GammaLib - Action #3625

Implement a more efficient Monte-Carlo sampler for exponentially and super-exponentially cut-off power laws

04/27/2021 11:16 PM - Knödlseder Jürgen

Status:	Closed	**Start date:**	04/27/2021
Priority:	Normal	**Due date:**	
Assigned To:	Knödlseder Jürgen	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	1.7.4		

Description

As reported in #3624, the super-exponentially cut-off power law Monte Carlo sampler is not very efficient when sampling deeply in the exponentially cut-off part of the power law. A more efficient sampler should be implemented to speed-up the computations.

Related issues:

Related to ctools - Bug # 3624: bug on csspec with Composite model	**Closed**	**04/27/2021**

History

#1 - 04/27/2021 11:17 PM - Knödlseder Jürgen

- Related to Bug #3624: bug on csspec with Composite model added

#2 - 04/27/2021 11:59 PM - Knödlseder Jürgen

- File mc-superexp-after-10000.png added

- File mc-superexp-before-10000.png added

- Status changed from New to In Progress

- Assigned To set to Knödlseder Jürgen

- % Done changed from 0 to 10

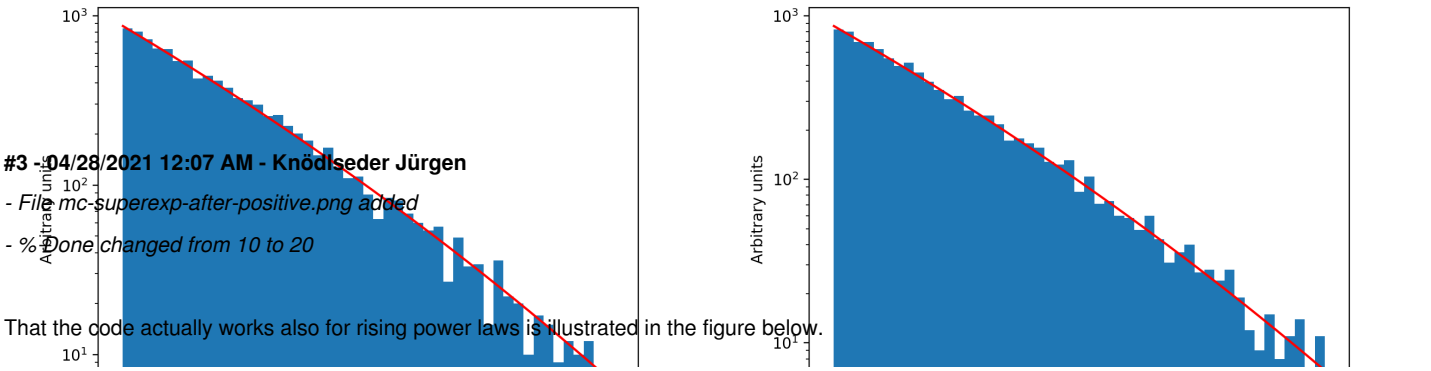
The Monte Carlo sampling in GModelSpectralSuperExpPlaw::mc is done using a rejection method that draws random energies for a power law and compares then the super-exponential cut-off model to the power law to decide whether to keep or to reject the energy. Since for the first energy bins the sampling is done deeply in the cut-off of the model, the energy is very often rejected before being accepted. Below some debugging of the Monte Carlo method, where samples indicates the number of random energies that are drawn before one is accepted. This number can go up to millions!

```
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=35.7830766905839 GeV samples=786164
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=40.3034500835591 GeV samples=345242
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=46.6816418047913 GeV samples=116586
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=36.3483683005087 GeV samples=284844
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=37.2933148694958 GeV samples=170091
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=31.566807167513 GeV samples=71186
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=31.0769872610411 GeV samples=159230
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=31.3044072005203 GeV samples=45231
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=37.9303234539904 GeV samples=324109
GModelSpectralSuperExpPlaw::mc(30 GeV,72.3259983091833 GeV,0 s (TT)): energy=45.909979097982 GeV samples=6397
```

I implemented a speed-up by simply re-normalising the random number generator used for rejection:

```
double acceptance_fraction;
double inv_ecut  = 1.0 / m_ecut.value();
double norm_emin = std::exp(- std::pow(emin.MeV() * inv_ecut, m_index2.value()));
double norm_emax = std::exp(- std::pow(emax.MeV() * inv_ecut, m_index2.value()));
double norm      = (norm_emin > norm_emax) ? norm_emin : norm_emax;
do {
   ...
   acceptance_fraction = std::exp(- std::pow(eng * inv_ecut, m_index2.value()));
} while (ran.uniform() * norm > acceptance_fraction);
```

That this works is illustrated in the figures below, that show on the left the old results and on the rights the results after implementing the sampling with improved efficiency. The red line indicates the analytical model, the blue histogram the sampled energies.

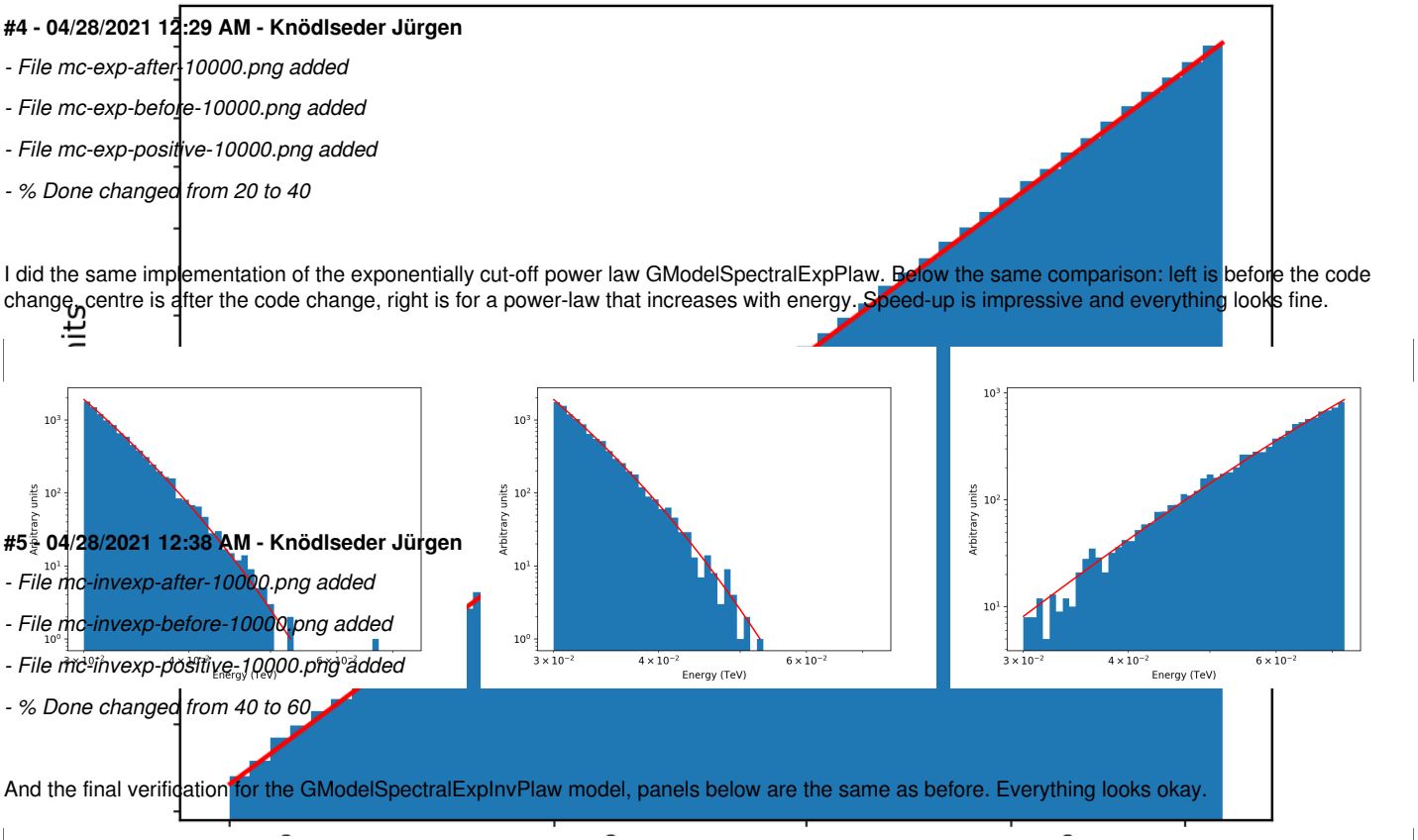


#3 - 04/28/2021 12:07 AM - Knödlseder Jürgen

- File mc-superexp-after-positive.png added

- % Done changed from 10 to 20

That the code actually works also for rising power laws is illustrated in the figure below.



#4 - 04/28/2021 12:29 AM - Knödlseder Jürgen

- File mc-exp-after-10000.png added

- File mc-exp-before-10000.png added

- File mc-exp-positive-10000.png added

- % Done changed from 20 to 40

I did the same implementation of the exponentially cut-off power law GModelSpectralExpPlaw. Below the same comparison: left is before the code change, centre is after the code change, right is for a power-law that increases with energy. Speed-up is impressive and everything looks fine.

#5 - 04/28/2021 12:38 AM - Knödlseder Jürgen

- File mc-invexp-after-10000.png added

- File mc-invexp-before-10000.png added

- File mc-invexp-positive-10000.png added

- % Done changed from 40 to 60

And the final verification for the GModelSpectralExpInvPlaw model, panels below are the same as before. Everything looks okay.

#6 - 04/28/2021 12:39 AM - Knödlseder Jürgen

- Target version set to 1.7.4

- % Done changed from 60 to 70

#7 - 04/29/2021 06:04 PM - Knödlseder Jürgen

- Status changed from In Progress to Closed

- % Done changed from 70 to 100

The code was merged into bugfix release 1.7.4 and devel. Close issue now.

Files

mc-superexp-after-10000.png	45.3 KB	04/27/2021	Knödlseder Jürgen
mc-superexp-before-10000.png	45.6 KB	04/27/2021	Knödlseder Jürgen
mc-superexp-after-positive.png	41 KB	04/27/2021	Knödlseder Jürgen
mc-exp-after-10000.png	42.4 KB	04/27/2021	Knödlseder Jürgen
mc-exp-before-10000.png	42.8 KB	04/27/2021	Knödlseder Jürgen
mc-exp-positive-10000.png	42.1 KB	04/27/2021	Knödlseder Jürgen
mc-invexp-after-10000.png	42.4 KB	04/27/2021	Knödlseder Jürgen
mc-invexp-before-10000.png	42.8 KB	04/27/2021	Knödlseder Jürgen
mc-invexp-positive-10000.png	40.7 KB	04/27/2021	Knödlseder Jürgen