ctools - Feature #1335

Write csresmap

10/13/2014 03:53 PM - Mayer Michael

Status:	Closed	**Start date:**	10/13/2014
Priority:	Normal	**Due date:**	
Assigned To:	Mayer Michael	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	00-08-00		

Description

To quickly produce residual count maps, we could have a cscript that does the job for you (following #1330). I am not so sure about the naming csresmap yet.

Related issues:

Related to ctools - Bug # 1337: The SWIG interface seems to have trouble with...	**New**	**10/15/2014**

History

#1 - 10/13/2014 04:03 PM - Mayer Michael

I wrote a first version of csresmap. It can be found on branch *1335-write-csresmap*. The script support three algorithms to compute the residuals:

1. "SUB" - subtracts the model map from the data map
2. "SUBDIV" - subtracts the model map from the data map and divides the result by the model map
3. "SUBDIVSQRT" - subtracts the model map from the data map and divides the result by the square root of the model map

I did a quick test by passing an optimised GObservations-object, which worked fine.

#2 - 10/13/2014 11:54 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- % Done changed from 0 to 80

Thanks. I merged this into the devel branch.

Just one thing. I think it's better practice to use import ctools and import gammalib instead of from gammalib import *, etc. Don't recall why this is the case, but I move now writing new code with the import gammalib directive. Could you also update the code accordingly?

Also, could you add a short test case to test/test_cscripts.sh.in?

#3 - 10/14/2014 05:36 PM - Mayer Michael

- Status changed from In Progress to Feedback

- % Done changed from 80 to 100

Import style changed and test case added to test/test_cscripts.sh.in (also on branch *1335-write-csresmap*)

#4 - 10/15/2014 07:17 AM - Knödlseder Jürgen

- Target version set to 00-08-00

Thanks for doing the changes.

I went over the code and recognized that you had to save the intermediate results to disk before you could compute the residual map. To have the same logic as in the ctools (i.e. the run() method works in memory while the execute() writes the result to disk), I added a cube() set method to ctmodel and changed the script to avoid intermediate results stored to disk. I had a little trouble implementing this as the code

```
map = bin.cube().map()
```

lead to an empty sky map while

```
cube = bin.cube()
map = cube.map().copy()
```

gave the correctly filled skymap. This seems to be related to the memory management of the SWIG interface (e.g., without the copy() there was a segmentation fault). Interestingly, for getting the model map I could directly write

```
map = model.cube().map()
```

The difference on the ctools level is that for ctmodel, the cube() method returns a reference to an object that is a member of ctmodel while for ctbin, the cube() method returns an object by value that is constructed on-the-fly. Making the cube object a member of ctbin will probably solve the issue. So there seems to be an issue with the SWIG interface. I opened an issue you track this problem (#1337).

Nevertheless, I integrated csresmap in the devel branch.

#5 - 10/17/2014 03:21 PM - Mayer Michael

I also sometimes stumbled upon this SWIG problem in the past (my fault I did not report). Returning just a copy of the objects was the easiest workaround in most cases. Good to open a new issue on that.

Thanks for integrating csresmap.
I realised that we can also easily derive an excess map (which is very common in VHE astronomy) with that tool. We just have to use the algorithm "SUB" and remove all GModelSky instances from the model container before. However, this feature is currently quite hidden and I was wondering how we could advertise this possibility to the user.
In addition, do we need any other algorithms to be implemented?

#6 - 10/17/2014 03:45 PM - Knödlseder Jürgen

Mayer Michael wrote:

> I also sometimes stumbled upon this SWIG problem in the past (my fault I did not report). Returning just a copy of the objects was the easiest workaround in most cases. Good to open a new issue on that.
>
> Thanks for integrating csresmap.
> I realised that we can also easily derive an excess map (which is very common in VHE astronomy) with that tool. We just have to use the algorithm "SUB" and remove all GModelSky instances from the model container before. However, this feature is currently quite hidden and I was wondering how we could advertise this possibility to the user.

Interesting idea. For the moment I would tend to create a new script for that although this duplicates some code. By the way: I added recently a ctool base class to ctools which is in the src/support folder and which provides common features to all tools. This allowed to reduce the code duplication in the existing ctools. They now all derive from ctool.

We should do the same for the cscripts, but I had not yet the time to look into that. We can also put common script code into a module to reduce code duplication, if necessary. This may become more and more important as the code base is growing.

> In addition, do we need any other algorithms to be implemented?

csresmap shows the spatial residuals, but we may also want to see the spectral residuals.

#7 - 10/17/2014 07:02 PM - Mayer Michael

> Interesting idea. For the moment I would tend to create a new script for that although this duplicates some code. By the way: I added recently a ctool base class to ctools which is in the src/support folder and which provides common features to all tools. This allowed to reduce the code duplication in the existing ctools. They now all derive from ctool.

Yes, I saw that and like the idea of the ctool base class. It especially helps with handling/loading the observations and models.

> We should do the same for the cscripts, but I had not yet the time to look into that. We can also put common script code into a module to reduce code duplication, if necessary. This may become more and more important as the code base is growing.

I might check the available cscripts sometime the next week to find out what code could be shared. Having a general cscript, as well sounds like a good idea.

> csresmap shows the spatial residuals, but we may also want to see the spectral residuals.

I agree. Wouldn't such functionality however be better placed in the spectrum code in obsutils? Since spectrum points are probably a commonly requested feature, we might also think about putting their computation into a cscript - csspecpoints?

#8 - 10/17/2014 08:16 PM - Knödlseder Jürgen

Mayer Michael wrote:

> > Interesting idea. For the moment I would tend to create a new script for that although this duplicates some code. By the way: I added recently a ctool base class to ctools which is in the src/support folder and which provides common features to all tools. This allowed to reduce the code duplication in the existing ctools. They now all derive from ctool.
>
> Yes, I saw that and like the idea of the ctool base class. It especially helps with handling/loading the observations and models.
>
> > We should do the same for the cscripts, but I had not yet the time to look into that. We can also put common script code into a module to reduce code duplication, if necessary. This may become more and more important as the code base is growing.
>
> I might check the available cscripts sometime the next week to find out what code could be shared. Having a general cscript, as well sounds like a good idea.

Indeed. We may see whether cscript should derive from ctool or directly from GBase. If possible, ctool would probably appropriate. The ctool base class has in fact a Python interface hence we may use it directly in cscripts base class. We could even add Python specific stuff in the interface if needed, which then would avoid to create a specific cscript class.

> > csresmap shows the spatial residuals, but we may also want to see the spectral residuals.
>
> I agree. Wouldn't such functionality however be better placed in the spectrum code in obsutils? Since spectrum points are probably a commonly requested feature, we might also think about putting their computation into a cscript - csspecpoints?

What about csspectrum? I would argue that having points is implicit for a spectrum.

#9 - 11/10/2014 10:24 AM - Mayer Michael

I agree very much to add csspectrum. I guess before that we might need to add ctupperlimit (see also #1263), which can be called for every energy bin (or only if the TS is less than a certain value).

#10 - 11/11/2014 11:53 PM - Knödlseder Jürgen

- Status changed from Feedback to Closed