

2020

2021

2022



PVAnalytics: A Python Package for Automated Processing of Solar Time Series Data

Kirsten Perry (NREL), William Vining (Sandia), Kevin Anderson (NREL), Matthew Muller (NREL), Cliff Hansen (Sandia)

PV Performance Modeling and Monitoring Workshop Salt Lake City, Aug 24, 2022

pvanalytics v0.1.2 **PVPerformance** MODELING COLLABORATIVE

pvanalytics v0.1.1

pvanalytics v0.1.0

https://github.com/pvlib/pvanalytics

Contents

| 1 | PVAnalytics Background |
|---|-------------------------------|
| 2 | Package Features |
| 3 | Algorithm Validation |
| 4 | Documentation Updates |
| 5 | Automated testing |
| 6 | Community growth |
| 7 | PVAnalytics v0.1.3 and Beyond |

PVAnalytics Background

- Solar time series data can vary significantly in quality or lack critical metadata
- Several solar metrics dependent on data cleaning/filtering [1]
 - Performance loss rate (PLR)
 - Power production forecasting
 - Soiling loss
- PVAnalytics Python library: automated processing of solar time series data, including QA/QC
 - Data quality control and filtering
 - Identifying system characteristics, such as mounting configuration, tilt, and azimuth
 - Feature identification: clipping, day-night masking, clearsky detection
 - <u>https://pvanalytics.readthedocs.io/en/stable/</u>

[1] Lindig et. al. International collaboration framework for the calculation of performance loss rates: Data quality, benchmarks, and trends (towards a uniform methodology). Progress in Photovoltaics, 2021.

PVAnalytics Background (Continued)

- Design Principles behind PVAnalytics:
 - Open-source: tested, documented, and reusable
 - Independent of analysis workflow
 - Collection point for code which implements published algorithms
 - Collaboration between Sandia and NREL
 - Started as DuraMAT project: DOE-led consortium for PV module reliability and durability
 - Functions adapted from Solar Forecast Arbiter [1] and NREL PV Fleets Initiative [2]

[1] https://solarforecastarbiter-core.readthedocs.io/en/latest/

[2] D. Jordan et. al. *Photovoltaic fleet degradation insights*. Progress in Photovoltaics, 2022.





PLR distribution from the PV Fleets Initiative [2]

Package Features: Basic Time Series Filtering

Outlier detection and filtering: Hampel, Zscore, and Tukey filters

Stale data detection and filtering: Looks for consecutive repeating data

Interpolated data detection and filtering





Package Features: Advanced Time Series Filtering

Detecting missing data periods: Assign daily data a "completeness" score



[1] K. Perry, M. Muller. Automated Shift Detection in Sensor-Based PV Power and Irradiance Time Series. 2022 PVSC.

Data shift detection and filtering:

Uses changepoint detection to find massive, abrupt capacity changes. Described further in [1]



NREL | 6

Package Features: Feature Detection

- Day-night masking
 - Logic-based routine for masking day periods from night periods
- Clipping detection and filtering
 - Adapted from logic-based filter described in [1]
- Shading detection
 - Uses morphological image processing methods to identify shadows in GHI data [2]

[1] K. Perry, et. al. *Performance comparison of clipping detection techniques in AC power time series*. 2021 PVSC.

[2] Martin, C. E., Hansen, C. W., An Image Processing Algorithm to Identify Near-Field Shading in Irradiance Measurements, preprint 2016

Day-night masking on an AC power time series



Package Features: Irradiance Checks

Compare GHI sensor-based Irradiance quality checks: **Clearsky period** data to clearsky data. Filter consistency and physical filtering: Reno clearsky where GHI is within daily limits of GHI, DNI, and DHI method (1) insolation limit using QCrad criteria RMIS GHI RMIS GHI RMIS DHI 1000 Clearsky GHI ا اrradiance [W/m2] ج 00 RMIS DNI Within Daily Insolation Limit QCRAD Consistent 00 800 Irradiance (W/m^2) 00 600 00 Ital 400 00 Horizon 500 00 200 Global 100 00 0 04 02 03 05 06 02 Feb 2019 03 04 05 Feb 2019 15:00 18:00 21:00 06:00 09:00 12:00 00:00 03:00 20-Jan Date Date

[1] Reno, M.J. and C.W. Hansen, "Identification of periods of clear sky irradiance in time series of GHI measurements" Renewable Energy, v90, p. 520-531, 2016.

NREL | 8

Clearsky day filtering:

Package Features: System Characteristics

- Mounting configuration
 - Fixed-tilt or single-axis tracking
 - Uses daily power profile to classify time series stream

• Azimuth and tilt

- Estimate using AC power time series
- Work in progress: multiple methods in package are currently being validated





Algorithm Validation

- Continued validation of each algorithm
 - How well does each algorithm perform on labeled data sets?
 - Quantifiable metrics: accuracy and F1-score
 - Labeled data sets to encourage further development
- Technical documentation/publications benchmarking each algorithm's performance

https://datahub.duramat.org/project/example-data

| Y Projects / Example and Evalua | ation Data | | | | | | |
|---------------------------------|--|---|--|----------------|---------------------|--|--|
| ≡ Project | Fy: | ample and Eval | uation Data - Dataset | \$ | | | |
| Verview | LAC | | | | | | |
| Datasets | Proje | ect ID | e46c953f-0d84-41ct | 8-9526-2a219 | 950e6c92 | | |
| r Tags | | | | | | | |
| olar (3) | Search datasets | | | | | | |
| lata shift (2) | | | | | | | |
| AL training (2) | | | | Order by: | Relevance | | |
| IREL (2) | | reky Pariod Data Sat | | | | | |
| coiling (2) | Labeled Glearsky Period Data Set Resource | | | | | | |
| ime series (2) | This dataset contains I following columns: me | labeled clearsky periods for 1 r easured_on (the timestamp | ninute-interval data for the NREL Sanyo 1 s | system. The (| CSV contains the | | |
| raining data (2) | | | | | | | |
| sutomated (1) | Inverter Clippi | ing ML Training Set - Real I | Jata | | | | |
| Cleaning (1) | 64 Resources Real PV Field data for e | evaluating, validating, or trainin | g machine learning algorithms to recogniz | e clipping ev | ents. | | |
| learsky (1) | https://doi.org/10.2194 | 48/1874779 | | | | | |
| how More Tags | 🚽 Labeled Time | e Series Data Shifts | | | | | |
| Institution | 1 Resource | | | | | | |
| IREL (10) | This data set includes sensor-based power ar | 101 daily summed time series ind irradiance streams, was | that are labeled for abrupt data shifts. The | e data, which | includes individual | | |
| T Data Source Type | | | | | | | |
| Addeling and Simulation (4) | Solar Mountin | ng Configuration Satellite Ir | nagery Data | | | | |
| ield Deployment (1) | This data set includes I | labeled solar array configuration | on/type satellite images. Data is provided i | n training, te | st, and validation | | |
| | ronders, and is used by | Che Private | | | | | |

Publicly available, labeled data sets on the DuraMAT DataHub

Documentation: Example Gallery

- Example gallery for majority of the package functions (v0.1.2)
 - Example data for running each algorithm
 - Plots illustrating algorithm results

| API Reference | Example G | allery | | Note: |
|--|--------------------------------------|----------------------------|------------------------------------|--|
| Example Gallery | Linumpic O | Junery | | Click <u>here</u> to download the full example code |
| Z-Score Outlier Detection Tukey Outlier Detection Hampel Outlier Detection | This gallery shows examp welcome! | les of pvanalytics functio | onality. Community contributions a | ^{rre} Clearsky Limits for Daily |
| Clear-Sky Detection | | | | Insolation |
| Interpolated Data Periods Clearsky Limits for Daily | | | | Checking the clearsky limits for daily insolation data. |
| Data Shift Detection & Filtering | | | | Identifying and filtering out invalid irradiance data is a useful way to reduce noise du ing analysis. In this example, we use monolitics analytic inclusion data is a static limit () to data mine when the |
| Clearsky Limits for | ********** | ********** | ****** | daily insolation lies between a minimum and a maximum value. Irradiance measure- |
| Irradiance Data Stale Data Periods Clipping Detection | Z-Score Outlier Detection | Tukey Outlier Detection | Hampel Outlier Detection | ments and clear-sky irradiance on each day are integrated with the trapezoid rule to culate daily insolation. For this example we will use data from the RMIS weather sys located on the NREL campus in Colorado, USA. |
| QCrad Limits for Irradiance Data | | | | <pre>import pvanalytics from pvanalytics.quality.irradiance import daily_insolation_limits</pre> |
| Missing Data Periods | | | | <pre>import pvlib import matplotlib.pyplot as plt</pre> |
| QCrad Consistency for Irradiance Data | - / \ | 40- 40- 40- | | import pandas as pd import pathlib |
| Day-Night Masking Release Notes | | | | First, read in data from the RMIS NREL system. This data set contains 5-minute rigit aligned data. It includes POA. GHL DNL DHL and GNI measurements. |
| Quick search | Clear-Sky Detection | Interpolated Data | Clearsky Limits for | |
| Go | | Periods | Daily Insolation | <pre>pvanalytics_urr = patilizerat(pvanalyticstile_).parent rmis_file = pvanalytics_dir / /dtai/ / 'irradiance_RMIS_NERL.csv' data = pd_read_csv(rmis_file, index_col=0, parse_dates=True) # Noke the dotetime index_tz_roware. data.index = data.index_tz_localize("Etc/GMI+7")</pre> |
| | | No | 40 | Now model clear-sky irradiance for the location and times of the measured data: |
| | Ann | | | |
| | Data Shift Detection | Clearsky Limits for | | |
| | & Filtering | Irradiance Data | | |

https://pvanalytics.readthedocs.io/en/stable/generated/gallery/index.html

Apply PVAnalytics to Your Own Data

How can you easily implement PVAnalytics functions to your own data?

CSV containing data streams (power, irradiance, temperature) is labeled as False. The data is sampled at 15-minute intervals.



Import CSV into our example documentation, and change any metadata parameters (lat-long coordinates, data frequency, etc)



https://pvanalytics.readthedocs.io/en/stable/generated/gallery/index.html

NREL | 12

Documentation: Function Descriptions

- Page for each model function containing:
 - Brief description
 - Input parameters: data type, description
 - Outputs: data type, description
 - Published reference for the function, if applicable
 - Additional notes as needed
 - Examples in the gallery using the function

| Identify state values in the data. For a window of length N, the last value (index N-1) is considered stale if all values in the window are close to the first value (index O.). Parameters rtol and atol have the same meaning as in numpy.sllclose(). Parameters: x (Scries) – data to be processed • window (int, default 0) – number of consecutive values which, if unchanged, indicates stale data • rtol([data, default 1c-3) – relative tolerance for detecting a change in data values • atol (locat, default 1c-3) – absolute tolerance for detecting a change in data values • atol (locat, default 1c-3) – absolute tolerance for detecting a change in data values • atol (locat, default 1c-3) – absolute tolerance for detecting a change in data values • atol (locat, default 1c-3) – absolute tolerance for detecting a change in data values • atol (locat, default 1c-3) – absolute tolerance for detecting a change in data values • atol (locat, default 1ca) – How much of the window to mark True when a sequence of stale values is detected. Can one be of tali', end', or 'all'. • If 'all' the default) then every point in the window except the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the first point is marked True. • If 'all 'then every point in the window including the | analytics.qual | <pre>ty.gaps.stale_values_diff(x, window=6, rtol=1e-05, 'toi!')</pre> |
|---|---|---|
| For a window of length N, the last value (index N-1) is considered stale if all values in the window are close to the first value (index o.). Parameters rtol and atol have the same meaning as in numpy.slicless(). Parameters: • x (Scries) - data to be processed • window (ing, default 0) - number of consecutive values which, if unchanged, indicates stale data • rtol (float, default 1c-5) - relative tolerance for detecting a change in data values • atol (float, default 1c-5) - absolute tolerance for detecting a change in data values • mark (str, default 1c-6) - beloute tolerance for detecting a change in data values • mark (str, default 1c-6) - number of consecutive values which, if unchanged, indicates stale data • nark (str, default 1c-6) - number of consecutive values in detected. Can one be of tali', 'end', or 'all'. • If 'all the of the window to mark Tree when a sequence of stale values is detected. Can one be of tali', 'end', or 'all'. • If 'all then the first urindow - 1 values in a stale sequence sequence are marked Tree. • If 'all then every point in the window including the first point is marked Tree. • If 'all then every point in the values in the sequence of data smarked Tree. • If 'all then every point in the values in a stale sequence of data smarked Tree. • If 'all then every point in the values in the sequence of 'all', 'end', or 'all'. • If 'all then every point in the values in the sequence of 'all'. • If 'all then e | Identify stale val | ues in the data. |
| Parameters rtol and atol have the same meaning as in numpy.sllclose(). Parameters: * x (Series) – data to be processed • window (int, default 6) – number of consecutive values which, if unchange, indicate states de dat • rtol (float, default 1c-5) – relative tolerance for detecting a change in data values • atol (float, default 1c-5) – absolute tolerance for detecting a change in data values • atol (float, default 1c-5) – the trive when a sequence of state values is detected. Can one be of 'tail', 'end', or 'all'. • If 'all' the default it hen every point in the window except the first point is marked True. • If 'all' then the first unindow - <i>x</i> values in a stale sequence sequence are marked Faise and all subsequent values in the sequence are marked Faise. • If 'all' then every point in the window including the first point is marked True. • If 'all' then every point in the window including the first point is marked True. • If 'all' then every point in the window including the first point is marked True. • True for each value that is part of a stale sequence of data Returns: • True for each value that is part of a stale sequence of data • Autometer - <i>x</i> (window < <i>x</i> or mark is not one of 'tail', 'end', or 'all'. • Notes Copyright (c) 2019 SolarArbiter. See the file LICENSES/SOLARFORECASTARBITER_LICENSE at the top level directory of this distribution and at | For a window of in the window ar | length N, the last value (index N-1) is considered stale if all values e close to the first value (index 0). |
| Parameters: * x (Series) – data to be processed window (<i>ing. default 6</i>) – number of consecutive values which, if unchanged, indicates state data rtol (<i>float, default 1e-5</i>) – relative tolerance for detecting a change in data values atol (<i>float, default 1e-5</i>) – beloute tolerance for detecting a change in data values mark (<i>str. default 1e-5</i>) – beloute tolerance for detecting a change in data values mark (<i>str. default 1e-5</i>) – beloute tolerance for detecting a change in data values mark (<i>str. default 1e-5</i>) – beloute tolerance for detecting a change in data values in trial (the default) then every point in the window except the first point is marked True. If 'ail' (the default) then every point in the window except the first point is marked True. If 'ail' (the default) then every point in the window including the first point is marked True. If 'ail' (the or every point in the window including the first point is marked True. If 'ail' (the or every point in the window including the first point is marked True. True for each value that is part of a stale sequence of data Returns is ValueError – If window < 2 or mark is not one of 'tail', 'end', or 'ail'. Notes Copyright (t) 2019 SolarArbiter. See the file LICENSES/SOLARFORECASTARBITER_LICENSE at the top level directory of this distribution and at | Parameters rtol | and atol have the same meaning as in numpy.allclose(). |
| Return type: True for each value that is part of a stale sequence of data Return type: Series Naises: ValueError – If window < 2 or mark is not one of 'tail', 'end', or 'ail'. | Parameters: | x (Series) - data to be processed window (<i>int</i>, <i>default</i> 0) - number of consecutive values which, if unchanges1, indicates stated data rtol (<i>float</i>, <i>default</i> 1e-5) - relative tolerance for detecting a change in data values atol (<i>float</i>, <i>default</i> 1e-3) - absolute tolerance for detecting a change in data values mark (<i>str</i>, <i>default</i> 1:-3) - absolute tolerance for detecting a change in data values mark (<i>str</i>, <i>default</i> 1:-3) - absolute tolerance for detecting a change in data values in ark (<i>str</i>, <i>default</i> 1:-3) - absolute tolerance for detecting a change in data values in ark (<i>str</i>, <i>default</i> 1:-3) - 1 - How much of the window to mark True when a sequence of stale values is detected. Can one be of 'tail', 'end', or 'all'. if 'tail' (the default) then every point in the window <i>including</i> the sequence are marked <i>True</i>. if 'all' then every point in the window <i>including</i> the first point is marked <i>True</i>. |
| 'all'. Notes Copyright (c) 2019 SolarArbiter. See the file LICENSES/SOLARFORECASTARBITER_LICENSE at the top level directory of this distribution and at https://github.com/pytlib/pvanalytics/blob/master/LICENSES/SOLARFORECASTARBITER_LICENSE | Returns: Return type: Raises: | True for each value that is part of a stale sequence of data Series ValueError – If window < 2 or mark is not one of 'tail', 'end', or |
| Copyright (c) 2019 SolarArbiter. See the file LICENSES/SOLARFORECASTARBITER_LICENSE at the top level directory of this distribution and at https://githuo.com/pvib/pvanalytics/blob/master/LICENSES/SOLARFORECASTARBITER_LICENSE | Notos | 'all'. |
| | Copyright (c) 20 LICENSES/SOL distribution and https://github.co | 19 SolarArbiter. See the file ARFORECASTARBITER_LICENSE at the top level directory of this at m/pvlib/pvanalytics/blob/master/LICENSES/SOLARFORECASTARBITER_LICENSE |
| | xamples u | sing |
| xamples using | anarycic | o.duarrch.Eabo.orare_varneo_nrii |
| xamples using vanalytics.quality.gaps.stale_values_diff | | |
| ixamples using vanalytics.quality.gaps.stale_values_diff | | |
| xamples using vanalytics.quality.gaps.stale_values_diff | Stale Data Perio | ds |

Function description for

pvanalytics.quality.stale_values_diff

https://pvanalytics.readthedocs.io/en/stable/api.html

Automated Testing

- Comprehensive unit-testing for all package functions
 - ~100% test coverage
 - Uses Pytest and Coveralls
- Since package is in its infancy, no speed benchmarks have been taken (yet!)

30 checks passed Image: Constraint of the state s

Package checks required to pass before merging PR

Int and test passing coverage 100% DOI 10.5281/zenodo.6110569

Current test coverage

Community growth

- Github
 - 88 completed pull requests
 - Code contributions from 6 people (see lower right)
- Lots of opportunity to increase community growth as PVAnalytics is still in its infancy
- You can contribute!
 - Generate issues for features you'd like to see, add code via our PR process, etc.



Github stars over time



Special thanks to all our contributors!

PVAnalytics v0.1.3 and Beyond

- No expected ETA for next release but we're actively working on new functions/documentation
- Future version features:
 - Daylight savings time (DST) and time-drift detection algorithms for time series
 - Adding plotting module to easily validate time-series data visually

Thank you!

www.nrel.gov

kirsten.perry@nrel.gov

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 Number 38258. 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.

