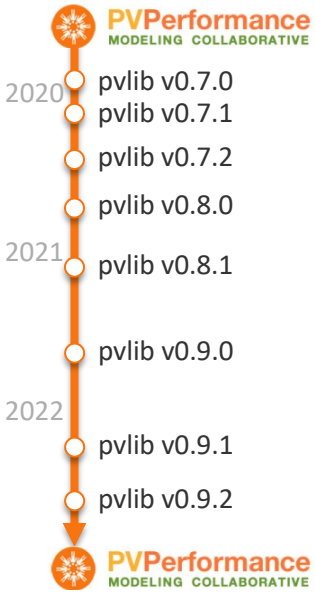


pvlib python 2022 update

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

PV Performance Modeling and Monitoring Workshop
Salt Lake City, Aug 23, 2022

Contents

What is pvlib?

A python library for PV performance modeling that is **community-driven**, **free**, **open-source**, and **well-documented**

Modeling Toolbox

Stand-alone models for:

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

…and more!

System Energy Yield

Weather-to-power following the PVPMC workflow

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

Completely 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

Online documentation: https://pvlib-python.readthedocs.io

New models (v0.7.0 – v0.9.2)

pvlib.bifacial
- infinite_sheds

pvlib.iam
- martin_ruiz
- martin_ruiz_diffuse
- marion_diffuse

pvlib.temperature
- faiman
- fuentes
- ross
- noct_sam
- prilliman transient model

pvlib.snow
- Marion model

pvlib.soiling
- Kimber model
- Humboldt State model

pvlib.shading
- sky_diffuse_passias

pvlib.inverter
- fit_sandia
- sandia_multi
- pvwatts_multi

pvlib.ivtools
- fit_sde_sandia
- fit_sdm_cec_sam
- fit_sdm_desoto
- fit_pvsyst_sandia
- fit_desoto_sandia

pvlib.spectrum
- spectrl2

pvlib.scaling
- Wavelet variability model

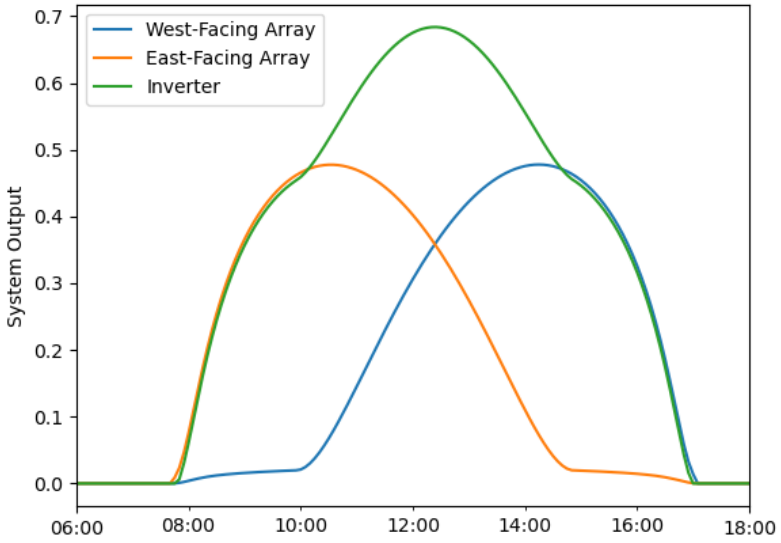
pvlib.tracking
- slope-aware backtracking

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

ModelChain: Multi-Array Systems (v0.9.0)

Major refactor for non-homogeneous systems, e.g.:

- **Different subarray orientations**:
  - Multiple orientations (e.g. rooftop)
  - Part tracking, part fixed-tilt
- **Different subarray electrical properties**:
  - e.g. part 300W, part 305W
- **New classes**:
  - Array
  - FixedMount, SingleAxisTrackerMount



Thanks to Will Vining!
https://github.com/wfvining

Google Summer of Code 2021: pvlib.iotools

GSoC project adding data import for:

- **BSRN**: global ground station network (WRMC)
- **CAMS**: satellite-based irradiance
- **McClear**: satellite-based clear-sky irradiance
- **PVGIS**: various satellite/reanalysis data

- **MERRA2**: reanalysis weather (NASA)
- **ERA5**: reanalysis weather (ECMWF)

Thanks to Adam Jensen!
https://github.com/adamrjensen/

Google Summer of Code

- Participants work with an open-source organization on a 10 week programming project
- Good fit for students, but any open-source beginner can apply
- https://summerofcode.withgoogle.com/
- Get in touch for 2023!

Other Code Changes (v0.7.0 – v0.9.2)

- pvlib.forecast was deprecated in v0.9.1
  - To be removed altogether in some future release

- Partial model reorg by modeling topic
  - pvlib.iam, pvlib.inverter, etc

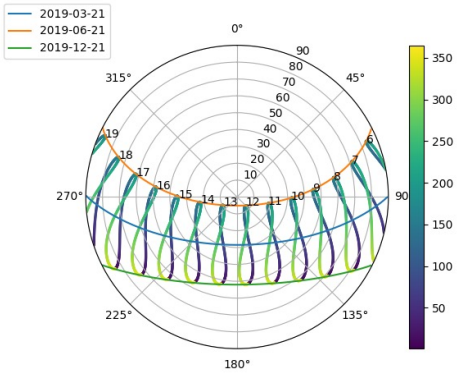
- Consistency improvements, especially in pvlib.iotools

… and many other enhancements and bug fixes

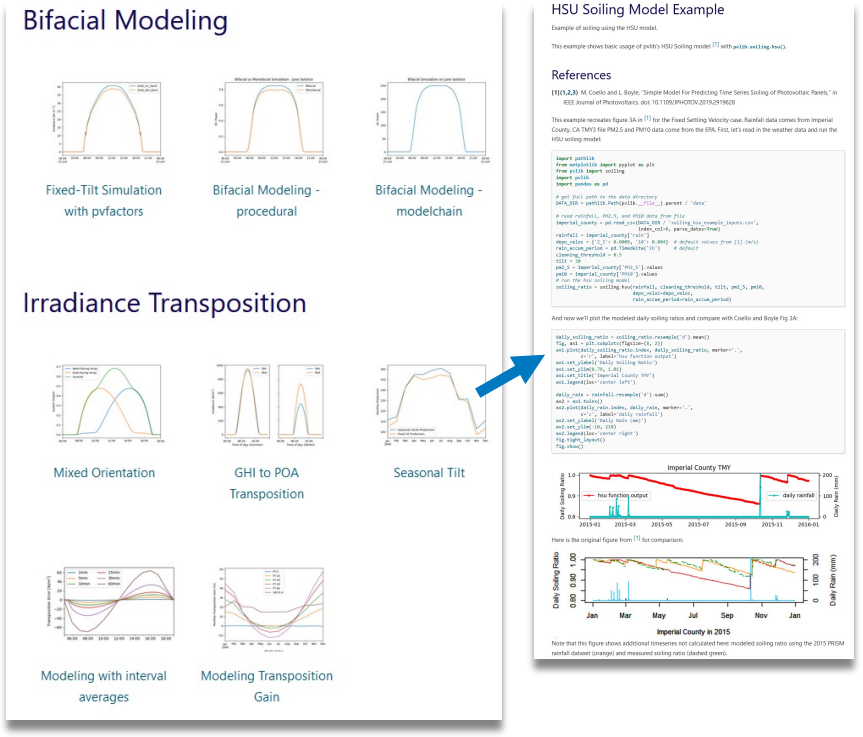
Full details: https://pvlib-python.readthedocs.io/en/stable/whatsnew.html

Documentation: Example Gallery

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



Wanna make cool plots like this one?
Go check out the example gallery!



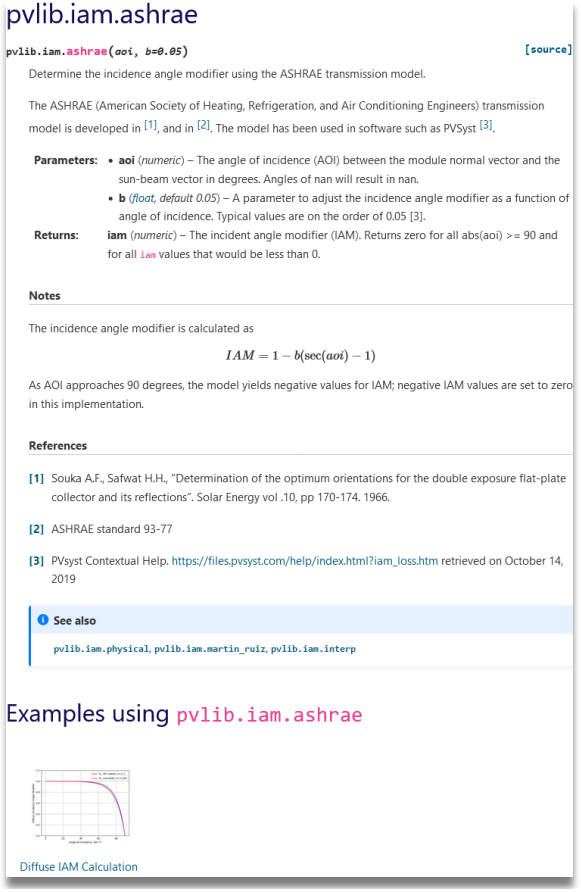
https://pvlib-python.readthedocs.io/en/stable/gallery/index.html

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!



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

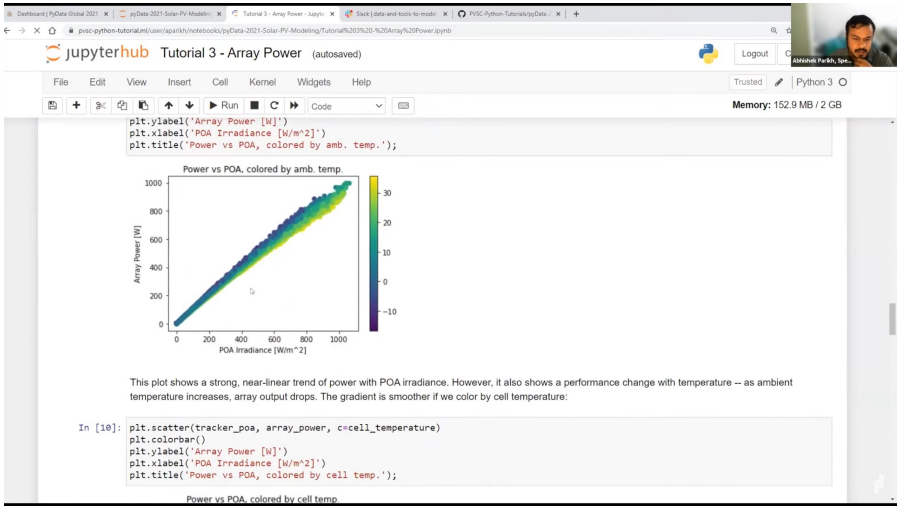
Examples using pvlib.iam.ashrae

Diffuse IAM Calculation

Documentation: Tutorials

Interactive tutorials for PV system performance:
- Modeling concepts
- Implementation in pvlib
- Previous tutorials at PVSC and PyData Global

The next one is here, tomorrow afternoon!
Led by Silvana Ovaitt, don't miss it!



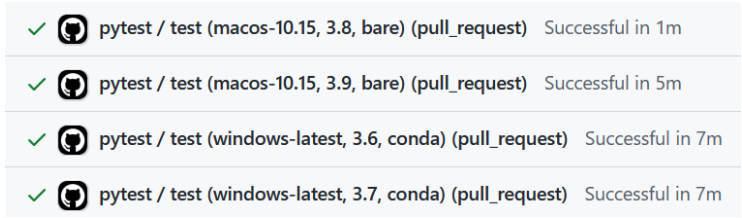
PVSC 2021
Source material: https://github.com/PVSC-Python-Tutorials/PVSC48-Python-Tutorial

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

Automated Testing

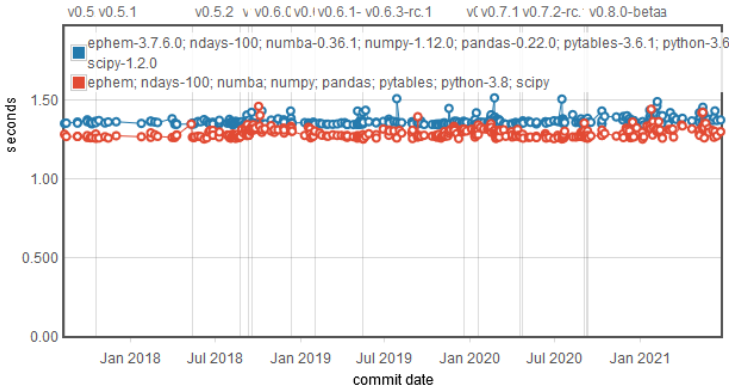
Correctness Testing

- Comprehensive coverage of the entire package
- Essential to avoid accidentally introducing bugs in development
- pvlib has >10k LOC for testing alone



Speed Benchmarks

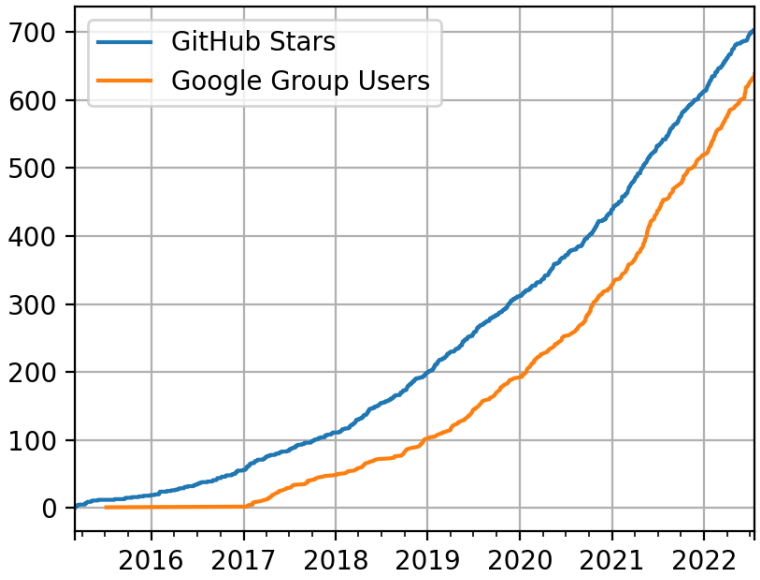
- Monitor execution speed of test cases to detect performance regressions over time
- Uses airspeed velocity (asv): https://asv.readthedocs.io
- Benchmark Results:
  - https://pvlib-benchmarker.github.io/pvlib-benchmarks/



Community Growth

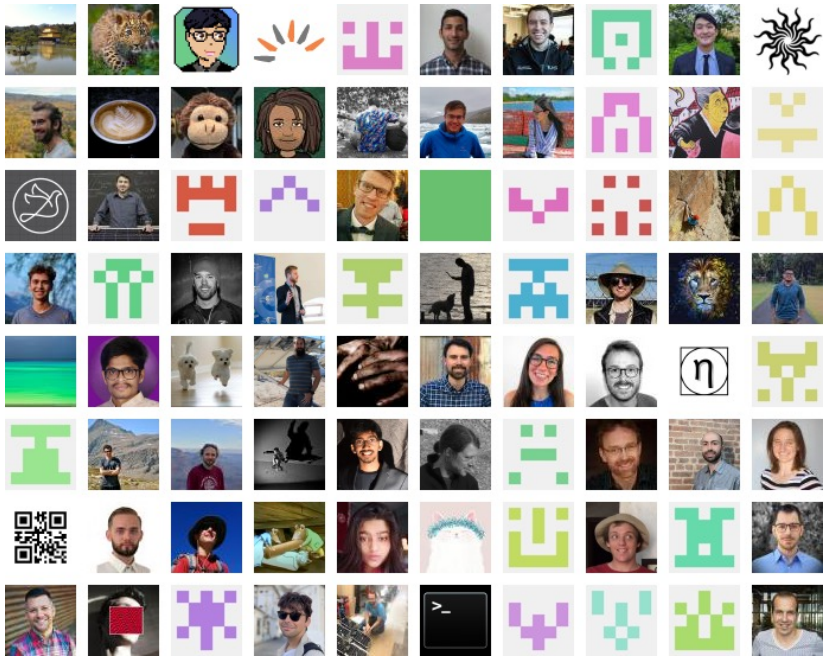
Google Group (user discussion, announcements)
- 650+ members
- Roughly quadrupled since 2019 workshop
- https://groups.google.com/g/pvlib-python

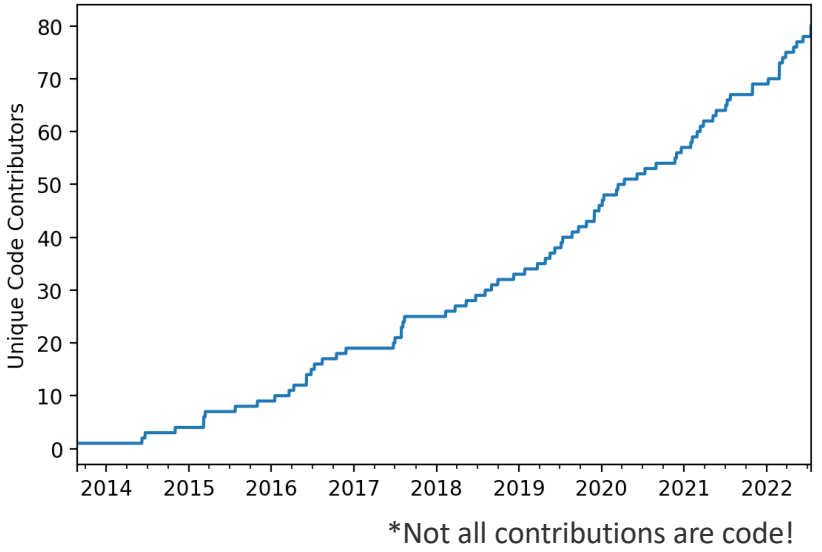
GitHub (code development)
- 700+ pull requests
- Code contributions from 80+ people
- https://github.com/pvlib/pvlib-python



Numbers as of 2022-07-20

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 1.0?

Does 1.0.0 come right after 0.9.*?

- No. Expect 0.10.0 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?

- Change all the things we want to change:
  - Package-wide consistency in naming (mostly there already, but still room for improvement)
- 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”
- **What else? We’d love to get your feedback!** Come to the pvlib user discussion tomorrow, 3pm

Thank You

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

NREL/PR-5K00-83420

This work was authored in part by Alliance for Sustainable Energy, LLC, the manager and operator of the National Renewable Energy Laboratory for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under Solar Energy Technologies Office (SETO) Agreement Numbers 34348. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.