Providing Tools for the Analysis of Solar Energetic Particles as Jupyter Notebooks – Experiences from the SERPENTINE Project

J. Gieseler, C. Palmroos, N. Dresing, A. Kouloumvakos, D. Morosan, D. J. Price, D. Trotta, A. Yli-Laurila, and R. Vainio

Data, Analysis, and Software in Heliophysics (DASH)
10 Oct 2023

UNIVERSITY OF TURKU

This work has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 101004159

SERPENTINE project

“The Solar EneRgetic ParticlE aNalysis plaTform for the INner hEliosphere (SERPENTINE) project will answer several outstanding questions about the origin of Solar Energetic Particle (SEP) events and provides an advanced platform for the analysis and visualization of high-level datasets to benefit the wider heliophysics community.”

- 3-year EU project bringing together multiple universities across Europe
- Will provide catalogs and other high level datasets at data.serpentine-h2020.eu
- Provides tools aimed at scientific users without much programming experience
  - Jupyter Notebooks
  - JupyterHub server at serpentine-h2020.eu/hub
  - Streamlit web-app’s like solar-mach.github.io

Provide the necessary options:

```
In [2]: body_list = ['STEREO-A', 'Earth', 'BepiColombo', 'PSP', 'Solar Orbiter']
        vsw_list = [380, 290, 300, 340, 362]  # values from Lario et al. 2022, doi:10.3847/1538-4357/ac6efd
        date = '2021-10-9 6:30:00'
```

The default coordinate system is Carrington coordinates, alternatively one could select the Earth-centered Stonyhurst coordinate system:

```
In [3]: coord_sys = 'Stonyhurst'      # 'Carrington' (default) or 'Stonyhurst'
```

Now we also want to indicate the position and direction of a flare, and the (assumed) solar wind speed at its location:

```
In [4]: reference_long = 351          # Carrington longitude of reference (None to omit)
        reference_lat = 0             # Carrington latitude of reference (None to omit)
        reference_vsw = 340           # define solar wind speed at reference
```

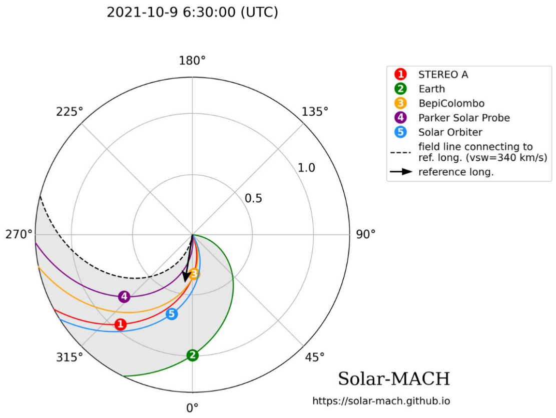
In addition, we explicitly provide all availabe plotting options:

```
In [5]: plot_spirals = True           # plot Parker spirals for each body
        plot_sun_body_line = False    # plot straight line between Sun and body
        long_offset = 270             # longitudinal offset for polar plot; defines where Earth's longitude is (by default 27
        numbered_markers = True       # plot each body with a numbered marker
```

Finally, initializing and plotting with these options.

```
In [6]: sm = SolarMACH(date, body_list, vsw_list, reference_long, reference_lat, coord_sys)

        sm.plot(plot_spirals=plot_spirals, plot_sun_body_line=plot_sun_body_line, long_offset=long_offset,
                reference_vsw=reference_vsw, numbered_markers=numbered_markers,
                long_sector=[351, 85], long_sector_vsw=[340, 290], long_sector_color='lightgrey')
```



All the data can also be obtained as a Pandas DataFrame for further use:

```
In [7]: sm.coord_table
```

Out[7]:

	Spacecraft/Body	Stonyhurst longitude (°)	Stonyhurst latitude (°)	Heliocentric distance (AU)	Longitudinal separation to Earth's longitude	Latitudinal separation to Earth's latitude	Vsw	Magnetic footpoint longitude (Stonyhurst)	Longitudinal separation between body and reference_long	Longitudinal separation between body's mangetic footpoint and reference_long	Latitudinal separation between body and reference_lat
0	STEREO-A	-38.917004	7.276757	0.957026	-38.917005	1.009886	380	23.300249	-389.917004	32.300249	7.276757
1	Earth	0.000001	6.266871	0.998914	0.000000	0.000000	290	85.360062	-350.999999	94.360062	6.266871
2	BepiColombo	2.551346	2.176375	0.330621	2.551344	-4.090496	300	29.803439	-348.448654	38.803439	2.176375
3	PSP	-47.855146	3.694387	0.765364	-47.855148	-2.572484	340	8.154330	-398.855146	17.154330	3.694387
4	Solar Orbiter	-14.772829	2.339315	0.678619	-14.772830	-3.927556	362	31.915162	-365.772829	40.915162	2.339315

Jupyter Notebooks

- Collection of easy-to-use Notebooks for (but not limited to) the analysis of solar energetic particles
- Providing GUI’s *(ipywidgets)* and example codes
- Based on own PyPI package [*seppy*]() for functions (and existing *solarmach*, *solo-epd-loader*)
- Available with instructions on [github.com/serpentine-h2020/serpentine]()

First import the necessary library

In [1]:
```
from seppy.tools import Event
import seppy.tools.widgets as w
import datetime, os
```

Choose the spacecraft, sensor, view direction and particle species:

In [2]:
```
display(w.spacecraft_drop, w.sensor_drop, w.view_drop, w.species_drop)
```

Spacecraft: Solar Orbiter

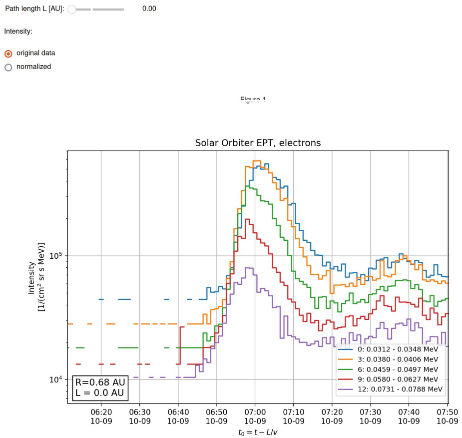
Sensor: EPT

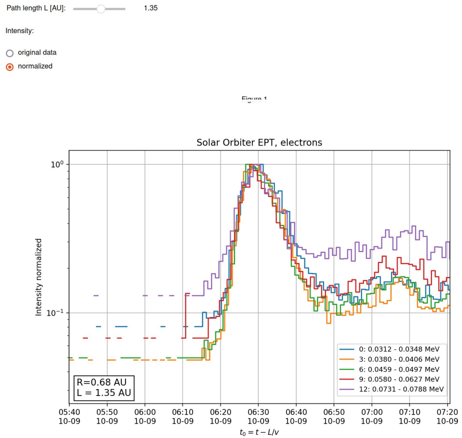
Viewing: sun

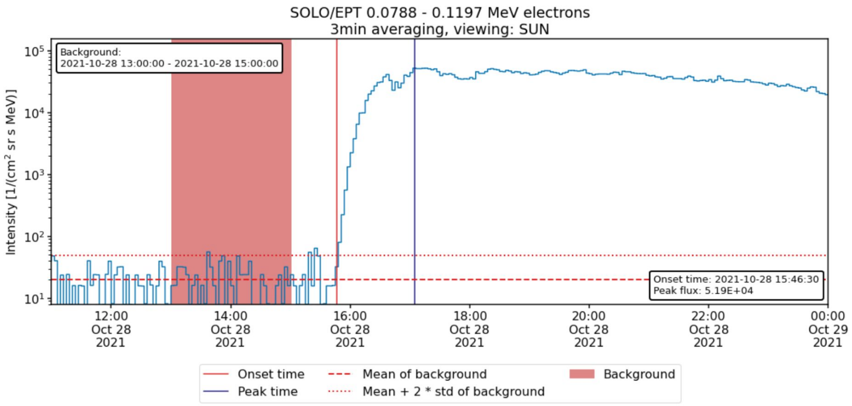
Species: electrons

Set the path to your data folder:

In [3]:
```
# Path for the downloaded data (by default the current directory)
data_path = f"{os.getcwd()}/data/"
```







Jupyter Notebooks

- Collection of easy-to-use Notebooks for (but not limited to) the analysis of solar energetic particles
- Providing GUI's *(ipywidgets)* and example codes
- Based on own PyPI package *seppy* for functions (and existing *solarmach*, *solo-epd-loader*)
- Available with instructions on github.com/serpentine-h2020/serpentine

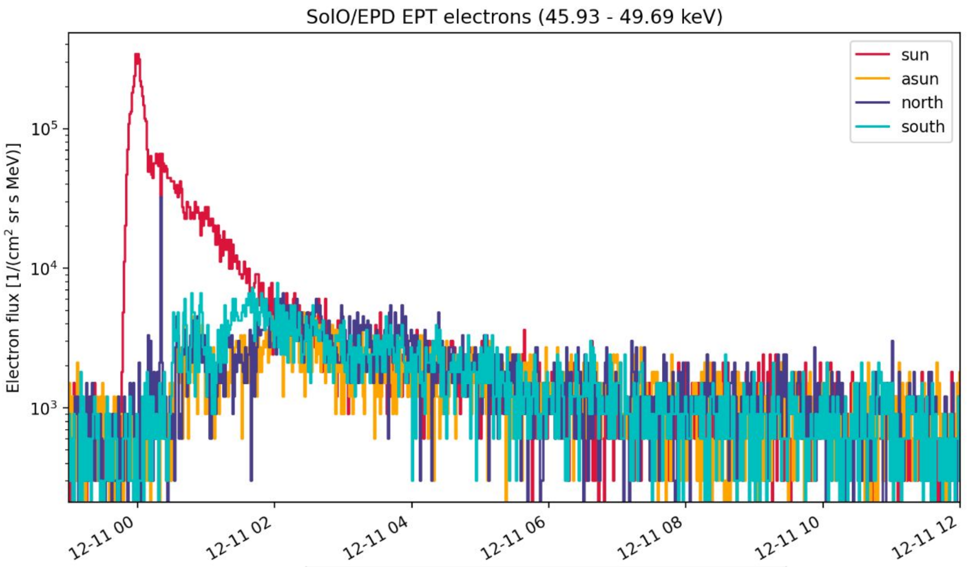
```
# energy channel to use; cf. "energies" for the energies
channel = 6

fig, ax = plt.subplots(figsize=(10, 6), dpi=200)
ax = df_electrons_sun['Electron_Flux'][f'Electron_Flux_{channel}'].plot(logy=True, label='sun',
                                                                          color=color['sun'],
                                                                          drawstyle="steps-mid")
ax = df_electrons_asun['Electron_Flux'][f'Electron_Flux_{channel}'].plot(logy=True, label='asun',
                                                                          color=color['asun'],
                                                                          drawstyle="steps-mid")
ax = df_electrons_north['Electron_Flux'][f'Electron_Flux_{channel}'].plot(logy=True, label='north',
                                                                          color=color['north'],
                                                                          drawstyle="steps-mid")
ax = df_electrons_south['Electron_Flux'][f'Electron_Flux_{channel}'].plot(logy=True, label='south',
                                                                          color=color['south'],
                                                                          drawstyle="steps-mid")

ax.set_xlim([dt.datetime(2020, 12, 10, 23, 0), dt.datetime(2020, 12, 11, 12, 0)])

ax.set_ylabel(r"Electron flux [1/(cm$^2$ sr s MeV)]")
ax.set_title(f"SolO/EPD EPT electrons ({1000*energies['Electron_Bins_Low_Energy'][channel]:.2f}"
             + f" - {1000*energies['Electron_Bins_Low_Energy'][channel+1]:.2f} keV)")
ax.legend()
```

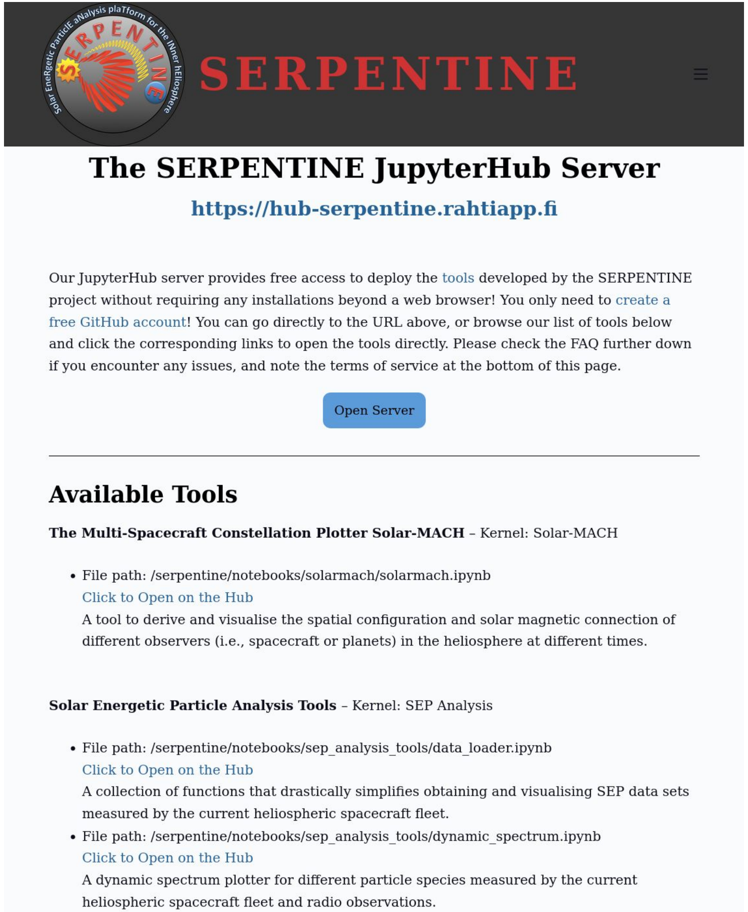
```
<matplotlib.legend.Legend at 0x7f4bf8fd4820>
```



JupyterHub server

All Notebooks also available on SERPENTINE’s own JupyterHub server

- Hosted at CSC, Finland (non-profit state enterprise)
- Login with GitHub account
- Providing pre-configured conda environments
- Clones all Notebooks from the Notebook GitHub repo
- Backup of Notebooks changed by user



SERPENTINE

The SERPENTINE JupyterHub Server

https://hub-serpentine.rahtiapp.fi

Our JupyterHub server provides free access to deploy the tools developed by the SERPENTINE project without requiring any installations beyond a web browser! You only need to create a free GitHub account! You can go directly to the URL above, or browse our list of tools below and click the corresponding links to open the tools directly. Please check the FAQ further down if you encounter any issues, and note the terms of service at the bottom of this page.

Open Server

Available Tools

The Multi-Spacecraft Constellation Plotter Solar-MACH – Kernel: Solar-MACH

- File path: /serpentine/notebooks/solarmach/solarmach.ipynb
  Click to Open on the Hub
  A tool to derive and visualise the spatial configuration and solar magnetic connection of different observers (i.e., spacecraft or planets) in the heliosphere at different times.

Solar Energetic Particle Analysis Tools – Kernel: SEP Analysis

- File path: /serpentine/notebooks/sep_analysis_tools/data_loader.ipynb
  Click to Open on the Hub
  A collection of functions that drastically simplifies obtaining and visualising SEP data sets measured by the current heliospheric spacecraft fleet.
- File path: /serpentine/notebooks/sep_analysis_tools/dynamic_spectrum.ipynb
  Click to Open on the Hub
  A dynamic spectrum plotter for different particle species measured by the current heliospheric spacecraft fleet and radio observations.

Streamlit

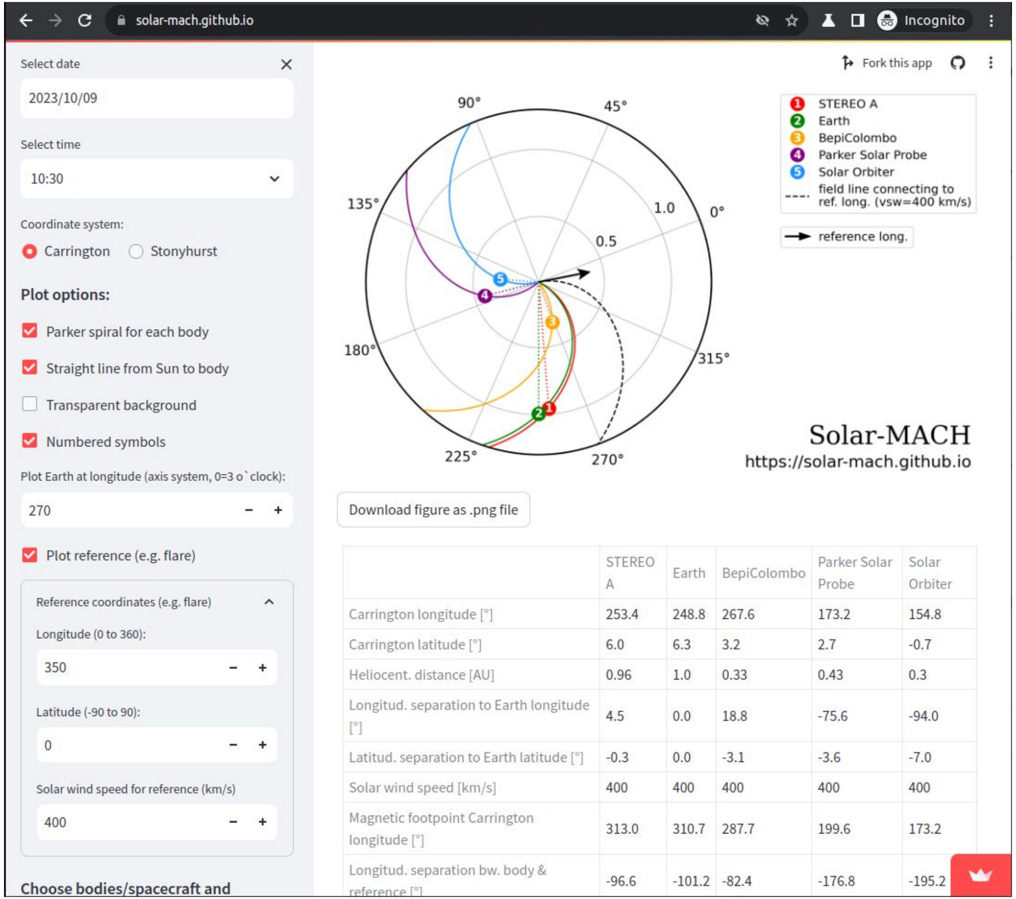
Alternative to Notebook: *streamlit* web-apps

- Open-source Python package, similar to *plotly*'s *dash*
- Provides Python tool as a web-page
- Run locally, host on own server or in the cloud ($ for more advanced hosting), with the latter being really hassle-free
- No coding required for the user at all

Examples

```
import datetime
import streamlit as st

d = st.date_input("When's your birthday", datetime.date(2019, 7, 6))
st.write('Your birthday is:', d)
```



Problems & Lessons learned

General:
- Users really like the easiest solution for them *(surprise!)*
- How to share your (Notebook) tools in general?
- How to collaboratively work on them?

Jupyter Notebooks:
- Updates to main Notebook difficult to integrate into version with user's content
- Taking care of requirements on user's computers

JupyterHub:
- Easy for new users
- Can be difficult/expensive to maintain

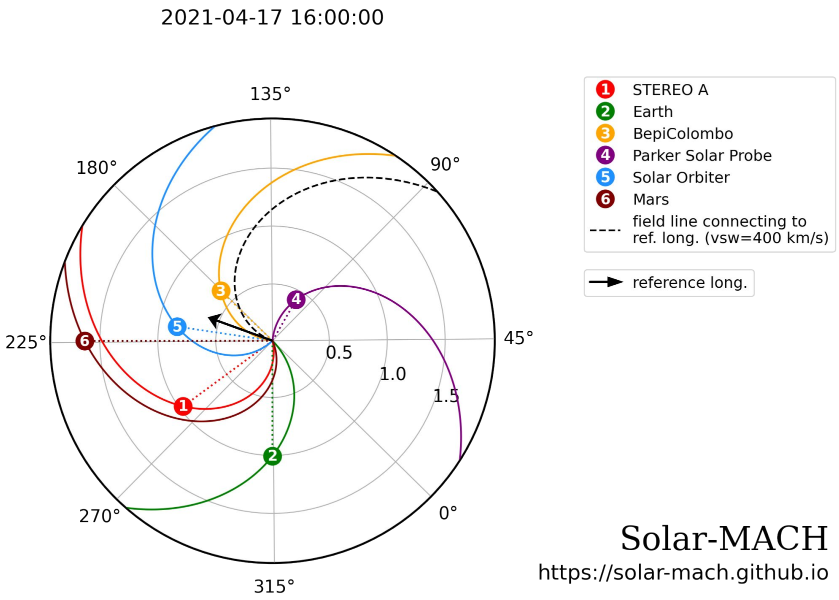
Streamlit:
- Very easy for new users, also easy for developers
- Will become more complicated with more options (like all GUI's)
- Limitations for developers and users (e.g., saving, editing things)
  ⇒ Useful for tools with limited scope in functions, but wide audience

Publications & Links

Gieseler et al. (2023). Solar-MACH: An open- source tool to analyze solar magnetic connection configurations. *Front. Astronomy Space Phys.* [doi:10.3389/fspas.2022.1058810](https://doi.org/10.3389/fspas.2022.1058810)

Palmroos et al. (2022). Solar energetic particle time series analysis with Python. *Front. Astronomy Space Phys.* [doi:10.3389/fspas.2022.1073578](https://doi.org/10.3389/fspas.2022.1073578)

- https://serpentine-h2020.eu
- https://github.com/serpentine-h2020/serpentine
- https://github.com/jgieseler/solarmach
- https://solar-mach.github.io
- jan.gieseler@utu.fi
- https://twitter.com/JanGieseler
- https://fediscience.org/@JanGieseler
- https://bsky.app/profile/jangieseler.bsky.social



Solar-MACH
https://solar-mach.github.io

UNIVERSITY OF TURKU