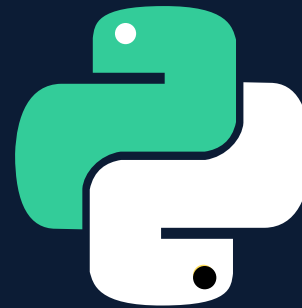




```
pip  
install  
pvradar
```

New python library for soiling modelling

2024 European PVPMC, Copenhagen

Thore Müller, PVRADAR Labs GmbH

PVRADAR TEAM

PV ENGINEERING MEETS SOFTWARE DEVELOPMENT



THORE MÜLLER
CEO

7 years in soiling
measurement and
cleaning optimization



KONSTANTIN POGORELOV
CTO

20 years in software
development and team-
leadership



FRANCO CLANDESTINO
Head of Product

5 years as specialist in
yield estimation and
bifacial PV

Trusted by:



Our partners:

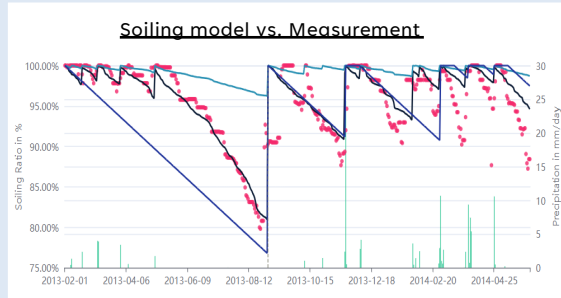


METHODOLOGY

Soiling modelling

Accurate soiling predictions

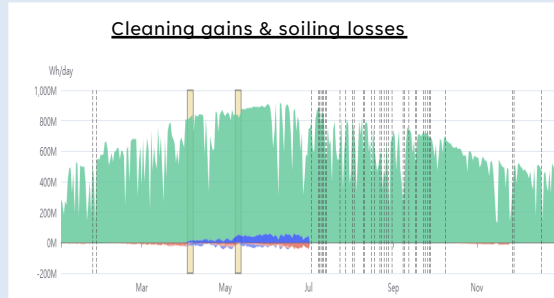
- Based on 20 years of historic data: satellite data and meteorological measurements
- Location-specific model parameters
- Considering plant technical parameters



Yield estimation

Energy gains and losses

- Quantification of energy losses and cleaning gains
- Integrated yield estimation model
- Accounting for clipping and curtailment losses



Cashflow optimization

Lifetime cost estimation

- Evaluation based on lifetime cost of ownership
- Maximizing energy production while limiting cleaning expenses
- Compare costs and benefits of different cleaning systems



OUR SOLUTION: READY-TO-USE LICENSES, TAILOR-MADE SOFTWARE SOLUTIONS AND EXPERT CONSULTING

Web Application

For project developers

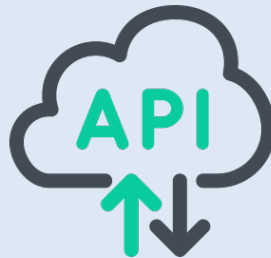
- Start a new project by clicking on a map
- Optimize cleaning schedules for tractor + brush, semi- and full- autonomous robots – dry and wet
- Create convincing PDF reports for project documentation



Soiling API

For asset managers

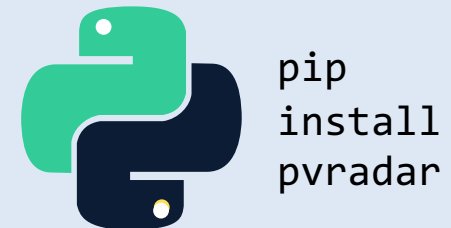
- Get monthly soiling loss factors: P50, P90 and P90 for any location worldwide
[\[Integration with Terabase PlantPredict\]](#)
- Soiling predictions and cleaning alarms: short-term (14 days) and medium term (90 days)
[\[Coming soon\]](#)
- Get input data: precipitation, PM2.5, PM10 and more



Python Library

For researchers

- Build your own models and compare to pvradar and pvlib
- Validate or fit parameters against measurement data
- Share models & results within your organization



ESTIMATE SOILING LOSSES? ... AND FIND AN OPTIMAL CLEANING STRATEGY ...

1. Build a model

tasks:

- read literature,
- define model logic

our solution:

You are left with
the fun part!

2. Execute model

- get data (multiple sources)
- homogenize data (same resolution & time-zone, matching units)
- deal with outliers, nan values, missing values

data depot

3. Validate model

- get soiling measurement data
- compare model vs. meas.
- fit model parameters
- plot results

measurement DB
python library

BUILD A SOILING MODEL

```
1 def my_soiling_function(  
2     rainfall: pd.Series,  
3     soiling_rate: float = 0.001,  
4     rain_cleaning_threshold: float = 1,  
5     initial_soiling_level: float = 0  
6 ) -> pd.Series:  
7  
8     index = rainfall.index  
9     soiling_levels = pd.Series(index=index, dtype=float)  
10  
11     soiling_levels[index[0]] = initial_soiling_level  
12  
13     for i_day in range(1, len(index)):  
14         current_date = index[i_day]  
15         previous_date = index[i_day - 1]  
16  
17         soiling_levels[current_date] = soiling_levels[previous_date] + soiling_rate  
18  
19         if rainfall[previous_date] > rain_cleaning_threshold:  
20             soiling_levels[current_date] = 0  
21  
22     return soiling_levels
```

Inputs: Rainfall as series
+ 3 parameters

1. Initialize soiling_levels

2. Iterating over every day in the index

3. Increase soiling level by rate

4. Reset soiling level if precipitation
exceeds threshold

✓ 0.0s

Python

LET'S EVALUATE THE MODEL

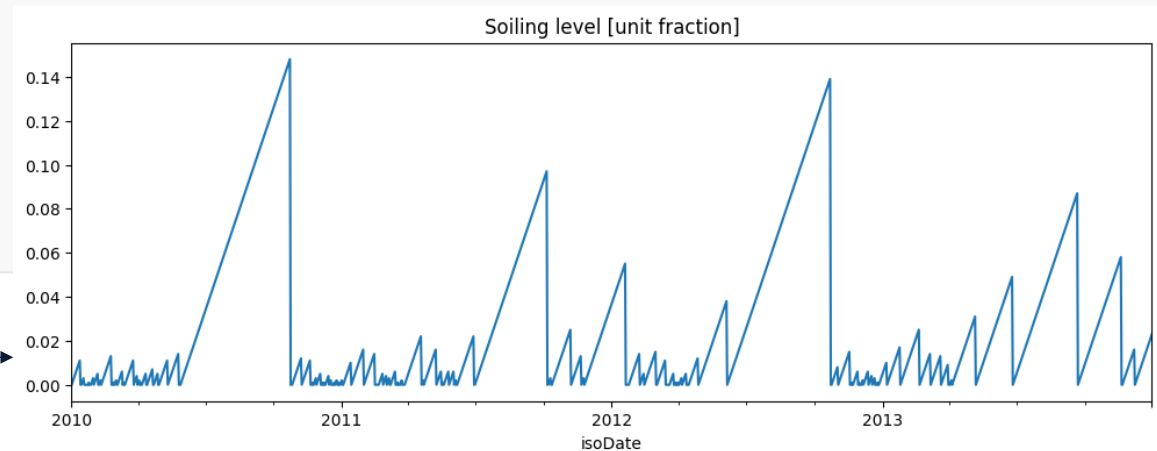
Option 1: Explicit Call

```
1 my_rainfall_series: pd.Series
2 my_soiling_level = my_soiling_function(my_rainfall_series)
3 my_soiling_level = my_soiling_function(my_rainfall_series, soiling_rate=0.001, rain_cleaning_threshold=1)
```

Python

Option 2: With modelling context - SolarSite

```
1 site = SolarSite(
2     location=Location(39.137813, -121.941262),
3     params={
4         "mountingType": "fixed",
5         "tilt": 30
6     },
7     period = Period.year_range(2010, 2013)
8 )
9
10 my_soiling_level = site.calculate(my_soiling_function)
```



WHERE DOES THE RAINFALL COME FROM?

```
1 site.models
```

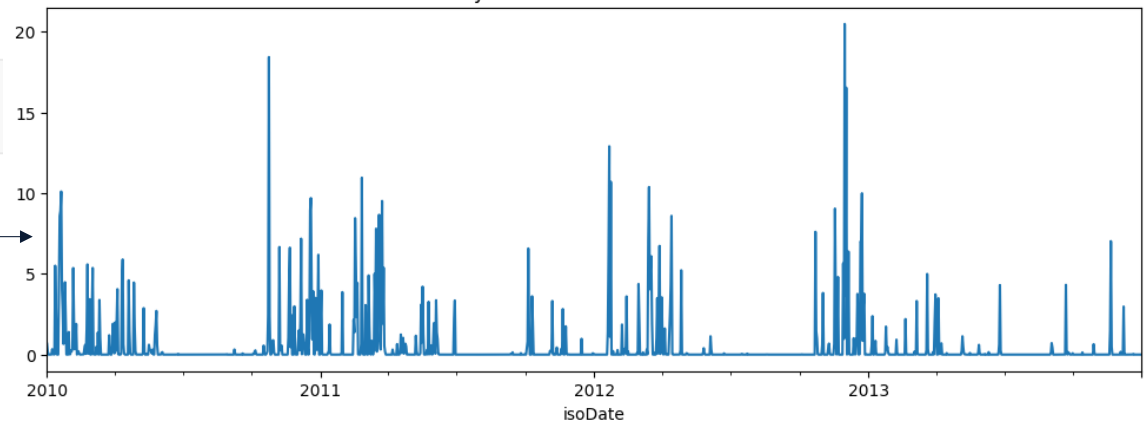
✓ 0.0s

Python

```
{'rainfall': <Total daily rainfall in mm. Source: MERRA2>,  
'soiling_level': <Avg. daily soiling level in unit fraction. Source: PVRADAR>,  
'pm2_5': <Avg. daily concentration of particulate matter with less than 2.5 µm in g/m³ air. Source: MERRA2>,  
'pm10': <Avg. daily concentration of particulate matter with less than 10 µm in g/m³ air. Source: MERRA2>}
```

```
1 rainfall_merra = site.calculate(model='rainfall')
```

Total daily rainfall in mm. Source: MERRA2



```
1 rainfall_merra = site.calculate(model='rainfall', freq='W', agg='avg', tz='UTC-3')
```

Python

REGISTERING PVLIB MODELS

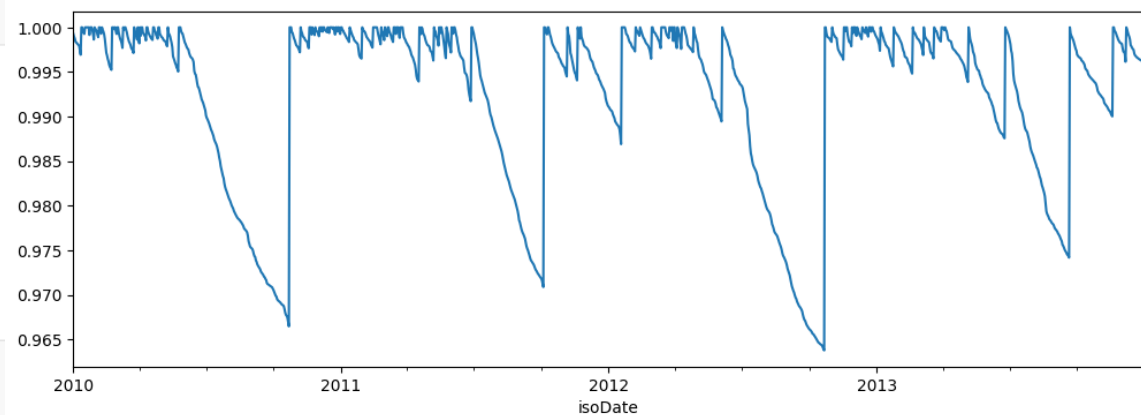
```
1 import pvlib
2 site.register_model(pvlib.soiling.kimber)
3 site.calculate('kimber').plot()
```

Python

```
1 site.register_model(pvlib.soiling.hsu, {'cleaning_threshold': 1, 'surface_tilt': 30})
2 site.calculate('hsu').plot()
```

✓ 13.7s

Python



```
1 site.models
```

✓ 0.0s

Python

```
{'rainfall': <Total daily rainfall in mm. Source: MERRA2>,
 'soiling_level': <Avg. daily soiling level in unit fraction. Source: PVRADAR>,
 'pm2_5': <Avg. daily concentration of particulate matter with less than 2.5 µm in g/m³ air. Source: MERRA2>,
 'pm10': <Avg. daily concentration of particulate matter with less than 10 µm in g/m³ air. Source: MERRA2>}
```

CHANGING THE SOURCE OF RAINFALL DATA

```
1 site.get_available_noaa_stations(sort_by='distance')[0:10]
```

Python

	lat	lon	start	end	coverage	completeness	distance	station_id	score
8	39.187500	-122.026900	2002-01-01	2023-12-31	1.000000	0.790411	9232.016236	USC00041948	1.960104
21	39.159623	-121.751110	2010-05-09	2016-12-23	0.912389	0.466967	16594.348612	US1CASU0005	1.132328
23	39.224066	-121.594033	2011-11-06	2023-12-31	0.538672	0.949109	31463.323787	US1CAYB0003	1.079682
0	39.102050	-121.568840	2002-01-01	2023-12-31	1.000000	1.000000	32409.679540	USW00093205	2.325226
18	39.369133	-121.692574	2013-03-08	2016-11-15	0.204654	1.000000	33505.871433	US1CABT0018	1.317137
16	38.926100	-121.544700	2002-01-01	2023-12-31	1.000000	0.743151	41609.735087	USC00046194	1.405015
26	39.013475	-121.424317	2011-11-06	2023-12-31	0.538672	0.697201	46769.058722	US1CAYB0004	0.361177
1	39.494300	-121.622260	2002-01-01	2023-12-31	1.000000	0.995205	48266.257699	USW00093210	2.155037
6	38.682900	-121.794000	2002-01-01	2023-12-31	1.000000	0.976712	52222.771114	USC00049781	2.014606
9	39.523160	-122.304800	2002-01-01	2023-12-31	1.000000	0.960274	53103.679381	USC00049699	1.918047

```
1 from pvradar.sdk.modelling.pvradar_models import noaa_precipitation
2 site.register_model(noaa_precipitation, {'station_id': 'USW00093205'})
```

Python

AVAILABLE DATA

<u>meteorological series</u>	<u>data sources</u>	<u>site information</u>	<u>helper functions</u>
rainfall	MERRA2	Köppen-Geiger class	tracker angle
snowfall	NOAA	vegetation type	sun angle
snowdepth	PRISM	land use	incidence angle
pm2_5 & pm10	...	country name	...
air temperature		...	
relative humidity			yield
wind speed	local weather stations		...
wind direction	...		
irradiance			
different component			
horizontal and POA			
...			

MODELLING SNOW SOILING

```
1 site.models
```

✓ 0.0s

Python

```
{'rainfall': <Total daily rainfall in mm. Source: MERRA2>,  
'soiling_level': <Avg. daily soiling level in unit fraction. Source: PVRADAR>,  
'pm2_5': <Avg. daily concentration of particulate matter with less than 2.5 µm in g/m³ air. Source: MERRA2>,  
'pm10': <Avg. daily concentration of particulate matter with less than 10 µm in g/m³ air. Source: MERRA2>,  
'snow_total': <Monthly total snowfall in cm>,  
'snow_event': <Number of snowfall events each month.>,  
'relative_humidity': <Avg. monthly relative humidity in %>,  
'air_temp': <Avg. monthly air temperature in °C>,  
'poa_global': <Monthly total in-plane irradiation in Wh/m²>}
```

from site
model parameters

```
def loss_townsend(  
    snow_total,  
    snow_events,  
    relative_humidity,  
    temp_air,  
    poa_global,  
    surface_tilt,  
    slant_height,  
    lower_edge_height,  
    string_factor=1.0,  
    angle_of_repose=40):
```

```
1 site.register_model(pvlib.snow.loss_townsend)
```

Python

BUILDING STACKED MODELS

```
1 def my_combined_model(  
2     my_soiling_function:pd.Series,  
3     loss_townsend:pd.Series  
4 ):  
5  
6     monthly_soiling = my_soiling_function.groupby(pd.Grouper(freq='M')).mean()  
7     combined_loss = monthly_soiling + loss_townsend # add soiling and snow losses  
8  
9     p50_loss_factors = combined_loss.groupby(pd.Grouper(combined_loss.index.dt.month)).mean() # group by month of year  
10  
11     return p50_loss_factors
```

Python

```
1 def my_AI_model(  
2     all_data_I_need:pd.Series,  
3     any_parameter_I_define:float  
4 ):  
5     # ...  
6     # some dark magic  
7     # ...  
8  
9     return result
```

Python

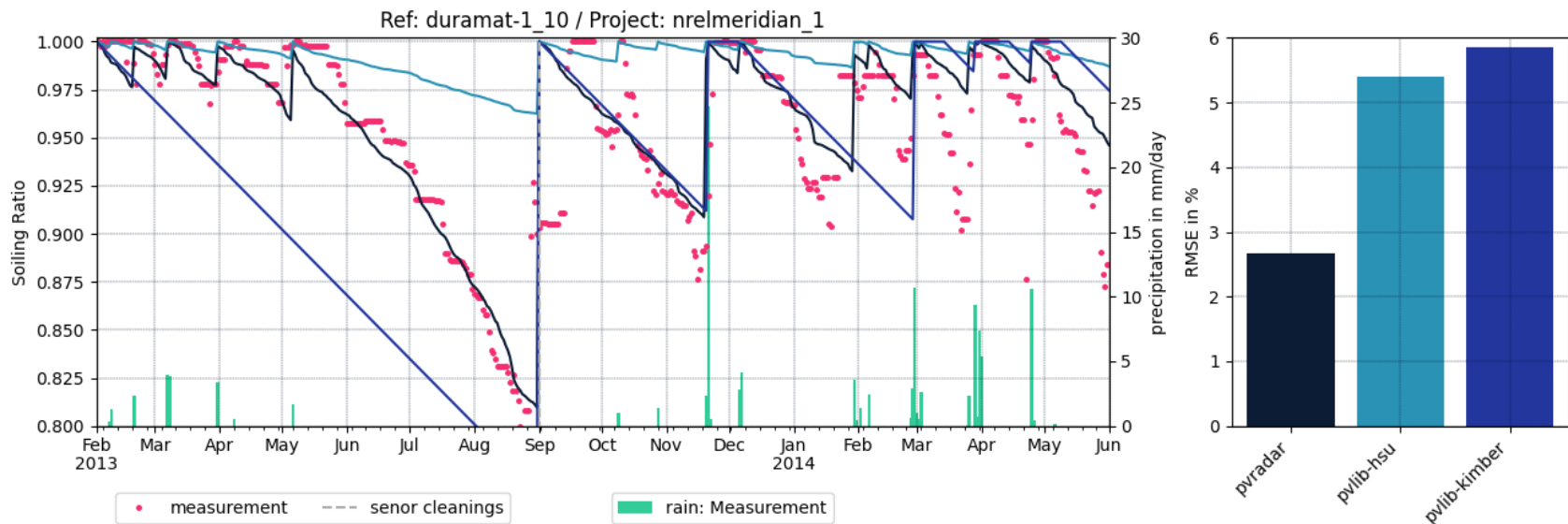
FIT MODEL PARAMETERS

```
1 measurement = Reference.from_database_new(reference_id='duramat-1_10')
2 site.fit(model='my_soiling_model', params=['soiling_rate', 'cleaning_threshold'], measurement=measurement)
```

✓ 0.0s

Python

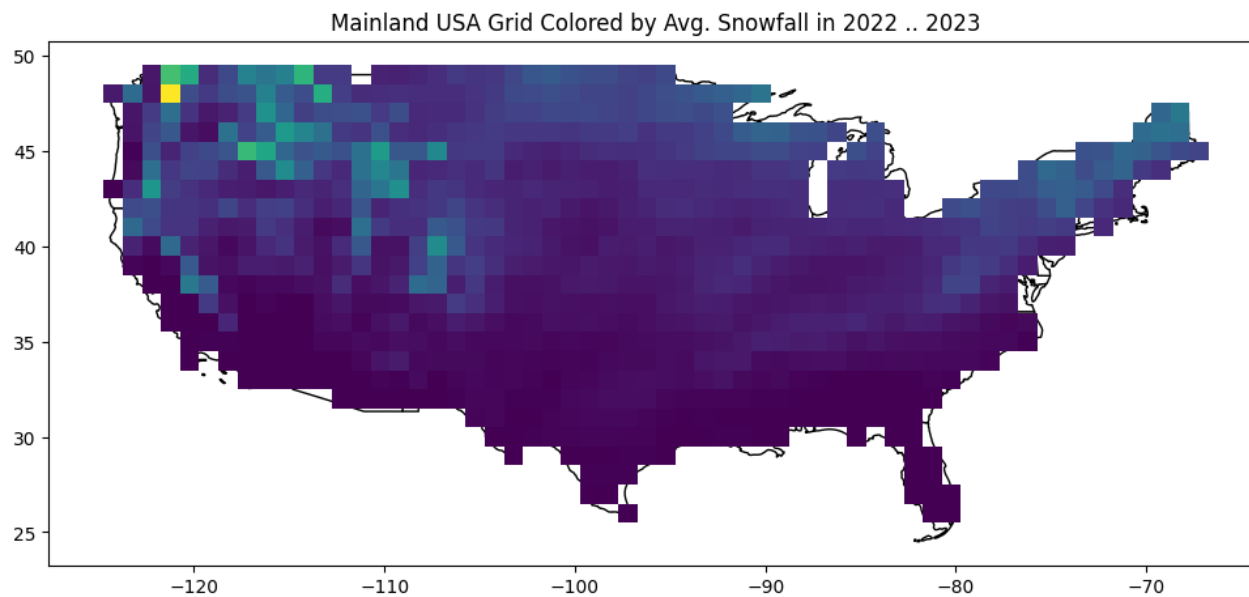
```
{'soiling_rate': 0.011, 'cleaning_threshold': 2}
```



DRAW MAPS

```
1 def avg_snowfall(  
2     snowfall: pd.Series  
3 ) -> float:  
4     return snowfall.mean()  
5  
6 map = Map('USA', period=Period.year_range(2022,2023))  
7 map.calculate('avg_snowfall')
```

Python



OUR OFFER

- **python library:** open source, release soon, currently beta-testing
- **data depot:** constantly adding more data
- **measurement database:** approx. 100 years soiling
- **model repository**
 - as part of the library
 - as notebooks on our website
- **access:**
 - library is open source
 - data is partially public, partially restricted (API key)
- **looking for early adopters and beta testers:** contact us!
- **special offer for researches and public orgs:** [contact us for full and free access!](#)

UPCOMING WEBINAR – NEXT WEEK



LIVE WEBINAR

Beyond the 2% Default: Introducing PVRADAR soiling loss integration within PlantPredict

A deep dive into PVRADAR's physical soiling model,
underlying datasets, and integration within PlantPredict.
PLUS, a preview of PlantPredict next major release (v12).

 **Wednesday, August 28, 2024**

 **8AM PT (11AM ET / 5PM CEST)**



Let's work together!



Thore Müller

Co-Founder and CEO

+49 (1512) 6663494

thore.mueller@pvradar.com



Franco Clandestino

Co-Founder and Head of Product

+49 (1573) 6762968

franco.clandestino@pvradar.com

PVRADAR Labs GmbH, Dieding 1, 85560 Ebersberg
pvradar.com