GammaLib - Action #1800

Implement alternative exponentially cut-off power law spectral model

06/22/2016 03:18 PM - Ziegler Alexander

Status:	Closed	**Start date:**	06/22/2016
Priority:	Normal	**Due date:**	
Assigned To:	Ziegler Alexander	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.2.0		

Description

I have a question about the spectral model Exponentially cut-off power law, as described here:
http://cta.irap.omp.eu/ctools/user_manual/getting_started/models.html#exponentially-cut-off-power-law

From the model description, I assume that the parameter used to describe the cutoff in the model is directly E_cut. Yet, there is the other possibility to use as free parameter lambda=1/E_cut for the fitting procedure instead of fitting E_cut directly.
However, the bias of the cutoff-energy is different for those two possible estimators (either using directly E_cut or using lamda=1/E_cut). I wonder what is used internally for the fitting of that model in ctools/gammalib - from a quick look at the source code (GModelSpectralExpPlaw.cpp) I am not absolutely sure about that!

History

#1 - 06/22/2016 04:36 PM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

The parameter that is used is E_cut (what counts is in fact the gradient, hence you may check eval_gradients()).

#2 - 06/22/2016 05:52 PM - Ziegler Alexander

Thanks for your answer - I had a look at eval_gradients() and found the calculations.
Just because of interest - would it be much work to implement a second model which uses lambda=1/E_cut instead of E_cut?

#3 - 06/22/2016 06:41 PM - Knödlseder Jürgen

I don't believe that the work would be large. Why don't you give it a try?

#4 - 06/22/2016 07:06 PM - Ziegler Alexander

I will probably do that!

#5 - 06/23/2016 12:22 AM - Knödlseder Jürgen

- Tracker changed from Support to Action

- Subject changed from fit parameters Exponentially cut-off power law to Implement alternative exponentially cut-off power law spectral model

- Assigned To changed from Knödlseder Jürgen to Ziegler Alexander

#6 - 06/30/2016 04:25 PM - Ziegler Alexander

- % Done changed from 0 to 50

I have started to implement the alternative cutoff model and already used it for some analysis - everything seems to work fine.
I just have a minor question:
For calculating the derivatives in the eval_gradients(), why is there the derivative with respect to the factor_values needed (at first, I missed the comment and hence, also the additional multiplication with the scale factor)? Does that mean that the parameters which are really varied are just the factor_values?
I also implemented and registered the swig file for the python wrapping.
Do you want the model to go in the software? What else is needed to be done then?

#7 - 07/01/2016 12:50 AM - Knödlseder Jürgen

Indeed, what is varied are only the factor values, and the gradients have to be computed with respect to these values.

The reasoning is that for a fit to be well behaved it is best to have all parameters of the same order. This is achieved by factorising each value into a factor value and a scale.

I definitely would like the model to go in the software. What's the name that you have given the model?

You should also add a unit test for the model. This is done in test_GModel.cpp. You should add an XML file to test/data and then add a method that tests the new class as much as possible (you can inspire the method from TestGModel::test_eplaw for example, but don't hesitate to add more tests).

You should also document the new model. This is done in doc/source/users/user_manual/modules/model.rst.

#8 - 07/01/2016 08:14 AM - Ziegler Alexander

> I definitely would like the model to go in the software. What's the name that you have given the model?

I named the model GModelSpectralExpPlaw2, type is ExpCutoff2. Instead of the parameter Cutoff, there is the parameter lambda defined as 1/cutoff-energy.
I will try to add the unit test and documentation as soon as possible, thanks for the explanations.

#9 - 07/01/2016 05:14 PM - Ziegler Alexander

- % Done changed from 50 to 70

progress:

- added xml file to test/data
- added unit test (same tests as in TestGModel::test_eplaw)
- added model to sphinx docu
- updated model doxygen docu

Furthermore, I shortly tested the flux_kernel() and e_flux_kernel() classes/calculations by comparing the results with the original ExpPlaw-model.

Remaining todo: check everything related to mc simulation.

#10 - 07/04/2016 04:09 PM - Ziegler Alexander

- % Done changed from 70 to 90

I'm nearly done with the checks - last one testing mc data production and analysis of them is running at the moment.
I have few questions:

- I tried to checkout my GitLab gammalib fork via ssh and didn't succeed - is this not supported (just for interest- I didn't get why it is not working, and used https instead)?
- how to handle the authorship of the new files (GModelSpectralExpPlaw2.hpp, GModelSpectralExpPlaw2.cpp, GModelSpectralExpPlaw2.i)? I added new code to these new files, but also a lot was just taken from the model GModelSpectralExpPlaw which was the start point for development.
- what to do when I'm ready for pull-request (fork with new branch is already created)? Just leave again a message here (the status tracker only offers 'In Progress' or 'Closed')? What will happen then - code review from your side and feedback to me?

#11 - 07/18/2016 10:58 PM - Knödlseder Jürgen

- Target version set to 1.2.0

Sorry for the late response.

user#190 wrote:

> I'm nearly done with the checks - last one testing mc data production and analysis of them is running at the moment.
> I have few questions:
>
> - I tried to checkout my GitLab gammalib fork via ssh and didn't succeed - is this not supported (just for interest- I didn't get why it is not working, and used https instead)?

Indeed, only https is supported.

> - how to handle the authorship of the new files (GModelSpectralExpPlaw2.hpp, GModelSpectralExpPlaw2.cpp, GModelSpectralExpPlaw2.i)? I added new code to these new files, but also a lot was just taken from the model GModelSpectralExpPlaw which was the start point for development.

These are new files that were initiated by yourself, hence you should be the author of these files.

> - what to do when I'm ready for pull-request (fork with new branch is already created)? Just leave again a message here (the status tracker only offers 'In Progress' or 'Closed')? What will happen then - code review from your side and feedback to me?

Just move the status to "Pull Request". You were still listed as "Reporter" in Redmine, that's probably the reason why you had no other choices. I changed this now. Put in the issue the branch name, I will then have a look.

By the way: I'm about to release version 1.1 of GammaLib, and by experience it's better not to merge last minute changes in the release. I will put the target version of your feature thus to 1.2.

#12 - 07/20/2016 09:36 AM - Ziegler Alexander

- Status changed from In Progress to Pull request

- % Done changed from 90 to 100

Ready for pull request - new model was tested and already used in analysis by me.
For the tests I had a closer look, one sees that the optimizer results using the new spectral model (type='ExpCutoff2') in the same minimum as for the original exponential cut-off power law parametrization (type='ExpCutoff'), the optimized function value given in the log file of ctlike is identical - I think this is also what should be observed.

Branch name is: **ExpPlaw2**

#13 - 07/21/2016 09:59 AM - Ziegler Alexander

- % Done changed from 100 to 90

Current status: code was reviewed, few adaptions need to be done.

#14 - 07/25/2016 11:08 AM - Ziegler Alexander

Just realized that using lambda as parameter/function name is for python wrapping cause of the built-in lambda functions not optimal - there are minor problems in the swig wrapper file, e.g. the getters/setters for lambda are replaced/prepended by default to _lambda.
I think best solution is to change allover to another parameter name for the cutoff parameter, e.g. alpha.

#15 - 07/25/2016 12:37 PM - Knödlseder Jürgen

user#190 wrote:

> Just realized that using lambda as parameter/function name is for python wrapping cause of the built-in lambda functions not optimal - there are minor problems in the swig wrapper file, e.g. the getters/setters for lambda are replaced/prepended by default to _lambda.
> I think best solution is to change allover to another parameter name for the cutoff parameter, e.g. alpha.

Right, lambda is a problem in Python. How about a more "understandable" name than alpha (which is probably never used for this purpose)? For example InverseCutoffEnergy in the XML file and then inverse_cutoff() as method?

#16 - 07/25/2016 01:23 PM - Ziegler Alexander

> Right, lambda is a problem in Python. How about a more “understandable” name than alpha (which is probably never used for this purpose)? For example InverseCutoffEnergy in the XML file and then inverse_cutoff() as method?

I also thought about that, but I think, to be consistent with the usage up till now, the name of the parameter in the code should be the same as used for the method - otherwise it gets more confusing, or not ?
We have to get basically 3 different namings for one parameter in a proper format:
parameter name used in code / parameter name in the xml-file / names used for the methods on the parameter (intuitively I would relate them rather to the parameter name used in the code if that one is different to the xml-specifier).

The problem is complicated/ too long parameter names blow up the code. I think its easier with short parameter names. For the moment I already switched to alpha and as specifier in the xml-file it is used with capital letter. But if you have a good suggestion/solution I am willing to change again.

#17 - 07/25/2016 02:16 PM - Knödlseder Jürgen

user#190 wrote:

> Right, lambda is a problem in Python. How about a more “understandable” name than alpha (which is probably never used for this purpose)? For example InverseCutoffEnergy in the XML file and then inverse_cutoff() as method?

> I also thought about that, but I think, to be consistent with the usage up till now, the name of the parameter in the code should be the same as used for the method - otherwise it gets more confusing, or not ?

In principle yes, but we deviate already from that in some cases. For example in the Power Law the XML name is PivotEnergy while the method is called pivot. The same is true for other cases. And in particular this is now also true for the Fermi/LAT legacy format.

Changing the method names has potentially an important impact because it breaks the code. The XML changes do not really break anything since there is the legacy support.

> We have to get basically 3 different namings for one parameter in a proper format:
> parameter name used in code / parameter name in the xml-file / names used for the methods on the parameter (intuitively I would relate them rather to the parameter name used in the code if that one is different to the xml-specifier).
>
> The problem is complicated/ too long parameter names blow up the code. I think its easier with short parameter names. For the moment I already switched to alpha and as specifier in the xml-file it is used with capital letter. But if you have a good suggestion/solution I am willing to change again.

There is indeed always the tension between understandability and readability of the code. The first would push for longer names since they are more explicit, while the second pushes for shorter name so that they simply fit on your screen.

The way how it's implemented is using longer names for XML files (which are the most likely way how a user will specify parameters), and the shorter names for the methods, which are likely not very often used.

#18 - 07/25/2016 03:01 PM - Ziegler Alexander

> The way how it's implemented is using longer names for XML files (which are the most likely way how a user will specify parameters), and the shorter names for the methods, which are likely not very often used.

I will then switch to the following scheme:
xml identifier: InverseCutoff (in argreement with the identifier for the parametrization with cutoff-energy, here it is just Cutoff)
code paramter: alpha
methods (getter/setter): inverse_cutoff()

Though I think that nevertheless this might be confusing, with a 'new parameter lambda' appearing in the formula when looking in the code/ doxygen docu for the first time. But I also have no better ideas.

#19 - 07/25/2016 03:18 PM - Knödlseder Jürgen

user#190 wrote:

> The way how it's implemented is using longer names for XML files (which are the most likely way how a user will specify parameters), and the shorter names for the methods, which are likely not very often used.

> I will then switch to the following scheme:
> xml identifier: InverseCutoff (in argreement with the identifier for the parametrization with cutoff-energy, here it is just Cutoff)

I changed Cutoff to CutoffEnergy (upon request from Christopher Deil, and I agree with him that this is more understandable).

> code paramter: alpha

Can't you keep lambda in the code (it is not exposed to Python)?

> methods (getter/setter): inverse_cutoff()
>
> Though I think that nevertheless this might be confusing, with a 'new parameter lambda' appearing in the formula when looking in the code/ doxygen docu for the first time. But I also have no better ideas.

Agree, but we just have to say what this lambda is, see for example
http://cta.irap.omp.eu/ctools-devel/users/user_manual/getting_started/models.html#exponentially-cut-off-power-law.

#20 - 07/25/2016 03:48 PM - Ziegler Alexander

Agreed on your proposals above, will do the implementation.

#21 - 07/25/2016 03:52 PM - Knödlseder Jürgen

By the way: you may check the latest code from the devel branch. We now can distinguish spectral models by the parameter names, so that variants with different parameters can have the same model name. Parameter names can now be specified upon construction, which allows appending the required variants into the registry (you can compare for example how GModelSpectralPlaw and GModelSpectralPlaw2 work now).

#22 - 07/26/2016 09:37 AM - Ziegler Alexander

The model is now updated according to our discussions. Major changes are:

- new naming scheme:
  classname: GModelSpectralExpPlaw2 > GModelSpectralExpInvPlaw
  type (xml): ExponentialInverseCutoffPowerLaw
- additional utility functions/value constructor supporting direct usage of cutoff energy
- use helper functions from GTools for xml read-in/write-out

  We now can distinguish spectral models by the parameter names, so that variants with different parameters can have the same model name. Parameter names can now be specified upon construction, which allows appending the required variants into the registry (you can compare for example how GModelSpectralPlaw and GModelSpectralPlaw2 work now).

That sounds interesting, when I find time, I will have a look at it.

#23 - 07/26/2016 09:55 AM - Knödlseder Jürgen

Thanks a lot.

Just for information: I will put out release 1.1 probably tonight (just waiting for the science verification to finish), hence the model will go in release 1.2. But once release 1.1 is out I'm ready to merge the code into devel.

#24 - 07/28/2016 11:02 PM - Knödlseder Jürgen

I would be ready to merge the code in, could eventually to the last tweeks if you don't find time. Just let me know which branch to use.

#25 - 07/29/2016 08:54 AM - Ziegler Alexander

Branch is **ExpPlaw2**, everything should be up to date - I also tested the model again shortly after the very last changes. Thanks!

#26 - 07/29/2016 11:21 AM - Knödlseder Jürgen

- Status changed from Pull request to Closed

- % Done changed from 90 to 100

Code has been adapted to new interface and integrated into devel.

Please note that the new model has also the type ExponentialCutoffPowerLaw, the distinction is made through the parameter names:

```
<?xml version="1.0" standalone="no"?>
<source_library title="source library">
  <source name="Crab" type="PointSource">
    <spectrum type="ExponentialCutoffPowerLaw">
      <parameter name="Prefactor"          scale="1e-16" value="5.7"  min="1e-07" max="1000.0" free="1"/>
      <parameter name="Index"              scale="-1"    value="2.48" min="0.0"   max="+5.0"   free="1"/>
      <parameter name="InverseCutoffEnergy" scale="1e-6"  value="1.0"  min="0.0"   max="100.0"  free="1"/>
      <parameter name="PivotEnergy"        scale="1e6"   value="0.3"  min="0.01"  max="1000.0" free="0"/>
    </spectrum>
    <spatialModel type="SkyDirFunction">
      <parameter name="RA"  scale="1" value="83.6331" min="-360" max="360" free="0"/>
      <parameter name="DEC" scale="1" value="22.0145" min="-90"  max="90"  free="0"/>
    </spatialModel>
  </source>
</source_library>
```

#27 - 07/29/2016 12:20 PM - Ziegler Alexander

Great, thanks!