ctools - Support #2170

Cannot save response information in simulated observation file

07/21/2017 12:13 PM - Di Venere Leonardo

Status:	Closed	**Start date:**	07/21/2017
Priority:	High	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.4.0		

Description

I'm simulating a list of observations using ctobssim. I set the pointing and response parameters (caldb and irf) in each observation and then initialize ctobssim via:

```
sim = ctools.ctobssim(obs)
```

The I run ctobssim and save the results with the execute() method.
In this way I save the event fits file and the xml observation list, in which I have the event list information, but nothing about the response (calibration database and irf name).
Is there a way to force ctobssim (or the GCTAObservation.write() method) to write these information in the output observation xml?

History

#1 - 07/24/2017 05:28 PM - Di Venere Leonardo

I looked into the gammalib methods that handle the xml creation.

The method GCTAResponseIrf::write(GXmlElement& xml) should handle the caldb information into the output xml. It adds the caldb and irf information only if the variables "m_xml_caldb" and "m_xml_rspname" are not empty. However, these variables are filled only if the GCTAResponseIrf object is created from an input observation xml file containing these information, using the method GCTAResponseIrf::read(const GXmlElement& xml).
When the response is created with other methods, such as "GCTAResponseIrf::load(const std::string& rspname)", which is my case, the variables are not written and the information are therefore not saved in the output xml.

Is this behaviour intended/desirable?
In case not, I guess that we should just define the "m_xml_caldb" and "m_xml_rspname" variables also in the load() method to solve the bug.
Otherwise, is there another way to force these information in the output xml?

#2 - 07/25/2017 03:23 AM - Knödlseder Jürgen

- Subject changed from cannot save response information in simulated observation file to Cannot save response information in simulated observation file

- Assigned To set to Knödlseder Jürgen

- Priority changed from Normal to High

- Target version set to 1.4.0

The behavior is indeed intended, but I currently see no reason not to change it. Originally I have followed the principle that only what is read in is also written out. I need to check if there are any side effects in changing the behavior.

#3 - 07/25/2017 08:05 PM - Knödlseder Jürgen

Here is a test run I did with the current ctools version (1.4.0.dev1). In this case the response information is written in the output file. Here the input XML file I use to initialise the run:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<observation_list title="observation list">
  <observation name="GPS" id="110000" instrument="CTA">
    <parameter name="Pointing" ra="186.1561" dec="-64.019" />
    <parameter name="EnergyBoundaries" emin="30000" emax="160000000" />
    <parameter name="GoodTimeIntervals" tmin="662774400" tmax="662776200" />
    <parameter name="TimeReference" mjdrefi="51544" mjdreff="0.5" timeunit="s" timesys="TT" timeref="LOCAL" />
    <parameter name="RegionOfInterest" ra="186.1561" dec="-64.019" rad="5" />
    <parameter name="Deadtime" deadc="0.98" />
  </observation>
  <observation name="GPS" id="110001" instrument="CTA">
    <parameter name="Pointing" ra="186.1561" dec="-64.019" />
    <parameter name="EnergyBoundaries" emin="30000" emax="160000000" />
    <parameter name="GoodTimeIntervals" tmin="662776320" tmax="662778120" />
    <parameter name="TimeReference" mjdrefi="51544" mjdreff="0.5" timeunit="s" timesys="TT" timeref="LOCAL" />
    <parameter name="RegionOfInterest" ra="186.1561" dec="-64.019" rad="5" />
    <parameter name="Deadtime" deadc="0.98" />
  </observation>
</observation_list>
```

Note that there is no response information in the XML file.
I then do the following:

```
>>> import gammalib
>>> import ctools
>>> obs=gammalib.GObservations('obs.xml')
>>> sim=ctools.ctobssim(obs)
>>> sim.execute()
Calibration database [prod2]
Instrument response function [South_0.5h]
Input model definition XML file [$CTOOLS/share/models/crab.xml]
Output event data file or observation definition XML file [out.xml]
```

The output file now looks as follows:

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<observation_list title="observation list">
  <observation name="GPS" id="110000" instrument="CTA">
    <parameter name="EventList" file="sim_events_000001.fits" />
    <parameter name="Calibration" database="prod2" response="South_0.5h" />
  </observation>
  <observation name="GPS" id="110001" instrument="CTA">
    <parameter name="EventList" file="sim_events_000002.fits" />
    <parameter name="Calibration" database="prod2" response="South_0.5h" />
  </observation>
</observation_list>
```

The response information is appended.

The reason for this is that ctobssim make use of ctool::set_obs_response() which loads the response information in fact via an XML file:

```
      // Load response information
      std::string database = (*this)["caldb"].string();
      std::string irf      = (*this)["irf"].string();

      // Create an XML element containing the database and IRF name. This
      // kluge will make sure that the information is later written
      // to the observation definition XML file, in case an observation
      // definition XML file is written.
      std::string parameter = "parameter name=\"Calibration\""
                              " database=\""+database+"\""
                              " response=\""+irf+"\"";
```

```
    GXmlElement xml;
    xml.append(parameter);

    // Create CTA response
    GCTAResponseIrf response(xml);
```

So ctobssim is in fact doing already what you would like to see.

Could you post exactly the way how you setup your run? I expect that you construct the obs container directly without going through an XML file?

#4 - 07/26/2017 10:22 AM - Di Venere Leonardo

I construct the obs container using the same method used in the method set_obs() defined in the script cscripts/obsutils.py.
Once I have the observation list defined providing the pointing, energy, time, ... information, I define the response by creating a GCaldb object and setting the obs response via obs.response() method:

```
caldb = gammalib.GCaldb("cta","prod2")
for o in obs:
   o.response("South_0.5h",caldb)
sim=ctools.ctobssim(obs)
sim.execute()
```

Once I set the response, ctobssim does not query the caldb and irf parameters any more and the caldb information is not written in the output obs list.

The issue is the same if I define the obs list from an XML file, like the one in your example, instead of providing explicitly all the information:

```
obs=gammalib.GObservations('obs.xml')
caldb = gammalib.GCaldb("cta","prod2")
for o in obs:
   o.response("South_0.5h",caldb)
sim=ctools.ctobssim(obs)
sim.execute()
```

The reason is that the response object is created without an XML element (either explicit or via the ctobssim workaround).

user#3 wrote:

> Here is a test run I did with the current ctools version (1.4.0.dev1). In this case the response information is written in the output file. Here the input XML file I use to initialise the run:
> [...]Note that there is no response information in the XML file.
> I then do the following:
> [...]
> The output file now looks as follows:
> [...]The response information is appended.
>
> The reason for this is that ctobssim make use of ctool::set_obs_response() which loads the response information in fact via an XML file:
> [...]
> So ctobssim is in fact doing already what you would like to see.

The method that you tested is interesting and solves the issue, even though it might be limiting if I want to specify a different response for each obsevartion in the container.
However, I could also solve the problem using the same procedure used in ctool::set_obs_response(), without calling ctobssim explicitly:

```
obs = gammalib.GObservations('obs.xml')
xml = gammalib.GXmlElement()
```

```
string = 'parameter name="Calibration" database="prod2" response="South_0.5h"'
xml.append(string)
rsp = gammalib.GCTAResponseIrf(xml)
for o in obs:
    o.response(rsp)
obs.save("output_obs_list.xml")
```

The output obs list now contains the caldb information and in principle I could specify a different caldb for each observation.
This is a good workaround to avoid the problem and does not depend on whether I call ctobssim or not.

I don't know if it is worth modifying the gammalib to force the caldb information in the output xml file even when one adopts the method I was using before, so that the two ways of creating the response provide consisten outputs.

#5 - 07/26/2017 11:11 AM - Knödlseder Jürgen

- Status changed from New to Feedback

- % Done changed from 0 to 90

I changed the code so that the calibration information is now written into the XML file.

I merged the change into devel so that you can test yourself whether it's now working as expected. I wait for your feedback before closing the issue.

#6 - 07/26/2017 11:41 AM - Di Venere Leonardo

It works.
I tested this code using the devel branch:

```
obs = gammalib.GObservations('obs.xml')
caldb = gammalib.GCaldb("cta","prod2")
for o in obs:
   o.response("South_0.5h",caldb)
obs.save("output_obs_list.xml")
```

Now the output observation list contains the caldb information.
Thanks!

#7 - 07/26/2017 01:02 PM - Knödlseder Jürgen

- Status changed from Feedback to Closed

- % Done changed from 90 to 100