The Journal of Open Source Software

solposx: A Python package for determining solar position and atmospheric refraction

Adam R. Jensen [1*¶], **Ioannis Sifnaios** [1*], **Kevin S. Anderson** [2*], and **Echedey Luis** [3]

1 Technical University of Denmark (DTU), Denmark **2** Sandia National Laboratories, USA **3** Universidad Politécnica de Madrid (UPM), Spain ¶ Corresponding author * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Open Journals

Reviewers:

- @openjournals

Submitted: 01 January 1970
Published: unpublished

Summary

solposx is a Python package of reference algorithms for calculating the sun's position and atmospheric refraction. The package includes 11 solar position algorithms and 6 refraction models from the past 50 years. All functions follow a standardized design pattern, making it easy to compare different algorithms. The provided algorithm implementations have been thoroughly vetted, making the package a valuable research tool and a reliable reference for implementing solar position algorithms in other programming languages or applications.

Statement of need

Calculating the sun's position is a fundamental task in solar energy research, for example, when modeling solar irradiance, estimating the yield of photovoltaic (PV) systems, or determining rotation angles for solar trackers. For this reason, the literature contains numerous solar position algorithms (SPAs) (Blanco et al., 2020; Michalsky, 1988; Reda & Andreas, 2004; Spencer, 1971; Walraven, 1978).

Existing SPAs vary in accuracy, computational speed, and period of validity. These characteristics are usually tradeoffs, and thus the choice of algorithm depends on the specific application. Some algorithms have been developed to be computationally lightweight for use in solar tracker microcontrollers, and as a tradeoff, are inaccurate for past and future years. In contrast, high-accuracy algorithms may consist of several hundred mathematical operations to retain validity for hundreds or even thousands of years. One example of such an algorithm is the SPA from NREL, whose high accuracy and extensive period of validity come at the cost of being computationally expensive and impractical to implement.

Solar position algorithms are already available in several open source software packages, such as the PV modeling software packages `pvlib-python` (Anderson et al., 2023) and `pysolar` (Stafford, 2007), the astronomy packages `pyephem` (Rhodes, 2011) and `skyfield` (Rhodes, 2019), and the sun physics package `sunpy` (The SunPy Community et al., 2020). These packages are tailored to very specific purposes and only contain one or a few different solar position algorithms. Consequently, there are many solar position algorithms for which an open source reference implementation is not available. This makes it difficult to evaluate the tradeoffs of the various solar position algorithms, which is necessary in order to make informed decisions on which algorithm to choose for a specific application.

SolarPositionX (`solposx`) is a Python package for calculating solar position angles and atmospheric refraction corrections. The package provides reference implementations of a large number of solar position and refraction correction algorithms spanning 50 years of the scientific literature. The SPAs range from simple algorithms based on fitted equations to research-grade astronomy algorithms based on complex ephemerides. As of `solposx` version v1.0.0, the package includes 11 different solar position algorithms and 6 algorithms for estimating atmospheric refraction. The "X" in `solposx` refers to the modular design of the package, allowing users to seamlessly switch between a variety of algorithms depending on their desired needs. An overview of the modules and functions is provided in Figure 1.

The solar position functions follow a standard pattern, taking three main input parameters (times, latitude, and longitude) and returning a Pandas `DataFrame` with solar elevation, zenith, and azimuth angles. This makes it extremely easy to compare and switch between SPAs, regardless of whether the functions execute code from within the `solposx` package, rely on external Python packages (which is the case for the `skyfield` and `sg2` functions), or retrieve data remotely (which is the case for NASA's Horizons service (NASA Jet Propulsion Laboratory, California Institute of Technology, 2025)). The refraction correction models also follow a standardized pattern where the main input is an array or series of solar elevation angles and the output is the atmospheric refraction correction angles.

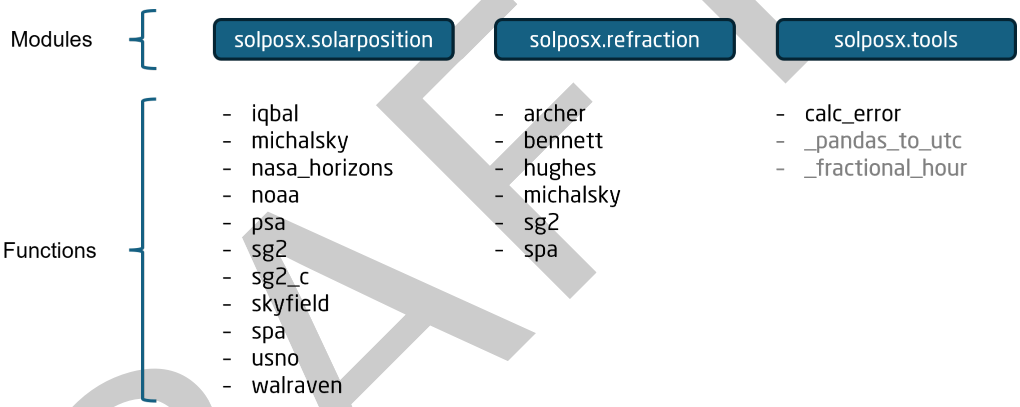

Figure 1: Overview of modules and functions in the `solposx` package.

The package relies heavily on the `Pandas` Python package (McKinney, 2010), due to its convenient `DatetimeIndex` class. The reason for this choice is that it offers a very convenient way to handle timestamps, including timezone information and conversion between timezones.

Besides direct applications of calculating solar position, one of the main use cases of the package is providing verified reference implementations to users who are implementing algorithms in other languages or applications. Access to verified reference implementations is an essential tool as solar position algorithms tend to be sensitive to small implementation details. For example, a small detail such as using an incorrect rounding convention, e.g., rounding towards zero vs. rounding down, can result in solar position angles being off by more than 0.1 degrees, an error much larger than the claimed accuracy of most SPAs. Such subtle but serious implementation errors are, in the authors' experience, almost inevitable when implementing SPAs, creating a need for correct and accessible reference implementations. With access to vetted and tested reference implementations of these SPAs, users can generate reliable test values for validating and debugging their own implementations. Notably, the `solposx` package has already been used for research purposes, most recently in a study comparing the performance of solar position algorithms for PV applications (Jensen et al., 2025).

`solposx` is developed openly on GitHub and released under a BSD 3-clause license, allowing permissive use with attribution. The package is extensively tested, ensuring that the algorithms work for a large range of inputs and remain consistent. In general, `solposx` has been developed following modern best practices for packaging, documentation, and testing. Additional algorithms are expected to be added as new algorithms are developed or if additional historical algorithms of interest are identified.

Acknowledgements

Adam R. Jensen and Ioannis Sifnaios were supported by the Danish Energy Agency through grant no. 134232-510237.

This work was supported in part by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the Solar Energy Technologies Office Award Number 52788. Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. This paper describes objective technical results and analysis. Any subjective views or opinions that might be expressed in the paper do not necessarily represent the views of the U.S. Department of Energy or the United States Government.

References

Anderson, K. S., Hansen, C. W., Holmgren, W. F., Jensen, A. R., Mikofski, M. A., & Driesse, A. (2023). Pvlib python: 2023 project update. *Journal of Open Source Software*, *8*(92), 5994. https://doi.org/10.21105/joss.05994

Blanco, M. J., Milidonis, K., & Bonanos, A. M. (2020). Updating the PSA sun position algorithm. *Solar Energy*, *212*, 339–341. https://doi.org/10.1016/j.solener.2020.10.084

Jensen, A. R., Sifnaios, I., & Anderson, K. S. (2025). *Solar Position Algorithms*. https://pvpmc.sandia.gov/download/8943/?tmstv=1754599268

McKinney, Wes. (2010). Data Structures for Statistical Computing in Python. In Stéfan van der Walt & Jarrod Millman (Eds.), *Proceedings of the 9th Python in Science Conference* (pp. 56–61). https://doi.org/10.25080/Majora-92bf1922-00a

Michalsky, J. J. (1988). The astronomical almanac's algorithm for approximate solar position (1950–2050). *Solar Energy*, *40*(3), 227–235. https://doi.org/10.1016/0038-092x(88)90045-x

NASA Jet Propulsion Laboratory, California Institute of Technology. (2025). *Horizons - solar system dynamics*. https://ssd.jpl.nasa.gov

Reda, I., & Andreas, A. (2004). Solar position algorithm for solar radiation applications. *Solar Energy*, *76*(5), 577–589. https://doi.org/10.1016/j.solener.2003.12.003

Rhodes, B. (2011). *PyEphem: Astronomical Ephemeris for Python*. Astrophysics Source Code Library, ascl:1112.014. https://rhodesmill.org/pyephem/

Rhodes, B. (2019). *Skyfield: High precision research-grade positions for planets and Earth satellites generator*. Astrophysics Source Code Library, record ascl:1907.024.

Spencer, J. (1971). Fourier series representation of the position of the sun. *Search*, *2*(5), 172.

Stafford, B. (2007). *Pysolar: Python libraries for simulating solar irradiation*. https://pypi.org/project/pysolar/

The SunPy Community, 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., … Dang, T. K. (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

Walraven, R. (1978). Calculating the position of the sun. *Solar Energy*, *20*(5), 393–397. https://doi.org/10.1016/0038-092X(78)90155-X