
open_plan

Reiner Lemoine Institut and Open Plan team

Dec 13, 2023

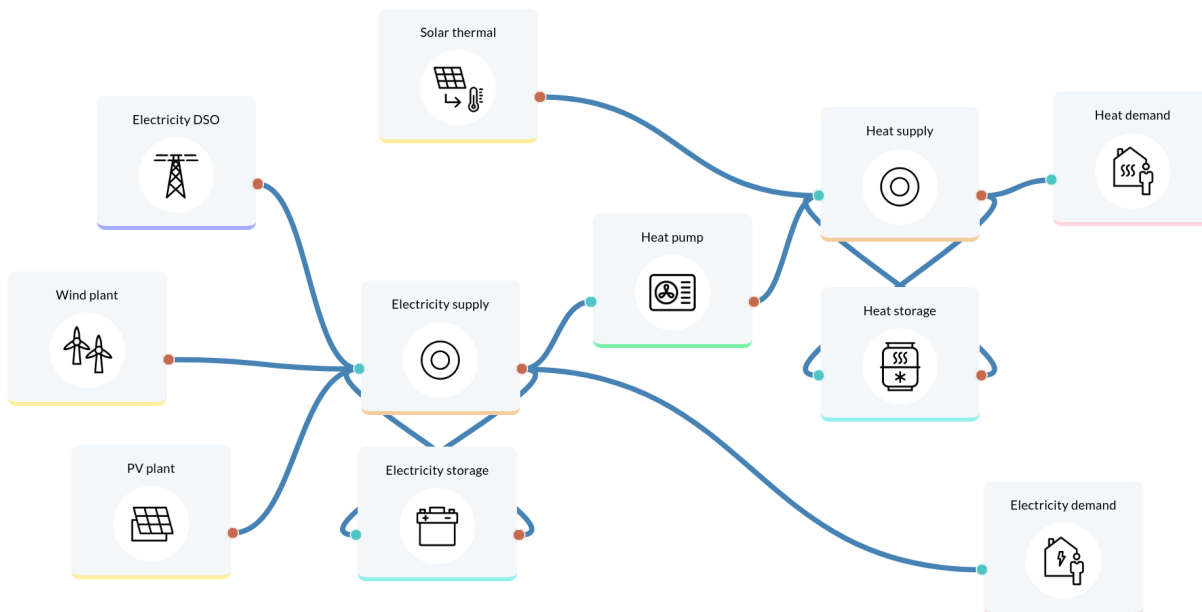
GETTING STARTED

1	Maintainers	3
1.1	Quick start	3
1.2	Installation	13
1.3	Basic structure	15
1.4	Industrial site Max Bögl	17
1.5	Electricity supply to community	17
1.6	Heat supply to community	18
1.7	Sector coupling for office building	18
1.8	Hydrogen production for a technology park	19
1.9	Sector coupling for single household	20
1.10	Sector coupling for apartment building	20
1.11	The mathematics behind open-plan-tool simply explained	21
1.12	Assumptions	26
1.13	Component models	36
1.14	Constraints	44
1.15	Limitations	45
1.16	Input parameters	50
1.17	Outputs of a simulation	72
1.18	License	90
1.19	Contact Us	91
1.20	About project	91
1.21	Credits	91

The open-plan-tool is an open-source tool for designing and optimizing multi-energy carriers energy systems and operational strategies for the supply of neighborhoods, industrial parks and industries.

Its versatility allows you to create different projects and scenarios. You will be able to evaluate different cases and compare results between scenarios with just a few clicks. We also provide you with some use cases that can serve as a reference for your own projects.

Modeling your energy systems has never been so easy before, the tool was developed with a strong focus on easy use without requiring programming language expertise. Using modular components, you can configure your energy system.



You will be asked to provide the required data via individual data, a collection of csv files or a unique JSON file with particular format. The input data is split into the following categories: Project description, which entails the general information regarding the project (country, coordinates, etc.), as well as the economic data such as the discount factor, project duration, or tax; System configuration, in which the user specifies the technical and financial data of each asset; Constrains, are a fundamental part of the definition of the linear problem and describe the degree of autonomy and the share of renewable energies. This set of input data is then translated to a linear programming problem, also known as a constrained optimization problem. The open-plan-tool is based on the multi-vector simulator (MVS), which in turn is based on the oemof-solph python library, with which a problem is described by specifying an objective function to minimize annual energy supply costs, decision variables and limits and constraints.

The simulation outputs are also separated into categories: KPI indicators, economic results and technical results (that include the optimized capacities and dispatch of each asset).

Explore what open-plan-tool has to offer:

Online tool: [open-plan-tool](#)

Publications: [List of publications](#)

GitHub: [Online Repository](#)

Youtube: [Watch Our Videos](#)

The open-plan-tool project consortium is composed of the [Reiner Lemoine Institute \(RLI\)](#), [German Solar Energy Society \[Deutsche Gesellschaft für Sonnenenergie \(DGS\)\]](#) and the [Potsdam Institute for Climate Impact Research](#). It is funded within the framework of the “Technology-oriented systems analysis” funding area of the BMWi’s 7th Energy Research Programme “Innovation for the energy transition”.

The open-plan-tool project complements other previous open-science research projects of RLI, such as open_eGO (OpenEnergyPlatform), open_FRED (feed-in time series on OpenEnergyPlatform) and enables the comparison, validation and improvement of energy system modelling.

MAINTAINERS

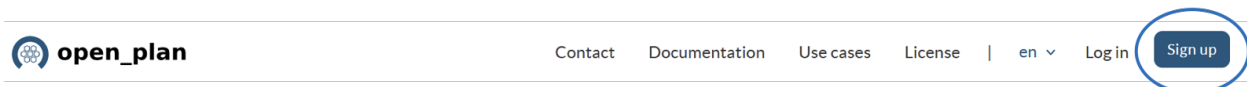
The open-plan-tool is currently maintained by staff from the [Reiner Lemoine Institute](#) the [Potsdam Institute for Climate Impact Research](#) and the [German Solar Energy Society](#).

1.1 Quick start

1.1.1 First steps

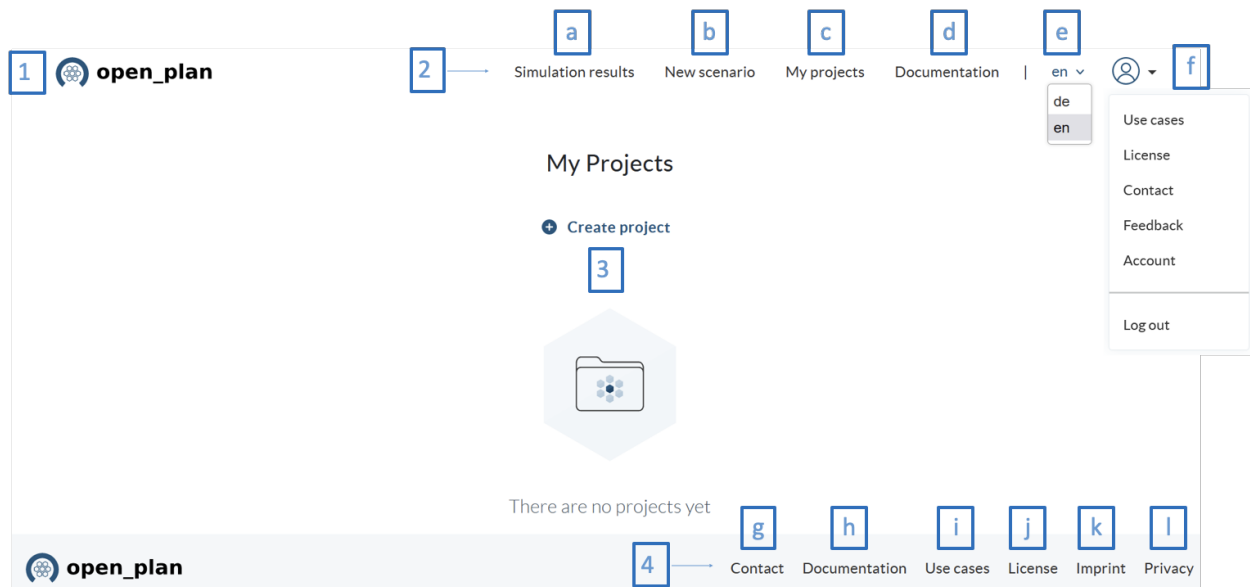
Create an account

The first step to start using open-plan-tool is to create an account. To do this, you must click on the “Sign up” button at the top right corner and fill in the following information: your name, email address, username and password. Finally, do not forget to read and accept the privacy statement. You should receive an email with a link to confirm the account creation. You can find our privacy policy [here](#).



Explore the dashboard

When you log in, the following dashboard is displayed. In the following image the points described below are labeled with numbers and letters.



1. Logo. Clicking here will take you back to the home screen and your projects. Here you find all the projects you have created, and you can create new projects.

2. Top navigation menu. Located on the upper right part of the screen, here you will find the shortcuts to:

- a) Documentation. Key information so that you can get to know the tool and develop your projects.
- b) Use cases. Here you will find some use cases where the functionalities of the system and its application in different projects and scenarios are presented.
- c) We are happy to hear about your experience with open_plan, so feel free to share your questions, comments and suggestions here.
- d) Change the language. English (en) and German (de) are supported.
- e) Profile drop-down options.

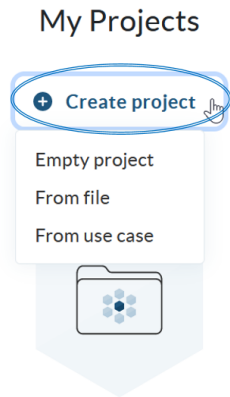
3. Dashboard. Here is the link to create a new project or the list of current projects when they already exist.

4. Bottom navigation menu. Located at the bottom right of the screen. Here you will find:

- g) Contact. General information about the project, a section where you can leave your feedback and find the link to GitHub, where you can follow the development or find useful information related to how to host the tool by yourself.
- j) License. You can find the license information here.
- k) Imprint. Important information about the tool and some disclaimers are in this section.
- l) Privacy. Information on data protection, data processing and the legal and policy issues around them.

Create a project

To create a project in open-plan-tool, the user has three options:



There are no projects yet

Empty project

Here a project is created from scratch. When you select this option you must configure your project and for that you must include the following information:

- Name
- Description
- Country
- Location (coordinates)
- Duration (years)
- Currency
- Discount factor

From file

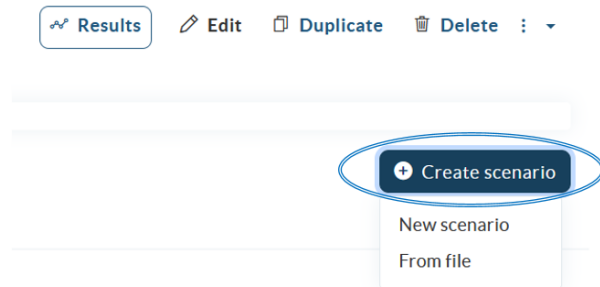
It is also possible to load a project from a file. In this case you must include the name of the project and upload the file in json format. This option makes it possible to share projects among each other.

From use case

If you want to start from one of the configured use cases, you can do so in this option. You will see a pop-up window with a link to the use cases, and a drop-down list where you can choose the use case to use. Once selected, it will appear in the “My projects” section.

Create a scenario

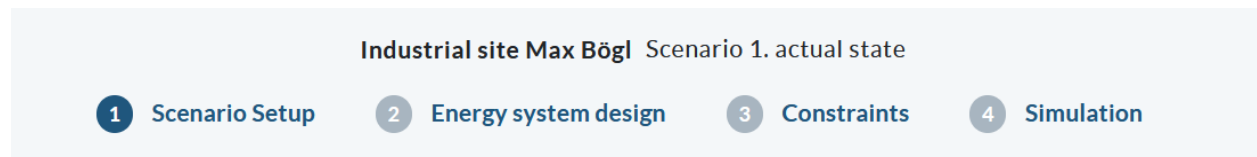
Once the project is created, it is possible to create various scenarios within the project. To do this, you have two options: you can create a new scenario by defining all its parameters or you can load a previously created scenario from a json file. So here there is also the option to share scenarios among each other.



When creating a scenario from scratch, there are four steps to go through:

- 1) Scenario setup
- 2) Energy system design
- 3) Constraints
- 4) Simulation.

At the top you will see the name of the project (in bold type), the name of the scenario, the four steps for scenario creation, and an identifier which step you are in.



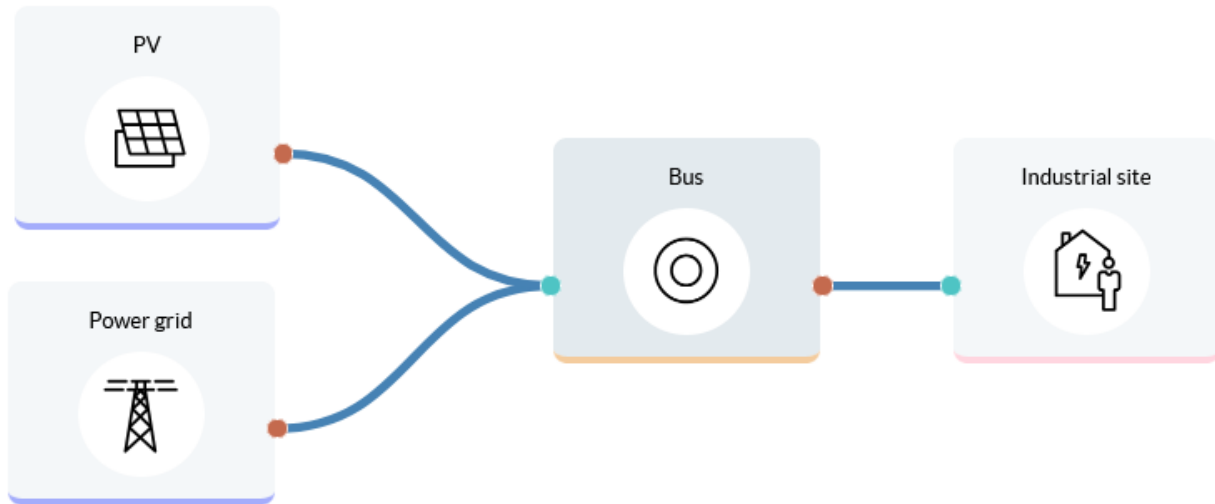
Below is a brief description of what should be done at each step.

1) Scenario setup

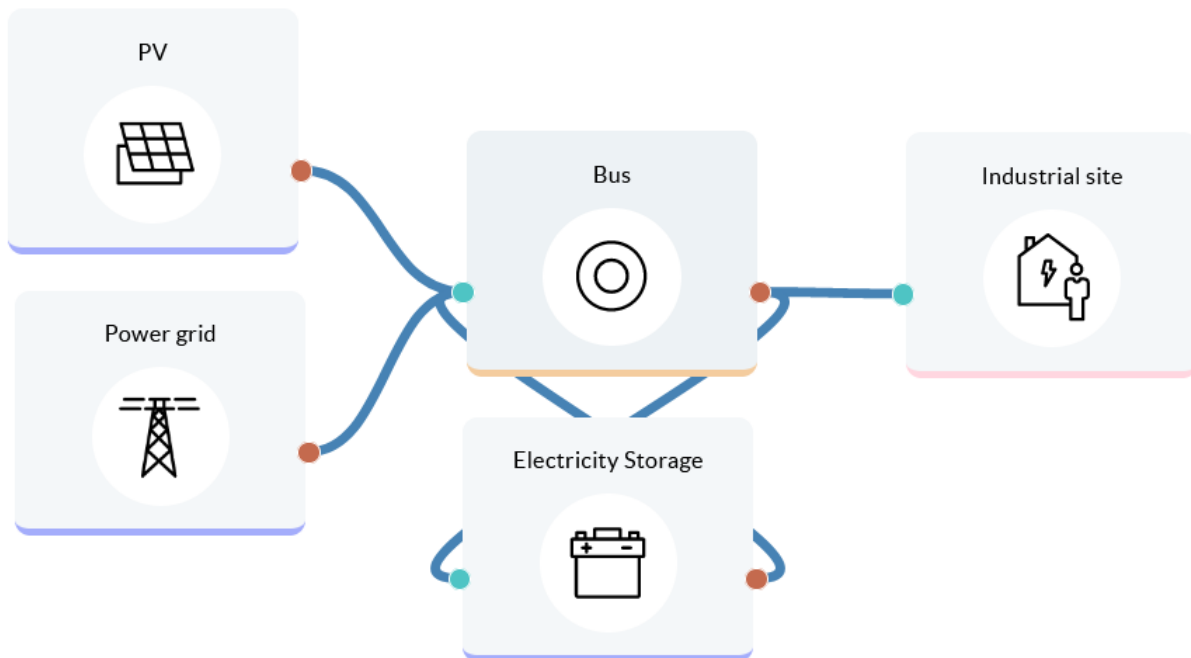
The setup consists of assigning a name to the scenario, its corresponding description, the evaluated period (number of days) the simulation will run, the length of the time steps of the simulation in minutes, the start date (keep in mind that this date is important for getting the data from the time series and for plotting the data) and the fixed project costs which include the planning and development costs of the project.

2) Energy system design

In this section the energy system will be designed using different components, which are located on the left panel and classified in different categories: **Providers, Production, Conversion, Storage, Demand and Bus**. In the graphic panel drag the components you need to design your energy system and do not forget to include the buses. Note that assets are connected to each other using a bus which is the identifier for the energy carrier (e.g. electricity or heat). Interconnecting two buses or two assets directly is not allowed. Connect the components together using the green and red terminals. The green terminals represent inputs, while the red terminals represent outputs, see the following example:



The components representing **battery energy storage systems (BESS)** have been defined with one input and one output. The BESS can be connected directly to the electrical bus; please note that the bus is supplied and feeds the battery at the same time.



When you click on the components, a screen appears where you can configure the different parameters. The input parameters are different for each component category, however, you will typically find three ways to complete the information: spaces to enter values, drop-down lists with default information or buttons to load time series (in this case, a graph will be displayed where the loaded data series can be previewed). The input field for the components that belong to one category (e.g. the conversion components) contains usually the same parameters, but there are also components that may differ slightly, such as the heat pump. Below we show you as an example some of the component setup screens.

bus-1



Name

bus-1

Energy carrier

Choose...

Close

Save

pv plant-1



Name

pv_plant-1

Economical parameters

Fixed project costs (€) ?

0

Investment costs (€/unit) ?

4000

Fixed operational costs (€/unit/year) ?

120

Variable operational costs (€/kWh) ?

0,1

Technical parameters

Asset lifetime (Year) ?

20

Maximum capacity (kWp) ?

e.g. 1000

Renewable asset ?

Yes

Optimize capacity ?

No

Installed capacity (kWp) ?

0

Age of the installed plant (Year) ?

0

Timeseries vector ?

Durchsuchen...

Keine Datei ausgewählt.

Close

Save

Community heat demand ✕

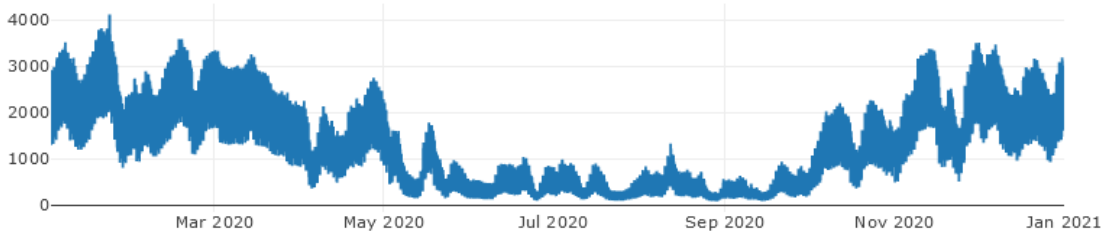
Name

heat_demand-1

Technical parameters

Timeseries vector ?

Durchsuchen... Keine Datei ausgewählt.



uploaded data

Close Save

Before proceeding to the next section, be sure to complete the information requirements for each component and save the energy system.

3) Constraints

In the third section system-wide constraints can be set. At the moment two constraints, the degree of autonomy and the share of renewables, are implemented:

- **Degree of autonomy:** This constraint is a lower boundary for the degree of autonomy of the energy system. The factor can take values between 0 and 1, with the value close to zero showing a degree of autonomy with high dependence on the energy supplier, while a degree of autonomy of 1 represents a fully autonomous energy system.
- **Share of renewables:** This constraint is a lower boundary for the renewable share of the energy system, where both local generation as well as the renewable share of the generation mix supplied by the energy providers are taken into account.

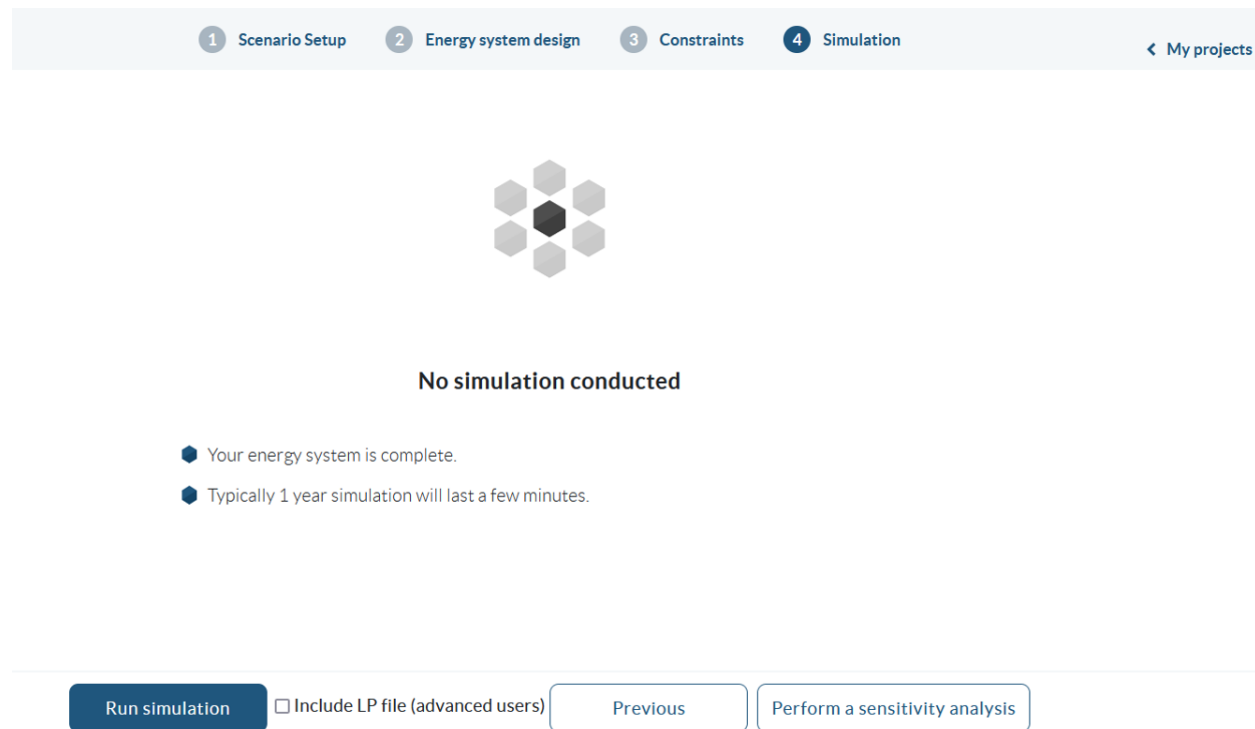
Within each constraint you can decide whether to activate it or not, and in case of activation set a value that must be met by the system.

4) Simulation

Once the scenario parameters are set, you proceed to the simulation panel. At the bottom you will find the button to run the simulation.

Once the simulation is done you get the results of your scenario by clicking on the button **Check results dashboard**.

Further you can select the option whether you want to include an LP file. With the LP file you can look the mathematical formulation consisting of objective function and constraints. It is recommended to choose a small number of days to evaluate (e.g. one day) to keep the LP file readable. After running the simulation you can download the LP file and open it in a text editor.



1 Scenario Setup 2 Energy system design 3 Constraints 4 Simulation < My projects

No simulation conducted

- Your energy system is complete.
- Typically 1 year simulation will last a few minutes.

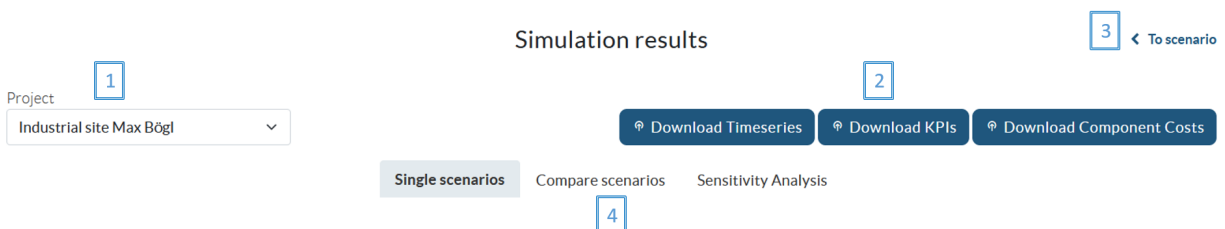
Run simulation ☐ Include LP file (advanced users) Previous Perform a sensitivity analysis

Simulation results

In the **results dashboard** you can have a detailed look at the results of your scenarios. When pressing the button you are automatically in the currently simulated scenario, but you can select another project [1] or scenario [2]. Only scenarios that have already been simulated are displayed.

In the upper right part you have the possibility to download all timeseries of the scenario including given timeseries and simulation results, the resulting Key Performance Indicators (KPIs), and the component costs as Excel files [3]. Further you can return to the scenario setup [4].

To view the most relevant results directly, you have the three options: **single scenarios, compare scenarios and sensitivity analysis** [5]. Please note that the option of sensitivity analysis is not fully implemented yet. At the moment you get an error as soon as you click the button. We proceed working on this feature in the future.



Simulation results 3 < To scenario

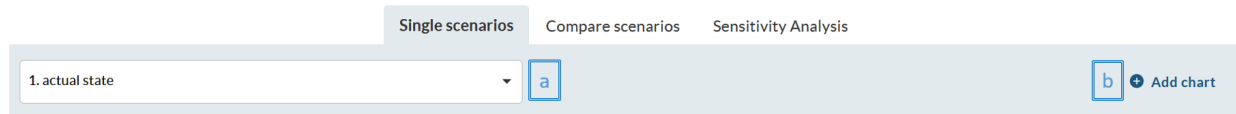
Project 1 Industrial site Max Bögl

2 Download Timeseries Download KPIs Download Component Costs

Single scenarios 4 Compare scenarios Sensitivity Analysis

Single scenarios

This option makes it easy to view the results of one scenario at a time. On the left is a drop-down menu where the scenario is selected (a) and on the right is the option to add a new chart (b).



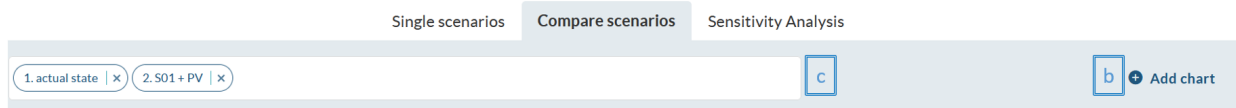
Subsequently, the scenario KPIs will be presented, which include: Degree of Autonomy, Levelized costs of electricity equivalent, Onsite energy fraction, Renewable factor and Renewable share of local generation.

Then, you will be able to visualize the energy system, all its components and connections. Finally, some charts summarizing the results of the scenario are presented, including the overall cost breakdown, the energy series (in KW), the installed and optimized capacity, as well as a Sankey diagram.

Additional charts can be included, as shown in the image above with item b. The charts are interactive, as you can see the value by hovering the pointer over the chart, and there is a menu that appears in the upper right corner of the chart area. Here you can zoom in and out, reset the axes, download the image as .png format, among other options. In the legend of the charts you can select which data series to view or hide with a single click. Tables and charts can be exported in .xls, .pdf format. To do so, you can locate the three dots in the upper right corner of the tables or charts, click on them and select the alternative that suits you best.

Compare scenarios

It is also possible to compare the results of multiple scenarios. You only need to include the scenarios to be compared (c). Remember that you can add additional charts if necessary (b).



For each scenario, a column with the values will appear in the KPI table. Also, the cost, energy and installed and optimized capacity charts show the values of the scenarios. This facilitates the comparison of the data.

As for the previous option, the charts are interactive, can be adjusted according to your needs, and both the table and the charts can be exported in different formats.

Sensitivity analysis

This functionality is not fully implemented yet. At the moment you get an error as soon as you click the button. We proceed working on this feature in the future.

1.1.2 Feedback or Question

We are happy to hear about your experience with open-plan-tool, so feel free to share your questions, comments and suggestions [here](#). We will get back to you as soon as possible. We also have a FAQ section, your question may already be answered there. Remember that on the project's GitHub page you can keep track of the developments that are in progress or those that have been completed.

1.1.3 FAQ

To be completed

1.2 Installation

open-plan-tool runs already live at open-plan-tool <https://open-plan.rl-institut.de> and the simulation server at <https://mvs-open-plan.rl-institut.de/> However it is possible to install both of them locally on your machine or on your own server

Prior to the deploy you should find yourself at the root of the open-plan-tool GUI repository

```
git clone https://github.com/open-plan-tool/gui.git open-plan-tool-GUI
cd open-plan-tool-GUI
```

1.2.1 Deploy GUI locally using and using our open plan MVS server

Prior to be able to develop locally, you might need to install postgres, simply google `install postgres` followed by your os name (linux/mac/windows)

1. Create a virtual environment
2. Activate your virtual environment
3. Install the dependencies with `pip install -r app/requirements/postgres.txt`
4. Install extra local development dependencies with `pip install -r app/dev_requirements.txt`
5. Move to the app folder with `cd app`
6. Create environment variables (only replace content surrounded by <>)

```
SQL_ENGINE=django.db.backends.postgresql
SQL_DATABASE=<your db name>
SQL_USER=<your user name>
SQL_PASSWORD=<your password>
SQL_HOST=localhost
SQL_PORT=5432
DEBUG=(True|False)
```

8. Add an environment variable `MVS_HOST_API` and set the url of the simulation server you wish to use for your models (<https://mvs-open-plan.rl-institut.de/> if you wish to use our simulation server). You can deploy your own [simulation server](#) locally if you need
9. Execute the `local_setup.sh` file (`. local_setup.sh` on linux/mac `bash local_setup.sh` on windows) you might have to make it executable first. Answer yes to the question
10. Start the local server with `python manage.py runserver`

11. You can then login with `testUser` and `ASas12`, or create your own account

1.2.2 Deploy using Docker Compose

The following commands should get everything up and running, using the web based version of the MVS API.

You need to be able to run `docker-compose` inside your terminal. If you can't you should install [Docker desktop](#) first.

- Clone the repository locally `git clone --single-branch --branch main https://github.com/open-plan-tool/gui.git open_plan_gui`
- Move inside the created folder (`cd open_plan_gui`)
- Edit the `.envs/epa.postgres` and `.envs/db.postgres` environment files
 - Change the value assigned to `EPA_SECRET_KEY` with a [randomly generated one](#)
 - Make sure to replace dummy names with you preferred names
 - The value assigned to the variables `POSTGRES_DB`, `POSTGRES_USER`, `POSTGRES_PASSWORD` in `.envs/db.postgres` should match the ones of the variables `SQL_DATABASE`, `SQL_USER`, `SQL_PASSWORD` in `.envs/epa.postgres`, respectively
 - Define an environment variable `MVS_HOST_API` in `.envs/epa.postgres` and set the url of the simulation server you wish to use for your models (for example `MVS_API_HOST="<url to your favorite simulation server>"`), you can deploy your own [simulation server](#) locally if you need
 - Assign the domain of your website (without `http://` or `https://`) to `TRUSTED_HOST`, see <https://docs.djangoproject.com/en/4.2/ref/settings/#csrf-trusted-origins> for more information

Next you can either provide the following commands inside a terminal (with ubuntu you might have to prepend `sudo`)

```
docker-compose --file=docker-compose-postgres.yml up -d --build
```

(you can replace `postgres` by `mysql` if you want to use `mysql`)

```
docker-compose --file=docker-compose-postgres.yml exec -u root app_pg sh initial_setup.sh
```

(this will also load a default `testUser` account with sample scenario).

Or you can run a python script with the following command

```
python deploy.py -db postgres
```

Finally, open a browser and navigate to <http://localhost:8080> (or to <http://localhost:8090> if you chose to use `mysql` instead of `postgres`): you should see the login page of the `open-plan-tool` app. You can then login with `testUser` and `ASas12`, or create your own account.

Proxy settings (optional)

If you use a proxy you will need to set `USE_PROXY=True` and edit `PROXY_ADDRESS=http://proxy_address:port` with your proxy settings in `.envs/epa.postgres`.

Note: If you wish to use `mysql` instead of `postgres`, simply replace `postgres` by `mysql` and `app_pg` by `app` in the above commands or filenames

Note: Grab a cup of coffee or tea for this...

1.2.3 Test Account

You can access a preconfigured project using the following login credentials: `testUser:ASas12,.`

1.2.4 Tear down (uninstall) docker containers

To remove the application (including relevant images, volumes etc.), one can use the following commands in terminal:

```
docker-compose down --file=docker-compose-postgres.yml -v
```

you can add `--rmi local` if you wish to also remove the images (this will take you a long time to rebuild the docker containers from scratch if you want to redeploy the app later then)

Or you can run a python script with the following command

```
python deploy.py -db postgres --down
```

1.3 Basic structure

This repository contains the code for the user interface. The simulations are performed by [multi-vector-simulator](#) on a dedicated server (see the [open-plan-tool/simulation-server](#) repository). Once a simulation is over the results are sent back to the user interface where one can analyse them.

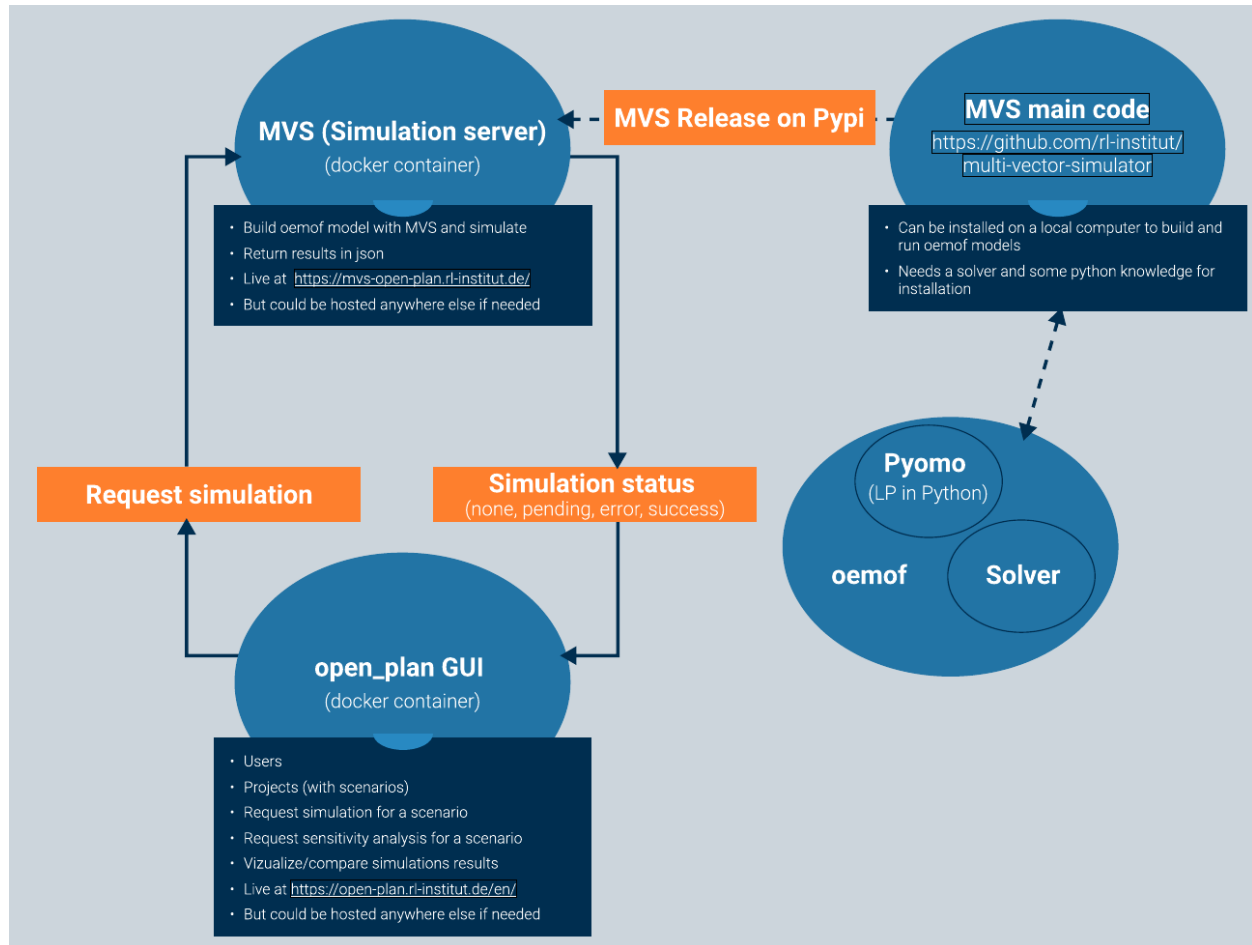
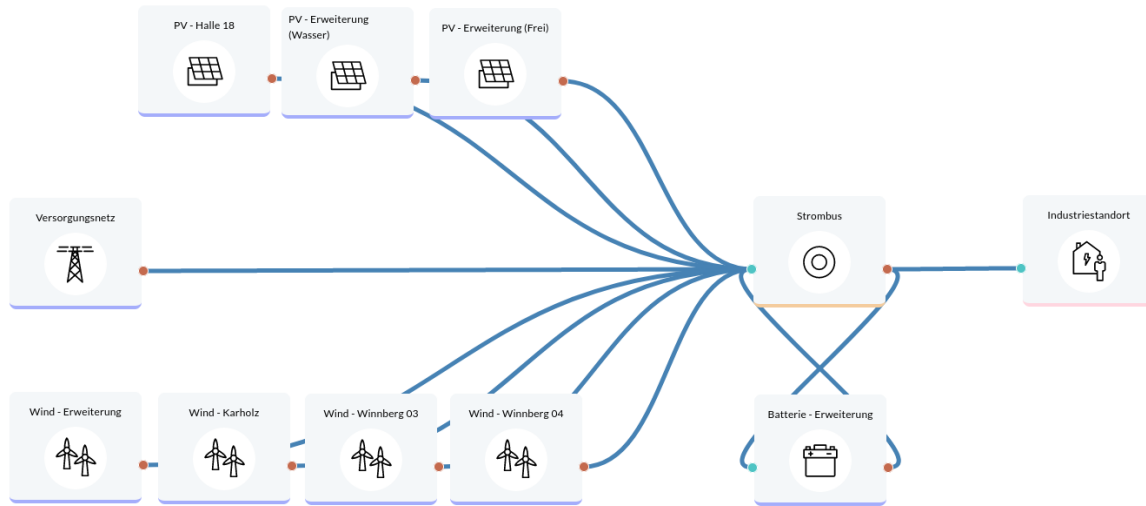


Fig. 1: open-plan structure

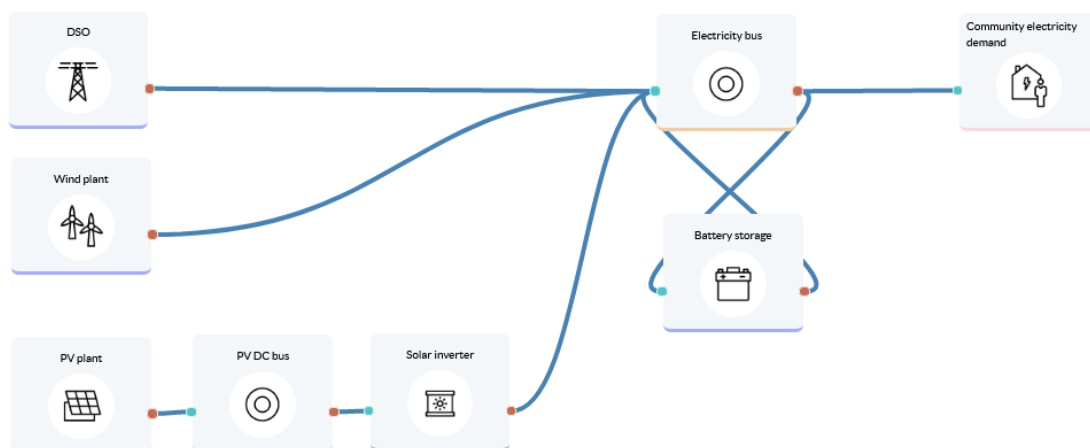
1.4 Industrial site Max Bögl

An industrial site with an annual electricity consumption of 26 GWh and a peak load of 6.3 MW is aiming to increase its self-sufficiency rate and reduce its grid connection power by expanding PV, wind power and a battery. This is shown in four different scenarios. Lifetime: 20 years



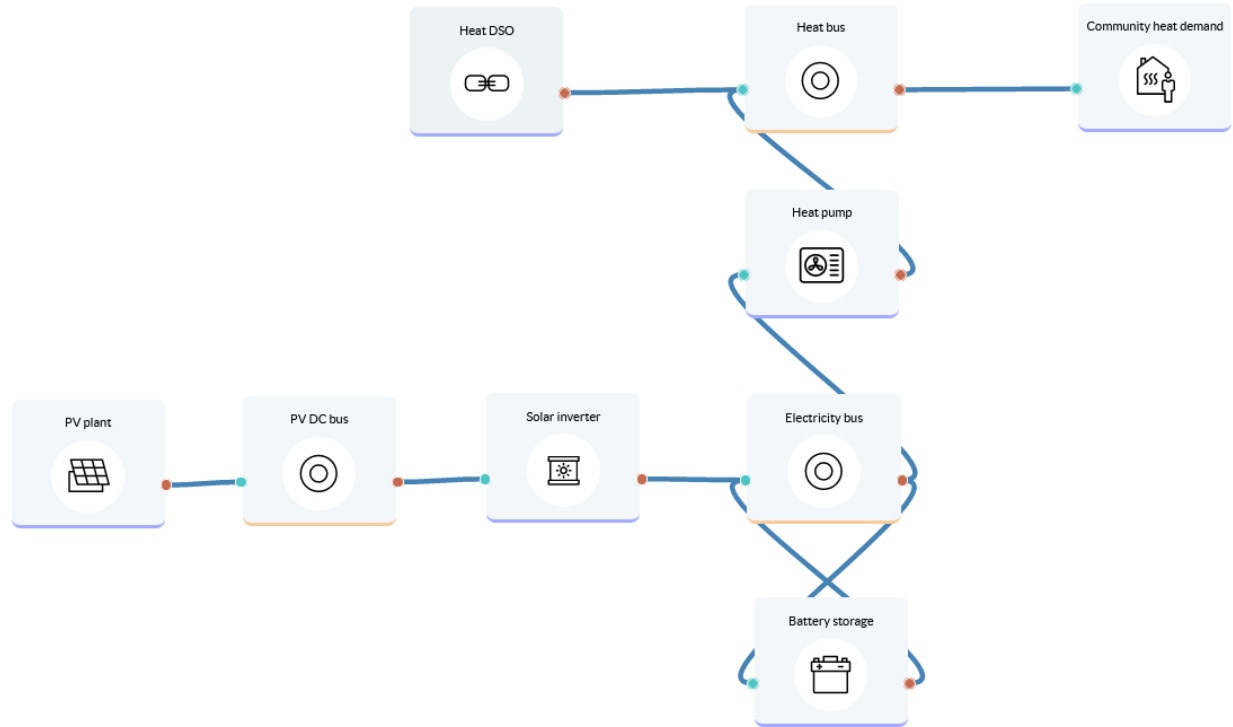
1.5 Electricity supply to community

A community of 3,000 residents wants to determine how they can increase the self-sufficiency of their electricity supply. PV and wind as renewable sources are considered, and battery storage. Lifetime: 20 years



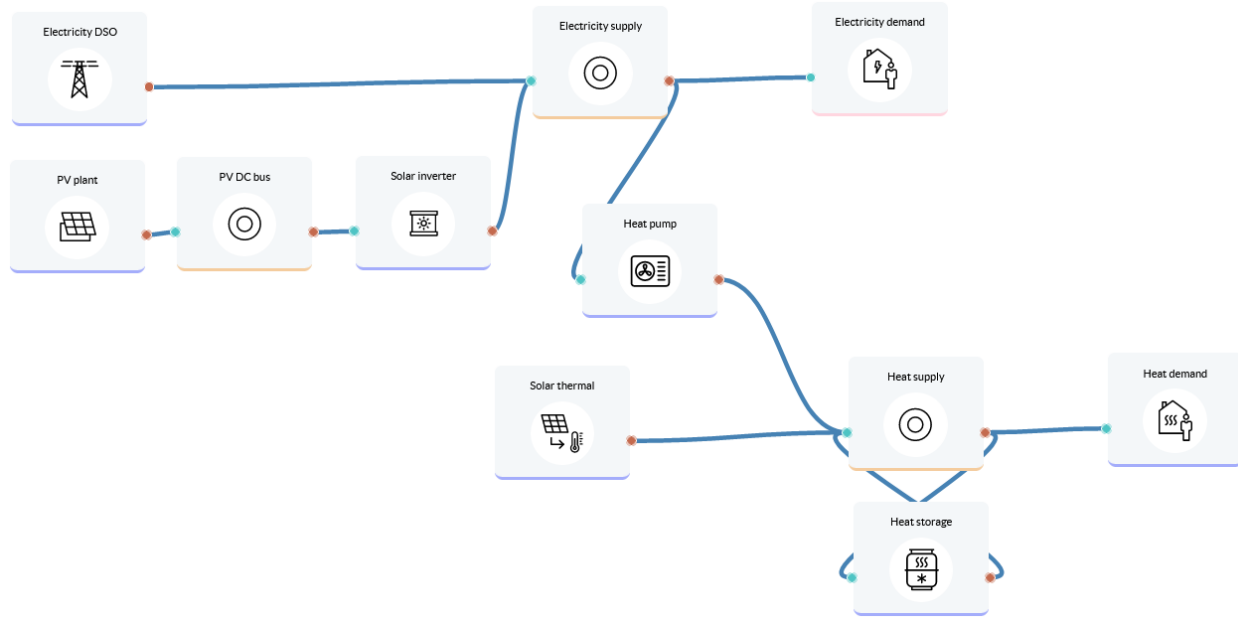
1.6 Heat supply to community

A community of 3,000 residents wants to determine how they can increase the self-sufficiency and renewable share of their heat supply. Lifetime: 20 years



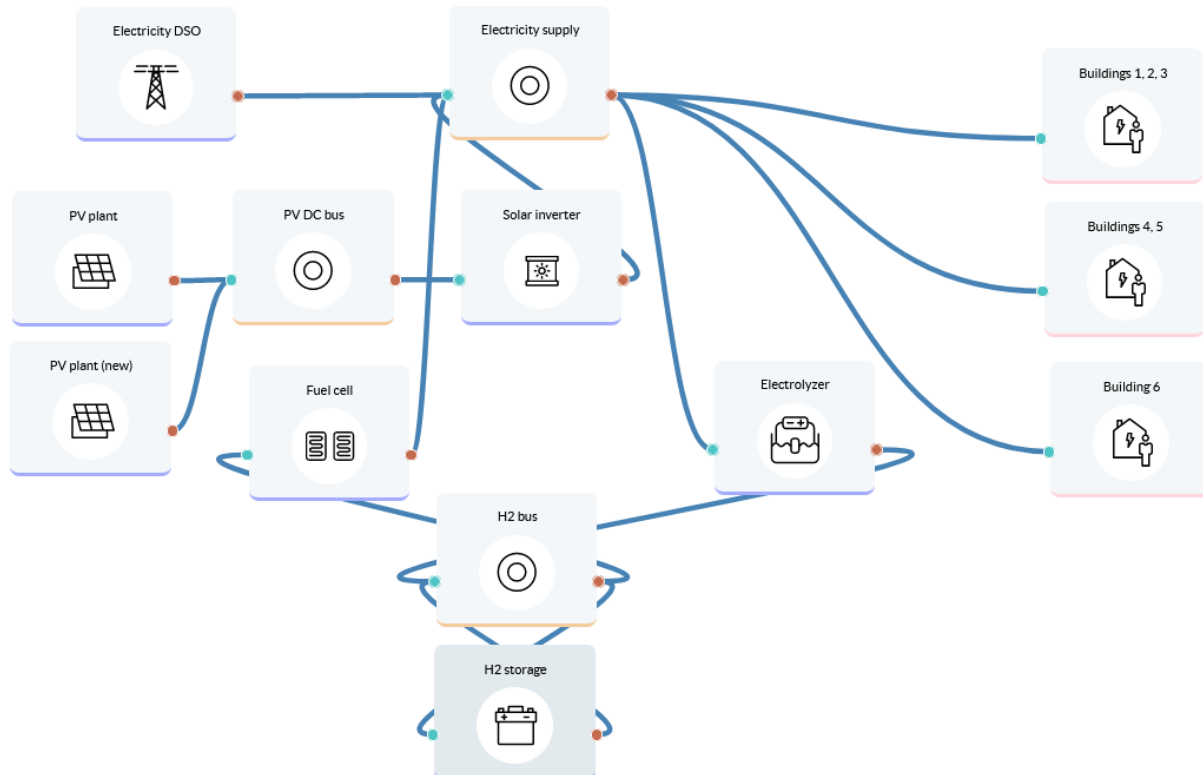
1.7 Sector coupling for office building

An office building aims to improve the sustainability, overall efficiency and flexibility of its energy system by introducing renewables and considering sector coupling for electricity and heat. Lifetime: 20 years



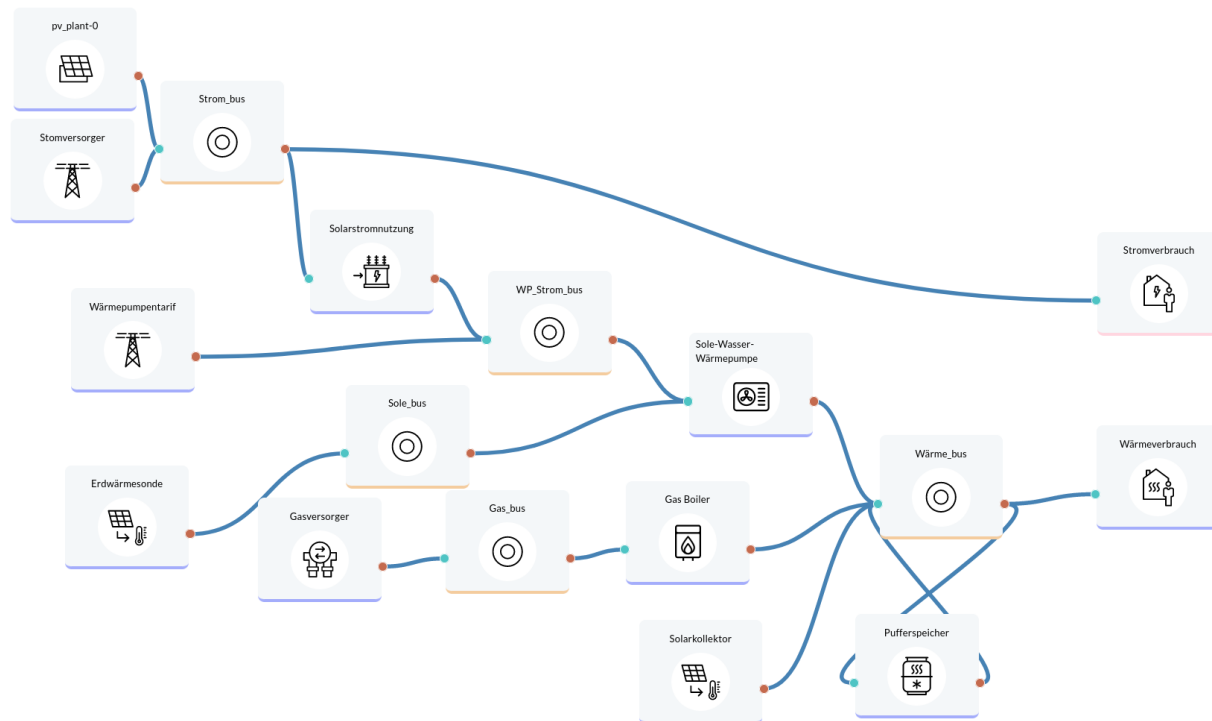
1.8 Hydrogen production for a technology park

A technology park considers locally producing hydrogen from excess PV generation, either for using as a means of electricity storage or for external sales. Lifetime: 20 years



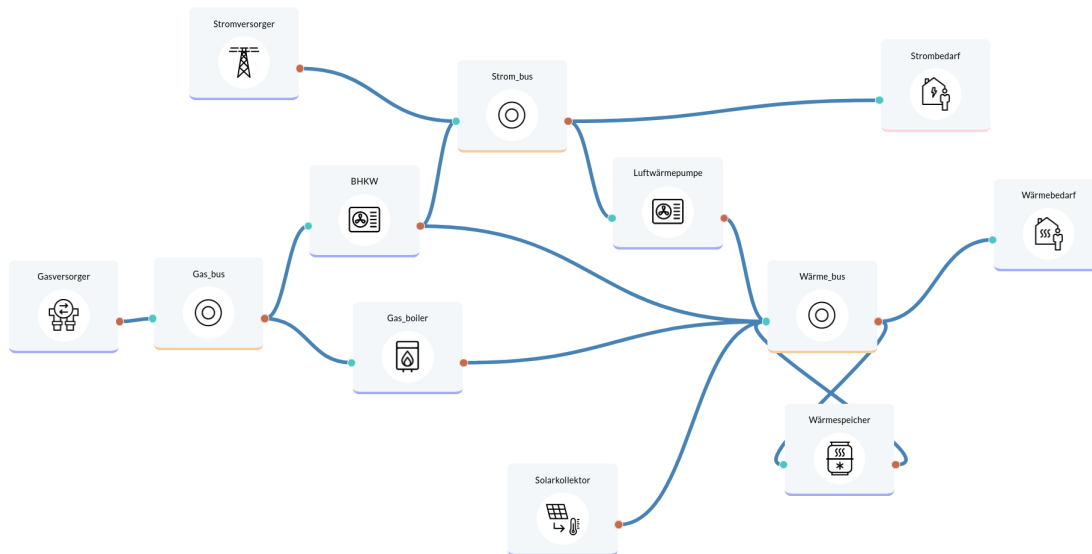
1.9 Sector coupling for single household

Optimization of the home supply near Constance for a single family house with the following data: 3500kWh/year electricity consumption BDEW h0 profile 28600kWh/year heat consumption (heating + hot water) modeled by outdoor temperature according to BDEW. Weather data from German weather service for Konstanz 2018. Cost data for components largely from Danish technology catalog. Assumed roof pitch of 30° with south orientation for solar collector and PV modules. Lifetime: 20 years



1.10 Sector coupling for apartment building

Optimization of an apartment building (approx. 20 apartments) near Konstanz with the following data: 70,000 kWh/year electricity consumption BDEW H0 profile 286,000 kWh/year heat consumption (heating + hot water) modeled by outdoor temperature according to BDEW. Weather data from German weather service for Constance 2018. Cost data for components largely from Danish technology catalog. Lifetime: 20 years



1.11 The mathematics behind open-plan-tool simply explained

1.11.1 Aim of the open-plan-tool

The aim of the open-plan-tool is to provide an open source application that has the capability to model small to medium-sized energy systems with emphasis on renewable energy technologies and optimize them cost-efficiently. The easy to use graphical user interface and also the in-depth insight into the optimization program and its visible open source code encourages end-users and also researchers to use it. .. [Write some more about the aim of open-plan-tool.....]

One important aspect of future scenarios is the determination of optimal dimensioning and combinations of various technologies. To compute an optimal individual energy system open-plan-tool uses linear optimization. When we think about how we want to set up our energy system, we have several goals. For example, we want to minimize the cost of the energy system or the amount of CO₂ that we emit. Furthermore, we want to make sure that demand can be met at any point in time, as well as a variety of other goals. To help us find a way to satisfy all these goals, we can use linear optimization. What was briefly outlined in the last section is basically a non-mathematical way of describing the main and secondary conditions that we need to use linear optimization. In the following example, this will be further illustrated.

[Das folgende ist ein einfaches Beispiel, was die Methodik der linearen Optimierung anhand eines zweidimensionalen Lösungsraums erklären soll.] The following example describes the principle of linear optimization in a two-dimensional space.

1.11.2 An energy system with various technologies

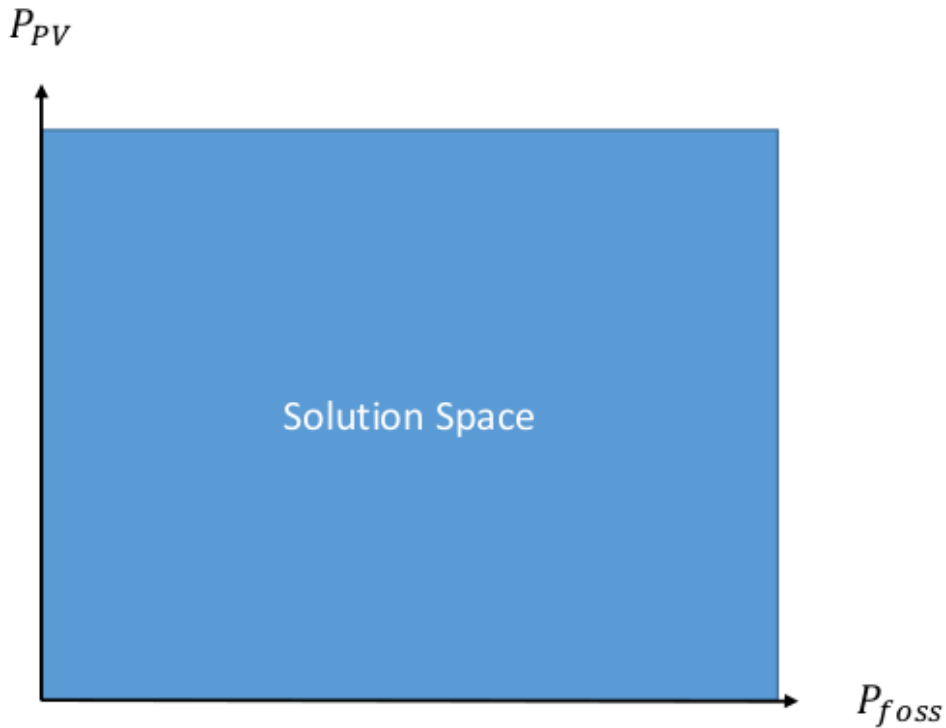
When it comes to electricity generation, we can imagine a simple energy system using solar energy and fossil fuels. To use solar energy we need PV modules, and to use the fossil fuel, we need a fossil fuel power station. For both technologies we are trying to find out how much capacity we should build to get the cost minimum solution. We also have a battery and we have the consumer for which we know the exact demand.

[Picture Energysystem]

Using this data, we can now think about our optimal solution. In this case our objective is to minimize costs. The objective function is described as a linear function that we want to minimize. [spez. Kosten + repetitive installed

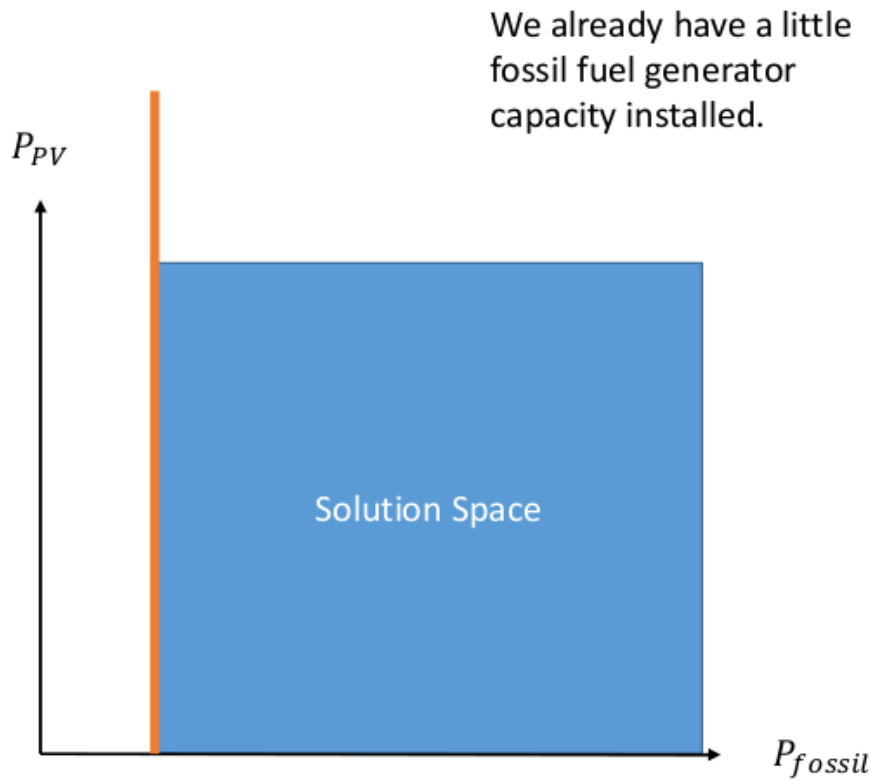
capacities]

We can now represent this graphically as a solution space, which shows us all the possible input combinations. On the x-axis we have the fossil fuel generator capacity, and on the y-axis we have the capacity of a PV plant. Any point within the solution space is a possible solution, and using linear optimization, we will find the optimum.

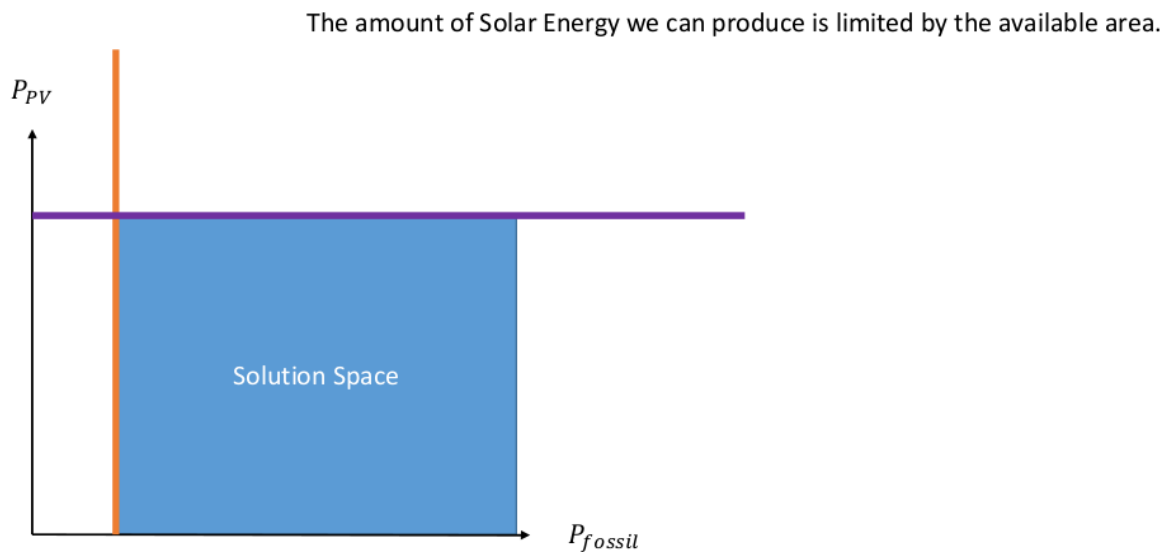


If we just want to minimize the costs, we would have to say that the optimum is (0,0), as this costs us the least. Therefore, we need to add more information, or more secondary conditions. An optimization is linear as long as the main and secondary conditions only contain linear functions. In the following section, we will look at a few secondary conditions.

In our example, we assume that a small fossil fuel generator has already been installed, and consequently, the solution space is reduced, as shown in the picture.



Another secondary condition is that the amount of solar capacity that we can build is restricted by the area that we can actually build solar cells on, which is represented by the purple line.

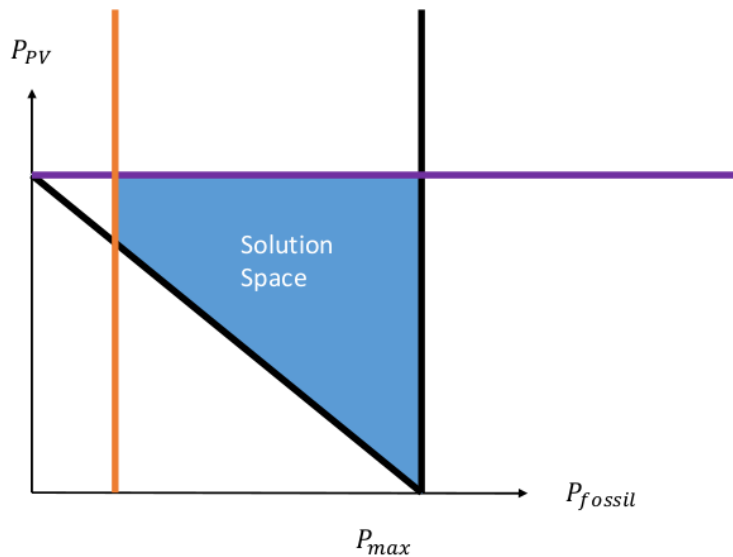


We also know that we do not want to install more capacity than necessary, meaning that the generation capacity of the

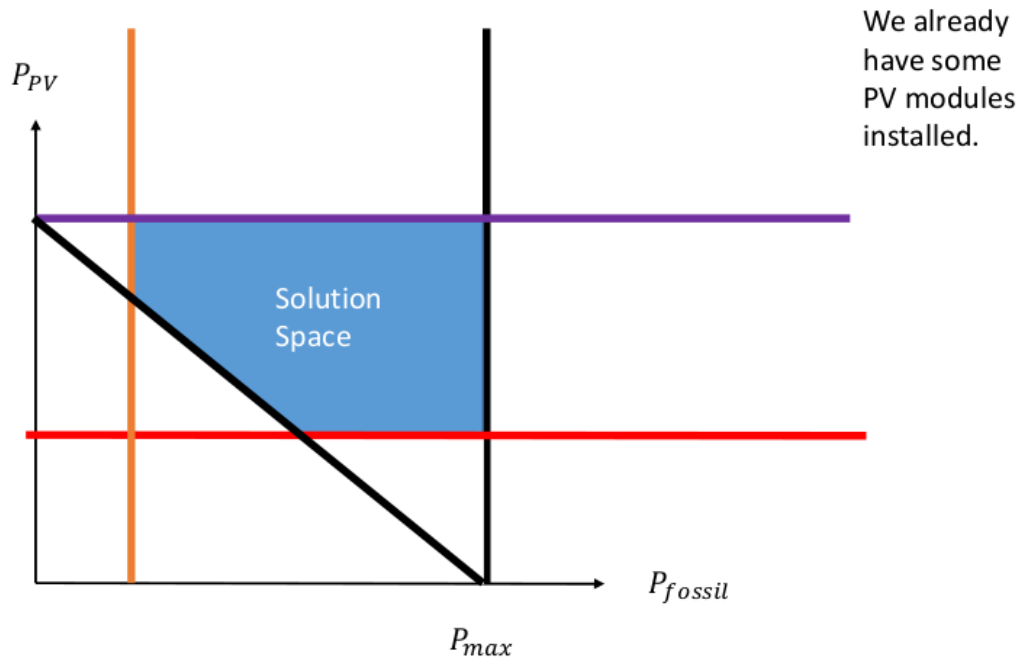
fossil fuel generator should not exceed the peak demand, which is shown by the black straight line.

We also have to be able to meet the peak demand. We need to make sure that we have enough capacity installed to meet this demand, which is depicted by the diagonal line, which shows us all the combinations of solar and fossil fuel capacity that let us meet peak demand. However, all the solutions above the diagonal line are also theoretically possible.

We don't need to exceed the peak demand P_{max} with the capacity of the fossil fuel generators. Also the combination of the fossil fuel generator capacity and the solar capacity has to at least meet peak demand at any point in time, which is represented by the diagonal line.

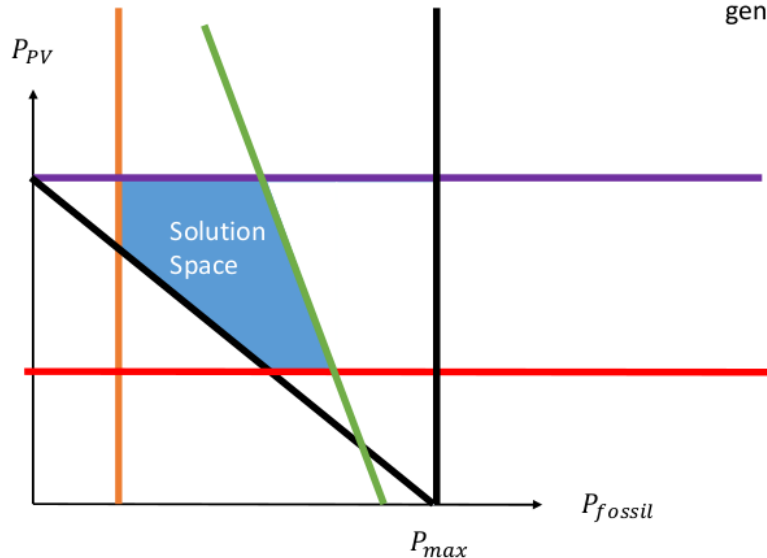


We also have some PV plants installed and consequently our solution space is reduced again.



Finally, we also want to make sure that our energy system is sustainable, and therefore, we define a maximum amount of CO2 that we want to emit, which is represented by the green line. After having reduced the solution space again, we now turn to solving the optimization problem.

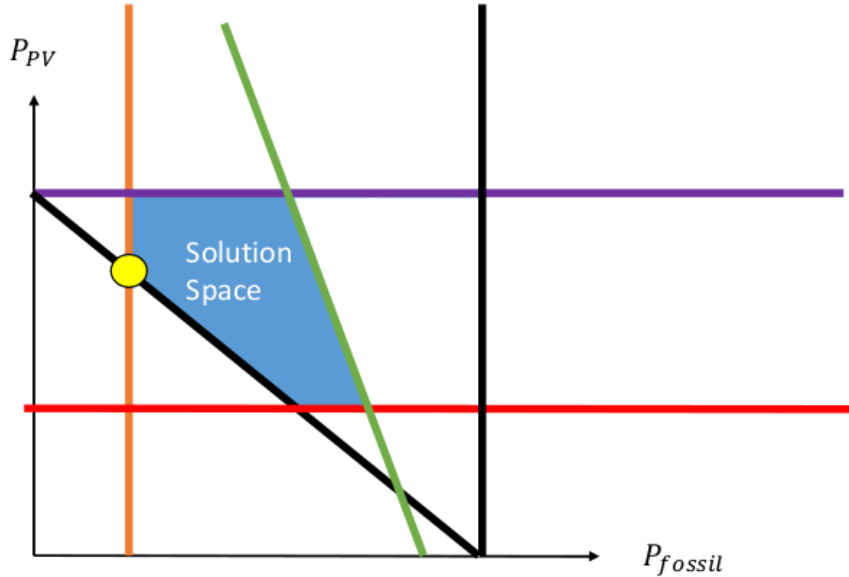
We only want to emit a certain maximum amount of CO2. While the production of solar cells also leads to some CO2 emissions, the fossil fuel generator emits substantially more.



After we have defined our solution space, the next step is to find the optimum. Several ways of solving these problems have been developed, one of which is the simplex method. This can be done on paper, but as the number of equations rises, this becomes more and more difficult.

In open-plan-tool this is done by a solver, which can solve the optimization, given that the equations are in a certain form. The solver then proceeds in two steps. In the first step, it checks if there is a solution to the problem, and as soon as a solution is found, the solver proceeds to the second step. In the second step, the solver then tries to find a better solution and continues this process iteratively until it has found the best solution. To do this, the solver moves along the edges of the solutions space, as the optimum will always lie on the edge of the solution space in a linear optimization model as long as there is an optimum. In our simple example, this means that the solution has to lie somewhere on the edge of our solutions space. In this case the solution is the yellow point.

The Yellow Point represents the optimum, that we obtain using the simplex method.



It is also possible that several solutions exist. Graphically, this would mean that an entire edge of the constraint to the solution space would be an optimum, meaning that we have several solutions that give us the same optimal result. In this case we can pick any point of the input combinations that lead us to the optimal solution. If we increase the complexity, by either adding more secondary conditions, or by expanding the main condition, the solution space becomes more complex, and can go from 3 Dimensional to 50 Dimensional or even more. When the solution space becomes more complex, it becomes basically impossible to graphically demonstrate how the solution space is solved, but the principle is exactly the same in a two dimensional problem or a 50 dimensional problem, it just takes longer for the solver to do its work.

1.12 Assumptions

The open-plan-tool uses the programming framework `oemof-solph` at its core and builds an energy system model based upon its nomenclature. As such, the energy system model can be described with a linear equation system. The most important aspects of a linear equation system are described below in a generalized way, and additionally explained through the use of an example. This will enable the clear comparison to other energy system models.

1.12.1 Economic Dispatch

Linear programming is a mathematical modelling and optimization technique for a system of a linear objective function subject to linear constraints. The goal of a linear programming problem is to find the optimal value for the objective function, be it a maximum or a minimum. open-plan-tool is based on `oemof-solph`, which in turn uses `Pyomo` to create a linear problem. The economic dispatch problem in the open-plan-tool has the objective of minimizing the production cost by allocating the total demand among the generating units at each time step. The equation is the following:

$$\min Z = \sum_i a_i \cdot CAP_i + \sum_i \sum_t c_{var,i} \cdot E_i(t)$$

$$CAP_i \geq 0$$

$$E_i(t) \geq 0 \quad \forall t$$

i : asset

a_i : asset annuity [currency/kWp/year, currency/kW/year, currency/kWh/year]

CAP_i : asset capacity [kWp, kW, kWh]

$c_{var,i}$: variable operational or dispatch cost [currency/kWh, currency/L]

$E_i(t)$: asset dispatch [kWh]

The annual cost function of each asset includes the capital expenditure (investment cost) and residual value, as well as the operating expenses of each asset. It is expressed as follows:

$$a_i = \left(capex_i + \sum_{k=1}^n \frac{capex_i}{(1+d)^{k \cdot t_a}} - c_{res,i} \right) \cdot CRF(T) + opex_i$$

$$CRF(T) = \frac{d \cdot (1+d)^T}{(1+d)^T - 1}$$

$capex_i$: specific investment costs [currency/unit]

n : number of replacements of an asset within project lifetime T

t_a : asset lifetime [years]

CRF : capital recovery factor

$c_{res,i}$: residual value of asset i at the end of project lifetime T [currency/unit]

$opex_i$: annual operational and management costs [currency/unit/year]

d : discount factor

T : project lifetime [years]

The CRF is a ratio used to calculate the present value of the annuity. The discount factor can be replaced by the weighted average cost of capital (WACC), calculated by the user.

The lifetime of the asset t_a and the lifetime of the project T can be different from each other; as a result, the number of replacements n is estimated using the equation below:

$$n = \text{round} \left(\frac{T}{t_a} + 0.5 \right) - 1$$

The residual value is also known as the salvage value and it represents an estimate of the monetary value of an asset at the end of the project lifetime T . open-plan-tool considers a linear depreciation over T and accounts for the time value of money through the use of the following equation:

$$c_{res,i} = \frac{capex_i}{(1+d)^{n \cdot t_a}} \cdot \frac{1}{T} \cdot \frac{(n+1) \cdot t_a - T}{(1+d)^T}$$

1.12.2 Energy Balance Equation

One main constraint that the optimization model is subject to is the energy balance equation, which specifically maintains equality between the total incoming and outgoing energy of a bus. This balancing equation is applicable to all bus types, be it electrical, thermal, hydrogen or any other energy carrier.

$$\sum E_{in,i}(t) - \sum E_{out,j}(t) = 0 \quad \forall t$$

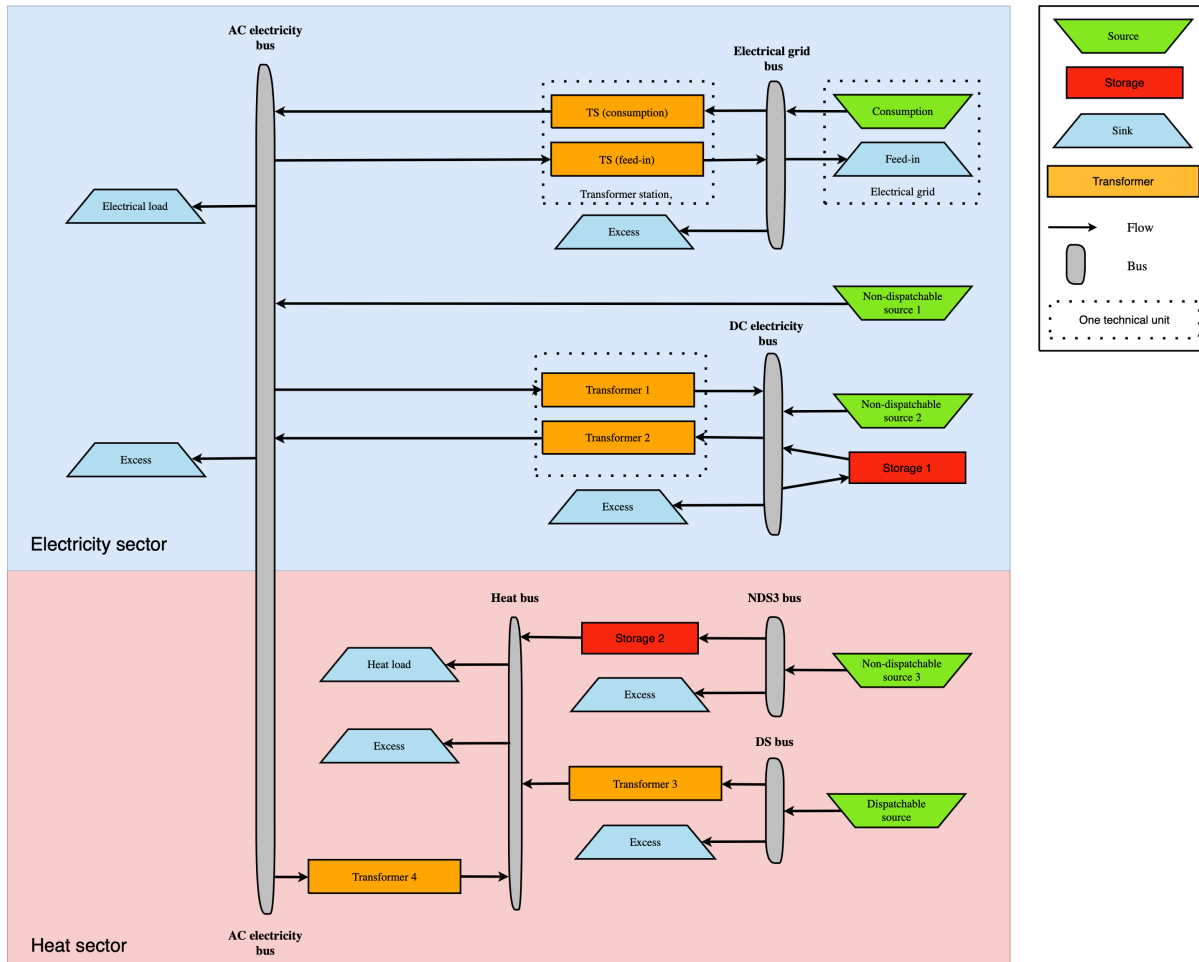
$E_{in,i}$: energy flowing from asset i to the bus

$E_{out,j}$: energy flowing from the bus to asset j

It is very important to note that assets i and j can be the same asset (e.g. a battery with an electrical inflow/outflow). *Oemof-solph* allows both E_{in} and E_{out} to be larger than zero in the same time step t (see *Infeasible bi-directional flow in one timestep*).

1.12.3 Example: Sector Coupled Energy System

In order to understand the component models, a generic sector coupled energy system example is shown in the figure below. It brings together the electricity and heat sector through a transformer (Transformer 4) which connects the two sector buses.



For the sake of simplicity, the following table gives an example for each asset type with an abbreviation to be used in the energy balance and component equations.

Table 1: Asset Types and Examples

Asset Type	Asset Example	Abbreviation	Unit
Non-dispatchable source 1	Wind turbine	wind	kW
Non-dispatchable source 2	Photovoltaic panels	pv	kWp
Storage 1	Battery energy storage	bat	kWh
Transformer 1	Rectifier	rec	kW
Transformer 2	Solar inverter	inv	kW
Non-dispatchable source 3	Solar thermal collector	stc	kWth
Storage 2	Thermal energy storage	tes	kWth
Dispatchable source	Heat source (e.g., biogas)	heat	L
Transformer 3	Turbine	turb	kWth
Transformer 4	Heat pump	hp	kWth

All grid and dispatchable source asset types are assumed to be available 100% of the time with no consumption limits. For each bus in the system, the open-plan-tool automatically includes a sink component for excess energy related to the bus, which is denoted E_{ex} in the equations. This excess sink accounts for the extra energy in the system that has to be dumped.

Electricity Grid Equation

The electricity grid is modeled through a feed-in and a consumption node. Transformers limit the peak flow into or from the local electricity line, and electricity sold to the grid experiences losses in the transformer (ts, f).

$$E_{grid,c}(t) - E_{grid,f}(t) + E_{ts,f}(t) \cdot \eta_{ts,f} - E_{ts,c}(t) = 0 \quad \forall t$$

$E_{grid,c}$: energy consumed from the electricity grid

$E_{grid,f}$: energy fed into the electricity grid

$E_{ts,c}$: transformer station feed-in

$\eta_{ts,f}$: transformer station efficiency

$E_{ts,f}$: transformer station consumption

Non-Dispatchable Source Equations

Non-dispatchable sources in the sector coupled energy system example are wind, PV and solar thermal power. Their generation is determined by the provided timeseries of instantaneous generation, providing α, β, γ in relation to wind, PV and solar thermal power respectively.

$$E_{wind}(t) = CAP_{wind} \cdot \alpha_{wind}(t) \quad \forall t$$

$$E_{pv}(t) = CAP_{pv} \cdot \beta_{pv}(t) \quad \forall t$$

$$E_{stc}(t) = CAP_{stc} \cdot \gamma_{stc}(t) \quad \forall t$$

E_{wind} : energy generated from the wind turbine
 CAP_{wind} : wind turbine capacity [kW]
 α_{wind} : instantaneous wind turbine performance metric [kWh/kW]
 E_{pv} : energy generated from the PV panels
 CAP_{pv} : PV panel capacity [kWp]
 β_{pv} : instantaneous PV specific yield [kWh/kWp]
 E_{stc} : energy generated from the solar thermal collector
 CAP_{stc} : Solar thermal collector capacity [kWth]
 γ_{stc} : instantaneous collector's production [kWh/kWth]

Storage Model

There are two storages in the defined example system: An electrical energy storage (Storage 1, *bat*) and a thermal energy storage (Storage 2, *tes*). Below, the equations for Storage 1 are provided, but Storage 2 follows analogous equations for charge, discharge and bounds.

$$E_{bat}(t) = E_{bat}(t-1) + E_{bat,in}(t) \cdot \eta_{bat,in} - \frac{E_{bat,out}}{\eta_{bat,out}} - E_{bat}(t-1) \cdot \epsilon \quad \forall t$$

$$CAP_{bat} \cdot SOC_{min} \leq E_{bat}(t) \leq CAP_{bat} \cdot SOC_{max} \quad \forall t$$

$$0 \leq E_{bat}(t) - E_{bat}(t-1) \leq CAP_{bat} \cdot C_{rate,in} \quad \forall t$$

$$0 \leq E_{bat}(t-1) - E_{bat}(t) \leq CAP_{bat} \cdot C_{rate,out} \quad \forall t$$

E_{bat} : energy stored in the battery at time t
 $E_{bat,in}$: battery charging energy
 $\eta_{bat,in}$: battery charging efficiency
 $E_{bat,out}$: battery discharging energy
 $\eta_{bat,out}$: battery discharging efficiency
 ϵ : decay per time step
 CAP_{bat} : battery capacity [kWh]
 SOC_{min} : minimum state of charge
 SOC_{max} : maximum state of charge
 $C_{rate,in}$: battery charging rate
 $C_{rate,out}$: battery discharging rate

DC Electricity Bus Equation

The following equation illustrates an example of a DC bus with a battery, PV and a bi-directional inverter.

$$E_{pv}(t) + E_{bat,out}(t) \cdot \eta_{bat,out} + E_{rec}(t) \cdot \eta_{rec} - E_{inv}(t) - E_{bat,in} - E_{ex}(t) = 0 \quad \forall t$$

E_{rec} : rectifier energy
 η_{rec} : rectifier efficiency
 E_{inv} : inverter energy

AC Electricity Bus Equation

This equation describes the local electricity grid and all connected assets:

$$E_{ts,c}(t) \cdot \eta_{ts,c} + E_{wind}(t) + E_{inv}(t) \cdot \eta_{inv} - E_{ts,c}(t) - E_{rec}(t) - E_{hp}(t) - E_{el}(t) - E_{ex}(t) = 0 \quad \forall t$$

$\eta_{ts,c}$: transformer station efficiency

η_{inv} : inverter efficiency

E_{hp} : heat pump electrical consumption

E_{el} : electrical load

Heat Bus Equation

This equation describes the heat bus and all connected assets:

$$E_{tes}(t) \cdot \eta_{tes} + E_{turb}(t) \cdot \eta_{turb} + E_{hp}(t) \cdot COP - E_{th}(t) - E_{ex}(t) = 0$$

η_{tes} : thermal storage efficiency

η_{turb} : turbine efficiency

COP : heat pump coefficient of performance

E_{th} : heat load

NDS3 Bus Equation

The NDS3 Bus is an example of a bus which does not serve both as the input and output of a storage system. Instead, the thermal storage is charged from the NDS3 bus, but discharges into the heat bus.

$$E_{stc}(t) - E_{tes}(t) - E_{ex}(t) = 0$$

E_{tes} : thermal energy storage

DS Bus Equation

The DS Bus shows an example of a fuel source providing an energy carrier (biogas) to a transformer (turbine).

$$E_{heat}(t) - E_{turb}(t) - E_{ex}(t) = 0$$

E_{heat} : thermal energy (biogas) production

E_{turb} : turbine (biogas turbine) energy

1.12.4 Cost calculations

The optimization of the open-plan-tool is mainly based on costs. There is, however, the possibility of introducing additional constraints which will impact the optimization results e.g. implementing a maximum installable capacity limit (comp. *maximumCap*) or adding constraints for certain key performance indicators (see *Constraints*). In order to optimize the energy systems properly, the economic data provided with the input data has to be pre-processed (also see *Economic Dispatch*) and then also post-processed when evaluating the results. The following assumptions are therefore important:

- **Project lifetime:** The simulation has a defined project lifetime, for which continuous operation is assumed - which means that the first year of operation is considered to be the same as the last year of operation. Existing and optimized assets have to be replaced (if their lifetime precedes the system lifetime) to make this possible.
- **Simulation duration:** It is advisable to simulate the whole year to find the most suitable combination of energy assets for your system. Sometimes however you might want to look at specific seasons to see their effect - this is possible by choosing a specific start date and simulation duration.
- **Asset costs:** Each asset can have development costs, specific investment costs, specific operation and management costs as well as dispatch costs.
 - *Replacement costs* are calculated based on the lifetime of the assets, and residual values are paid at the end of the project.
 - *Development costs* are costs that will occur regardless of the installed capacity of an asset - even if it is not installed at all. It stands for system planning and licensing costs. If you have optimized your energy system and see that an asset might not be favourable (zero optimized capacities), you might want to run the simulation again and remove the asset, or remove the development costs of the asset.
 - *Specific investment costs* and *specific operation and maintenance costs* are used to calculate the annual expenditures that an asset has per year, in the process also adding the replacement costs.
 - *Dispatch price* can often be set to zero, but are supposed to cover instances where utilization of an asset requires increased operation and maintenance or leads to wear.
- **Pre-existing capacities:** It is possible to add assets that already exist in your energy system with their capacity and age.
 - *Replacements* - To ensure that the energy system operates continuously, the existing assets are replaced with the same capacities when they reached their end of life within the project lifetime.
 - *Replacement costs* are calculated based on the lifetime of the asset in general and the age of the pre-existing capacities
- **Fix project costs:** It is possible to define fix costs of the project - this is important if you want to compare different project locations with each other. You can define...
 - *Development costs*, which could for example stand for the cost of licenses of the whole energy system
 - *(Specific) investment costs*, which could be an investment into land or buildings at the project site. When you define a lifetime for the investment, the MVS will also consider replacements and reimbursements.
 - *(Specific) operation and management costs*, which can cover eg. the salaries of at the project site

1.12.5 Weighting of energy carriers

To be able to calculate sector-wide key performance indicators, it is necessary to assign weights to the energy carriers based on their usable potential. In the conference paper handed in to the CIRED workshop, we have proposed a methodology comparable to Gasoline Gallon Equivalents.

After thorough consideration, it has been decided to base the equivalence in tonnes of oil equivalent (TOE). Electricity has been chosen as a baseline energy carrier, as our pilot sites mainly revolve around it and also because we believe that this energy carrier will play a larger role in the future. For converting the results into a more conventional unit, we choose crude oil as a secondary baseline energy carrier. This also enables comparisons with crude oil price developments in the market. For most KPIs, the baseline energy carrier used is of no relevance as the result is not dependent on it. This is the case for KPIs such as the share of renewables at the project location or its self-sufficiency. The choice of the baseline energy carrier is relevant only for the levelized cost of energy (LCOE), as it will either provide a system-wide supply cost in Euro per kWh electrical or per kg crude oil.

First, the conversion factors to kg crude oil equivalent [1] were determined (see *Conversion factors: kg crude oil equivalent (kgoe) per unit of a fuel* below). These are equivalent to the energy carrier weighting factors with baseline energy carrier crude oil.

Following conversion factors and energy carriers are defined:

Table 2: Conversion factors: kg crude oil equivalent (kgoe) per unit of a fuel

Energy carrier	Unit	Value
H2 [3]	kgoe/kgH2	2.87804
LNG	kgoe/kg	1.0913364
Crude oil	kgoe/kg	1
Gas oil/diesel	kgoe/litre	0.81513008
Kerosene	kgoe/litre	0.0859814
Gasoline	kgoe/litre	0.75111238
LPG	kgoe/litre	0.55654228
Ethane	kgoe/litre	0.44278427
Electricity	kgoe/kWh(el)	0.0859814
Biodiesel	kgoe/litre	0.00540881
Ethanol	kgoe/litre	0.0036478
Natural gas	kgoe/litre	0.00080244
Heat	kgoe/kWh(therm)	0.086
Heat	kgoe/kcal	0.0001
Heat	kgoe/BTU	0.000025

The values of ethanol and biodiesel seem comparably low in [1] and [2] and do not seem to be representative of the net heating value (or lower heating value) that was expected to be used here.

From this, the energy weighting factors are calculated using the electricity content for crude oil as baseline (see *Electricity equivalent conversion per unit of a fuel* below).

Table 3: Electricity equivalent conversion per unit of a fuel

Energy carrier	Unit	Value
LNG	kWh(elec)/kg	12.6927
Crude oil	kWh(elec)/kg	11.6304
Diesel	kWh(elec)/litre	9.4803
Kerosene	kWh(elec)/litre	8.9080
Gasoline	kWh(elec)/litre	8.7358
LPG	kWh(elec)/litre	6.4728
Ethane	kWh(elec)/litre	5.1498
H2	kWh(elec)/kgH2	33.4728
Electricity	kWh(elec)/kWh(el)	1
Biodiesel	kWh(elec)/litre	0.0629
Ethanol	kWh(elec)/litre	0.0424
Natural gas	kWh(elec)/litre	0.009
Heat	kWh(elec)/kWh(therm)	1.0002
Heat	kWh(elec)/kcal	0.0011
Heat	kWh(elec)/BTU	0.0003

With this, the equivalent potential of an energy carrier $E_{\{elec,i\}}$, compared to electricity, can be calculated with its

conversion factor w_i as:

$$E_{elec,i} = E_i \cdot w_i$$

As it can be noticed, the conversion factor between heat (kWh(therm)) and electricity (kWh(el)) is almost 1. The deviation stems from the data available in source [1] and [2]. The equivalency of heat and electricity can be a source of discussion, as from an exergy point of view these energy carriers can not be considered equivalent. When combined, say with a heat pump, the equivalency can also result in ripple effects in combination with the minimal renewable factor or the minimal degree of autonomy, which need to be evaluated during the pilot simulations.

For the most part, the energy carrier weighting factors are similar to the lower heating value of the fuel in question. A stark deviation is noticable for ethanol and biodiesel. This deviation should be investigated further. In the future, it should be discussed whether it would be better to directly use the lower heating values of a fuel as its energy carrier weighting factor, as this would be more intuitive.

Note: The `energy_vector` of each of the assets and busses must be identical in spelling to one of the energy carriers defined in the above table. Spaces should be translated to underscores (ie. Crude oil as an energy carrier is defined as `Crude_oil` in the input files). Other energy carriers can not be parsed and will raise a warning. Please note that *Heat* currently has to be measured in kWh(thermal).

Code

Currently, the energy carrier conversion factors are defined in `constants.py` with `DEFAULT_WEIGHTS_ENERGY_CARRIERS`. New energy carriers should be added to its list when needed. Unknown carriers raise an `UnknownEnergyVectorError` error.

Comment

Please note that the energy carrier weighting factor is not applied dependent on the LABEL of the energy asset, but based on its energy vector. Let us consider an example:

In our system, we have a dispatchable *diesel fuel source*, with dispatch carrying the unit *l Diesel*. The energy vector needs to be defined as *Diesel* for the energy carrier weighting to be applied, ie. the energy vector of *diesel fuel source* needs to be *Diesel*. This will also have implications for the KPI: For example, the *degree of sector coupling* will reach its maximum, when the system only has heat demand and all of it is provided by processing diesel fuel. If you want to portrait diesel as something inherent to heat supply, you will need to make the diesel source a heat source, and set its *dispatch costs* to currency/kWh, ie. divide the diesel costs by the heating value of the fuel.

Comment

In the open-plan-tool, there is no distinction between energy carriers and energy vector. For *Electricity* of the *Electricity* vector this may be self-explanatory. However, the energy carriers of the *Heat* vector can have different technical characteristics: A fluid on different temperature levels. As the open-plan-tool measures the energy content of a flow in kWh(thermal) however, this distinction is only relevant for the end user to be aware of, as two assets that have different energy carriers as an output should not be connected to one and the same bus if a detailed analysis is expected. An example of this would be, that a system where the output of the diesel boiler as well as the output of a solar thermal panel are connected to the same bus, eventhough they can not both supply the same kind of heat demands (radiator vs. floor heating). This, however, is something that the end-user has to be aware of themselves, eg. by defining self-explanatory labels.

1.12.6 Emission factors

In order to optimise the energy system with minimum emissions, it is important to calculate emission per unit of fuel consumption.

In table *Emission factors: Kg of CO2 equivalent per unit of fuel consumption* the emission factors for energy carriers are defined. These values are based on direct emissions during stationary consumption of the mentioned fuels.

Table 4: Emission factors: Kg of CO2 equivalent per unit of fuel consumption

Energy carrier	Unit	Value	Source
Diesel	kgCO2eq/litre	2.7	[4] Page No. 26
Gasoline	kgCO2eq/litre	2.3	[4] Page No. 26
Kerosene	kgCO2eq/litre	2.5	[4] Page No. 26
Natural gas	kgCO2eq/m3	1.9	[4] Page No. 26
LPG	kgCO2eq/litre	1.6	[4] Page No. 26
Biodiesel	kgCO2eq/litre	0.000125	[5] Page No. 6
Bioethanol	kgCO2eq/litre	0.0000807	[5] Page No. 6
Biogas	kgCO2eq/m3	0.12	[6] Page No. 1

In table *CO2 Emission factors: grams of CO2 equivalent per kWh of electricity consumption* the CO2 emissions for Germany and the four pilot sites (Norway, Spain, Romania, India) are defined:

Table 5: CO2 Emission factors: grams of CO2 equivalent per kWh of electricity consumption

Country	Unit	Value	Source
Germany	gCO2eq/kWh	338	[7] Fig. 2
Norway	gCO2eq/kWh	19	[7] Fig. 2
Spain	gCO2eq/kWh	207	[7] Fig. 2
Romania	gCO2eq/kWh	293	[7] Fig. 2
India	gCO2eq/kWh	708	[8] Page No. 7

The values mentioned in the table above account for emissions during the complete life cycle. This includes emissions during energy production, energy conversion, energy storage and energy transmission.

1.12.7 Input verification

The inputs for a simulation with the open-plan-tool are subjected to a couple of verification tests to make sure that the inputs result in valid oemof simulations. This should ensure:

- Uniqueness of labels (`C1.check_for_label_duplicates`): This function checks if any LABEL provided for the energy system model in `dict_values` is a duplicate. This is not allowed, as oemof can not build a model with identical labels.
- No levelized costs of generation lower than feed-in tariff of same energy vector in case of investment optimization (`optimizeCap` is `True`) (`C1.check_feedin_tariff_vs_levelized_cost_of_generation_of_providers`): Raises error if `feed-in tariff > levelized costs of generation` if `maximumCap` is `None` for energy asset in `ENERGY_PRODUCTION`. This is not allowed, as oemof otherwise may be subjected to an unbound problem, ie. a business case in which an asset should be installed with infinite capacities to maximize revenue. If `maximumCap` is not `None` a `logging.warning` is shown as the maximum capacity of the asset will be installed.

- No feed-in tariff higher than energy price from an energy provider (C1.`check_feedin_tariff_vs_energy_price`): Raises error if feed-in tariff > energy price of any asset in `energyProvider.csv`. This is not allowed, as oemof otherwise is subjected to an unbound and unrealistic problem, eg. one where the owner should consume electricity to feed it directly back into the grid for its revenue.
- Assets have well-defined energy vectors and belong to an existing bus (C1.`check_if_energy_vector_of_all_assets_is_valid`): Validates for all assets, whether `energyVector` is defined within `DEFAULT_WEIGHTS_ENERGY_CARRIERS` and within the `energyBusses`.
- Energy carriers used in the simulation have defined factors for the electricity equivalency weighting (C1.`check_if_energy_vector_is_defined_in_DEFAULT_WEIGHTS_ENERGY_CARRIERS`): Raises an error message if an energy vector is unknown. It then needs to be added to the `DEFAULT_WEIGHTS_ENERGY_CARRIERS` in `constants.py`
- An energy bus is always connected to one inflow and one outflow (C1.`check_for_sufficient_assets_on_busses`): Validating model regarding busses - each bus has to have more than two assets connected to it, excluding energy excess sinks
- Time series of `energyProduction` assets that are to be optimized have specific generation profiles (C1.`check_non_dispatchable_source_time_series`, C1.`check_time_series_values_between_0_and_1`): Raises error if time series of non-dispatchable sources are not between [0, 1].
- Provided timeseries are checked for NaN values, which are replaced by zeroes (C0.`replace_nans_in_timeseries_with_0`).
- Asset capacities connected to each bus are sized sufficiently to fulfill the maximum demand (C1.`check_energy_system_can_fulfill_max_demand`): Logs a `logging.warning` message if the aggregated installed capacity and maximum capacity (if applicable) of all conversion, generation and storage assets connected to one bus is smaller than the maximum demand. The check is applied to each bus of the energy system. Check passes when the potential peak supply is larger than or equal to the peak demand on the bus, or if the maximum capacity of an asset is set to `None` when optimizing.

1.13 Component models

The component models of the open-plan-tool result from the used python-library `oemof-solph` for energy modeling.

It requires component models to be simplified and linearized. This is the reason why the open-plan-tool can provide a pre-feasibility study of a specific system setup, but not the final sizing and system design. The types of assets are presented below.

1.13.1 Energy consumption

Demands within the open-plan-tool are added as energy consumption assets. Most importantly, they are defined by a timeseries, representing the demand profile, and their energy vector. A number of demand profiles can be defined for one energy system, both of the same and different energy vectors. The main optimization goal for the open-plan-tool is to supply the defined demand without fail for all of the timesteps in the simulation with the least cost of supply (comp. *Economic Dispatch*).

1.13.2 Energy production

Non-dispatchable sources of generation

Examples:

- PV plants
- Wind plants
- Run-of-the-river hydro power plants
- Solar thermal collectors

Variable renewable energy (VRE) sources, like wind and PV, are non-dispatchable due to their fluctuations in supply.

The fluctuating nature of non-dispatchable sources is represented by generation time series that show the respective production for each time step of the simulated period. In energy system modelling it is common to use hourly time series.

If you cannot provide time series for your VRE assets you can consider to calculate them by using models for generating feed-in time series from weather data. The following is a list of examples, which is not exhaustive:

- PV: [pvlib](#), [Renewables Ninja](#) (download capacity factors), [alittle](#)
- Wind: [windpowerlib](#), [Renewables Ninja](#) (download capacity factors), [alittle](#)
- Hydro power (run-of-the-river): [hydropowerlib](#)
- Solar thermal: [flat plate collectors](#) of [oemof.thermal](#)

Dispatchable sources of generation

Examples:

- Fuel sources
- Deep-ground geothermal plant (ground assumed to allow unlimited extraction of heat, not depending on season)

Fuel sources are added as dispatchable sources, which can have development, investment, operational and dispatch costs.

Fuel sources are for example needed as source for a diesel generator (diesel), biogas plant (gas) or a condensing power plant (gas, coal, ...), see [Energy conversion](#).

1.13.3 Energy conversion

Examples:

- Electric transformers (rectifiers, inverters, transformer stations, charge controllers)
- HVAC and Heat pumps (as heater and/or chiller)
- Combined heat and power (CHP) and other condensing power plants
- Diesel generators
- Electrolyzers
- Biogas power plants

Conversion assets are added as transformers.

The parameters *dispatch_price*, *efficiency* and *installedCap* of transformers are assigned to their output flows. This means that these parameters need to be provided for the output of the asset and that the costs of the input, (e.g. cost of fuel) are not included in its *dispatch_price* but in the *dispatch_price* of the fuel source, see *Dispatchable sources of generation*.

Conversion assets can be defined with multiple inputs or multiple outputs, but one asset currently cannot have both, multiple inputs and multiple outputs. Note that multiple inputs/output is possible but this feature is not currently tested.

Electric transformers

Electric rectifiers and inverters that are transforming electricity in one direction only, are simply added as transformers. Bidirectional converters and transformer stations are defined by two transformers that are optimized independently from each other, if optimized. The same accounts for charge controllers for a *Battery energy storage system (BESS)* that are defined by two transformers, one for charging and one for discharging. The parameters *dispatch_price*, *efficiency* and *installedCap* need to be given for the electrical output power of the electric transformers.

Note: When using two conversion objects to emulate a bidirectional conversion asset, their capacity should be inter-dependent. This is currently not the case, see *Infeasible bi-directional flow in one timestep*.

Heating, Ventilation, and Air Conditioning (HVAC)

Like other conversion assets, devices for heating, ventilation and air conditioning (HVAC) are added as transformers. As the parameters *dispatch_price*, *efficiency* and *installedCap* are assigned to the output flows they need to be given for the nominal heat output of the HVAC.

Different types of HVAC can be modelled. Except for an air source device with ambient temperature as heat reservoir, the device could be modelled with two inputs (electricity and heat) in case the user is interested in the heat reservoir. This has not been tested yet. Also note that currently efficiencies are assigned to the output flows the see *issue #799*. Theoretically, a HVAC device can be modelled with multiple outputs (heat, cooling, ...); this has not been tested yet.

The efficiency of HVAC systems is defined by the coefficient of performance (COP), which is strongly dependent on the temperature. In order to take account of this, the efficiency can be defined as time series. If you do not provide your own COP time series you can calculate them with `oemof.thermal`, see *documentation on compression heat pumps and chillers* and *documentation on absorption chillers*.

Electrolyzers

Electrolyzers are added as transformers with a constant or time dependent but in any case pre-defined efficiency. The parameters *dispatch_price*, *efficiency* and *installedCap* need to be given for the output of the electrolyzers (hydrogen).

Currently, electrolyzers are modelled with only one input flow (electricity), not taking into account the costs of water; see *issue #799*. The minimal operation level and consumption in standby mode are not taken into account, yet, see *issue #50*.

Condensing power plants and Combined heat and power (CHP)

Condensing power plants are added as transformers with one input (fuel) and one output (electricity), while CHP plants are defined with two outputs (electricity and heat). The parameters *dispatch_price*, *efficiency* and *installedCap* need to be given for the electrical output power (and nominal heat output) of the power plant, while fuel costs need to be included in the *dispatch_price* of the fuel source.

The ratio between the heat and electricity output of a CHP is currently simulated as fix values. This might be changed in the future by using the *ExtractionTurbineCHP* or the *GenericCHP* component of oemof, see [issue #803](#)

Note that multiple inputs/output have not been tested yet.

Other fuel powered plants

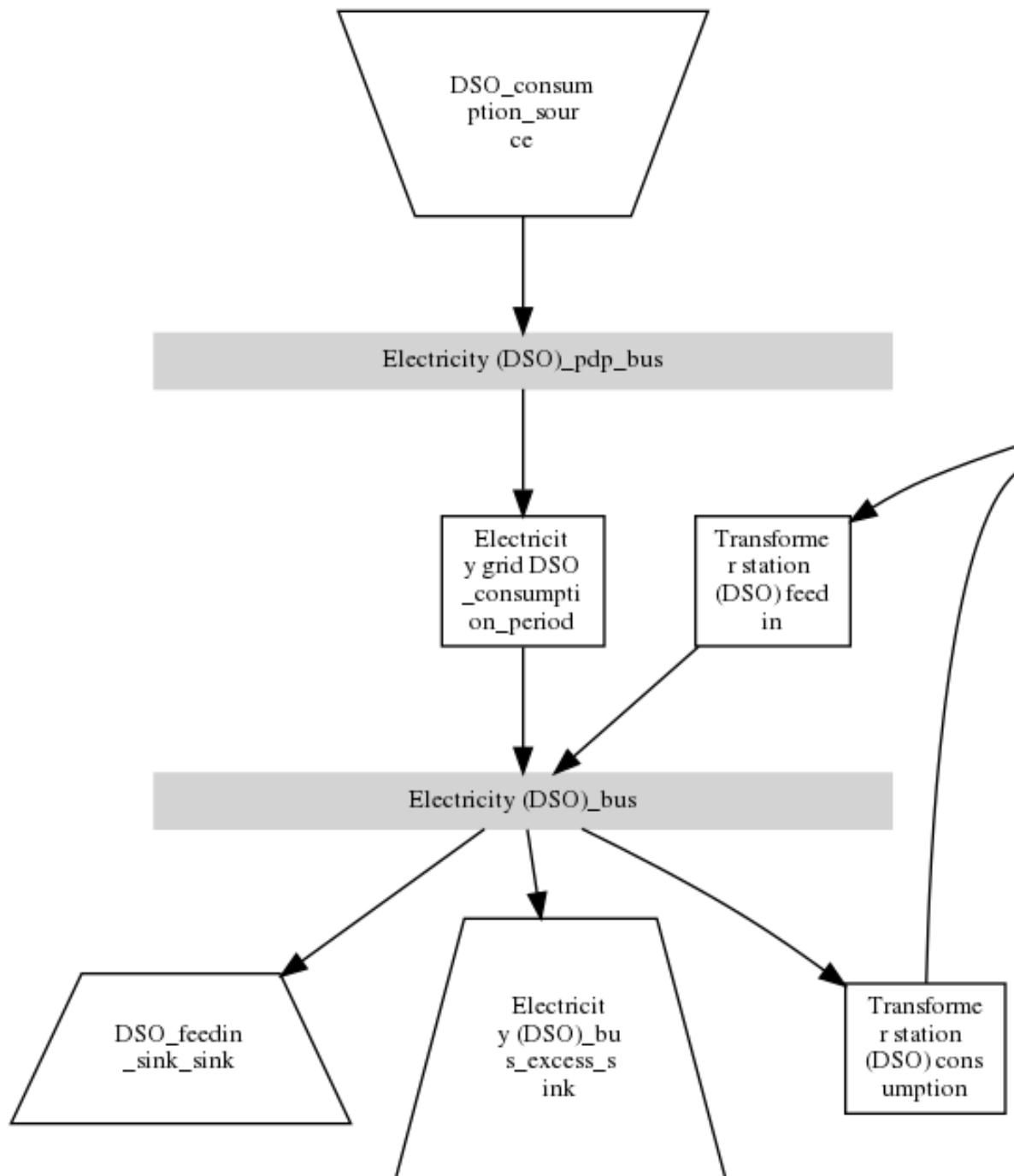
Fuel powered conversion assets, such as diesel generators and biogas power plants, are added as transformers. The parameters *dispatch_price*, *efficiency* and *installedCap* need to be given for the electrical output power of the diesel generator or biogas power plant. As described above, the costs for diesel and gas need to be included in the *dispatch_price* of the fuel source.

1.13.4 Energy providers

The energy providers are the most complex assets in the open-plan-tool model. They are composed of a number of sub-assets

- Energy consumption source, providing the energy required from the system with a certain price
- Energy peak demand pricing “transformers”, which represent the costs induced due to peak demand
- Bus connecting energy consumption source and energy peak demand pricing transformers
- Energy feed-in sink, able to take in generation that is provided to the energy provider for revenue
- Optionally: Transformer Station connecting the energy provider bus to the energy bus of the LES

With all these components, the energy provider can be visualized as follows:



Variable energy consumption prices (time-series)

Energy consumption prices can be added as values that vary over time.

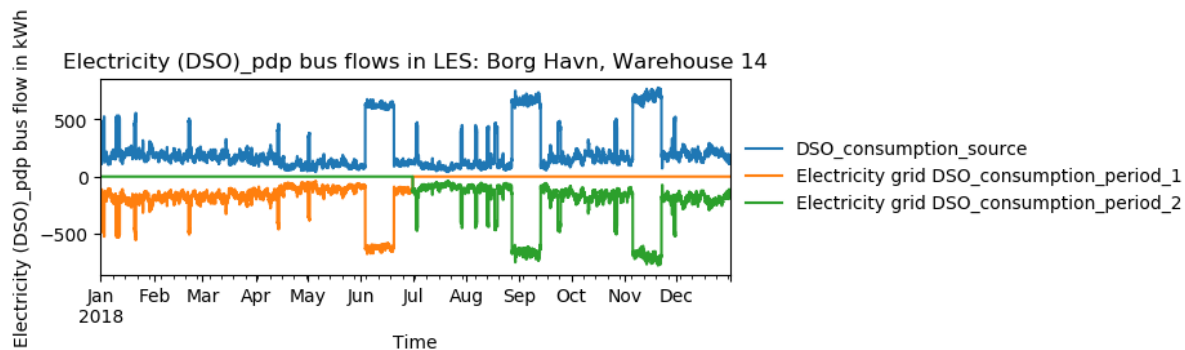
Peak demand pricing

A peak demand pricing scheme is based on an electricity tariff, that requires the consumer not only to pay for the aggregated energy consumption in a time period (eg. kWh electricity), but also for the maximum peak demand (load, eg. kW power) towards the grid of the energy provider within a specific pricing period.

In the open-plan-tool, this information is gathered in the provider assets with:

- `multi_vector_simulator.utils.constants_json_strings.PEAK_DEMAND_PRICING_PERIOD` as the period used in peak demand pricing. Possible values are 1 (yearly), 2 (half-yearly), 3 (each trimester), 4 (quarterly), 6 (every 2 months) and 12 (each month). If you have a *simulation_duration* < 365 days, the periods will still be set up assuming a year! This means, that if you are simulating 14 days, you will never be able to have more than one peak demand pricing period in place.
- `multi_vector_simulator.utils.constants_json_strings.PEAK_DEMAND_PRICING` as the costs per peak load unit, eg. kW

To represent the peak demand pricing, the open-plan-tool adds a “transformer” that is optimized with specific operation and maintenance costs per year equal to the `PEAK_DEMAND_PRICING` for each of the pricing periods. For two peak demand pricing periods, the resulting dispatch could look as following:



1.13.5 Energy storage

Energy storages such as battery storages, thermal storages or H2 storages are modelled with the `GenericStorage` component of `oemof.solph`. They are designed for one input and one output.

The state of charge of a storage at the first and last time step of an optimization are equal. Charge and discharge of the whole capacity of the energy storage are possible within one time step in case the capacity of the storage is not optimized. In case of capacity optimization charge and discharge is limited by the *c-rate*.

Battery energy storage system (BESS)

BESS are modelled as `GenericStorage` like described above. The BESS can either be connected directly to the electricity bus of the LES or via a charge controller that manages the BESS. When choosing the second option, the capacity of the charge controller can be optimized individually, which takes its specific costs and lifetime into consideration. A charge controller is defined by two transformers, see section *Energy conversion* above.

Note that capacity reduction over the lifetime of a BESS that may occur due to different effects during aging cannot be taken into consideration in the open-plan-tool. A possible workaround for this could be to manipulate the lifetime.

Hydrogen storage (H2)

Hydrogen storages are modelled as all storage types in the open-plan-tool as `GenericStorage` like described above.

The most common hydrogen storages store H2 as liquid under temperatures lower than -253 °C or under high pressures. The energy needed to provide these requirements cannot be modelled via the storage component as another energy sector such as cooling or electricity is needed. It could therefore, be modelled as an additional demand of the energy system, see [issue #811](#)

Thermal energy storage

Thermal energy storages of the type sensible heat storage (SHS) are modelled as `GenericStorage` like described above. The implementation of a specific type of SHS, the stratified thermal energy storage, is described in section *Stratified thermal energy storage*. The modelling of latent-heat (or Phase-change) and chemical storages have not been tested with the open-plan-tool, but might be achieved by precalculations.

Stratified thermal energy storage

Stratified thermal energy storage is defined by the two optional parameters *fixed_thermal_losses_relative* and *fixed_thermal_losses_absolute*. These two parameters are used to take into account temperature dependent losses of a thermal storage. To model a thermal energy storage without stratification, the two parameters are not set. The default values of *fixed_thermal_losses_relative* and *fixed_thermal_losses_absolute* are zero. Except for these two additional parameters the stratified thermal storage is implemented in the same way as other storage components.

Precalculations of the *installedCap*, *efficiency*, *fixed_thermal_losses_relative* and *fixed_thermal_losses_absolute* can be done orientating on the stratified thermal storage component of `oemof.thermal`. The parameters U-value, volume and surface of the storage, which are required to calculate *installedCap*, can be precalculated as well.

The efficiency η of the storage is calculated as follows:

$$\eta = 1 - \text{loss_rate}$$

This example shows how to do precalculations using stratified thermal storage specific input data:

```
from oemof.thermal.stratified_thermal_storage import (
    calculate_storage_u_value,
    calculate_storage_dimensions,
    calculate_capacities,
    calculate_losses,
)

# Precalculation
u_value = calculate_storage_u_value(
```

(continues on next page)

(continued from previous page)

```

input_data['s_iso'],
input_data['lamb_iso'],
input_data['alpha_inside'],
input_data['alpha_outside'])

volume, surface = calculate_storage_dimensions(
    input_data['height'],
    input_data['diameter']
)

nominal_storage_capacity = calculate_capacities(
    volume,
    input_data['temp_h'],
    input_data['temp_c'])

loss_rate, fixed_losses_relative, fixed_losses_absolute = calculate_losses(
    u_value,
    input_data['diameter'],
    input_data['temp_h'],
    input_data['temp_c'],
    input_data['temp_env'])

```

Please see the *oemof.thermal* [examples](#) and the [documentation](#) for further information.

For an investment optimization the height of the storage should be left open in the precalculations and *installedCap* should be set to 0 or NaN.

An implementation of the stratified thermal storage component has been done in [pvcompare](#). You can find the precalculations of the stratified thermal energy storage made in *pvcompare* [here](#).

1.13.6 Energy excess

Note: Energy excess components are implemented **automatically** by the open-plan-tool! You do not need to define them yourself.

An energy excess sink is placed on each of the LES energy busses, and therefore energy excess is allowed to take place on each bus of the LES. This means that there are assumed to be sufficient vents (heat) or resistors (electricity) to dump excess (waste) generation. Excess generation can only take place when a non-dispatchable source is present or if an asset is allowed to supply energy without any fuel or dispatch costs.

In case of excessive excess energy, a warning is issued that it seems to be cheaper to have high excess generation than investing into more capacities. High excess energy can for example result into an optimized inverter capacity that is smaller than the peak generation of installed PV. The model becomes unrealistic when the excess is very high.

1.14 Constraints

1.14.1 Minimal renewable factor constraint

The minimal renewable factor constraint requires the capacity and dispatch optimization of the open-plan-tool to reach at least the minimal renewable factor defined within the constraint. The renewable share of the optimized energy system may also be higher than the minimal renewable factor.

The minimal renewable factor is applied to the minimal renewable factor of the whole, sector-coupled energy system, but not to specific sectors. As such, energy carrier weighting plays a role and may lead to unexpected results. The constraint reads as follows:

$$\text{minimalrenewablefactor} \leq \frac{\sum \text{renewablegeneration} \cdot \text{weightingfactor}}{\sum \text{renewablegeneration} \cdot \text{weightingfactor} + \sum \text{non-renewablegeneration} \cdot \text{weightingfactor}}$$

Please be aware that the minimal renewable factor constraint defines bounds for the *Renewable factor (renewable_factor)* of the system, ie. taking into account both local generation as well as renewable supply from the energy providers. The constraint explicitly does not aim to reach a certain *Renewable share of local generation (renewable_share_of_local_generation)* on-site.

Depending on the energy system, especially when working with assets which are not to be capacity-optimized, it is possible that the minimal renewable factor criterion cannot be met. The simulation terminates in that case. If you are not sure if your energy system can meet the constraint, set all `optimize_Cap` parameters of your optimizable assets to `True`, and then investigate further.

Also, if you are aiming at very high minimal renewable factors, the simulation time can increase drastically. If you do not get a result after an excessive simulation time (e.g. 10 times the simulation without constraints), you should consider terminating the simulation and trying with a lower minimum renewable share.

1.14.2 Minimal degree of autonomy constraint

The minimal degree of autonomy constraint requires the capacity and dispatch optimization of the open-plan-tool to reach at least the minimal degree of autonomy defined within the constraint. The degree of autonomy of the optimized energy system may also be higher than the minimal degree of autonomy. For more details, refer to the definition of *degree of autonomy*

The minimal degree of autonomy is applied to the whole, sector-coupled energy system, but not to specific sectors. As such, energy carrier weighting plays a role and may lead to unexpected results.

The constraint reads as follows:

$$\text{minimal degree of autonomy} \leq DA = \frac{\sum E_{\text{demand},i} \cdot w_i - \sum E_{\text{consumption,provider},j} \cdot w_j}{\sum E_{\text{demand},i} \cdot w_i}$$

Depending on the energy system, especially when working with assets which are not subject to the optimization of their capacities, it is possible that the minimal degree of autonomy criterion cannot be met. The simulation terminates in that case. If you are not sure if your energy system can meet the constraint, set all `optimizeCap` parameters of your optimizable assets to `True`, and then investigate further.

1.14.3 Maximum emission constraint

The maximum emission constraint limits the maximum amount of total emissions per year of the energy system. It allows the capacity and dispatch optimization of the open-plan-tool to result into a maximum amount of emissions defined by the maximum emission constraint. The yearly emissions of the optimized energy system may also be lower than the maximum emission constraint.

Note: The maximum emissions constraint currently does not take into consideration life cycle emissions, also see *Total GHG emissions (total_emissions)* section for an explanation.

The unit of the constraint is $kgCO_2eq/a$. To pick a realistic value for this constraint you can e.g.:

- Firstly, optimize your system without the constraint to get an idea about the scale of the emissions and then, secondly, set the constraint and lower the emissions step by step until you reach an unbound problem (which then represents the non-achievable minimum of emissions for your energy system)
- Check the emissions targets of your region/country and disaggregate the number

1.14.4 Net zero energy (NZE) constraint

The net zero energy (NZE) constraint requires the capacity and dispatch optimization of the open-plan-tool to result into a net zero system, but can also result in a plus energy system. The degree of NZE of the optimized energy system may be higher than 1, in case of a plus energy system. Please find the definition of net zero energy (NZE) and the KPI here: *Degree of Net Zero Energy (degree_of_nze)*.

Some definitions of NZE systems in literature allow the energy system's demand solely be provided by locally generated renewable energy. In the open-plan-tool this is not the case - all locally generated energy is taken into consideration. To enlarge the share of renewables in the energy system you can use the *Minimal renewable factor constraint*.

The NZE constraint is applied to the whole, sector-coupled energy system, but not to specific sectors. As such, energy carrier weighting plays a role and may lead to unexpected results. The constraint reads as follows:

$$\sum_i E_{feedin,provider}(i) \cdot w_i - E_{consumption,provider}(i) \cdot w_i \geq 0$$

Depending on the energy system, especially when working with assets which are not subject to the optimization of their capacities, it is possible that the NZE criterion cannot be met. The simulation terminates in that case. If you are not sure whether your energy system can meet the constraint, set all *optimizeCap* parameters of your optimizable assets to True, and then investigate further.

1.15 Limitations

When running simulations with the open-plan-tool, there are certain peculiarities to be aware of. The peculiarities can be considered as limitations, some of which are merely model assumptions and others are drawbacks of the model. A number of those are inherited due to the nature of the open-plan-tool and its underlying modules. The following table (*Limitations*) lists the limitations of open-plan-tool based on their type.

Table 6: Limitations

Inherited	Can be addressed
<i>Infeasible bi-directional flow in one timestep</i>	<i>Need to model one technical unit with two transformer assets</i>
<i>Simplified linear component models</i>	<i>Random excess energy distribution</i>
<i>No degradation of efficiencies over a component lifetime</i>	<i>Renewable energy share definition relative to energy carriers</i>
<i>Perfect foresight</i>	<i>Energy carrier weighting</i>
	<i>Events of energy shortage or grid interruption cannot be modelled</i>

1.15.1 Infeasible bi-directional flow in one timestep

Limitation

It is not possible to model two flows in opposite directions during the same time step.

Reason

The open-plan-tool is based on the python library `oemof-solph`. Its generic components are used to set up the energy system. As a ground rule, the components of `oemof-solph` are unidirectional. This means that for an asset that is bidirectional two transformer objects have to be used. Examples for this are:

- Physical bi-directional assets, eg. inverters
- Logical bi-directional assets, eg. consumption from the grid and feed-in to the grid

To achieve the real-life constraint one flow has to be zero when the other is larger zero, one would have to implement following relation:

$$E_{in} \cdot E_{out} = 0$$

However, this relation creates a non-linear problem and can not be implemented in `oemof-solph`.

Implications

This limitation means that the open-plan-tool might result in infeasible dispatch of assets. For instance, a bus might be supplied by a rectifier and itself supplying an inverter at the same time step t , which cannot logically happen if these assets are part of one physical bi-directional inverter. Another case that could occur is feeding the grid and consuming from it at the same time t .

Under certain conditions, including excess generation as well as dispatch costs of zero, the infeasible dispatch can also be observed for batteries and result in a parallel charge and discharge of the battery. If this occurs, a solution may be to set a marginal dispatch cost of battery charge.

1.15.2 Simplified linear component models

Limitation

The open-plan-tool simplifies the component model of some assets.

- Generators have an efficiency that is not load-dependent
- Storage have a charging efficiency that is not SOC-dependent
- Turbines are implemented without ramp rates

Reason

The open-plan-tool is based `oemof-solph` python library and uses its generic components to set up an energy system. Transformers and storages cannot have variable efficiencies, because otherwise the system of equation to solve would not be linear.

Implications

Simplifying the implementation of some component specifications can be beneficial for the ease of the model, however, it contributes to the lack of realism and might result in less accurate values. The open-plan-tool trades off the decreased level of detail for a quick evaluation of its scenarios, which are often only used for a pre-feasibility analysis.

1.15.3 No degradation of efficiencies over a component lifetime

Limitation

The open-plan-tool does not degrade the efficiencies of assets over the lifetime of the project, eg. in the case of production assets like PV panels.

Reason

The simulation of the open-plan-tool is only based on a single reference year, and it is not possible to take into account multi-year degradation of asset efficiency.

Implications

This results in an overestimation of the energy generated by the asset, which implies that the calculation of some other results might also be overestimated (e.g. overestimation of feed-in energy). The user can circumvent this by applying a degradation factor manually to the generation time series used as an input for the open-plan-tool.

1.15.4 Perfect foresight

Limitation

The optimal solution of the energy system is based on perfect foresight.

Reason

As the open-plan-tool and thus `oemof-solph`, which is handling the energy system model, know the generation and demand profiles for the whole simulation time and solve the optimization problem based on a linear equation system, the solver knows their dispatch for certain, whereas in reality the generation and demand could only be forecasted.

Implications

The perfect foresight can lead to suspicious dispatch of assets, for example charging of a battery right before a (in real-life) random blackout occurs. The systems optimized with the open-plan-tool therefore, represent their optimal potential, which in reality could not be reached. The open-plan-tool has thus a tendency to underestimate the needed battery capacity or the minimal state of charge for backup purposes, and also designs the PV system and backup power according to perfect forecasts. In reality, operational margins would need to be considered.

1.15.5 Optimization precision

Limitation

Marginal capacities and flows below a threshold of 10^{-6} are rounded to zero.

Reason

The open-plan-tool makes use of the open energy modelling framework (oemof) by using `oemof-solph`. For the open-plan-tool, we use the `cbc-solver` with a `ratioGap=0.03`. This influences the precision of the optimized decision variables, ie. the optimized capacities as well as the dispatch of the assets. In some cases the dispatch and capacities vary around 0 with fluctuations of the order of floating point precision (well below $<10e-6$), thus resulting sometimes in marginal fluctuations dispatch or capacities around 0. When calculating KPI from these decision variables, the results can be nonsensical, for example leading to SoC curves with negative values or values far above the viable value 1. As the reason for these inconsistencies is known, the open-plan-tool enforces the capacities and dispatch of to be above $10e-6$, ie. all capacities or flows smaller than that are automatically set to zero. This is applied to absolute values, so that irregular (and incorrect) values for decision variables can still be detected.

Implications

If your energy system has demand or resource profiles that include marginal values below the threshold of 10^{-6} , the open-plan-tool will not result in appropriate results. For example, that means that if you have an energy system with usually is measured in *MW* but one demand is in the *W* range, the dispatch of assets serving this minor demand is not displayed correctly. Please consider using *kW* or even *W* as a base unit then.

1.15.6 Extension of KPIs necessary

Limitation

Some important KPIs usually required by developers are currently not implemented within open-plan-tool:

- Internal rate of return (IRR)
- Payback period
- Return on equity (ROE),

Reason

The open-plan-tool is a work in progress and this can still be addressed in the future.

Implications

The absence of such indicators might affect decision-making.

1.15.7 Random excess energy distribution

Limitation

There is random excess distribution between the feed-in sink and the excess sink when no feed-in-tariff is assumed in the system.

Reason

Since there is no feed-in-tariff to benefit from, the open-plan-tool randomly distributes the excess energy between the feed-in and excess sinks. As such, the distribution of excess energy changes when running several simulations for the same input files.

Implications

On the first glance, the distribution of excess energy onto both feed-in sink and excess sink may seem off to the end-user. Other than these inconveniences, there are no real implications that affect the capacity and dispatch optimization. When a degree of self-supply and self-consumption is defined, the limitation might tarnish these results.

1.15.8 Renewable energy share definition relative to energy carriers

Limitation

The current renewable energy share depends on the share of renewable energy production assets directly feeding the load. The equation to calculate the share also includes the energy carrier rating as described here below:

$$RES = \frac{\sum_i E_{RE,generation}(i) \cdot w_i}{\sum_i E_{RE,generation}(i) \cdot w_i + \sum_k E_{nonRE,generation}(k) \cdot w_k}$$

with i : renewable energy asset
 k : non-renewable energy asset

Reason

The open-plan-tool is a work in progress and this can still be addressed in the future.

Implications

This might result in different values when comparing them to other models. Another way to calculate it is by considering the share of energy consumption supplied from renewable sources.

1.15.9 Energy carrier weighting

Limitation

The open-plan-tool assumes a usable energy content rating for every energy carrier. The current version assumes that 1 kWh thermal is equivalent to 1 kWh electricity.

Reason

This is an approach that open-plan-tool currently uses.

Implications

By weighing the energy carriers according to their energy content (Gasoline Gallon Equivalent (GGE)), the open-plan-tool might result in values that can't be directly assessed. Those ratings affect the calculation of the levelized cost of the energy carriers, but also the minimum renewable energy share constraint.

1.15.10 Events of energy shortage or grid interruption cannot be modelled

Limitation

The open-plan-tool assumes no shortage or grid interruption in the system.

Reason

The aim of the open-plan-tool does not cover this scenario.

Implications

Electricity shortages due to power cuts might happen in real life and the open-plan-tool currently omits this scenario. If a system is self-sufficient but relies on grid-connected PV systems, the latter stop feeding the load if any power cuts occur and the battery storage systems might not be enough to serve the load thus resulting energy shortage.

1.15.11 Need to model one technical unit with two transformer assets

Limitation

Two transformer objects representing one technical unit in real life are currently unlinked in terms of capacity and attributed costs.

Reason

The open-plan-tool uses oemof-solph's generic components which are unidirectional so for a bidirectional asset, two transformer objects have to be used.

Implications

Since only one input is allowed, such technical units are modelled as two separate transformers that are currently unlinked in the open-plan-tool (e.g., hybrid inverter, heat pump, distribution transformer, etc.). This raises a difficulty to define costs in the input data. It also results in two optimized capacities for one logical unit.

This limitation can be addressed with a constraint which links both capacities of one logical unit, and therefore solves both the problem to attribute costs and the previously differing capacities.

1.16 Input parameters

1.16.1 Table of parameters

The input parameters are gathered in the table below. Each parameter is provided with unit, type and example values. For more information about one parameter, please click on it.

Table 7: Parameters summary

Parameter	Type	Unit	Default
<i>age_installed</i>	numeric	a	0
<i>c-rate</i>	numeric	Factor	1
<i>country</i>	str		
<i>currency</i>	str		
<i>development_costs</i>	numeric	€	0
<i>discount_factor</i>	numeric	Factor	0.05
<i>dispatch_price</i>	numeric	€/kWh	0.01
<i>efficiency</i>	numeric	Factor	0.8
<i>emission_factor</i>	numeric	kgCO ₂ eq/asset unit	0.4
<i>energy_price</i>	numeric	€/kWh	0.3
<i>energyVector</i>	str		Electricity
<i>evaluated_period</i>	numeric		365
<i>feedin_cap</i>	numeric	unit	
<i>feedin_tariff</i>	numeric	€/kWh	0.1
<i>file_name</i>	str		
<i>fixed_thermal_losses_absol</i>	numeric	factor	0
<i>fixed_thermal_losses_relati</i>	numeric	factor	0
<i>inflow_direction</i>	str		
<i>installedCap</i>	numeric	unit	0

continues on next page

Table 7 – continued from previous page

Parameter	Type	Unit	Default
<i>label</i>	str		
<i>latitude</i>	numeric		
<i>lifetime</i>	numeric	a	20
<i>longitude</i>	numeric		
<i>maximum_emissions</i>	numeric	kgCO2eq/a	
<i>maximumCap</i>	numeric		
unit			
<i>mini-mal_degree_of_autonomy</i>	numeric	factor	0.6
<i>mini-mal_renewable_factor</i>	numeric	factor	0.8
<i>net_zero_energy</i>	boolean		False
<i>optimizeCap</i>	boolean		False
<i>outflow_direction</i>	str		
<i>output_lp_file</i>	boolean		False
<i>peak_demand_pricing</i>	numeric	€/kW	0
<i>peak_demand_pricing_peri</i>	numeric	times per year	1
<i>project_duration</i>	numeric	a	20
<i>project_id</i>	str		
<i>project_name</i>	str		
<i>renewable_share</i>	numeric	Factor	0.44
<i>renewableAsset</i>	boolean		True
<i>scenario_description</i>	str		
<i>scenario_id</i>	str		
<i>scenario_name</i>	str		
<i>soc_initial</i>	numeric	None or factor	
<i>soc_max</i>	numeric	Factor	1
<i>soc_min</i>	numeric	Factor	0
<i>specific_costs</i>	numeric	€/unit:	1000
<i>specific_costs_om</i>	numeric	€/(:unit:*a)	10
<i>start_date</i>	str		
<i>storage_filename</i>	str		
<i>tax</i>	numeric	Factor	0
<i>timestep</i>	numeric		60
<i>type_oemof</i>	str		
<i>unit</i>	str		kW
<i>beta</i>	factor	beta	0

1.16.2 List of parameters

Below is the list of all the parameters of the open-plan-tool, sorted in alphabetical order. Each of the parameters has the following properties

Definition

parameter's definition, could also contain potential use cases of the parameter

Type

str (text), numeric (integer or double precision number), boolean (True or False)

Unit

physical unit

Example

an example of parameter's value

Restrictions

specific restrictions on the parameter's value (e.g., "positive integer number", "must be an even number", "must be one of ['val1', 'val2']")

Default

default parameter's value

age_installed

Definition_Short

Number of years the asset has already been in operation

Definition_Long

If the project lasts longer than its remaining lifetime, the replacement costs of the asset will be taken into account.

Type

numeric

Unit

a

Example

10

Restrictions

Natural number

Default

0

c-rate

Definition_Short

Maximum permissible power at which the storage can be charged or discharged relative to the nominal capacity of the storage

Definition_Long

The C rate indicates the reciprocal of the time for which a battery of the specified capacity can be charged or discharged with the maximum charge or discharge current. A C-rate of 1 implies that the battery can be fully charged or discharged completely in a single timestep. A C-rate of 0.5 implies that the battery needs at least 2 timesteps to be fully charged or discharged.

Type

numeric

Unit

Factor

Example

1

Restrictions

Real number between 0 and 1

Default

1

country**Definition_Short**

Name of the country where the project is being deployed

Type

str

Unit

nan

Example

Norway

Restrictions

nan

Default

nan

currency**Definition_Short**

The currency of the country where the project is implemented

Type

str

Unit

nan

Example

EUR

Restrictions

nan

Default

nan

development_costs**Definition_Short**

Planning and development costs

Definition_Long

This could be planning and development costs which do not depend on the (optimized) capacities of the assets.

Type

numeric

Unit

€

Example

10000

Restrictions

Positive real number

Default

0

discount_factor

Definition_Short

The factor which accounts for the depreciation in the value of money in the future compared to the current value of the same money

Definition_Long

The common method is to calculate the weighted average cost of capital (WACC) and use it as the discount rate.

Type

numeric

Unit

Factor

Example

0.06

Restrictions

Real number

Default

0.05

dispatch_price

Definition_Short

Costs associated with a flow through/from the asset (OPEX_var or fuel costs)

Definition_Long

This could be fuel costs for fuel sources like biogas or oil or operational costs for thermal power plants which only occur when operating the plant.

Type

numeric

Unit

€/kWh

Example

0.01

Restrictions

nan

Default

0.01

efficiency

Definition_Short

Ratio of energy output to energy input

Definition_Long

The battery efficiency is the ratio of the energy taken out from the battery to the energy put into the battery.

Type

numeric

Unit

Factor

Example

0.95

Restrictions

Positive real number

Default

0.8

emission_factor

Definition_Short

Emissions per unit dispatch of an asset

Type

numeric

Unit

kgCO₂eq/asset unit

Example

14.4

Restrictions

Positive real number

Default

0.4

energy_price

Definition_Short

Price of the energy carrier sourced from the utility grid

Definition_Long

Can be also a timeseries

Type

numeric

Unit

€/kWh

Example

0.3

Restrictions

nan

Default

0.3

energyVector

Definition_Short

Energy commodity

Definition_Long

Convention: For an energy conversion asset define the commodity of the output. For a sink define the commodity based on the input flow. For a source define the commodity based on the output flow. For a storage, define the commodity based on the stored energy carrier.

Type

str

Unit

nan

Example

Electricity

Restrictions

One of "Electricity", "Gas", "Bio-Gas", "Diesel", "Heat", "H2"

Default

Electricity

evaluated_period

Definition_Short

The number of days for which the simulation is to be run

Type

numeric

Unit

nan

Example

365

Restrictions

Natural number

Default

365

feedin_cap**Definition_Short**

Maximum capacity for feeding electricity into the grid

Type

numeric

Unit

unit

Example

1000

Restrictions

Acceptable values are either a positive real number or None

Default

nan

feedin_tariff**Definition_Short**

Price received for feeding electricity into the grid

Definition_Long

Can be also a timeseries

Type

numeric

Unit

€/kWh

Example

0.1

Restrictions

nan

Default

0.1

file_name**Definition_Short**

Name of the csv file containing the input generation or demand timeseries

Type

str

Unit

nan

Example

demand_harbor.csv

Restrictions

This file must be placed in a folder named “time_series” inside your input folder.

Default

nan

fixed_thermal_losses_absolute

Definition_Short

Thermal losses of the storage independent of the state of charge and independent of nominal storage capacity between two consecutive timesteps

Type

numeric

Unit

factor

Example

0.0003

Restrictions

Between 0 and 1

Default

0

fixed_thermal_losses_relative

Definition_Short

Thermal losses of storage independent of state of charge between two consecutive timesteps relative to nominal storage capacity

Type

numeric

Unit

factor

Example

0.0016

Restrictions

Between 0 and 1

Default

0

inflow_direction

Definition_Short

Label of the bus/component from which the energyVector is arriving into the asset

Type

str

Unit

nan

Example

Electricity

Restrictions

nan

Default

nan

installedCap**Definition_Short**

Already existing installed capacity

Definition_Long

If the project lasts longer than its remaining lifetime, the replacement costs of the asset will be taken into account.

Type

numeric

Unit

unit

Example

50

Restrictions

nan

Default

0

label**Definition_Short**

Name of the asset

Type

str

Unit

nan

Example

pv_plant_01

Restrictions

Input the names in a computer friendly format, preferably with underscores instead of spaces, and avoiding special characters

Default

nan

latitude

Definition_Short

Latitude coordinate of the project's geographical location

Type

numeric

Unit

nan

Example

45.641603

Restrictions

Should follow geographical convention

Default

nan

lifetime

Definition_Short

Number of operational years of the asset until it has to be replaced

Type

numeric

Unit

a

Example

20

Restrictions

Natural number

Default

20

longitude

Definition_Short

Longitude coordinate of the project's geographical location

Type

numeric

Unit

nan

Example

10.95787

Restrictions

Should follow geographical convention

Default

nan

maximum_emissions**Definition_Short**

Maximum amount of total emissions which are allowed in the optimized energy system

Type

numeric

Unit

kgCO₂eq/a

Example

100000

Restrictions

Acceptable values are either a positive real number or None.

Default

nan

maximumCap**Definition_Short**

Maximum total capacity of an asset that can be installed at the project site

Definition_Long

This includes the already existing installed and additional capacity possible. An example would be that a roof can only carry 50 kW PV (maximum capacity), whereas the installed capacity is already 10 kW. The optimization would only be allowed to add 40 kW PV at maximum.

Type

numeric

Unit

unit

Example

1000

Restrictions

Acceptable values are either a positive real number or None.

Default

nan

minimal_degree_of_autonomy**Definition_Short**

Minimum degree of autonomy that needs to be met by the optimization

Definition_Long

This constraint defines a lower bound on the [degree of autonomy](https://open-plan-documentation.readthedocs.io/en/latest/model/simulation_outputs.html#degree-of-autonomy) of the energy system. Note that constraint is applied to the the whole, sector-coupled energy system, but not to specific sectors individually.

Type

numeric

Unit

factor

Example

0.6

Restrictions

Real number between 0 and 1

Default

0.6

minimal_renewable_factor

Definition_Short

Minimum share of energy supplied by renewable generation that needs to be met by the optimization

Definition_Long

This constraint defines a lower bound on the [renewable share](https://open-plan-documentation.readthedocs.io/en/latest/model/simulation_outputs.html#renewable-factor) of the system, which takes into account both local generation as well as the renewable share of energy providers. Note that constraint is applied to the renewable share of the whole, sector-coupled energy system, but not to specific sectors individually.

Type

numeric

Unit

factor

Example

0.8

Restrictions

Real number between 0 and 1

Default

0.8

net_zero_energy

Definition_Short

Specifies whether optimization needs to result into a net zero energy system (True) or not (False)

Type

boolean

Unit

nan

Example

True

Restrictions

Acceptable values are either Yes or No.

Default

False

optimizeCap

Definition_Short

Choose if capacity optimization should be performed for this asset.

Type

boolean

Unit

nan

Example

True

Restrictions

Acceptable values are either Yes or No.

Default

False

outflow_direction

Definition_Short

Label of bus/component towards which the energyVector is leaving from the asset

Type

str

Unit

nan

Example

PV plant (mono)

Restrictions

nan

Default

nan

output_lp_file

Definition_Short

Enables the output of the linear programming (lp) file with the linear equation system describing the optimization problem

Definition_Long

The lp file contains the objective function and all constraints. It enables the user to look at the underlying equations of the optimization.

Type

boolean

Unit

nan

Example

False

Restrictions

Acceptable values are either True or False

Default

False

peak_demand_pricing

Definition_Short

Grid fee to be paid based on the peak demand of a given period

Type

numeric

Unit

€/kW

Example

60

Restrictions

nan

Default

0

peak_demand_pricing_period

Definition_Short

Number of reference periods in one year for the peak demand pricing

Type

numeric

Unit

times per year

Example

2

Restrictions

Only one of the following are acceptable values: 1 (yearly), 2, 3 ,4, 6, 12 (monthly)

Default

1

project_duration

Definition_Short

The number of years the project is intended to be operational

Definition_Long

The project duration also sets the installation time of the assets used in the simulation. After the project ends these assets are ‘sold’ and the refund is charged against the initial investment costs.

Type

numeric

Unit

a

Example

30

Restrictions

Natural number

Default

20

project_id**Definition_Short**

Assign a project ID as per your preference.

Type

str

Unit

nan

Example

1

Restrictions

Cannot be the same as an already existing project ID

Default

nan

project_name**Definition_Short**

Assign a project name as per your preference.

Type

str

Unit

nan

Example

Borg Havn

Restrictions

nan

Default

nan

renewable_share

Definition_Short

Share of renewables in the generation mix of the energy supplied by the DSO utility

Type

numeric

Unit

Factor

Example

0.5

Restrictions

Real number between 0 and 1

Default

0.44

renewableAsset

Definition_Short

Choose if this asset should be considered as renewable.

Definition_Long

This parameter is necessary to consider the renewable share constraint correctly.

Type

boolean

Unit

nan

Example

True

Restrictions

Acceptable values are either Yes or No.

Default

True

scenario_description

Definition_Short

Brief description of the scenario being simulated

Type

str

Unit

nan

Example

This scenario simulates a sector-coupled energy system

Restrictions

nan

Default

nan

scenario_id**Definition_Short**

Assign a scenario ID as per your preference.

Type

str

Unit

nan

Example

1

Restrictions

Cannot be the same as an already existing scenario ID within the project

Default

nan

scenario_name**Definition_Short**

Assign a scenario name as per your preference.

Type

str

Unit

nan

Example

Warehouse 14

Restrictions

nan

Default

nan

soc_initial**Definition_Short**

State of charge of the storage in the zeroth time step

Definition_Long

The state of charge is specified as a factor of the nominal capacity.

Type

numeric

Unit

None or factor

Example

nan

Restrictions

nan

Default

nan

soc_max

Definition_Short

The maximum permissible level of charge of the storage as a factor of the nominal capacity

Definition_Long

When the battery is filled to its nominal capacity the state of charge is represented by the value 1.

Type

numeric

Unit

Factor

Example

nan

Restrictions

Real number between 0 and 1

Default

1

soc_min

Definition_Short

The minimum permissible level of charge of the storage as a factor of the nominal capacity

Definition_Long

When the battery is fully discharged the state of charge is represented by the value 0.

Type

numeric

Unit

Factor

Example

nan

Restrictions

Real number between 0 and 1

Default

0

specific_costs**Definition_Short**

Specific investment costs of the asset related to the installed capacity (CAPEX)

Type

numeric

Unit

€/unit:

Example

1000

Restrictions

nan

Default

1000

specific_costs_om**Definition_Short**

Specific operational and maintenance costs of the asset related to the installed capacity (OPEX_fix)

Type

numeric

Unit

€/(:unit:*a)

Example

10

Restrictions

nan

Default

10

start_date**Definition_Short**

Date and time when the simulation starts with the first step

Definition_Long

This date will be used for the results graphs as well as the timeseries upload.

Type

str

Unit

nan

Example

2018-01-01 00:00:00

Restrictions

Acceptable format is YYYY-MM-DD HH:MM:SS

Default

nan

storage_filename

Definition_Short

Name of a csv file containing the properties of a storage component

Type

str

Unit

nan

Example

storage_01.csv

Restrictions

Follows the convention of 'storage_xx.csv' where 'xx' is a number. This file must be placed in a folder named "csv_elements" inside your input folder.

Default

nan

tax

Definition_Short

Tax factor

Type

numeric

Unit

Factor

Example

0

Restrictions

Between 0 and 1

Default

0

timestep

Definition_Short

Length of the time steps

Type

numeric

Unit

nan

Example

60

Restrictions

Can only be 60 minutes at the moment

Default

60

type_oemof**Definition_Short**

Input the type of OEMOF component

Definition_Long

A photovoltaic plant would be a source, a solar inverter would be a transformer, etc. The *type_oemof* will later on be determined through the EPA.

Type

str

Unit

nan

Example

sink

Restrictions

sink or *source* or one of the other component classes of OEMOF.

Default

nan

unit**Definition_Short**

Unit associated with the capacity of the component

Type

str

Unit

nan

Example

Storage could have units like kW or kWh, transformer station could have kVA, and so on.

Restrictions

Appropriate scientific unit

Default

kW

beta**Definition_Short**

Power loss index for CHPs, usually known as beta coefficient

Definition_Long

0.6

Type

factor

Unit

beta

Example

Between 0 and 1

Restrictions

numeric

Default

0

1.17 Outputs of a simulation

After optimization of an energy system, the open-plan-tool evaluates the simulation output. It evaluates the flows, costs and performance of the system. As a result, it can calculate a number of *key performance indicators (KPI)*, namely *economic*, *technical* and *environmental* KPI. Not all of them are displayed in the graphical current user interface, they are nevertheless listed here for information.

1.17.1 Overview of Key Performance Indicators

Technical KPI are calculated to assess the performance of a simulated energy system, ie. represent the technical system configuration and operation. They are calculated based on the asset capacities and asset dispatch. They should allow the comparison of different energy system topologies and different project sites with each other. These are the calculated technical KPI:

- *Aggregated flow*
- *Average flow*
- *Degree of Autonomy*
- *Degree of Net Zero Energy*
- *Dispatch of an asset*
- *Onsite energy fraction*
- *Onsite energy matching*
- *Optimal additional capacity*
- *Peak flow*
- *Renewable factor*
- *Renewable share of local generation*
- *Energy import*

- *Energy demand*
- *Energy excess*
- *Energy export*
- *Total local generation*
- *Total non-renewable local generation*
- *Total renewable local generation*
- *Total non-renewable energy use*
- *Total renewable energy use*

Economic KPI are calculated to assess the costs of a simulated energy system. They include the costs per asset as well as the system's overall costs. Relative values like the levelized costs of supply allow a comparison to other investment options. These are the calculated economic KPI:

- *Annual operation, maintenance and dispatch expenses*
- *Annuity*
- *Costs attributed to a specific sector*
- *Operation and maintenance costs*
- *Dispatch costs*
- *Investment costs*
- *Operation, maintenance and dispatch costs*
- *Net Present Costs (NPC)*
- *Upfront investment costs*
- *Levelized cost of throughput*
- *Levelized costs of electricity equivalent*
- *Replacement costs*

Environmental KPI are calculated to assess the impact of a simulated energy system on the environment. These are the calculated environmental KPI:

- *Specific GHG per electricity equivalent*
- *Total GHG emissions*

In the sections *economic*, *technical* and *environmental* KPI, these indicators are further defined and in *Files* the possible exportable figures and files are presented. This takes place with the following structure:

Definition

Definition of the defined KPI, can be used as tooltips.

Type

One of Numeric, Figure, Excel File, JSON, Time series, Logfile or html/pdf

Unit

Unit of the KPI, multiple units possible if KPI can be applied to individual sectors (see also: *Suffixes of KPI*).

Valid Interval

Expected valid range of the KPI. Exceptions are possible under certain conditions.

Related indicators

List of indicators that are related to the described KPI, either because they are part of its calculation or can be compared to it.

Besides these parameters attributes, the underlying equation of a specific KPI may be presented and explained, or further hints might be provided for the parameter evaluation or for special cases.

1.17.2 Suffixes of KPI

The KPI of the open-plan-tool can be calculated per asset, for each sector or for the overall system.

KPI calculated per asset are not included in the scalar results of the automatic report or in the stored Excel file, but are displayed separately. They do not need suffixes, as they are always displayed in tables next to the respective asset.

KPI calculated for each vector are specifically these KPI that aggregate the dispatch and costs of multiple assets. For cost-related KPI, such aggregating KPI have the energy vector they are describing as a suffix. An example would be the `attributed_costs` of each energy vector - the attributed costs of the electricity and H2 sector would be called `attributed_costs_electricity` and `attributed_costs_H2` respectively. For technical KPI, this suffix also applies, but additionally, due to the *energy carrier weighting*, they also feature the suffix `electricity` equivalent when the weighting has been applied. The energy demand of the system is an example: the demand per sector would be `total_demand_electricity` and `total_demand_H2`. To be able to aggregate these cost into an overall KPI for the system, the electricity equivalents of both values are calculated. They then are named `total_demand_electricity_electricity_equivalent` and `total_demand_H2_electricity_equivalent`.

KPI that describe the costs of the overall energy system do not have suffixes. Technical KPI often have the suffix `electricity_equivalent` to underline the energy carrier that the parameter is relative to.

1.17.3 Economic KPI

All the KPI related to costs described below are provided in net present value.

Net Present Costs (NPC) (`costs_total`)

Definition

Net present costs of the system for the whole project duration, includes all operation, maintenance and dispatch costs as well as the investment costs (including replacements). Applied to a single asset, the costs can also be called present costs of the asset.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Operation and maintenance costs (`costs_cost_om`) | Dispatch costs (`costs_dispatch`) | Investment costs (`costs_investment_over_lifetime`) | Operation, maintenance and dispatch costs (`costs_om_total`) | Up-front investment costs (`costs_upfront_in_year_zero`)

The Net present costs (NPC) is the present value of all the costs associated with installation, operation, maintenance and replacement of energy assets within the optimized multi-vector energy system over the whole project lifetime,

deducting the present value of the residual value of asset at project end and as well as all the revenues that it earns over the project lifetime. The capital recovery factor (CRF) is used to calculate the present value of the cash flows.

$$NPC = \sum_i (c_{specific} + c_{replacement} + c_{residual}) \cdot CAP_i + \sum_i \sum_t E_i(t) \cdot p_{dispatch}$$

Operation and maintenance costs (costs_cost_om)

Definition

Costs for fix annual operation and maintenance costs over the whole project lifetime, which do not depend on the assets dispatch but solely on installed capacity. An example would be the maintenance costs for cleaning the installed PV capacity.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Dispatch costs (costs_dispatch) | Investment costs (costs_investment_over_lifetime) | Operation, maintenance and dispatch costs (costs_om_total) | Net Present Costs (NPC) (costs_total) | Upfront investment costs (costs_upfront_in_year_zero)

Operation, maintenance and dispatch costs (costs_om_total)

Definition

Costs for annual operation and maintenance costs as well as dispatch of all assets of the energy system, for the whole project duration.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Operation and maintenance costs (costs_cost_om) | Dispatch costs (costs_dispatch) | Investment costs (costs_investment_over_lifetime) | Net Present Costs (NPC) (costs_total) | Upfront investment costs (costs_upfront_in_year_zero)

Dispatch costs (costs_dispatch)**Definition**

Dispatch costs over the whole project lifetime including all expenditures that depend on the dispatch of assets (e.g. fuel costs, electricity consumption from the external grid, costs for operation and maintainance that depend on the throughput of an asset)

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Operation and maintenance costs (costs_cost_om) | Investment costs (costs_investment_over_lifetime) | Operation, maintenance and dispatch costs (costs_om_total) | Net Present Costs (NPC) (costs_total) | Upfront investment costs (costs_upfront_in_year_zero)

Investment costs (costs_investment_over_lifetime)**Definition**

Investment costs over the whole project lifetime, including all replacement costs.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Operation and maintenance costs (costs_cost_om) | Dispatch costs (costs_dispatch) | Operation, maintenance and dispatch costs (costs_om_total) | Net Present Costs (NPC) (costs_total) | Upfront investment costs (costs_upfront_in_year_zero) | Replacement costs (replacement_costs_during_project_lifetime)

Upfront investment costs (costs_upfront_in_year_zero)**Definition**

The costs which will have to be paid upfront when project begins, ie. In year 0. These are the investment and fix project costs into the chosen configuration.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Operation and maintenance costs (costs_cost_om) | Dispatch costs (costs_dispatch) | Investment costs

(costs_investment_over_lifetime) | Operation, maintenance and dispatch costs (costs_om_total) | Net Present Costs (NPC) (costs_total)

Replacement costs (replacement_costs_during_project_lifetime)

Definition

Costs for replacement of assets which occur over the project lifetime.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Investment costs (costs_investment_over_lifetime)

Costs attributed to a specific sector (attributed_costs)

Definition

Costs attributed to supplying the demand of a specific sector, based on the *net present costs (NPC)* of the energy system and the share of the sector demand compared to the overall system demand.

Type

Numeric

Unit

currency

Valid Interval

≥ 0

Related indicators

Net Present Costs (NPC) (costs_total)

A multi-vector energy system connects energy vectors into a joined energy system and the system is then designed to have an optimal, joined operation. With other systems, the costs associated to each individual energy vector would be used to calculate the costs to supply the individual sector. With the multi-vector system, this could lead to distorted costs - for example if there is a lot of PV (electricity sector), which in the end is only supplying an electrolyzer (H2 sector). The investment and operational costs of the electricity sector assets would thus turn out to be very high, which could be considered unfair as the electricity from PV is solely used to provide the H2 demand. Therefore, we define the *attributed costs of each energy vector*, to determine how much of the overall system costs should be attributed to one sector, depending on the energy demand it has compared to the other sectors. To be able to compare the demands of different energy carriers, *energy carrier weighting* is applied.

Annuity (annuity_total)**Definition**

Annuity of the assets costs over the project lifetime or the energy system's *net present costs (NPC)* .

Type

Numeric

Unit

currency/a

Valid Interval

≥ 0

Related indicators

Annual operation, maintenance and dispatch expenses (annuity_om) | Net Present Costs (NPC) (costs_total)

Annual operation, maintenance and dispatch expenses (annuity_om)**Definition**

Annuity of the operation, maintenance and dispatch costs of the asset or energy system, i.e. ballpark number of the annual expenses for asset or system operation.

Type

Numeric

Unit

currency/a

Valid Interval

≥ 0

Related indicators

Annuity (annuity_total) | Operation, maintenance and dispatch costs (costs_om_total)

Levelized costs of electricity equivalent (levelized_costs_of_electricity_equivalent)**Definition**

Levelized cost of energy of the sector-coupled energy system, calculated from the systems annuity and the total system demand in electricity equivalent.

Type

Numeric

Unit

currency/kWheq

Valid Interval

≥ 0

Related indicators

Net Present Costs (NPC) (costs_total) | Energy demand (total_demand)

Specific electricity supply costs, eg. levelized costs of electricity are commonly used to compare the supply costs of different investment decisions or also energy provider prices to local generation costs. However, a multi-vector energy system connects energy vectors into a joined energy system and the optimization objective of the open-plan-tool then is to minimize the overall energy costs, without distinguishing between the different sectors. This sector-coupled energy system is then designed to have an optimal, joined operation. With other systems, the costs associated to each individual

energy vector would be used to calculate the levelized costs of energy (LCOEnergy). With the multi-vector system, this could lead to distorted costs - for example if there is a lot of PV (electricity sector), which in the end is only supplying an electrolyzer (H2 sector). The LCOE of electricity would thus turn out to be very high, which could be considered unfair as the electricity from PV is solely used to provide the H2 demand. Therefore, we define the *attributed costs of each energy vector*, to determine how much of the overall system costs should be attributed to one sector, depending on the energy demand it has compared to the other sectors. To be able to compare the demands of different energy carriers, *energy carrier weighting* is applied.

Therefore the levelized costs of energy (LCOEnergy) for energy carrier i are defined based on the annuity of the attributed costs, the CRF and the demand of one energy sector $E_{dem,i}$:

$$LCOEnergy_i = \frac{Attributed\ costs \cdot CRF}{\sum_t E_{dem,i}(t)}$$

The LCOEnergy are calculated for each sector (resulting in the levelized costs of electricity, heat, H2...), but also for the overall energy system. For the overall energy system, the levelized costs of electricity equivalent are calculated, as this system may supply different energy vectors.

Levelized cost of throughput (levelized_cost_of_energy_of_asset)

Definition

Cost per kWh throughput through an asset, based on the assets costs during the project lifetime as well as the total throughput through the asset in the project lifetime. For generation assets, equivalent to the levelized cost of generation.

Type

Numeric

Unit

currency/kWh

Valid Interval

≥ 0

Related indicators

Annuity (annuity_total) | Aggregated flow (annual_total_flow)

This KPI measures the cost of generating 1 kWh for each asset in the system. It can be used to assess and compare the available alternative methods of energy production. The levelized cost of energy of an asset ($LCOE\ ASSET_i$) is usually obtained by looking at the lifetime costs of building and operating the asset per unit of total energy throughput of an asset over the assumed lifetime [currency/kWh].

Since not all assets are production assets, the open-plan-tool distinguishes between the type of assets. For assets to convert or produce energy the open-plan-tool calculates the $LCOE\ ASSET_i$ by dividing the total annuity a_i of the asset i by the total flow $\sum_t E_i(t)$.

$$LCOE\ ASSET_i = \frac{a_i}{\sum_t E_i(t)}$$

For assets that store energy, the open-plan-tool sums the annuity for storage capacity $a_{i,sc}$, input power $a_{i,ip}$ and output power $a_{i,op}$ and divides it by the output power total flow $\sum_t E_{i,op}(t)$.

$$LCOE\ ASSET_i = \frac{a_{i,sc} + a_{i,ip} + a_{i,op}}{\sum_t E_{i,op}(t)}$$

If the total flow is 0 in any of the previous cases, then the $LCOE\ ASSET$ is set to None.

$$LCOE\ ASSET_i = None$$

For assets that consume energy, the open-plan-tool outputs 0 for the $LCOE\ ASSET_i$.

$$LCOE\ ASSET_i = 0$$

1.17.4 Technical KPI

Optimal additional capacity (optimizedAddCap)

Definition

Capacity added to installed capacity for optimal economic system performance.

Type

Numeric

Unit

kW, kWh, kWp, ...

Valid Interval

≥ 0

Related indicators

Peak flow (peak_flow)

Dispatch of an asset (flow)

Definition

Optimized dispatch of an asset in the optimized energy system, ie. its generation or throughput.

Type

Time series (with time stamps and values)

Unit

kW, kgH2, ...

Valid Interval

nan

Related indicators

plot_dispatch | *Peak flow (peak_flow)* | *Aggregated flow (annual_total_flow)* | *Average flow (average_flow)*

Peak flow (peak_flow)

Definition

Peak of the dispatch of an asset.

Type

Numeric

Unit

kW

Valid Interval

≥ 0

Related indicators

Average flow (average_flow) | *Aggregated flow (annual_total_flow)*

Average flow (average_flow)**Definition**

Average value of the assets dispatch. The ratio of average dispatch to peak dispatch indicates how much the asset is used in comparison to its actual installed capacity.

Type

Numeric

Unit

kWh

Valid Interval

≥ 0

Related indicators

Aggregated flow (annual_total_flow) | Peak flow (peak_flow)

Aggregated flow (annual_total_flow)**Definition**

Dispatch of the asset over a year, aggregated generation, demand or throughput.

Type

Numeric

Unit

kWh

Valid Interval

≥ 0

Related indicators

Average flow (average_flow) | Peak flow (peak_flow)

Energy demand (total_demand)**Definition**

Demand of energy in local energy system over a the project lifetime.

Type

Numeric

Unit

kWh, kWheq, ...

Valid Interval

≥ 0

Related indicators

None

Energy export (total_feedin)**Definition**

Feed-in of energy into external grid.

Type

Numeric

Unit

kWh, kWheq, ...

Valid Interval

≥ 0

Related indicators

Onsite energy fraction (onsite_energy_fraction)

Energy import (total_consumption_from_energy_provider)**Definition**

Aggregated energy imports into the local energy system from the provider.

Type

Numeric

Unit

kWh, kWheq, ...

Valid Interval

≥ 0

Related indicators

None

Total non-renewable local generation (total_internal_non-renewable_generation)**Definition**

Aggregated amount of non-renewable energy generated within the energy system.

Type

Numeric

Unit

kWheq

Valid Interval

≥ 0

Related indicators

Total local generation (total_internal_generation) | Total renewable local generation (total_internal_renewable_generation)

Total renewable local generation (total_internal_renewable_generation)**Definition**

Aggregated amount of renewable energy generated within the energy system.

Type

Numeric

Unit

kWheq

Valid Interval

≥ 0

Related indicators

Total local generation (total_internal_generation) | Total non-renewable local generation (total_internal_non-renewable_generation)

Total local generation (total_internal_generation)**Definition**

Aggregated amount of energy generated within the energy system.

Type

Numeric

Unit

kWheq

Valid Interval

≥ 0

Related indicators

Total non-renewable local generation (total_internal_non-renewable_generation) | Total renewable local generation (total_internal_renewable_generation)

Energy excess (total_excess)**Definition**

Excess of energy, ie. unused energy in local energy system.

Type

Numeric

Unit

kWh, kWheq, ...

Valid Interval

≥ 0

Related indicators

None

Total renewable energy use (total_renewable_energy_use)**Definition**

Aggregated amount of renewable energy used within the energy system (ie. Including local generation and external supply).

Type

Numeric

Unit

kWheleq

Valid Interval

≥ 0

Related indicators

Total non-renewable energy use (total_non-renewable_energy_use)

Total non-renewable energy use (total_non-renewable_energy_use)**Definition**

Aggregated amount of non-renewable energy used within the energy system (ie. Including local generation and external supply).

Type

Numeric

Unit

kWheleq

Valid Interval

≥ 0

Related indicators

Total renewable energy use (total_renewable_energy_use)

Renewable share of local generation (renewable_share_of_local_generation)**Definition**

The renewable share of local generation describes how much of the energy generated locally is produced from renewable sources. It does not take into account the consumption from energy providers.

Type

Numeric

Unit

Factor

Valid Interval

$[0,1]$

Related indicators

Renewable factor (renewable_factor)

The renewable share of local generation describes how much of the energy generated locally is produced from renewable sources. It does not take into account the consumption from energy providers.

The renewable share of local generation for each sector does not utilize energy carrier weighting but has a limited, single-vector view:

$$REGen_v = \frac{\sum_i E_{r\text{gen},i}}{\sum_j E_{\text{gen},j}}$$

with v : Energy vector

$r\text{gen}$: Renewable generation

gen : Renewable and non-renewable generation

i, j : Asset 1,2,...

For the system-wide share of local renewable generation, energy carrier weighting is used:

$$REGen = \frac{\sum_i E_{r\text{gen},i} \cdot w_i}{\sum_j E_{\text{gen},j} \cdot w_j}$$

with $r\text{gen}$: Renewable generation

gen : Renewable and non-renewable generation

i, j : Assets 1,2,...

w_i, w_j : Energy carrier weighting factor for output of asset i/j

Example

An energy system is composed of a heat and an electricity side. Following are the energy flows:

- 100 kWh from a local PV plant
- 0 kWh local generation for the heat side

This results in:

- A single-vector renewable share of local generation of 0% for the heat sector.
- A single-vector renewable share of local generation of 100% for the electricity sector.
- A system-wide renewable share of local generation of 100%.

Renewable factor (renewable_factor)

Definition

Describes the share of the energy influx to the local energy system that is provided from renewable sources. This includes both local generation as well as consumption from energy providers.

Type

Numeric

Unit

Factor

Valid Interval

[0,1]

Related indicators

Renewable share of local generation (renewable_share_of_local_generation) | Onsite energy fraction (onsite_energy_fraction) | Onsite energy matching (onsite_energy_matching)

Describes the share of the energy influx to the local energy system that is provided from renewable sources. This includes both local generation as well as consumption from energy providers.

$$RF = \frac{\sum_i E_{r\text{gen},i} \cdot w_i + RES \cdot E_{grid}}{\sum_j E_{gen,j} \cdot w_j + \sum_k E_{grid}(k) \cdot w_k}$$

with *r_{gen}*: Renewable generation

gen: Renewable and non-renewable generation

i, j: Assets 1,2,...

RES: Renewable energy share of energy provider

k: Energy provider 1,2...

w_i, w_j, w_k: Energy carrier weighting factor for output of asset *i/j/k*

Example

An energy system is composed of a heat and an electricity side. Following are the energy flows:

- 100 kWh from a local PV plant
- 0 kWh local generation for the heat side
- 100 kWh consumption from the electricity provider, who has a renewable factor of 50%

Again, the heat sector would have a renewable factor of 0% when considered separately, and the electricity side would have an renewable factor of 75%. This results in a system-wide renewable share of:

$$RF = \frac{100kWh(el) \cdot \frac{kWh(e\text{leq})}{kWh(el)} + 50kWh(el) \cdot \frac{kWh(e\text{leq})}{kWh(el)}}{200kWh(el) \cdot \frac{kWh(e\text{leq})}{kWh(el)}} = 3/4 = 75 \%$$

The renewable factor, just like the *Renewable share of local generation (renewable_share_of_local_generation)*, cannot indicate how much renewable energy is used in each of the sectors. In the future, it might be possible to get a clearer picture of the flows between the sectors with the proposed *degree of sector-coupling*.

Degree of sector-coupling (DSC)

To assess how much an optimized multi-vector energy system makes use of the potential of sector-coupling, it is planned to introduce the degree of sector-coupling in the future. This level of interconnection is to be calculated with the ratio of energy flows in between the sectors (ie. those, where energy carriers are converted to another energy carrier) to the energy demand supplied:

$$DSC = \frac{\sum_{i,j} E_{conversion}(i,j) \cdot w_i}{\sum_i E_{demand}(i) \cdot w_i}$$

with *i, j*: Electricity,H2...

Note: This feature is currently not implemented.

Onsite energy fraction (onsite_energy_fraction)

Definition

Onsite energy fraction is also referred to as self-consumption. It describes the fraction of all locally generated energy that is consumed by the system itself.

Type

Numeric

Unit

Factor

Valid Interval

[0,1]

Related indicators

Onsite energy matching (onsite_energy_matching)

Onsite energy fraction is also referred to as “self-consumption”. It describes the fraction of all locally generated energy that is consumed by the system itself. (see [1] and [2]).

An OEF close to zero shows that only a very small amount of locally generated energy is consumed by the system itself. It is at the same time an indicator that a large amount is fed into the grid instead. A OEF close to one shows that almost all locally produced energy is consumed by the system itself.

$$OEF = \frac{\sum_i (E_{generation}(i) - E_{gridfeedin}(i)) \cdot w_i}{\sum_i E_{generation}(i) \cdot w_i}$$

$$OEF \in [0,1]$$

Onsite energy matching (onsite_energy_matching)

Definition

The onsite energy matching is also referred to as self-sufficiency. It describes the fraction of the total demand that can be covered by the locally generated energy.

Type

Numeric

Unit

Factor

Valid Interval

[0,1]

Related indicators

Onsite energy fraction (onsite_energy_fraction) | Energy export (total_feedin)

The onsite energy matching is also referred to as “self-sufficiency”. It describes the fraction of the total demand that can be covered by the locally generated energy (see [1] and [2]).

An OEM close to zero shows that very little of the demand can be covered by locally produced energy. An OEM close to one shows that almost all of the demand can be covered with locally generated energy. Per definition OEM cannot be greater than 1 because the excess generated energy would automatically be fed into the grid or an excess sink.

$$OEM = \frac{\sum_i (E_{generation}(i) - E_{gridfeedin}(i) - E_{excess}(i)) \cdot w_i}{\sum_i E_{demand}(i) \cdot w_i}$$

$$OEM \in [0,1]$$

Note: The feed into the grid should only be positive.

Degree of Autonomy (degree_of_autonomy)

Definition

A degree of autonomy close to zero shows high dependence on the energy provider, while a degree of autonomy of 1 represents an autonomous or net-energy system and a degree of autonomy higher 1 a surplus-energy system.

Type

Numeric

Unit

Factor

Valid Interval

[0,1]

Related indicators

Energy demand (total_demand)

The degree of autonomy describes the overall energy consumed minus the energy consumed from the grid divided by the overall energy consumed. Adapted from this definition [3].

A degree of autonomy close to zero shows high dependence on the grid operator, while a degree of autonomy of one represents an autonomous system. Note that this key parameter indicator does not take into account the outflow from the system to the grid operator (also called feedin). As above, we apply a weighting based on Electricity Equivalent.

$$DegreeofAutonomy = \frac{\sum_i E_{demand,i} \cdot w_i - \sum_j E_{consumption,provider,j} \cdot w_j}{\sum_i E_{demand,i} \cdot w_i}$$

Degree of Net Zero Energy (degree_of_nze)

Definition

The degree of net zero energy describes the ability of an energy system to provide its aggregated annual demand through local sources.

Type

Numeric

Unit

Factor

Valid Interval

≥ 0

Related indicators

Energy export (total_feedin) | Energy import (total_consumption_from_energy_provider)

The degree of net zero energy describes the ability of an energy system to provide its aggregated annual demand through local sources. For that, the balance between local generation as well as consumption from and feed-in towards the energy provider is compared. In a net zero energy system, demand can be supplied by energy import, but then local energy generation must provide an equally high energy export of energy in the course of the year. In a plus energy system, the export exceeds the import, while local generation can supply all demand (from an aggregated perspective). To calculate the degree of NZE, the margin between grid feed-in and grid consumption is compared to the overall demand.

Some definitions of NZE systems require that the local demand is solely covered by locally generated renewable energy. In the open-plan-tool this is not the case - all locally generated energy is taken into consideration. For information about the share of renewables in the local energy system checkout *Renewable share of local generation (renewable_share_of_local_generation)*.

A degree of NZE lower than 1 shows that the energy system can not reach a net zero balance, and indicates by how much it fails to do so, while a degree of NZE of 1 represents a net zero energy system and a degree of NZE higher 1 a plus-energy system.

As above, we apply a weighting based on Electricity Equivalent.

$$\text{Degree of NZE} = 1 + \frac{\sum_i (E_{\text{gridfeedin}}(i) - E_{\text{gridconsumption}}(i)) \cdot w_i}{\sum_i E_{\text{demand},i} \cdot w_i}$$

1.17.5 Environmental KPI

Total GHG emissions (total_emissions)

Definition

Total greenhouse gas emissions in kg.

Type

Numeric

Unit

kg GHG_{eq}

Valid Interval

>=0

Related indicators

Renewable factor (renewable_factor) | Specific GHG per electricity equivalent (specific_emissions_per_electricity_equivalent)

The total emissions of the MES in question are calculated with all aggregated energy flows from the generation assets including energy providers and their subsequent emission factor:

$$\text{Total_emissions} = \sum_i E_{\text{gen}}(i) \cdot \text{emission_factor}(i)$$

with i : generation assets 1,2,...

The emissions of each generation asset and provider are also calculated and displayed separately in the outputs of the open-plan-tool.

Specific GHG per electricity equivalent (specific_emissions_per_electricity_equivalent)

Definition

Specific GHG emissions per supplied electricity equivalent.

Type

Numeric

Unit

kg GHG_{eq}/kWh

Valid Interval

>=0

Related indicators*Total GHG emissions (total_emissions)*

The specific emissions per electricity equivalent of the MES are calculated in $\text{kg/kWh}_{\text{elec}}$:

$$\text{Specific_emissions} = \frac{\text{Total_emissions}}{\text{total_demand}_{\text{elec}}}$$

Emissions can be of different nature: CO2 emissions, CO2 equivalents, greenhouse gases, ...

Currently the emissions do not include life cycle emissions of energy conversion or storage assets, nor are they calculated separately for the energy sectors. For the latter, the problem of the assignment of assets to sectors arises e.g. emissions caused by an electrolyser would be counted to the electricity sector although you might want to count it for the H2 sector, as the purpose of the electrolyser is to feed the H2 sector. Therefore, we will have to verify whether or not we can apply the energy carrier weighting also for this KPI.

1.18 License

MIT License

Copyright (c) (for the two commits dated on 2021-12-13) Intracom SA Telecom Solutions

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Copyright (c) from 2021-07-09 (any commit not made on 2021-12-13) Reiner Lemoine Institut

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.19 Contact Us

You can address your questions about open-plan-tool at the following email:

open_plan@rl-institut.de

1.20 About project

1.20.1 Project's page

https://reiner-lemoine-institut.de/open_plan_2020/

1.20.2 Project's website

<https://open-plan-tool.org>

1.20.3 Reuses

The source code of the project's graphical interface has been forked twice

<https://github.com/in-RET/open-plan-gui> and https://github.com/rl-institut/cp_nigeria_app

1.21 Credits

1.21.1 Funding

Bundesministerium für Wirtschaft und Energie (<https://www.bmwk.de>)

1.21.2 Previous work

This code is based from previous open-source work done building a user interface to the [multi-vector-simulator](#) tool in the [Horizon2020](#) ELAND project. In open_plan project's scope a new design and more features are added, based on feedback collected in workshops held with stakeholders.