ctools - Feature #1364

Write csspec

11/12/2014 11:17 AM - Mayer Michael

Status:	Closed	**Start date:**	11/12/2014
Priority:	Normal	**Due date:**	
Assigned To:	Mayer Michael	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.0.0		

Description

To calculate spectral points, the cscript csspec needs to be written. For a user-defined energy binning the tool should be able to fit the spectrum normalisation in each bin and derive data points from the fit. In every bin, the TS value and the upper limit (see #1363) should be computed and stored.
We still might discuss the structure of the file we store (ascii, FITS or XML)?

History

#1 - 11/12/2014 04:01 PM - Mayer Michael

I have pushed a first version of csspec to branch *1364-write-csspec*. Currently, only unbinned analysis is supported (for binned some more thinking is required how to handle the cube energy binning in individual energy bins). The script has no save() function. We could edit this as soon we have decided on an output format. Also the get_parameter()-function has to be updated and improved. As soon as we have ctulimit available, we should also add this here. The script should merely serve as a starting point.

#2 - 02/11/2015 09:24 AM - Mayer Michael

Thinking about the output format for the spectral points created with csspec, I was wondering whether we could reuse the GMWLSpectrum class. This class, however, would have to be extended to also store TS and upper limit values in each bin. We could further discuss if we need an npred entry for each spectral point.
Accordingly, my proposal is to add m_ts, m_npred and m_ulimit to GMWLDatum and adapt the load/save method from GMWLSpectrum. Does that sound reasonable?

#3 - 02/11/2015 12:46 PM - Sanchez David

make sense to me

david

#4 - 02/11/2015 01:02 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Mayer Michael

The purpose of GMWLSpectrum is to read SED information at any wavelength, hence the underlying data format is very simple and general, and I would not recommend to extend this for the specific purpose of a script.

Why not just using the GFits module to write out the spectral information into a FITS file (in a format that can be read back using GMWLSpectrum). If you need help with setting up the FITS output I can certainly jump in, but usage of GFits and related classes is really very simple.

#5 - 02/11/2015 01:14 PM - Mayer Michael

> Why not just using the GFits module to write out the spectral information into a FITS file (in a format that can be read back using GMWLSpectrum). If you need help with setting up the FITS output I can certainly jump in, but usage of GFits and related classes is really very simple.

I agree, this can be also done easily. My original thought was to have the output format somehow connected to gammalib. This would simplify csspec::save() and generally would allow for simpler data access after the fit. In case GMWLSpectrum is not the right place, we might add a derived class GCTASpectrum for this purpose at some point. But I agree that for now we can define the format within the script itself.

#6 - 02/13/2015 07:13 PM - Mayer Michael

- File spectrum.png added

- Status changed from In Progress to Feedback

- % Done changed from 0 to 90

On branch *1364-write-csspec*, I pushed a complete new version of csspec. I would say the script is quite final except that the upper limit computation is still commented out in line 288ff (see #1363).
Currently, only unbinned analysis is supported. We still could discuss, if and how a binned analysis in each energy bin should be supported. The script could query for the binning parameters in case a parameter (e.g. use_binned) is set to true.
Nevertheless, if a binned observation is passed in the xml file, the script won't work because ctselect cannot operate on a binned observation.

Furthermore, I added a test case for csspec and created a new cscript, called csplotspec (usage should be straight forward). The new script creates an SED out of the fitted model file, the source name, a potential butterfly file and the spectrum points file. Enjoy smile.png

#7 - 02/16/2015 10:34 PM - Knödlseder Jürgen

I've seen your note but had not yet had the time to look into that. Will do as soon as possible ...

#8 - 02/17/2015 08:17 AM - Mayer Michael

Thanks. You might want to update the branch to the latest version, since I merged in the new functionality to pass undefined parameters to ctselect (#1422).

#9 - 02/19/2015 01:56 PM - Sanchez David

Hi

looking at csspec.py, it make the same as analysisutils.py; I will update the later to use csspec.

#10 - 03/02/2015 09:37 PM - Knödlseder Jürgen

- Target version set to 1.0.0

Mayer Michael wrote:

> Thanks. You might want to update the branch to the latest version, since I merged in the new functionality to pass undefined parameters to ctselect (#1422).

Looks good.

I would just propose to use the following FITS column names to be more compatible with what is found usually in FITS files:
Energy, e_Energy, Flux, e_Flux, TS, and UpperLimit.

I merged the code into the devel branch.

I recognized that there was also scripts/csplotspec.py in the code which violates the rule of not using external packages (here matplotlib). Even if the inclusion is conditional, it created troubles in the past. Hence all matplotlib stuff should not be part of the official ctools distribution. You may create a script called show_spectrum.py in the examples folder that displays the FITS file. For the moment the examples folder is the place where no official scripts reside.

Note that there is still the documentation missing. You may create a new branch to add this as the old branch had a merge conflict.

#11 - 03/03/2015 09:38 AM - Mayer Michael

> I would just propose to use the following FITS column names to be more compatible with what is found usually in FITS files:

Energy, e_Energy, Flux, e_Flux, TS, and UpperLimit.
Fine with me. Just a quick comment: When plotting the spectrum in a log-log scale, the energy errors are usually asymmetric (since you want them to appear symmetrically on the log-scale). Should we account for that, e.g. having eu_Energy, ed_Energy?
Related to that, a further thing comes to my mind: Is it planned to also have asymmetric errors on fit parameters?

> I recognized that there was also scripts/csplotspec.py in the code which violates the rule of not using external packages (here matplotlib). Even if the inclusion is conditional, it created troubles in the past. Hence all matplotlib stuff should not be part of the official ctools distribution. You may create a script called show_spectrum.py in the examples folder that displays the FITS file. For the moment the examples folder is the place where no official scripts reside.

Oh right, the script is much better placed in the example folder. Nevertheless it is structured like a cscript to allow for quick and easy plotting of a spectrum.

> Note that there is still the documentation missing. You may create a new branch to add this as the old branch had a merge conflict.

Ok I'll do that.

One other thing I am not sure about is how to handle binned observations, which are currently not supported. In my view there could be three possibilities for the user:

1. use unbinned analysis in every energy bin (supported)
2. use binned analysis in every energy bin (passing an unbinned observation and the binning in each energy bin happens according to user

parameters)
3. use the binning passed in a binned observation. I.e. the user could pass an already binned observation. The binning is encoded and used for the energy bins. Accordingly, one would only fit the flux normalisation in each each map (energy-independent). I am not sure if this is needed at all.

In addition, we might want to be able to split the computation, i.e. providing the user parameters binmin and binmax, similar as in cttsmap.
I guess all these points are not that urgent, but should we create a new issue to not forget them?

#12 - 03/30/2015 01:48 PM - Mayer Michael

- Status changed from Feedback to Pull request

- % Done changed from 90 to 100

I rebased branch *1364-write-csspec* on devel and updated the script csspec. I added the boolean parameter binned that the user can steer if she wants to perform a binned or unbinned analysis in each energy bin. In case of a binned analysis, we now run ctbin, ctexpcube, ctpsfcube and ctbkgcube after ctselect and before ctlike. I added another test case for this scenario.
In addition, I changed the output FITS file to have ed_Energy and eu_Energy, since energy errors are certainly asymmetric since the mean energy is the logarithmic mean.
In addition, I added some logging to keep track of what the tool is doing. Furthermore, I put the upper limit computation in a try-except statement to prevent the whole script to fail in case one upper limit computation fails. Would be good to update this script to the new version.

I also thought a bit more about what happens if the user passes a binned analysis to this tool. An error is thrown, which is quite cryptic. We might therefore think about adding an option to ctool:get_observation to check for binned or unbinned observation and throws a meaningful exception if the wrong type is given. I created #1454 for this case.

#13 - 04/10/2015 10:49 PM - Knödlseder Jürgen

- Status changed from Pull request to Closed

I merged the code into devel.

Files

spectrum.png	30.4 KB	02/13/2015	Mayer Michael