pvlib 2023 update: pvlib-python, pvanalytics, twoaxistracking

pvlib-python

Cliff Hansen (Sandia)
Kevin Anderson (Sandia)
Will Holmgren (DNV)
Mark Mikofski (DNV)
Adam R. Jensen (DTU)
Anton Driesse (PV Performance Labs)

pvanalytics

Cliff Hansen (Sandia)
Will Vining (Sandia)
Kevin Anderson (Sandia)
Kirsten Perry (NREL)

twoaxistracking

Adam R. Jensen (DTU)
Kevin Anderson (Sandia)

2023 European PVPMC Workshop
Mendrisio, Switzerland, November 9, 2023

1. What is pvlib?
2. What is pvlib python
3. Documentation and Tutorials
4. Enhancements since v0.6 (2018)
5. Community Growth
6. Introductions to pvanalytics and twoaxistracking

What is pvlib?

A python ecosystem of compatible packages for PV systems modeling and analysis that are **community-driven**, **free**, **open-source**, and **well-documented**

pvlib-python

Library of functions for weather-to-power modeling

Customizable end-to-end PV system modeling (ModelChain)

Batteries-included data import library

pvanalytics

Library of functions for analysis of data from PV systems

Filtering and quality checks

Feature labeling: e.g., inverter clipping

twoaxistracking

Simulate two-axis tracking solar collectors

Emphasis on self-shading

Find us at: https://github.com/pvlib

pvlib

What is pvlib python?

A python library for PV performance modeling

Modeling Toolbox

Stand-alone models for:

Atmosphere	Snow
Solar position	Soiling
Transposition	Shading
Bifacial	I-V curves
Temperature	Inverters
Clear-sky	IAM

…and more!

Weather-to-power workflow

Customizable end-to-end PV system modeling (ModelChain)

Scriptable and automatable by design

Data I/O

Batteries-included data import:

TMY	SURFRAD
EPW	SOLRAD
NSRDB	MIDC
PVGIS	BSRN
CAMS	UO SRML
ECMWF MACC	NOAA USCRN

pvlib

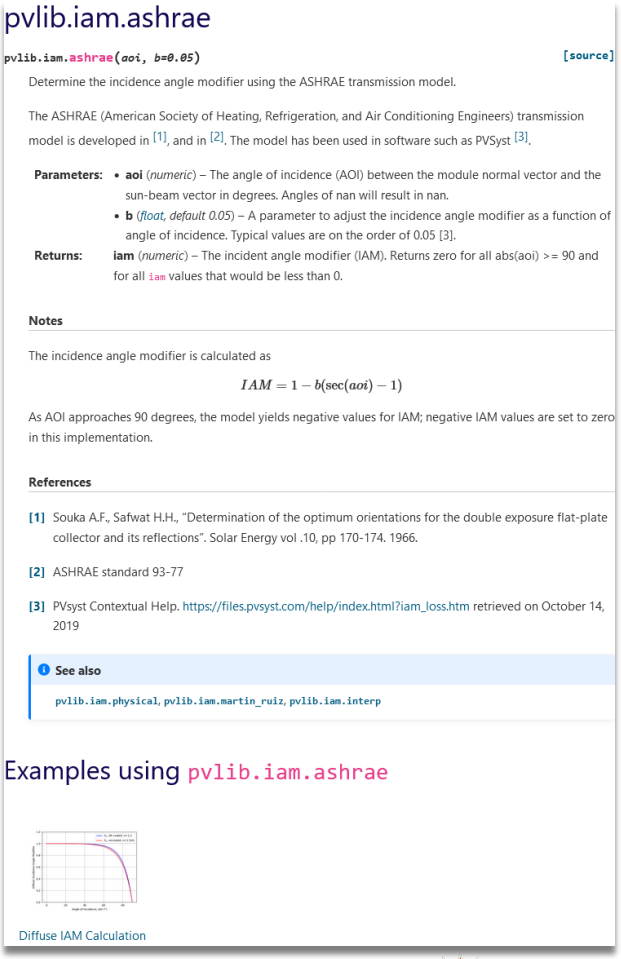
pvlib python Documentation: Model Descriptions

Each model function has a page with:

- Brief model description
- Inputs: description, data types, units
- Outputs: description, date types, units
- Published reference(s) for the model
- Links to other relevant functions
- Links to relevant gallery examples
- Other notes as needed

Several hundred model-level pages, all built automatically from in-code documentation

https://pvlib-python.readthedocs.io



pvlib.iam.ashrae

pvlib.iam.ashrae(aoi, b=0.05) [source]

Determine the incidence angle modifier using the ASHRAE transmission model.

The ASHRAE (American Society of Heating, Refrigeration, and Air Conditioning Engineers) transmission model is developed in [1], and in [2]. The model has been used in software such as PVSyst [3].

Parameters:
- aoi (numeric) – The angle of incidence (AOI) between the module normal vector and the sun-beam vector in degrees. Angles of nan will result in nan.
- b (float, default 0.05) – A parameter to adjust the incidence angle modifier as a function of angle of incidence. Typical values are on the order of 0.05 [3].

Returns: iam (numeric) – The incident angle modifier (IAM). Returns zero for all abs(aoi) >= 90 and for all iam values that would be less than 0.

Notes

The incidence angle modifier is calculated as

$$IAM = 1 - b(\sec(aoi) - 1)$$

As AOI approaches 90 degrees, the model yields negative values for IAM; negative IAM values are set to zero in this implementation.

References

[1] Souka A.F., Safwat H.H., "Determination of the optimum orientations for the double exposure flat-plate collector and its reflections". Solar Energy vol .10, pp 170-174. 1966.

[2] ASHRAE standard 93-77

[3] PVsyst Contextual Help. https://files.pvsyst.com/help/index.html?iam_loss.htm retrieved on October 14, 2019

See also

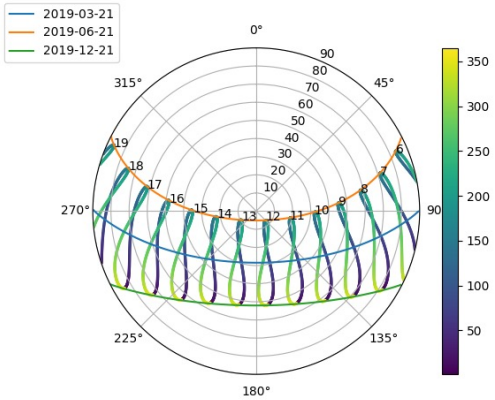
pvlib.iam.physical, pvlib.iam.martin_ruiz, pvlib.iam.interp

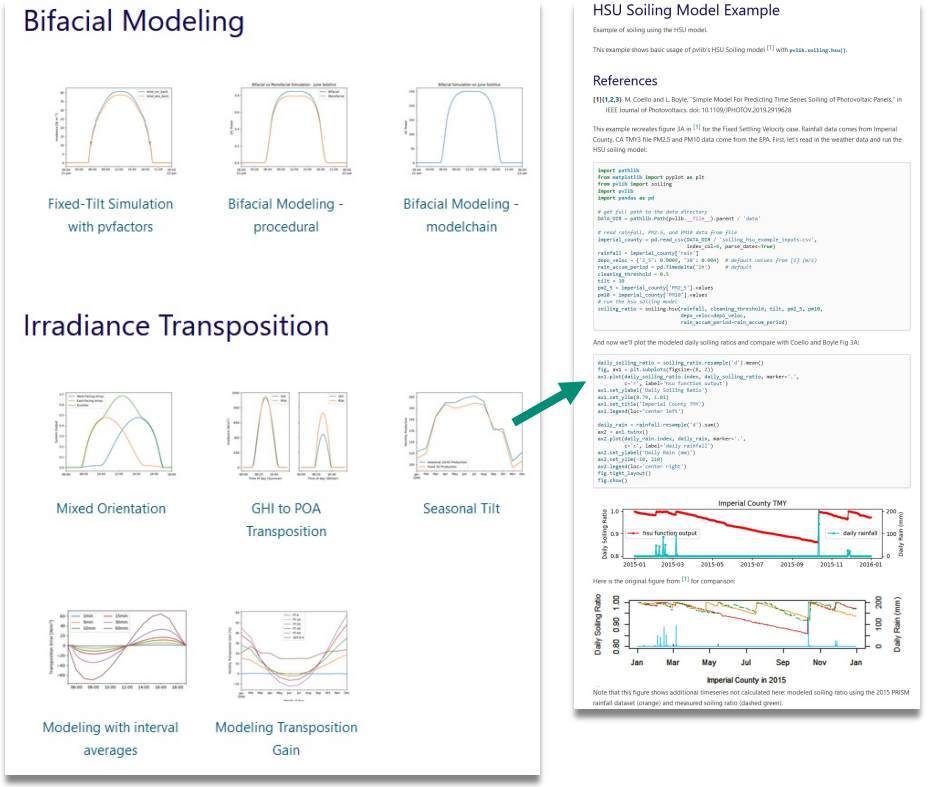
Examples using pvlib.iam.ashrae

Diffuse IAM Calculation

pvlib python Documentation: Example Gallery

pvlib “cookbook” -- small self-contained scripts for various modeling tasks, intended as a starting point for your own code.





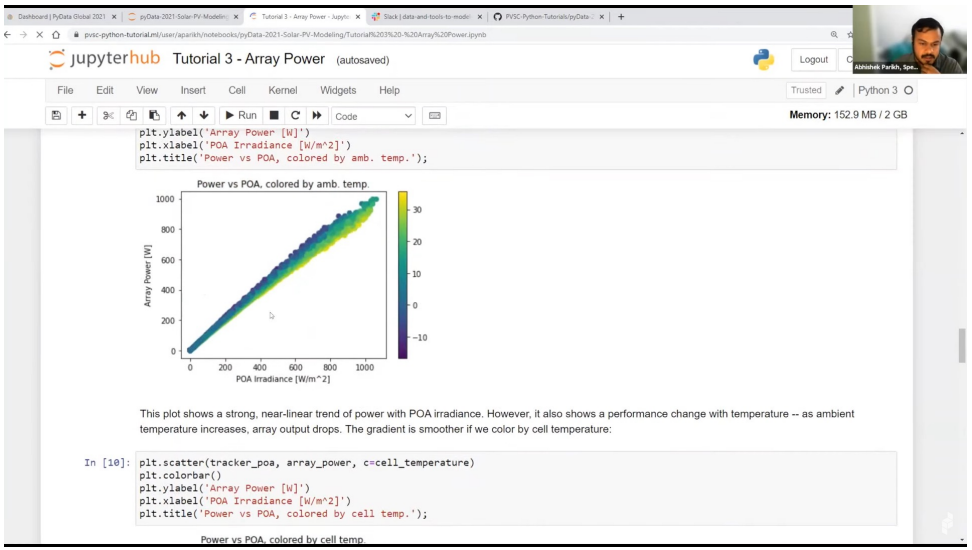
Want to make cool plots like this one?
Check out the example gallery!

https://pvlib-python.readthedocs.io/en/stable/gallery/index.html

pvlib python: Tutorials

In-person and recorded tutorials for:
- Modeling concepts
- Implementation in pvlib

The next one is here, today!



Past tutorials

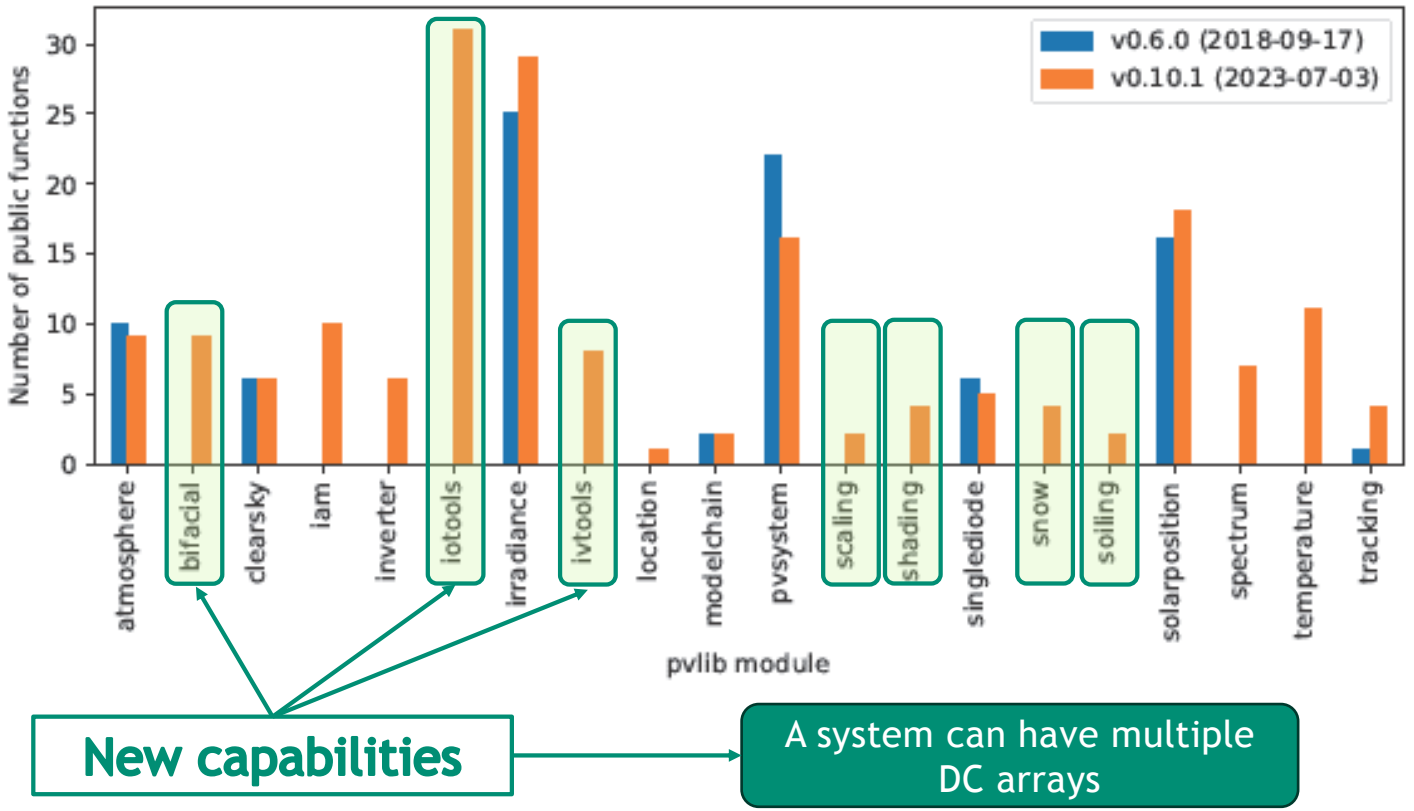
50th PVSC: https://github.com/PVSC-Python-Tutorials/PVSC50

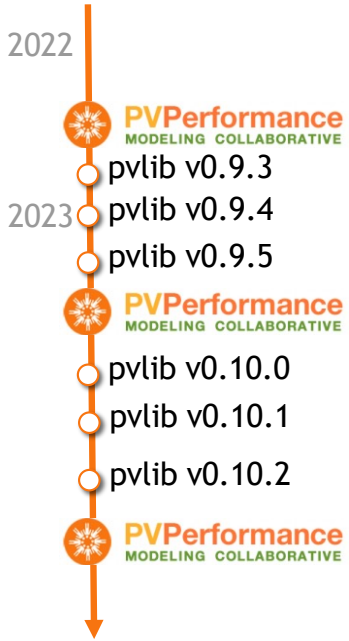
PVPMC 2022: https://github.com/PVSC-Python-Tutorials/PVPMC_2022

PVSC 2021: https://github.com/PVSC-Python-Tutorials/PVSC48-Python-Tutorial

PyData Global 2021
Youtube recording: https://www.youtube.com/watch?v=sweUakFg3I8
Source material: https://github.com/PVSC-Python-Tutorials/pyData-2021-Solar-PV-Modeling

pvlib python since v0.6 (2018)

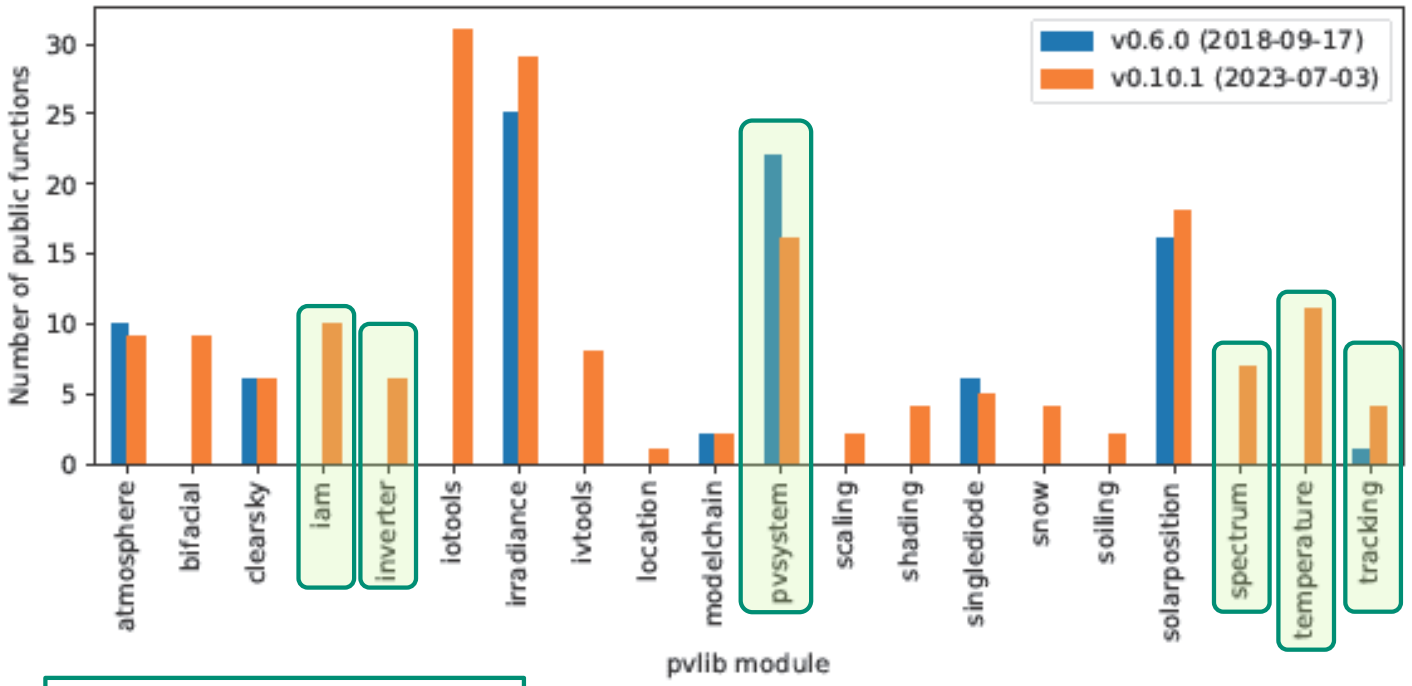


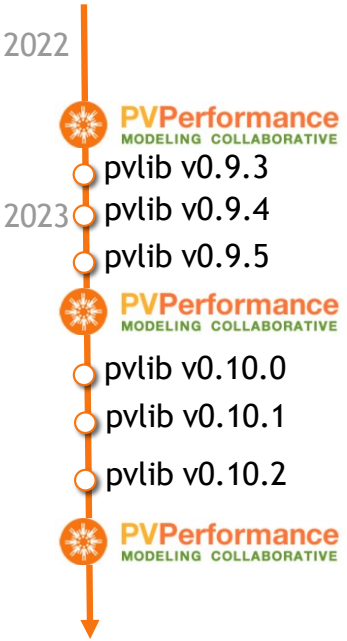


Full details: https://pvlib-python.readthedocs.io/en/stable/whatsnew.html

pvlib

pvlib python since v0.6 (2018)





Better organization

Full details: https://pvlib-python.readthedocs.io/en/stable/whatsnew.html

pvlib

pvlib python: Community Growth

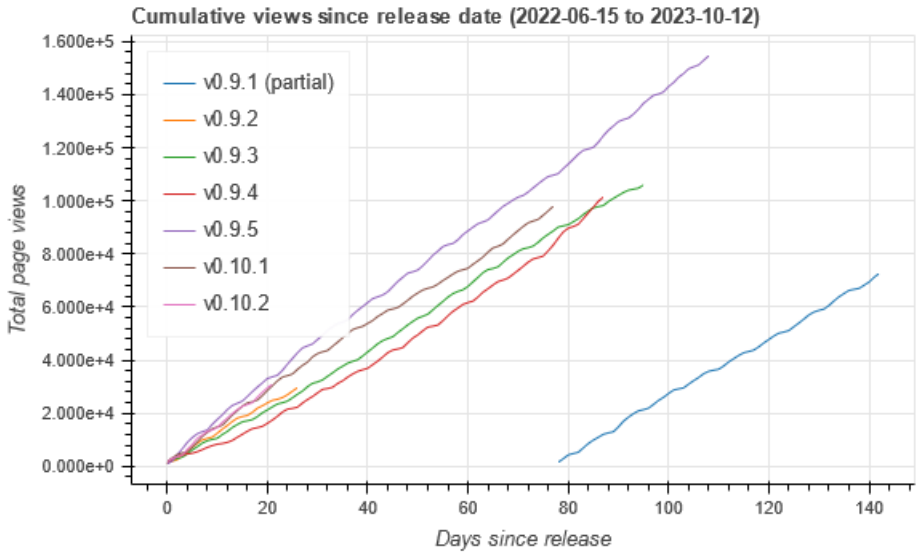
Google Group (user discussion, announcements)
- ~~600+~~ 700+ members
- https://groups.google.com/g/pvlib-python

GitHub (code development)
- Code contributions from ~~80+~~ ~~90+~~ 100+ people
- https://github.com/pvlib/pvlib-python

Citations
- 300+ since 2022
- Influence outside of PV modeling, e.g.,

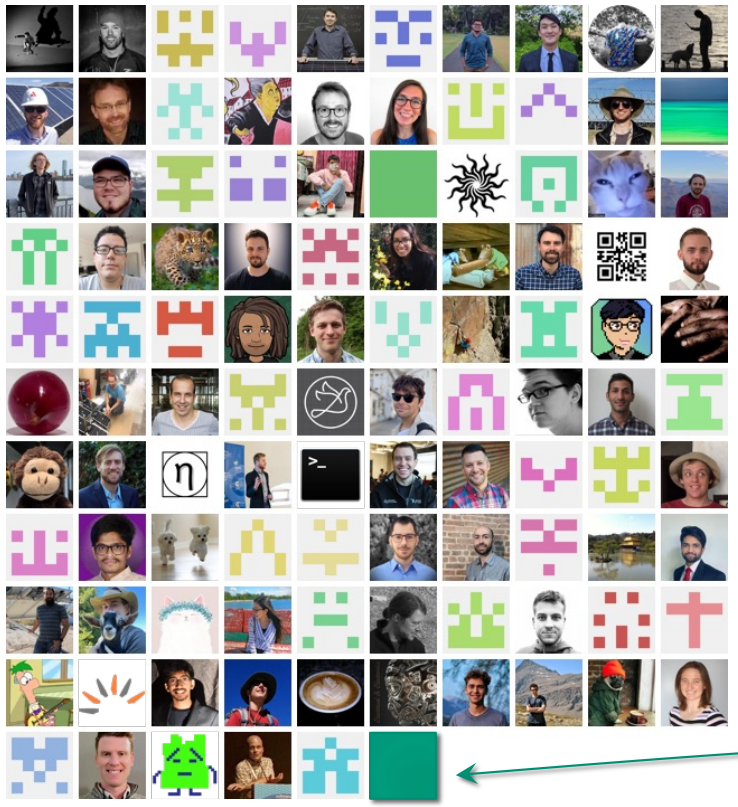
J. Rowland et al., *Scale-dependent influence of permafrost on riverbank erosion rates*. ESS Open Archive. February 09, 2023.

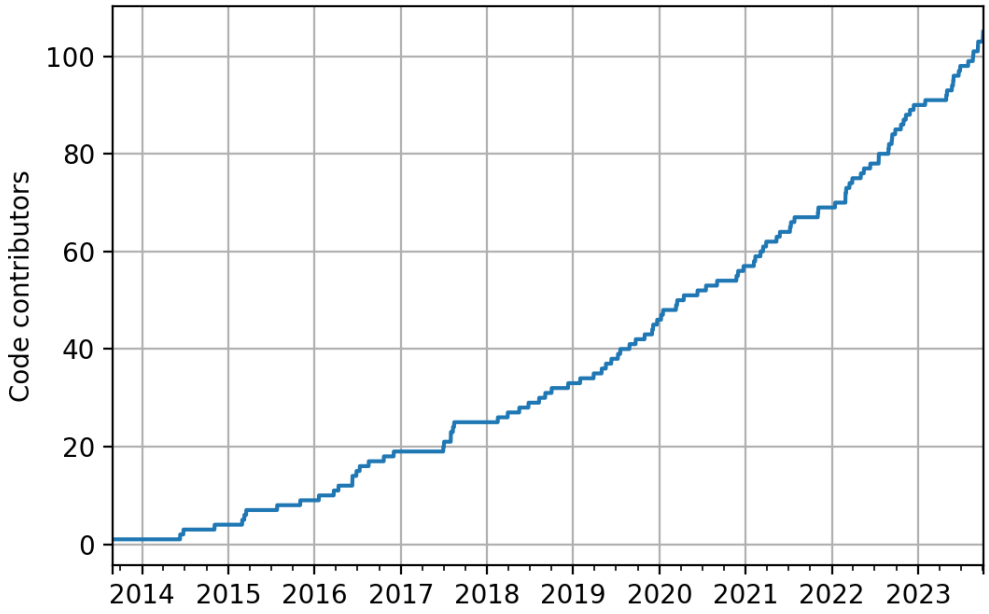
40k page views / month



pvlib python documentation page views

pvlib python: GitHub Contributors





*Not all contributions are code!

This software is made possible by contributions from people like you. You can help!

https://pvlib-python.readthedocs.io/en/stable/contributing.html

pvlib

What is pvanalytics?

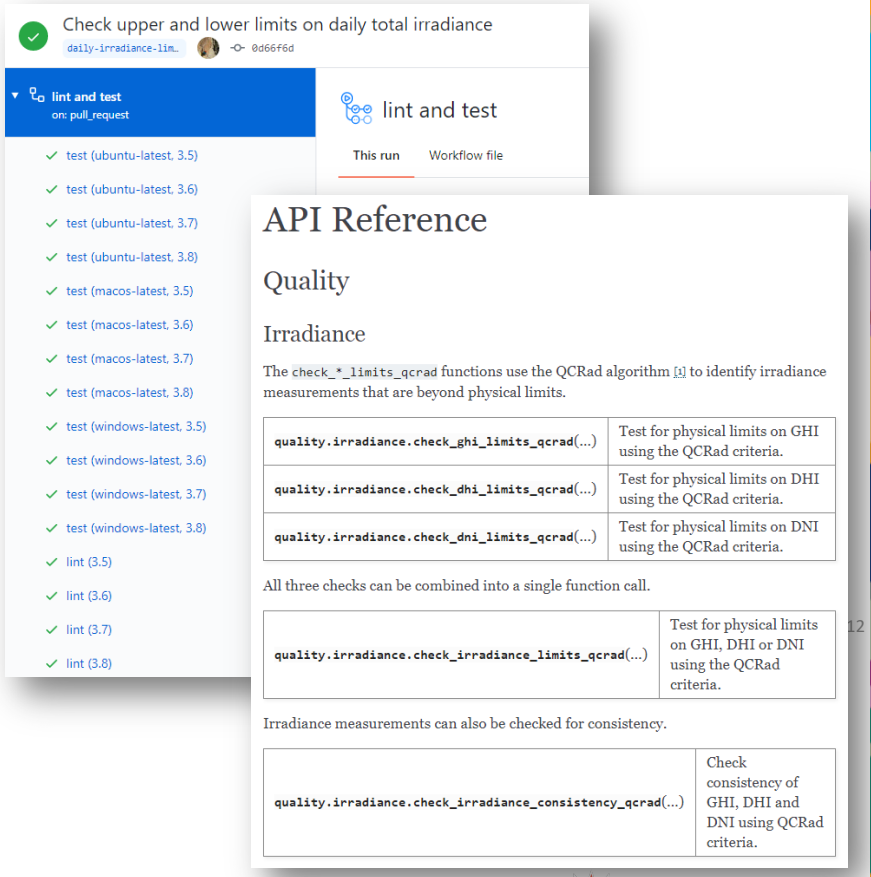
- Workflow-independent library of base functions
- Fully compatible with pvlib-python
- Launched Feb 2020, v0.1.3 Dec 2022
- 6 contributors, ~~23~~ 24 forks, ~~69~~ 76 stars

Quality control
- Plausibility of irradiance and weather measurements
- Identification of missing, interpolated, or stale data
- Outlier detection
- Identification of timestamp problems such as daylight savings shifts

Feature identification
- Inverter clipping
- Clear-sky periods
- Day/night detection from power or irradiance

Identification of system properties
- Tilt and azimuth from power data
- Differentiation between fixed and tracking PV systems

Metrics
- NREL weather corrected performance ratio

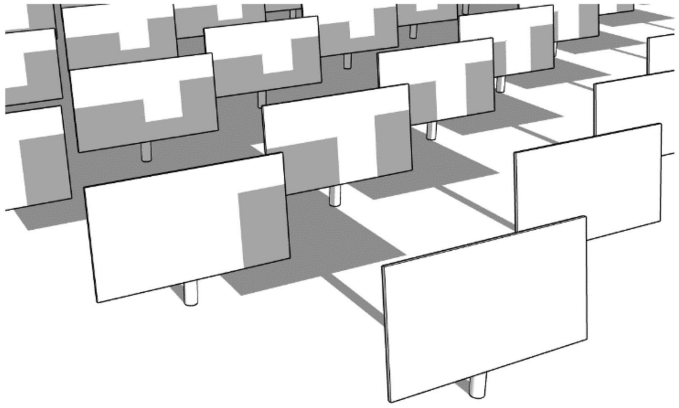


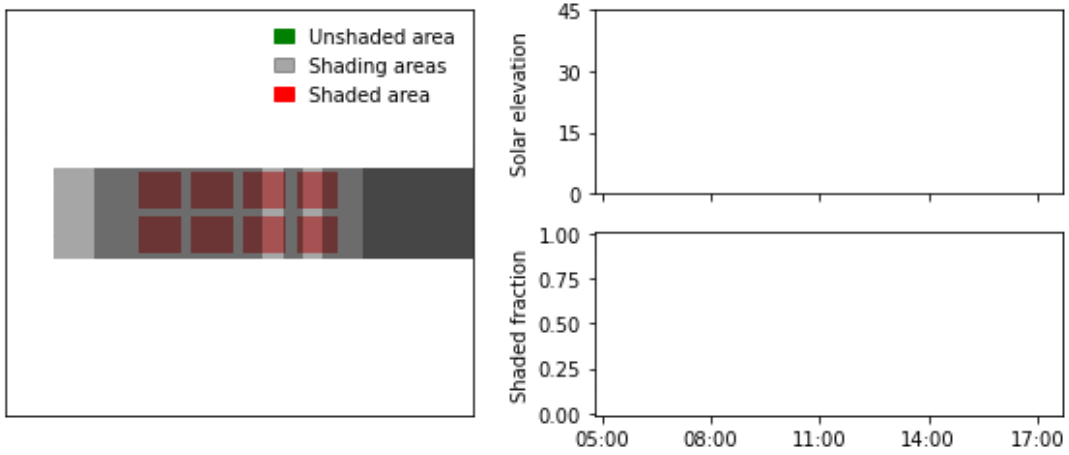
What is pvlib/twoaxistracking?

Shading of two-axis trackers

- Fully customizable field layouts
- Arbitrary panel shape
- Differentiation between active and frame area
- Extensive documentation, validated against literature

Validated against literature!





pvlib

User's group: pvlib-python, pvanalytics

pvlib-python

Cliff Hansen (Sandia)
Kevin Anderson (Sandia)
Will Holmgren (DNV)
Mark Mikofski (DNV)
Adam R. Jensen (DTU)
Anton Driesse (PV Performance Labs)

pvanalytics

Cliff Hansen (Sandia)
Will Vining (Sandia)
Kevin Anderson (Sandia)
Kirsten Perry (NREL)

2023 European PVPMC Workshop
Mendrisio, Switzerland, November 9, 2023

Objectives

1. Communicate what's planned for pvlib-python and pvanalytics
2. Identify and prioritize development ideas
3. Solicit involvement and contributions
4. Stickers!

pvlib python 1.0?

Does 1.0.0 come right after 0.10.*?

- No. Expect 0.11.0 to come next. There is no ETA for pvlib 1.0.0 yet ☺

What does 1.0 mean?

- A declaration that pvlib is no longer “beta” (whatever that means)
- Mostly, no more changes that break people’s code (until 2.0, anyway)

What needs to happen before 1.0?

- Package-wide consistency in naming (mostly there already, but still room for improvement)
- Re-organization of code modules (much has already been done)
- Fill in some modeling gaps: transformer losses, direct shading, etc.
- Rewrite/reorg the docs to follow an intentional strategy instead of the current ad-hoc “pile of info”

New features in development: pvlib-python

Additions/improvements where publications and/or data are available

- ❑ Functions for horizon shading (e.g., input digital elevation data, output angle from ground to horizon) (stalled)
- ❑ Function(s) for LCOE (started, help wanted)
- ❑ Cell, module and string electrical mismatch calculations (started, help wanted)
- ❑ Functions to interchange data with PAN/OND files (started, help needed)

Additions where new publications and/or data may be needed

- ❑ Functions to translate parameters among models:
  - ✓ temperature models
  - ✓ incidence angle modifier (IAM) models
  - ❑ IV curve models
- ❑ Functions to fit models to data: temperature, IV curve and other power models

New features requested: pvlib-python

Additions/improvements where publications and/or data are available

- Degradation of DC components

Additions where new publications and/or data may be needed

- Better models for inverters:
  - Off-unity power factor, temperature derating, MPPT voltage limits, current limits, "smarter clipping"
- Models for DC optimizers, AC transformer losses

What else?

- Features with available references?
- Features needing research before implementation in pvlib?

Improving the user experience: pvlib-python

Overhaul the documentation

- The “middle” layer between the home page for pvlib-python and each function’s document page

Library of data for model benchmarking and comparison

- Non-expert users can be challenged when faced with e.g. seven clear-sky models
- Provide some guide to assist in selecting appropriate models
- Illustrate model accuracy and limitations

More examples (good progress here)

- We want to have examples showing how to use most major features

✓ **More frequent releases (quarterly)**

Quarterly community meetings (contact cwhanse@sandia.gov)

pvlib

What are your ideas?

pvanalytics

www.github.com/pvlib/pvanalytics.git

Welcoming contributors, developers, maintainers

Project will mature through contact with users

Guiding principles:

- Workflow-independent library of documented functions
  - Many functions have a common signature: input time series, output Boolean series
  - E.g., clear sky labeling, clipped inverters, GHI passes quality checks
- Workflows are higher level objects that combine sequences of functions for common use cases, e.g. performance ratio calculation
  - Prioritize flexibility
  - Classes could help automate workflows
  - Example workflows should help to clarify definitions

Thank You

www.github.com/pvlib/pvlib-python
https://pvlib-python.readthedocs.io

www.github.com/pvlib/pvanalytics
https://pvanalytics.readthedocs.io

SAND 2023-XXXXC

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.