Operational research for urban solar development

“PV failure detection based on operational time series”

26/11/2024
Alexandre Mathieu

Curriculum
Plan

Day	Time	Duration	Content
Wednesday 13/11/2024	11h15-12h45 14h15-15h45	1h30 + 1h30	50% Lecture / 50 % Hands-on
Tuesday 26/11/2024	9h45-13h00	1h30 + 1h30	25% Lecture / 75 % Hands-on
Monday 02/12/2024	13h15-16h15	3h	15% Lecture / 85 % Hands-on
Monday 09/12/2024	8h-11h 13h15-16h15	6h	10% Lecture / 90 % Hands-on/Project
Tuesday 10/12/2024	8h-11h	3h	10% Lecture / 90 % Project
Monday 16/12/2024	8-11h	3h	10% Lecture / 90 % Project
Thursday 19/12/2024	9h45-12h45	3h	10% Lecture / 90 % Project
Monday 06/01/2025	13h15-14h45	1h30	100% Project
Monday 13/01/2025	9h45-11h45	1h30	100% Project
Total		**27h**	

Agenda

Review notebook last week

PV performance model steps

Notebook: Shading effect on irradiance

Review first notebook

Notebook recap 13/11/2024

Google collab link: https://colab.research.google.com/drive/1y9-l7Vf9I7qTfaDfnh4ubo5kybaRqehb?usp=sharing

Correction: https://github.com/AlexandreHugoMathieu/pvfault_detection_solar_academy/blob/master/notebooks/python_intro_poa.ipynb

UNIVERSITÉ SAVOIE MONT BLANC

Review first notebook

Notebook recap 13/11/2024

Python commands 1/4

ts

a	1
b	2
c	3

index

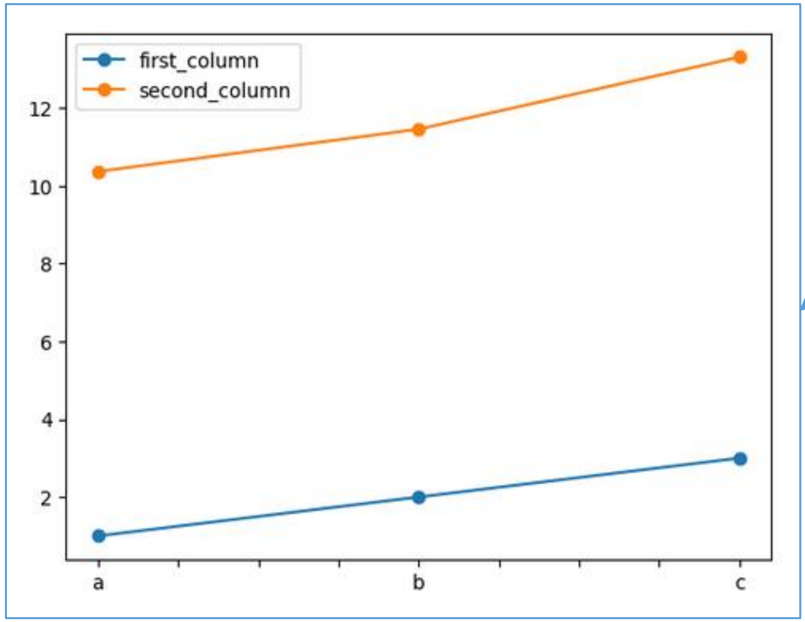
values

df

first_column second_column

a	1	2
b	2	4
c	3	6

index




```
import numpy as np # import to your python instance the package “numpy”  and rename it “np” (helpful for math calculations)
import pandas as pd # import to your python instance the package “pandas” (helpful for data structure and calculations)

ts = pd.Series([1, 2,3], index=['a','b','c']) # Initiate a pandas serie into variable “ts”
ts2 = ts + ts/2 + np.cos(ts) + np.pi # Make calculate with “ts” and store it into “ts2”
print(ts2) # print serie ts

ts.plot(marker="o") # Make a plot of ts with “o” (circle) marker

df = pd.DataFrame() # Initiate an empty dataframe into variable “df”
df["first_column"] = ts # Store “ts” serie in a column labeled “first_column”
df["second_column"] = ts2 * 2 # Store “ts2” serie  in another column labeled “second_column”

df.plot(marker="o") # Make a plot of df with “o” (circle) marker

df.loc[“a”, :] # Select the entire row with “a” as index
df.loc[“a”, “first_column”] # Select the value with “a” as index and “first_column” as column
```

UNIVERSITÉ SAVOIE MONT BLANC

Review first notebook

Notebook recap 13/11/2024

Python commands 2/4

	poa_global	poa_direct	poa_diffuse	poa_sky_diffuse	poa_ground_diffuse
2022-01-01 00:00:00+01:00	NaN	NaN	NaN	NaN	NaN
2022-01-01 01:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 02:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 03:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 04:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 05:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 06:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 07:00:00+01:00	0.000000	0.000000	0.000000	0.000000	0.000000
2022-01-01 08:00:00+01:00	0.593235	0.000000	0.593235	0.589085	0.004150
2022-01-01 09:00:00+01:00	71.788066	46.706296	25.081770	24.724777	0.356993
2022-01-01 10:00:00+01:00	210.546485	150.470436	60.076049	59.167899	0.908150
2022-01-01 11:00:00+01:00	376.976885	313.961064	63.015821	61.519000	1.496821

```
####### Calculate POA, the lazy way #######

from pvlib.irradiance import get_total_irradiance # import the function "get_total_irradiance from pvlib"
# On  another note, pvlib* is a very useful package for PV modeling with plenty of convenient functions, do not hesitate to look it up on the web

beta = 20 # tilt [°]
azimuth = 180 # azimuth [°]
rho = 0.2 # albedo

solar_position = pd.read_csv("solarpos_data.csv") # Import the data file "solarpos_data.csv" which contains the sun path (azimuth and elevation)  with datetime index
weather_data = pd.read_csv("sat_data.csv", index_col=0) # Import the data file "sat_data.csv" which irradiance (dni, ghi, dhi) with datetime index

data = get_total_irradiance(beta, azimuth, solar_position["zenith"], solar_position["azimuth"], weather_data["dni"], weather_data["ghi"], weather_data["dhi"], albedo=rho) # Directly apply the isotropic models

print(data.head(12)) # Show the first 12 lines of the DataFrame
```

values

Pvlib ref
*William F. Holmgren, Clifford W. Hansen, and Mark A. Mikofski. "pvlib python: a python package for modeling solar energy systems." Journal of Open Source Software, 3(29), 884, (2018). https://doi.org/10.21105/joss.00884

Review first notebook

Notebook recap 13/11/2024

Python commands 3/4

```
####### Function definition #######

# Function, useful to store few lines of code you want to reuse and apply with different inputs
def my_func_name(argument1, argument2):
# Define the function "my_func_name" with the "def" command and a small increment tab to the right
        y = argument1 + argument2 * 2
        return y

my_func_name(1,2) # Apply the function with the two mandatory arguments and return 5

my_func_name(1,argument2=3) # Return 7

my_func_name(argument2=3, argument1=2) # Return 8
```

Review first notebook

Notebook recap 13/11/2024

Python commands 4/4

```
####### Function definition #######

# Function, useful to store few lines of code you want to reuse and apply with different inputs
def my_func_name2(argument1, argument2, argument3=0):
# Define the function "my_func_name" with the "def" command and a small increment tab to the right
# Argument3 is optional
        y = argument1 + argument2 * 2  + argument3
        return y

my_func_name2(1,2,2) # Apply the function with three arguments and return 7

my_func_name2(1,2) # Apply the function with the two mandatory arguments and return 5

my_func_name(argument2=3, argument1=2, 3)
# Crash because all arguments need to be specified on the right of a keyword argument

        -> “”” SyntaxError: positional argument follows keyword argument”””
```

https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.irradiance.get_total_irradiance.html

Review first notebook

Online Documentation

pvlib   User Guide   Example Gallery   API reference   What's New   Contributing   Search Ctrl +

Section Navigation

- es
- Position
- Clear sky
- Airmass and atmospheric models
- Irradiance
  - Methods for irradiance calculations
  - Decomposing and combining irradiance
  - Transposition models
    - pvlib.irradiance.get_total_irradiance
    - pvlib.irradiance.get_sky_diffuse
    - pvlib.irradiance.isotropic
    - pvlib.irradiance.perez
    - pvlib.irradiance.perez_driesse
    - pvlib.irradiance.haydavies
    - pvlib.irradiance.klucher
    - pvlib.irradiance.reindl
    - pvlib.irradiance.king
    - pvlib.irradiance.ghi_from_poa_driesse_2023
  - DNI estimation models
  - Clearness index models

API reference > ··· > Transposition models > pvlib.irradi...

pvlib.irradiance.get_total_irradiance

```
pvlib.irradiance.get_total_irradiance(surface_tilt, surface_azimuth, solar_zenith, solar_azimuth, dni, ghi, dhi, dni_extra=None, airmass=None, albedo=0.25, surface_type=None, model='isotropic', model_perez='allsitescomposite1990') #
```

[source]

Determine total in-plane irradiance and its beam, sky diffuse and ground reflected components, using the specified sky diffuse irradiance model.

$$I_{tot} = I_{beam} + I_{skydiffuse} + I_{ground}$$

Sky diffuse models include:

- isotropic (default)
- klucher
- haydavies
- reindl
- king
- perez
- perez-driesse

Parameters:

- **surface_tilt** (*numeric*) – Panel tilt from horizontal. [degree]
- **surface_azimuth** (*numeric*) – Panel azimuth from north. [degree]

https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.irradiance.get_total_irradiance.html

pvlib   User Guide   Example Gallery   API reference   What's New   Contributing   Search Ctrl +

Review first notebook

Online Documentation

Section Navigation

- ...es
- Solar Position
- Clear sky
- Airmass and atmospheric models
- Irradiance
  - Methods for irradiance calculations
  - Decomposing and combining irradiance
  - Transposition models
    - pvlib.irradiance.get_total_irradiance
    - pvlib.irradiance.get_sky_diffuse
    - pvlib.irradiance.isotropic
    - pvlib.irradiance.perez
    - pvlib.irradiance.perez_driesse
    - pvlib.irradiance.haydavies
    - pvlib.irradiance.klucher
    - pvlib.irradiance.reindl
    - pvlib.irradiance.king
    - pvlib.irradiance.ghi_from_poa_driesse_2023
  - DNI estimation models
  - Clearness index models

API reference > ... > Transposition models > pvlib.irradi...

pvlib.irradiance.get_total_irradiance

```
pvlib.irradiance.get_total_irradiance(surface_tilt, surface_azimuth, solar_zenith, solar_azimuth, dni, ghi, dhi, dni_extra=None, airmass=None, albedo=0.25, surface_type=None, model='isotropic', model_perez='allsitescomposite1990') #
```

[source]

Determine total in-plane irradiance and its beam, sky diffuse and ground reflected components, using the specified sky diffuse irradiance model.

$$I_{tot} = I_{beam} + I_{skydiffuse} + I_{ground}$$

Mandatory arguments

Sky diffuse models include:

- isotropic (default)
- klucher
- haydavies
- reindl
- king
- perez
- perez-driesse

Parameters:

- **surface_tilt** (*numeric*) – Panel tilt from horizontal. [degree]
- **surface_azimuth** (*numeric*) – Panel azimuth from north. [degree]

https://pvlib-python.readthedocs.io/en/stable/reference/generated/pvlib.irradiance.get_total_irradiance.html

Review first notebook

Online Documentation

pvlib

User Guide | Example Gallery | API reference | What's New | Contributing | Search Ctrl +

API reference > ··· > Transposition models > pvlib.irradi...

Section Navigation

- Solar Position
- Clear sky
- Airmass and atmospheric models
- Irradiance
  - Methods for irradiance calculations
  - Decomposing and combining irradiance
  - Transposition models
    - pvlib.irradiance.get_total_irradiance
    - pvlib.irradiance.get_sky_diffuse
    - pvlib.irradiance.isotropic
    - pvlib.irradiance.perez
    - pvlib.irradiance.perez_driesse
    - pvlib.irradiance.haydavies
    - pvlib.irradiance.klucher
    - pvlib.irradiance.reindl
    - pvlib.irradiance.king
    - pvlib.irradiance.ghi_from_poa_driesse_2023
  - DNI estimation models
  - Clearness index models

pvlib.irradiance.get_total_irradiance

```
pvlib.irradiance.get_total_irradiance(surface_tilt, surface_azimuth,
solar_zenith, solar_azimuth, dni, ghi, dhi, dni_extra=None, airmass=None,
albedo=0.25, surface_type=None, model='isotropic',
model_perez='allsitescomposite1990') #
```

[source]

Determine total in-plane irradiance and its beam, sky diffuse and ground reflected components, using the specified sky diffuse irradiance model.

$$I_{tot} = I_{beam} + I_{skydiffuse} + I_{ground}$$

Sky diffuse models include:

- isotropic (default)
- klucher
- haydavies
- reindl
- king
- perez
- perez-driesse

Mandatory arguments

Optional (default) arguments

Parameters:

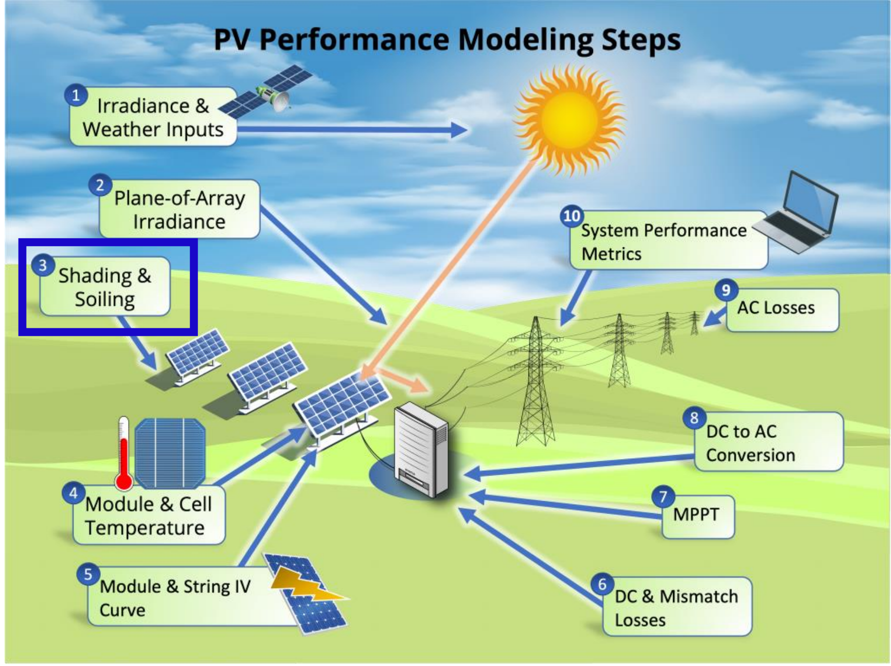
- **surface_tilt** (*numeric*) – Panel tilt from horizontal. [degree]
- **surface_azimuth** (*numeric*) – Panel azimuth from north. [degree]

Agenda

Review notebook last week

PV performance model steps

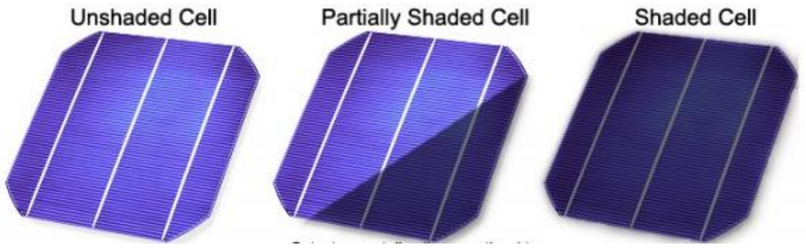
Modeling steps



3. Shading

Shadow can come from near and far elements and their impact can be distinguished into two categories

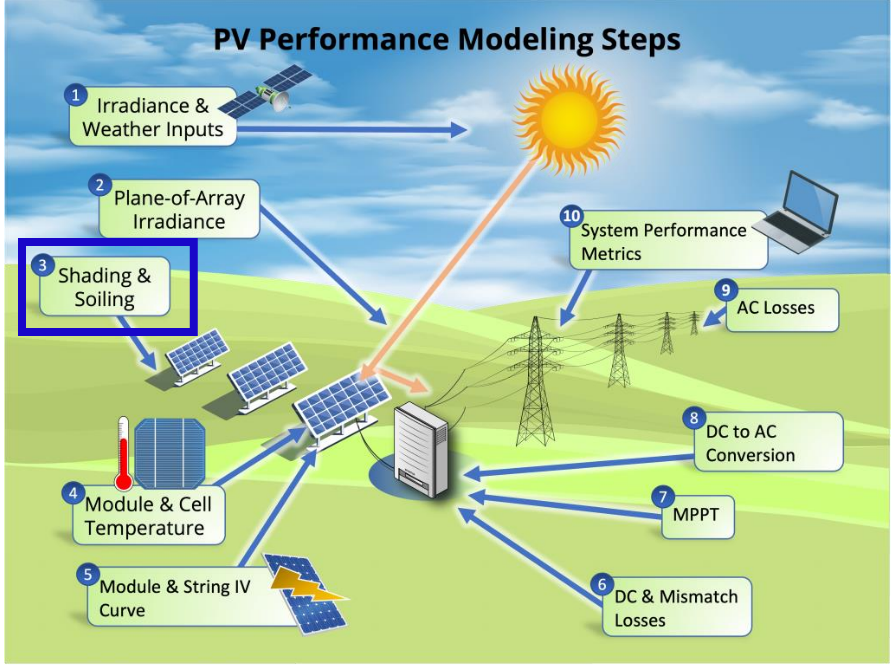
1. **Partial shading** refers to a condition where some but not all of the solar cells or panels in a PV array are exposed to sunlight while others are shaded.
2. **Full shading** occurs when the entire PV array is covered and deprived of direct sunlight.



*https://mcisolutions.ca/effect-of-shade-on-solar-panels/

UNIVERSITÉ SAVOIE MONT BLANC

Modeling steps



3. Shading

Shadow can come from near and far elements and their impact can be distinguished into two categories

1. **Partial shading** refers to a condition where some but not all of the solar cells or panels in a PV array are exposed to sunlight while others are shaded.
2. **Full shading** occurs when the entire PV array is covered and deprived of direct sunlight.

In average, if a PV panel is around 8% covered, bypass diodes start to activate and **might** reduce the module power production to much lower levels.

Demonstrated in one of pvlib example:
https://pvlib-python.readthedocs.io/en/stable/gallery/shading/plot_partial_module_shading_simple.html

*https://mcisolutions.ca/effect-of-shade-on-solar-panels/

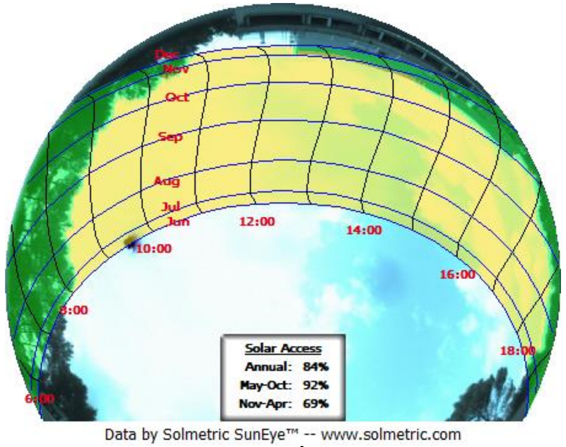
Modeling steps

3. Shading

The Fisheye camera enables to account for the in-situ (full) shading in PV modeling.

South

East



West

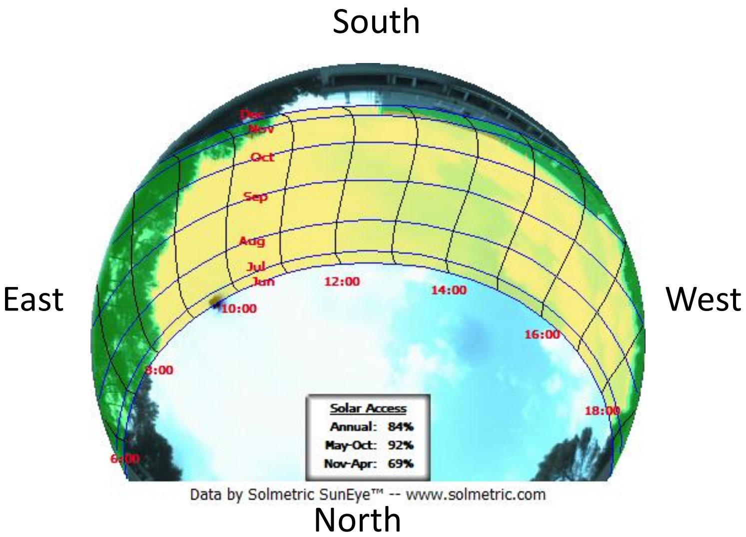
North

Example in South of France

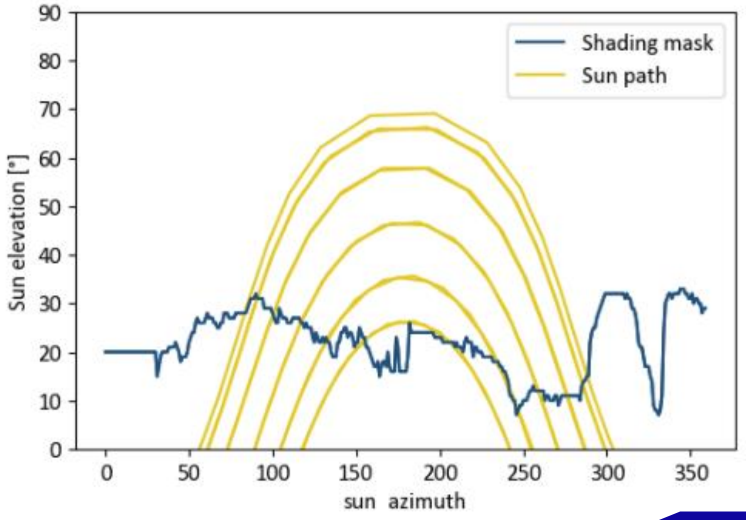
Modeling steps

3. Shading

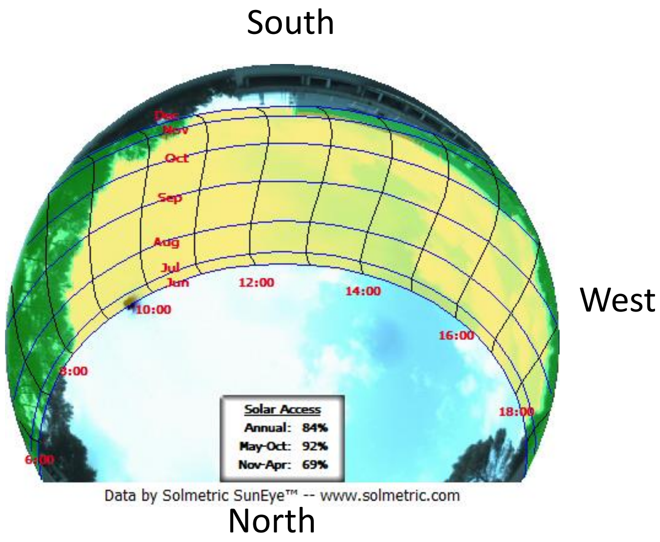
The Fisheye camera enables to account for the in-situ (full) shading in PV modeling.



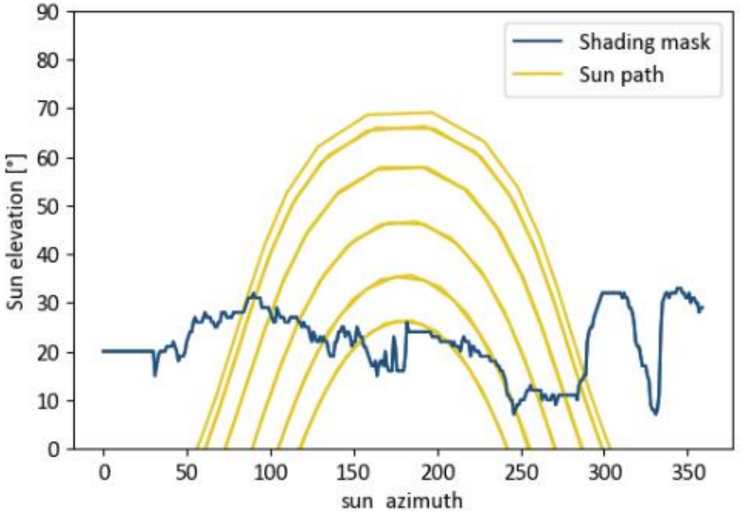
Example in South of France



Modeling steps



Example in South of France

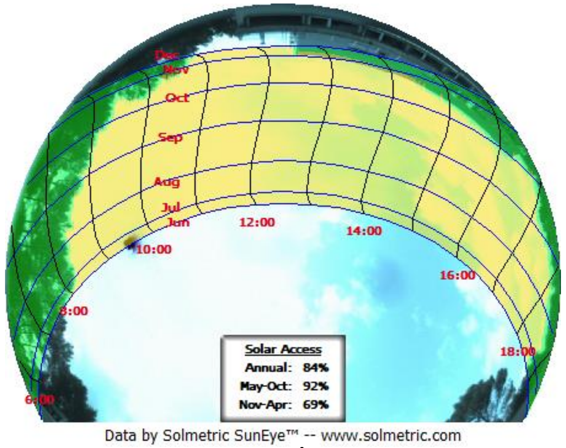


Modeling steps

3. Shading

The Fisheye camera enables to account for the in-situ (full) shading in PV modeling.

South

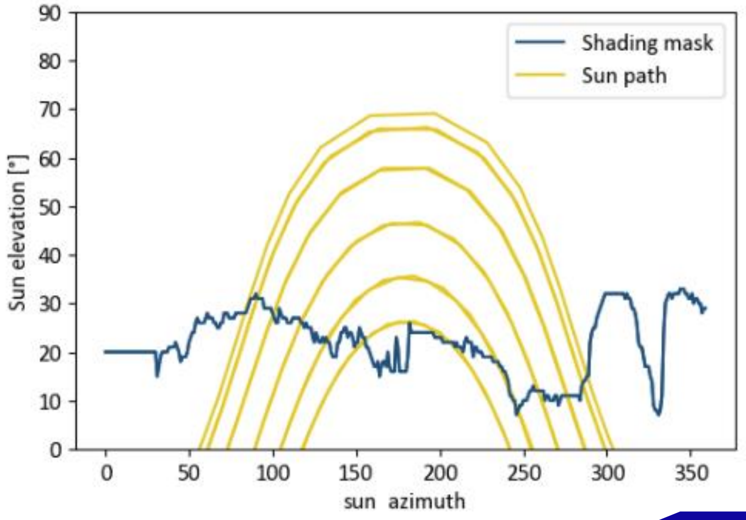
East  West

North

Example in South of France

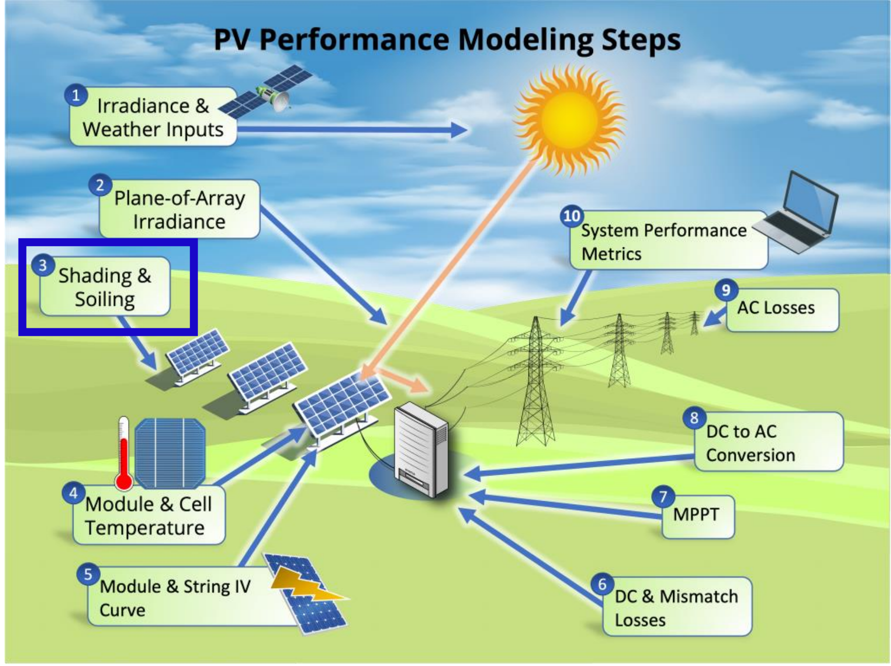
Under the black curve, simple assumptions lead to:

- $POA_b = 0 \frac{W}{m2}$
- POA_d, POA_{grd} are not significantly modified



UNIVERSITÉ SAVOIE MONT BLANC

Modeling steps



3. Soiling losses

Big subject in the PV industry:

“**Soiling losses** refer to loss in power resulting from snow, dirt, dust and other particles that cover the surface of the PV module.” (Maghami et al., 2016)

Report of 130 pages on PV soiling: IEA PVPS, Soiling Losses – Impact on the Performance of Photovoltaic Power Plants 2022

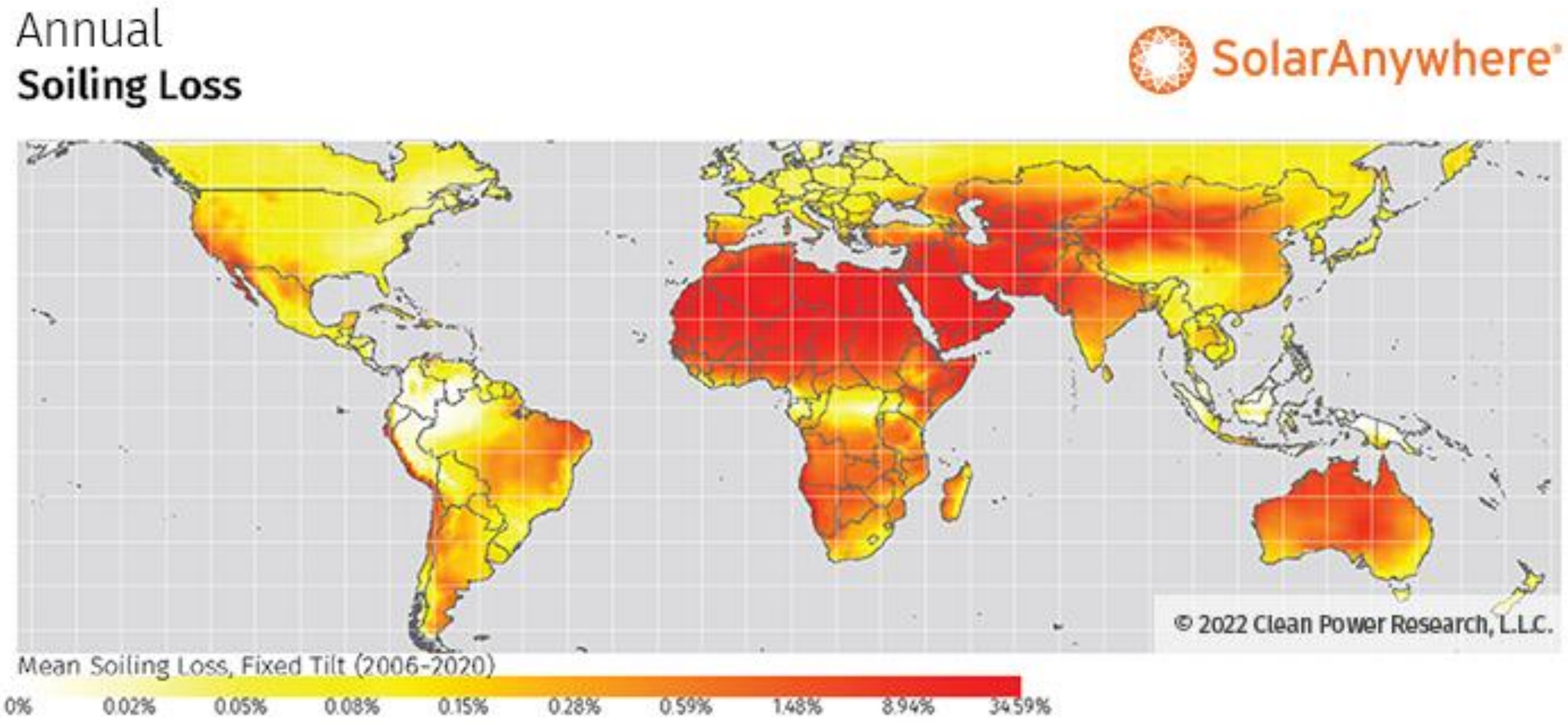
Modeling steps

3. Soiling losses

Can roughly be ignored in France according to the HSU model.

Figure 1: Soiling Loss Map Based on SolarAnywhere Data and HSU Soiling Model

Annual Mean Soiling Loss (2006-2020); Fixed-tilt PV System



Model HSU: M. Coello and L. Boyle, "Simple Model for Predicting Time Series Soiling of Photovoltaic Panels, Sept. 2019.

Modeling steps

3. Shading / Terrain horizon mask

Time for some
hands-on exercises,
Again!

Modeling steps

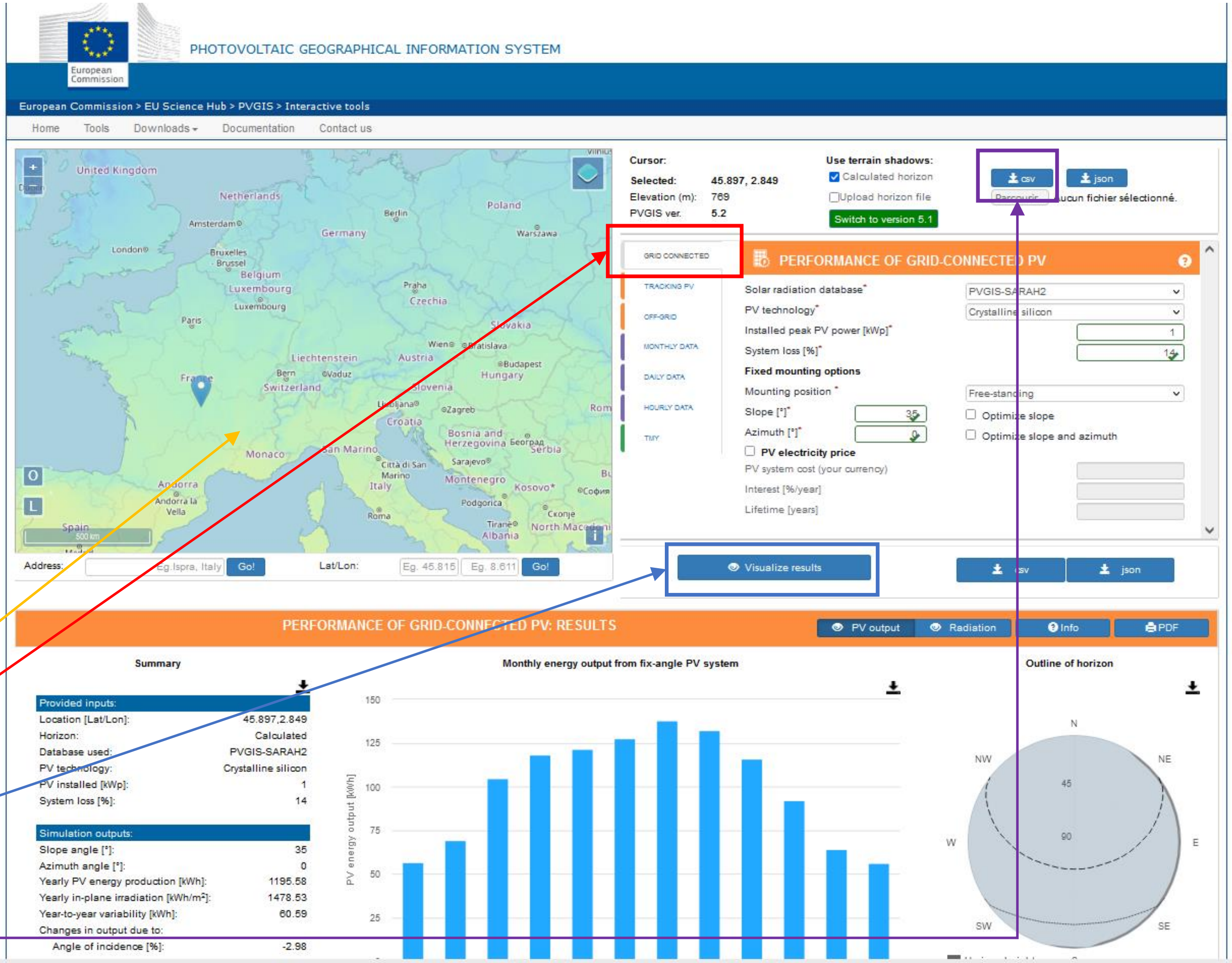
3. Shading / Terrain horizon mask

PVGIS: Website/Online Tool to estimate power production: https://re.jrc.ec.europa.eu/pvg_tools/en/

- Enables to extract the horizon mask with a Digital Surface Model (DSM).

Instructions:

1. Generate a simulation on PVGIS
   a. Click on the map on Grenoble and select the « Grid connected tab »
   b. Vizualize
   c. Extract the horizon file in csv format



2. Follow the instructions on the jupyter notebook and calculate the modified POA on one year. https://colab.research.google.com/drive/1hB1pmBw-n7RiS99vCHQi2OKicfcQxpcO?usp=sharing

UNIVERSITÉ SAVOIE MONT BLANC

Resources

- Modeling guide PVPMC: https://pvpmc.sandia.gov/modeling-guide/
- Python / Pvlib tutorial: https://pvsc-python-tutorials.github.io/PVSC48-Python-Tutorial/
- To go further:
  - The Use of Advanced Algorithms in PV Failure Monitoring: https://iea-pvps.org/wp-content/uploads/2021/10/Final-Report-IEA-PVPS-T13-19_2021_PV-Failure-Monitoring.pdf

Shading

PV education 7.2, widget: https://www.pveducation.org/pvcdrom/modules-and-arrays/shading

- How to take into account shading from LIDAR data:
  Solar Energy Potential Assessment on Rooftops and Facades in Large Built Environments Based on LiDAR Data, Image Processing, and Cloud Computing. Methodological Background, Application, and Validation in Geneva (Solar Cadaster), Desthieux et al.
  https://www.frontiersin.org/journals/built-environment/articles/10.3389/fbuil.2018.00014/full

That’s it