GAMERA - A Modular Framework For Spectral Modeling In VHE Astronomy

J. Hahn[*]
Max-Planck-Institut für Kernphysik, P.O. Box 103980, D 69029 Heidelberg, Germany
E-mail: joachim.hahn@mpi-hd.mpg.de

GAMERA is a new open-source C++ package which handles the spectral modelling of non-thermally emitting astrophysical sources in a simple and modular way. It allows the user to devise time-dependent models of leptonic and hadronic particle populations in a general astrophysical context (including SNRs, PWNs and AGNs) and to compute their subsequent photon emission. Moreover, this package also contains the necessary tools to create Monte-Carlo population synthesis models. In this poster, I will explain the basic design concept of GAMERA and present several examples of its implementation.

The 34th International Cosmic Ray Conference,
30 July- 6 August, 2015
The Hague, The Netherlands

[*]Speaker.

1. Introduction

The interpretation of spectra in VHE astronomy often involves data modelling. Depending on the scope of the interpretation, typical spectral models differ vastly in sophistication: They may or may not include the time-evolution of particle spectra, time-dependent energy losses, multiple emission zones (either independent or interacting), particle propagation and escape, etc. Thus, from case to case the requirements on the modelling code may differ strongly, and models often have to be completely reinvented for every new project. This is where the new `GAMERA` library is intended to help: It is designed to provide the user with a modular library that allows it to put together spectral models in an easy way - from very basic, one-zone static models to more sophisticated, multi-zone time-dependent ones. In the following, the structure of the `GAMERA` library will be presented and its functionality will be demonstrated on a few examples.

2. The **GAMERA** library

2.1 Dependencies, License and Download

The `GAMERA` library is written in C++ and depends on the C++ standard libraries. Further dependencies are ROOT[1] and the GNU Scientific Library[2]. GAMERA is licenced under the GPL2.1 or later. It is available at `github.com/JoachimHahn/GAMERA`, where also coding examples and documentation can be found.

2.2 Classes

The `GAMERA` library currently consists of five separate classes. In this modular approach, objects may interact through Get and Set functions. A sketch on how these classes may be used together is shown in Fig. 1; the classes will be briefly introduced below.

- **`Particles:`** This class allows for the time-evolution of particle spectra $N(E)$. The classic continuity equation (see e.g. [1]) which governs this time evolution,

$$\frac{\partial N}{\partial t} = \frac{\partial}{\partial E}(P \cdot N) - \frac{N}{\tau} + Q, \tag{2.1}$$

  where P is the energy loss rate, τ is the catastrophic particle loss time-scale and Q is the particle source term, is solved numerically in a piece-wise linear scheme with several optional slope-limiters. This method allows for generic, time-dependent functions of P (e.g. due to Synchrotron losses in a changing B-Field), τ (e.g. changing ambient density and thus p-p collision time scales) and Q (e.g. quiescent and active phases of a central engine).

- **`Radiation:`** Here, photon spectra are calculated from spectra of electrons and hadrons. The available radiation mechanisms and their implementation are as follows: Synchrotron Radiation ([2],[3]), non-thermal Bremsstrahlung ([4]), Inverse-Compton (IC) Radiation ([2]) and Radiation from π^0-decay ([5]). The input to a `Radiation` class object may be a particle

[1] `root.cern.ch`

[2] `www.gnu.org/software/gsl/`

spectrum that has been previously calculated in the `Particles` class, which then constitutes a time-dependent model. Alternatively, a pre-defined particle spectrum can be used as the input, which then corresponds to a static model (see Fig. 1).

- **`Dynamics`:** A class that comprises frequently used dynamical models of typical VHE sources. For instance, for Supernova remnant blast waves it includes the classical Sedov solution and the model put forward by Truelove and McKee [6] as well as the Thin-Shell approximation (see e.g. [7]). Also, a simple cosmic ray diffusion model can be found in this class.
- **`Utils`:** This utilities class holds tools that are freqently used in population studies, like models on the Galactic spiral arm structure, gas distributions in the Milky Way, magnetic field models and also basic main-sequence wind structure models.
- **`InstrumentResponse`:** This class allows the user to compute synthetic data points from a simulated gamma-ray spectrum. To that end, instrument response functions (IRFs) have to be provided, but also some pre-implemented sets of IRFs may be used.

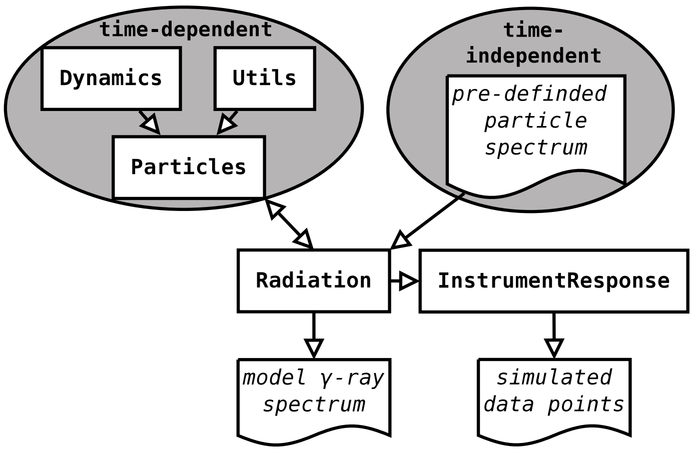

Figure 1: Possible structure of spectral models with GAMERA.

2.3 Multiple Zones

Due to the modular design of GAMERA, the addition of emission zones is straight-forward: It merely requires an additional instantiation of the aforementioned classes. Also, the interaction between zones is possible using Get and Set functions. A possible scheme is shown in Fig. 2.

2.4 Caveats

There are several short-comings of the GAMERA library. For instance, computational speed depends on the physics model. Since the step size in the iteration used in the `Particles` class depends on the energy loss rate, e.g. leptonic models with both high magnetic fields ($B > 1$ mG) and large values of the maximum particle energy ($E_{\max} > 1$ PeV) can become very slow. Also, it is

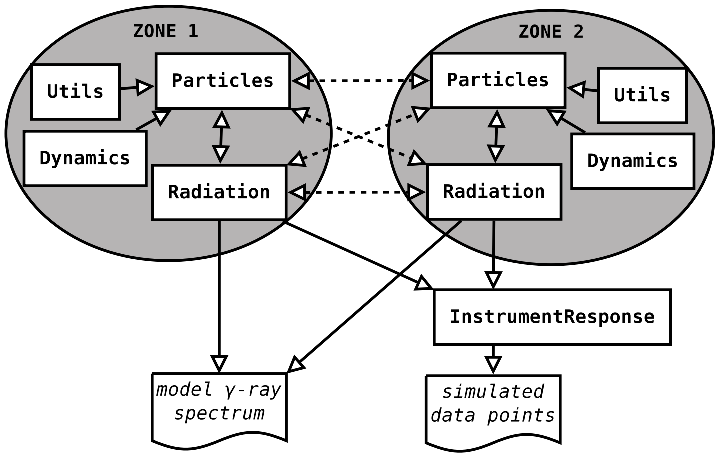

Figure 2: Possible structure of a time-dependent two-zone model with GAMERA. Interaction between zones is possible as indicated by the dashed arrows.

currently not possible to perform a simultaneous spatial and spectral particle evolution, like e.g. in the case of energy-dependent diffusion. This feature is planned for future versions of GAMERA. In the meantime, the application of multiple zones can circumvent this problem to some extent.

3. Example I: A Time-dependent Model for Young PWNe

The modular approach of GAMERA as well as its generic algorithm for the time-evolution of particle spectra allows for the straight-forward implementation of almost arbitrary spectral source models. Here, the GAMERA implementation of the pulsar wind nebula (PWN) model recently published by Torres et al. [8] will be demonstrated using the following set of parameters[3]:

$d = 5$ kpc	$n_H = 0.1$ cm^{-3}	$T_{FIR} = 70$ K	$\omega_{FIR} = 0.25$ eV/cm^3
$T_{NIR} = 5000$ K	$\omega_{NIR} = 0.5$ eV/cm^3	$n = 3$	$\tau_0 = 500$ yrs
$\dot{E}_0 = 3\times10^{39}$	$M_{ej} = 10$ M$_\odot$	$E_0 = 10^{51}$ erg	$\eta = 0.03$
$\varepsilon = 0.2$	$\gamma_b \times m_e = 300$ GeV	$\alpha_1 = 1.5$	$\alpha_2 = 2.5$

The first step is to code the time-dependent models of the electron injection power $Q(t)$, the magnetic field $B(t)$, the PWN dynamics $R(t), V(t)$ and the maximum electron energy $E_{max}(t)$ as given in the paper. This function fills a 2D-vector, holding row $\{t, Q(t), B(t), R(t), V(t), E_{max}(t)\}$ and is called `FillParameterVector` in this example[4]. The resulting vector is then given to the `Particles` class object, which will calculate the time-dependent energy losses due to Synchrotron radiation and adiabatic expansion, and it will attributio the proper value of the maximum

[3]using the same notation as in [8], see the publication for more details.

[4]The specific implementation of this function is independent of GAMERA and is thus not elaborated on here.

electron energy for the injection spectrum in each time step. Inverse Compton losses are derived from a look-up table that is created by the `Radiation` class. To that end, the ambient radiation fields have to be provided, either by grey-body photon distributions or by using an input file for arbitrary spectral shapes. In the aforementioned model [8], the ambient density is a constant and can be simply specified by a Set function which corresponds to fixing this value in all time steps. This density value is necessary for the calculation of Bremsstrahlung losses. The electron injection spectrum has a broken power-law shape in the model by Torres et al., so the corresponding parameters also have to be specified. It should be noted, however, that, at the cost of some computational speed, arbitrary injection spectra are supported.

The following lines show the necessary program code to perform the steps above:

```
Particles *fParticles = new Particles();
Radiation *fRadiation = new Radiation();
fParticles->SetParameterEvolutionLookup(FillParameterVector());
fRadiation->AddThermalTargetPhotons(T_CMB,eDens_CMB);
fRadiation->AddThermalTargetPhotons(T_FIR,eDens_FIR);
fRadiation->AddThermalTargetPhotons(T_NIR,eDens_NIR);
fParticles->SetICLossLookup(fRadiation->CreateICLossLookup());
fParticles->SetAmbientDensity(n)
fParticles->SetBreakEnergy(ebreak);
fParticles->SetLowSpectralIndex(indexlow);
fParticles->SetSpectralIndex(indexhigh);
fParticles->SetAge(age);
fParticles->ComputeParticleSpectrum("electrons");
```

In the top left panel of Fig. 3 the resulting electron spectra at several time steps are shown.

The next step is to calculate the radiation spectra emitted by these electrons, which is done in the `Radiation` class. To that end, the ambient density (for Bremsstrahlung[5]) and the B-Field (for Synchrotron emission) at the time of emission have to be set. The radiation fields required for IC radiation have already been defined for the computation of the particle spectra. It is straight-forward to calculate a Synchrotron-Self-Compton (SSC) component[6], which will add the Synchrotron spectrum to the set of photon fields for which the IC emission is calculated. Finally, the distance to the source has to be provided.

```
fRadiation->SetAmbientDensity(fParticles->GetAmbientDensity());
fRadiation->SetBField(fParticles->GetBField());
fRadiation->SetElectrons(fParticles->GetParticleSpectrum());
fRadiation->SetSSCTargetPhotons(fParticles->GetRadius());
fRadiation->SetDistance(d);
fRadiation->CalculateDifferentialPhotonSpectrum(ebins,emin,emax);
```

The top right panel of Fig. 3 displays the radiation spectra that are emitted by the electron distributions in the top left. In the bottom left panel the individual radiation components for the model PWN at an age of 1 kyr are shown.

Finally, it is possible to use the InstrumentResponse class to generate synthetic data points. In the following example, `alpha` is the ratio between the exposure in the spectral analysis region to

[5]and in principle also for π^0 decay, which is not relevant in this leptonic model, however.

[6]It should be noted that this requires a source extension.

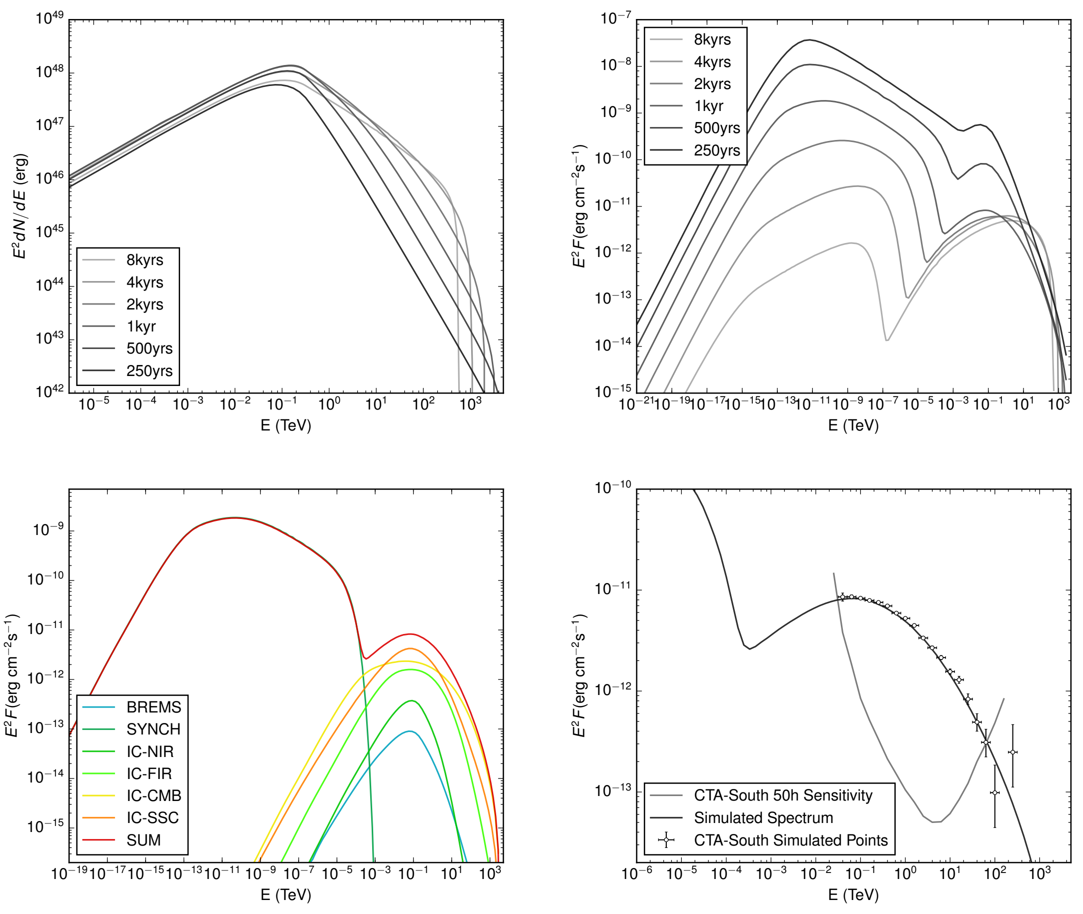

Figure 3: Electron and radiation spectra from the `GAMERA` implementation of the PWN model put forward by [8]. Top left: Electron spectrum evolution. Top right: Radiation spectrum evolution. Bottom left: Individual radiation components at an age of 1 kyr. Bottom right: Simulated data points using the prod-2 CTA south IRFs.

that in the background extraction region. In this example, the prod-2 CTA south IRFs[7] have been used, assuming an observation time of 50h.

```
InstrumentResponse *fIRF = new InstrumentResponse();
fIRF->SetupCTASouth();
fIRF->SetAlpha(alpha);
fIRF->SetObservationTime(observationTime);
fIRF->SimulateSpectrum(fRadiation->GetSpectrum());
```

The resulting synthetic data points are shown in the bottom right panel of Fig. 3. It should be noted that most of the example code is independent of the actual physical model, which is condensed in the `FillParameterVector` function. Thus, once a modeling framework exists, it can be easily applied to similar scenarios.

[7]available at: `portal.cta-observatory.org/Pages/CTA-Performance.aspx`

4. Example II: First Steps In A Population Synthesis

In this example, a random set of pulsar positions will be generated and the ambient B-Field strengths as well as the combined HI and H_2 number densities at these positions in the Galaxy will be calculated. Additionally, the combined HI and H_2 column densities towards the source positions, as seen from Earth's perspective, will be determined. Such calculations may correspond to the first steps in a source population synthesis study.

```
GalacticStructures *fUtils = new GalacticStructures();
/* set up Galactic model structure */
fUtils->SetSpiralArmModel("Vallee2005");
fUtils->SetCentralStructureModel("Bar");
fUtils->SetGasModel("Ferriere2001_a");
fUtils->SetBFieldModel("Jaffe2010");
fUtils->SetArmWidth(0.54);
fUtils->SetSurfaceDensityModel("IusifovKucuk");
fUtils->SetScaleHeight(0.05);
double xyzSol[] = {0.,8.5,0.};
double HI,H2,density,BField,l,b;
for(int i = 0; i < nSources; i++) {
  /* dice PSR coordinates in Cartesian space and calc. B&dens */
  fUtils->DiceGalacticPosition(x,y,z);
  HI = fUtils->HIDensity(x,y,z);
  H2 = fUtils->H2Density(x,y,z);
  density = fUtils->ModulateGasDensityWithSpirals(HI+H2,x,y,z);
  BField = fUtils->CalculateBField(x,y);
  /* go to Galactic Coordinates and calculate column densities */
  fUtils->GetGalactic(x,y,z,xyzSol,l,b);
  fUtils->CalculateGasColumnDensity(xyzSol,l,b,"HI");
  fUtils->CalculateGasColumnDensity(xyzSol,l,b,"H2");
}
```

The following model components were used in this example: Spiral arms by Vallée [9], a large-scale galactic gas model based on Ferrière [10], the pulsar surface density model by Iusifov&Küçük [11] and the magnetic field model by Jaffe [12]. The top panels of Fig. 4 provide a top-down view on the Galactic Disk showing B-Field and density values that have been averaged along the z-axis. Over-plotted is the aforementioned simulated sample of pulsars and the structure of the spiral arms. The lower panel shows the corresponding combined H_2+HI gas column density map as it would appear from the Earth's position in the Galaxy, over-plotted with the simulated pulsar positions in Galactic coordinates.

A further step in a population synthesis model would include the classes listed in Section 2.2 to calculate the radiation spectra at each source position. For instance, the model in the previous section could be inserted here for a population synthesis of young Galactic PWNe.

5. Summary

In this work, the GAMERA library has been presented. It is designed to provide an intuitive, user-friendly and flexible open-source tool for spectral modelling in gamma-ray astronomy. GAMERA allows for time-dependent or static spectral modeling using one or multiple emission zones. It

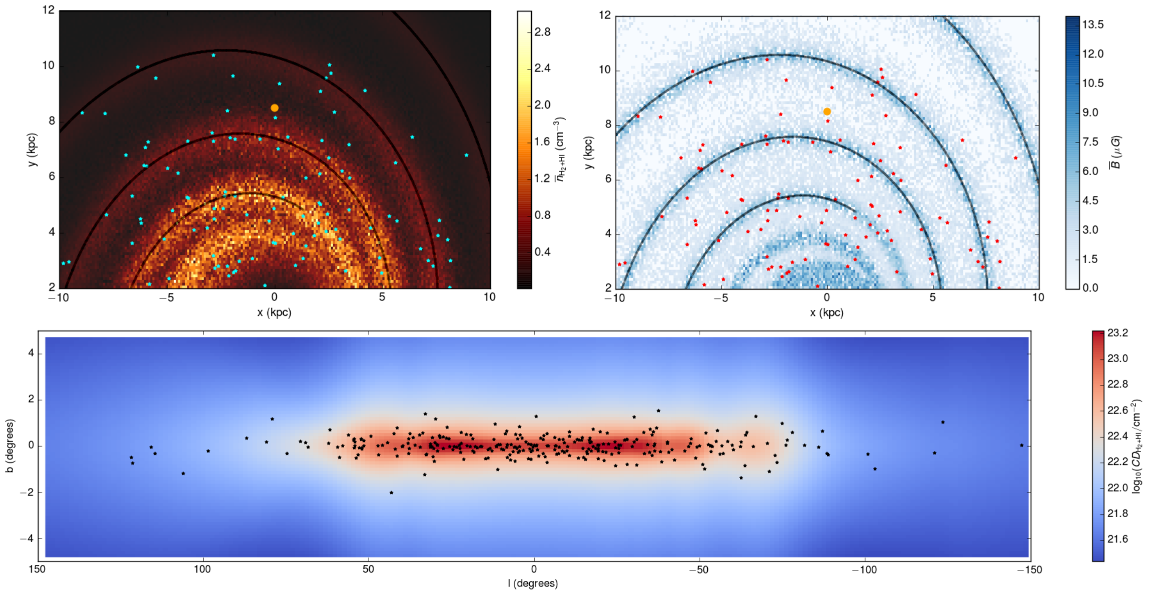

Figure 4: Galactic structures simulated with GAMERA. Top left: H_2+HI density in the x-y plane, averaged along the z-direction. Top right: Average B-Field strength. Bottom: H_2+HI column density from the Earth's perspective. In all plots, the stars correspond to a simulated pulsar population. The yellow dot indicates our position in the galaxy.

comes with the leptonic and hadronic radiation processes most relevant for VHE astronomy. Additionally, tools for population studies as well as the generation of synthetic data points are available. These features have been demonstrated on a time-dependent PWN model and with an example showing the first necessary steps for a population synthesis study.

References

[1] Ginzburg, V. L. and Syrovatskii, S. I. 1964, The Origin of Cosmic Rays, New York: Macmillan.

[2] Blumenthal, G. R. & Gould, R. J. 1970, Rev. Mod. Phys. 42, 237-271.

[3] Ghisellini et al. 1988, ApJL 334, 5-8.

[4] Baring, M. G. et al. 1999, ApJ 513, 311-338.

[5] Kafexhiu, E. et al. 2014, PhysRevD. 90, 12.

[6] Truelove, J. K. and McKee, C. F. 1999, ApJS 120, 299-326.

[7] Ptuskin, V. S. and Zirakashvili, V. N. 2005, A&A 429, 755-765.

[8] Torres, D. F. et al. 2014, JHEAp 1, 31-62.

[9] Vallée, J. P. 2005, AJ 130, 569-575

[10] Ferrière, K. M. 2001, RvMP 73, 1031-1066.

[11] Yusifov, I. & Küçük, I. 2004, A&A 422, 545-553.

[12] Jaffe, T. R. et al. 2010, MNRAS 401, 1013-1028.