

#15 - 11/06/2015 03:28 PM - Mayer Michael

I've added the files to the branch *1566-csspec-update-for-binned-mode*

#16 - 11/06/2015 04:07 PM - Knödlseder Jürgen

- Status changed from Pull request to Closed

Merged into devel.

Also updated the reference documentation.

#17 - 11/06/2015 04:12 PM - Mayer Michael

Great.
For the moment, we should also note in the documentation that the Npred column will not be filled for binned analyses. The reason is that GObservation::npred (which calls GCTAResponsCube::nroi) is not implemented for binned CTA observations.

#18 - 11/06/2015 04:15 PM - Knödlseder Jürgen

Mayer Michael wrote:

> Great.
> For the moment, we should also note in the documentation that the Npred column will not be filled for binned analyses. The reason is that GObservation::npread (which calls GCTAResponsCube::nroi) is not implemented for binned CTA observations.

Indeed, I have not really changed the text but removed the unused parameters. Do not hesitate to also update the text. User documentation is as important as code!

#19 - 11/06/2015 04:26 PM - Mayer Michael

I added some sentences to the doc.
I have pushed it to the above branch.

#20 - 11/07/2015 12:19 AM - Knödlseder Jürgen

Merged into devel.