GammaLib - Feature #1520

GModels for Dark Matter Halo Density Profiles

08/07/2015 07:06 PM - Kelley-Hoskins Nathan

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assigned To:	Kelley-Hoskins Nathan	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:			

Description

As dark matter studies are a significant part of gamma-ray astrophysics research (and my thesis :p), it would be nice to have one or several GModelSpatialRadial models for DM Halos. This feature should at least include the popular density profile:

- Zhao(Hernquist/NFW) Profile (which becomes a NFW profile with alpha=1,beta=3,gamma=1)

There are other profiles that have been used in literature as well:
- Einasto Profile
- Burkert Profile

Their density profiles are in the attached picture.

This feature should also let the user choose the interaction type, Self-Annihilation or Decay. This changes how the density profile is integrated when computing the J factor. (See the J factors in the attached picture). I believe the J factor is more or less what the GModelSpatialRadial::eval() function is calculating?

As these are all spherically-symmetric halos at some 'distance to halo center', and due to the potential combinations of Profile/Interaction, my first suggestion is to put all DM profiles into one GModelSpatialRadialDarkMatterHalo, and have model parameters for switching between profiles and interaction types. Python usage would then look something like this:

```
model = gammalib.GModelSpatialRadialDarkMatterHalo()
model['profile' ] = 'zhao'  # or einasto or burkert
model['interact'] = 'selfannihilation' # or decay
model['halodist'] = 2.5e20  # distance from Earth to halo center, in meters (8kpc to Galactic Center = 2.5e20meters)
model['scale_radius'] = ### # r_s, meters
model['scale_density'] = ### # \rho_s, density at the scale radius, g/cm^3 ?

# if zhao profile
model['alpha'] = 1 # float
model['beta' ] = 3 # float
model['gamma'] = 2 # float

# if einasto profile
model['alpha'] = 0.17 # float
```

Another possibility is to have separate GModelSpatialRadial for each profile and interaction type, e.g.
- GModelSpatialRadialZhaoAnnihilation
- GModelSpatialRadialZhaoDecay
- GModelSpatialRadialEinastoAnnihilation
- GModelSpatialRadialEinastoDecay
- GModelSpatialRadialBurkertAnnihilation
- GModelSpatialRadialBurkertDecay

Or, another alternative is :
+ use GModelSpatialRadial only for subclassing radially-symmetric-in-skycoordinates models, and
+ make a GModelSpatialSpherical base class for all spherically-symmetric-in-space models at some 'halo distance' from earth, and put each of these Profile/Interaction combinations into a separate subclass of this GModelSpatialSpherical.

I think these alternatives will take more code and time to develop, which is why I'd prefer the earlier GModelSpatialRadialDarkMatterHalo idea.

Does anyone have any feedback or suggestions on how this feature should be organised?
Any suggestions for additional profiles?
And when I add this feature, what is the GModelSpatialRadial::eval() function calculating (Can I use the J-factor or normalized J-factor?)

History

#1 - 08/13/2015 11:37 AM - Mayer Michael

I guess you would also have to think about the XML interface. I am not familiar with these models, but it should be possible to access each model parameter from the XML file, too. I therefore think it might be worth to have individual classes for each model.
Is it common to fit the extension or position of these models? If not wouldn't it be easier to just use FITS templates?

#2 - 08/13/2015 02:29 PM - Kelley-Hoskins Nathan

I was hoping to keep the profile and interaction values as model parameters (and not private variables), though always fixed, to make sure they are naturally saved to/read from the xml files.

Its not common to fit the position/halodist, since these are usually known beforehand through measuring star or galactic rotation velocities.

The extension (scale radius/density,exponents) is usually what's being searched for, to improve DM upper limits, so it'd be nice to give the likelihood engine the freedom to play with those parameters, rather than using a static fits image that has to be changed every time you want to test a different scale parameter.

I guess a better example would be:

```
model = gammalib.GModelSpatialRadialDarkMatterHalo('zhao','selfannihilation') # sets and fixes these model parameters
model['profile'      ].string('einasto') # though you can still change them to something else
model['interaction'  ].string('decay')
model['halodist'     ].real(2.5e20) # distance from Earth to halo center, in meters (8kpc to Galactic Center = 2.5e20meters)
model['ra'           ].real(44.44 ) # deg
model['dec'          ].real(44.44 ) # deg
model['scale_radius' ].real(2e14  ) # r_s, meters
model['scale_density'].real(1e-5  ) # \rho_s, density at the scale radius, g/cm^3 ?
model['alpha'        ].real(1     ) # float
model['beta'         ].real(3     ) # float, only zhao
model['gamma'        ].real(2     ) # float, only zhao
```

I guess I'd be worried about having 6 separate model cpp/hpp files with huge names (that are only used in DM studies), when their only significant difference is in the eval() function.

#3 - 08/20/2015 10:36 PM - Knödlseder Jürgen

I got today this request from Gonzalo Rodriguez Fdez. I guess it is related to this thread hence I post it here. I asked Gonzalo to sign-up and participate in the discussion.

Dear Jurgen,

In order to study the expected signal from dark matter within CTOOLS
we have to created several ascii files with the expected flux for
differences wimp mass and for difference annihilation channels. In
this way the analysis and the code con be a little messy.
I would like to suggest the solution adopted by the Fermi
collaboration in their science tools. They use the DMFit tool to
calculated the gamma spectrum, then from a xml file they just model
the spectrum using 4 parameters, normalization, channel, branching
ratio and mass.

Do you think that this solutions can be implemented in CTOOLS?

Regards,
Gonzalo.

#4 - 08/21/2015 08:11 AM - Doro Michele

Hi all,

few comments:
1/ I prefer one unique function with parameters for DM density
2/ The integration of NFW profiles (divergent at the center) can be tricky, and needs some careful attention
3/ There is a public code (Clumpy) with many such features, including intrinsic boost from substructures, parameterization of DM ann/decay spectrum, etc, (c++), I wonder whether it can be linked to CTOOLS or a standalone code is better
4/ In the end of 2015/begin of 2016, several DM papers from CTA are expected. It would be good to have a complete cross-check with CTOOLS.

Thanks for the support!
Michele

#5 - 08/21/2015 03:32 PM - Rodriguez Fernandez Gonzalo

Here, Fermi source model, you can find how the dark matter energy distribution is defined in the sciences tools definition files.

Thanks,
Gonzalo.

#6 - 08/21/2015 04:16 PM - Knödlseder Jürgen

Thanks for the link.

So you are talking in fact about a spectral model (the thread was more for a spatial model). I guess we need both, spectral and spatial model components.

The spectral model from DMFit looks similar to a file function, just a bit more sophisticated. Is the format of the data file defined somewhere? Is it a fixed format (number of rows and columns fixed) or can this evolve?

#7 - 08/22/2015 10:49 AM - Doro Michele

Hi,

the Cirelli et al DM spectral have become quite standard for now. They have fixed columns and rows. And they should not change in the future often. Possibly new channels/new models may appear or some corrections.

Michele

#8 - 08/23/2015 09:17 PM - Knödlseder Jürgen

Doro Michele wrote:

> Hi,
>
> the Cirelli et al DM spectral have become quite standard for now. They have fixed columns and rows. And they should not change in the future often. Possibly new channels/new models may appear or some corrections.
>
> Michele

Wouldn't it make sense to transform this ASCII file into a FITS file?

In that way metadata can be added to the header, and one would also easily cope with a possible evolution of the model.

#9 - 08/24/2015 07:59 AM - Doro Michele

Hi Juergen,

I am not accustomed to .fits format, so I cannot judge the complexity of doing it. But, being just a table of rows and columns with some headers probably, I guess it could be easily done.

M

#10 - 08/24/2015 11:01 AM - Rodriguez Fernandez Gonzalo

- File gammamc_dif.dat added

- File test.py added

Hi Juergen,

In fact the implementation in Fermi tools is just a ASCII file with the dNdE for difference X mass and channels.

Gonzalo.

#11 - 01/05/2016 02:11 PM - Kelley-Hoskins Nathan

- Assigned To set to Kelley-Hoskins Nathan

- Start date deleted (08/07/2015)

#12 - 01/20/2016 05:29 PM - Kelley-Hoskins Nathan

- File Screenshot 2016-01-20 17.23.14.png added

Well, it turns out the integral you need to calculate the J factor does not have an analytical solution. I think this means every time eval() is called, its going to have to numerically integrate that function, which is going to be slow. I see a GFunction and GIntegral, I imagine I can use these to do the integration in this model's eval()?

#13 - 01/22/2016 04:17 PM - Knödlseder Jürgen

user#111 wrote:

> Well, it turns out the integral you need to calculate the J factor does not have an analytical solution. I think this means every time eval() is called, its going to have to numerically integrate that function, which is going to be slow. I see a GFunction and GIntegral, I imagine I can use these to do the integration in this model's eval()?

Yes, you can use GFunction and GIntegral for that.

Is it possible to pre-compute things and put them into a cache (that's how it works for some of the other models)?

#14 - 02/15/2016 04:44 PM - Kelley-Hoskins Nathan

- File CheckDMHaloRadius_NewGModel.pdf added

It seems theres no sane way to separate the dm halo flux into spectral and spatial components with the right units (spatial=sr^1, spectral=cm^-2*s^-1*MeV^-1). My first thought was to put the integral into a GModelSpatial, but the integral ends up with units of cm*sr^-1, so I don't think I can do that, and theres no other component with units 1/cm I can move with the integral.

I guess I need to instead put in a separate GModelDarkMatterHalo class, rather than trying to build one from Spectral/Spatial components.

#15 - 02/16/2016 10:10 AM - Knödlseder Jürgen

I think there is no fundamental problem, you just need to get the units right.

The trick is to get a spatial template that normalizes to unity when integrated over the solid angle, so that the spectral component provides the flux density.

You can achieve this for any spatial template {{latex($f(l, \Omega)$)}} (in whatever units it is) by computing {{latex($Spatial = \frac{\int_l f(l, \Omega) dl}{\int_\Omega \int_l f(l, \Omega) dl d\Omega}$)}}

#16 - 02/16/2016 11:32 AM - Kelley-Hoskins Nathan

My flux equation is this: {{latex($\frac{d \Phi}{dE d\Omega} = \frac{\left \langle \sigma v \right \rangle}{8\pi m_\chi^2}\rho_s^2 * \frac{dN}{dE} * \int_{l=0}^{\infty} \left (\frac{ \sqrt{l^2+d^2-2 l d Cos(\theta) } }{r_s} \right)^{\left (1+ \left (\frac{ \sqrt{l^2+d^2-2ldCos(\theta)}} }{r_s} \right)\alpha \right)^{ -\frac{2(\beta-\gamma)}{\alpha}} dl$)}}

The I don't think there is a component of that equation that can be split off that just has the units 1/sr.

> get a spatial template that normalizes to unity when integrated over the solid angle

I'm not sure what is meant by this, or what the advantage is? Doesn't this mess with the physical meaning of the above flux equation? And what would the limits of integration be for the solid angle integral?

#17 - 02/16/2016 03:56 PM - Kelley-Hoskins Nathan

{{latex($Spatial = \frac{\int_l f(l, \Omega) dl}{\int_\Omega \int_l f(l, \Omega) dl d\Omega}$)}}

Ah, I think I'm starting to understand. I'll try this.

#18 - 02/16/2016 06:57 PM - Kelley-Hoskins Nathan

Is there a 2D integrator hidden somewhere in gammalib? It looks like GFunction/GIntegral only take one input parameter for integration. The integrand {{latex($f(l,\theta)sin(\theta)$)}} has no analytical integral with respect to {{latex(l)}} or {{latex(θ)}} ...

{{latex($f(g(l,d,\theta,r_s),\alpha,\beta,\gamma) = g^{ - 2 \gamma } {\left(1 + g^\alpha \right) }^{ -2 \frac{ \beta - \gamma }{ \alpha } } $)}}

where

{{latex($g(l,d,\theta,r_s)= \frac{\sqrt{l^2+d^2-2ldCos(\theta) }}{r_s} $)}}

#19 - 02/16/2016 07:50 PM - Knödlseder Jürgen

user#111 wrote:

Is there a 2D integrator hidden somewhere in gammalib? It looks like GFunction/GIntegral only take one input parameter for integration. The integrand {{latex($f(l,\theta)sin(\theta)$)}} has no analytical integral with respect to {{latex(l)}} or {{latex(θ)}} ...

{{latex($f(g(l,d,\theta,r_s),\alpha,\beta,\gamma) = g^{ - 2 \gamma } {\left(1 + g^\alpha \right) }^{ -2 \frac{ \beta - \gamma }{ \alpha } } $)}}

where

{{latex($g(l,d,\theta,r_s)= \frac{\sqrt{l^2+d^2-2ldCos(\theta) }}{r_s} $)}}

For 2D you just nest a 1D integral within another (using the GFunction/GIntegral classes).

#20 - 04/15/2017 08:55 PM - Kelley-Hoskins Nathan

- Status changed from New to Pull request

I've spent maybe 12 hours the past week (unsuccessfully) trying to rebase all my changes with updates, and I'm starting to drive myself a little crazy. Can my nkelhos/gammalib/1520-dm-profiles branch be pulled into gammalib/gammalib/1520-dm-profiles, so I can start from there?

The classes themselves compile and work, but their profiles still need to be checked for scientific validity.

#21 - 06/05/2018 02:31 PM - Knödlseder Jürgen

- Status changed from Pull request to Closed

- % Done changed from 0 to 100

Models are merged in since quite some time.

Files

Dark Matter Profiles and J-factor Interactions.pdf	606 KB	08/07/2015	Kelley-Hoskins Nathan
gammamc_dif.dat	914 KB	08/24/2015	Rodriguez Fernandez Gonzalo
test.py	1.89 KB	08/24/2015	Rodriguez Fernandez Gonzalo
Screenshot 2016-01-20 17.23.14.png	73 KB	01/20/2016	Kelley-Hoskins Nathan
CheckDMHaloRadius_NewGModel.pdf	598 KB	02/15/2016	Kelley-Hoskins Nathan