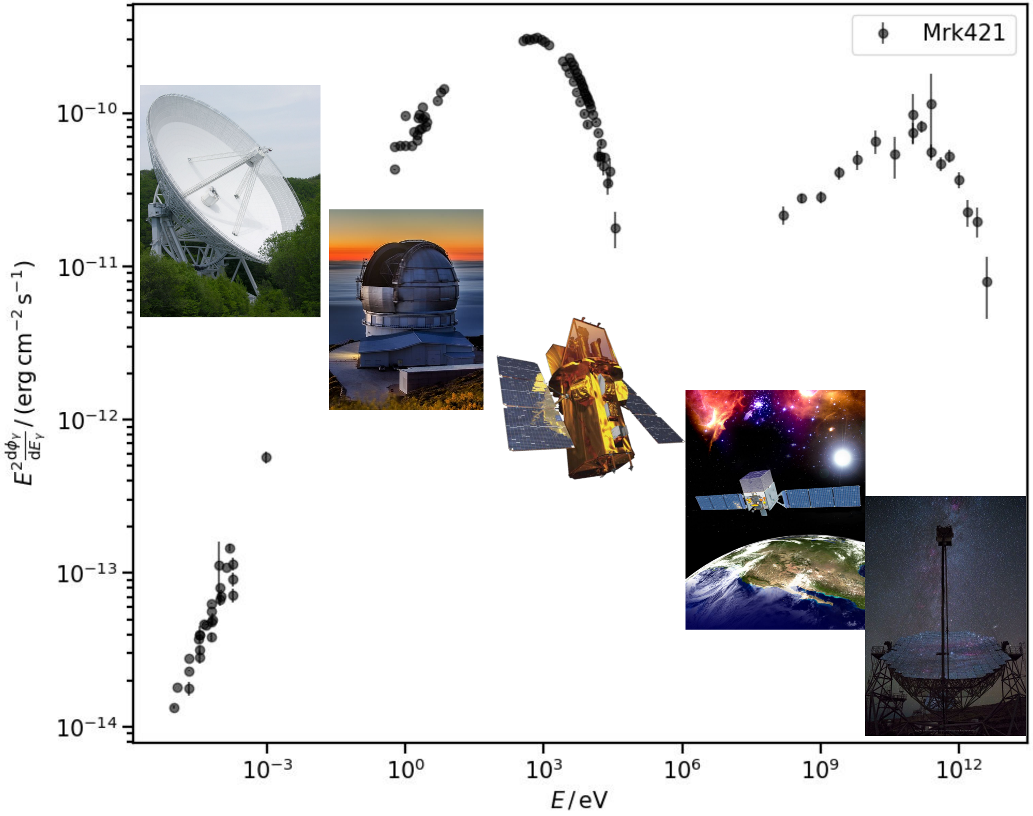
Institut de Física d'Altes Energies

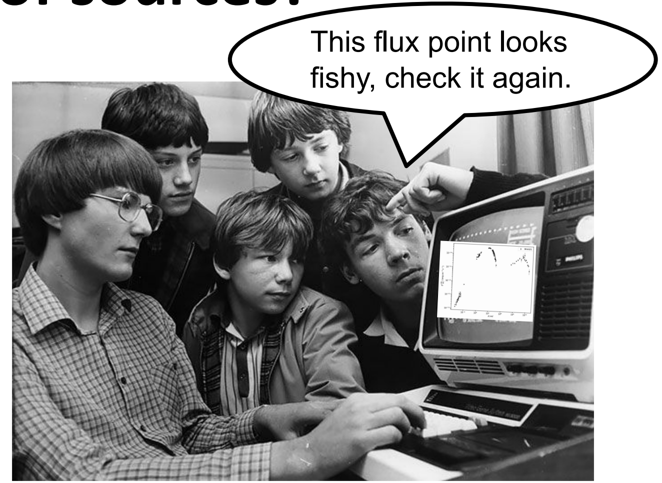
A new approach to AGN modelling with open-source software: the agnpy experience

Cosimo Nigro, IFAE

Astrophysics Workshop on Numerical Multimessenger Modelling
27 Feb - 1 Mar 2023, Ruhr-Universität Bochum

How do we measure the MWL emission of sources?



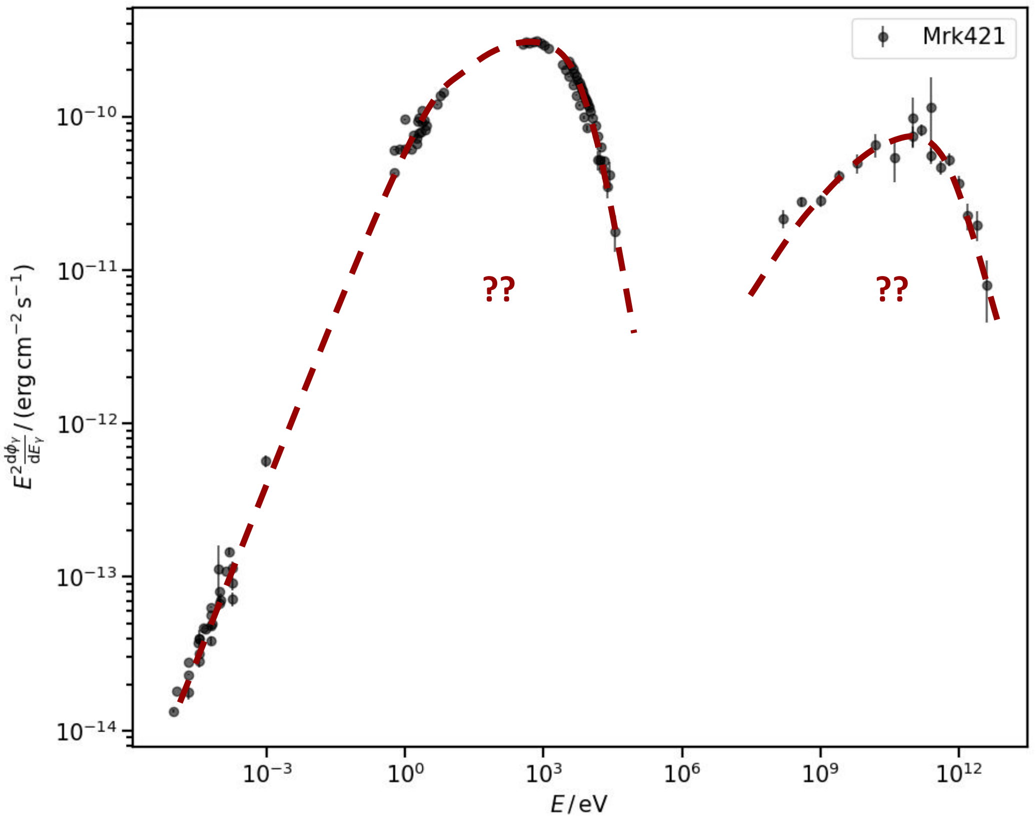


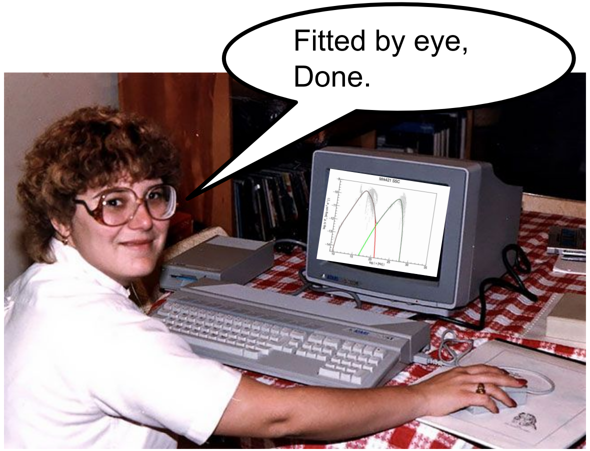
> Flux is measured by several instruments in different energy bands. Each measurement is a result of the **work of a small team often using the same software**;

> each **collaboration** implements some **review or cross-check system for their analyses**;

> there is, in general, a **free exchange of data and software**.

How do we interpret their emission? Who does it?



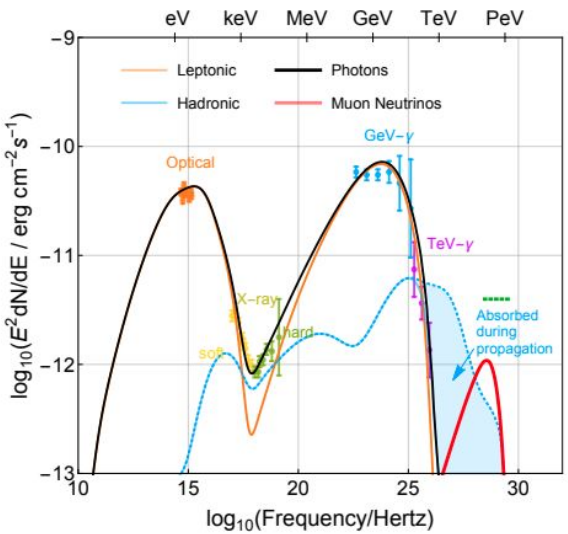


> Even in the collaborations reducing and analysing the data, **modelling** is performed by **few persons with closed-source software**;

> there was **no review or validation of results** until the recent hadronic code comparison;

> there is **no free exchange of software**, only results are exchanged.

Why do we perceive data analysis and modelling differently?



One of the authors of this model and plot (Walter) complained in a seminar that experimentalists should release the "raw data" because he did not trust "what they were doing".

It is legitimate to ask: why are we instead supposed to trust a software that he and only few of his collaborators can access?

> No doubt **these closed-source software have shaped the understanding of the field**;

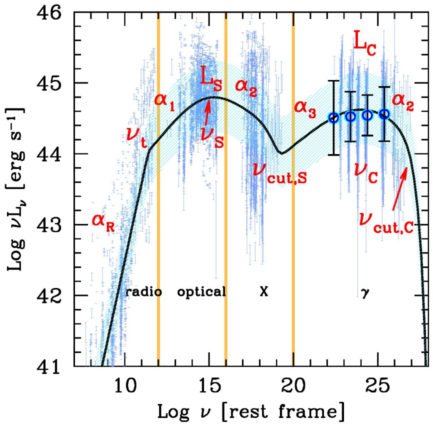
> but their **results are not reproducible** (I cannot re-perform the calculations in a paper in autonomy and verify its conclusions), moreover the interpretation is **accessible** only to a **restricted group of people**;

> the closed-source approach might be a choice, or a necessity, but: **once the knowledge you generate is consolidated, should you share the tools used to produce this knowledge with the community?**

Why should modelling tools be shared?

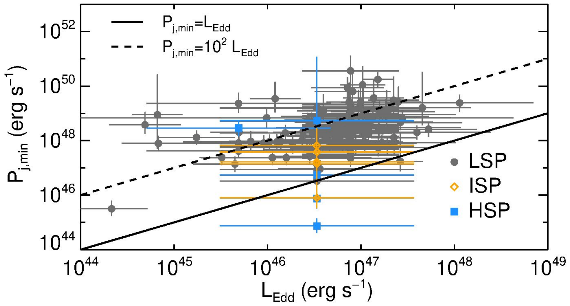
> **To achieve maximum scientific exploitation**: the wealth of MWL data accumulated in the last decades cannot be interpreted by small groups with closed-source software;

> I suspect there is a class of problems that the old generation of closed-source software cannot solve. **It is suited for individual source studies** but **not for large-scale systematic studies with a large sample of sources**.



> Ghisellini et al. (2017) fitted the spectrum of 747 blazars assuming a leptonic model. An analytical approximation was used to model the broad-band emission.

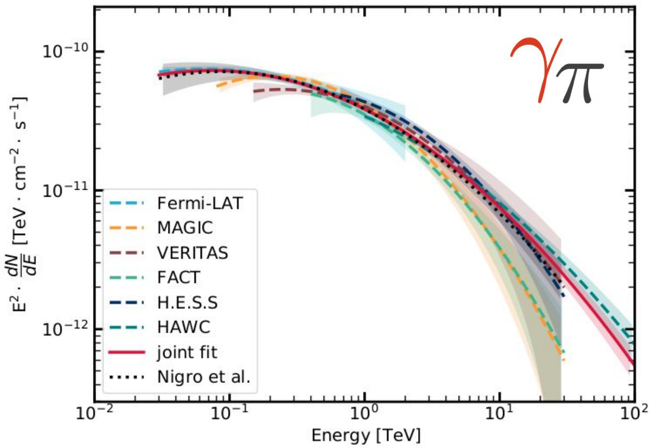
> Liodakis et al. (2020) fitted 145 sources high-energy emission assuming a proton synchrotron emission. Largest and only systematic statistical analysis of blazars with hadronic models, high-energy emission fitted with a polynomial.



The connection with data-analysis tools

> Why is it difficult to connect the modelling software with data-analysis tools?

- probably written in old programming language, not interfaceable with modern data-analysis tools;
- difficult to distribute the statistical analysis of hundreds of sources on a computer cluster without a modelling software provided via package managers and tested on different environments;
- maybe this old modelling software is lagging some years behind the capabilities of modern data-analysis tools.



Crab Nebula spectrum obtained from all operating gamma-ray instrument using standardised data and open-source software [Albert, A. et al. (2022)](#).

> What is the actual status of data-analysis tools in high-energy astrophysics?

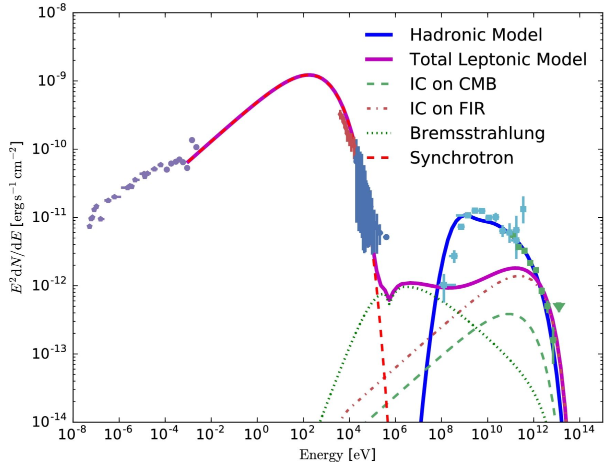
- we are steadily moving towards the adoption of standardised data and open-source analysis tools;
- can we do the same for modelling?

Open-source tools for modelling

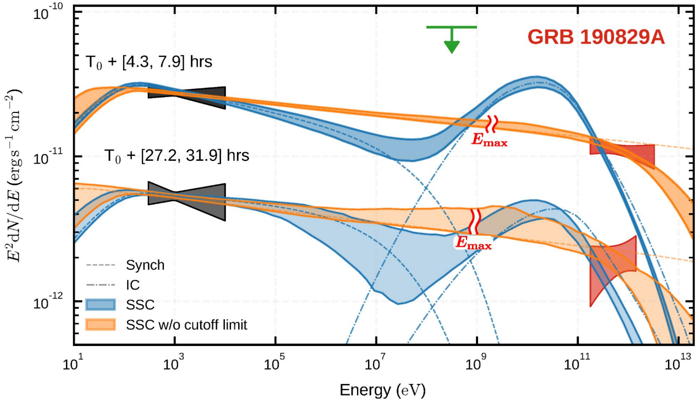
> Several **open-source software** have been developed in recent years to interpreting the non-thermal emission of astrophysical sources (naima, gamera, jetset, agnpy, BHJet, FLAREMODEL);

> designed for different sources (galactic or extragalactic) but easily expanded to science cases where same radiative processes occur;

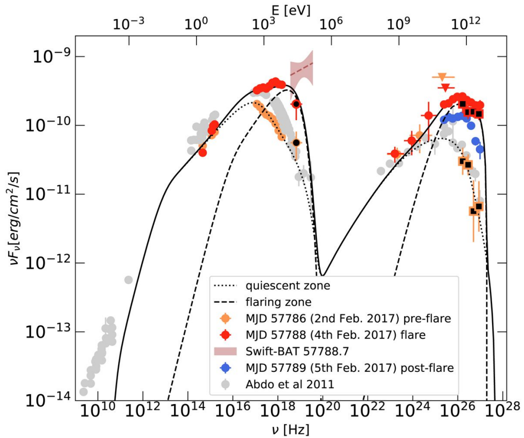
> **naima deserves a special mention** as it was the pioneer (released in 2015) and proved to be adaptable to several scientific cases.



Naima applied to model a SNR Ahnen, M. L. et al. (2017)



Naima applied to model a GRB H.E.S.S. Coll. (2021)



Naima applied to model a AGN Acciari, V. A. et al. (2021)

My experience: agnpy

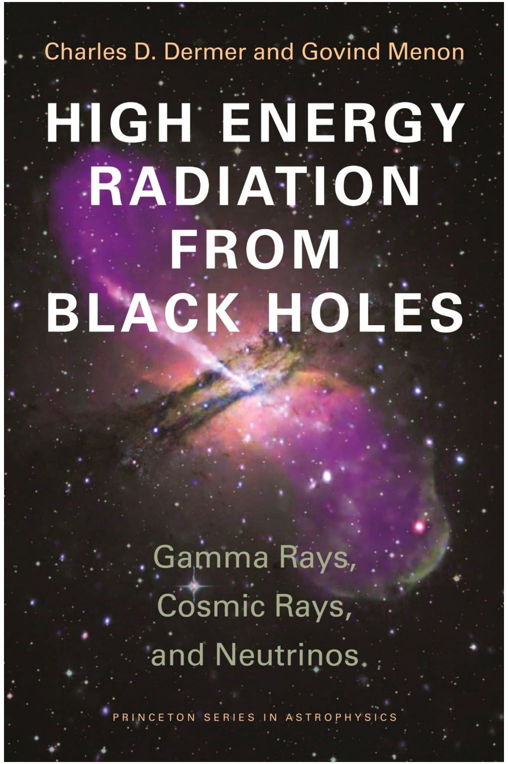
> I wanted to create the equivalent of naima for AGN: I was analysing the gamma-ray emission from FSRQs and could not use naima for external Compton;

> I had done some development for Gammapy, used this experience to code some classes using **numpy** and **astropy** to compute the basic leptonic radiative processes and develop a proper external Compton treatment;

> I **built the software on the work of Dermer and Finke**;

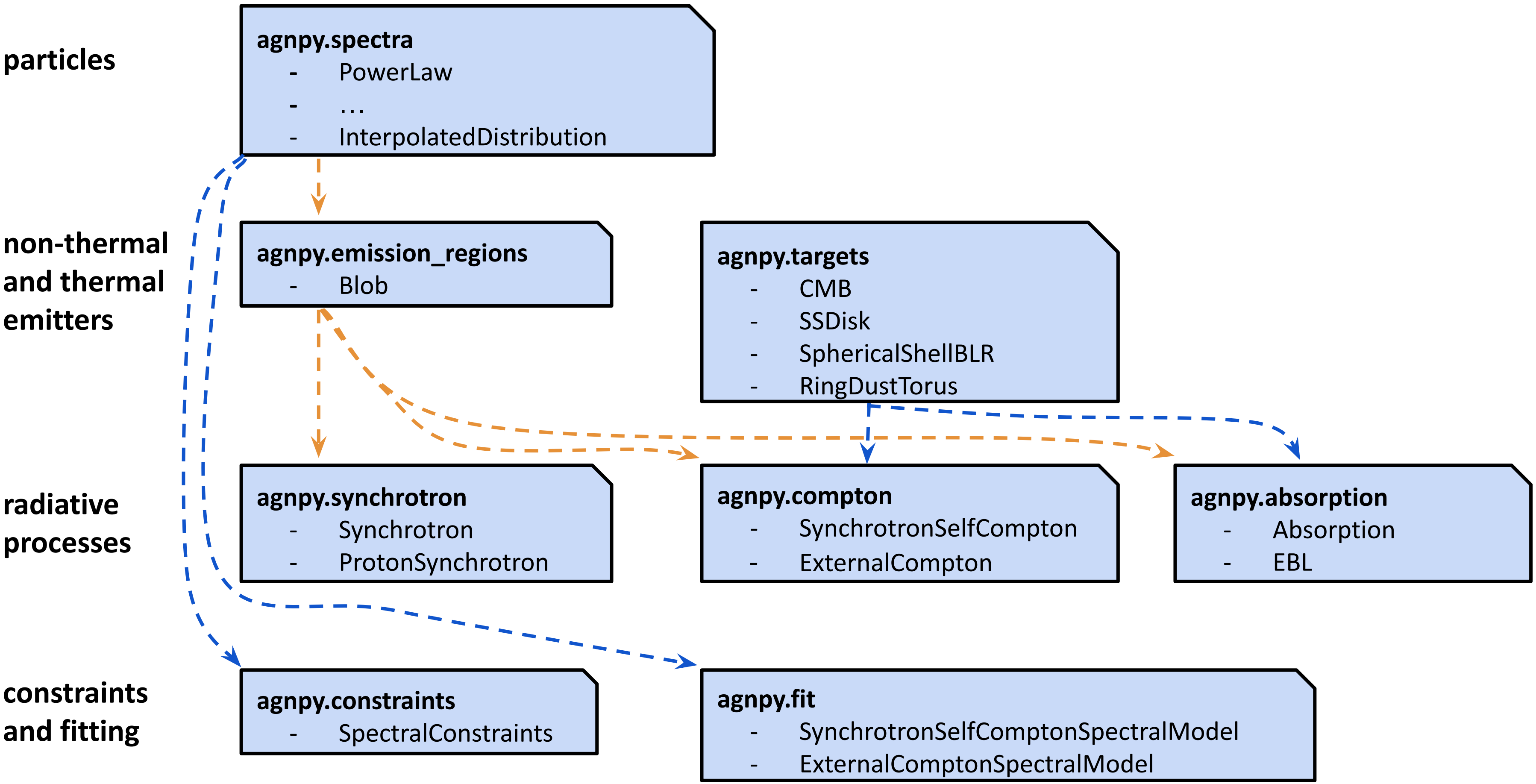
> it was natural for me to make it public:
- I had used open-source tools during my whole thesis (Astropy, Gammapy),
- thought that there were many people with a basic knowledge of radiative processes but without software tools to apply them.

> **Useful links: [docs, github, slack].**



astropy-powered
astropy.org

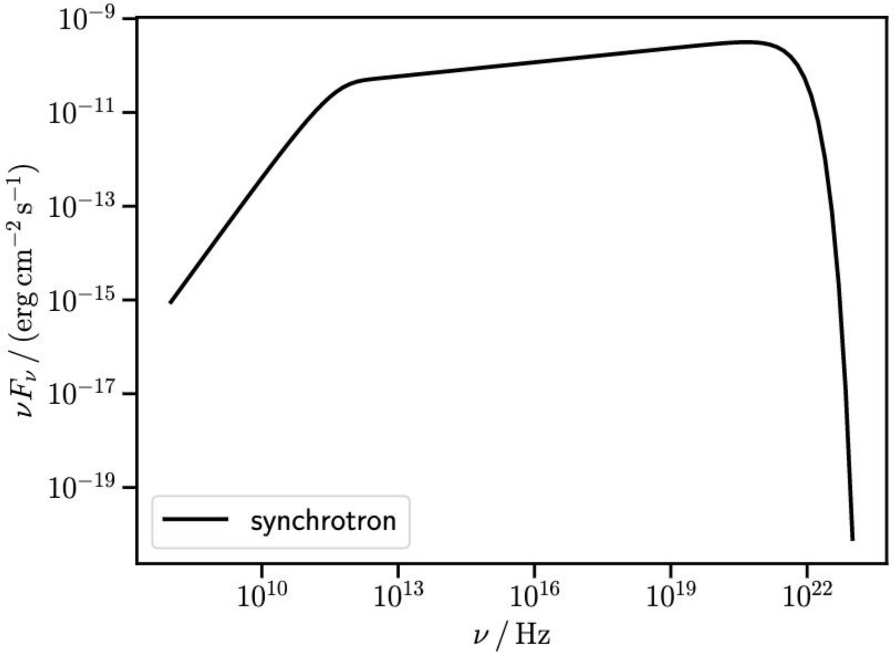
Software structure



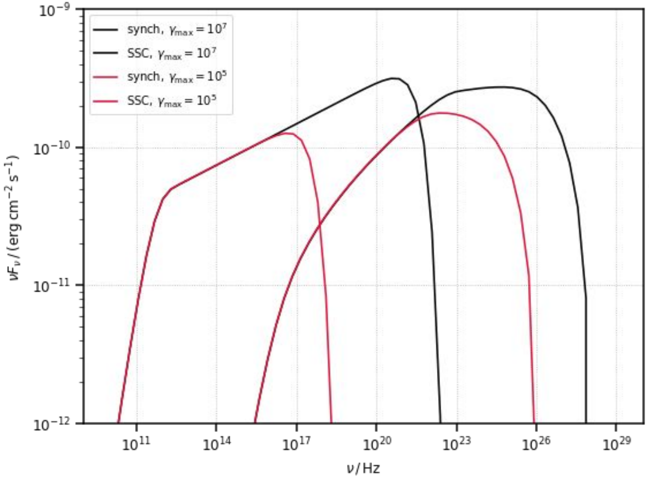
Applications: computing the SED of a simple radiative process

> I wanted to be able to compute the SED for a given radiative process with a few lines of python:

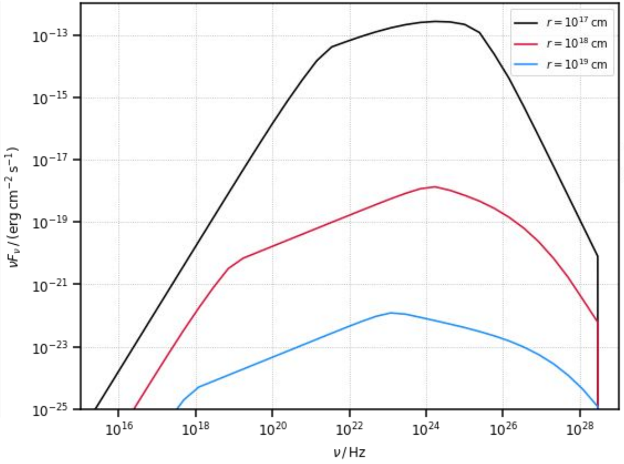
```
import numpy as np
import astropy.units as u
from agnpy.emission_regions import Blob
from agnpy.synchrotron import Synchrotron
from agnpy.utils.plot import plot_sed
import matplotlib.pyplot as plt

# define the emission region and the radiative process
blob = Blob()
synch = Synchrotron(blob)

# compute the SED over an array of frequencies
nu = np.logspace(8, 23) * u.Hz
sed = synch.sed_flux(nu)

# plot it
plot_sed(nu, sed, label="Synchrotron")
plt.show()
```



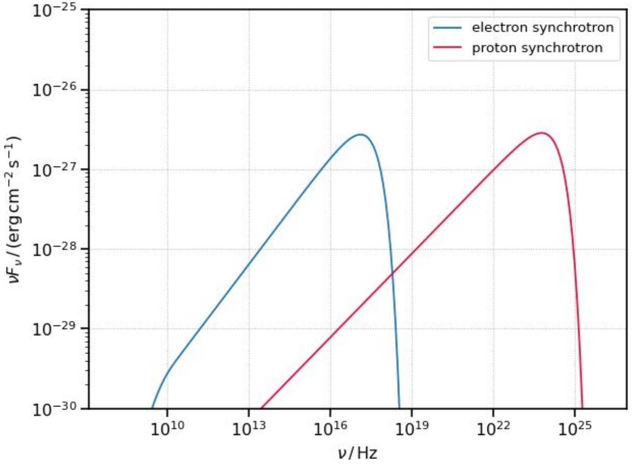
Applications: radiative processes implemented



Synchrotron + SSC



External Compton

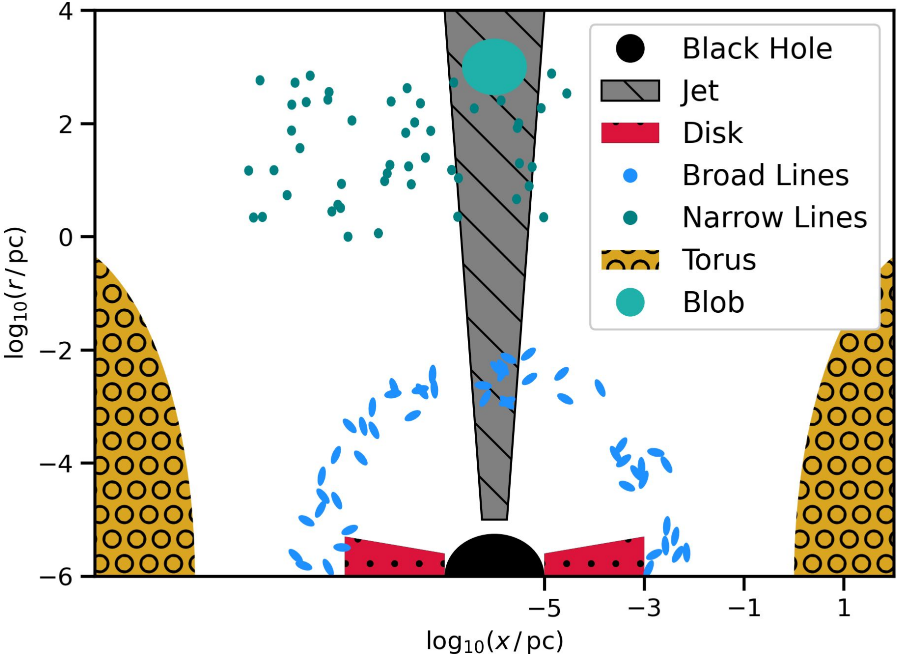


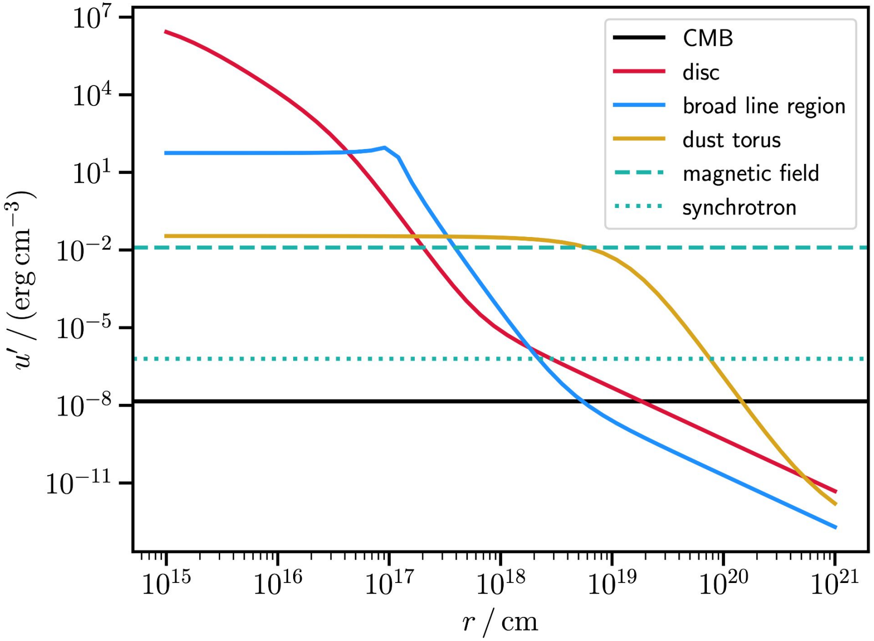
New: proton synchrotron

New: Started to implement p-gamma using Kelner and Aharonian, at the moment in agnpy, but <u>we would like to move it in an external package when it grows.</u>

Applications: energy density of the target photon fields

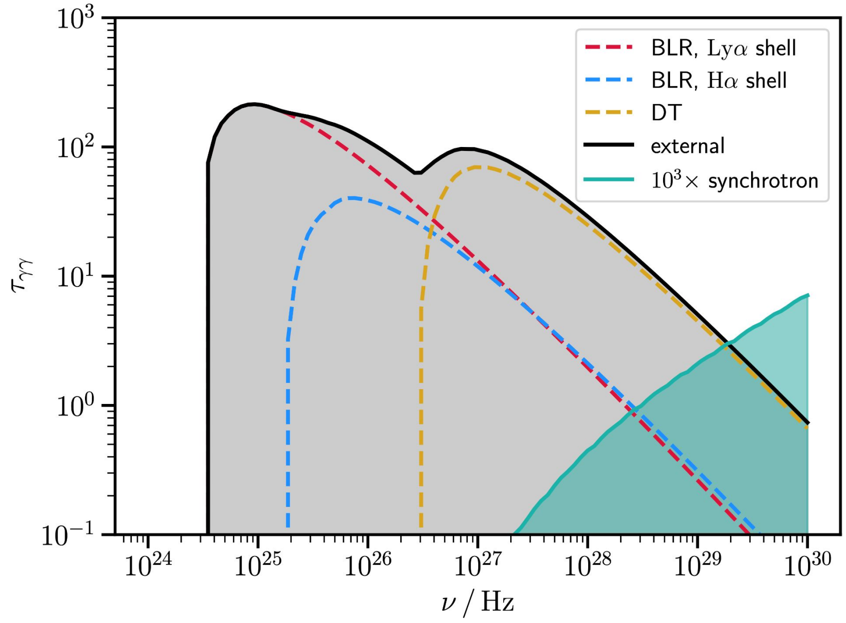
> In case of EC on different targets, we might want to determine their energy density, *u* [erg cm^{-3}];
> agnpy computes *u* as a function of the distance from the central BH, *r*;
> the black-body thermal emission of the disk and DT can also be computed.

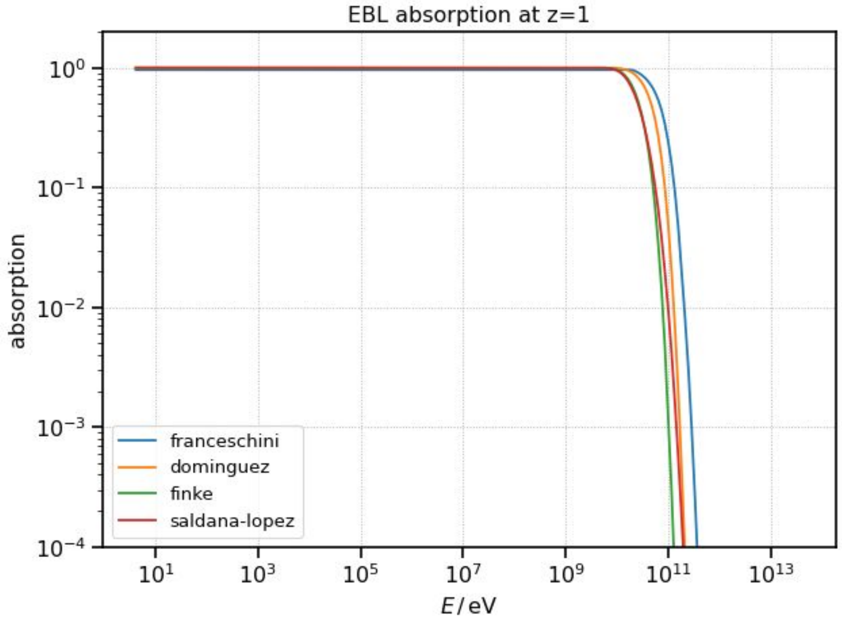




Applications: absorption on soft photon fields

- agnpy computes the γγ absorption produced on photon fields internal and external to the emission region:
  - internal: synchrotron radiation.
  - external: BLR, DT and a monochromatic point-source;
- EBL models by Franceschini, Dominguez, Finke, Saldana-Lopez.





Applications: time-dependent modelling?

> agnpy does not include any routine for the solution of the differential equation describing the particles time evolution;

- it contains a class allowing for self-consistent modelling;
- break and maximum Lorentz factor of the particle distributions, γ_b and γ_{max}, constrained accounting for the interplay between acceleration, cooling and escape processes (simple parametrisation).

> an InterpolatedDistribution is available to interpolate arbitrary values of density and lorentz factors representing a particle energy distribution (e.g. output of a cooling code).

property **gamma_max_ballistic**

Naive estimation of maximum Lorentz factor of electrons comparing acceleration time scale with ballistic time scale. For the latter we assume that the particles crosses the blob radius.

$$\begin{aligned}(\mathrm{d}E/\mathrm{d}t)_{\mathrm{acc}} &= \xi c E / R_L \\ T_{\mathrm{acc}} &= E \,/\, (\mathrm{d}E/\mathrm{d}t)_{\mathrm{acc}} = R_L/(\xi c) \\ T_{\mathrm{bal}} &= R_b/c \\ T_{\mathrm{acc}} &< T_{\mathrm{bal}} \Rightarrow \gamma_{\mathrm{max}} < \frac{\xi R_b e B}{m_e c^2}\end{aligned}$$

property **gamma_max_synch**

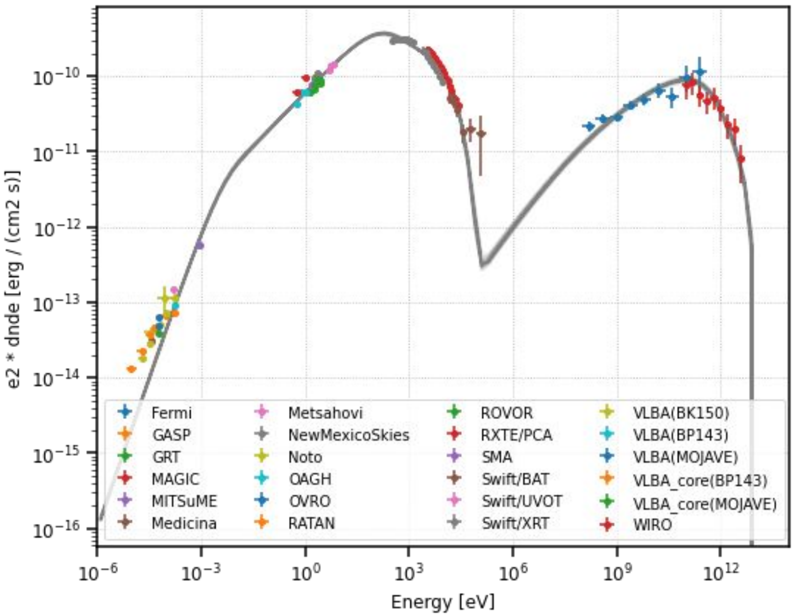
Simple estimation of maximum Lorentz factor of electrons comparing the acceleration time scale with the synchrotron energy loss

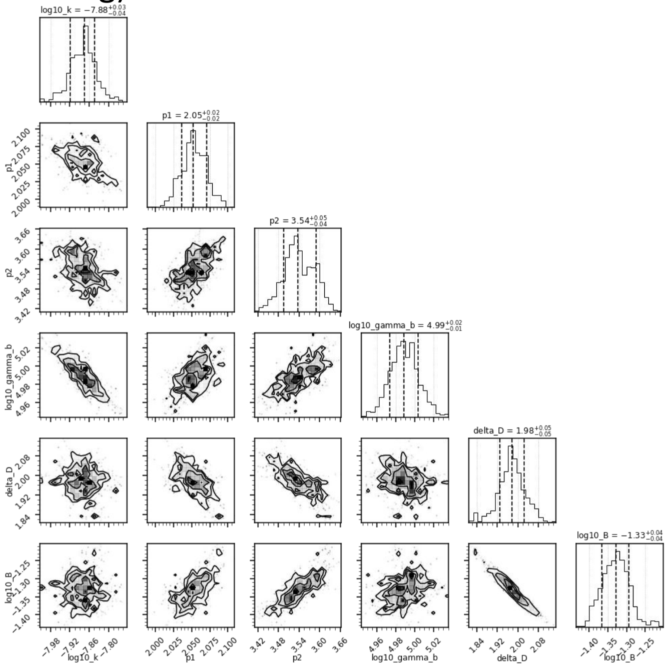
$$\begin{aligned}(\mathrm{d}E/\mathrm{d}t)_{\mathrm{acc}} &= \xi c E / R_L \\ (\mathrm{d}E/\mathrm{d}t)_{\mathrm{synch}} &= 4/3 \sigma_T c U_B \gamma^2 \\ (\mathrm{d}E/\mathrm{d}t)_{\mathrm{acc}} &= (\mathrm{d}E/\mathrm{d}t)_{\mathrm{synch}} \Rightarrow \gamma_{\mathrm{max}} < \sqrt{\frac{6\pi\xi e}{\sigma_T B}}\end{aligned}$$

Applications: fitting

> I wanted agnpy to remain a software for modelling (did not want to implement data-handling routines already available in other packages);

> **I created wrappers for sherpa and Gammapy.** Agnpy models can be directly imported in sherpa or Gammapy and fitted to data handled by these software (allows forward folding).



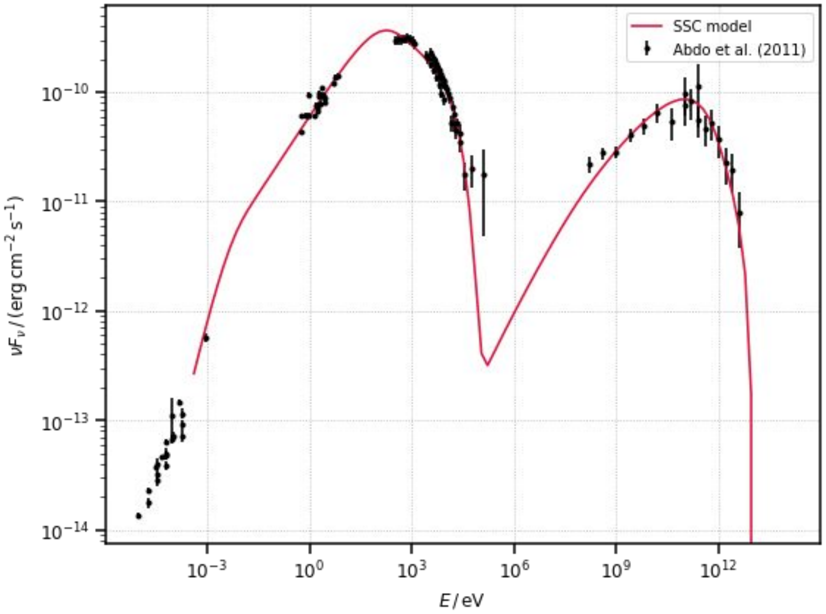


mcmc fit performed with Gammapy and agnpy models

Applications: fitting

> I wanted agnpy to remain a software for modelling (did not want to implement data-handling routines already available in other packages);

> **I created wrappers for sherpa and Gammapy.** Agnpy models can be directly imported in sherpa or Gammapy and fitted to data handled by these software (allows forward folding).



χ^2 fit performed with sherpa and agnpy models

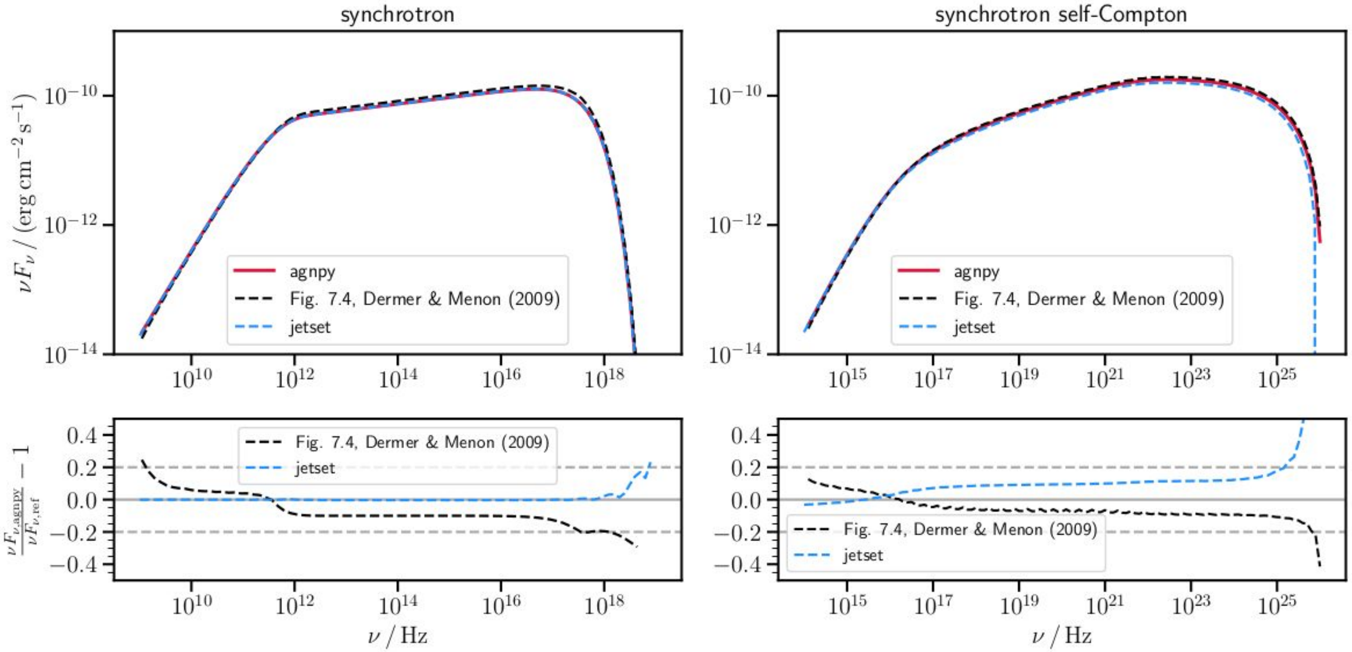
```
Fit succesful =  True
Method                = levmar
Statistic             = chi2
Initial fit statistic = 3533.13
Final fit statistic   = 270.786 at function evaluation 71
Data points           = 86
Degrees of freedom    = 80
Probability [Q-value] = 1.47258e-22
Reduced statistic     = 3.38482
Change in statistic   = 3262.35
   ssc.log10_k      -7.88461      +/- 0.0702959
   ssc.p1           2.05281       +/- 0.0231709
   ssc.p2           3.53711       +/- 0.0517896
   ssc.log10_gamma_b    4.99003       +/- 0.0228676
   ssc.delta_D      19.809        +/- 0.612714
   ssc.log10_B      -1.33284      +/- 0.0389128
CPU times: user 19 s, sys: 6.93 s, total: 25.9 s
Wall time: 26.1 s
```

Data from 24 instruments fitted in 25 s.

Validation

> We **validated the software by comparing its results against literature and against other open-source software** (when possible);

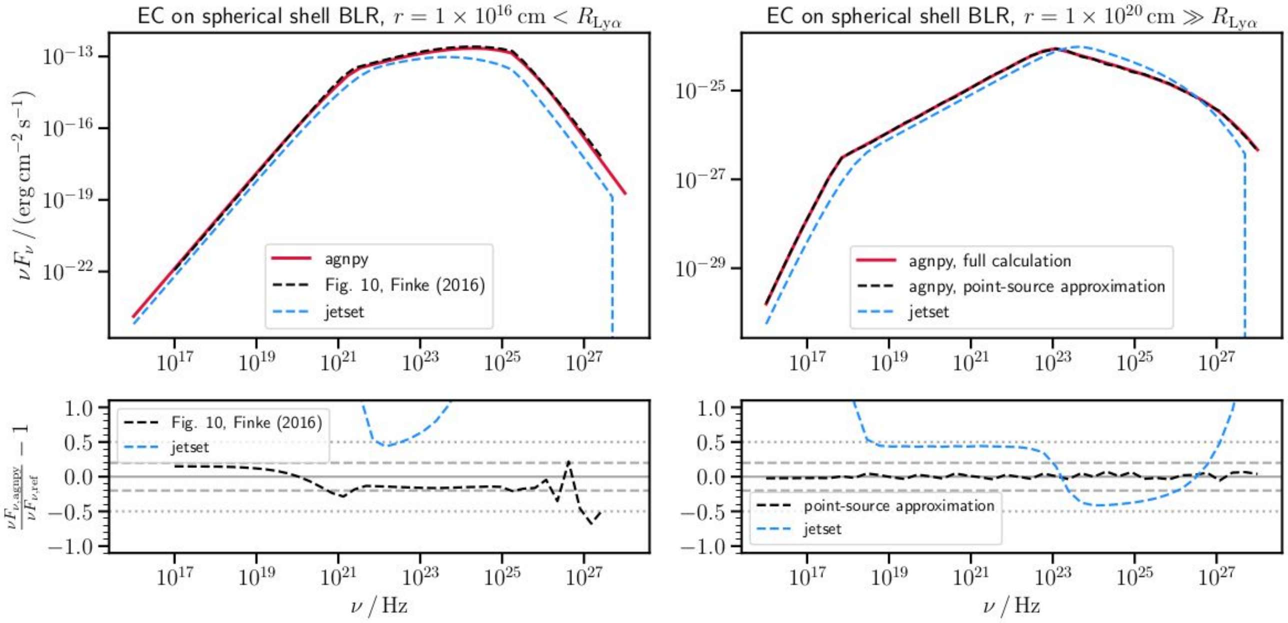
> <u>when using the same assumptions, an agreement within 30% is achieved</u>.



Validation

> When comparing against software relying on different assumptions an agreement within a factor 0.5 − 2 is achieved;

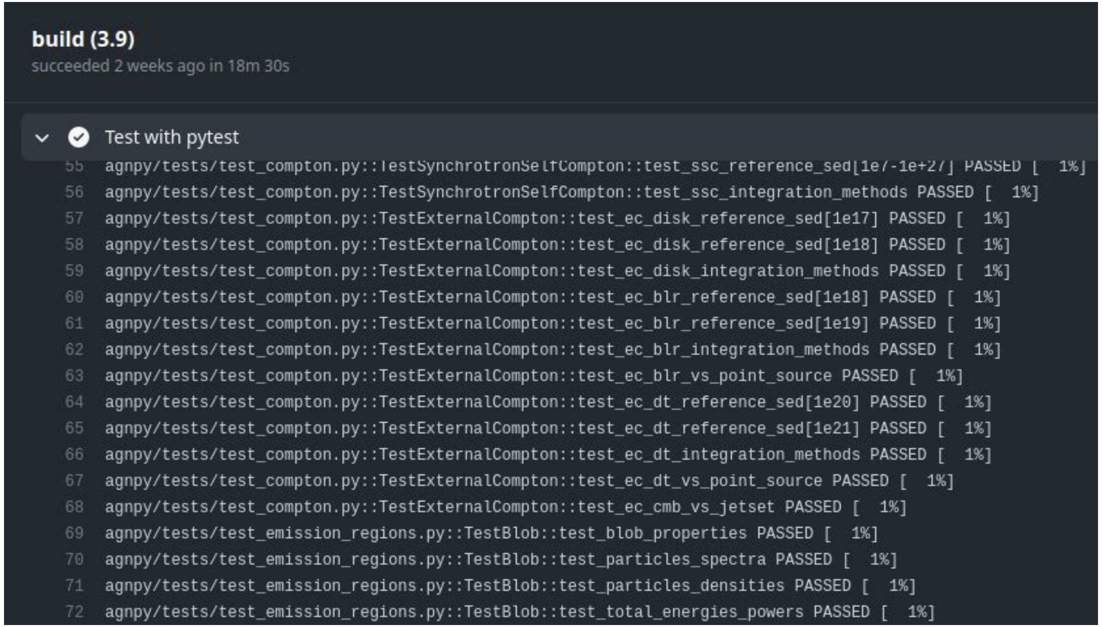
> internal consistency checks are implemented checking compatibility of different implementations of a similar scenario (e.g. arbitrary target vs point-like source).



Validation

> The validation is embedded, in the form of tests, in the software continuous integration (CI) cycle: they are performed each time a new change is merged into the master branch;

> The tests checks that the numerical deviations from the reference models remain within a certain fixed value.



```
build (3.9)
succeeded 2 weeks ago in 18m 30s

Test with pytest
agnpy/tests/test_compton.py::TestSynchrotronSelfCompton::test_ssc_reference_sed[1e7-1e+27] PASSED [  1%]
agnpy/tests/test_compton.py::TestSynchrotronSelfCompton::test_ssc_integration_methods PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_disk_reference_sed[1e17] PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_disk_reference_sed[1e18] PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_disk_integration_methods PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_blr_reference_sed[1e18] PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_blr_reference_sed[1e19] PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_blr_integration_methods PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_blr_vs_point_source PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_dt_reference_sed[1e20] PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_dt_reference_sed[1e21] PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_dt_integration_methods PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_dt_vs_point_source PASSED [  1%]
agnpy/tests/test_compton.py::TestExternalCompton::test_ec_cmb_vs_jetset PASSED [  1%]
agnpy/tests/test_emission_regions.py::TestBlob::test_blob_properties PASSED [  1%]
agnpy/tests/test_emission_regions.py::TestBlob::test_particles_spectra PASSED [  1%]
agnpy/tests/test_emission_regions.py::TestBlob::test_particles_densities PASSED [  1%]
agnpy/tests/test_emission_regions.py::TestBlob::test_total_energies_powers PASSED [  1%]
```

A balance of my experience: positive aspects

> It was **positively received by the community**:

> affiliated to the Astropy project;

> release paper published in A&A;

> used in 6 publications and several projects (MAGIC, VERITAS, CTA);

> received support from other developers (Andrea) and from “classical” modellers (Finke, Dermer, Matteo);

> **gathered a team of 5 stable contributors**;

Affiliated Packages Registry

The following table lists all currently registered affiliated packages. They are determined from the json file, which is the actual authoritative registry.

Total number of affiliated packages: **46**

agnpy

Website Repository PyPI

Modelling Active Galactic Nuclei radiative processes with python. agnpy focuses on the numerical computation of the photon spectra produced by leptonic radiative processes in jetted Active Galactic Nuclei (AGN).

Maintainer(s): Cosimo Nigro

Functionality: Specialized package | Astropy integration: Partial | Docs: Good | Tests: Good | Development: Heavy development | Python 3: Yes

Home ▸ All issues ▸ Volume 660 (April 2022) ▸ A&A, 660 (2022) A18 ▸ Full HTML

Free Access

Issue	A&A **Volume** 660, April 2022
Article Number	A18
Number of page(s)	20
Section	Numerical methods and codes
DOI	https://doi.org/10.1051/0004-6361/202142000
Published online	01 April 2022

A&A 660, A18 (2022)

agnpy: An open-source python package modelling the radiative processes of jetted active galactic nuclei

C. Nigro[1], J. Sitarek[2], P. Gliwny[3], D. Sanchez[4], A. Tramacere[5] and M. Craig[6]

+

Received: 11 August 2021 | **Accepted:** 29 December 2021

Abstract

Context. Modelling the broadband emission of jetted active galactic nuclei (AGN) constitutes one of the main research topics of extragalactic astrophysics in the multi-wavelength and multi-messenger domain.

All volumes

Special issues

Forthcoming articles

Press releases

Table of Contents

Article

Abstract | Full HTML | PDF (3.190 MB)

ePUB (7.124 MB) | References

Simbad Objects | NASA ADS Abstract Service

Metrics

Show article metrics

Altmetric 8

Services

A balance of my experience: negative aspects

> Just dumping software online does not really count as making it open-source. Documentation, testing, and distributions (requirements for modern software) demand a lot of technical commitment;

> the lifetime of these type of projects is the same as that of the academic careers of the developers (often not staff);

> people developing these tools are employed for technical work on other stuff and they do this in their “free” research time;

> funding agencies maybe still indifferent to the topic, but at least journals are changing their approach;

> I feel it would be good to establish better communication between developers and a general direction towards which these developments are going.

The way forward

> A new class of open-source modelling tools is opening the interpretation effort to the community.

> **consolidate the knowledge of traditional "modellers" and provide it to the community**;

> open-source software **can solve classes of problems where the old closed-source generation fails**, e.g. systematic analysis of large samples of sources;

> **can we consider this as a *first-generation* of open-source tools and start to think of a new one**?
- It can be built on *fundamental solvers*, each tasked with a particular mathematical / physical problem, that are developed compatibly with each other;

> maybe we can form a "collaboration" and eventually ask for funding?

> we have to adopt the mindset in which **interpretation software is as important as the data-analysis one**;

> **the software that we write for physical interpretation has become part of our experimental apparatus**;

> testing physical hypotheses is one of the underpinnings of the scientific method, but so is reproducibility of results. Open-source scientific software realises both.