ctools - Feature #1578

Add convenience scripts for simple tasks

11/17/2015 02:56 PM - Mayer Michael

Status:	Closed	**Start date:**	11/17/2015
Priority:	Normal	**Due date:**	
Assigned To:	Mayer Michael	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

I have a number of scripts for very simple tasks I would like to share with the ctools community. However, I am not sure if each of them justifies to be an own cscript. I would like to hear an opinion about the following putative tools/scripts and where they could be located:

- obs_erange.py
  This script would take an observation (XML or fits) file as input and retrieve the absolut energy range of the observation (e.g. two observation in the container, one with 0.5-10 TeV and the other with 0.1-1 TeV would return 0.1-10 TeV) and maybe save it as GEbounds object in a fits file.

- obs_time.py
  Would compute the sum of observation time using an input observation file and print it on the screen (Ontime and deadtime in different units, e.g sec, min and hours)

- model_merger.py
  Would take an arbitrary number of XML model files and merge them into one file

- obs2ds9.py
  Would take an observation definition file (XML or fits) and translate the pointings into a ds9 region file

- model2ds9.py
  Same as above but using a model XML file as input

History

#1 - 11/17/2015 03:01 PM - Mayer Michael

One could also think about merging the first two scripts into one script. Same is true for the last two scripts.

#2 - 11/18/2015 04:47 PM - Mayer Michael

- File csobsinfo.log added

- Status changed from New to Feedback

- Assigned To set to Mayer Michael

- % Done changed from 0 to 50

I have merged ideas 1,2 and 4 into one cscript called csobsinfo. The tool would take one input parameter, namely an observation container (binned,unbinned, or xml) and prints an extensive summary.
There are two hidden parameters: offset and ds9file, which are false and NONE on default. Specifying offset=yes induces the query for two further parameters, target_ra and target_dec. The script then computes the offset from the target for each pointing. If the ds9file parameter is different from "NONE" a ds9 region file containing the pointing positions is written.
To illustrate the output I have attached csobsinfo.log. In my view, the output of e.g. the energy range could be important to proceed with ctbin. I.e. on wouldn't specify the energy range in ctbin from 0.1-100 TeV if csobsinfo shows that we only have events above 500 GeV.

The script provides methods to return an array of zenith angle, azimuth angle and offsets. These are important quantities a user might want to plot distributions of to illustrate the dataset.

The script was added on branch *1578-add_convenience-scripts*

#3 - 11/19/2015 11:06 AM - Mayer Michael

To the same branch, I've added a script to merge several model XML files into one.

#4 - 11/19/2015 11:14 AM - Mayer Michael

- File csobsinfo.log added

I realised I attached the wrong .log file. Here is the correct one.

#5 - 11/19/2015 03:48 PM - Mayer Michael

- % Done changed from 50 to 80

From the above bullet points there is one still open. The translation of an model XML file into a ds9 region file. I am not sure yet if this should be a cscript providing many parameters (e.g. region colours, test size of the output etc), or just a simple python script that takes a model XML file and translates it using a hard-coded pre-defined algorithm, e.g. point sources as white crosses, extended sources as green circles, ...

#6 - 11/19/2015 11:05 PM - Knödlseder Jürgen

These scripts all look very useful. I think it's indeed best to have all in cscripts to keep a uniform usage and interface.

Concerning csobsinfo I would rename target_ra and target_dec to ra and dec to keep things simple.

For every script we add it is important to:

- add a unit test (so that we can automatically control that the script works)
- add (at least) a reference page (can now be displayed using csobsinfo --help)

#7 - 11/19/2015 11:06 PM - Knödlseder Jürgen

Mayer Michael wrote:

> From the above bullet points there is one still open. The translation of an model XML file into a ds9 region file. I am not sure yet if this should be a cscript providing many parameters (e.g. region colours, test size of the output etc), or just a simple python script that takes a model XML file and translates it using a hard-coded pre-defined algorithm, e.g. point sources as white crosses, extended sources as green circles, ...

I think it should be a cscript with parameters being essentially all hidden (there is no limit to the number of parameters). This provides full flexibility but keeps things simple.

#8 - 11/20/2015 05:41 PM - Mayer Michael

I have added docs and unit tests for csobsinfo.
What is your opinion about modelmerge.py? It is similar as tsmerge.py (except much simpler). You think both scripts should be converted to cscripts, too? There is currently no doc and unit tests for both (except a short printout on start). An alternative could be on the reference manual page to add a section with "other scripts", where they can be listed.
I will take care of csmodel2ds9 next week - do we have a better name?

#9 - 11/20/2015 06:02 PM - Knödlseder Jürgen

Mayer Michael wrote:

> I have added docs and unit tests for csobsinfo.

Thanks.

> What is your opinion about modelmerge.py? It is similar as tsmerge.py (except much simpler). You think both scripts should be converted to cscripts, too? There is currently no doc and unit tests for both (except a short printout on start). An alternative could be on the reference manual page to add a section with "other scripts", where they can be listed.

The problem is how to ensure the long term maintenance of these scripts. I think we should only deliver things for which we can continuously (and automatically) control that they work. For the moment the only exception is the examples folder which contains things that are not tested automatically (I test some of the scripts by hand). In the long run, however, we should also test this automatically, otherwise the maintenance will become a night mare.

I recognized that tsmerge.py is an exception.

I think it would be best to transform also this script into a cscript, including unit testing and documentation. It's certainly best for an end user if all tools "behave" the same way.

> I will take care of csmodel2ds9 next week - do we have a better name?

I'm wondering whether we should make this similar to csobsinfo, hence csmodelinfo or csmodinfo? It could be a tool that generally displays what is in a model, and once added the ds9file parameter it would create a DS9 file.

So at the end we would have:

- csobsinfo (shows what is in an observations definition XML)
- csmodelinfo (shows what is in a model definition XML)
- csmodelmerge (merges model definition XMLs)
- cstsmerge (formerly tsmerge.py; maybe cstsmapmerge?)

Note that I also had in mind to create a GUI tool, similar to Fermi's ModelEditor, to manipulate model definition XMLs. Such a GUI could be written using the Python Tkinter module (this is Python standard, so there should be not cross-platform issues). Such a tool could also be used to merge models. But it probably requires quite some development effort to get a nice GUI, hence I would put this for later and go for now with csmodelmerge.

#10 - 11/24/2015 06:40 PM - Mayer Michael

- Status changed from Feedback to Pull request

- Target version set to Release 1.0 paper

- % Done changed from 80 to 100

I have added the following scripts on branch *1578-add_convenience-scripts*:

- csobsinfo
- csmodelinfo
- csmodelmerge
- cstsmapmerge

They all come with individual unit tests and pages for the reference manual. I guess we thus could remove tsmerge.py. When writing csmodelinfo, I needed a method on the gammalib level. The GModel class now has the function has_ts to signal if the TS value has been computed. The change is available in the gammalib repository on branch *1578-add-has_ts-method-to-GModel*.

Note that when merging either this issue or #1579, there might be conflicts as I was editing the same file (Makefile.am in cscripts and index.rst in the docs).

#11 - 11/24/2015 07:28 PM - Mayer Michael

Just noticed: we should probably update csinfo as well.

#12 - 11/25/2015 12:08 AM - Knödlseder Jürgen

Mayer Michael wrote:

> Just noticed: we should probably update csinfo as well.

Yes, csinfo is not following the general logic of cscripts. And documentation is also missing.

#13 - 11/25/2015 12:09 AM - Knödlseder Jürgen

- Status changed from Pull request to Closed

- Target version deleted (Release 1.0 paper)

Merged into devel.

#14 - 11/25/2015 03:24 PM - Mayer Michael

> Yes, csinfo is not following the general logic of cscripts. And documentation is also missing.

Do you think csinfo should become a cscript? It tests for instance the python import of gammalib and ctools. If this tool was be a cscript, we need to import gammalib and ctools already at the top of the file.
To me it is unclear what csinfo should do apart from listing what is in the index.rst file of the reference manual. In addition, testing the setup of the installation is already covered by make check, isn't it?

#15 - 11/25/2015 07:22 PM - Knödlseder Jürgen

I think Christoph's goal was of having something that allows checking whether the installation was successful. As the script has no user parameters I'm not sure we need a reference page for that.

We could include a sentence in the install instructions that after installation and setup of the init scripts you may run csinfo to see whether everything is okay (make check verifies stuff before installation, csinfo after).

Files

csobsinfo.log	2.8 KB	11/18/2015	Mayer Michael
csobsinfo.log	4.44 KB	11/19/2015	Mayer Michael