q100opt

Release 0.0.0

May 03, 2021

Contents

CHAPTER 1

Overview

docs	
tests	
package	

Model builder for oemof-solph optimisation models with a focus an district energy systems.

- Free software: MIT license

1.1 Installation

You can install the in-development version with:

```
pip install https://github.com/quarree100/q100opt/archive/master.zip
```

1.2 Documentation

https://q100opt.readthedocs.io/

1.3 Development

To run all the tests run:

```
tox
```

Note, to combine the coverage data from all the tox environments run:

Windows	`set PYTEST_ADDOPTS=--cov-append` `tox`
Other	`PYTEST_ADDOPTS=--cov-append tox`

CHAPTER 2

Installation

At the command line:

```
pip install q100opt
```

CHAPTER 3

Usage

To use q100opt in a project:

```
import q100opt
```

CHAPTER 4

Reference

4.1 q100opt

4.1.1 setup_model module

Function for reading data and setting up an oemof-solph EnergySystem.

Please use this module with care. It is work in progress!

Contact: Johannes Röder <johannes.roeder@uni-bremen.de>

SPDX-License-Identifier: MIT

q100opt.setup_model.**add_buses**(*table*)

Instantiates the oemof-solph.Buses based on tabular data.

Retruns the Buses in a Dictionary and in a List. If excess and shortage is given, additional sinks and sources are created.

Parameters **table** (*pandas.DataFrame*) – Dateframe with all Buses.

Returns

tuple (*a tuple containing:*) –

- **nodes ([list]): A list with all oemof-solph Buses of the** Dataframe table.
- **busd ([dict]): Dictionary with all oemof Bus object.** Keys are equal to the label of the bus.

Examples

```
>>> import pandas as pd
>>> from q100opt.setup_model import add_buses
>>> data_bus = pd.DataFrame([['label_1', 0, 0, 0, 0],
```

(continues on next page)

(continued from previous page)

```
... ['label_2', 0, 0, 0, 0]],
... columns=['label', 'excess', 'shortage', 'shortage_costs',
... 'excess_costs'])
>>> nodes, buses = add_buses(data_bus)
```

q100opt.setup_model.**add_links**(*tab*, *busd*)

...

q100opt.setup_model.**add_sinks**(*tab*, *busd*, *timeseries=None*)

Parameters

- **tab** (*pd.DataFrame*) – Table with parameters of Sinks.
- **busd** (*dict*) – Dictionary with Buses.
- **timeseries** (*pd.DataFrame*) – (Optional) Table with all timeseries parameters.

Returns **list** (*oemof Sinks (non fix sources) objects.*)

Note: No investment possible.

---

q100opt.setup_model.**add_sinks_fix**(*tab*, *busd*, *timeseries*)

Add fix sinks, e.g. for energy demands.

Parameters

- **tab** (*pd.DataFrame*) – Table with parameters of Sinks.
- **busd** (*dict*) – Dictionary with Buses.
- **timeseries** (*pd.DataFrame*) – (Required) Table with all timeseries parameters.

Returns **list** (*oemof Sinks (non fix sources) objects.*)

Note: No investment possible.

---

q100opt.setup_model.**add_sources**(*tab*, *busd*, *timeseries=None*)

Parameters

- **tab** (*pd.DataFrame*) – Table with parameters of Sources.
- **busd** (*dict*) – Dictionary with Buses.
- **timeseries** (*pd.DataFrame*) – (Optional) Table with all timeseries parameters.

Returns **list** (*Oemof Sources (non fix sources) objects.*)

q100opt.setup_model.**add_sources_fix**(*tab*, *busd*, *timeseries*)

Parameters

- **tab** (*pd.DataFrame*) – Table with parameters of Sources.
- **busd** (*dict*) – Dictionary with Buses.
- **timeseries** (*pd.DataFrame*) – Table with all timeseries parameters.

Returns **list** (*List with oemof Source (only fix source) objects.*)

Note: At the moment, there are no additional flow attributes allowed, and *nominal_value* must be given in the table.

q100opt.setup_model.**add_storages**(*tab*, *busd*)

Parameters

- **tab** (*pd.DataFrame*) – Table with parameters of Storages.
- **busd** (*dict*) – Dictionary with Buses.

Returns **list** (*oemof GenericStorage components.*)

q100opt.setup_model.**add_transformer**(*tab*, *busd*, *timeseries=None*)

Parameters

- **tab** (*pandas.DataFrame*) – Table with all Transformer parameter
- **busd** (*dict*) – Dictionary with all oemof-solph Bus objects.
- **timeseries** (*pandas.DataFrame*) – Table with all Timeseries for Transformer.

Returns **list** (*oemof-solph Transformer objects.*)

q100opt.setup_model.**check_active**(*dct*)

Checks for active components.

Delete not "active" rows, and the column 'active' of all components dataframes.

Parameters **dct** (*dict*) – Holding the Dataframes of solph components

Returns *dict*

q100opt.setup_model.**check_nonconvex_invest_type**(*dct*)

Checks if flow attribute 'invest.nonconvex' is type bool, if the attribute is present.

Parameters **dct** (*dict*) – Dictionary with all paramerters for the oemof-solph components.

Returns **dict** (*Updated Dictionary is returned.*)

q100opt.setup_model.**get_flow_att**(*row*, *ts*)

Parameters

- **row** (*pd.Series*) – Series with all attributes given by the parameter table (equal 1 row)
- **ts** (*pd.DataFrame*) – DataFrame with all input time series for the oemof-solph model.

Returns **dict** (*All Flow specific attribues.*)

q100opt.setup_model.**get_invest_obj**(*row*)

Filters all attributes for the investment attributes with the prefix'invest.', if attribute 'investment' occurs, and if attribute *investment* is set to 1.

If the invest attribute "offset" is given and if it is > 0, the invest attribute "nonconvex=True" is added.

Parameters **row** (*pd.Series*) – Parameters for single oemof object.

Returns *dict*

q100opt.setup_model.**load_csv_data**(*path*)

Loading csv data.

Loading all csv files of the given path as pandas DataFrames into a dictionary. The keys of the dictionary are the names of the csv files (without .csv).

Parameters **path** (*str*)

Returns *dict*

q100opt.setup_model.**load_xlsx_data**(*filename*)
Reads all sheets of xlsx file into dictionary.

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

5.1 Bug reports

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.2 Documentation improvements

q100opt could always use more documentation, whether as part of the official q100opt docs, in docstrings, or even on the web in blog posts, articles, and such.

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/quarree100/q100opt/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that code contributions are welcome :)

5.4 Development

To set up *q100opt* for local development:

1. Fork q100opt (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:YOURGITHUBNAME/q100opt.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes run all the checks and docs builder with tox one command:

```
tox
```

5. Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- pytest -k test_myfeature
```

To run all the test environments in *parallel*:

```
tox -p auto
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 6

Authors

- quarree100 modeling team - https://quarree100.de/

CHAPTER 7

Changelog

7.1 0.0.0 (2020-08-26)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

q

Index