The Journal of Open Source Software

atlite: A Lightweight Python Package for Calculating Renewable Power Potentials and Time Series

Fabian Hofmann[1], **Johannes Hampp**[2], **Fabian Neumann**[3], **Tom Brown**[4], **and Jonas Hörsch**[5]

1 Frankfurt Institute for Advanced Studies **2** Center for international Development and Environmental Research, Justus-Liebig University Giessen **3** Institute for Automation and Applied Informatics, Karlsruhe Institute of Technology (KIT) **4** Institute of Energy Technology, Technical University of Berlin **5** Climate Analytics gGmbH, Berlin

DOI: 10.21105/joss.03294

Software

- Review
- Repository
- Archive

Editor: Tim Tröndle

Reviewers:

- @kanderso-nrel
- @crvernon

Submitted: 21 April 2021
Published: 24 June 2021

Summary

Renewable energy sources are likely to build the backbone of the future global energy system. One important key to a successful energy transition is to analyse the weather-dependent energy outputs of existing and eligible renewable resources. `atlite` is an open Python software package for retrieving global historical weather data and converting it to power generation potentials and time series for renewable energy technologies like wind turbines or solar photovoltaic panels based on detailed mathematical models. It further provides weather-dependent output on the demand side like building heating demand and heat pump performance. The software is optimized to aggregate data over multiple large regions with user-defined weightings based on land use or energy yield.

Statement of need

Deriving weather-based time series and maximum capacity potentials for renewables over large regions is a common problem in energy system modelling. Websites with exposed open APIs such as renewables.ninja (Pfenninger & Staffell, 2016; Staffell & Pfenninger, 2016) exist for such purpose but are difficult to use for local execution, e.g. in cluster environments, and restricted to non-commercial use. Further, by design, they neither expose the underlying datasets nor methods for deriving time series, here referred to as conversion functions/methods. This makes them unsuited for utilizing different weather datasets or exploring alternative conversion functions. The pvlib (Holmgren et al., 2018) is suited for local execution and allows interchangeable input data but is specialized to PV systems only and intended for single location modelling. Other packages like the Danish REatlas (Andresen et al., 2015) face obstacles with accessibility, are based on proprietary code, miss documentation and are restricted in flexibility regarding their inputs.

The purpose of `atlite` is to fill this gap and provide an open, community-driven library. `atlite` was initially built as a lightweight alternative to REatlas and has evolved further to contain multiple additional features. `atlite` is designed with extensibility in mind for new renewable technologies and conversion methods. An abstraction layer for weather datasets enables interchangability of the underlying datasets. By leveraging the Python packages xarray (Hoyer & Hamman, 2017), dask (Dask Development Team, 2016) and rasterio (Gillies & others, 2021), `atlite` makes use of parallelization and memory efficient backends thus performing well even on large datasets.

Basic Concept

The starting point of most `atlite` functionalities is the `atlite.Cutout` class. It serves as a container for a spatio-temporal subset of one or more topology and weather datasets. Since such datasets are typically global and span multiple decades, the Cutout class allows `atlite` to reduce the scope to a more manageable size. As illustrated in Figure 1, a typical workflow consists of three steps: Cutout creation, Cutout preparation and Cutout conversion.

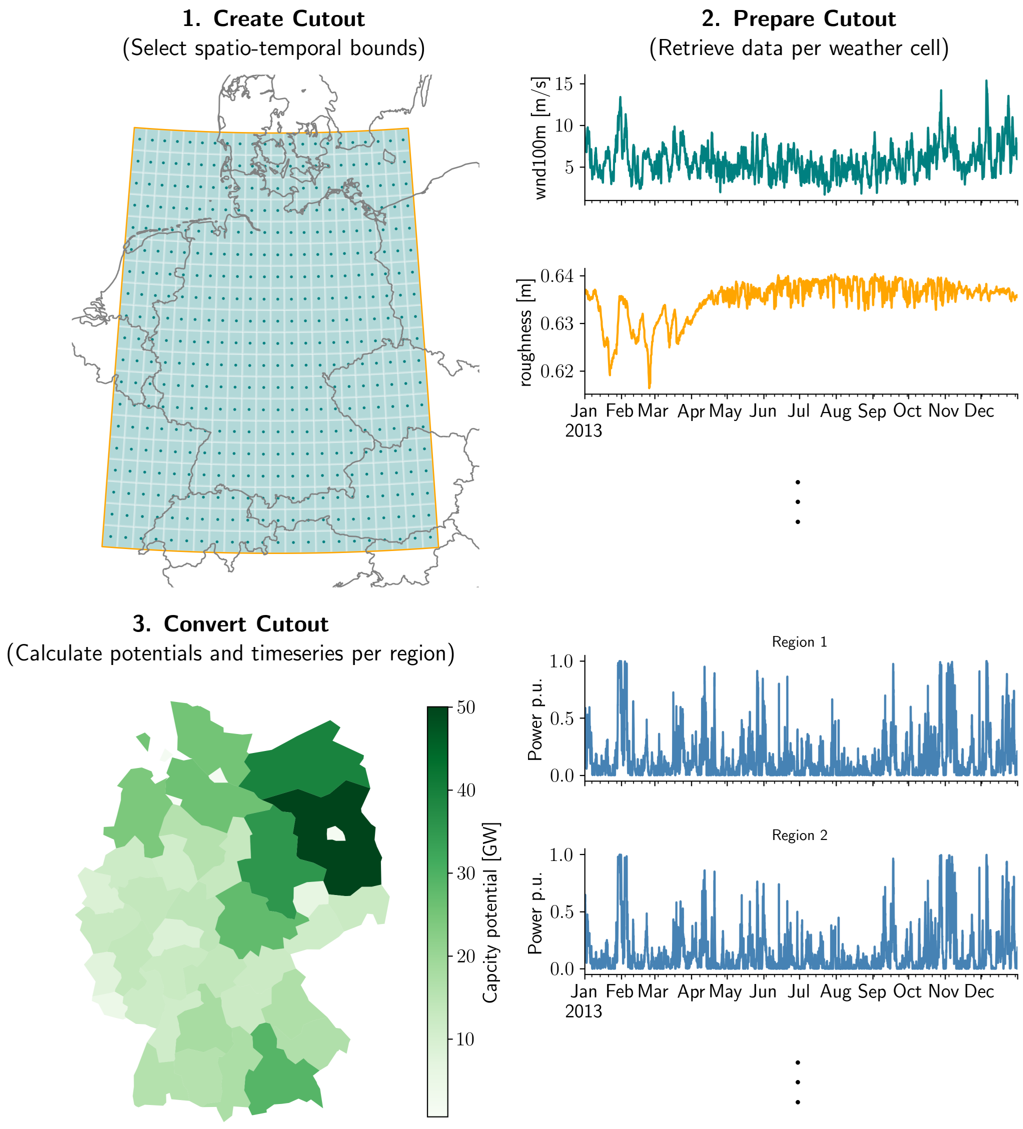

Figure 1: A typical workflow in `atlite` consists of the three steps: 1. Cutout creation, 2. Preparation, 3. Conversion.

Cutout Creation and Preparation

The Cutout creation requires specifications of the geographical and temporal bounds, the path of the associated `netcdf` file to be created and the data source referred to as *module*.

Optionally, the temporal and spatial resolution may be adjusted. The default is set to 1 hour and 0.25° latitude times 0.25° longitude. So far, `atlite` supports three different *modules*:

1. ECMWF Reanalysis v5 (ERA5) provides various weather-related variables in an hourly resolution from 1950 onward on a spatial grid with a 0.25° x 0.25° resolution, most of which is reanalysis data. `atlite` automatically retrieves the raw data using the Climate Data Store (CDS) API after the initial set up by the user. When the requested data points diverge from the original grid, the API retrieves interpolated values based on the original grid data.
2. Heliosat (SARAH-2) provides satellite-based solar data in a 30 min resolution from 1983 to 2015 on a spatial grid ranging from -65° to +65° longitude/latitude with a resolution of 0.05° x 0.05°. In case of a diverging Cutout grid, a resampling function provided by `atlite` projects the data accordingly. The full dataset cannot be automatically retrieved and must be downloaded by the user beforehand.
3. GEBCO is a bathymetric dataset covering terrain heights on a 15 arc-second resolved spatial grid. Using an averaging resampling method, the data is projected to the Cutout resolution. The full dataset cannot be automatically retrieved and must be downloaded by the user beforehand.

Creating a Cutout triggers the program to initialize the grid cells and the coordinate system on which the data will lay. As indicated in Figure 1, the shapes of the grid cells are created such that their coordinates are centered in the middle. As soon as the Cutout preparation is executed, `atlite` retrieves/loads data variables, adds them to the Cutout and finally stores the Cutout in a `netcdf` file. `atlite` groups weather variables into *features*, which can be used as front-end keys for preparing a subset of the available weather variables. The following table shows the variable groups for all datasets.

feature	ERA5 variables	SARAH-2 variables	GEBCO variables
height	height		height
wind	wnd100m, roughness		
influx	influx_toa, influx_direct, influx_diffuse, albedo	influx_direct, influx_diffuse	
temperature	temperature, soil_temperature		
runoff	runoff		

A Cutout may combine features from different sources, e.g. 'height' from GEBCO and 'runoff' from ERA5. Future versions of atlite will likely introduce the possibility to retrieve explicit weather variables from the CDS API. Further, the climate projection dataset CORDEX, for which support was dropped with the latest release `v0.2` due to compatibility issues, is likely to be reintroduced.

Conversion Functions

`atlite` currently offers conversion functions for deriving time series and static potentials from Cutouts for the following types of renewables:

- **Solar photovoltaic** – Two alternative solar panel models are provided based on Huld et al. (2010) and Beyer et al. (2004), both of which use the clear sky model from Reindl et al. (1990) and a solar azimuth and altitude position tracking based on Kalogirou (2009), Michalsky (1988) and Sproul (2007) combined with a surface orientation algorithm following Sproul (2007). Optionally, optimal latitude heuristics from Landau (2017) are supported.

- **Solar thermal collector** – Low-temperature heat for space or district heating is implemented based on the formulation in Henning & Palzer (2014), which combines average global radiation with storage losses dependent on the current outside temperature.
- **Wind turbine** – The wind turbine power output is calculated from down-scaled wind speeds at hub height using either a custom power curve, one of 16 predefined wind turbine configurations, or any of those listed in the OEP Wind Turbine Library. Optionally, convolution with a Gaussian kernel for region-specific calibration given real-world reference data as presented by Andresen et al. (2015) is supported.
- **Hydro run-off power** – A heuristic approach uses surface run-off weather data (e.g. from rainfall or melting snow) which is normalized to match reported energy production figures by the EIA. The resulting time series are optionally weighted by the height of the run-off location and may be smoothed for a more realistic representation.
- **Hydro reservoir and dam power** – Following Liu et al. (2019) and Lehner & Grill (2013), run-off data is aggregated to and collected in basins which are obtained and estimated in their size with the help of the HydroSHEDS dataset.
- **Heating demand** – Space heating demand is obtained with a simple degree-day approximation where the difference between outside ground-level temperature and a reference temperature scaled by a linear factor yields the desired estimate.

The conversion functions are highly flexible and allow the user to calculate different types of outputs, which arise from the set of input arguments. In energy system models, network nodes are often associated with geographical regions which serve as catchment areas for electric loads, renewable energy potentials and more. As indicated in the third step of Figure 1, `atlite`'s conversion functions allow projecting renewable time series on a set of regions. Therefore, `atlite` internally computes the Indicator Matrix **I** with values $I_{r,x,y}$ representing the per-unit overlap between region r and the grid cell at (x, y). Then, the resulting time series $\varphi_r(t)$ for region r is given by

$$\varphi_r(t) = \sum_{x,y} I_{r,x,y}\, \varphi_{x,y}(t),$$

where $\varphi_{x,y}(t)$ is the converted time series of grid cell (x, y). Further, the user may define custom weightings $\lambda_{x,y}$ of the grid cells, referred to as layout, representing, for instance, the spatial distribution of installed capacities within a region, which modifies the above equation to

$$\varphi_r(t) = \sum_{x,y} I_{r,x,y}\, \lambda_{x,y}\, \varphi_{x,y}(t).$$

The conversion functions may optionally return the per-unit time series $\tilde{\varphi}_r(t) = \varphi_r(t)/c_r$ where c_r is the installed capacity per region given by

$$c_r = \sum_{x,y} I_{r,x,y}\, \lambda_{x,y},$$

which may be returned as an output as well.

Land-Use Restrictions

Land-use restrictions limit the deployment of renewables infrastructure. Wind turbines, for example, may only be placed in eligible places which have to fulfill general and country-specific requirements, e.g. being outside of protected areas or at a sufficient distance to residential areas.

`atlite` provides a performant, parallelized implementation to calculate land-use availabilities within all grid cells of a Cutout. As illustrated in Figure 2, the entries $A_{r,x,y}$ of an Availability Matrix **A** indicate the overlap of the eligible area of region r with grid cell at (x, y). Note that this is analogous to the Indicator Matrix **I** but with reduced area. The user can exclude geometric shapes or geographic rasters of arbitrary projection, like specific codes of the Corine Land Cover (CLC) database. To determine capacity expansion potentials per region, `atlite` does not use an explicit placement algorithm (e.g. for wind turbines), but the product of available area and allowed deployment density. The implementation is inspired by the GLAES (Ryberg et al., 2018) software package, which itself is no longer maintained and incompatible with newer versions of the underlying GDAL software.

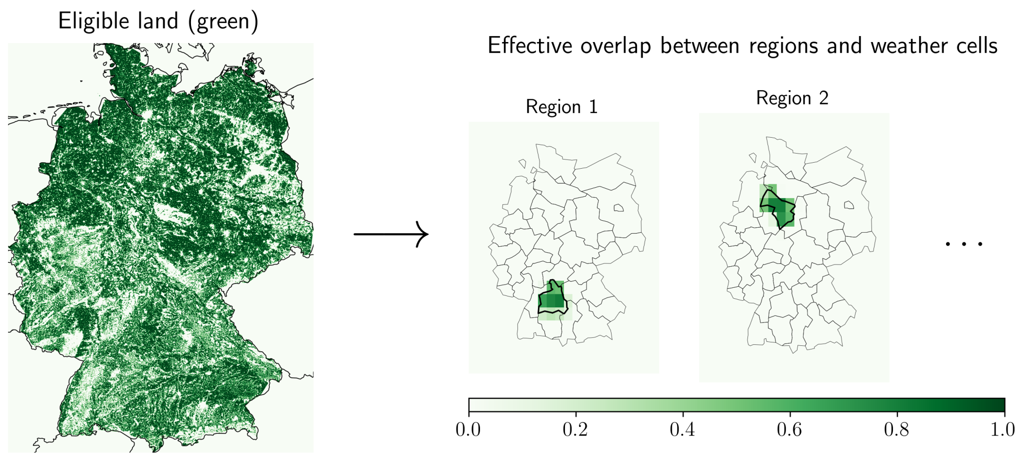

Figure 2: Example of a land-use restrictions calculated with `atlite`. The left side shows a highly-resolved raster with available areas in green. In this example all urban and forest-like sites are excluded areas, drawn in white. The right side visualizes exemplary entries per region r of the resulting Availability Matrix **A**.

Related Research

`atlite` is used by several research projects and groups. The PyPSA-Eur workflow (Hörsch et al., 2018) is an open model dataset of the European power system which exploits the full potential of `atlite` including Cutout preparation and conversion to wind, solar and hydro reservoir time series with restricted land-use availabilities. The sector-coupled extension PyPSA-Eur-sec (Brown et al., 2018) calculates heat-demand profiles as well as heat pump coefficients with `atlite`. The Euro Calliope studied in Tröndle et al. (2020) uses `atlite` to generate hydroelectricity time series from reservoirs. The interactive tool model.energy also employs the `atlite` libary.

Availability

Stable versions of the `atlite` package are available for Linux, MacOS and Windows via `pip` in the Python Package Index (PyPI) and for `conda` on conda-forge. Upstream versions and development branches are available in the projects GitHub repository. Documentation including examples are available on Read the Docs. The `atlite` package is released under GPLv3 and welcomes contributions via the project's GitHub repository.

Acknowledgements

We thank all contributors who helped to develop `atlite`. Fabian Hofmann is funded by the German Federal Ministry for Education and Research under grant nr. FKZ03EI1028A (EnergiesysAI).

References

Andresen, G. B., Søndergaard, A. A., & Greiner, M. (2015). Validation of Danish wind time series from a new global renewable energy atlas for energy system analysis. *Energy*, *93*, 1074–1088. https://doi.org/10.1016/j.energy.2015.09.071

Beyer, H. G., Heilscher, G., & Bofinger, S. (2004). A robust model for the MPP performance of different types of PV-modules applied for the performance check of grid connected systems. *EuroSun. Freiburg*.

Brown, T., Schlachtberger, D., Kies, A., Schramm, S., & Greiner, M. (2018). Synergies of sector coupling and transmission extension in a cost-optimised, highly renewable European energy system. *Energy*, *160*, 720–739. https://doi.org/10.1016/j.energy.2018.06.222

Dask Development Team. (2016). *Dask: Library for dynamic task scheduling* [Manual]. https://dask.org

Gillies, S., & others. (2021). *Rasterio: Geospatial raster i/o for Python programmers* (Version 1.2) [Computer software]. Mapbox. https://github.com/mapbox/rasterio

Henning, H.-M., & Palzer, A. (2014). A comprehensive model for the German electricity and heat sector in a future energy system with a dominant contribution from renewable energy technologiesPart I: Methodology. *Renewable and Sustainable Energy Reviews*, *30*, 1003–1018. https://doi.org/10.1016/j.rser.2013.09.012

Holmgren, W. F., Hansen, C. W., & Mikofski, M. A. (2018). Pvlib python: A python package for modeling solar energy systems. *Journal of Open Source Software*, *3*(29), 884. https://doi.org/10.21105/joss.00884

Hoyer, S., & Hamman, J. J. (2017). Xarray: N-D labeled Arrays and Datasets in Python. *Journal of Open Research Software*, *5*, 10. https://doi.org/10.5334/jors.148

Hörsch, J., Hofmann, F., Schlachtberger, D., & Brown, T. (2018). PyPSA-Eur: An open optimisation model of the European transmission system. *Energy Strategy Reviews*, *22*, 207–215. https://doi.org/10.1016/j.esr.2018.08.012

Huld, T., Gottschalg, R., Beyer, H. G., & Topič, M. (2010). Mapping the performance of PV modules, effects of module type and data averaging. *Solar Energy*, *84*(2), 324–338. https://doi.org/10.1016/j.solener.2009.12.002

Kalogirou, S. (2009). *Solar energy engineering: Processes and systems*. Elsevier/Academic Press. ISBN: 978-0-12-374501-9

Landau, C. R. (2017). *Optimum tilt of solar panels*. https://www.solarpaneltilt.com/

Lehner, B., & Grill, G. (2013). Global river hydrography and network routing: Baseline data and new approaches to study the world's large river systems. *Hydrological Processes*, *27*(15), 2171–2186. https://doi.org/10.1002/hyp.9740

Liu, H., Andresen, G. B., Brown, T., & Greiner, M. (2019). A validated high-resolution hydro power time-series model for energy systems analysis. *MethodsX*, *6*, 1370–1378. https://doi.org/10.1016/j.mex.2019.05.024

Michalsky, J. J. (1988). The Astronomical Almanac's algorithm for approximate solar position (1950). *Solar Energy*, *40*(3), 227–235. https://doi.org/10.1016/0038-092x(88)90045-x

Pfenninger, S., & Staffell, I. (2016). Long-term patterns of European PV output using 30 years of validated hourly reanalysis and satellite data. *Energy*, *114*, 1251–1265. https://doi.org/10.1016/j.energy.2016.08.060

Reindl, D. T., Beckman, W. A., & Duffie, J. A. (1990). Diffuse fraction correlations. *Solar Energy*, *45*(1), 1–7. https://doi.org/10.1016/0038-092X(90)90060-P

Ryberg, D., Robinius, M., & Stolten, D. (2018). Evaluating land eligibility constraints of renewable energy sources in europe. *Energies*, *11*(5), 1246. https://doi.org/10.3390/en11051246

Sproul, A. B. (2007). Derivation of the solar geometric relationships using vector analysis. *Renewable Energy*, *32*(7), 1187–1205. https://doi.org/10.1016/j.renene.2006.05.001

Staffell, I., & Pfenninger, S. (2016). Using bias-corrected reanalysis to simulate current and future wind power output. *Energy*, *114*, 1224–1239. https://doi.org/10.1016/j.energy.2016.08.068

Tröndle, T., Lilliestam, J., Marelli, S., & Pfenninger, S. (2020). Trade-Offs between Geographic Scale, Cost, and Infrastructure Requirements for Fully Renewable Electricity in Europe. *Joule*, *4*(9), 1929–1948. https://doi.org/10.1016/j.joule.2020.07.018