

GammaLib - Feature #1036

Implement energy resolution

12/10/2013 11:00 AM - Deil Christoph

Status:	Closed	Start date:	12/10/2013
Priority:	High	Due date:	
Assigned To:	Forest Florent	% Done:	100%
Category:		Estimated time:	8.00 hours
Target version:	Stage Florent		
Description Energy resolution is one of the few (or the only) major missing piece to get correct spectral results from gammalib / ctools. Here we can discuss how it should be implemented. References (we probably want to stick to OGIP standards as far as possible): <ul style="list-style-type: none">• Formula 7.16 in http://inspirehep.net/record/1122589• http://adsabs.harvard.edu/abs/2001ApJ...548.1010D *http://heasarc.gsfc.nasa.gov/docs/heasarc/caldb/docs/memos/cal_gen_92_002/cal_gen_92_002.html			
Subtasks:			
Action # 1110: Write a test case			Closed
Action # 1109: Add GCTAResponse::load_edisp method and link it to GCTAResponse::load fo...			Closed
Action # 1108: Add integration over measured energies to GCTAResponse::nedisp			Closed
Action # 1107: Implement integration over true photon energy in GModelSky::integrate_en...			Closed
Action # 1106: Add mc() method to abstract GCTAEdisp interface			Closed
Action # 1105: Create GCTAEdispPerfTable class			Closed
Action # 1104: Add Monte Carlo sampling and minimum / maximum energies for energy dispe...			Rejected
Action # 1424: Complete and verify RMF implementation of energy dispersion.			Closed
Action # 1423: Verify performance table implementation of energy dispersion (GEdispPerf...			Closed
Action # 1126: Add GCTAEdisp2D class			Closed
Action # 1120: Add GCTAEdispRMF class			Closed

History

#1 - 12/10/2013 11:04 AM - Deil Christoph

We have an existing C++ implementation (using GSL via ROOT for heavy numerical lifting) of XSPEC-style forward folding with the energy resolution matrix in HESS.

The plan is to have this open-sourced on Github before the coding sprint, but in any case we can probably re-use some parts for Gammalib.

For testing we can compare to XSPEC.

#2 - 12/10/2013 12:03 PM - Deil Christoph

- Description updated

#3 - 12/10/2013 12:04 PM - Deil Christoph

Jürgen, can you briefly comment on the status in gammalib and possible sub-tasks?

Is there an RMF reader? Which classes would we work on? ...

#4 - 12/10/2013 12:15 PM - Knödlseider Jürgen

Deil Christoph wrote:

Jürgen, can you briefly comment on the status in gammalib and possible sub-tasks?

Is there an RMF reader? Which classes would we work on? ...

Nothing has really be done for energy resolution, but the structure is in principle there to plug-in a probability density function taking true and reconstructed energies as parameters. So maybe a starting point would be to implement a very simple energy dispersion function (e.g. a Gaussian), and see if we can get the code working with such a function. The code would be probably awefully slow in a first run, but can then be optimised once we get the functionality working

Work is needed here on the response side, but also on the MC side, so that we can do some testing.

Concerning RMFs, there is the GRmf class available, and I recently added support for variable-length columns to the FITS module so that I can read the files that are currently produced by HESS. My plan is to introduce a generic XSpec observation interface, so that any data in XSpec format can be used in any GammaLib analysis. This should also support energy redistribution, which would be easier as it's just a multiplication of matrix elements to a data vector.

So there are two paths:

- implement a class derived from GCTAEdisp (for example a simple Gaussian), to be used for unbinned and binned analysis
- implement full XSpec support, using the GRmf class for Xspec analysis

#5 - 12/11/2013 08:26 PM - Knödlseider Jürgen

- Target version set to 2nd coding sprint

#6 - 12/11/2013 10:13 PM - Knödlseider Jürgen

Here a list of things needed to implement the energy dispersion for unbinned analysis:

- GModelSky::integrate_energy: implement integration over true energy
- GCTAResponse: add a method that returns the true energy integration boundaries for a given measured energy; this probably needs a similar method to be implement for GCTAEdisp.
- GCTAResponse::nedisp: implement method (integration of energy dispersion over [emin,emax] of dataspace)
- GCTAEdispPerfTable: implement Gaussian energy dispersion based on rms values in performance table
- GCTAResponse::load_edisp: implement method that loads energy dispersion information (similar to load_psf()), and call this method from GCTAResponse::load (so far, GCTAResponse::load_edisp is defined empty in GCTAResponse.hpp)
- Add MC sampling to GCTAResponse::mc

I hope that's it ...

#7 - 12/11/2013 10:35 PM - Deil Christoph

- Assigned To set to Deil Christoph

#8 - 12/12/2013 09:23 AM - Deil Christoph

Jürgen, thanks for making this list, very helpful!

We also want energy resolution in binned analysis.

Can you please also make a list of things needed for that?

#9 - 12/12/2013 09:47 AM - Knödlseider Jürgen

Deil Christoph wrote:

Jürgen, thanks for making this list, very helpful!

We also want energy resolution in binned analysis.

Can you please also make a list of things needed for that?

The changes I listed should in fact do the job for unbinned and binned analysis. But for binned you probably would like to precompute stuff beforehand so that computation is faster. Hence some additional thinking is required here. This is probably linked to issue #803 which asks for implementing a interface that is also optimized for binned analysis (so far the interfaces are only optimized for unbinned).

However, if you'd like to use the RMFs, as defined during the last coding sprint, additional work is needed. But we even have not a fitter yet for Xspec like analysis, hence when we implement the fitter we probably could do the RMFs in one shot.

Just one thing: when implementing energy dispersion we should always have the possibility to switch it off, mainly for computational speed reasons, but also for comparison reasons. Shouldn't be complicated to implement, by maybe needs a flag that can be toggled by a specific GCTAResponse method.

#10 - 12/12/2013 10:10 AM - Deil Christoph

I think we'd like to use RMFs for energy resolution, at least to start with.

RMFs are already available in some of the HESS exporters and we haven't talked about another data format yet.

Isn't the use of RMFs to store energy resolution info independent of the question whether we do an XSPEC-style analysis (i.e. for a source region) or a cube-style likelihood analysis? For now we just want to get correct spectra for small sources in the FOV, so I think it's OK to apply the correct energy resolution from that position to all positions in the FOV, no?

#11 - 12/12/2013 10:19 AM - Knödlseider Jürgen

Deil Christoph wrote:

I think we'd like to use RMFs for energy resolution, at least to start with.
RMFs are already available in some of the HESS exporters and we haven't talked about another data format yet.

Isn't the use of RMFs to store energy resolution info independent of the question whether we do an XSPEC-style analysis (i.e. for a source region) or a cube-style likelihood analysis? For now we just want to get correct spectra for small sources in the FOV, so I think it's OK to apply the correct energy resolution from that position to all positions in the FOV, no?

I think we should implement the full framework, and then we could add a GCTAEdispRmf method which is equivalent to the GCTAAeffArf method. But in fact, this is not really the issue. The issue is that for the standard binned/unbinned analysis, you have to implement the set of methods described above. For an Xspec-style analysis, you won't need any of these methods, as energy dispersion then boils down to a mere vector multiplication. And this one would implement at the OnOff fitter level I guess.

#12 - 01/28/2014 10:19 AM - Knödlseider Jürgen

That's our working branch for this feature: 1036-implement-energy-resolution

#13 - 01/29/2014 01:48 AM - Knödlseider Jürgen

- Status changed from New to In Progress

#14 - 01/31/2014 09:34 AM - Deil Christoph

Jürgen, could you please review (and merge if OK) the following branches into the issue_1036 integration branch?

https://github.com/cdeil/gammalib/compare/action_1109

https://github.com/cdeil/ctools/compare/action_1109

Currently the ctools test fail indicating what to work on next:

* Test cscripts *

Test cspull:

Traceback (most recent call last):

File "/Users/deil/code/ctools/scripts/cspull.py", line 360, in <module>
app.execute()

File "/Users/deil/code/ctools/scripts/cspull.py", line 173, in execute
self.run()

File "/Users/deil/code/ctools/scripts/cspull.py", line 210, in run
result = self.trial(seed)

File "/Users/deil/code/ctools/scripts/cspull.py", line 286, in trial
like = obsutils.fit(obs, log=self.m_log, debug=self.m_debug)

File "/Users/deil/code/ctools/scripts/obsutils.py", line 147, in fit
like.run()

File "/Users/deil/code/ctools/pyext/ctools.py", line 319, in run
return _ctools.ctlike_run(self)

RuntimeError: *** ERROR in GModelSky::integrate_energy(GEvent&, GTime&, GObservation&, bool): Feature not implemented. In case that you need this feature for your application please submit a feature request on <https://sourceforge.net/projects/gammalib/>, join this error message and provide a detailed description of your needs.

pull.dat file is not valid

#15 - 02/15/2014 11:48 AM - Deil Christoph

Both Ellis and I won't have much time to work on GammaLib / ctools in the next month, so we'd like to wrap this up for now.

Jürgen, can you merge the existing code into devel (I don't know if this is already done) and make separate issues for future work so that we can come back to this ~ in April?

I think some things that still need to be done:

- Add option to ctlike to apply energy resolution in the fit (I think I did that already and if not it should be easy to do).
- More unit tests? (where can we see the coverage of the code we wrote? I doubt it's well-tested at the moment.)
- Some numerical precision / computation speed tests?
- Make high-level tests (pull distributions) simulating / fitting spectra with energy dispersion.
- Write high-level documentation on energy dispersion (including maybe a plot or two) in the user guide.

#16 - 02/15/2014 02:03 PM - Knödseder Jürgen

Deil Christoph wrote:

Both Ellis and I won't have much time to work on GammaLib / ctools in the next month, so we'd like to wrap this up for now.

Jürgen, can you merge the existing code into devel (I don't know if this is already done) and make separate issues for future work so that we can come back to this ~ in April?

Most of the code is already in devel, I just started to look in the new code using the RMFs. Some work is needed here (see #1120).

I think some things that still need to be done:

- Add option to ctlike to apply energy resolution in the fit (I think I did that already and if not it should be easy to do).

It's there already (parameter edisp). I made some first tests and everything seems to work. But we need science validation now!

- More unit tests? (where can we see the coverage of the code we wrote? I doubt it's well-tested at the moment.)

Look for example here: <https://cta-sonar.irap.omp.eu/drilldown/measures/1140?metric=coverage&rids%5B%5D=1209>

- GCTAEdisp: 61.3%
- GCTAEdispPerfTable: 62.8%

Is not so bad, but can of course be improved.

- Some numerical precision / computation speed tests?

Indeed, needed.

- Make high-level tests (pull distributions) simulating / fitting spectra with energy dispersion.

Also. I can start doing this at some point.

- Write high-level documentation on energy dispersion (including maybe a plot or two) in the user guide.

Agree.

#17 - 02/24/2014 10:46 AM - Deil Christoph

Knödlseider Jürgen wrote:

Deil Christoph wrote:

- Add option to ctlike to apply energy resolution in the fit (I think I did that already and if not it should be easy to do).

It's there already (parameter edisp). I made some first tests and everything seems to work. But we need science validation now!

I tried a simple example and energy dispersion doesn't seem to work with ctlike in devel at the moment!?

https://github.com/cdeil/gammalib-tutorials/tree/master/energy_dispersion

Maybe I'm making a simple mistake?

Can you share the test case where it works for you?

#18 - 04/18/2014 12:35 PM - Deil Christoph

Jürgen, do you have a working example of a ctlike fit with energy resolution?

#19 - 07/19/2014 02:10 AM - Knödlseider Jürgen

- Target version deleted (2nd coding sprint)

#20 - 02/18/2015 06:34 PM - Knödlseider Jürgen

- Target version set to Stage Florent

#21 - 02/18/2015 06:41 PM - Knödlseider Jürgen

- Due date set to 01/29/2014

due to changes in a related task

#22 - 02/18/2015 06:41 PM - Knödlseider Jürgen

- Due date set to 01/29/2014

due to changes in a related task

#23 - 02/18/2015 06:42 PM - Knödlseider Jürgen

- Due date set to 01/31/2014

due to changes in a related task

#24 - 06/06/2015 11:19 PM - Knödlseeder Jürgen

- File noedisp_2D_mean.png added
- File noedisp_2D_normalization.png added
- File noedisp_2D_sigma.png added
- File noedisp_perf_mean.png added
- File noedisp_perf_normalization.png added
- File noedisp_perf_sigma.png added
- File noedisp_rmf_mean.png added
- File noedisp_rmf_normalization.png added
- File noedisp_rmf_sigma.png added
- Assigned To changed from Deil Christoph to Hassan Collado Tarek

Here is an overview over the current performances. The following plots show pull distributions obtained for a Gaussian model:

```
<spectrum type="Gaussian">  
  <parameter name="Normalization" scale="1e-10" value="1.0" min="1e-07" max="1000.0" free="1"/>  
  <parameter name="Mean" scale="1e6" value="5.0" min="0.01" max="100.0" free="1"/>  
  <parameter name="Sigma" scale="1e6" value="1.0" min="0.01" max="100.0" free="1"/>  
</spectrum>
```

The following 6 rows show plots for performance table response functions (row 1-2), RMF response functions (row 3-4) and 2D response functions (row 5-6). The first row of each set of two rows shows the pull distribution without energy dispersion, while the second row shows the pull distribution with energy dispersion. While energy dispersion works fine for the performance tables, there are some biases in the RMF and 2D response functions.

Response	Edisp	Normalization	Mean	Sigma
Perf	No			
Perf	Yes			
RMF	No			
RMF	Yes			
2D	No			
2D	Yes			

#25 - 06/06/2015 11:20 PM - Knödlseider Jürgen

- File edisp_2D_mean.png added
- File edisp_2D_normalization.png added
- File edisp_2D_sigma.png added
- File edisp_perf_mean.png added
- File edisp_perf_normalization.png added
- File edisp_perf_sigma.png added
- File edisp_rmf_mean.png added
- File edisp_rmf_normalization.png added
- File edisp_rmf_sigma.png added

#26 - 06/06/2015 11:20 PM - Knödlseider Jürgen

- Assigned To changed from Hassan Collado Tarek to Forest Florent

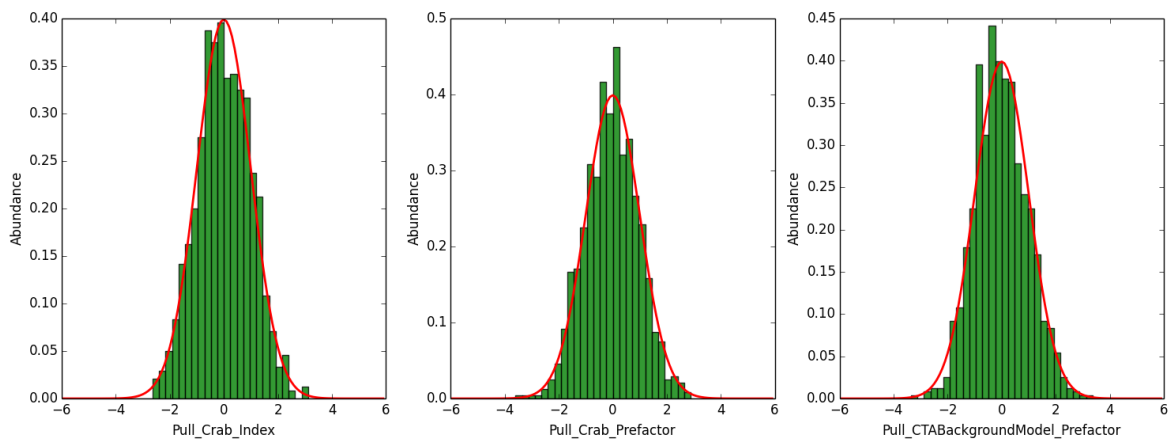
#27 - 06/08/2015 11:16 AM - Mayer Michael

- File test_edisp2d_crab_plaw.png added

Thanks for making this effort. I repeated this test using a power law instead of a Gaussian spectral model and did not find a bias for the 2D format. This is how I executed cspull:

```
cspull edisp=yes
RA of pointing (deg) (0-360) [83.6331]
Dec of pointing (deg) (-90-90) [22.0145]
Duration (in s) [450]
Lower energy limit (TeV) [0.1]
Upper energy limit (TeV) [10]
Calibration database [prod2]
Instrument response function [South_50h]
Number of energy bins (0=unbinned) [0]
Source model [$CTOOLS/share/models/crab.xml]
Output file name [pull.dat]
Number of trials [200] 1000
```

The results are attached. They look fine to me. Is it possible that the Gaussian model itself has a bias?



#28 - 06/08/2015 11:44 AM - Knödlseider Jürgen

Thanks for making the check.

No, the Gaussian model has no bias as the energy dispersion works well for performance tables. I suspect that the problem is that there is a lot of noise in the actual energy dispersion matrix and that the integration method is not accurate enough. I'm working on that.

Also note that the Gaussian model is much more sensitive to energy dispersion problems than the power law model is, so it may just be that you

cannot detect the effects.

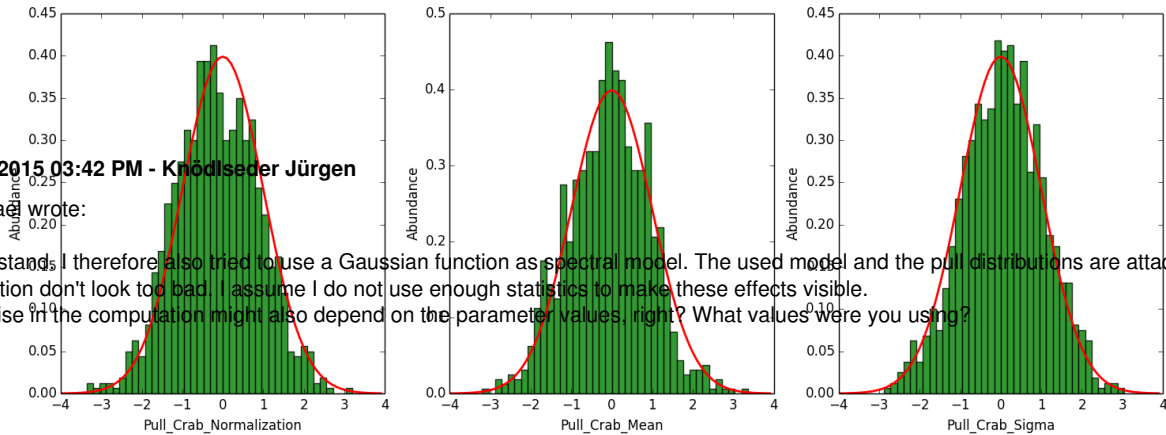
#29 - 06/08/2015 03:39 PM - Mayer Michael

- File `point_gauss.xml` added

- File `test_edisp2d_crab_gauss.png` added

I understand. I therefore also tried to use a Gaussian function as spectral model. The used model and the pull distributions are attached. The pull distribution don't look too bad. I assume I do not use enough statistics to make these effects visible.

The noise in the computation might also depend on the parameter values, right? What values were you using?



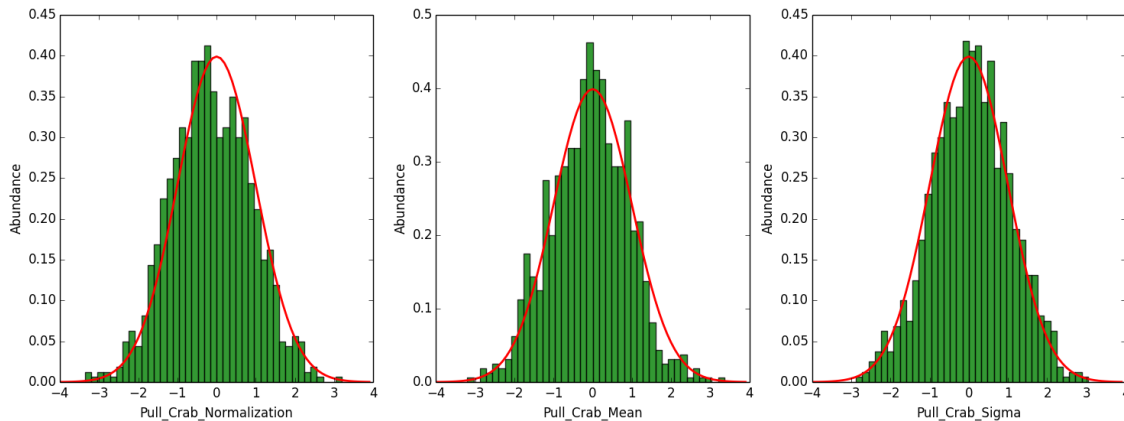
#30 - 06/08/2015 03:42 PM - Knödlseider Jürgen

Mayer Michael wrote:

I understand. I therefore also tried to use a Gaussian function as spectral model. The used model and the pull distributions are attached. The pull distribution don't look too bad. I assume I do not use enough statistics to make these effects visible.

The noise in the computation might also depend on the parameter values, right? What values were you using?

I use a line at 5 TeV with a sigma of 1 TeV (see above).



#31 - 06/08/2015 03:50 PM - Mayer Michael

I was using a line at 2 TeV with 0.4 TeV width. So mine would be more pronounced. Could that influence the pull distributions?

#32 - 06/08/2015 04:06 PM - Knödlseider Jürgen

Maybe. What response function are you using (I still suspect that the noise in the response function is influencing the results)?

#33 - 06/08/2015 04:30 PM - Mayer Michael

I am using 'prod2' and 'South_50h'. Of course I can try to do the same exercise using the HESS IRFs in order to see if there is a difference.

#34 - 06/08/2015 05:48 PM - Knödlseider Jürgen

Mayer Michael wrote:

I am using 'prod2' and 'South_50h'. Of course I can try to do the same exercise using the HESS IRFs in order to see if there is a difference.

I confirm that using the 'prod2' and 'South_50h' I get much better results. I will post pull distributions once they have a sufficient number of samples.

#35 - 06/19/2015 11:51 PM - Knödlseider Jürgen

- Status changed from In Progress to Closed

Files

noedisp_2D_mean.png	43.3 KB	06/06/2015	Knödlseider Jürgen
noedisp_2D_normalization.png	45.8 KB	06/06/2015	Knödlseider Jürgen
noedisp_2D_sigma.png	44.2 KB	06/06/2015	Knödlseider Jürgen
noedisp_perf_mean.png	43.3 KB	06/06/2015	Knödlseider Jürgen
noedisp_perf_normalization.png	45.6 KB	06/06/2015	Knödlseider Jürgen
noedisp_perf_sigma.png	44.2 KB	06/06/2015	Knödlseider Jürgen
noedisp_rmf_mean.png	43.4 KB	06/06/2015	Knödlseider Jürgen
noedisp_rmf_normalization.png	45.7 KB	06/06/2015	Knödlseider Jürgen
noedisp_rmf_sigma.png	44.2 KB	06/06/2015	Knödlseider Jürgen
edisp_2D_mean.png	43.4 KB	06/06/2015	Knödlseider Jürgen
edisp_2D_normalization.png	45.9 KB	06/06/2015	Knödlseider Jürgen
edisp_2D_sigma.png	44.7 KB	06/06/2015	Knödlseider Jürgen
edisp_perf_mean.png	43.4 KB	06/06/2015	Knödlseider Jürgen
edisp_perf_normalization.png	45.8 KB	06/06/2015	Knödlseider Jürgen
edisp_perf_sigma.png	43.9 KB	06/06/2015	Knödlseider Jürgen
edisp_rmf_mean.png	43.8 KB	06/06/2015	Knödlseider Jürgen
edisp_rmf_normalization.png	45.9 KB	06/06/2015	Knödlseider Jürgen
edisp_rmf_sigma.png	44.7 KB	06/06/2015	Knödlseider Jürgen
test_edisp2d_crab_plaw.png	77.9 KB	06/08/2015	Mayer Michael
point_gauss.xml	1.2 KB	06/08/2015	Mayer Michael
test_edisp2d_crab_gauss.png	75.2 KB	06/08/2015	Mayer Michael