`naima`: A package for radiative modelling and parent particle population inference

Víctor Zabalza

University of Leicester, UK

PyGamma15 workshop

November 16, 2015

The need for radiative modelling

- In non-thermal astrophysics, the goal is to understand the origin and evolution of relativistic particle populations.
- Radiative modelling is needed to infer the properties of these populations from observed photon spectra. The first step is deriving the present-age particle distribution.

The need for radiative modelling

- In non-thermal astrophysics, the goal is to understand the origin and evolution of relativistic particle populations.
- Radiative modelling is needed to infer the properties of these populations from observed photon spectra. The first step is deriving the present-age particle distribution.
- From this modelling, we can discuss:
  - Acceleration properties
  - Source energetics
  - Medium properties (B, gas density, etc…)

The need for radiative modelling

- In non-thermal astrophysics, the goal is to understand the origin and evolution of relativistic particle populations.
- Radiative modelling is needed to infer the properties of these populations from observed photon spectra. The first step is deriving the present-age particle distribution.
- From this modelling, we can discuss:
  - Acceleration properties
  - Source energetics
  - Medium properties (*B*, gas density, etc…)
- Simple modelling might lead to false conclusions: know your source!

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science^{TM}:

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science$^{\text{TM}}$:
  - Re-writing existing code!

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science^{TM}:
    - Re-writing existing code!
    - Hidden bugs

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science^{TM}:
  - Re-writing existing code!
  - Hidden bugs
  - Most implementation details unknown to paper readers.

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science^{TM}:
  - Re-writing existing code!
  - Hidden bugs
  - Most implementation details unknown to paper readers.
  - The results might not be easily reproducible by other researchers.

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science^{TM}:
  - Re-writing existing code!
  - Hidden bugs
  - Most implementation details unknown to paper readers.
  - The results might not be easily reproducible by other researchers.
- How to avoid this: use and contribute to Open Source radiative models

Situation until now

- Practically all radiative modelling is done with private code, even simple cases.
- This is Bad For Science^{TM}:
  - Re-writing existing code!
  - Hidden bugs
  - Most implementation details unknown to paper readers.
  - The results might not be easily reproducible by other researchers.
- How to avoid this: use and contribute to Open Source radiative models

`naima`

naima is a Python package that includes proven implementations of radiative models.

- Radiative models are implemented as modular classes.

`naima` is a Python package that includes proven implementations of radiative models.

- Radiative models are implemented as modular classes.
- Flexible: use provided particle distribution functions or roll your own.

naima is a Python package that includes proven implementations of radiative models.

- Radiative models are implemented as modular classes.
- Flexible: use provided particle distribution functions or roll your own.
- Uses astropy units on all inputs and outputs.

naima is a Python package that includes proven implementations of radiative models.

- Radiative models are implemented as modular classes.
- Flexible: use provided particle distribution functions or roll your own.
- Uses astropy units on all inputs and outputs.
- Reasonably fast implementation, and memoization of expensive functions means they are not recomputed if called with same arguments.

naima is a Python package that includes proven implementations of radiative models.

- Radiative models are implemented as modular classes.
- Flexible: use provided particle distribution functions or roll your own.
- Uses astropy units on all inputs and outputs.
- Reasonably fast implementation, and memoization of expensive functions means they are not recomputed if called with same arguments.

All these models are *one-zone models*, but can be used as basis for more complex models by linking several of them.

Available models

- Synchrotron for random B-field (Aharonian et al. 2010)

- Synchrotron for random *B*-field (Aharonian et al. 2010)
- Inverse Compton for graybody seed photon fields (Khangulyan et al. 2014)
  - Can use dominant ISRFs with values at solar system: `CMB`, `FIR`, `NIR`, and/or
  - you can provide any number of graybodies giving name, temperature and energy density.
  - Anisotropic IC used if an interaction angle is provided in the graybody definition.

- Synchrotron for random *B*-field (Aharonian et al. 2010)
- Inverse Compton for graybody seed photon fields (Khangulyan et al. 2014)
  - Can use dominant ISRFs with values at solar system: `CMB`, `FIR`, `NIR`, and/or
  - you can provide any number of graybodies giving name, temperature and energy density.
  - Anisotropic IC used if an interaction angle is provided in the graybody definition.
- Nonthermal Bremsstrahlung (Baring et al. 1999)

Available models

- Synchrotron for random B-field (Aharonian et al. 2010)
- Inverse Compton for graybody seed photon fields (Khangulyan et al. 2014)
  - Can use dominant ISRFs with values at solar system: `CMB`, `FIR`, `NIR`, and/or
  - you can provide any number of graybodies giving name, temperature and energy density.
  - Anisotropic IC used if an interaction angle is provided in the graybody definition.
- Nonthermal Bremsstrahlung (Baring et al. 1999)
- Pion Decay (Kafexhiu et al. 2014)
  - Good approximation for all the relevant proton energy range.

Available models

- Synchrotron for random *B*-field (Aharonian et al. 2010)
- Inverse Compton for graybody seed photon fields (Khangulyan et al. 2014)
  - Can use dominant ISRFs with values at solar system: `CMB`, `FIR`, `NIR`, and/or
  - you can provide any number of graybodies giving name, temperature and energy density.
  - Anisotropic IC used if an interaction angle is provided in the graybody definition.
- Nonthermal Bremsstrahlung (Baring et al. 1999)
- Pion Decay (Kafexhiu et al. 2014)
  - Good approximation for all the relevant proton energy range.

All model classes have these attributes:

- `.flux(energy, [distance])`: Compute differential flux
- `.sed(energy, [distance])`: Compute SED
- `.compute_W[p,e](Emin, Emax)`: Compute total energy in electrons or protons between energies `Emin` and `Emax`.
- `.particle_distribution`: Access to the instance for the particle distribution.

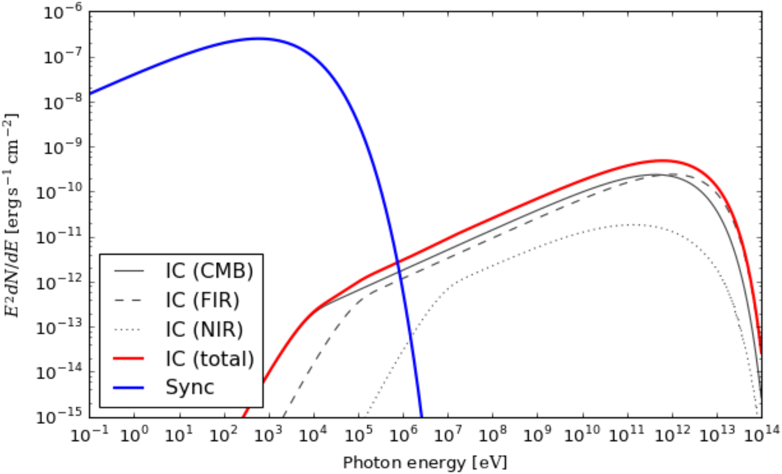
Radiative models in naima (short demo)

```python
from naima.models import (ExponentialCutoffPowerLaw,
            InverseCompton, Synchrotron)

# define particle distribution
ECPL = ExponentialCutoffPowerLaw(
                amplitude=1e36*u.Unit('1/eV'),
                e_0=1*u.TeV, alpha=2.1,
                e_cutoff=13*u.TeV, beta=2)

# define radiative models
IC = InverseCompton(ECPL,
                seed_photon_fields=['CMB', 'FIR', 'NIR'])
SYN = Synchrotron(ECPL, B=100*u.uG)

# compute models at given photon energies
spectrum_energy = np.logspace(-1,14,1000)*u.eV
sed_IC = IC.sed(spectrum_energy, distance=1.5*u.kpc)
sed_SYN = SYN.sed(spectrum_energy, distance=1.5*u.kpc)
```


10^{-6}
10^{-7}
10^{-8}
10^{-9}
10^{-10}
10^{-11}
10^{-12}
10^{-13}
10^{-14}
10^{-15}
$E^2 dN/dE$ [erg s^{-1} cm^{-2}]
IC (CMB)
IC (FIR)
IC (NIR)
IC (total)
Sync
10^{-1} 10^{0} 10^{1} 10^{2} 10^{3} 10^{4} 10^{5} 10^{6} 10^{7} 10^{8} 10^{9} 10^{10} 10^{11} 10^{12} 10^{13} 10^{14}
Photon energy [eV]

Using `naima` for particle distribution inference

How can radiative models be used in the discussion of an observational result?

Using `naima` for particle distribution inference

How can radiative models be used in the discussion of an observational result?

1. Find the particle distribution function and radiative channels that can describe the observed spectrum.
2. Infer the best-fit parameters and uncertainties.
3. Discuss the physical implications.

Using `naima` for particle distribution inference

`naima` provides tools to fit these radiative models to observed nonthermal SEDs:

Using naima for particle distribution inference

naima provides tools to fit these radiative models to observed nonthermal SEDs:

- Using MCMC sampling (through emcee) of a χ^2-like likelihood function, obtaining in a single run:
  - The Maximum Likelihood (ML) parameters.

Using naima for particle distribution inference

naima provides tools to fit these radiative models to observed nonthermal SEDs:

- Using MCMC sampling (through emcee) of a χ^2-like likelihood function, obtaining in a single run:
  - The Maximum Likelihood (ML) parameters.
  - The posterior distribution of the free parameters, from which marginalised median values and uncertainties can be derived.

Using `naima` for particle distribution inference

`naima` provides tools to fit these radiative models to observed nonthermal SEDs:

- Using MCMC sampling (through `emcee`) of a χ^2-like likelihood function, obtaining in a single run:
  - The Maximum Likelihood (ML) parameters.
  - The posterior distribution of the free parameters, from which marginalised median values and uncertainties can be derived.
  - The posterior distribution of secondary values.

Using naima for particle distribution inference

naima provides tools to fit these radiative models to observed nonthermal SEDs:

- Using MCMC sampling (through emcee) of a χ^2-like likelihood function, obtaining in a single run:
  - The Maximum Likelihood (ML) parameters.
  - The posterior distribution of the free parameters, from which marginalised median values and uncertainties can be derived.
  - The posterior distribution of secondary values.
- Tools to save, analyse, and plot the results of the MCMC run.

Using naima for particle distribution inference

naima provides tools to fit these radiative models to observed nonthermal SEDs:

- Using MCMC sampling (through emcee) of a χ^2-like likelihood function, obtaining in a single run:
  - The Maximum Likelihood (ML) parameters.
  - The posterior distribution of the free parameters, from which marginalised median values and uncertainties can be derived.
  - The posterior distribution of secondary values.
- Tools to save, analyse, and plot the results of the MCMC run.
- Wrappers for the radiative models to be used in the sherpa spectral analysis package.

Using `naima` for particle distribution inference

`naima` provides tools to fit these radiative models to observed nonthermal SEDs:

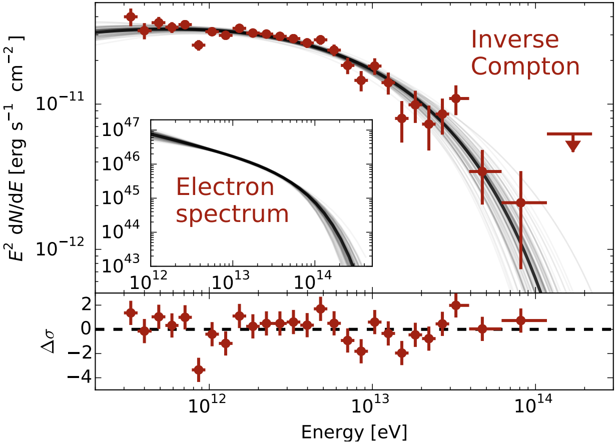
- Using MCMC sampling (through `emcee`) of a χ^2-like likelihood function, obtaining in a single run:
  - The Maximum Likelihood (ML) parameters.
  - The posterior distribution of the free parameters, from which marginalised median values and uncertainties can be derived.
  - The posterior distribution of secondary values.
- Tools to save, analyse, and plot the results of the MCMC run.
- Wrappers for the radiative models to be used in the `sherpa` spectral analysis package.

Caveat

The χ^2-like likelihood function assumes Gaussian, independent uncertainties on the flux measurements. Use with care.

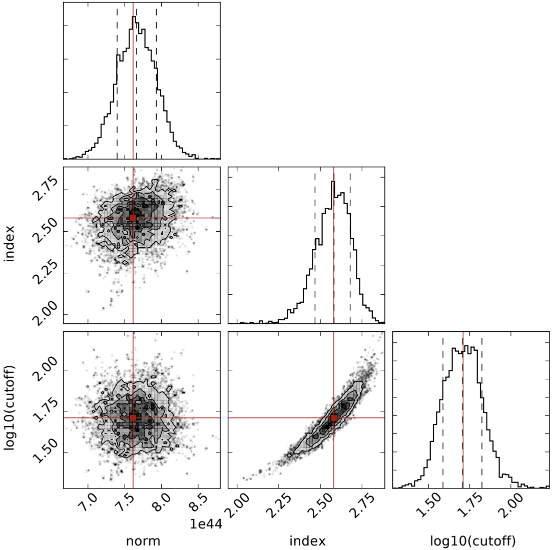
Model parameters:

- Exponential Cutoff Power Law electron energy distribution
- Seed photon fields: CMB and galactic far infrared (from GALPROP: $T = 27\,\mathrm{K}$, $u = 0.4\,\mathrm{eV\,cm^{-3}}$).



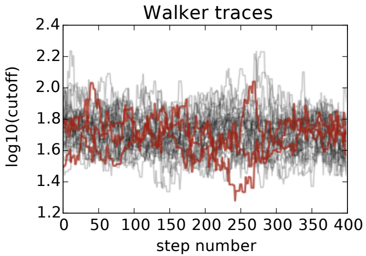
Inferred particle distribution parameters (median and 1σ uncertainty):

- $\Gamma_e = 2.6 \pm 0.1$
- $E_{\rm e,cutoff} = 51^{+15}_{-12}$ TeV
- $W_{\rm e}(E_{\rm e} > 1\,{\rm TeV}) = (1.07 \pm 0.08) \times 10^{47}$ erg



HESS spectrum of RX J1713–3946— leptonic model

Information for each of the parameters included in the MCMC run:

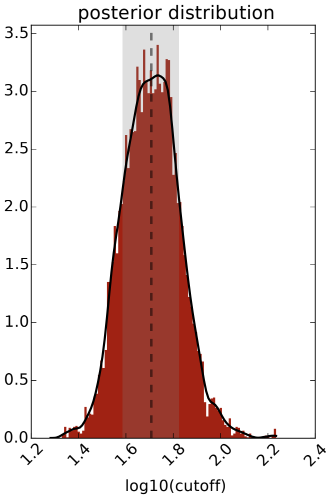


Walkers: 32
Steps in chain: 400
Autocorrelation time: 35.8
Mean acceptance fraction: 0.624
Distribution properties for the whole chain:

− median: 1.71
− std: 0.12
− median with uncertainties based on the 16th and 84th percentiles ($\sim 1\sigma$):

$$\log10(\text{cutoff}) = 1.71^{+0.117}_{-0.12}$$

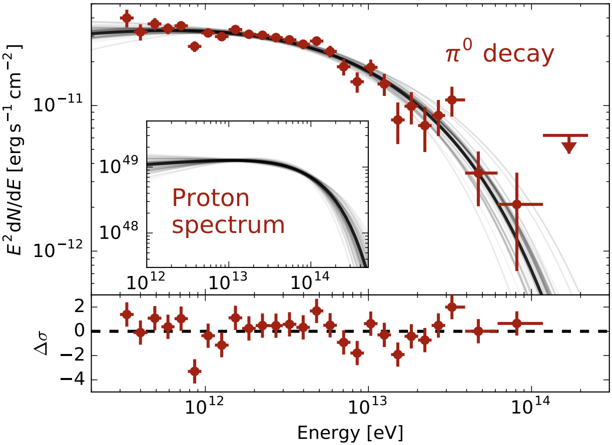
$$\text{cutoff} = 51^{+15.7}_{-12.3}$$



HESS spectrum of RX J1713–3946— hadronic model

Model parameters:

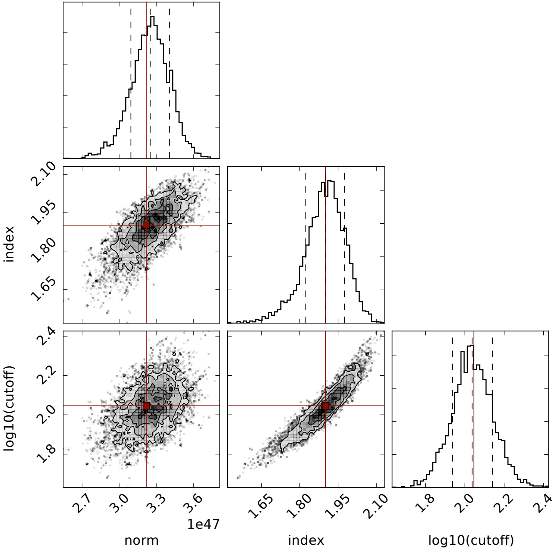
- Exponential Cutoff Power Law proton energy distribution
- $n_{\mathrm{H}} = 1\,\mathrm{cm}^{-3}$ assumed.



HESS spectrum of RX J1713–3946— hadronic model

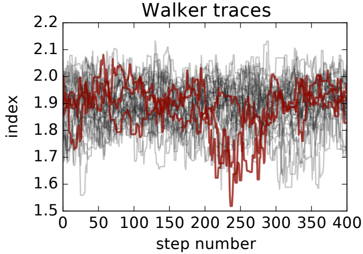
Inferred particle distribution parameters (median and 1σ uncertainty):

- $\Gamma_\mathrm{p} = 1.90 \pm 0.08$
- $E_\mathrm{p,cutoff} = 109^{+30}_{-20}$ TeV
- $W_\mathrm{p}(E_\mathrm{p} > 1\,\mathrm{TeV}) = (6.1 \pm 0.2) \times 10^{49}$ erg



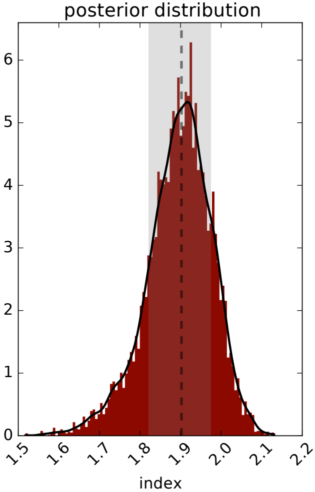
HESS spectrum of RX J1713–3946— hadronic model

Information for each of the parameters included in the MCMC run:



Walkers: 32
Steps in chain: 400
Autocorrelation time: 16.9
Mean acceptance fraction: 0.635
Distribution properties for the whole chain:

- median: 1.9, std: 0.0812
- median with uncertainties based on the 16th and 84th percentiles ($\sim 1\sigma$):
  index = 1.90 ± 0.08



- `naima` provides stable, consistent implementations of proven radiative models.

Conclusions

- `naima` provides stable, consistent implementations of proven radiative models.
- These can be used on their own or used in other packages (e.g. `gammapy`).

Conclusions

- `naima` provides stable, consistent implementations of proven radiative models.
- These can be used on their own or used in other packages (e.g. `gammapy`).
- SED fitting can be done with the `emcee` wrapper in `naima`, and forward-folding with the model wrappers for `sherpa`.

Conclusions

- `naima` provides stable, consistent implementations of proven radiative models.
- These can be used on their own or used in other packages (e.g. `gammapy`).
- SED fitting can be done with the `emcee` wrapper in `naima`, and forward-folding with the model wrappers for `sherpa`.
- Tutorial on naima models and fitting on Thursday at 11 AM.