SoloLab V0.3

A Tool for Multi-Instrument Analysis of Solar Orbiter Data

David L. Paipa-Leon, Ph.D.

Installation guide - SoloLab V0.3

Python Tool for multi-instrument studies with Solar Orbiter
Created by David Paipa

February 11, 2026

Contents

1 Introduction

This tutorial will help you in the installation the requirements for using soloLab, a tool to process and visualize multi-instrument data from Solar Orbiter: A spacecraft launched in February 2020 with 10 instruments onboard dedicated to study the sun from up close (at distances down to 0.27 AU or 60 $R_\odot$). Among these instruments, the three instruments we are concerned about in the early development of this tool are:

- **STIX**: the **S**pectrometer **T**elescope for **I**maging **X**-rays.
- **RPW**: the **R**adio and **P**lasma **W**aves instrument.
- **EPD**: the **E**nergetic **P**articles **D**etector.

Since the spacecraft was launched *recently*, the scientific teams of each instrument are still figuring out the behavior of the measurements they take therefore, specialized libraries for processing data from the Solar Orbiter instruments are still under development (or revision).

Traditionally, data analysis in heliophysics was done using a custom-made wrapper library called SolarSoftWare (SSW) written in the programming language IDL. Unfortunately, IDL is not free to use and lacks support for many modern functionalities such as machine-learning libraries. There is growing community support in porting existing IDL code to Python. SunPy is currently one of the most active efforts in this direction.

The utility of a pseudo-automatic visualization tool in python comes from the need of measuring the timing properties of different emissions of solar flares, which can provide information on the physical processes of particle acceleration in flares and their properties. Moreover, comparing time series from multiple instruments can lead to interesting results in other areas of heliophysics, not only measuring the properties of solar transients, but also improving our understanding of the quiet Sun and the heliosphere.

SoloLab V0.3 uses python libraries and custom made scripts to treat data obtained by Solar Orbiter, helping in the following tasks:

- **Data extraction:** STIX FITS files and RPW CDF files.
- **Data processing:** Background subtraction, energy shifts, and frequency filtering.
- **Data visualization:** Multi-instrument spectrograms and time profiles.
- **Estimations and fits:** Frequency Drift Rate Analysis (FDRA) and electron abundance estimations.

Considerations and contact

Created by David Paipa [contact]

This code is **not** an official Solar Orbiter software and is still a work in progress initially developed as a tool for my thesis.

2 System Requirements

Before installation ensure:

- Python 3.10 or 3.11
- Internet connection
- Approximately 2 GB free disk space

It is recommended to install the software inside an isolated virtual environment.

3 Environment Setup

Two installation methods are supported:

- Conda (recommended)
- Native Python using venv

Only one method should be used.

3.1 Method 1 — Conda (Recommended)

Conda simplifies dependency management, particularly for scientific libraries.

3.1.1 Install Miniconda or Anaconda

Download from:
https://www.anaconda.com/download
Follow your operating system instructions.

3.1.2 Create the Environment

```
conda create -n sololab_env python=3.10
conda activate sololab_env
```

Install pip inside the environment:

```
conda install pip
```

3.2 Method 2 — Native Python (venv)

Ensure Python 3.10+ is installed.

3.2.1 Create Virtual Environment

```
python -m venv sololab_env
```

Activate:
Windows:

```
sololab_env\Scripts\activate
```

Linux / macOS:

```
source sololab_env/bin/activate
```

Upgrade pip:

```
pip install --upgrade pip
```

4 Installing SoloLab Dependencies

Place the provided *requirements.txt* file in your working directory.

With the environment activated:

```
pip install -r requirements.txt
```

Verify installed packages:

```
pip list
```

5 CDF Library and pyCDF

The module `spacepy.pycdf` requires the NASA CDF C library.

Verify spacepy:

```
python -c "import spacepy"
```

If `pycdf` fails:

- Download CDF from NASA CDF
- Follow platform installation instructions
- Ensure environment variables are properly defined

Verify:

```
python -c "from spacepy import pycdf"
```

Troubleshooting guide:
pyCDF troubleshooting

6 Jupyter Notebooks

If not included in your requirements file:

```
pip install jupyterlab ipykernel
```

Register the environment:

```
python -m ipykernel install --user --name sololab_env
```

Launch:

```
jupyter lab
```

If plots do not display:

```
%matplotlib inline
```

7 Verification Test

Run:

```
import numpy
import sunpy
import spacepy
import cdflib
print("SoloLab environment correctly installed.")
```

If no errors appear, the installation is complete.

8 Troubleshooting

8.1 Jupyter Does Not Open

Ensure the environment is activated before launching.

8.2 pyCDF Import Errors

Check that:

- CDF C library is installed
- Environment variables are set correctly
- spacepy is installed inside the active environment

8.3 Kernel Issues

Ensure notebook kernel is:

```
Python (sololab_env)
```