

PVDeg: Python tools to calculate photovoltaic degradation responses and parameters

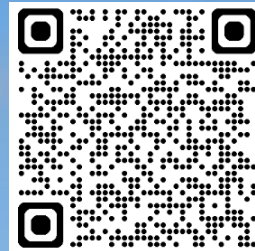
Rajiv Daxini*, Martin Springer

**Rajiv.Daxini@nlr.gov*

BSD-3-Clause

pip install pvdeg

2nd IESO Open-Source Software Summit 2026



PVDeg GitHub



Daxini et al., "PVDeg: a Python package for modeling degradation on solar photovoltaic systems." *Journal of Open Source Software* (under review)

Why model PV Degradation?

As PV deployment expands into more demanding environments, degradation threatens module lifetime and performance.

Modeling is critical for:

- **Anticipating performance losses** before they happen
- **Guiding material selection** for durability
- **Enabling proactive maintenance** strategies
- **Extending operational lifetime** across diverse climates

The problem

No open-source software combines **physics-based degradation** with **uncertainty quantification** and **geospatial scaling** — until PVDeg.

Who it's for

PV reliability researchers, materials scientists, and engineers — in the **laboratory** and the **field**.

State of the field

Existing tools

- **pvlib-python, SAM** — simulate energy yield, **not degradation**
- **PVAnalytics, SolarDataTools** — analyze performance data
- **RdTools, PVplr** — extract degradation **rates** from field data

The limitation

- These are **top-down**: performance is measured, rates are extracted — but the **cause of degradation is often unclear**.

PVDeg: bottom-up approach

Start with the **known physics** of a degradation mechanism → evaluate its **impact on system performance**.



Many PV
performance tools!

Software design

① Core functions

- Functions: **degradation mechanisms, material properties**, and **environmental stressors**
- Input: **standardized weather DataFrame** + **location metadata**
- Return: **timeseries** or a **summary number**

Examples: `humidity.module()`
(moisture ingress),
`letid.calc_letid_outdoors()`
(LeTID).

② Scenario class

- Wraps core functions into a reproducible **pipeline** that defines **locations, module configurations**, and **degradation mechanisms**
- Serves as an interface for multiparameter and multiphysics studies
- Compare climates, configurations, etc.
- Complex studies made **user-friendly** and **reproducible**

③ Geospatial

- Distribute degradation calculations across **geographic regions** with **parallel processing**
- Integrates public (**NSRDB, PVGIS**) climate data
- Samples **latitude-longitude grids** (uniform or stochastic)
- Built-in mapping **visualizations**

④ Monte Carlo

- Propagates uncertainty for laboratory-to-field extrapolation
- Samples parameter **distributions** and their **correlations**
- Produces **confidence intervals**, not single values
- Identifies which parameters **most affect reliability risk**

Method described in Springer et al. (2022), Future-proofing.

Software design

① Core functions

- Functions: **degradation mechanisms**, **material properties**, and **environmental stressors**
- Input: **standardized weather DataFrame** + **location metadata**
- Return: **timeseries** or a **summary number**

Examples: `humidity.module()`
(moisture ingress),
`letid.calc_letid_outdoors()`
(LeTID).

② Scenario class

- Wraps core functions into a reproducible **pipeline** that defines **locations**, **module configurations**, and **degradation mechanisms**
- Serves as an interface for multiparameter and multiphysics studies
- Compare climates, configurations, etc.
- Complex studies made **user-friendly** and **reproducible**

③ Geospatial

- Distribute degradation calculations across **geographic regions** with **parallel processing**
- Integrates public (**NSRDB**, **PVGIS**) climate data
- Samples **latitude-longitude grids** (uniform or stochastic)
- Built-in mapping **visualizations**

④ Monte Carlo

- Propagates uncertainty for laboratory-to-field extrapolation
- Samples parameter **distributions** and their **correlations**
- Produces **confidence intervals**, not single values
- Identifies which parameters **most affect reliability risk**

Method described in Springer et al. (2022), Future-proofing.

Software design

① Core functions

- Functions: **degradation mechanisms**, **material properties**, and **environmental stressors**
- Input: **standardized weather DataFrame** + **location metadata**
- Return: **timeseries** or a **summary number**

Examples: `humidity.module()`
(moisture ingress),
`letid.calc_letid_outdoors()`
(LeTID).

② Scenario class

- Wraps core functions into a reproducible **pipeline** that defines **locations**, **module configurations**, and **degradation mechanisms**
- Serves as an interface for multiparameter and multiphysics studies
- Compare climates, configurations, etc.
- Complex studies made **user-friendly** and **reproducible**

③ Geospatial

- Distribute degradation calculations across **geographic regions** with **parallel processing**
- Integrates public (**NSRDB**, **PVGIS**) climate data
- Samples **latitude-longitude grids** (uniform or stochastic)
- Built-in mapping **visualizations**

④ Monte Carlo

- Propagates uncertainty for laboratory-to-field extrapolation
- Samples parameter **distributions** and their **correlations**
- Produces **confidence intervals**, not single values
- Identifies which parameters **most affect reliability risk**

Method described in Springer et al. (2022), Future-proofing.

Software design

① Core functions

- Functions: **degradation mechanisms**, **material properties**, and **environmental stressors**
- Input: **standardized weather DataFrame** + **location metadata**
- Return: **timeseries** or a **summary number**

Examples: `humidity.module()`
(moisture ingress),
`letid.calc_letid_outdoors()`
(LeTID).

② Scenario class

- Wraps core functions into a reproducible **pipeline** that defines **locations**, **module configurations**, and **degradation mechanisms**
- Serves as an interface for multiparameter and multiphysics studies
- Compare climates, configurations, etc.
- Complex studies made **user-friendly** and **reproducible**

③ Geospatial

- Distribute degradation calculations across **geographic regions** with **parallel processing**
- Integrates public (**NSRDB**, **PVGIS**) climate data
- Samples **latitude-longitude grids** (uniform or stochastic)
- Built-in mapping **visualizations**

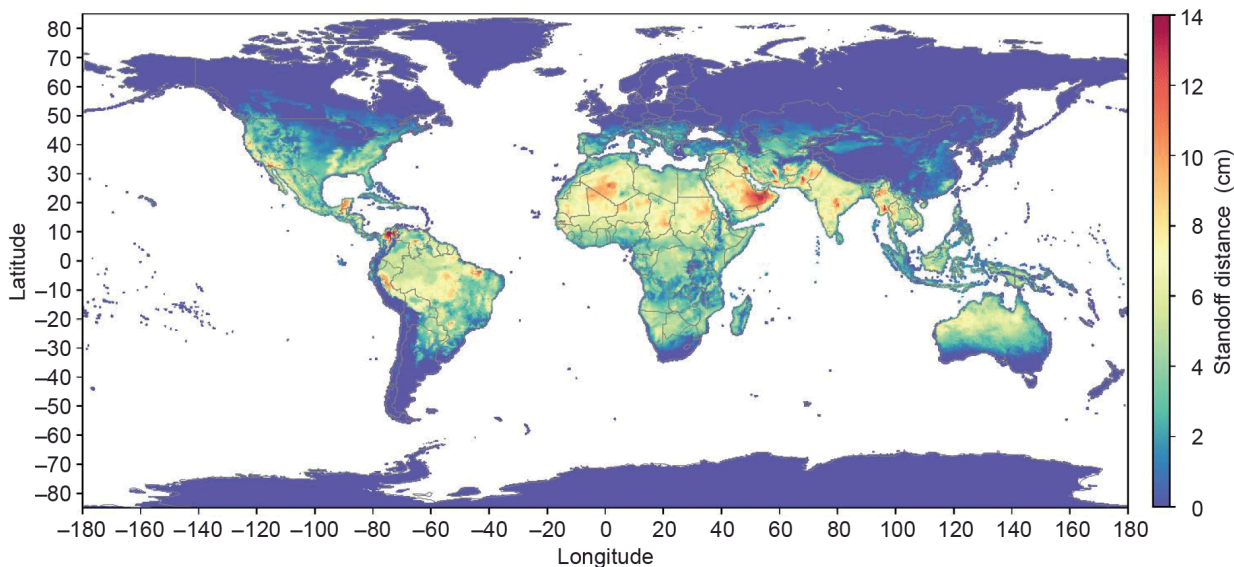
④ Monte Carlo

- Propagates uncertainty for laboratory-to-field extrapolation
- Samples parameter **distributions** and their **correlations**
- Produces **confidence intervals**, not single values
- Identifies which parameters **most affect reliability risk**

Method described in Springer et al. (2022), Future-proofing.

Case study: IEC TS 63126 standoff

- Optimal PV panel standoff distance (module underside to roof) — critical for cooling, fire safety, and clearance.
- PVDeg supports IEC TS 63126 standardization through global standoff maps.



Open datasets and tutorials

Community-driven open datasets

- **Degradation** parameters and **kinetic** coefficients
- **UV-albedo** data
- **Permeation** properties (H_2O , O_2 , acetic acid)
- Query through the **core API**
- Continuously **expanded** — contributions welcome

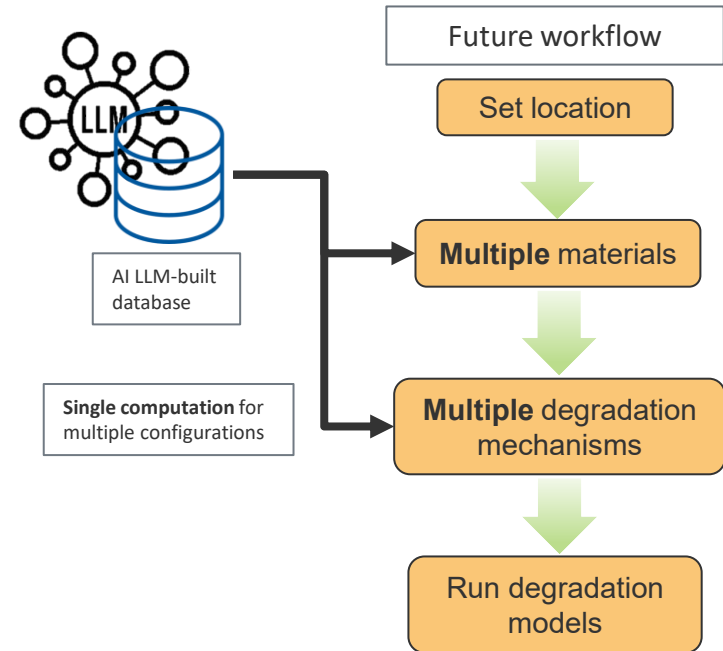
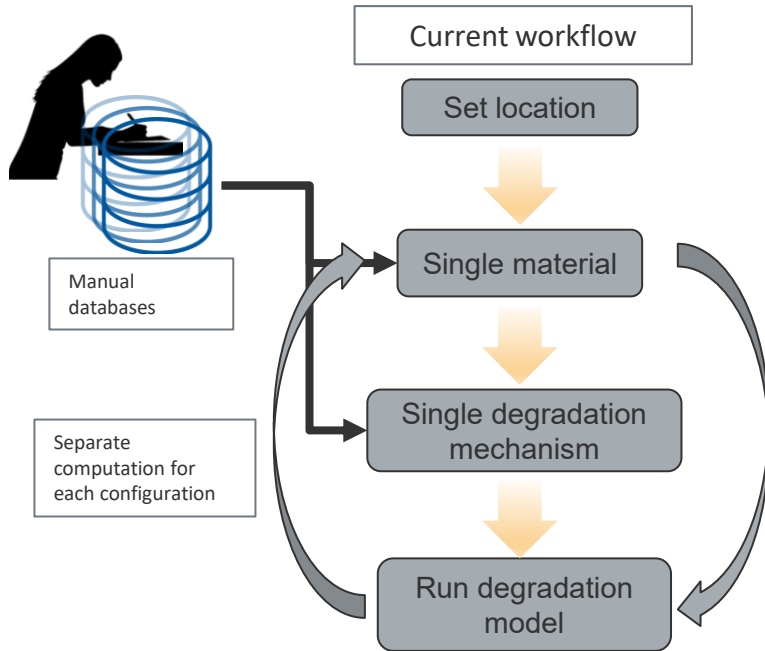
Tutorials & tools

- Comprehensive suite of **Jupyter notebooks**
- Core mechanisms · Scenario · geospatial · uncertainty
- **Interactive** and **reproducible**
- Integrates real datasets (**NSRDB**, **PVGIS**)

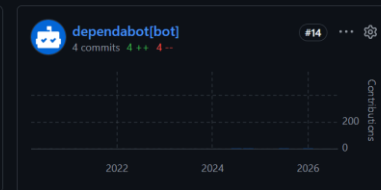
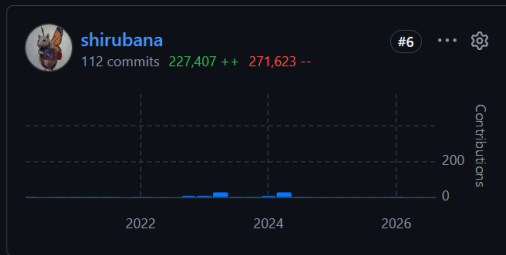
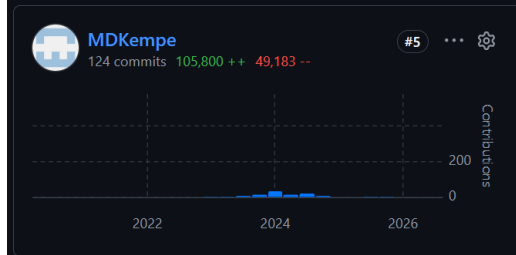
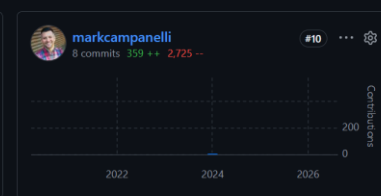
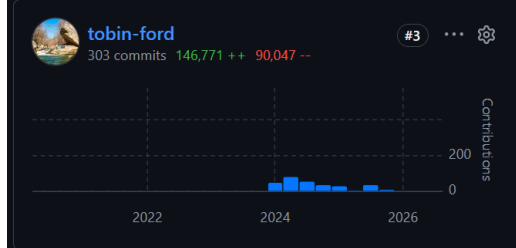
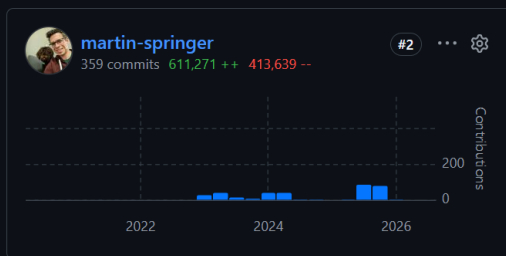
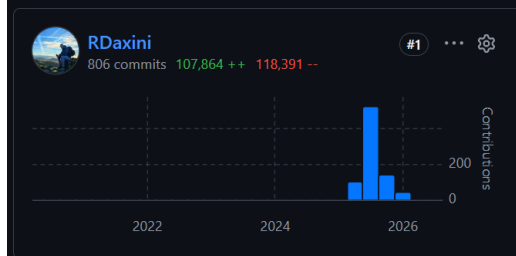
PVDeg is a **reproducible, field-relevant** platform, and a growing shared knowledge base for the community.

PVDeg: what's next?

v0.8.0 will deliver enhanced multi-job chaining for sequential analysis in the Scenario class



We thank all contributors!



Summary

PVDeg: the missing link between materials science and field performance



PVDeg

Bottom-up PV
degradation
modeling



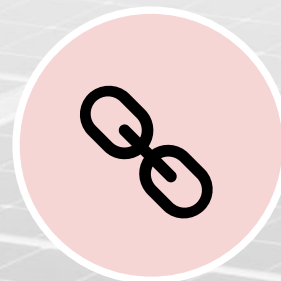
Open datasets

Transparent and
reproducible
computation



Tutorials

Step-by-step Jupyter
notebook tutorials
for accessibility



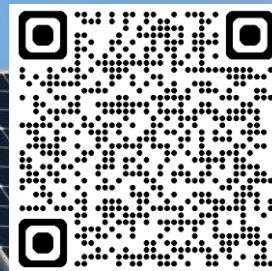
Chain analysis

Sequential pipelines
chaining together
calculations



AI

LLM-driven datasets



This work was authored in part by the National Laboratory of the Rockies for the U.S. Department of Energy (DOE), operated under Contract No. DE-AC36-08GO28308. Lawrence Berkeley National Laboratory is funded by the DOE under award DE-AC02-05CH11231. Funding provided as part of the Durable Modules Consortium (DuraMAT), an Energy Materials Network Consortium funded by the U.S. Department of Energy, Office of Critical Minerals and Energy Innovation, Solar Energy Technologies Office. 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.

Thank you

Rajiv.Daxini@nlr.gov

NLR/PR-5900-101585