EUSO-Offline: A comprehensive simulation and analysis framework

Published by IOP Publishing for Sissa Medialab

Received: *September 7, 2023*
Accepted: *October 26, 2023*
Published: *January 2, 2024*

EUSO-Offline: A comprehensive simulation and analysis framework

The JEM-EUSO collaboration

E-mail: jeser@uchicago.edu, gfilippatos@mines.edu, thomas.paul@lehman.cuny.edu, guarino@na.infn.it

Abstract: The complexity of modern cosmic ray observatories and the rich data sets they capture often require a sophisticated software framework to support the simulation of physical processes, detector response, as well as reconstruction and analysis of real and simulated data. Here we present the EUSO-Offline framework. The code base was originally developed by the Pierre Auger Collaboration, and portions of it have been adopted by other collaborations to suit their needs. We have extended this software to fulfill the requirements of Ultra-High Energy Cosmic Ray detectors and very high energy neutrino detectors developed for the Joint Exploratory Missions for an Extreme Universe Observatory (JEM-EUSO). These path-finder instruments constitute a program to chart the path to a future space-based mission like POEMMA. For completeness, we describe the overall structure of the framework developed by the Auger collaboration and continue with a description of the JEM-EUSO simulation and reconstruction capabilities. The framework is written predominantly in modern C++ (compliled against C++17) and incorporates third-party libraries chosen based on functionality and our best judgment regarding support and longevity. Modularity is a central notion in the framework design, a requirement for large collaborations in which many individuals contribute to a common code base and often want to compare different approaches to a given problem. For the same reason, the framework is designed to be highly configurable, which allows us to contend with a variety of JEM-EUSO missions and observation scenarios. We also discuss how we incorporate broad, industry-standard testing coverage which is necessary to ensure quality and maintainability of a relatively large code base, and the tools we employ to support a multitude of computing platforms and enable fast, reliable installation of external packages. Finally, we provide a few examples of simulation and reconstruction applications using EUSO-Offline.

Keywords: Simulation methods and programs; Software architectures (event data models, frameworks and databases); Analysis and statistical methods

ArXiv ePrint: 2309.02577

https://doi.org/10.1088/1748-0221/19/01/P01007

Contents

1 Introduction

The Joint Exploratory Missions for an Extreme Universe Observatory (JEM-EUSO) [1] comprises a collection of experiments in pursuit of a future space-based cosmic ray observatory, such as the Probe of Extreme Multi-Messenger Astrophysics (POEMMA) [2]. The objective of such an observatory is elucidation of the origins and nature of Ultra-High-Energy ($E > 20$ EeV) Cosmic Rays (UHECR), and discovery of very high energy ($E > 20$ PeV) neutrinos originating from astrophysical transient sources [3].

Several pathfinder missions designed to establish the technologies and techniques to realize this objective have been completed or are in preparation. EUSO-Balloon, a first prototype instrument flew on a short duration (one night) high-altitude (40 km) balloon flight in 2014 [4]. In 2017, a long duration super pressure balloon flight was launched but gathered only limited data owing to an apparent flaw in the balloon [5]. A cross-calibration instrument, EUSO-TA is deployed adjacent to a Telescope Array (TA) fluorescence station [6]. Since 2019, Mini-EUSO [7] has been taking data on board the International Space Station. A second long duration super pressure balloon flight, EUSO-SPB2 was launched on May 13th from Wanaka, NZ [8, 9] but was prematurely terminated over the Pacific ocean due to a balloon leak after only 37 hours at float.

The first four of these pathfinders comprised telescopes capable of sensing the fluorescence light produced by excited atmospheric nitrogen as the UHECR-induced cascade of particles develops and deposits energy as it traverses the Earth's atmosphere. These Fluorescence Telescopes (FT) all employed Fresnel lenses for focusing, and increasingly refined versions of a Photo Detector Module (PDM) for sensing the impinging photons. EUSO-SPB2 carried a science payload comprising two telescopes. The first of these again employed the fluorescence detection technique, but in this case, using Schmidt optics and 3 PDMs. The second, a Cherenkov telescope (CT), was designed to detect the Cherenkov radiation emitted by the particle cascade using silicon photomultipliers capable of ultra-fast integration of the signal. This telescope can be pointed above or below the Earth's limb to detect showers produced when cosmic rays or neutrinos respectively interact in the Earth producing up-going particle cascades. While Cherenkov imaging has been utilized extensively on ground by gamma-ray telescopes such as MAGIC, HESS, or Veritas [10–12], EUSO-SPB2 was the first aerial mission to attempt this type of observation.

Simulating and reconstructing data from such a variety of instruments requires a highly configurable and modular software framework. One software design with a demonstrated history of success in this regard is the $\overline{\text{Off}}\underline{\text{line}}$ software developed by the Pierre Auger Collaboration starting in 2003 [13]. In this article, we describe an extension of the $\overline{\text{Off}}\underline{\text{line}}$ framework to accommodate the requirements of the JEM-EUSO missions. An earlier (and complementary) software package, ESAF, is discussed in [14, 15] with a comparison between both frameworks outlined in the appendix of [15].

This paper is organized as follows. In section 2 we provide an overview of the framework design and discuss how the modularity and configuration requirements are satisfied. Section 3 contains a description of the techniques used for swift installation of external dependencies, environment virtualization, configuration and build, as well the methods used for testing coverage and continuous integration. Section 4 contains a few examples of applications developed within the EUSO-$\overline{\text{Off}}\underline{\text{line}}$ framework. We offer some concluding remarks in section 5.

2 Framework

As noted in section 1, the EUSO-$\overline{\text{Off}}\underline{\text{line}}$ framework was inherited from the $\overline{\text{Off}}\underline{\text{line}}$ code project initiated by the Pierre Auger Collaboration [13], and further developed by the NA61/SHINE software team [16] as well as some cosmic ray experiments using the radio detection technique [17]. The overall framework design has been largely retained for the EUSO-$\overline{\text{Off}}\underline{\text{line}}$ software. For the sake of completeness, we first provide an overview of the framework design.

2.1 Overview

The framework consists of the following principal components: a collection of event simulation and reconstruction algorithms contained in *Modules*; a *Run Controller* which commands the modules to execute in a particular sequence; an *Event Data Model* (EDM) from which modules read information from other upstream modules and to which they write their results; a *Detector Description* which provides an interface to information about the detector configuration as well as other data describing various conditions like atmospheric properties; and a *Central Config* which directs the modules and framework components to their configuration data and records provenance. The scheme of pipelining a collection of modules that communicate via the EDM and read from the Detector

Description separates algorithms from data. This approach is not particularly Object Oriented, but it does effectively support the collaborative development of simulation and reconstruction algorithms. The framework components are illustrated in figure 1, and discussed in more detail below.

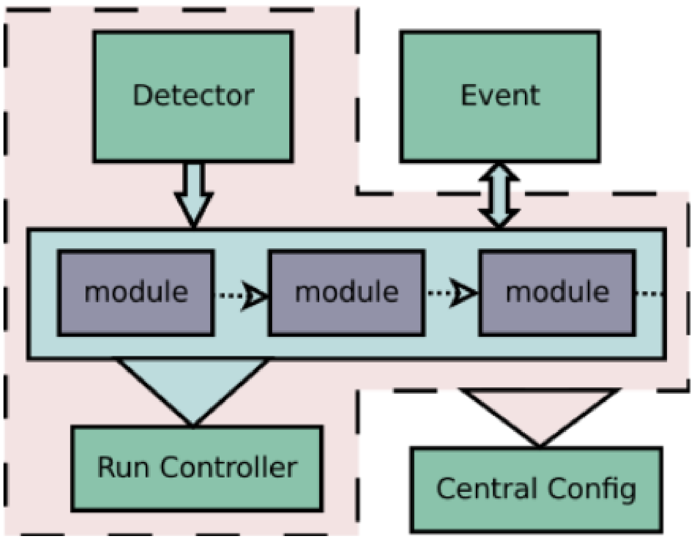


Figure 1. General organization of the EUSO-$\overline{\text{Off}}\underline{\text{line}}$ framework. The blue-shaded region indicates interaction among modules, the Run Controller, the Detector Description and the EDM. The pink-shaded region indicates the components of the framework which are configurable via the Central Config. See the text for detailed explanation.

2.2 Modules and run control

Simulation and reconstruction steps are generally organized in a simple pipeline. Collaborators prepare algorithms in Modules. Modules inherit a common interface that declares the methods they carry out during processing, and which provides a macro to register each module with the framework thereby making it aware of which modules exist for potential use at run time.

The modular design facilitates the comparison of algorithms and supports building a variety of applications by combining modules in various sequences. One can, for instance, swap out a module for reading in simulated showers with a module to simulate laser shots (or other light sources) in the instrument field of view (FoV). Specification of a module sequence is done at run time and does not require compiling any code.

Module sequences are directed by the *Run Controller*, which invokes module execution according to a set of user-provided instructions. A XML-based tool is used for specifying sequencing instructions, including support for nested loops and loops which iteratively process the data. Modules can signal the Run Controller to break a loop or skip downstream modules if certain conditions arise. For instance, if one step in a collection of reconstruction algorithms fails, a signal to skip to the next event can be relayed to the Run Controller.

The approach of using simple instructions provided in an XML file together with a mechanism for modules to signal the Run Controller has proved sufficiently flexible for our applications. However, we are planning to introduce Python bindings which will allow a more flexible system in the future.

2.3 Event data model and detector description

The $\overline{\text{Off}}\underline{\text{line}}$ framework provides parallel hierarchies for accessing data. The *Event Data Model* is used for reading and writing information specific for each event. The read-only *Detector Description* supports retrieval of static configuration data or relatively slowly varying information such as detector health or atmospheric monitoring data.

2.3.1 Detector description

Configuration and conditions information accessible via the Detector Description can include detector materials and geometry as well as time-dependent information such calibration data, background measurements, and atmospheric monitoring data. The Detector description is meant to provide a single endpoint for these sorts of data, preventing possible errors resulting from inadvertently providing the same information in different pieces of code or configuration files.

The Detector Description interface is structured to follow the hierarchy normally associated with actual detector components. Requests sent to the Detector Description are relayed to a registry of *Managers*, each of which is capable of extracting particular data from a particular source. In this way, the user sees a single interface, while a back-end of Managers deals with the potentially involved task of reading data from different sources. For example, different Managers might read data from XML files or from databases, thus isolating the (possible) complexity of data access in manageable units of code. The general scheme is illustrated in figure 2.

Managers in the registry are polled for information using a chain of responsibility, as indicated by the arrow in figure 2. When the registry receives a request, it asks the first manager in the registry if it can answer the request. If it cannot, it proceeds to the next manager in the chain. This allows for fall-back solutions to common problems. For instance, time-dependent information stored in a database may have missing entries for some time intervals. This can be addressed by providing a fall-back manager downstream from the manager that provides database access. Specification of which data sources are accessed and in what order are provided in a configuration file.

A plug-in mechanism in the Atmosphere supports *Atmosphere Models*. These models are used to compute fluorescence and Cherenkov yields, Rayleigh and Mie scattering and absorption as well as ozone absorption, as further described in section 4.1. These models can be configured to access parametric data stored in configuration files or time-dependent data stored in databases. A template is provided to assist collaborators in developing different interchangeable Atmosphere Models, in much the same vein as interchangeable simulation and reconstruction Modules can be developed. Model selection is afforded at run time through a configuration file read by a model registry which works in the same manner as the manager registry discussed above. More details on this plug-in mechanism can be found in reference [13].

Lazy evaluation is used to access all detector data so that only requested data are actually loaded into the Detector Description and cached for future access, possibly flagged with a time interval for which the data are valid. Data that are valid only for a specific time interval are flushed and reloaded when an event is read which contains a time stamp outside the interval of validity. As the Detector is meant to provide a read-only interface, the entire interface is logically *const*, though it is physically non-*const* since it must load itself from the various data sources.

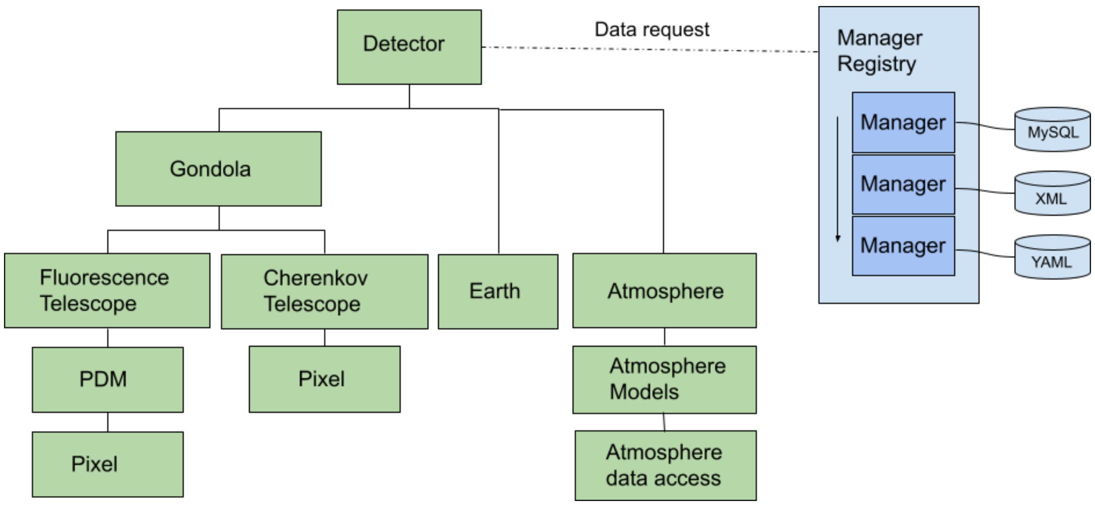


Figure 2. Manager mechanism for the case of the EUSO-SPB2 instrument. Left: the User Interface to detector information comprises a hierarchy of objects that follow the hierarchy naturally associated with the Detector. Modules can query this interface for information needed during data processing. The science payload lives under the Gondola, and users can query for things like the Gondola orientation, or the configuration of the FT and CT instruments that live below it in the hierarchy. Notice that, the Atmosphere and the Earth are also considered to be parts of the detector (since the Atmosphere functions as a giant calorimeter). Notice that the Detector interface in this figure is structured to follow the hierarchy associated with the SPB2 instrument. The structure can be selected using the $\overline{\text{Off}}\underline{\text{line}}$ configuration mechanism. For example, one can select the EUSO-TA instrument configuration in an XML file, and an interface will be constructed which is appropriate to that instrument (that is, with no Cherenkov telescope and no gondola objects.) An additional mechanism is also available to perform pre-processing of data read by various Managers. We use this primarily to provide higher-level access to atmospheric data. Various *Atmosphere Models* can be accessed to compute quantities such as fluorescence yield, scattering, and absorption. The Earth object can be queried for albedo estimates for the sea or different sorts of terrain. Right: the Detector sends its requests to the Manager Registry, which finds the appropriate data source and returns the requested data to the Detector interface. See the text for more detail.

2.3.2 Event data model

The *Event Data Model* (EDM) contains the raw, calibrated and reconstructed data as well as Monte Carlo true parameters for the case of simulations. It provides the backbone for communication among physics modules. The structure of the EDM comprises a collection of classes organized with the hierarchy normally associated with the detector, similar to the Detector Description interface. Physics modules access the event information through a reference to the top of the hierarchy, which is provided to the module interface by the Run Controller. Since the Event enables inter-module communication, reference semantics are used to access objects in the EDM, and constructors are private in order to prevent accidental copying.

The EDM is instrumented with a protocol allowing modules to discover its constituents at any point in processing, and thereby determine whether the input data required to carry out the desired processing are available (and take appropriate action if not). The event interface cannot be modified by Modules. This somewhat limits flexibility, but it does help to ensure interchangeability of physics modules, an essential feature of $\overline{\text{Off}}\underline{\text{line}}$ as it allows collaborators to try out different approaches to the same problem without interfering with one another.

2024 JINST 19 P01007

Offline is also equipped to populate parts of the event by reading formats employed by most popular air shower simulation packages [18–21], as well as data formats used by the different JEM-EUSO pathfinders. A *DataWriter* module is available to write out the event in the same ROOT formats employed by the JEM-EUSO instruments; simulation true parameters is stored in a separate ROOT tree in the same file(s) containing reconstruction and experimental configuration information.

2.4 Configuration

Offline provides an XML and XML Schema based system to organize data used to configure the software for the variety of simulation and reconstruction tasks required to support the various JEM-EUSO instruments. The *Central Config* configuration tool points modules and framework components to the location of their configuration data and connects to Xerces-based [22] XML parsers to assist in reading information from these locations. The Xerces API is wrapped with a custom interface which provides a simpler interface at the cost of somewhat reduced flexibility. Our XML parser also provides conveniences such as automatic casting to JEM-EUSO data containers as well as automatic unit conversion, obviating the need to enforce a prescribed units convention. Syntax and content checking of the configuration files are implemented using W3C XML Schema validation. Xerces was adopted early on by the Pierre Auger Observatory Offline [13] partly because of its comparatively complete support for the XML Schema standard.

The *Central Config* records *all* configuration data accessed during a run and stores them in an XML log file. The log file can subsequently be read in to reproduce a run with an identical configuration. This allows collaborators to exchange configuration data and compare results. The logging mechanism is also used to record the external libraries which are used for each run. We have found such provenance tracking to be indispensable for simulation production and data analysis; years of experience have shown that it always happens that someone eventually needs to look up precisely what configuration was used to produce a simulation or process data.

3 DevOps

In this section, we discuss the tools we have adopted for code distribution, continuous integration and deployment (CI/CD), code building, and dependency handling.

The EUSO-Offline code repository is hosted on GitLab [23]. In addition to repository hosting, we employ most of the GitLab project management tools. Code development is conducted on dedicated branches documented in merge requests and peer-reviewed before merging into the main branch. Issues are tracked using the GitLab issue tracking system, which allows for tight integration of issues and Merge Requests (MRs).

A significant effort has been devoted to establishing thorough testing coverage. This is crucial for all software projects of any size, and for the case of JEM-EUSO the need is particularly pronounced owing to the multiple experimental configurations which must be supported. The GitLab CI/CD is used to build the code using both the gcc and clang compilers with each check-in, as well as to run unit tests, regression tests, linting and static analysis using Clang-Tidy [24] and Clang Static Analyzer [25]. The built-in clang sanitizers are employed to detect run-time errors. Older unit tests are implemented with the help of CppUnit [26], while more recently implemented tests are based on GoogleTest [27]. Regression tests run full sequences of modules to detect unexpected changes in simulation and reconstruction results. We have developed an in-house tool to assist with this. The build system employs CMake [28] to write the build tool configuration (for example GNU Make [29] or Ninja [30]).

The external packages upon which EUSO-$\overline{\text{Off}}\underline{\text{line}}$ is built were to some extent inherited from the Pierre Auger $\overline{\text{Off}}\underline{\text{line}}$ software. The choices of externals were dictated not only by functionality and open-source requirements, but by the best guess at longevity. Some functionality is provided to the client code via façads, as in the case of reading XML files, or via a bridge, as in the case of the detector description described previously. The collection of external libraries includes Xerces [22] for XML parsing, CLHEP [31] for expressions evaluation, Geant4 [6, 32, 33] for detailed detector simulation, Boost [34] for its numerous C++ extensions, mysql [35] and sqlite [36] for database implementations, pytorch-cpu [37] for machine learning applications and ROOT [38] for file input and output. The use of ROOT has been limited to input/output since the first design of $\overline{\text{Off}}\underline{\text{line}}$ in the early 2000s owing to issues at the time with bugs, the non-idiomatic C++ design, and concerns about vendor lock-in; more recently the extensive collection of readily available solutions provided by the huge open-source community mostly obviates the need for ROOT utilities, with the arguable exception of input/output.

We use Anaconda (a.k.a. conda) [39] together with the Libmamba [40] dependency solver to install all externals as pre-compiled binaries and to create a virtual environment largely isolated from the local system, which in turn simplifies the package detection performed by CMake. This allows for dependency installation that takes a few minutes rather than the many hours typically required to build large packages like Geant4 and ROOT from source. We also generate conda packages of in-house software like EtoS and EtoT [41].

4 Simulation and reconstruction capabilities

In this section, we provide an overview of some of the simulation and reconstruction capabilities of the EUSO-$\overline{\text{Off}}\underline{\text{line}}$ framework. One purpose of this section is to illustrate some of the advantages of algorithm modularization and flexible configuration. We also demonstrate some of the benefits of implementing simulation, reconstruction and analysis of all the JEM-EUSO missions in a single, common framework.

The majority of the simulation modules were originally developed by members of the Pierre Auger collaboration, starting with input of shower simulation in any of several popular shower generator output formats and continuing with the light production and the transport of photons through the atmosphere [42]. For the JEM-EUSO missions, it was necessary to extend the code in a number of ways, itemized below:

- The effects of Ozone absorption is included as a *model* accessible via the Atmosphere interface.
- The Earth interface is added to the DD to provide access to albedo estimates of different sorts of terrain which can be used for simulation of Cherenkov light reflected from the Earth's surface into an orbiting telescope.
- A reader for the EASCherSim [21, 43] Monte Carlo generator is available. This generator supports the simulation of Cherenkov light emitted at very small angles with respect to the shower axis.

- A configurable Geant4-based telescope simulation module is available, which can model any of the JEM-EUSO instrument designs by simply specifying the desired XML configuration.
- Custom Fresnel optics simulation (written originally for ESAF [44]) is incorporated into the Geant4 simulation of the telescopes.
- Background simulation modules (both night-glow and spot-like) is prepared and can model background light observed by the Fluorescence telescopes in either tilt or nadir pointing modes.
- Simulation of the trigger logic for the different instruments has been prepared.
- A convolutional neural network was developed and trained on simulated data to perform fast in-flight classification of events in order to prioritize event downlinking [45].

4.1 Simulation

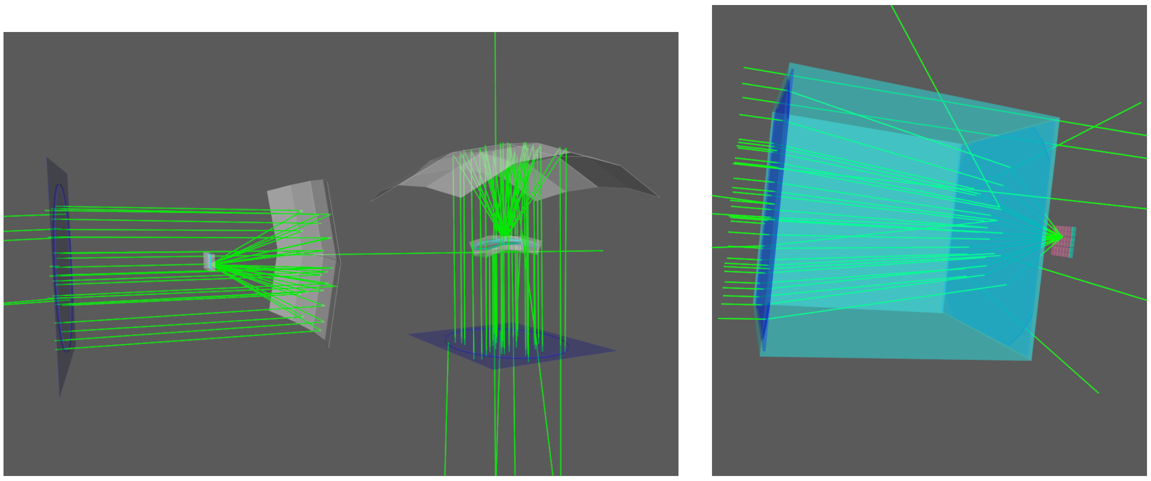
Simulations generally commence either with input from a cosmic ray air shower generator, or simulation of calibration laser shot. A common input/output interface, discussed in some detail in [13], connects the $\overline{\text{Off}}\underline{\text{line}}$ Event to the appropriate back-end reader. We currently provide readers for the air shower generators `CORSIKA` [18], `CONEX` [19] and `EASCherSim`.[1] Laser simulation is performed by a module distributed with the $\overline{\text{Off}}\underline{\text{line}}$ software. Simulation of fluorescence detection is then performed by a sequence of Modules under the direction of the Run Controller and sequencing instructions provided in a configuration file, as explained in section 2 and outlined in more detail below.

The first module in the FT simulation sequence positions the generator-level shower either by defining its distance and azimuthal orientation with respect to the telescope's optical axis (for nearly horizontal showers observed by balloon-borne instruments) or by pinning the shower axis to a position on the Earth (for the case of vertical showers, as measured for instance by EUSO-TA). At this step, it is possible to randomize the positioning parameters and to impose cuts based on the instrument field of view (FoV). FoV cuts increase the speed of simulations with randomized positioning parameters, since subsequent modules can then deal only with visible portions of the shower.

Once the shower is positioned, a downstream module converts the charged particle distributions provided by the shower generators `CORSIKA` or `CONEX` into fluorescence and Cherenkov photons. These conversions are performed with the help of fluorescence yield and Cherenkov generation Atmosphere Models using the configurable back-end of the Atmosphere interface described in section 2.3.1. A number of different models are available and selectable at run time using a configuration file. At present, we estimate the Cherenkov emission using a parametric approach following Hillas or Nerling [46, 47]. The fluorescence yield is estimated using the approach provided in Auger-$\overline{\text{Off}}\underline{\text{line}}$, in which wavelength-dependent measurements from Airfly [48] are combined with the Nagano absolute fluorescence yield estimate [49]. Ozone absorption is computed using data from the ozone sounding program of the US National Oceanic and Atmospheric Administration (NOAA) [50]. Details are provided in [51].

Subsequent modules simulate the light propagation from the shower to the detector aperture. Again, several configurable Atmosphere Models are available to assist in the computation of the effects

[1] https://c4341.gitlab.io/easchersim/index.html.



(a) Geant4 simulation of the EUSO-SPB2 instrument, which employs Schmitt optics. The green lines represent photons; for clarity of the image, the number of photons injected is far fewer than in a typical event. Mirrors and cameras are shown in grey, and the blue regions represent the entrance pupils. The instrument on the right is the Fluorescence telescope, which focuses light on 3 Photo Detection Modules. The Cherenkov telescope is on the left, and employs silicon photomultipliers. The Cherenkov telescope can be rotated to view the regions just above and below the Earth's limb.

(b) Geant4 simulation of the EUSO-TA instrument, which comprises 2 Fresnel lenses and 1 Photo Detection Module.

Figure 3. Geant4 simulations of the EUSO-SPB2 (a) and EUSO-TA (b) instruments.

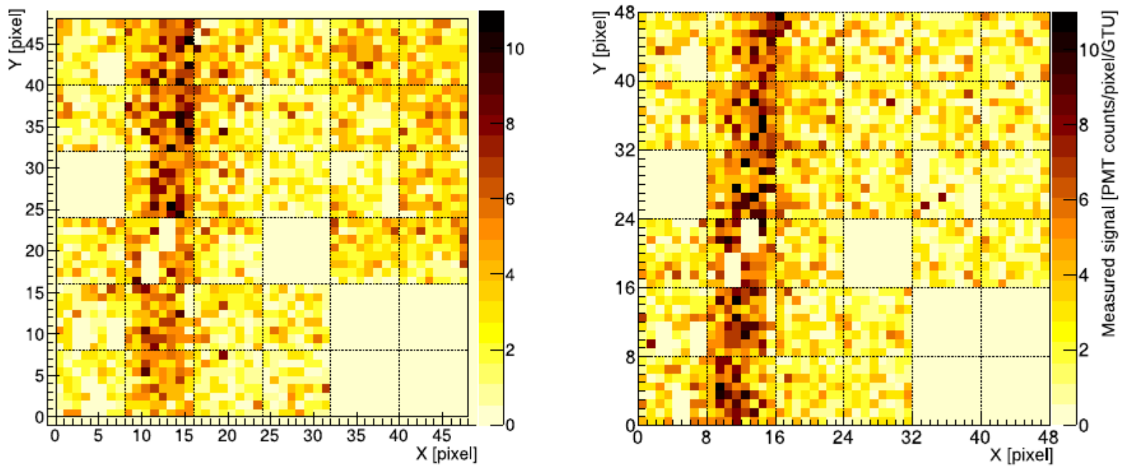
of Rayleigh and Mie scattering and absorption as well as ozone absorption, the latter of which is particularly consequential for a space-based observatory `EASCherSim` contains its own Rayleigh, Mie and ozone models. At this point, the most computationally intensive parts of the simulation are complete and results can be written to file by the *DataWriter* described in section 2.3.2. Using these partially simulated events, we can more quickly develop downstream detector simulation and triggering algorithms.

Once the number of photons arriving at the detector aperture has been computed, full detector simulation can be performed. Our method-of-choice for simulating light propagation from the aperture to the sensors employs Geant4 for ray tracing through the various optical interfaces. Simulated event examples of the EUSO-SPB2 payload and the EUSO-TA instrument are shown in figure 3. The Geant4 simulation can be configured to model all of the JEM-EUSO missions.

Simulation of the camera efficiency, electronics response, and digitization is performed by our own custom modules which are tuned using laboratory measurements.

Downstream modules perform background simulations. Night sky background is accounted for by appending the signal trace of each pixel with a background component derived from measurements, including the effects of tilting the optical axis. Spot-like events from terrestrial light sources are simulated using data recorded by different JEM-EUSO instruments. The pre-trigger fully simulated event can be written to file at this stage.

Triggering algorithms are next applied to the background-contaminated signal. This step takes advantage of the modular design, as different trigger modules can be used together or separately



(a) Data (b) Simulation

Figure 4. A UHECR track with an energy of 10^{18} eV and an impact point 2.6 km from the detector. The color code shows the counts per pixel per time frame of 2.5 μs(Gate Time Unit). The shower parameters were provided to us by the TA collaboration based on their reconstruction of this shower recorded at the TA site Black Rock Mesa. The zenith angle of the shower was 8° and its azimuth was 82°. Panel (a) shows the track as recorded in EUSO-TA while panel (b) shows the simulation of such a shower within EUSO-$\overline{\text{Off}}\underline{\text{line}}$. Reprinted from [6], Copyright (2018), with permission from Elsevier.

based on the instrument in question and the algorithm under evaluation. Triggered events can be written to file in the same format as the JEM-EUSO mission data, or they can be passed directly to subsequent reconstruction modules, discussed in section 4.2. Figure 4 shows a comparison between a shower recorded by EUSO-TA [6] and a shower of (approximately) equivalent energy and geometry simulated with EUSO-$\overline{\text{Off}}\underline{\text{line}}$. Parameters for the shower simulation are based on reconstruction results provided by the Telescope Array collaboration [52].

4.2 Reconstruction

The structure of EUSO-$\overline{\text{Off}}\underline{\text{line}}$ enables straightforward validation of reconstruction processes using simulated data since simulations record ground-truth in the Event Data Model. For example, photons in the simulation keep track of whether they originate from fluorescence, Cherenkov, or laser, so the photons arriving in each pixel can be broken into components according to their origin. The simulation also keeps track of whether a photoelectron was produced due to a signal of interest or background. As mentioned above, reconstruction can be performed in the same run as simulation or can commence from libraries of fully (or even partially) simulated events. Reconstruction of simulated and real data can be naturally broken up into a pipeline of processing modules, each of which can be approached using different algorithms which can be compared to one another using the mechanisms discussed in section 2. We outline here the modules currently in use. In the final sub-section we briefly discuss using machine learning inside the EUSO-$\overline{\text{Off}}\underline{\text{line}}$ framework.

The goal of the reconstruction is to identify the main properties of the incoming air shower, which are arrival direction, primary energy, and composition. But before this can be done, the pixels containing the light of the observed air shower (or laser) need to be identified and the backgrounds

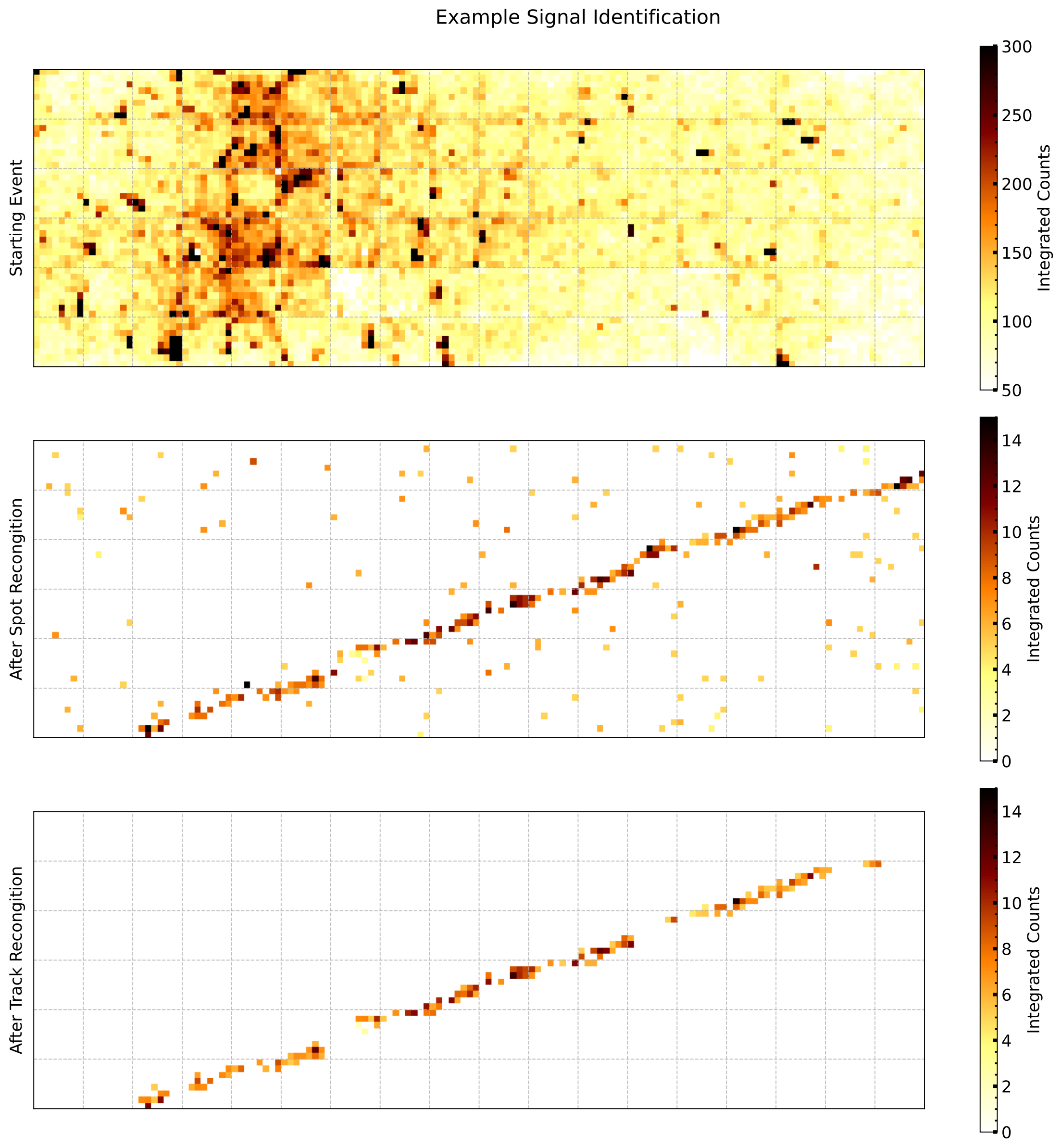


Figure 5. Recorded laser track on the EUSO-SPB2 focal surface during the field campaign in the summer of 2022 at the TA site. The x and y axis represent the camera x and y coordinates in units of pixels. Telescope pointing at nadir, laser located 24 km away travelling nearly horizontally. Integrated counts over 128 frames (top), pixels remaining after spot identification (middle), and pixels remaining after track identification (bottom).

need to be removed, both for simulated and real data. An example of progression from noisy data to identified track is shown in figure 5.

The background removal is performed on a pixel-by-pixel basis requiring a threshold for the signal based on the mean and standard deviation of the individual pixel trace. By default a 5σ excess is required but is a configurable parameter. The background removal process is independent of the trigger conditions. Depending on the parameters used, an event which passes the hardware trigger, or simulated trigger, may be entirely or partially removed by the background subtraction.

In order to identify a track within the remaining pixels, the subsequent modules clean the track by requiring those pixels containing the track not to be isolated. Only pixels that are part of a cluster of at least N pixels within a given radius are kept. In addition, only clusters that have neighboring clusters in time are considered to guarantee a moving track in the camera as expected for an air shower signal. An example of such a “cleaned” track is shown in the middle of figure 5. Finally, a line is fitted to the “cleaned” track and all remaining outlier pixels are removed.

Subsequent modules determine the properties of the primary particle that created the identified track in the data. The first step is a geometrical reconstruction deploying a well-established standard method. The Shower Detection Plane (SDP) is defined using the pointing of each signal pixel weighted by its calibrated signal strength. The defined SDP allows one to determine an elevation angle for each signal pixel. These angles can be used in the next module to determine the shower direction based on a time-angle fit. More details on this method can be found for example in [53]. The technique is also used by the Pierre Auger and TA collaboration for their geometric reconstruction.

With the geometry of the shower in hand, a final module uses the number of photons detected to estimate the energy of the shower (or calibration laser shot). This is done by summing all the signals recorded at the detector and assuming isotropic emission from the source, which is optimal for lasers. For the case of an EAS, the data are fitted to a Gaiser Hillas function, the maximum of which provides the shower $X_{\max}$ parameter. The total energy deposited by the shower is determined by integrating the fitted function [54].

4.3 Machine learning methods

Using the EUSO-Offline machinery, we have developed a convolutional neural network for in-flight event classification during the EUSO-SPB2 [45] mission. This neural network is trained on simulated data prior to flight and runs onboard the instrument using raw data.

EUSO-Offline is currently distributed with the torch libraries [55]. By utilizing Just In Time (JIT) compilation tracing, models trained in a variety of environments can be loaded into EUSO-Offline for inference. This added level of flexibility allows for easy testing of models on a large variety of data, without the added overhead that is needed for training complex models, such as GPU-accelerated computation. JIT compilation also allows for different architectures to be applied to data and simulations, allowing for EUSO-Offline to utilize the latest available machine learning methods without changes to the framework.

The modular nature of Offline allows for a greater deal of flexibility in utilizing machine learning techniques. For example, the onboard classification scheme for EUSO-SPB2, which utilized a recurrent convolutional neural network, exists as a module which can be added to any sequence, at any point. This allows for various models to be applied to simulated data quickly, which was useful in the development of the in-flight classification method. The binary classifier can also be applied to recorded data in parallel to the traditional track finding methods described in section 4.2, and the performance of the two methods can be directly compared.

While the current use cases of machine learning in EUSO-Offline have been focused on binary classification, the machinery exists for more complex analysis. We anticipate applying machine learning methods to address some of the reconstruction tasks discussed above in section 4.2.

5 Conclusion

We have adapted the $\overline{\text{Off}}\underline{\text{line}}$ framework originally developed by the Pierre Auger Observatory to the requirements of the JEM-EUSO missions. This framework provides the essential machinery to facilitate collaborative development of algorithms to address the variety of simulation and reconstruction use-cases required to analyze data from current and pending JEM-EUSO mission configurations. The modular design provides a straightforward way to compare different approaches to a given problem, and the configuration machinery provides the flexibility necessary to deal with the variety of simulation and reconstruction applications as well as the different instrument designs. The user interface to event data and time-dependent detector and atmospheric data is kept as simple as is feasible, while the back-end handles the complexity required to deal with different data sources and formats. This software has been used in production for the analysis of data from the EUSO-Balloon, EUSO-TA, EUSO-SPB1 and EUSO-SPB2 missions.

While at the moment the software package is available to members of the JEM-EUSO collaboration, a strategy is being developed to follow the OpenData/OpenSource [56] principles. This requires a close communication with the Pierre Auger collaboration which is proprietary of the common, base part of the code. Its current policy requires the acceptance of the Pierre Auger Collaboration before using the code. In the mean time other avenues like publishing only independent parts of the package are being investigated. Furthermore the outlined practice in section 3, following the industry standard for software development, provide the basis for a stable, maintainable code base. Once the code is public, its design principle of modularity and high configurability as well as a automatic generated API documentation directly guarantees that $\overline{\text{Off}}\underline{\text{line}}$ follows the FAIR [57] principles.

Acknowledgments

First and foremost the authors would like to thank the Auger collaboration for providing the framework code as well as a majority of the simulation code. This allowed quick development and the confidence of using and adapting a well-tested software package.

This work was partially supported by Basic Science Interdisciplinary Research Projects of RIKEN and JSPS KAKENHI Grant (22340063, 23340081, and 24244042), by the ASI-INAF agreement n.2017-14-H.O, by the Italian Ministry of Foreign Affairs and International Cooperation, by the Italian Space Agency through the ASI INFN agreements Mini-EUSO n. 2016-1-U.0, n. 2017-8-H.0, OBP (n. 2020-26-Hh.0) and n. 2021-8-HH.0, by NASA award 11-APRA-21730058, 16-APROBES16-0023, 17-APRA17-0066, NNX17AJ82G, NNX13AH54G, 80NSSC18K0246, 80NSSC18K0473, 80NSSC19K0626, and 80NSSC18K0464 in the U.S.A., by the French space agency CNES, by the Deutsches Zentrum für Luft- und Raumfahrt, the Helmholtz Alliance for Astroparticle Physics funded by the Initiative and Networking Fund of the Helmholtz Association (Germany), by Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany Excellence Strategy -EXC-2094-390783311, by National Science Centre in Poland grants 2017/27/B/ST9/02162 and 2020/37/B/ST9/01821, as well as VEGA grant agency project 2/0132/17, and by State Space Corporation ROSCOSMOS and the Interdisciplinary Scientific and Educational School of Moscow University "Fundamental and Applied Space Research".

References

[1] C. Fuglesang, *The EUSO program: Imaging of ultra-high energy cosmic rays by high-speed UV-video from space*, *Nucl. Instrum. Meth. A* **873** (2017) 1.

[2] POEMMA collaboration, *The POEMMA (Probe of Extreme Multi-Messenger Astrophysics) observatory*, *JCAP* **06** (2021) 007 [arXiv:2012.07945].

[3] T.M. Venters et al., *POEMMA's Target of Opportunity Sensitivity to Cosmic Neutrino Transient Sources*, *Phys. Rev. D* **102** (2020) 123013 [arXiv:1906.07209].

[4] J.H. Adams et al., *A Review of the EUSO-Balloon Pathfinder for the JEM-EUSO Program*, *Space Sci. Rev.* **218** (2022) 3.

[5] JEM-EUSO collaboration, *EUSO-SPB1 mission and science*, *Astropart. Phys.* **154** (2024) 102891 [INSPIRE].

[6] G. Abdellaoui et al., *EUSO-TA — First results from a ground-based EUSO telescope*, *Astropart. Phys.* **102** (2018) 98.

[7] S. Bacholle et al., *Mini-EUSO Mission to Study Earth UV Emissions on board the ISS*, *Astrophys. J. Suppl.* **253** (2021) 36 [arXiv:2010.01937].

[8] J. Eser, A. Olinto and L. Wiencke, *Science and mission status of EUSO-SPB2*, *PoS* **ICRC2021** (2021) 404 [arXiv:2112.08509].

[9] J. Eser, A. Olinto and L. Wiencke, *Overview and First Results of EUSO-SPB2*, *PoS* **ICRC2023** (2023) 397 [arXiv:2308.15693].

[10] E. Lorenz and M. Martinez, *The MAGIC telescope*, *Astron. Geophys.* **46** (2005) 621.

[11] G. Vasileiadis, *The HESS experimental project*, *Nucl. Instrum. Meth. A* **553** (2005) 268.

[12] J. Holder et al., *The first VERITAS telescope*, *Astropart. Phys.* **25** (2006) 391 [astro-ph/0604119].

[13] S. Argiro et al., *The Offline Software Framework of the Pierre Auger Observatory*, *Nucl. Instrum. Meth. A* **580** (2007) 1485 [arXiv:0707.1652].

[14] C. Berat et al., *ESAF: Full Simulation of Space-Based Extensive Air Showers Detectors*, *Astropart. Phys.* **33** (2010) 221 [arXiv:0907.5275].

[15] S. Abe et al., *Developments and results in the context of the JEM-EUSO program obtained with the ESAF simulation and analysis framework*, *Eur. Phys. J. C* **83** (2023) 1028 [arXiv:2311.12656] [INSPIRE].

[16] R. Sipos et al., *The offline software framework of the NA61/SHINE experiment*, *J. Phys. Conf. Ser.* **396** (2012) 022045.

[17] Pierre Auger collaboration, *Advanced Functionality for Radio Analysis in the Offline Software Framework of the Pierre Auger Observatory*, *Nucl. Instrum. Meth. A* **635** (2011) 92 [arXiv:1101.4473].

[18] D. Heck et al., *CORSIKA: A Monte Carlo code to simulate extensive air showers*, Tech. Rep. FZKA-6019, FZKA, Karlsruhe,Germany (1998).

[19] T. Bergmann et al., *One-dimensional Hybrid Approach to Extensive Air Shower Simulation*, *Astropart. Phys.* **26** (2007) 420 [astro-ph/0606564].

[20] S.J. Sciutto, *AIRES: A system for air shower simulations*, astro-ph/9911331 [DOI:10.13140/RG.2.2.12566.40002].

2024 JINST 19 P01007

[21] A.L. Cummings, R. Aloisio and J.F. Krizmanic, *Modeling of the Tau and Muon Neutrino-induced Optical Cherenkov Signals from Upward-moving Extensive Air Showers*, *Phys. Rev. D* **103** (2021) 043017 [arXiv:2011.09869].

[22] xerces, https://xerces.apache.org/xerces-c/.

[23] gitlab, https://about.gitlab.com/why-gitlab/.

[24] clang tidy, https://clang.llvm.org/extra/clang-tidy/.

[25] clang-static analyzer, https://clang-analyzer.llvm.org/.

[26] CPPUnit, https://cppunit.sourceforge.net/doc/1.8.0/.

[27] gtest, https://google.github.io/googletest/primer.html.

[28] cmake, https://cmake.org/.

[29] make, https://www.gnu.org/software/make/.

[30] ninja, https://ninja-build.org/.

[31] CLHEP, https://gitlab.cern.ch/CLHEP/CLHEP.

[32] GEANT4 collaboration, *GEANT4 — a simulation toolkit*, *Nucl. Instrum. Meth. A* **506** (2003) 250.

[33] J. Allison et al., *Geant4 developments and applications*, *IEEE Trans. Nucl. Sci.* **53** (2006) 270.

[34] boost, https://www.boost.org/.

[35] mySQL, https://www.mysql.com/.

[36] SQLite, https://www.sqlite.org/index.html.

[37] pyTorch, https://pytorch.org/.

[38] ROOT, https://root.cern.ch/.

[39] conda, https://www.anaconda.com/.

[40] mamba, https://www.anaconda.com/blog/a-faster-conda-for-a-growing-community.

[41] L.W. Piotrowski et al., *On-line and off-line data analysis for the EUSO-TA experiment*, *Nucl. Instrum. Meth. A* **773** (2015) 164.

[42] Pierre Auger collaboration, *Developments and results in the context of the JEM-EUSO program obtained with the ESAF Simulation and Analysis Framework*, in preparation (2023).

[43] A. Cummings, R. Aloisio, J. Eser and J. Krizmanic, *Modeling the optical Cherenkov signals by cosmic ray extensive air showers directly observed from suborbital and orbital altitudes*, *Phys. Rev. D* **104** (2021) 063029 [arXiv:2105.03255].

[44] S. Biktemerova, M. Gonchar and D. Naumov, *Integration of ESAF and Geant4 for simulation of space based telescopes*, in the proceeding of the *31st International Cosmic Ray Conference*, Łodz, Poland, 7–15 July 2009, pp. 427–429.

[45] G. Filippatos et al., *Expected Performance of the EUSO-SPB2 Fluorescence Telescope*, *PoS* **ICRC2021** (2021) 405 [arXiv:2112.07561].

[46] A.M. Hillas, *The sensitivity of Cherenkov radiation pulses to the longitudinal development of cosmic-ray showers*, *J. Phys. G* **8** (1982) 1475.

[47] F. Nerling, J. Bluemer, R. Engel and M. Risse, *Universality of electron distributions in high-energy air showers: Description of Cherenkov light production*, *Astropart. Phys.* **24** (2006) 421 [astro-ph/0506729].

[48] AIRFLY collaboration, *Measurement of the pressure dependence of air fluorescence emission induced by electrons*, *Astropart. Phys.* **28** (2007) 41 [astro-ph/0703132].

[49] M. Nagano, K. Kobayakawa, N. Sakaki and K. Ando, *New measurement on photon yields from air and the application to the energy estimation of primary cosmic rays*, *Astropart. Phys.* **22** (2004) 235 [astro-ph/0406474].

[50] National Oceanic and Atmospheric Administration, Earth System Research Laboratory, Global Monitoring Division, Ozone and Water Vapor Group, https://gml.noaa.gov/dv/iadv/.

[51] S. Falk, *Atmospheric influence on space-based observation of high-energy cosmic rays*, *J. Phys. Conf. Ser.* **632** (2015) 012091.

[52] Telescope Array collaboration, *Telescope array experiment*, *Nucl. Phys. B Proc. Suppl.* **175-176** (2008) 221.

[53] JEM-EUSO collaboration, *First observations of speed of light tracks by a fluorescence detector looking down on the atmosphere*, 2018 *JINST* **13** P05023 [arXiv:1808.02557].

[54] A.M. Hillas, *Cosmic Rays*, Pergamon Press (1972).

[55] A. Paszke et al., *PyTorch: An Imperative Style, High-Performance Deep Learning Library*, in *Advances in Neural Information Processing Systems*, H. Wallach et al., eds., vol. 32, pp. 8026–8037 Curran Associates, Inc. (2019).

[56] *Berlin Declaration on Open Access to Knowledge in the Sciences and Humanities*, 2015, https://openaccess.mpg.de/Berlin-Declaration.

[57] M.D. Wilkinson et al., *The FAIR Guiding Principles for scientific data management and stewardship*, *Sci. Data* **3** (2016) 160018.

The JEM-EUSO collaboration

S. Abe[ff], J.H. Adams Jr.[ld], D. Allard[cb], P. Alldredge[ld], R. Aloisio[ep], L. Anchordoqui[le], A. Anzalone[ed,eh], E. Arnone[ek,el], B. Baret[cb], D. Barghini[el,ek,em], M. Battisti[cb,ek,el], R. Bellotti[ea,eb], A.A. Belov[ib], M. Bertaina[ek,el], P.F. Bertone[lf], M. Bianciotto[ek,el], F. Bisconti[ei], C. Blaksley[fg], S. Blin-Bondil[cb], K. Bolmgren[ja], S. Briz[lb], J. Burton[ld], F. Cafagna[ea], G. Cambié[ei,ej], D. Campana[ef], F. Capel[db], R. Caruso[ec,ed], M. Casolino[ei,ej,fg], C. Cassardo[ek,el], A. Castellina[ek,em], K. Černý[ba], M.J. Christl[lf], R. Colalillo[ef,eg], L. Conti[en,ei], G. Cotto[ek,el], H.J. Crawford[la], R. Cremonini[el], A. Creusot[cb], A. Cummings[lm], A. de Castro Gónzalez[lb], C. de la Taille[ca], R. Diesing[lb], P. Dinaucourt[ca], A. Di Nola[eg], T. Ebisuzaki[fg], J. Eser[lb,*], S. Falk[da], F. Fenu[eo], S. Ferrarese[ek,el], G. Filippatos[lc,*], W.W. Finch[lc], F. Flaminio[eg], C. Fornaro[en,ei], M. Fouka[ac], D. Fuehne[lc], C. Fuglesang[ja], M. Fukushima[fa], D. Gardiol[ek,em], G.K. Garipov[ib], A. Golzio[ek,el], P. Gorodetzky[cb], F. Guarino[ef,eg,*], C. Guépin[cd], A. Haungs[da], T. Heibges[lc], F. Isgrò[ef,eg], E.G. Judd[la], F. Kajino[fb], I. Kaneko[fg], S.-W. Kim[ga], P.A. Klimov[ib], J.F. Krizmanic[lj], V. Kungel[lc], E. Kuznetsov[ld], F. López Martínez[lb], D. Mandát[bb], M. Manfrin[ek,el], A. Marcelli[ej], L. Marcelli[ei], W. Marszał[ha], J.N. Matthews[lg], M. Mese[ef,eg], S.S. Meyer[lb], J. Mimouni[ab], H. Miyamoto[ek,el,ep], Y. Mizumoto[fd], A. Monaco[ea,eb], S. Nagataki[fg], J.M. Nachtman[li], D. Naumov[ia], A. Neronov[cb], T. Nonaka[fa], T. Ogawa[fg], S. Ogio[fa], H. Ohmori[fg], A.V. Olinto[lb], Y. Onel[li], G. Osteria[ef], A. Pagliaro[eh,ed], B. Panico[ef,eg], E. Parizot[cb,cc], I.H. Park[gb], T. Paul[le,*], M. Pech[bb], F. Perfetto[ef], P. Picozza[ei,ej], L.W. Piotrowski[hb], Z. Plebaniak[ei,ej], J. Posligua[li], R. Prevete[ef,eg], G. Prévôt[cb], M. Przybylak[ha], E. Reali[ei,ej], P. Reardon[ld], M.H. Reno[li], M. Ricci[ee], G. Romoli[ei,ej], H. Sagawa[fa], Z. Sahnoune[ac], N. Sakaki[fg], O.A. Saprykin[ic], F. Sarazin[lc], M. Sato[fe], P. Schovánek[bb], V. Scotti[ef,eg], S. Selman[cb], S.A. Sharakin[ib], K. Shinozaki[ha], J.F. Soriano[le], J. Szabelski[ha], N. Tajima[fg], T. Tajima[fg], Y. Takahashi[fe], M. Takeda[fa], Y. Takizawa[fg], S.B. Thomas[lg], L.G. Tkachev[ia], T. Tomida[fc], S. Toscano[ka], M. Traïche[aa], D. Trofimov[cb,ib], K. Tsuno[fg], M. Unger[da], P. Vallania[ek,em], L. Valore[ef,eg], T.M. Venters[lj], C. Vigorito[ek,el], M. Vrabel[ha], S. Wada[fg], J. Watts Jr.[ld], L. Wiencke[lc], D. Winn[lk], H. Wistrand[lc], I.V. Yashin[ib], R. Young[lf], M.Yu. Zotov[ib]

[aa] *Centre for Development of Advanced Technologies (CDTA), Algiers, Algeria*
[ab] *Laboratory of Mathematics and Sub-Atomic Physics (LPMPS), University Constantine I, Constantine, Algeria*
[ac] *Department of Astronomy, Centre of Research in Astronomy, Astrophysics and Geophysics (CRAAG), Algiers, Algeria*
[ba] *Joint Laboratory of Optics, Faculty of Science, Palacký University, Olomouc, Czech Republic*
[bb] *Institute of Physics of the Czech Academy of Sciences, Prague, Czech Republic*
[ca] *Omega, Ecole Polytechnique, CNRS/IN2P3, Palaiseau, France*
[cb] *Université Paris Cité, CNRS, AstroParticule et Cosmologie, F-75013 Paris, France*
[cc] *Institut Universitaire de France (IUF), France*
[cd] *Laboratoire Univers et Particules de Montpellier, Université Montpellier, CNRS/IN2P3, France*
[da] *Karlsruhe Institute of Technology (KIT), Germany*
[db] *Max Planck Institute for Physics, Munich, Germany*
[ea] *Istituto Nazionale di Fisica Nucleare — Sezione di Bari, Italy*
[eb] *Università degli Studi di Bari Aldo Moro, Italy*
[ec] *Dipartimento di Fisica e Astronomia "Ettore Majorana", Università di Catania, Italy*
[ed] *Istituto Nazionale di Fisica Nucleare — Sezione di Catania, Italy*
[ee] *Istituto Nazionale di Fisica Nucleare — Laboratori Nazionali di Frascati, Italy*
[ef] *Istituto Nazionale di Fisica Nucleare — Sezione di Napoli, Italy*
[eg] *Università di Napoli Federico II — Dipartimento di Fisica "Ettore Pancini", Italy*
[eh] *INAF — Istituto di Astrofisica Spaziale e Fisica Cosmica di Palermo, Italy*
[ei] *Istituto Nazionale di Fisica Nucleare — Sezione di Roma Tor Vergata, Italy*

[ej] Università di Roma Tor Vergata — Dipartimento di Fisica, Roma, Italy
[ek] Istituto Nazionale di Fisica Nucleare — Sezione di Torino, Italy
[el] Dipartimento di Fisica, Università di Torino, Italy
[em] Osservatorio Astrofisico di Torino, Istituto Nazionale di Astrofisica, Italy
[en] Uninettuno University, Rome, Italy
[eo] Agenzia Spaziale Italiana, Via del Politecnico, 00133, Roma, Italy
[ep] Gran Sasso Science Institute, L'Aquila, Italy
[fa] Institute for Cosmic Ray Research, University of Tokyo, Kashiwa, Japan
[fb] Konan University, Kobe, Japan
[fc] Shinshu University, Nagano, Japan
[fd] National Astronomical Observatory, Mitaka, Japan
[fe] Hokkaido University, Sapporo, Japan
[ff] Nihon University Chiyoda, Tokyo, Japan
[fg] RIKEN, Wako, Japan
[ga] Korea Astronomy and Space Science Institute, Republic of Korea
[gb] Sungkyunkwan University, Seoul, Republic of Korea
[ha] National Centre for Nuclear Research, Otwock, Poland
[hb] Faculty of Physics, University of Warsaw, Poland
[ia] Joint Institute for Nuclear Research, Dubna, Russia
[ib] Skobeltsyn Institute of Nuclear Physics, Lomonosov Moscow State University, Russia
[ic] Space Regatta Consortium, Korolev, Russia
[ja] KTH Royal Institute of Technology, Stockholm, Sweden
[ka] ISDC Data Centre for Astrophysics, Versoix, Switzerland
[la] Space Science Laboratory, University of California, Berkeley, CA, U.S.A.
[lb] University of Chicago, IL, U.S.A.
[lc] Colorado School of Mines, Golden, CO, U.S.A.
[ld] University of Alabama in Huntsville, Huntsville, AL, U.S.A.
[le] Lehman College, City University of New York (CUNY), NY, U.S.A.
[lf] NASA Marshall Space Flight Center, Huntsville, AL, U.S.A.
[lg] University of Utah, Salt Lake City, UT, U.S.A.
[li] University of Iowa, Iowa City, IA, U.S.A.
[lj] NASA Goddard Space Flight Center, Greenbelt, MD, U.S.A.
[lk] Fairfield University, Fairfield, CT, U.S.A.
[ll] Department of Physics and Astronomy, University of California, Irvine, U.S.A.
[lm] Pennsylvania State University, PA, U.S.A.

[*] Corresponding author.