SolarWindPy: A Heliophysics Data Analysis Tool Set

B. L. Alterman [1]

1 Independent Scientist

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970

Published: unpublished

Summary

The region of space within the Sun's envelope of influence is called the heliosphere (the bubble of solar influence extending beyond the planets). The field of heliophysics (the study of the Sun and its influence throughout the solar system) starts in the solar interior and extends out to the very local interstellar medium, just beyond the heliosphere. The solar wind is a stream of charged particles that continuously flows away from the Sun, carrying, mass, energy, and momentum along with an embedded magnetic field. In short, it mediates the interaction of the Sun with the heliosphere and this is a feature shared by stars and their astrospheres more broadly. Changes in the solar wind are one source of space weather, which is a critical threat to our technological infrastructure on Earth and in space. SolarWindPy provides a unified framework for analyzing the solar wind and related space weather data, filling the gap between packages targeting astronomy, remote observations of the Sun, and general timeseries analysis of spacecraft based data. The package is available via PyPI[1] and conda-forge[2] and can be installed using `pip install solarwindpy` or `conda install -c conda-forge solarwindpy`.

Statement of Need

There is a growing ecosystem of python libraries to enable astrophysics, solar physics, plasma physics, and space physics. The table below cites key examples. Notably, there are several packages that support different elements of space physics, including magnetospheric data analysis (Pysat), integration of magnetospheric observations (SpacePy), and the retrieval and analysis of heliophysics timeseries data (pySpedas and PyTplot). Tools for the dedicated analysis of solar wind observations are noticeably absent. SolarWindPy fills this gap by providing a unified framework for analyzing solar wind observations in combination with relevant information about the spacecraft from which the observations were made. The package targets heliophysics researchers analyzing spacecraft observations, from graduate students learning plasma analysis to experienced scientists conducting multi-mission data studies.

Library	Purpose	Citation
AstroPy	Astronomical observations.	Astropy Collaboration et al. (2022)
SunPy	Remote sensing observations of the Sun.	Barnes et al. (2020)
PlasmaPy	Theoretical plasma physics.	(PlasmaPy Community, 2025)
SpacePy	Timeseries analysis and magnetospheric modeling.	Morley et al. (n.d.)
Pysat	Magnetospheric mission data analysis.	Stoneback et al. (2023)
pySpedas	Retrieval and plotting of heliophysics timeseries.	(Grimes et al., 2022)

[1] https://pypi.org/project/solarwindpy/
[2] https://anaconda.org/conda-forge/solarwindpy

Library	Purpose	Citation
PyTplot	Timeseries and spectrograph data visualization.	(Harter & MAVENSDC Team, 2019)

The SolarWindPy framework utilizes a pythonic, class-based architecture that combines ion and magnetic field objects into a single, unified plasma. It is designed for both experienced researchers and to provide an intuitive scaffold for students learning to analyze spacecraft data. SolarWindPy's primary functionality (core, fitfunctions, plotting, instabilities, and solar_activity submodules) was written by the author and developed or utilized in support of multiple publications B. L. Alterman, Rivera, Lepri, & Raines (2025). The transformation from thesis research code to a production package deployable via PyPI and conda-forge was accomplished using AI-assisted development with specialized quality assurance infrastructure for the supporting infrastructure (test suites, documentation, and deployment workflows), while the core scientific functionality remains human-authored.

The package builds on well-established libraries including NumPy van der Walt et al. (2011), SciPy (Virtanen et al., 2020), Matplotlib (Hunter, 2007), and Pandas Mckinney (2013) to ensure that the dependencies are stable. The plotting functionality retains the mapping between timeseries and aggregated observations to enable researchers to easily extract subsets of their observations for detailed analysis. The plot labeling functionality maps the quantities plotted to their file names, improving the mapping from the user's analysis to the saved output. The non-linear fitting libraries (utilizing scipy optimize) are designed for multi-step fitting in which the user performs nested regression of one variable on parameters derived from fitting other quantities. Submodules for the analysis of magnetohydrodynamic turbulence parameters and kinetic instabilities are also provided. The `solar_activity` submodule provides the user with seamless access to solar activity indicators provided by the LASP Interactive Solar IRradiance Datacenter (LISIRD) (Leise et al., 2019) and the Solar Information Data Center (SIDC) at the Royal Observatory of Belgium (**?**). This tool enables easy comparison of solar wind parameters across different phases of the solar cycle and different solar cycles, which is an essential component of solar wind data analysis. SolarWindPy currently stores data in pandas DataFrames and Timeseries objects. However, there is a clear separation between the two libraries such that future development could transition to using more nuanced and scientifically-targeted data structures, for example those provided by xarray (Hoyer & Hamman, 2017), SunPy, or AstroPy.

AI-Assisted Development Workflow

SolarWindPy's evolution from thesis research code (B. L. Alterman et al., 2018; Benjamin L. Alterman, 2019; B. L. Alterman & Kasper, 2019) to a production software package required comprehensive testing, documentation, and deployment infrastructure. To be explicit about the scope of AI assistance: the core scientific modules (`core/`, `fitfunctions/`, `plotting/`, `instabilities/`, `solar_activity/`) containing the physics algorithms and analysis methods were developed by the author without AI assistance and represent the scholarly contribution of this work, validated through eight peer-reviewed publications B. L. Alterman, Rivera, Lepri, & Raines (2025). AI-assisted development was used exclusively for supporting infrastructure: test suites, continuous integration pipelines, package deployment workflows, and completion of docstring documentation.

This was accomplished using Claude Code (Anthropic, 2024) with custom AI development infrastructure designed for scientific computing quality assurance.

The implementation includes specialized domain-specific agents and automated validation workflows using pre-commit hooks for physics validation, test execution, and coverage monitoring. This systematic approach enabled rapid development of test suites for modules outside the original core implementation, completion of documentation including missing docstrings, and creation of continuous integration and deployment pipelines for PyPI, conda-forge, and ReadTheDocs. The current agent system contains 7 specialized agents with an extensible architecture designed for integration with Claude Code's skills system. The infrastructure incorporates git commit integration, GitHub Issues planning workflows, and comprehensive audit trails to ensure traceability of all AI-generated modifications, establishing an infrastructure for trustworthy AI-assisted scientific software.

The project targets 95% test coverage, with core physics and plasma functionality currently achieving comprehensive coverage (95%), while tests for advanced features such as fitfunctions and plotting capabilities remain in active development, bringing overall coverage to 78%. All code generated or modified by AI in the supporting infrastructure (representing the test suites, CI/CD pipelines, and packaging tooling) undergoes expert review to ensure correctness, while the scientific algorithms themselves remain entirely human-authored as evidenced by their multi-year publication history. The complete AI-assisted development infrastructure, including agent specifications, validation hooks, and workflow automation, is publicly available in the `.claude/` directory of the repository.

Acknowledgements

The author thanks L. Woodham and R. D'Amicis for discussions about Alfvénic turbulence and calculating the Elsasser variables. In line with the transition to AI-augmented software development, Claude code (Anthropic, 2024) was used in writing this paper.

References

Alterman, Benjamin L. (2019). *The significance of proton beams in the multiscale solar wind* [PhD thesis]. University of Michigan.

Alterman, B. L. (2025). Characterizing the Impact of Alfvén Wave Forcing in Interplanetary Space on the Distribution of Near-Earth Solar Wind Speeds. *The Astrophysical Journal*, *984*(2), L64. https://doi.org/10.3847/2041-8213/add0a6

Alterman, B. L., & D'Amicis, R. (2025a). On the Regulation of the Solar Wind Helium Abundance by the Hydrogen Compressibility. *Astrophysical Journal Letters (Accepted)*.

Alterman, B. L., & D'Amicis, R. (2025b). Cross Helicity and the Helium Abundance as an In Situ Metric of Solar Wind Acceleration. *The Astrophysical Journal*, *982*(2), L40. https://doi.org/10.3847/2041-8213/adb48e

Alterman, B. L., & Kasper, J. C. (2019). Helium Variation across Two Solar Cycles Reveals a Speed-dependent Phase Lag. *The Astrophysical Journal*, *879*(1), L6. https://doi.org/10.3847/2041-8213/ab2391

Alterman, Benjamin L., Kasper, J. C., Leamon, R. J., & McIntosh, S. W. (2021). Solar Wind Helium Abundance Heralds Solar Cycle Onset. *Solar Physics*, *296*(4), 67. https://doi.org/10.1007/s11207-021-01801-9

Alterman, B. L., Kasper, J. C., Stevens, M. L., & Koval, A. (2018). A Comparison of Alpha Particle and Proton Beam Differential Flows in Collisionally Young Solar Wind. *The Astrophysical Journal*, *864*(2), 112. https://doi.org/10.3847/1538-4357/aad23f

Alterman, B. L., Rivera, Y. J., Lepri, S. T., & Raines, J. M. (2025). The transition from slow to fast wind as observed in composition observations. *Astronomy and Astrophysics*, *694*, A265. https://doi.org/10.1051/0004-6361/202451550

Alterman, B. L., Rivera, Y. J., Lepri, S. T., Raines, J. M., & D'Amicis, R. (2025). The Evolution of Heavy Ion Abundances with Solar Activity. *arXiv e-Prints*, arXiv:2504.18092. https://doi.org/10.48550/arXiv.2504.18092

The Journal of Open Source Software

Anthropic. (2024). *Claude code: An agentic coding tool*. https://github.com/anthropics/claude-code

Astropy Collaboration, Price-Whelan, A. M., & others. (2018). The Astropy Project: Building an Open-science Project and Status of the v2.0 Core Package. *Aj*, *156*(3), 123. https://doi.org/10.3847/1538-3881/aabc4f

Astropy Collaboration, Price-Whelan, A. M., & others. (2022). The Astropy Project: Sustaining and Growing a Community-oriented Open-source Project and the Latest Major Release (v5.0) of the Core Package. *Apj*, *935*(2), 167. https://doi.org/10.3847/1538-4357/ac7c74

Astropy Collaboration, Robitaille, T. P., & others. (2013). Astropy: A community Python package for astronomy. *Aap*, *558*, A33. https://doi.org/10.1051/0004-6361/201322068

Barnes, W. T., Bobra, M. G., Christe, S. D., Freij, N., Hayes, L. A., Ireland, J., Mumford, S., Perez-Suarez, D., Ryan, D. F., Shih, A. Y., Chanda, P., Glogowski, K., Hewett, R., Hughitt, V. K., Hill, A., Hiware, K., Inglis, A., Kirk, M. S. F., Konge, S., … Dang, T. K. (2020). The SunPy Project: Open Source Development and Status of the Version 1.0 Core Package. *The Astrophysical Journal*, *890*(1), 68. https://doi.org/10.3847/1538-4357/ab4f7a

Grimes, E. W., Harter, B., Hatzigeorgiu, N., Drozdov, A., Lewis, J. W., Angelopoulos, V., Cao, X., Chu, X., Hori, T., Matsuda, S., Jun, C.-W., Nakamura, S., Kitahara, M., Segawa, T., Miyoshi, Y., & Le Contel, O. (2022). The space physics environment data analysis system in python. *Frontiers in Astronomy and Space Sciences*, *9*, 1020815. https://doi.org/10.3389/fspas.2022.1020815

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Harter, B., & MAVENSDC Team. (2019). PyTplot: A Python version of the IDL tplot libraries. In *GitHub repository*. GitHub. https://github.com/MAVENSDC/PyTplot

Hoyer, S., & Hamman, J. (2017). Xarray: N-D labeled arrays and datasets in Python. *Journal of Open Research Software*, *5*(1). https://doi.org/10.5334/jors.148

Hunter, J. D. (2007). Matplotlib: A 2D Graphics Environment. *Computing in Science & Engineering*, *9*(3), 90–95. https://doi.org/10.1109/MCSE.2007.55

Leise, H., Baltzer, T., Wilson, A., Lindholm, D., Snow, M., Woodraska, D., Béland, S., Coddington, O., & C., P. (2019). LASP Interactive Solar IRradiance Datacenter (LISIRD). *EGU General Assembly Conference Abstracts*. https://ui.adsabs.harvard.edu/abs/2019EGUGA..2112479L

Mckinney, W. (2010). Data Structures for Statistical Computing in Python. In S. van der Walt & J. Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 51–56).

McKinney, W. (2011). Pandas: A Foundational Python Library for Data Analysis and Statistics. *Python for High Performance and Scientific Computing*, 1–9.

Mckinney, W. (2013). *Python for Data Analysis*. O'Reilly. https://doi.org/10.1145/1985441.1985476

Morley, S. K., Niehof, J. T., Welling, D. T., Larsen, B. A., Brunet, A., Engel, M. A., Gieseler, J., Haiducek, J., Henderson, M., Hendry, A., Hirsch, M., Killick, P., Koller, J., Merrill, A., Rastatter, L., Reimer, A., Shih, A. Y., & Stricklan, A. (n.d.). *SpacePy*. Zenodo. https://doi.org/10.5281/zenodo.3252523

Mumford, S. J., & others. (2020). SunPy: A python package for solar physics. *Journal of Open Source Software*, *5*(46), 1832. https://doi.org/10.21105/joss.01832

Niehof, J. T., Morley, S. K., Welling, D. T., & Larsen, B. A. (2022). The SpacePy space science package at 12 years. *Frontiers in Astronomy and Space Sciences*, *9*. https://doi.org/10.3389/fspas.2022.1023612

PlasmaPy Community. (2025). *PlasmaPy* (Version 2025.8.0). Zenodo. https://doi.org/10.5281/zenodo.16747747

Stoneback, R. A., Burrell, A. G., Klenzing, J., & Depew, M. D. (2018). PYSAT: Python Satellite Data Analysis Toolkit. *Journal of Geophysical Research: Space Physics*, *123*(6), 5271–5283. https://doi.org/10.1029/2018JA025297

Stoneback, R. A., Burrell, A. G., Klenzing, J., & Smith, J. (2023). The pysat ecosystem. *Frontiers in Astronomy and Space Science*, *10*. https://doi.org/10.3389/fspas.2023.1119775

Stoneback, R. A., Klenzing, J. H., Burrell, A. G., Spence, C., Depew, M., Hargrave, N., Smith, J., Bose, V. von, Pembroke, A., Iyer, G., & Luis, S. (2021). *Python satellite data analysis toolkit (pysat) vX.y.z*. https://doi.org/10.5281/zenodo.1199703

The SunPy Community, Barnes, W. T., & others. (2020). The SunPy project: Open source development and status of the version 1.0 core package. *The Astrophysical Journal*, *890*, 68–68. https://doi.org/10.3847/1538-4357/ab4f7a

van der Walt, S., Colbert, S. C., & Varoquaux, G. (2011). The NumPy Array: A Structure for Efficient Numerical Computation. *Computing in Science & Engineering*, *13*(2), 22–30. https://doi.org/10.1109/MCSE.2011.37

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Van Der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., … Vazquez-Baeza, Y. (2020). SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nature Methods*, *17*(3), 261–272. https://doi.org/10.1038/s41592-019-0686-2