ctools - Feature #1769

Allow to specify an energy range for ctlike

04/19/2016 11:42 AM - Mayer Michael

Status:	New	**Start date:**	
Priority:	Normal	**Due date:**	
Assigned To:		**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

I think this feature has come up already some time ago. I think it would be very nice to support hidden parameters emin and emax in ctlike.
It would simplify the analysis with ctlike, if for any energy selection a preceding ctselect run could be avoided.

Example:
I have a dataset that was selected by energy threshold, FoV, etc. Inspecting the spectral data points from csspec I realise weird features for some energy bins. I therefore would like to run the fit in a particular energy range again for some more debugging information. For e.g. csresmap, I can already specify an energy range for the resulting map. To rerun ctlike, we would need to run ctselect (and/or ctbin etc) to prepare the data accordingly.
Instead we could support sth like ctlike emin=1.0 emax=3.0. For binned/stacked analysis, of course, we would extract maps from the cube (like we do for instance in csspec).

I guess in particular also csspec would be very much simplified by that feature.

History

#1 - 06/03/2016 12:22 AM - Knödlseder Jürgen

What I like with the actual ctlike tool is that it's interface is extremely simple. By adding energy limits we start to mix event selection and model fitting in the same tool. The next step would then be to add temporal selection, etc.

Is there a real speed penalty of running ctselect before ctlike? You can always combine both in a script and then call the script for getting the combined behavior. But I really would like to keep the tools as fundamental as possible.

#2 - 06/14/2016 03:45 PM - Mayer Michael

Sorry for the slow response.
I agree that the simplicity of this tool should indeed be kept the way it ist.

When running analyses interactively via the command line, an additional run of ctselect slows down the computation quite a bit since one has to:

1. Load the observation container into memory (can take up to 1-2 minutes for about 600 observations - large dataset)
2. Create a place to store the newly selected events (1 file per obs, i.e. need to create a new folder somewhere)
3. Reload observation container into memory for ctlike (another 1-2 minutes)

I guess the simplest workaround would be a cscript that can take additional selection arguments emin, emax, tmin, tmax , maybe even ra, dec and rad to selects and runs ctlike in-memory. Or do you think such a script would be too specific?

#3 - 06/17/2016 08:34 PM - Knödlseder Jürgen

I'm wondering whether it would be better to add this as a obsutils function?

On the other hand, I'm not sure whether we really want to support in the long run the obsutils module. At the end, people will want to adapt this level of scripts to their own needs, or may write their own Python scripts that do exactly what they want. That's somehow the entire goal for having modular tools.

#4 - 06/18/2016 10:07 AM - Mayer Michael

To be honest, I have never used the obsutils module. Most of the times it is more convenient to have full control over the individual tools.

I guess such things as fits in a certain energy range can be easier accomplished with workflows. We might for instance think about an enhanced workflow manager that can also run in-memory and pass the outcome from one tool to another.