examples Documentation

Isa

Jul 18, 2019
# Contents

## 1 OMEGAalpes Examples
1.1 Basic example: PV self-consumption .................................................. 3
1.2 Electrical system operation ................................................................. 4
1.3 Storage design ................................................................................. 7
1.4 Waste heat recovery ......................................................................... 9

## 2 Article study cases
2.1 Basic example: PV self-consumption .................................................. 17
2.2 Multi-actor modelling for MILP energy system optimisation: application to collective self consumption 17

## 3 Getting started tools for Jupyter Notebook
3.1 I. Installation through Anaconda ....................................................... 21
3.2 II. Installation through PIP ................................................................. 22
3.3 III. How to use the Jupyter Notebook ............................................... 22
OMEGAlpes stands for Generation of Optimization Models As Linear Programming for Energy Systems. It is an Open Source energy systems modelling tool for linear optimisation (LP, MILP).

Various examples and study cases have been developed on OMEGAlpes. They are stored in the following Gitlab: OMEGAlpes Examples

Examples are developed to help new omegalpes users,
Article study cases are developed for scientific concerns.
They are using a specified graph representation described here: OMEGAlpes Representation

To run both, you will first need to install OMEGAlpes library.
Please, have a look to the documentation: OMEGAlpes Documentation
Or to the OMEGAlpes Gitlab

Some examples are also developed on Jupyter notebook for a better understanding.

**Note:** The examples may be updated with the last developer version which may be different from the user (Pypi) version. The OMEGAlpes version used is indicated at the beginning of the example module.

**OMEGAlpes Examples and article study cases**
Please have a look to the following examples:

- Basic example: PV self-consumption
- Electrical system operation
- Storage design
- Waste heat recovery

The code is stored at the Gitlab: OMEGAlpes Examples in the folder “beginner_examples” or “examples”. Some of them have also been developed on a Jupyter notebook for a better understanding. If needed, have a look to the help on Jupyter notebook.

1.1 Basic example: PV self-consumption

In this PV self-consumption example, a single-house with roof-integrated photovoltaic panels (PV) is studied. Specifically, the study case is about demand-side management in order to maximize the self-consumption, by shifting two household appliances consumption (clothes washing machine and clothes dryer) and using a water tank for the domestic hot water consumption.

This example is available online at: PV self-consumption Examples
The code is available here: PV self-consumption code
1.2 Electrical system operation

This first module is an example of decision support for electrical system operations. The electrical system operator needs to decide whether to provide electricity from the grid_production A or B depending on their operating costs. The two grid productions are providing energy to a dwelling with a fixed electricity consumption profile.

The code is available here: Electrical system operation code

Notebook of electrical system operation example

1.2.1 COST PRODUCTION MINIMIZATION

This Module is an example of decision support for electricity grid_productions

The electrical system operator needs to decide whether to provide electricity from the grid_production A or B depending on their operating costs. The module includes : - A dwelling load with a fixed consumption profile - Two electricity grid_productions with variable operating costs

Objective :

The objective consists in minimizing the electricity grid_productions operating costs
Fig. 2: Figure 2: Principle diagram of the electrical system operation example
Operating steps:

1. Creating an empty model
2. Creating the unit dedicated to time management, and importing time-dependent data
3. Creating the dwelling consumption
4. Creating the production units
5. Objective: Minimizing the part of the electrical load covered by the electrical production plants
6. Creating the energy nodes and connecting units
7. Adding the energy node to the model
8. Optimisation process

```python
from StudyCase1 import dashboard, production_optim, print_results_optimization
```

Input parameters:

- Consumption profile is a file of 24 values of consumption in column
- Operating costs for both of the production unit has to be fulfilled for each hour of the day

```
dashboard(production_optim, print_results_optimization)
```

---

Dwelling consumption = 3353.0 kWh.
grid production A production = 2226.9999 kWh
grid production B production = 1126.0001 kWh

![Dwelling consumption graph]
1.3 Storage design

The storage_design module is an example of storage capacity optimization. A production unit and a storage system power a load with a fixed consumption profile. The production unit has a maximum power value and the storage system has maximum charging and discharging power values. The objective is to minimize the capacity of the storage system while meeting the load during the whole time horizon.

The code is available here: Storage design code

Notebook of storage design study case
1.3.1 STORAGE CAPACITY OPTIMIZATION

This Module is an example of storage capacity optimisation.

This example describes a simple microgrid with:
- A load profile known in advance
- An adjustable production unit, with a maximum power limit
- A storage system with power in charge and power in discharge

Objective:

The objective consists in minimizing the storage capacity while meeting the load during the whole time horizon.

Operating steps:

1. Creates an empty model
2. Creates the load or the load profile is already known
3. Creates the production unit - The production profile is unknown
4. Creates the storage
5. Objective: Minimize the storage capacity
6. Creates the energy node
7. Connect all units on the same energy node
8. Add node to the model
9. Optimisation process

[1]: from StudyCase2 import dashboard, storage, print_results_storage

Input parameters:

- The load profile has to define by values for each hours of the day. There is no incoherent value as long as you make them relative to each others

Load profile:

\[ [4,5,6,2,3,4,7,8,13,24,18,16,17,12,20,15,17,21,25,23,18,16,13,4] \]
• The maximum power has to be defined in kWh
• The maximum charging and discharging power have to be set in kW

---

Optimisation results

The optimal storage capacity is 47.0 kWh

Preparing to plot the energetic flows through the node energy_node.
Add power from load.
Add power from production.
Add power from storage.

1.4 Waste heat recovery

In the waste_heat_recovery module, an electro-intensive industrial process consumes electricity and rejects heat. This waste heat is recovered by a system composed of a heat pump in order to increase the heat temperature, and a thermal storage that is used to recover more energy and have a more constant use of the heat pump. This way, the waste heat is whether recovered or dissipated depending on the waste heat recovery system sizing. The heat is then injected on a district heat network to provide energy to a district heat load. A production unit of the district heat network provides the extra heat.

The code is available here: Waste heat recovery code

Notebook of waste heat recovery study case
Figure 4: Principle diagram of the waste heat recovery example.
1.4.1 WASTE HEAT RECOVERY:

This Module is an example of waste energy recovery

The system includes: - An electro-intensive industry - A dissipation load - A thermal storage system - A heat pump - A district heat network load - A district heat network production unit
Objective:
An electro-intensive industrial process consumes electricity and rejects heat. This waste energy is whether recovered or dissipated depending on the waste recovery system sizing. A storage system and a heat pump are used in order to recover the waste energy, which is then injected on a district heat network to provide heat to a district heat load. The missing heat will be provided by a district heat network production unit.

**Operating steps:**

1. Creating an empty model
2. Creating the unit dedicated to time management
3. Importing time-dependent data from files
4. Creating the electro-intensive industry unit
5. Creating unit for heat dissipation from the industrial process
6. Creating the thermal storage
7. Creating the heat pump
8. Creating the district heat load
9. Creating the heat production plants
10. Creating the heat node for the energy flows
11. Connecting units to the nodes
12. Minimizing the part of the heat load covered by the heat production plant
13. Adding all nodes (and connected units) to the optimization model
14. Writing into lp file
15. Running optimization and update values

**Input Parameters:**

1. Electricity-to-Heat conversion (0 to 1):
   - How much electricity consumption is converted to heat consumption

**Thermal storage parameters**

2. The maximal charging and discharging powers:
   - The maximum of transfer the storage is capable of transfer it. Défault: 5 MW = 5000.
3. The minimum charging and discharging powers:
   - The minimum of transfer the storage is capable of transfer.
   - When charging/discharging, the power should at least be 20% of the maximal charging/discharging powers
4. Storage capacity:
   - The maximum of power possible to store. Défault: 20MWh = 20000.
5. Initial state of charge (0 to 1):
   - How much power the store had before start the optimisation. Défault: 25% = 0.25
Heat pump parameters

6. The coefficient of performance

7. The heat pump has a electrical power limit

```
from StudyCases3 import waste, waste_results, dashboard
dashboard()
```

--- OPTIMIZATION RESULTS ---
District consumption = 367860.0 kWh.
Industry consumption = 167938.5 kWh.
District heat network production = 343513.85 kWh.
Industry heat exported = 15114.46 kWh.
Heat pump electricity consumption = 8115.38 kWh.
4 % of the load coming from the industry

Preparing to plot the energetic flows through the node heat_node_bef_valve.
  Add power from indus_heat_prod.
  Add power from dissipation.
  Add power from heat_node_bef_valve.

Preparing to plot the energetic flows through the node heat_node_aft_valve.
  Add power from heat_node_bef_valve.
  Add power from thermal_storage.
  Add power from heat_pump_heat_cons.

Preparing to plot the energetic flows through the node heat_node_aft_hp.
  Add power from heat_pump_heat_prod.
  Add power from heat_production.
  Add power from district_heat_load.

---

Power flow for the units connected to the node heat_node_bef_valve
Here you update the inputs. Then, press <b>Update</b> to run the optim...
Please have a look to the following case study:

- **Basic example: PV self-consumption**
- **Multi-actor modelling for MILP energy system optimisation: application to collective self consumption**

The code is stored at the Gitlab: OMEGAAlpes Examples in the folder “article_case_study”.

### 2.1 Basic example: PV self-consumption

In this PV self-consumption example, a single-house with roof-integrated photovoltaic panels (PV) is studied. Specifically, the study case is about demand-side management in order to maximize the self-consumption, by shifting two household appliances consumption (clothes washing machine and clothes dryer) and using a water tank for the domestic hot water consumption.

This example is available online at: PV self-consumption Examples
The code is available here: BS2019 PV self-consumption code

### 2.2 Multi-actor modelling for MILP energy system optimisation: application to collective self consumption

The shape of an energy project depends on the available technologies but also on stakeholders’ decisions although most energy-support-decision tools only focus on technical issues.
We aim to propose a multi-actor modelling based on actors’ objectives and constraints and to apply it on the model generation tool for optimization OMEGAAlpes. This modelling aims to help stakeholders to formalize their constraints and objectives and to negotiate them in a multi-actor design process. This modelling has been applied to a simplified collective self-consumption project.

The code is available here: BS2019 multi-actor Modelling code

If needed, have a look to the help on Jupyter notebook
2.2. Multi-actor modelling for MILP energy system optimisation: application to collective self-consumption

Fig. 2: Figure 2: Representation of the collective self-consumption project
What is Jupyter notebook? This is the web application linked to the OMEGAlpes project which has the ability to execute code from the browser, with the results of computations attached to the code which generated them. If you followed the standard install instructions, Jupyter is actually running on your own computer. It’s your computer acting as the server.

3.1 I. Installation through Anaconda

3.1.1 On Linux

To install Anaconda, follow these steps:

1. Download Anaconda 5.2 with Python Distribution Python 3.6 version
2. Save the folder in your “Downloads” (It is according to the language (ex: Telechargements in French))
3. Open your command window and install it by using the following command:
   ```bash
   ~/Downloads/Anaconda3-5.2.0-Linux-x86_64.sh
   ```
4. Agree all the terms and say yes at the end. You can choose or agree to the saving locations for the folder.
5. Close and open a new terminal to make the installation effective
6. To open Jupyter notebook, use the following command:
   ```bash
   jupyter notebook
   ```

3.1.2 On Windows

Just download the package with the latest version of Python 3 and Jupyter will be installed with it. When asked if you want to “add to path”, tick the box even if it’s not recommended. Otherwise, you can add Jupyter to the path -to be able to launch it from the terminal- by . Or if you installed with Anaconda you can open Jupyter directly among your programs (by clicking on the Start button), or from the Anaconda Navigator.
3.2 II. Installation through PIP

3.2.1 On Linux

If you have Python already installed on your computer, you can directly install Jupyter Notebook by using pip on the terminal by using the following command: `pip install`.

Note that the Python versions 2.7 or 3.3 or higher are required. For upgrading your Python version, type “sudo apt-get install python[the version you want]” on the terminal (example: “sudo apt-get install python3”).

You should also know that, by installing Jupyter with this method, you will not install all the Python libraries that are included on the Anaconda folder, you will only install Jupyter Notebook. So if later you need a library that you don’t have installed already, you will have to download and install it separately.

To install, follow these steps:

1. To make sure that you have the latest pip version, type on the terminal the following command line: “pip3 install –upgrade pip” (otherwise try “python -m pip install –upgrade pip” or “python -m pip install -U pip setuptools” or “sudo apt install python-pip”).
2. Then install Jupyter Notebook by typing “pip3 install jupyter” (or “pip install jupyter”) (you might need to reinitialize the computer after this step if the next step doesn’t work).
3. Finally, to open Jupyter, type “jupyter notebook” on the terminal. This will start a server and Jupyter Notebook will pop-up on a browser, on localhost:8888. Leave the server running on the terminal until you’re finished.

3.2.2 On Windows

Run the following commands on the terminal:

1. “py -m install –upgrade pip” to make sure you have the latest version of pip.
2. “py -m pip install jupyter” to install Jupyter Notebook.
3. “py -m notebook” to launch Jupyter from the terminal.

If any of these commands don’t work, try replacing ‘py’ with ‘python’ or ‘python3’.

3.3 III. How to use the Jupyter Notebook

For complete explanations on how to deal with Jupyter Notebook, go to the following link: https://www.codecademy.com/articles/how-to-use-jupyter-notebooks

Steps:

1. Open Jupyter through the command tab with the simple command: jupyter notebook
2. Make sure your project has been saved in one folder on your desktop
3. Then Run one of the case example

To run:

Choose the case example you want to work in.

Then, run cells after cells after completing the inputs.
3.3. III. How to use the Jupyter Notebook

To execute such simulations, you need to:

1. Click on the folder saved in your desktop (local work).
2. In the folder, select all the documents which composed your folder.
3. Click on one of the Simulation notebooks to open an case example of a Microgrid Energy System.

Fig. 1: Figure 1: Presentation of the dashboard
Name of the file you are working on

To run or stop running cells (on after the other)

To save your modifications (when you will have to change the inputs for example)

Fig. 2: Figure 2: How to run the cells