GammaLib - Action #773

Feature # 692 (Closed): Perform an extensive interface review of all classes

Review model module classes

02/20/2013 03:14 AM - Knödlseder Jürgen

Status:	Closed	**Start date:**	02/20/2013
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	20.00 hours
Target version:	00-08-00		

Description

Related issues:

Related to GammaLib - Action # 693: Review interface for GModelSpectral classes	**Closed**	**01/12/2013**

History

#1 - 02/20/2013 03:15 AM - Knödlseder Jürgen

The following classes have been reviewed:

Class	Comment
GModels	Derive from GContainer, implement container methods.
GModelPar	Revise interface completely, add autoscale() method, assure non-zero scale factor.
GModel	Return name by const reference
GModelSky	value and gradients take GPhoton as argument, inline declarations, improved documentation
GModelSpatial	Add autoscale() method, pass GPhoton to eval and eval_gradients methods, and GEnergy and GTime to mc method, inline declarations
GModelSpatialDiffuse	Pass GPhoton to eval and eval_gradients methods, and GEnergy and GTime to mc method
GModelSpatialDiffuseConst	Pass GPhoton to eval and eval_gradients methods, and GEnergy and GTime to mc method, inline declarations
GModelSpatialDiffuseCube	Pass GPhoton to eval and eval_gradients methods, and GEnergy and GTime to mc method, inline declarations
GModelSpatialDiffuseMap	Pass GPhoton to eval and eval_gradients methods, and GEnergy and GTime to mc method, inline declarations
GModelSpatialElliptical	Pass GEnergy and GTime to eval, eval_gradients and mc methods, inline declarations
GModelSpatialRadial	Pass GEnergy and GTime to eval, eval_gradients and mc methods, inline declarations
GModelSpectralConst	Add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, add value constructor and access methods, rename norm to value
GModelSpectralExpPlaw	Remove autoscale() method, auto-scale in parameter constructor, complete value constructor, add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, implement parameter access methods (inline), add pre computation cache for eval and eval_gradients methods, rename norm to prefactor and ecut to cutoff

GModelSpectralFunc	Complete value constructor, add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, implement parameter access methods (inline)
GModelSpectralPlaw	Remove autoscale() method, auto-scale in parameter constructor, complete value constructor, add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, implement parameter access methods (inline), rename norm to prefactor
GModelSpectralPlaw2	Complete value constructor, add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, implement parameter access methods (inline), emin() and emax() methods now return / take GEnergy
GModelSpectralNodes	Add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, implement parameter access methods and node manipulation methods
GModelSpectralLogParabola	Remove autoscale() method, auto-scale in parameter constructor, complete value constructor, add time argument to eval, eval_gradients and mc methods, remove const from eval_gradients, implement parameter access methods (inline), pivot() methods now return / take GEnergy, rename norm to prefactor
GModelTemporal	Remove const from eval_gradients
GModelTemporalConst	Add value contructor, remove const from eval_gradients, implement parameter access methods (inline)
GModelData	Nothing
GModelRegistry	size method inline
GModelSpatialRegistry	size method inline
GModelSpectralRegistry	size method inline
GModelTemporalRegistry	size method inline

#2 - 02/20/2013 03:26 AM - Knödlseder Jürgen

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

- % Done changed from 0 to 10

#3 - 02/20/2013 03:26 AM - Knödlseder Jürgen

- Remaining (hours) changed from 20.0 to 16.0

#4 - 02/21/2013 01:18 AM - Knödlseder Jürgen

- Remaining (hours) changed from 16.0 to 15.0

The GModels class access operator now returns models by reference instead of returning pointers.

#5 - 02/21/2013 07:00 AM - Knödlseder Jürgen

Jürgen Knödlseder wrote:

> The GModels class access operator now returns models by reference instead of returning pointers.

I just recognized that this was stupid. I reintroduced the code slicing problem. Instead, all container classes holding pointers of base classes should have pointer access operators! See #517.

#6 - 02/22/2013 12:24 AM - Knödlseder Jürgen

The GModels method should have a hasmodel() method that checks whether a model with a specific name is present in the container. The method should be inspired from the GPars::haspar() method.

In general, container classes that allow access by name should have a hasXXX method to check whether a specific named object is present.

#7 - 03/17/2013 10:19 PM - Knödlseder Jürgen

A generic autoscale() method should be added to a model components. The purpose of the autoscale() method is to set the scale factors to the real value and the value to 1, so that all model parameters are scaled using their actual values.

The best is to add an autoscale() method at the GModelPar level, so that each parameter can be autoscaled.

#8 - 03/17/2013 10:35 PM - Knödlseder Jürgen

- % Done changed from 10 to 20

Jürgen Knödlseder wrote:

> A generic autoscale() method should be added to a model components. The purpose of the autoscale() method is to set the scale factors to the real value and the value to 1, so that all model parameters are scaled using their actual values.
>
> The best is to add an autoscale() method at the GModelPar level, so that each parameter can be autoscaled.

An autoscale() method has been added to GModelPar and to the GModelSpatial, GModelSpectral and GModelTemporal classes. This method should be called systematically in the parameter constructors.

#9 - 03/19/2013 12:42 PM - Knödlseder Jürgen

- % Done changed from 20 to 50

- Remaining (hours) changed from 15.0 to 10.0

The GModelPar interface has been fully revised. This impacted many of the model classes.

The modification has been tested, and ctools compliance checked. Now the road is paved towards a review of all other interfaces.

#10 - 03/23/2013 09:53 PM - Knödlseder Jürgen

I just recognized a major problem with the GModelSpatialDiffuseCube class: this class does not only represent a spatial model, but a spatio-spectral model. As GModelSpatialDiffuseCube has the interface of a spatial model only, it does not allow energy dependent model evaluation. In fact, GModelSpatialDiffuseCube is not a component of a fully factorized model, it is the first example of a non-factorized model.

How to deal with that? Change the class inheritance scheme where GModelSky is only the base class and then we have derived classes for factorized or other model types? Or making GModelSky deal with all model types?

Or should we treat GModelSpatialDiffuseCube as a special case, because it is a spatio-spectral model but coexists with a spectral model in the Fermi-LAT XML interface?

The problem occurs in the response computation, which explicitly uses the spatial model's eval method with the sky direction as single parameter for model computation. The source energy is available at this point, so it could be used for computation.

It is interesting to recognized that the GModelSky::spatial method does not really relies on the fact that the model is factorized, except for the fact that the gradients need the correct pre-factors. This means that the irf method could in fact return the full response, and not only that of the spatial component.

The entire interfaces were in fact thought for a factorized sky model, and now we have a clear example for a non-factorized sky model.

A created action #808 to rethink the interface for non-factorized sky models.

#11 - 03/26/2013 12:40 AM - Knödlseder Jürgen

- % Done changed from 50 to 70

Finished the spatial classes.

#12 - 04/02/2013 10:01 PM - Knödlseder Jürgen

Jürgen Knödlseder wrote:

> The GModels method should have a hasmodel() method that checks whether a model with a specific name is present in the container. The method should be inspired from the GPars::haspar() method.

In general, container classes that allow access by name should have a hasXXX method to check whether a specific named object is present.

Done!

#13 - 04/02/2013 10:02 PM - Knödlseder Jürgen

- Status changed from In Progress to Feedback

- % Done changed from 70 to 100

- Remaining (hours) changed from 10.0 to 0.0

The model class review is now finished.

#14 - 12/03/2013 09:59 AM - Knödlseder Jürgen

- Status changed from Feedback to Closed