A Python package to find a magnetic cloud frame of reference to heliospheric observers using the Minimum Variance approach

A.M. Gulisano[1,2,3], A. Arja[4], R. Pafundi[5,6] & V. Bazzano[7]

[1] *Instituto Antártico Argentino, Dirección Nacional del Antártico, Argentina*
[2] *Instituto de Astronomía y Física del Espacio, CONICET–UBA, Argentina*
[3] *Departamento de Física, FCEN–UBA, Argentina*
[4] *Facultad de Matemática, Astronomía, Física y Computación, UNC, Argentina*
[5] *Instituto de Altos Estudios Espaciales Mario Gulich, CONAE–UNC, Argentina*
[6] *Facultad Regional Córdoba, UTN, Argentina*
[7] *Departamento de Ciencias de la Atmósfera y los Océanos, FCEN–UBA, Argentina*

Received: 09 February 2024 / Accepted: 27 May 2024

Resumen / Describimos el paquete de Python que desarrollamos y que está disponible públicamente para encontrar la orientación de estructuras interplanetarias con características específicas de configuración magnética, denominadas nubes magnéticas(MCs), que permite rotarlas en su marco de referencia local. Adaptamos nuestra previa implementación tipo línea de funciones en Matlab al paradigma Python de programación orientado a objetos. Nuestro fin es proporcionar un paquete fácil de instalar y ejecutar, con un repositorio de código abierto, que brinda estándares de calidad para llegar a una comunidad más amplia de astrofísicos y astrónomos interesados en la heliofísica y la relación Sol-Tierra. Teniendo en cuenta que una nube magnética tiene su propia identidad, estado o atributos y comportamiento (relaciones y métodos), el paradigma de Python es adecuado. Dado que no había librerías o paquetes para encontrar la orientación del eje de una MC implementados en Python y ofrecidos gratuitamente, consideramos nuestro proyecto como una contribución valiosa a la comunidad de heliofísica. En consecuencia, hemos elegido una licencia de Berkeley Software Distribution para su uso, en este artículo se provee información referida a los requerimientos y un breve tutorial para su instalación y uso.

Abstract / We describe the package we have developed that is publicly available to find the orientation of interplanetary structures, called magnetic clouds (MCs) due to specific characteristics of their magnetic configuration, that allows to rotate them to their local frame. We changed the function pipe-line structure of our Matlab previous implementation to the object-oriented programming Python paradigm to provide a package easy to install and run, with an open source repository. Our aim is to provide an easy to install and execute package with high quality standards to reach a wider community of astrophysicists and astronomers interested in heliophysics and Sun-Earth relationship. Taking into account that a magnetic cloud has its own identity, state or attributes, and behavior (relationships and methods), the Python paradigm is appropriate. Since there were no packages to find the MC axis orientation implemented in Python and freely offered, we regard our project as a valuable contribution to the heliophysics community. Accordingly, we have chosen a Berkeley Software Distribution license for its use, in this article information on the requirements and a brief tutorial for installation and usage are provided.

Keywords / solar wind — Sun: heliosphere — methods: numerical

1. Introducción

Magnetic clouds (MCs) are highly magnetized plasma structures characterized by low proton temperature and plasma β (Burlaga, 1991). When observed from a heliospheric perspective, their magnetic field vector exhibits a rotational variation. These structures are recognized as the most impactful features within the interplanetary medium. Despite extensive research on MCs (see Rodriguez et al., 2016, and references therein), consensus regarding their precise magnetic configuration remains elusive. This uncertainty primarily arises from the limitations of spacecraft-obtained magnetic field data, which provide only one-dimensional data along their trajectory (Gulisano et al., 2010a,b). Consequently, inferring the three-dimensional structure of the cloud needs certain assumptions. Traditionally, MCs have been treated as locally symmetric cylindrical structures. The Minimum Variance method (MV) has been widely utilized for determining the orientation of interplanetary structures. By analyzing the temporal series of magnetic field observations, MV can effectively estimate the axis orientation of the cloud, particularly when the spacecraft's trajectory is close to the cloud axis (Lepping & Wu, 2010; Gulisano et al., 2006). MV offers two significant advantages over more complex techniques used for orientation determination: it is relatively straightforward to implement, and it imposes minimal assump-

✉ *Contact* / adrianagulisano@gmail.com

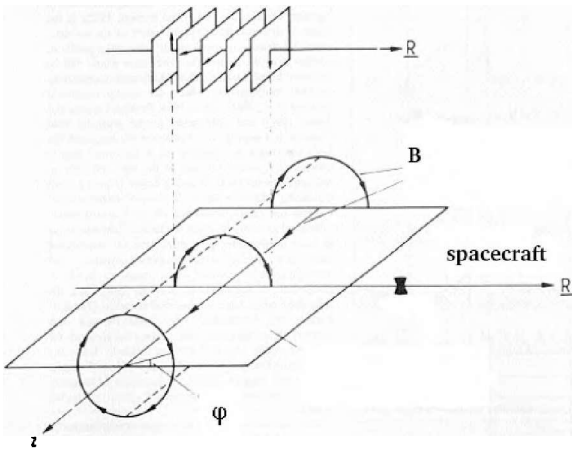

Fig. 1. Idealized scheme of the projection of the magnetic field lines that would be observed during the passage of a magnetic cloud in the plane defined by the spacecraft trajectory and the axis of the cloud adapted from(Bothmer & Schwenn, 1998)

tions on the magnetic configuration, relying solely on local cylindrical symmetry (thus maintaining model independence). This technique provides a good orientation as long as the impact parameter is low, otherwise large errors are present (Gulisano et al., 2007). The importance of a correct orientation can provide a way to estimate global properties in a model-independent fashion (Dasso et al., 2005). Another issue to take into account is the proper determination of the MC borders. In the following section, the reader can find a description of the project and a brief tutorial for requirements, installation, and usage, including the copyright license terms.

1.1. The MC frame

The local axis direction of the MC defines $z_{\rm cloud}$ (with $B_{z,\rm cloud} > 0$). Since the speed of the cloud is mainly in the Sun-Earth direction and is much higher than the spacecraft's speed, which can be assumed at rest during the time the cloud is observed, we consider a rectilinear spacecraft trajectory in the cloud frame. The trajectory defines a direction t, so we take $y_{\rm cloud}$ in the direction $z_{\rm cloud} \times t$ and $x_{\rm cloud}$ completes the right-handed orthogonal base $(x_{\rm cloud}, y_{\rm cloud}, z_{\rm cloud})$. So, $B_{x,\rm cloud}$, $B_{y,\rm cloud}$, and $B_{z,\rm cloud}$ are the components of $\vec{B}$ in this new frame. In this frame the impact parameter, p (that is, the minimum distance from the spacecraft's trajectory to the cloud axis), is small compared to the MC radius (R) or negligible. An MC can be described using a cylindrical magnetic configuration, $\vec{B}(r) = B_z(r)z + B_\phi(r)\phi$, we have $x_{\rm cloud} = r$ and $y_{\rm cloud} = \phi$ after the spacecraft has crossed the MC axis. So for a cylindrical flux rope, the magnetic field data obtained by the spacecraft will show $B_{x,\rm cloud} = 0$, a large and coherent variation of $B_{y,\rm cloud}$ (with a change of sign), and an intermediate and coherent variation of $B_{z,\rm cloud}$, from low values at one cloud edge, reaching the highest value at its axis and returning to low values at the other edge ($B_{z,\rm cloud} = 0$ is typically taken as the MC boundary). The MV technique provides the minimum, maximum, and intermediate variance directions, being useful to find $B_{x,\rm cloud}$, $B_{y,\rm cloud}$ and $B_{z,\rm cloud}$, respectively, as shown in Fig. 1 where an idealized scheme of the projection of the magnetic field lines that would be observed during the passage of a magnetic cloud in the plane defined by the spacecraft trajectory and the axis of the cloud is depicted.

2. The project

This section outlines the project, to create a publicly accessible package for determining the orientation of an MC and rotating it to its local frame. To enhance accessibility and usability, we transitioned from the functional pipeline structure of our previous Matlab implementation to the object-oriented programming (OOP) paradigm in Python, (Roth, 2022). This change allows for easy installation and execution of the package, with an open-source repository adhering to quality standards, thereby catering to a broader community of astrophysicists and astronomers interested in heliophysics and the Sun-Earth relationship. Recognizing that an MC possesses distinct characteristics, states, and behaviors, the Python paradigm was deemed suitable due to its characteristics for defining attributes, states, relationships, and methods inherent to the object-oriented approach. We crafted a recognizable logo for the project (not shown).

Given the absence of free packages for determining an MC axis orientation in Python, our project stands as a significant contribution to the heliophysics community. Consequently, we opted for the Berkeley Software Distribution (BSD) License, facilitating the open use and distribution of the project, the reader can find the license information in the GitHub repository:
https://github.com/adelarja/space_weather.git

2.1. Requirements

Users need a Python 3.9+ environment to run solarwindpy. These are the required dependencies: NumPy, Pandas, SciPy, matplotlib, typer, heliopy==0.15.4, requests, sunpy, h5netcdf, cdflib.

Test:
To run the swindpy tests, users have to clone the repository and use the pytest module.
$ pytest tests
Users are also able to run a suite of checks with tox:
$ pip install tox
$ tox

2.2. Installation

The package is available in pypi. Users can install it using pip the instruction is: $ pip install swindpy

2.3. Developers

Developers willing to contribute, change, or improve this package, should clone the repository and install the project with the following commands: $ git clone https://github.com/adelarja/space_weather.git
$ cd space−weather
$ pip install −e .

3. Tutorial

Users can find a readme.md in our repository and download the Python codes, which follow the PEP8 standard, and have a high test coverage of over 90 percent ensuring the quality of our implementation.
In this tutorial we will show how to rotate a magnetic cloud using the swindpy abstractions.
First of all, we need to import the libraries we are going to use and, set the date times we are interested in:

```
import numpy as np
from datetime import datetime
import matplotlib.pyplot as plt
import swindpy.plotter as plotter
from swindpy.data_manager import Period
from swindpy.data_manager import MagneticField
from swindpy.data_manager import DataManager
from swindpy.rotation import RotatedWind
```

For this example, we are going to use these dates:
10-Jan-1997 05:00 to 11-Jan-1997 02:00

We set the datetimes we are interested in:
date_from = datetime(1997, 1, 10, 5, 0, 0)
date_to = datetime(1997, 1, 11, 2, 0, 0)

We create the Period object
period=Period(date_from, date_to)

Using the DataManager, we retrieve the cdf information
cdf_data=DataManager.get_gse_magnetic_vector(period)

The obtained data constitute a list of MagneticField objects
(a swindpy abstraction), that has information about the datetime and the gse coordinates.
cdf_data[0]
MagneticField(time=Timestamp('1997-01-10 05:00:30'), bgse0=-1.8067849,bgse1=9.860464,bgse2=-8.717464)

Now, using the DataManager, we can filter Not A Number values and infinite values.
filtered_data=DataManager.filter_nan_and_inf_values (cdf_data)

Now we can obtain the Rotated field simply by calling a classmethod
rotated_wind = RotatedWind.get_rotated_wind(filtered_data)
Using the plotter method, we can plot non rotated magnetic field

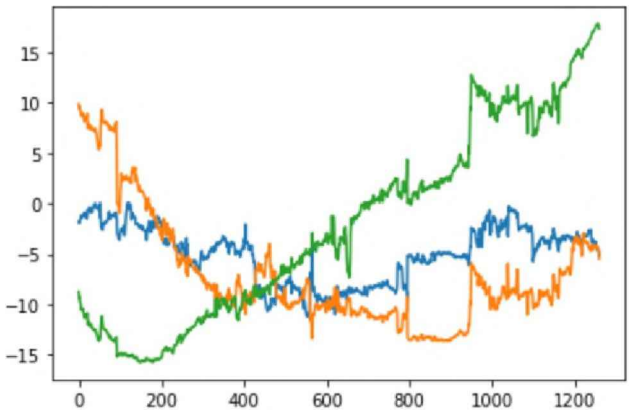

Fig. 2. Non-rotated data, magnetic field components [nT] in GSE coordinates in the vertical axis ($B_{z,\mathrm{GSE}}$ in green, $B_{y,\mathrm{GSE}}$ in orange and $B_{x,\mathrm{GSE}}$ in blue), time [minutes] from start in the horizontal axis.

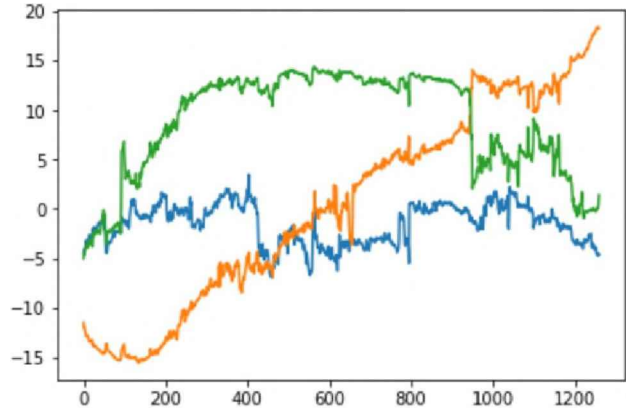

Fig. 3. Same as Fig. 2, but for the rotated data in the MC frame of reference, $B_{z,\mathrm{cloud}}$ in green, $B_{y,\mathrm{cloud}}$ in orange and $B_{x,\mathrm{cloud}}$ in blue.

and rotated magnetic field (we are also able to add labels, change size, etc).
Plotting non-rotated
plotter.plot_mf(filtered_data)
Plotting rotated
plotter.plot_rw(rotated_wind)
Obtain the rotation angles
theta, phi = get_rotation_angles(filtered_data)
Calculate gamma using calc_gamma
$gamma = Angle$("gamma",
calc_gamma(theta.angle, phi.angle))

The results:
print(theta, phi, gamma)
theta:
RAD: -0.31457481937133086
DEG: -18.02380949106747
phi:
RAD: 1.6979828788548021
DEG: 97.28725264385352
gamma
RAD: 0.31696887828176834

DEG: 18.160978962541225
In Fig. 2, we show the nonrotated field components of the MC in the geocentric-solar-ecliptic (GSE) coordinates; while Fig. 3 depicts the rotated ones.

3.1. Swindpy command line interface

We also created a CLI (Command Line Interface) that makes it easier for a user to process MC data.
If a user likes to plot a no-rotated cloud, he or she can do it using the next command:

swindpy plot-cloud 2021-01-01 2021-01-02

If a user likes to plot a rotated cloud, he or she can do that using the next command:

swindpy plot-rotated-cloud 2021-01-01 2021-01-02

To export data about the magnetic field in a period in csv format, use the next command:

swindpy to-csv 2021-01-01 2021-01-02 output

A user can also plot both, no rotated and rotated clouds to compare and make a quick analysis of the results:
swindpy plot-rotated-and-non-rotated 2021-01-01 2021-01-02
The previous Matlab implementation results in Figs. 4 and 5 for the same example data (indicated between vertical lines) are consistent with the results obtained in Figs. 2 and 3 as can be seen, providing validation for this new tool.

4. Conclusions

Our goal was to make an easy-to-use tool for the community publicly available to find the orientation of a magnetic cloud and rotate it to its local frame. We hope our contribution will be useful to other researchers and provide a good synergy for the area encouraging other groups to make their coding available to the community as well for transparency and reproducibility of the results.

Acknowledgements: A.M.G is member of the Carrera del Investigador Científico, CONICET. This work was partially supported by the Argentinean grants PICT 2019-02754 (FONCyT-ANPCyT) and UBACyT-20020190100247BA (UBA)

References

Bothmer V., Schwenn R., 1998, AnnGeo, 16, 1
Burlaga L.F.E., 1991, *Physics of the Inner Heliosphere II*, vol. 21, 1–22, Springer
Dasso S., et al., 2005, ASR, 35, 2172
Gulisano A., 2004, Tesis lic. ciencias físicas: Helicidad magnética en nubes interplanetarias.
Gulisano A.M., et al., 2006, BAAA, 49, 34

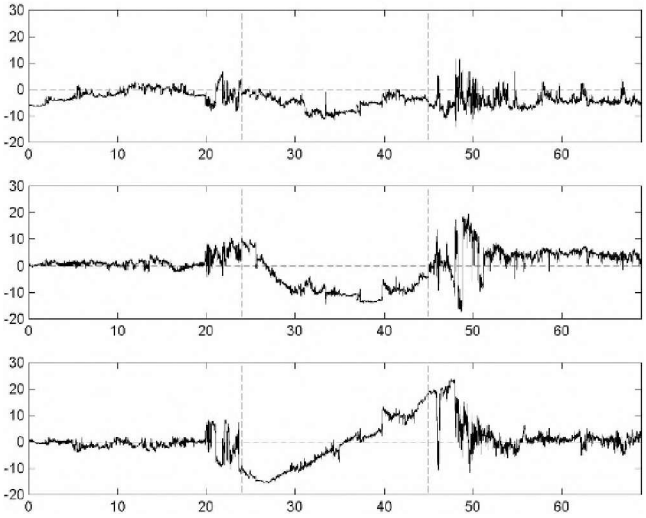

Fig. 4. Non-rotated MC data with the Matlab implementation, vertical lines provide the beginning and end of the MC. Each magnetic field component [nT] in GSE coordinates, are shown at the vertical axis $B_{x,\mathrm{GSE}}$ in the top panel, $B_{y,\mathrm{GSE}}$ in the middle panel and $B_{z,\mathrm{GSE}}$ in the bottom panel, time in hours from 1-9-1998 in the horizontal axis, (adapted from Gulisano, 2004)

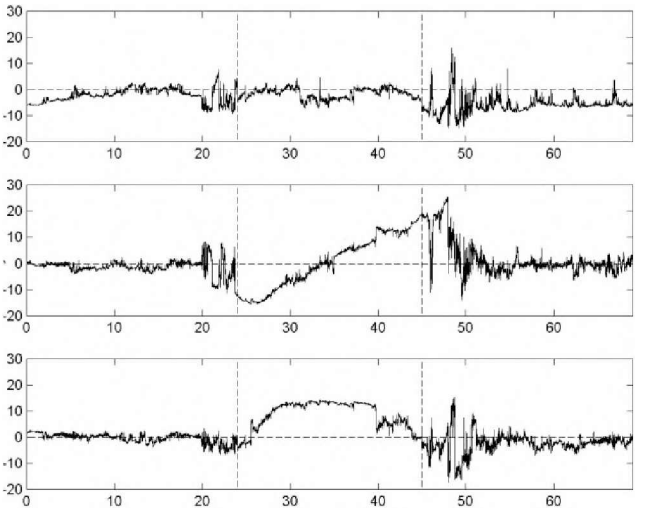

Fig. 5. Same as Fig. 4 except that the data have been rotated using the Matlab implementation, Each magnetic field component [nT] are shown at the vertical axis $B_{x,\mathrm{cloud}}$ in the top panel, $B_{y,\mathrm{cloud}}$ in the middle panel and $B_{z,\mathrm{cloud}}$ in the bottom panel, time in hours from 1-9-1998 in the horizontal axis (adapted from Gulisano, 2004)

Gulisano A.M., et al., 2007, ASR, 40, 1881
Gulisano A.M., et al., 2010a, M. Maksimovic, K. Issautier, N. Meyer-Vernet, M. Moncuquet, F. Pantellini (Eds.), *Twelfth International Solar Wind Conference*, 391–394, AIP CS
Gulisano A.M., et al., 2010b, A&A, 509, A39
Lepping R.P., Wu C.C., 2010, AnnGeo, 28, 1539
Rodriguez L., et al., 2016, SoPh, 291, 2145
Roth O., 2022, https://arxiv.org/abs/2208.14755