GammaLib - Feature #1435

Support bundling of events and IRFs in FITS files

03/05/2015 09:48 AM - Mayer Michael

Status:	Closed	**Start date:**	03/05/2015
Priority:	Normal	**Due date:**	
Assigned To:		**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

Following a Seevogh call on IRF formats, there was the proposal by Jürgen to bundle events and IRFs into one file. In HESS, we want to implement this approach into the FITS exporters. This could simplify the XML interface of an observation definition file. Currently, the user has to provide an individual file for events and each IRF.

```
<observation name="Crab" id="23523" instrument="HESS">
   <parameter name="EventList" file="events_23523.fits" />
   <parameter name="EffectiveArea" file="Aeff_23523.fits" />
   <parameter name="PointSpreadFunction" file="psf_23523.fits" />
   <parameter name="EnergyDispersion" file="edisp_23523.fits" />
   <parameter name="Background" file="bg_23523.fits" />
</observation>
```

Maybe we could also support one file per run containing all information:

```
<observation name="Crab" id="23523" instrument="HESS">
   <parameter name="Run" file="run_23523.fits" />
</observation>
```

History

#1 - 03/05/2015 09:58 AM - Deil Christoph

I think in X-ray astronomy tools one usually specifies where the EVENTS HDU is and then the header contains references to IRF and background HDUs (mostly in other files)? Is this an OGIP standard we should follow or just define it in a simple way via the XML observation lists like Michael suggests?

#2 - 03/05/2015 11:43 AM - Knödlseder Jürgen

OGIP standard is basically CALDB, which indeed stores information per HDU.

In this case I would simply implement a logic where with the current format

```
<observation name="Crab" id="23523" instrument="HESS">
   <parameter name="EventList" file="events_23523.fits" />
</observation>
```

GammaLib would automatically check whether response information is provided with the event file, and if present, the IRF will automatically be used. So it won't be a format change, but rather a change in GCTAObservation.

#3 - 03/09/2015 01:23 PM - Mayer Michael

Ok, I see. So here is what I think is needed:

- GCTAObservation::read(const GFits& fits) should check if response information is available and read it to GCTAResponseIrf
- Implement GCTAResponseIrf::read(const GFits& fits) to avoid reopening the FITS file.
- Adapt GCTAObservation::write(GFits& fits) to write out the response in case it was available before.
- Change GCTAObservation::write(GXmlElement& xml) and GCTAObservation::read(const GXmlElement& xml) to use the new format, too.

Do you think these steps are sufficient? And also, do you think, this new format provides an improvement and simplification to the user?

#4 - 03/09/2015 01:40 PM - Knödlseder Jürgen

Hi Michael,

Mayer Michael wrote:

> Ok, I see. So here is what I think is needed:
>
> - GCTAObservation::read(const GFits& fits) should check if response information is available and read it to GCTAResponseIrf

Ok.

> - Implement GCTAResponseIrf::read(const GFits& fits) to avoid reopening the FITS file.

We should also add a write(GFits& fits) method to enabling writing to a FITS file. I was then thinking to use this by default in ctobssim so that once simulated the user would never be asked again for the response files (as they are tied to the event file)

> - Adapt GCTAObservation::write(GFits& fits) to write out the response in case it was available before.

Ok.

> - Change GCTAObservation::write(GXmlElement& xml) and GCTAObservation::read(const GXmlElement& xml) to use the new format, too.

Is there any change needed? I guess the logic would be that specifying response information in the XML file should overwrite the one from the FITS file. But maybe this needs some thinking/discussion.

> Do you think these steps are sufficient? And also, do you think, this new format provides an improvement and simplification to the user?

It definitely provides a simplification. In principle it will make all the caldb and irf parameters in the ctools obsolete. We may think how to make the overall system evolve. Maybe we leave these parameters for the moment for the case that no response information is provided in the event file, but once ctobssim always writes out the response information one should not really need these parameters anymore.

Note, however, that so far the other response formats (performance table, ARF/RMF) are still separate from the event file, hence at least for those formats we still need to support the "old way". We could however think about converting these "old formats" internally into the response table format, having thus **always** the possibility to write out the IRF with the events. But this is certainly low priority. Let's stick for the moment to the response table case.

#5 - 03/09/2015 02:13 PM - Mayer Michael

Thanks for the feedback.

> We should also add a write(GFits& fits) method to enabling writing to a FITS file. I was then thinking to use this by default in ctobssim so that once simulated the user would never be asked again for the response files (as they are tied to the event file)

Definitely. I forgot to list that one.

> Is there any change needed? I guess the logic would be that specifying response information in the XML file should overwrite the one from the FITS file. But maybe this needs some thinking/discussion.

For the moment, GCTAObservation::write(GXmlElement& xml) has the following code:

```
// Write response information
if (m_response != NULL) {
    m_response->write(xml);
}
```

This might have to be adjusted since the response information do not have to be written into the xml file anymore.

> It definitely provides a simplification. In principle it will make all the caldb and irf parameters in the ctools obsolete. We may think how to make the overall system evolve. Maybe we leave these parameters for the moment for the case that no response information is provided in the event file, but once ctobssim always writes out the response information one should not really need these parameters anymore.

Sounds good.

> Note, however, that so far the other response formats (performance table, ARF/RMF) are still separate from the event file, hence at least for those formats we still need to support the "old way". We could however think about converting these "old formats" internally into the response table format, having thus always the possibility to write out the IRF with the events. But this is certainly low priority. Let's stick for the moment to the response table case.

I have thought about this problem, too. RMF and ARF are also stored as FITS, right? So we might only think about how to deal with performance tables. We could e.g. add GCTAAeff::write(GFits& fits) (also PSF, RMF and Background) and implement that in all derived classes.
I think we should definitely continue to support the old way to read the observation. It allows flexibility to quickly test different IRF versions or background models. After running e.g. ctselect or ctobsim, the IRFs are then connected to the events for further analysis.

#6 - 03/09/2015 02:47 PM - Knödlseder Jürgen

Mayer Michael wrote:

> For the moment, GCTAObservation::write(GXmlElement& xml) has the following code:
> [...]
> This might have to be adjusted since the response information do not have to be written into the xml file anymore.

Ok, I'm not fully sure what logic should be implemented.

So far we often implemented the logic "write out what was written in", yet it could well be that a user wants to add response information by hand and have this written to the XML file, and for that case the "write out what was written in" logic does not work.

> I have thought about this problem, too. RMF and ARF are also stored as FITS, right? So we might only think about how to deal with performance tables. We could e.g. add GCTAAeff::write(GFits& fits) (also PSF, RMF and Background) and implement that in all derived classes.

I won't do anything now for these classes as it's not clear to me whether we want to keep that response format anyways. We may postpone the decision.

> I think we should definitely continue to support the old way to read the observation. It allows flexibility to quickly test different IRF versions or background models. After running e.g. ctselect or ctobsim, the IRFs are then connected to the events for further analysis.

Ok. I would not connect IRF information in ctselect, though, as this is a step that should in principle not require IRF information. Changing in fact the background model makes pretty much sense, changing effective area and PSF makes less sense. So we need a mechanism that allows changing response components. This is easy using XML files, but more tricky from the ctools parameter interface. Maybe XML is sufficient?

#7 - 03/09/2015 03:16 PM - Mayer Michael

> So far we often implemented the logic "write out what was written in", yet it could well be that a user wants to add response information by hand and have this written to the XML file, and for that case the "write out what was written in" logic does not work.

I see your point. We will probably see what is needed when we have some use cases.

> I won't do anything now for these classes as it's not clear to me whether we want to keep that response format anyways. We may postpone the decision.

Fully agree

> Ok. I would not connect IRF information in ctselect, though, as this is a step that should in principle not require IRF information. Changing in fact the background model makes pretty much sense, changing effective area and PSF makes less sense. So we need a mechanism that allows changing response components. This is easy using XML files, but more tricky from the ctools parameter interface. Maybe XML is sufficient?

In ctselect we already use IRF information (in case usethres="DEFAULT"). Of course, in general this should not be mandatory (ultimately we might have sth like ctmkthres to be run beforehand).
I could also think that the user might want to change or test different PSF parametrisations (e.g. TripleGauss vs King).
For CTA, we might have a default setting (default Aeff, default PSF, default Background), where events and IRFs are bundled one file. In case the user wants to change a response component, the "old/current" XML format should be sufficient to use.

#8 - 05/28/2021 03:29 AM - Knödlseder Jürgen

- Status changed from New to Closed

- % Done changed from 0 to 100

The bundling is supported as illustrated by the analysis of the public H.E.S.S. data. I close the issue now.