Simplifying Space Physics Data Analysis with SciQLop: A Comprehensive Toolkit for In-Situ Measurements

Streamline Your Data Discovery, Retrieval, and Analysis Workflow

Alexis Jeandet[1], Nicolas Aunai[1], Vincent Génot[2], Patrick Boettcher[5], Benjamin Renard[3], Bayane Michotte de Welle[1], Nicolas André[2], Myriam Bouchemit[2], and Nicolas Dufourg[4]

[1]Laboratory Of Plasma Physics, CNRS, Palaiseau CEDEX, France [2]Institut de Recherche en Astrophysique et Planétologie, CNRS, CNES, UPS, Toulouse, France [3]AKKA, Toulouse, France
[4]CNES, Toulouse, France [5]YAISE, Villeconin, France

With the **SCI**entific **Q**t application for **L**earning from **O**bservations of **P**lasmas (**SciQLop**), analyzing space physics data is made easier. The project aims to solve the technical challenges involved in retrieving and interpreting data from remote servers, which can be daunting for students or newcomers. Even analyzing data from a single instrument on a given mission can raise some technical difficulties such as finding where to get them, how to get them and sometimes how to read them. These challenges can compound when building complex machine learning pipelines involving multiple instruments and even multiple spacecraft missions. The SciQLop project removes these technical difficulties while maintaining high performance, allowing scientists to focus on their data analysis.

Speasy : Making Space Physics Data Access Easy

Speasy is an open-source Python package that streamlines the discovery and retrieval of space physics data from remote servers. By providing a user-friendly API, Speasy removes the technical complexities involved in finding and downloading data from sources such as **CDAWeb**, **SSCWeb**, **CSA**, and **AMDA**, making it easier for researchers and students to focus on data analysis.

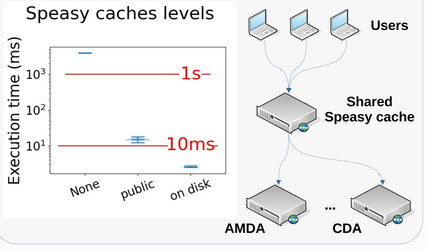
Key features:

- Really simple and user friendly API:

  With just one function, **get_data(what, [when])**, you can retrieve any combination of data products and time ranges. This makes it easy to access the specific data you need for your analysis, without having to navigate complex server structures or learn multiple APIs.

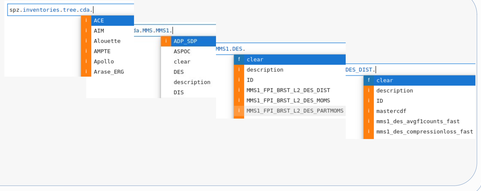
- Efficient multi-layer caching mechanism

  With two levels of caching - one on disk and one on a shared server that runs Speasy - the package can quickly access previously requested data without having to make slow requests to remote servers like AMDA or CDAWeb. This not only saves time but also reduces the load on the remote servers.



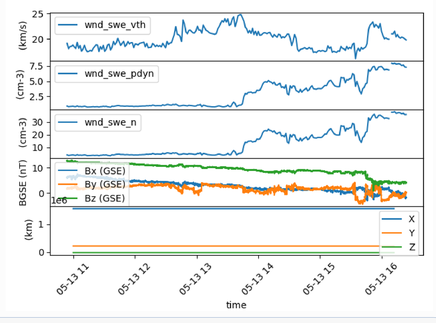
- Dynamic remote servers inventory:

  Speasy builds a dynamic inventory of accessible remote servers, providing users with a comprehensive list of available products. With runtime Python completion, users can easily navigate and discover products on remote servers without the need for extensive documentation or complex queries. This simplifies the discovery process and reduces the time required to find the relevant data, making it an efficient and user-friendly feature.



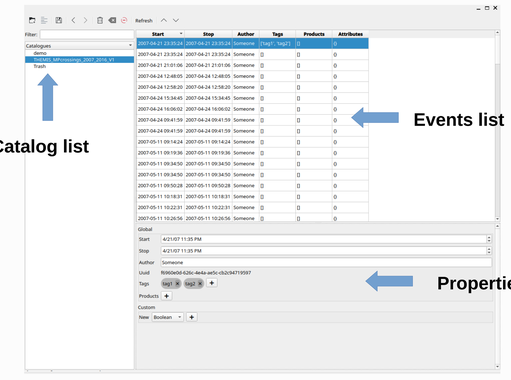

```
[8]: products = [
         spz.inventories.tree.amda.Parameters.Wind.SWE.wnd_swe_kp.wnd_swe_vth,
         spz.inventories.tree.amda.Parameters.Wind.SWE.wnd_swe_kp.wnd_swe_pdyn,
         spz.inventories.tree.amda.Parameters.Wind.SWE.wnd_swe_kp.wnd_swe_n,
         spz.inventories.tree.cda.Wind.WIND.MFI.WI_H2_MFI.BGSE,
         spz.inventories.tree.ssc.Trajectories.wind,
     ]

     data_several_dates: List[List[SpeasyVariable]] = spz.get_data(
         products,
         spz.inventories.tree.amda.TimeTables.SharedTimeTables.SOLAR_WIND.Magnetic_Clouds
     )

     for i in range(5):
         #fig = plt.figure(figsize=(20, 6))
         fig = plt.figure()
         gs = fig.add_gridspec(5, hspace=0)
         axes = gs.subplots(sharex=True, sharey=False)
         for j in range(5):
             data_several_dates[j][i].plot(ax=axes[j])
         plt.tight_layout()
     plt.show()
```

Figure 12

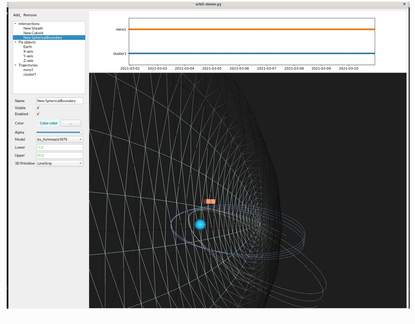


Efficient Event Catalog Management with TSCAT and TSCAT-GUI

Tscat and Tscat-gui are two Python packages that are currently in development, aimed at facilitating event management. Tscat serves as the backend for computing and organizing massive catalogs of events, which can be static lists or filtered results based on user-defined criteria. Tscat-gui provides a graphical interface for visualizing and editing these catalogs and events.



Broni & Orbit Viewer: Efficiently Generating Time Ranges from Spacecraft Trajectories and Physical Models Intersections



Broni and Orbit Viewer are a set of tools that are currently in development for space physics research. Broni serves as the backend for computing trajectory intersections based on spacecraft data and physical models, while Orbit Viewer provides a 3D graphical user interface for visualizing the resulting trajectories. These tools aim to simplify the process of analyzing spacecraft data by providing a streamlined approach to building time ranges from spacecraft trajectories and physical model intersections.

Sciqlop Explorer: A Fast and User-Friendly GUI for Space Physics Data Analysis

The SciQLop GUI app is a fast and extensible data analysis tool written in Python using PySide. Built with the aim of simplifying the analysis of in-situ space physics measurements, this app utilizes Speasy to access data from remote servers. In addition to its speed and simplicity, the app is also designed to be highly extensible, with an embedded IPython console for easy integration with other Python tools and libraries.

Key features:

- Simple and user-friendly interface:

  The user-friendly GUI of sciqlop is designed to be both fast and easy to use, making it accessible to scientists with varying levels of technical proficiency. It employs drag and drop extensively for user interactions, and ensures that plots can be zoomed in/out and panned quickly and smoothly.

- "Virtual products":

  With Sciqlop, scientists can quickly create their own virtual products by combining various data sources and algorithms using just a few lines of Python code. These virtual products can then be easily visualized and analyzed within the app.

Planed features:

- Catalogs:
  - Efficient event catalog browsing and editing: SciQLop will provide users with a fast and intuitive way to browse and edit event catalogs, allowing them to easily navigate through large amounts of data and make modifications. This will enable large-scale statistical analysis and other data-driven research.
  - Collaboration features: SciQLop will include features for sharing and tracking modifications to catalogs, making it easier for teams to collaborate and produce high-quality catalogs.

- GUI templates:

  Sciqlop will offer GUI templates that allow users to create and share GUI state presets. This feature enables scientists and students to quickly reproduce the desired plot layout and setups, promoting reproducibility and efficiency in their research.