pvlib python: 2023 project update

Kevin S. Anderson [*1], **Clifford W. Hansen** [1], **William F. Holmgren** [2], **Adam R. Jensen** [3], **Mark A. Mikofski** [2], **and Anton Driesse** [4]

1 Sandia National Laboratories **2** DNV **3** Technical University of Denmark **4** PV Performance Labs

DOI: 10.21105/joss.05994

Software

- Review
- Repository
- Archive

Editor: Rachel Kurchin

Reviewers:

- @EwaGomez
- @phoebe-p

Submitted: 18 September 2023
Published: 22 December 2023

Summary

pvlib python is a community-developed, open-source software toolbox for simulating the performance of solar photovoltaic (PV) energy components and systems. It provides reference implementations of over 100 empirical and physics-based models from the peer-reviewed scientific literature, including solar position algorithms, irradiance models, thermal models, and PV electrical models. In addition to individual low-level model implementations, pvlib python provides high-level workflows that chain these models together like building blocks to form complete "weather-to-power" photovoltaic system models. It also provides functions to fetch and import a wide variety of weather datasets useful for PV modeling.

pvlib python has been developed since 2013 and follows modern best practices for open-source python software, with comprehensive automated testing, standards-based packaging, and semantic versioning. Its source code is developed openly on GitHub and releases are distributed via the Python Package Index (PyPI) and the conda-forge repository. pvlib python's source code is made freely available under the permissive BSD-3 license.

Here we (the project's core developers) present an update on pvlib python, describing capability and community development since our 2018 publication (Holmgren, Hansen, & Mikofski, 2018).

Statement of need

PV performance models are used throughout the field of photovoltaics. The rapid increase in scale, technological diversity, and sophistication of the global solar energy industry demands correspondingly more capable models. Per the United States Department of Energy, "the importance of accurate modeling is hard to overstate" (Solar Energy Technologies Office, 2022).

Compared with other modern PV system modeling tools, pvlib python stands out in several key aspects. One is its toolbox design, providing the user a level of flexibility and customization beyond that of other tools. Other PV system modeling tools like SAM (Gilman et al., 2018), PVsyst (Mermoud, 1994), SolarFarmer (Mikofski et al., 2018), PlantPredict (Passow et al., 2017), and CASSYS (Pai & Thevenard, 2016)—to name a few software tools with comparable breadth of modeling capability—organize the user interface around pre-built modeling workflows. Instead, pvlib python makes the individual "building blocks" of PV system performance models accessible to the user. This allows the user to assemble their own model workflows, including the ability of incorporating custom modeling steps. This flexibility is essential for applications in both academia and industry. To our knowledge, the only other PV system modeling software with such a toolbox design is the original MATLAB version of pvlib (Andrews et al., 2014). pvlib python began as a translation of that code base and has since surpassed it in terms of capability, community uptake, and development attention.

*Author order is sorted by code commits to the project's main branch from 2018-09-07 to 2023-12-18.

Another key aspect of pvlib python is that it is both implemented and operated with a general-purpose programming language (Python), which allows pvlib python functions to be combined with capabilities in other Python packages, such as database query, data manipulation, numerical optimization, plotting, and reporting packages. In contrast, most other PV system modeling tools are used via some form of GUI. Some of these other tools are also accessible from Python via web APIs or wrapper libraries (Gilman et al., 2019), but these "black box" interfaces offer only limited ability to combine the PV models with functionality from other Python packages.

A final key aspect of pvlib python is its open peer review approach and foundation on published scientific research, allowing it to be developed by a decentralized and diverse community of PV researchers and practitioners without compromising its focus on transparent and reliable model implementations. This is in contrast to the inherent opaqueness of closed-source commercial software, which prevents users from inspecting the source code to ensure a model implementation's validity or traceability to a reference. It is also in contrast to other open-source PV projects, where code review and contributions typically come from a single institution.

These key aspects, along with sustained contributions from a passionate and committed community, have led to pvlib python's widespread adoption across the PV field (Stein & Hansen, 2022). In support of the claim that pvlib python provides meaningful value and addresses real needs, we offer these quantitative metrics:

1. Its 2018 JOSS publication, at the time of this writing, ranks 14th by citation count out of the 2000+ papers published by JOSS to date.
2. The Python Package Index (PyPI) classifies pvlib python as a "critical project" due to being in the top 1% of the index's packages by download count.
3. The project's online documentation receives over 400,000 page views per year.
4. pvlib python was found to be the third most-used python project in the broader open-source sustainability software landscape, with the first two being netCDF4 utilities applicable across many scientific fields (Augspurger et al., 2023).

Functionality additions

To meet new needs of the PV industry, substantial new functionality has been added in the roughly five years since the 2018 JOSS publication.

First, several dozen new models have been implemented, expanding the package's capability in both existing and new modeling areas and prompting the creation of several new modules within pvlib python. In response to the recent rapid increase in deployment of bifacial PV, a capability enhancement of particular note is the inclusion of models for simulating irradiance on the rear side of PV modules. Other notable additions include methods of fitting empirical PV performance models to measurements and models for performance loss mechanisms like soiling and snow coverage. Figure 1 summarizes the number of models (or functions) per module for pvlib python versions 0.6.0 (released 2018-09-17) and 0.10.1 (released 2023-07-03), showing a substantial capability expansion over the last five years.

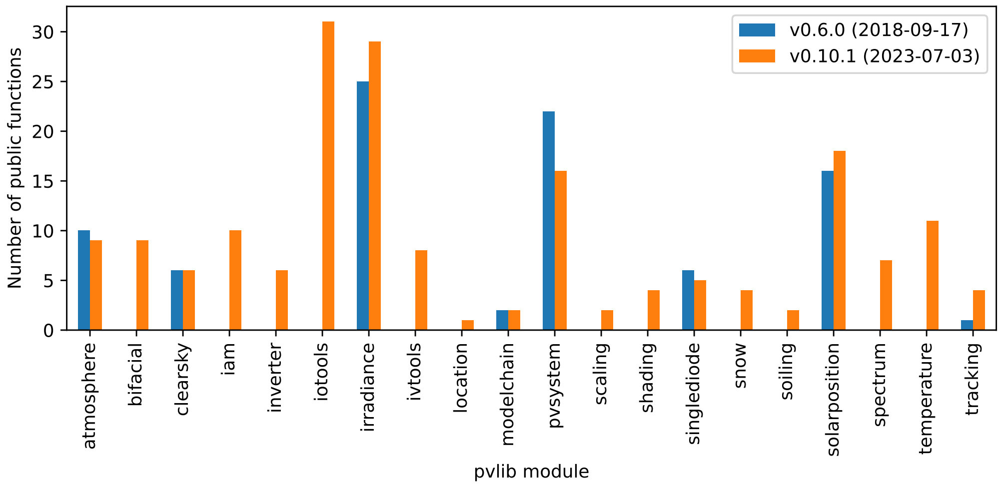

Figure 1: Comparison of public function counts for selected pvlib modules for v0.6.0 and v0.10.1. Some modules are smaller in v0.10.1 due to moving functions to new modules (e.g. from `pvsystem` to `iam`).

Second, in addition to the new function-level model implementations, the package's high-level classes have also been expanded to support the complexity of emerging system designs, including heterogeneous systems whose subsystems differ in mounting or electrical configuration and systems that require custom orientation/tracking models.

Third, a new subpackage `pvlib.iotools` has been created for fetching and importing datasets relevant to PV modeling. These functions provide a standardized interface for reading data files in various complex data formats, offering conveniences like the option to standardize dataset variable names and units to pvlib's conventions (Jensen et al., 2023). As of version 0.10.1, `pvlib.iotools` contains functions to download data from over ten online data providers, plus file reading/parsing functions for a dozen solar resource file formats.

These additions are discussed in more detail in (Hansen et al., 2023) and (Anderson et al., 2022). Complete descriptions of the changes in each release can be found in the project's documentation.

Community growth

It is difficult to fully describe the community around open-source projects like pvlib python, but some aspects can be quantified. Here we examine the community from a few convenient perspectives, emphasizing that these metrics provide a limited view of the community as a whole.

First, we examine contributors to pvlib python's code repository. The project's use of version control software enables easy quantification of repository additions (to code, documentation, tests, etc.) over time. The project's repository currently comprises contributions from over 100 people spanning industry, academia, and government research institutions. Figure 2 (left) shows the number of unique repository contributors over time, demonstrating continued and generally accelerating attraction of new contributors.

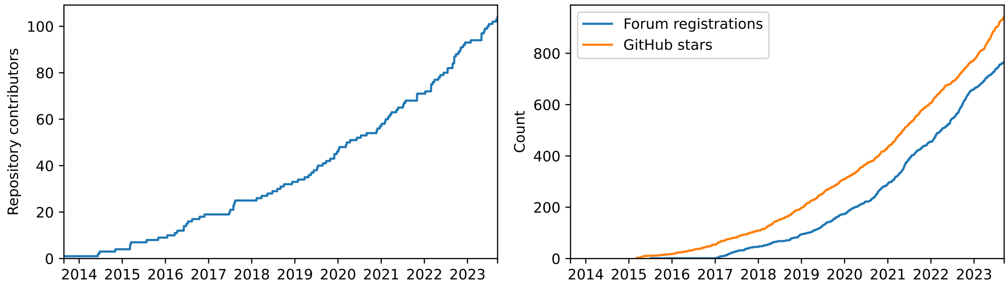

Figure 2: Total repository contributor count over time (left) and other community size statistics (right).

However, the project as a whole is the product of not only of those who contribute code but also those who submit bug reports, propose ideas for new features, participate in online forums, and support the project in other ways. Along those lines, two easily tracked metrics are the number of people registered in the pvlib python online discussion forum and the number of GitHub "stars" (an indicator of an individual's interest, akin to a browser bookmark) on the pvlib python code repository. Figure 2 (right) shows these counts over time. Although these numbers almost certainly underestimate the true size of the pvlib community, their increase over time indicates continued and accelerating community growth. Since the 2018 JOSS publication, pvlib python has doubled the number of maintainers to bring in new perspectives and to better support the growing community.

In addition to continuous interaction online, community members sometimes meet in person at user's group and tutorial sessions run by pvlib python maintainers and community members alike. To date, these meetings have been held at the IEEE Photovoltaics Specialists Conference (PVSC), the PVPMC Workshops, and the PyData Global conference. Figure 3 shows a timeline of these meetings, along with other notable events in the project's history.

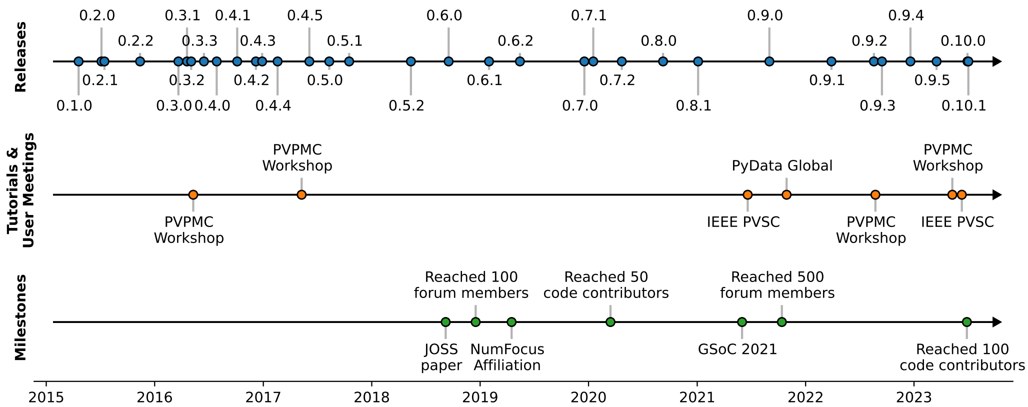

Figure 3: pvlib python major event timeline: releases (top), community events (middle), and other project milestones (bottom).

Finally, it is worth pointing out that pvlib python contributors and users are part of a broader community for the pvlib software "family", which includes pvanalytics, a package for PV data quality assurance and feature recognition algorithms (Perry et al., 2022), and twoaxistracking, a package for simulating self-shading in arrays of two-axis solar trackers (Jensen et al., 2022). Moreover, looking beyond pvlib and its affiliated packages, we see that Python is proving to be the most common programming language in general for open-source PV modeling and analysis software. The packages mentioned here make up one portion of a growing landscape of Python-for-PV projects (Holmgren, Hansen, Stein, et al., 2018).

Acknowledgements

Although much of the development and maintenance of pvlib python is on a volunteer basis, the project has also been supported by projects with various funding sources, including:

- The U.S. Department of Energy's Solar Energy Technology Office, through the PV Performance Modeling Collaborative (PVPMC) and other projects
- The Danish Energy Agency through grant nos. 134223-496801 and 134232-510237
- NumFOCUS's Small Development Grant program
- Google's Summer of Code program

pvlib python benefits enormously from building on top of various high-quality packages that have become de facto standards in the python ecosystem: numpy (Harris et al., 2020), pandas (McKinney, 2010), scipy (Virtanen et al., 2020), and numba (Lam et al., 2015) for numerics, matplotlib (Hunter, 2007) for plotting, sphinx (Komiya et al., 2023) for documentation, and pytest (Krekel et al., 2004) for automated testing. The project also benefits from online infrastructure generously provided free of charge, including GitHub (code development and automated testing) and ReadTheDocs.org (documentation building and hosting).

This work was supported by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under the Solar Energy Technologies Office Award Number 38267. 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., Holmgren, W., Hansen, C., Mikofski, M., Jensen, A. R., & Driesse, A. (2022). pvlib python 2022 update. *PV Performance Modeling and Monitoring Workshop*. https://doi.org/10.2172/2004407

Andrews, R. W., Stein, J. S., Hansen, C., & Riley, D. (2014, June). Introduction to the open source PV LIB for python photovoltaic system modelling package. *40th IEEE Photovoltaic Specialists Conference*. https://doi.org/10.1109/pvsc.2014.6925501

Augspurger, T., Malliaraki, E., Hopkins, J., & Brown, D. (2023). *The open source sustainability ecosystem*. The Linux Foundation.

Gilman, P., DiOrio, N. A., Freeman, J. M., Janzou, S., Dobos, A., & Ryberg, D. (2018). *SAM photovoltaic model technical reference 2016 update*. Office of Scientific and Technical Information (OSTI). https://doi.org/10.2172/1429291

Gilman, P., Janzou, S., Guittet, D., Freeman, J., DiOrio, N., Blair, N., Boyd, M., Neises, T., & Wagner, M. (2019). *PySAM (python wrapper for system advisor model "SAM") [SWR-19-57]*. National Renewable Energy Laboratory (NREL), Golden, CO (United States). https://doi.org/10.11578/DC.20190903.1

Hansen, C., Anderson, K., Vining, William, Holmgren, W., Mikofski, M., Jensen, A. R., Driesse, A., & Perry, K. (2023). pvlib 2023 update: pvlib-python, pvanalytics, twoaxistracking. *PV Performance Modeling and Monitoring Workshop*.

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., Río, J. F. del, 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

Holmgren, W. F., Hansen, C. W., & Mikofski, M. A. (2018). pvlib python: A python package for modeling solar energy systems. *Journal of Open Source Software*, *3*(29), 884. https://doi.org/10.21105/joss.00884

Holmgren, W. F., Hansen, C. W., Stein, J. S., & Mikofski, M. A. (2018). Review of open source tools for PV modeling. *45th IEEE Photovoltaic Specialists Conference*. https://doi.org/10.5281/zenodo.1401378

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

Jensen, A. R., Anderson, K. S., Holmgren, W. F., Mikofski, M. A., Hansen, C. W., Boeman, L. J., & Loonen, R. (2023). pvlib iotools—open-source python functions for seamless access to solar irradiance data. *Solar Energy*, *266*, 112092. https://doi.org/10.1016/j.solener.2023.112092

Jensen, A. R., Sifnaios, I., & Anderson, K. (2022). twoaxistracking – a python package for simulating self-shading of two-axis tracking solar collectors. *MethodsX*, *9*, 101876. https://doi.org/10.1016/j.mex.2022.101876

Komiya, T., Brandl, G., B., J.-F., SHIMIZUKAWA, T., Andersen, J. L., Turner, A., Finucane, S., Lehmann, R., Kampik, T., Magin, J., jacobmason, Dufresne, J., Waltman, J., Rodríguez, J. L. C., Ronacher, A., Geier, M., Shachnev, D., Ruana, R., Virtanen, P., … cocoatomo. (2023). *Sphinx-doc/sphinx: v7.0.1* (Version v7.0.1). Zenodo. https://doi.org/10.5281/zenodo.7931414

Krekel, H., Oliveira, B., Pfannschmidt, R., Bruynooghe, F., Laugher, B., & Bruhin, F. (2004). *Pytest*. https://github.com/pytest-dev/pytest

Lam, S. K., Pitrou, A., & Seibert, S. (2015). Numba: A LLVM-based python JIT compiler. *Proceedings of the Second Workshop on the LLVM Compiler Infrastructure in HPC*, 1–6. https://doi.org/10.1145/2833157.2833162

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

Mermoud, A. (1994). PVSYST: A user-friendly software for PV-systems simulation. *Twelfth European Photovoltaic Solar Energy Conference*.

Mikofski, M. A., Lynn, M., Byrne, J., Hamer, M., Neubert, A., & Newmiller, J. (2018, June). Accurate performance predictions of large PV systems with shading using submodule mismatch calculation. *7th World Conference on Photovoltaic Energy Conversion (WCPEC)*. https://doi.org/10.1109/pvsc.2018.8547323

Pai, A., & Thevenard, D. (2016, June). Introducing CASSYS: An open-source software for simulation of grid-connected photovoltaic systems. *43rd IEEE Photovoltaic Specialists Conference*. https://doi.org/10.1109/pvsc.2016.7749839

Passow, K., Ngan, L., Rich, G., Lee, M., & Kaplan, S. (2017, June). PlantPredict: Solar performance modeling made simple. *44th IEEE Photovoltaic Specialists Conference (PVSC)*. https://doi.org/10.1109/pvsc.2017.8366450

Perry, K., Vining, W., Anderson, K., Muller, M., & Hansen, C. (2022). PVAnalytics: A python package for automated processing of solar time series data. *PV Performance Modeling and Monitoring Workshop*. https://www.osti.gov/biblio/1887283

Solar Energy Technologies Office. (2022). *Modeling of photovoltaic systems: Basic challenges and DOE-funded tools*. U.S. Department of Energy, Office of Energy Efficiency and

Renewable Energy.

Stein, J., & Hansen, C. (2022). A twelve-year retrospective on pvlib: Open-source PV performance modeling library. *SETO-Funded Open-Source Software: Building Community Engagement for Lasting Impact*. https://www.energy.gov/sites/default/files/2022-10/DOE%20OSS%20Workshop%2C%20Josh%20Stein%2C%20Sandia.pdf

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., … SciPy 1.0 Contributors. (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, *17*, 261–272. https://doi.org/10.1038/s41592-019-0686-2