The Journal of Open Source Software

Sensipy: simulate gamma-ray observations of transient astrophysical sources

Jarred G. Green [1,2], **Barbara Patricelli** [3,4], **Antonio Stamerra** [3,5], and **Monica Seglar-Arroyo** [6]

1 Max-Planck-Institut für Physik, Boltzmannstr. 8, 85748 Garching, Germany **2** Technische Universität München, 85748 Garching, Germany **3** INAF - Osservatorio Astronomico di Roma, Via di Frascati 33, 00078, Monteporzio Catone, Italy **4** University of Pisa, Largo B. Pontecorvo 3, 56127 Pisa, Italy **5** Cherenkov Telescope Array Observatory, Via Gobetti, 40129 Bologna, Italy **6** Institut de Fisica d'Altes Energies (IFAE), The Barcelona Institute of Science and Technology, Campus UAB, 08193 Bellaterra (Barcelona), Spain

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

We present sensipy, an open-source Python toolkit for simulating observations of transient astrophysical sources, particularly in the high-energy (HE, keV-GeV) and very-high-energy (VHE, GeV-TeV) gamma-ray sky. The most explosive events in our universe are often short-lived, emitting the bulk of their energy in a relatively small time window. Due to their often rapidly fading emission profiles, understanding how and when to observe these sources is crucial to both test theoretical predictions and efficiently optimize telescope time for detection.

The information extracted from the tools included in sensipy can be used to help astronomers understand the detectability of theoretical models, justify the feasibility of proposed observations, estimate detection rates (events/year) for various classes of sources, and provide scheduling in realtime during gamma-ray and multi-messenger observational campaigns.

Statement of need

The need for a toolkit like sensipy became clear when we were attempting to estimate the detectability of VHE counterparts to GW signals of binary neutron star mergers (BNS) with the upcoming Cherenkov Telescope Array Observatory (CTAO). While this toolklit began development with the aim to optimize a strategy for such joint detections with CTAO, the usefulness of the package became apparent and can be applied not just to BNS mergers, but also to GRBs, AGN flares, novae, supernovae, and more.

Between GW, neutrino, optical, and orbiting gamma-ray experiments, thousands of low-latency alerts are sent out to the greater community each year. However, very few of these events actually result in detections in the VHE gamma-ray regime. This is due to many factors, including the relative rarity of events, rapid decay of fluxes, delay in telescope repointing, uncertainty regions of alerts, and observatory duty cycles. In the face of these challenges, sensipy aims to help answer the following questions for gamma-ray astronomers interested in optimizing their follow-up campaigns:

- Given a theoretical emission model, what are the detection possibilities with a given instrument?
- How much observing time is needed to detect a source given you are delayed in starting observations?
- How much time after the onset of a given event does the source become undetectable?

- How can intrinsic source properties (eg distance, flux), and observing conditions (eg latency, telescope pointing) affect detectability?
- How can these results for catalogs of simulated events inform follow-up strategies in realtime?

Functionality

The two main inputs to any `sensipy` pipeline are:

- an instrument response function (IRF), which describes how a telescope performs under specific observing conditions, provided in a Data Formats for Gamma-ray Astronomy (DFGA) compatible FITS file
- intrinsic time-dependent emission spectra for a source, which can be provided in either a FITS or CSV format.

Given these inputs, `sensipy` toolkit builds upon the primitives provided by `numpy`, `scipy`, `astropy`, and `gammapy` to provide the following main functionalities. In addition, mock datasets are provided with working code examples, and batteries are included for easy access to these models and the publicly-available CTAO IRFs.

Sensitivity Curve Calculation with `sensipy.sensitivity`

Sensitivity curves represent the minimum flux needed to detect a source at a given significance (usually 5σ) given an exposure time t_{exp}. Such curves are often used to compare the performances of different instruments, and `sensipy` can produce them in two flavors: integral and differential sensitivity curves. The sensitivity itself depends heavily on the rapidly-changing spectral shape of an event, which itself can be highly affected by distance due to the extragalactic background light (EBL). All of these factors are automatically into account by the `Sensitivity` module of `sensipy`:

[Plot of differential and integral sensitivity curves for CTAO calculated with sensipy]

Simulating Observations with `sensipy.source`

Given that the user has already calculated the sensitivity curve for a specific event, `sensipy` can determine if the source is actually detectable given a starting time. In the positive case, the exposure time necessary for detection is also provided. This answers the fundamental question: if we begin observations with a latency of X min after an alert, what observation time is required in order to eke out a detection?

Working with large catalogs with `sensipy.detectability`

`sensipy` can further be used to estimate the overall detectability of entire classes of objects, given a catalog or survey of simulated events under various conditions. By performing and collating a large number of observation simulations for various events and latencies t_L, the toolkit can help produce visualizations which describe the optimal observing conditions for such events.

[Two example heatmap plots calculated with sensipy]

Realtime applications with `sensipy.followup`

Tables of observation times for an entire class of objects can also be used as lookup tables (LUTs) during telescope observations in order to plan observation campaigns. For example, imagine the following pipeline:

- a catalog of simulated spectra is processed with the above pipeline under various observation conditions, and a LUT is created
- a transient alert arrives during normal telescope operation and telescopes start observing the event position with a latency of t_L
- the LUT can be filtered and interpolated in realtime in order to quickly calculate an informed estimate on the exposure time needed for a detection

Such pipelines based on `sensipy` modules are already being internally evaluated within the MAGIC, Large-Size Telescope (LST), and CTAO collaborations for followup of both GW and GRB alerts.

GW scheduling

In addition, the functions included in `sensipy.followup` may be used in tandem with scheduling software like `tilepy` for poorly-localized events. In such cases, these scheduling tools create an optimized list of telescope pointings on the sky, while `sensipy` is used to simultaneously to optimize the exposure time needed at each new pointing. It is in this context that members of CTAO collaboration began development of `sensipy`.

[Show example of a GW pointing scheduling for a well-localized event, if possible overlay pointing durations on top of each pointing]

Citations

Citations to entries in paper.bib should be in rMarkdown format.

If you want to cite a software repository URL (e.g. something on GitHub without a preferred citation) then you can do it with the example BibTeX entry below.

For a quick reference, the following citation commands can be used:

```
- `@author:2001`  ->  "Author et al. (2001)"
- `[@author:2001]` -> "(Author et al., 2001)"
- `[@author1:2001; @author2:2001]` -> "(Author1 et al., 2001; Author2 et al., 2002)"
```

Acknowledgements

We acknowledge contributions from some people

Statement on AI

Let's say how we did not use generative AI to assist in the writing of this manuscript.

References