Utilisation of Unexploited Industrial Waste Energy
Documentation Model
CS1 MAN x ETH

Hanne, Giulia, Laura, Marco, Florian

May 8, 2024

Contents

1 Overview

Our model is a Python-based tool designed for analyzing and optimizing the recovery of waste heat from various industrial and data center sources. Developed at ETH Zurich for the Case Study in the MEST programme, this tool facilitates the integration of MAN heat pumps into district heating systems by assessing the potential for waste heat utilization through spatial, thermal, and power compatibility analyses.

2 Repository Structure

The repository consists of two main Python scripts:

1. **main.py**: Orchestrates the workflow of the tool, handling data input, sensitivity analysis, filtering processes, and output generation.
2. **mapping.py**: Contains the utility functions required for data processing, including data loading, filtering, and spatial combination analyses.

3 Main Modules Description

3.1 main.py

This script acts as the entry point for running the waste heat potential calculations. It initializes the data, applies the necessary filters, executes the sensitivity analysis if enabled, and outputs the results.

3.1.1 Key Components

- **Imports**: Standard data handling (`pandas`), Date and time operations (`datetime`), Spatial and mapping functions from `mapping.py`.
- **Program Options**: Toggle sensitivity analysis, Define sensitivity analysis parameters.
- **File Paths**: Paths to various input data files, such as sources, sinks, and heat pump parameters.
- **Main Workflow**:
  - Reading input data files.
  - Conditionally performing sensitivity analysis.
  - Filtering data based on heat pump compatibility.
  - Spatial combination of heat sources and sinks.
  - Scoring and plotting results.
  - Writing output files.

3.1.2 Sensitivity Analysis

Depending on the sensitivity parameter set (e.g., minimum sink temperature), the script adjusts the operational parameters and re-runs the analysis to understand the impact of these variations on the potential of waste heat utilization.

3.2 mapping.py

This module provides a suite of functions used to manipulate and analyze the spatial and power data of heat sources and sinks.

3.2.1 Functions

- `read_input`: Loads data from Excel files into pandas DataFrames. Handles various types of sources including unconventional sources, data centers, and industries. The input data can be varied. New input data has to have the same format as our excel files. Specific units and column names are described in section 4.
- **Filtering Functions**:
  - `filter_single_temperature` and `filter_single_power`: Filter sources and sinks based on temperature and power criteria to ensure compatibility with heat pumps.
  - `combine_sinks_with_sources`: Spatially combines sources and sinks based on proximity and outputs potential combinations. Here, a buffer is created around each sink, using geopandas, and sources inside this buffer are the candidates for combinations to this sink.
  - `filter_temperature_matching` and `filter_power_matching`: Further refine combinations based on thermal and power matching criteria, ensuring that the combinations are viable for heat pump application.
- `score`: Scores each combination based on temperature and power matching coefficients to rank their effectiveness.
- `write_output`: Writes the analysis results to Excel and text files, providing a detailed account of viable heat recovery setups.
- `render_map`: Generates a geographic plot of the sources and sinks, illustrating the spatial relationships and highlighting potential recovery opportunities.

4 Input Data

The model is specifically designed for the excel files, we used as input. To run the model, future dataframes need exactly the same structure. Therefore the structure is described precisely in the following.

4.1 Heat sinks:

The heatpump dataframe should contain data on potential heat sinks, for our case that was district heating networks. The format of the data can be seen in table 1.

Column name	Datatype	Unit	Comment
id	int		sink id (starting at 1, identical to index)
latitude	float		
longitude	float		
name	string		plant name
sector	string		sector name
T_min	float	°C	minimum temperature *
T_max	float	°C	minimum temperature *
P	float	W	power
energy	float	TJ	energy
heatpumps	int[]		define empty array
datasource	string		data source name
$datasource_id$	string		data source id

Table 1: Heat sinks input data

* if $T_min = T_max$ → create both columns and plug in the same value

4.2 Heat sources:

Heat sources are imported in the same format, which is shown in table 2. In our case, we imported several dataframes for different sectors. For instance a dataframe for conventional industries (such as steel, cement, or chemical industries) and another dataframe for unconventional sources, such as waste water treatment plants or metro stations.

Column name	Datatype	Unit	Comment
id	int		source id (starting at 1, identical to index)
latitude	float		
longitude	float		
name	string		plant name
sector	string		sector name
T_min	float	°C	minimum temperature *
T_max	float	°C	minimum temperature *
P	float	W	power
energy	float	TJ	energy
heatpumps	int[]		define empty array
datasource	string		data source name
$datasource_id$	string		data source id

Table 2: Heat sinks input data

* if $T_min = T_max \rightarrow$ create both columns and plug in the same value

4.3 Heat pump and sensitivity data:

The heatpump data is imported in a dataframe containing the variable, an initial value and an array of sensitivity analysis values, which can be seen in 3. The latter are the values varied during the sensitivity analysis, while the initial value is used when the variable is fixed. The sensitivity analysis is explained in more detail in the final report of the Case Study project.

Variable	**Unit**	**Value**	**Sensitivity analysis values**
$T_{source,min}$	$[°C]$	-10	$\{-10, 5, 20, 35, 50\}$
$T_{source,max}$	$[°C]$	150	$\{90, 105, 120, 135, 150, 165, 180, 195, 210\}$
$T_{sink,min}$	$[°C]$	0	$\{0, 15, 30, 45, 60, 75\}$
$T_{sink,max}$	$[°C]$	150	$\{90, 105, 120, 135, 150, 165, 180, 195, 210\}$
ΔT	$[K]$	10.0	$\{0.0, 2.5, 5.0, 7.5, 10.0, 12.5, 15.0, 17.5, 20.0\}$
$P_{source,min,single}$	$[MW]$	1.0	$\{0.0001, 0.001, 0.01, 0.1, 1.0, 10.0\}$
$P_{sink,min}$	$[MW]$	24.2	$\{12, 18, 24, 30, 36, 40\}$
COP	$[-]$	3.00	$\{1.67, 2.00, 2.33, 2.67, 3.00, 3.33, 3.67, 4.00, 4.33, 4.67, 5.00\}$
$r_{spatial}$	$[m]$	1000	$\{250, 500, 750, 1000, 1250, 1500, 1750, 2000\}$

Table 3: Heat pumps input data

4.4 Full load hours:

For each sector, we made an assumption for the average full load hours. The values we used can be found in table 4. This needs to be done to compute the waste energy of each sector. Using this assumption, another uncertainty is introduced into the model and there is currently no option to run an automated sensitivity analysis for these input values.

5 Usage

To use this tool, the user needs to:

1. Populate the input Excel files appropriately.

Categories	Full load hours
Data centers	8000
DH	8000
Metro stations	7000
Wastewater treatment plants	8000
Food retail	5000
Food production	5000
Iron and steel	8000
Non-metallic minerals	8000
Refineries	8000
Paper and printing	8000
Chemical industry	8000
Non-ferrous metals	8000

Table 4: Input data full load hours

2. Configure the sensitivity analysis parameters in `main.py` if needed.
3. Run the `main.py` script to execute the analysis.

The script outputs detailed logs, maps, and data tables that help in understanding the potential for waste heat recovery and in making informed decisions about heat pump installations.

6 Requirements

- Python 3.x
- Pandas
- GeoPandas
- Matplotlib
- Shapely

7 Installation

To set up the required environment:

1. Install Python 3.x from the official site.
2. Install required packages using pip:

```
pip install pandas geopandas matplotlib shapely
```

Ensure that all input data paths in `main.py` are correctly set up to point to the respective data files before running the script.

8 Conclusion

The model provides a comprehensive framework for assessing waste heat sources and their integration with district heating systems through heat pumps. It automates the process of identifying viable combinations of sources and sinks, thereby facilitating efficient heat recovery strategies. This tool is especially useful for researchers and engineers focusing on sustainable energy solutions. It is built for MAN Energy Technologies and specifically suited for their heat pump data.