The Journal of Open Source Software

PyHeatDemand - Processing Tool for Heat Demand Data

Alexander Jüstel [1,2] and **Frank Strozyk** [2]

1 RWTH Aachen University, Geological Institute, Wüllnerstraße 2, 52062 Aachen, Germany **2** Fraunhofer IEG, Fraunhofer Research Institution for Energy Infrastructures and Geothermal Systems IEG, Kockerellstraße 17, 52062 Aachen, Germany

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor:

Submitted: 21 October 2023
Published: unpublished

Summary

PyHeatDemand is an open-source Python package for processing and harmonizing multi-scale-multi-type heat demand input data for

constructing local to transnational harmonized heat demand maps (rasters). Knowledge about the heat demand (MWh/area/year) of a respective building, district, city, state, country, or even on a continental scale is crucial for an adequate heat demand analysis or planning for providing power plant capacities. Mapping of the heat demand may also identify potential areas for new district heating networks or even geothermal power plants for climate-friendly heat production.

The aim of **PyHeatDemand** is to provide processing tools for heat demand input data of various categories on various scales. This includes heat demand input data provided as rasters or gridded polygons, heat demand input data associated with administrative areas (points or polygons), with building footprints (polygons), with street segments (lines), or with addresses directly provided in

MWh but also as gas usage, district heating usage, or sources of heat. It is also possible to calculate the heat demand based on a set of cultural data sets (building footprints, height of the buildings, population density, building type, etc.). The study area is first divided into a coarse mask before heat demands are calculated and harmonized for each cell with the size of the target resolution (e.g. 100 m x 100 m for states). We hereby make use of different spatial operations implemented in the GeoPandas and Shapely packages. The final heat demand map will be created utilizing the Rasterio package. Next to processing tools for the heat demand input data, workflows for analyzing the final heat demand map through the Rasterstats package are provided.

PyHeatDemand was developed as a result of works carried out within the Interreg NWE project DGE Rollout (Rollout of Deep Geothermal Energy).

Statement of need

Space and water heating for residential and commercial buildings amount to a third of the European Union's total final energy consumption. Approximately 75% of the primary energy and 50% of the thermal energy are still produced by burning fossil fuels, leading to high greenhouse gas emissions in the heating sector. The transition from centralized fossil-fueled district heating systems such as coal or gas power plants to district heating systems sourced by renewable energies such as geothermal energy or more decentralized individual solutions for city districts makes it necessary to map the heat demand for a more accurate planning of power plant capacities. In addition, heating and cooling plans become necessary according to directives of the European Union regarding energy efficiency to reach its aim of reducing greenhouse gas emissions by 55% of the 1990-levels by 2030.

Evaluating the heat demand (usually in MWh = Mega Watt Hours) on a national or regional scale, including space and water heating for each apartment or each building for every day of a year separately is from a perspective of resolution (spatial and temporal scale) and computing power not feasible. Therefore, heat demand maps summarize the heat demand on a lower spatial resolution (e.g. 100 m x 100 m raster) cumulated for one year (lower temporal resolution) for different sectors such as the residential and tertiary sectors. Maps for the industrial heat demand are not available as the input data is not publicly available or can be deduced from cultural data. Customized solutions are therefore necessary for this branch to reduce greenhouse gas emissions. Heat demand input values for the residential and commercial sectors are easily accessible and assessable. With the new directives regarding energy efficiency, it becomes necessary for every city or commune to evaluate their heat demand. And this is where **PyHeatDemand** comes into place. Combining the functionality of well-known geospatial Python libraries, the open-source package **PyHeatDemand** provides tools for public entities, researchers, or students for processing heat demand input data associated with an administrative area (point or polygon), with a building footprint (polygon), with a street segment (line), or with an address directly provided in MWh but also as gas usage, district heating usage, or other sources of heat. The resulting heat demand map data can be analyzed using zonal statistics and can be compared to other administrative areas when working on regional or national scales. If heat demand maps already exist for a specific region, they can be analyzed using tools within **PyHeatDemand**. With **PyHeatDemand**, it has never been easier to create and analyze heat demand maps.

PyHeatDemand Functionality

Processing Heat Demand Input Data

Heat demand maps can be calculated using either a top-down approach or a bottom-up approach (Fig. 1). For the top-down approach, aggregated heat demand input data for a certain area will be distributed according to higher resolution data sets (e.g. population density, landuse, etc.). In contrast to that, the bottom-up approach allows aggregating heat demand of higher resolution data sets to a lower resolution (e.g. from building level to a 100 m x 100 m raster).

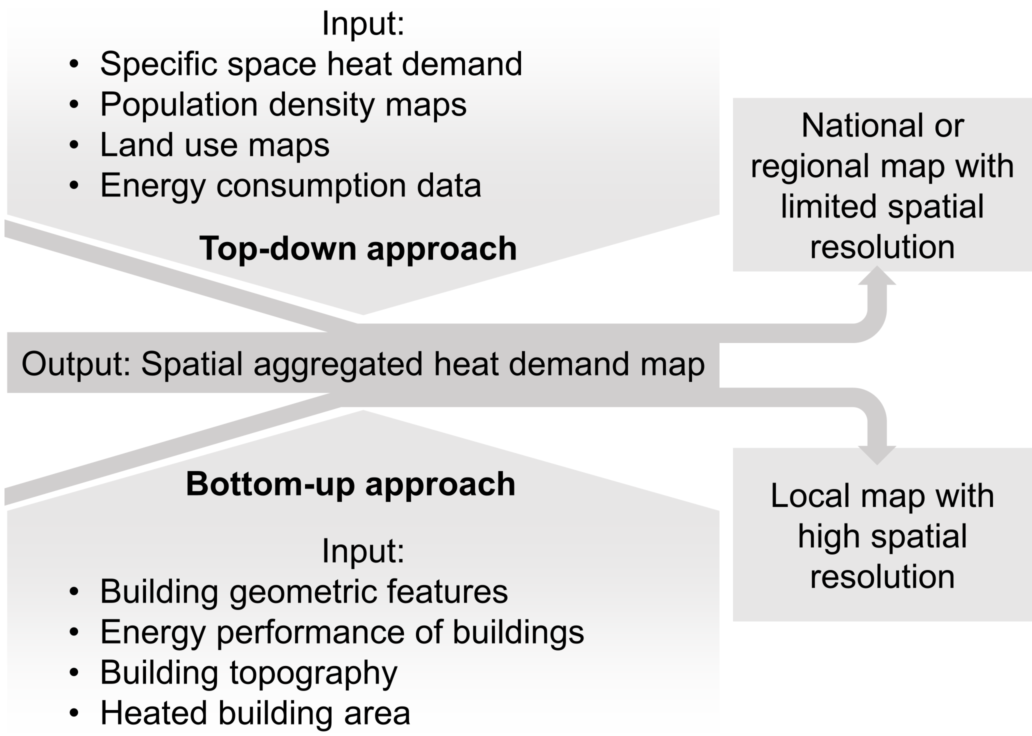

Figure 1: Input and output data for top-down and bottom-up approaches. Note, that the resulting spatial resolution can be the same for both approaches, but the spatial value of information is usually lower using a top-down approach.

PyHeatDemand processes geospatial data such as vector data (points, lines, polygons), raster data or address data. Therefore, we make use of the functionality implemented in well-known geospatial packages such as GeoPandas (Jordahl et al., 2021), Rasterio (Gillies & others, 2013), Rasterstats (Perry, 2023), GeoPy (Esmukov & others, 2023), or OSMnx (Boeing, 2017) and their underlying dependencies such as Shapely (Gillies & others, 2007), Pandas (The pandas development team, 2021), or NumPy (Harris et al., 2020). In particular, we are utilizing the powerful implementation of Spatial Indices in GeoPandas allowing for processing speed-ups by orders of magnitudes compared to performing regular overlays and spatial joins for the processing of heat demand input data.

The creation of a heat demand map follows a general workflow (Fig. 2) followed by a data-category-specific workflow for five defined input data categories (Fig. 4 & 5). The different input data categories are listed in the table below.

Data category	Description
1	HD data provided as (e.g. $100 * 100\ m^2$) raster or polygon grid with the same or in a different coordinate reference system
2	HD data provided as building footprints or street segments
3	HD data provided as a point or polygon layer, which contains the sum of the HD for regions of official administrative units
4	HD data provided in other data formats such as HD data associated with addresses

Data category	Description
5	No HD data available for the region

Depending on the scale of the heat demand map (local, regional, national, or even transnational), a global polygon mask is created from provided administrative boundaries with a cell size of 10 km by 10 km, for instance, and the target coordinate reference system. This mask is used to divide the study area into smaller chunks for a more reliable processing as only data within each mask will be processed separately. If necessary, the global mask will be cropped to the extent of the available heat demand input data and populated with polygons having already the final cell size such as 100 m x 100 m. For each cell, the cumulated heat demand in each cell will be calculated. The final polygon grid will be rasterized and merged with adjacent global cells to form a mosaic, the final heat demand map. If several input datasets are available for a region, i.e. different sources of energy, they can either be included in the calculation of the heat demand or the resulting rasters can be added to a final heat demand map. In addition to a uniform polygon mask with equal cell sizes, **PyHeatDemand** also offers the possibility to refine the polygon mask based on the centroid density of building footprints (Fig. 3) based on a simple QuadTree algorithm (Finkel, 1974). This allows to identify areas of high heat demand originating from a small number of buildings and areas of high heat demand from a large number of small buildings.

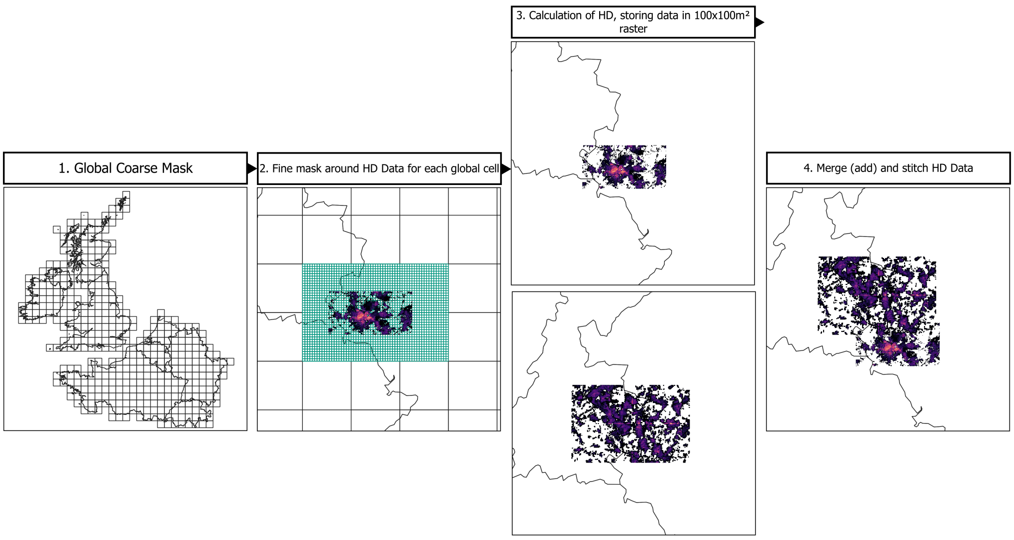

Figure 2: The main steps from creating a coarse matrix to a fine matrix to calculating the final heat demand data to merging and rasterizing the data.

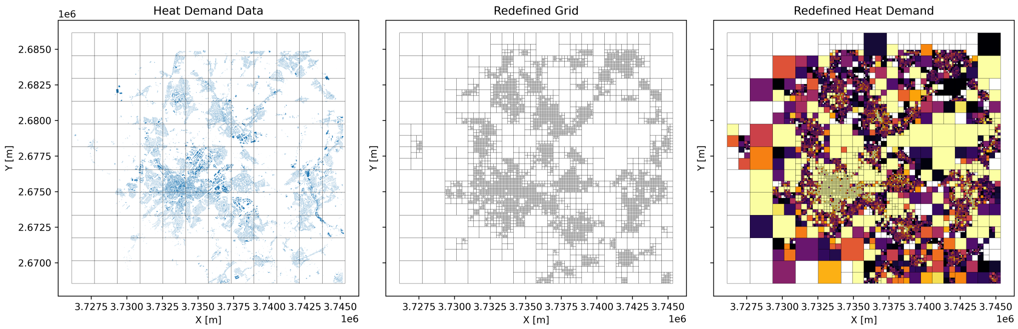

Figure 3: Grid refinement using heat demand data density.

The data processing for data categories 1 and 2 are very similar (Fig. 4) and correspond to a bottom-up approach. In the case of a raster for category 1, the raster is converted into gridded polygons. Gridded polygons and building footprints are treated equally (Fig. 5 top). The polygons containing the heat demand data are, if necessary, reprojected to the coordinate reference system and are overlain with the local mask (e.g. 100 m x 100 m cells). This cuts each heat demand polygon with the respective mask polygon. The heat demand of each subpolygon is proportional to its area compared to the area of the original polygon. The heat demand for all subpolygons in each cell is aggregated to result in the final heat demand for this cell.

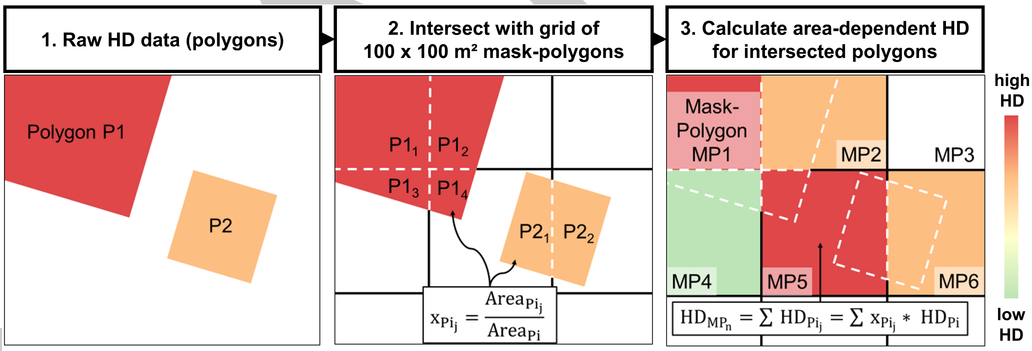

Figure 4: The main steps of the methodology to process the provided HD polygons for the heat demand data categories 1 and 2.

The data processing for data category 3 corresponds to a top-down approach (Fig. 5 bottom). The heat demand represented as points for an administrative unit will be distributed across the area using higher-resolution data sets. In the case illustrated below, the distribution of Hotmaps data (Fallahnejad, 2019) is used to distribute the available heat demands for the given administrative areas. For each administrative area, the provided total heat demand will distributed according to the share of each Hotmaps cell compared to the total Hotmaps heat demand of the respective area. The provided heat demand is now distributed across the cells and will treated from now on as category 1 or 2 input data to calculate the final heat demand map.

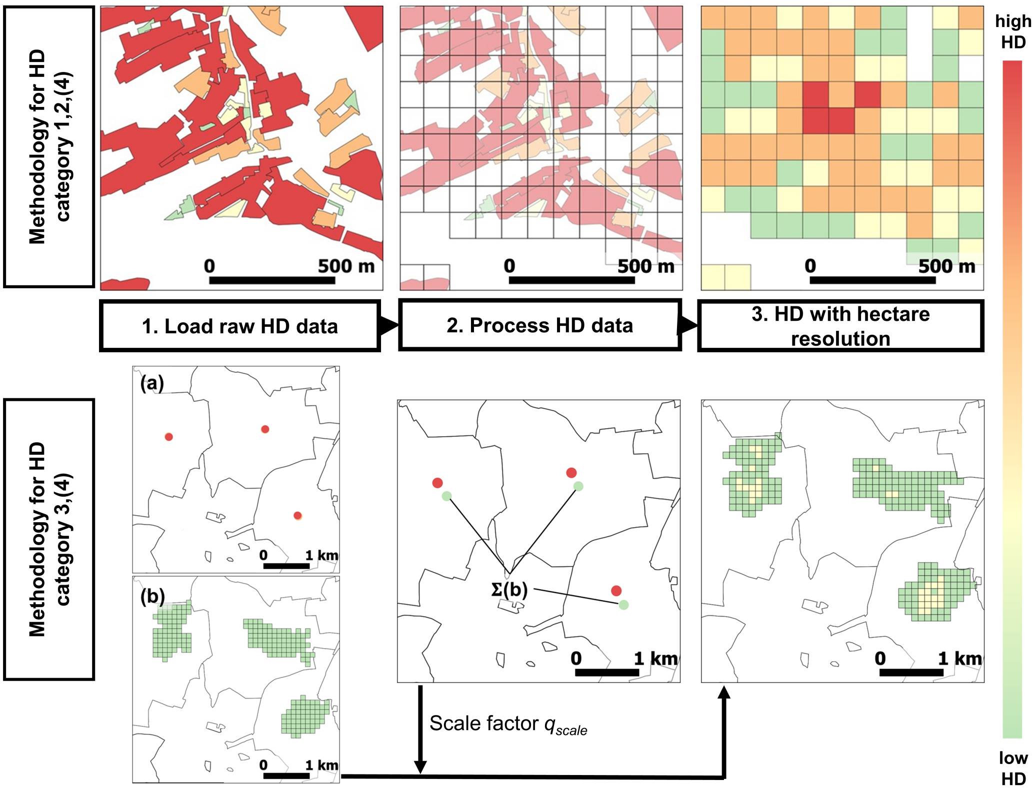

Figure 5: The main steps of the methodology to process the provided HD polygons for the heat demand data category 2 (top) and category 3 (bottom).

The data processing for data category 4 corresponds to a bottom-up approach. Here, the addresses will be converted using the GeoPy geolocator to coordinates. Based on these, the building footprints are extracted from OpenStreet Maps using OSMnx. From there on, the data will be treated as data category 2.

If no heat demand input data is available, the heat demand can be estimated using cultural data such as population density, landuse, and building-specific heat usage (Meha et al., 2020; Novosel et al., 2020) which will be implemented in a later development stage.

Processing Heat Demand Map Data

Heat demand maps may contain millions of cells. Evaluating each cell would not be feasible. Therefore, **PyHeatDemand** utilizes the rasterstats package (Perry, 2023) returning statistical values of the heat demand map for further analysis and results reporting.

State of the field

Python libraries for calculating heat demands are sparse, especially for aggregating heat demand on various scales and categories. While UrbanHeatPro (Molar-Cruz, 2021) utilizes a bottom-up approach to calculate heat demand profiles for urban areas, the Heat package by Malcolm Peacock (Peacock, 2023) generates heat demand time series from weather for EU countries. Repositories containing processing code for larger transnational heat demand projects like Hotmaps and Heat Roadmap Europe are unknown.

PyHeatDemand Outlook

The development and maintenance of **PyHeatDemand** will continue in the future. This will include adding bottom-up workflows based on building specifics to calculate the heat demand. In addition, we welcome contributions of users in the form of questions on how to use **PyHeatDemand**, bug reports, and feature requests.

PyHeatDemand Resources

The following resources are available for **PyHeatDemand**

- PyHeatDemand Github Repository
- PyHeatDemand Documentation
- DGE Rollout Webviewer

Acknowledgements

We would like to thank the open-source community for providing and constantly developing and maintaining great tools that can be combined and utilized for specific tasks such as working with heat demand data. The original codebase was developed within the framework of the Interreg NWE project DGE Rollout (Rollout for Deep Geothermal Energy) by Eileen Herbst and Elias Khashfe (Herbst, 2021). It was rewritten and optimized for **PyHeatDemand**.

References

Boeing, G. (2017). OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. *Computers, Environment and Urban Systems*, *65*, 126–139. https://doi.org/10.1016/j.compenvurbsys.2017.05.004

Esmukov, K., & others. (2023). *GeoPy: Geocoding library for python*. GeoPy. https://github.com/geopy/geopy

Fallahnejad, M. (2019). *Hotmaps-data-repository-structure*. Hotmaps-Wiki. https://wiki.hotmaps.eu/en/Hotmaps-open-data-repositories

Finkel, B., R. A. (1974). Quad trees a data structure for retrieval on composite keys. *Acta Informatica*, *4*, 1–9. https://doi.org/10.1007/BF00288933

Gillies, S., & others. (2007). *Shapely: Manipulation and analysis of geometric objects*. toblerity.org. https://github.com/Toblerity/Shapely

Gillies, S., & others. (2013). *Rasterio: Geospatial raster i/o for Python programmers*. Mapbox. https://github.com/mapbox/rasterio

Harris, C. R., Millman, K. J., Walt, S. J. van der, Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., Kerkwijk, M. H. van, Brett, M., Haldane, A., R'ıo, J. F. del, Wiebe, M., Peterson, P., … Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, *585*(7825), 357–362. https://doi.org/10.1038/s41586-020-2649-2

Herbst, K., E. (2021). A heat demand map of north-west europe – its impact on supply areas and identification of potential production areas for deep geothermal energy. *GeoKarlsruhe*, *2021*, 1. https://doi.org/10.48380/dggv-j2wj-nk88

Jordahl, K., Bossche, J. V. den, Fleischmann, M., & et al. (2021). *Geopandas/geopandas: v0.9.0* (Version v0.9.0). Zenodo. https://doi.org/10.5281/zenodo.4569086

Meha, D., Novosel, T., & Duić, N. (2020). Bottom-up and top-down heat demand mapping methods for small municipalities, case gllogoc. *Energy*, *199*, 117429. https://doi.org/10.1016/j.energy.2020.117429

Molar-Cruz, A. (2021). *UrbanHeatPro - a bottom-up model for the simulation of heat demand profiles of urban areas*. Chair of Renewable; Sustainable Energy Systems. https://github.com/tum-ens/UrbanHeatPro/tree/master

Novosel, T., Pukšec, T., Duić, N., & Domac, J. (2020). Heat demand mapping and district heating assessment in data-pour areas. *Renewable and Sustainable Energy Reviews*, *131*, 109987. https://doi.org/10.1016/j.rser.2020.109987

Peacock, M. (2023). *Heat - generation of heat demand time series from weather*. Malcolm Peacock. https://github.com/malcolmpeacock/heat

Perry, M. (2023). *Rasterstats: Summary statistics of geospatial raster datasets based on vector geometries*. Rasterstats. https://github.com/perrygeo/python-rasterstats

The pandas development team. (2021). *Pandas-dev/pandas: pandas* (Version v1.2.3). Zenodo. https://doi.org/10.5281/zenodo.4572994