

Densys Summer School

Application Session : Operational
planning of microgrids

01/06/2021

Description

- ◆ Design, simulate and compare the operation of a microgrid with the use of three controllers:
 - A idle controller
 - A rule-based controller
 - An optimization-based controller
- ◆ The microgrid is operated in grid-tied mode. It can purchase / sale energy from / to the grid.
- ◆ It consists of the following components:
 - A PV system
 - A Load
 - A lithium-ion battery

Microgrid simulator

- Simulate an operation planning policy on real data
- New datasets can be easily integrated (timeseries.csv)
- Microgrid topology can be easily changed (data/data.json)
- Results are stored in the results folder (e.g. case1_out.json)
- Plots are automatically generated and can be regenerated from the results.
- [https://github.com/bcornelusse/DENSYSSchool/tree/main/Operationnal planning/microgrid-bench](https://github.com/bcornelusse/DENSYSSchool/tree/main/Operationnal%20planning/microgrid-bench)

Device models

- Load: non_flexible_load + dishwasher + washing_dryer (timeseries)
- Non-steerable generation: pv_production (timeseries)
- Simple battery model: Limited capacity, max [dis]charge rates, [dis]charge efficiencies (.json file)
- Inputs:
 - ◆ “examples/data/application_data.json”
 - ◆ “examples/data/application_timeseries.csv”

Main requirements

- A Python 3.6 distribution
- Main libraries:
 - Numpy
 - Pandas: time-series management
 - Pyomo: mathematical programming
 - Matplotlib
- Install missing packages with « conda » or « pip» if needed.

1) Idle controller (Idle)

- The battery is not in use.
- The photovoltaic energy is consumed directly and the surplus is injected into the grid.
- Energy is imported from the grid when necessary.
- Used as a baseline for comparison with other controllers.

2) Rule-based controller (RBC)

- It stores in the battery as much as possible when the consumption exceeds generation, and discharges it as much as possible otherwise.
- It has no information on the future state of the microgrid. It only imposes actions for the current time-step.
- The goal is the maximization of self-consumption.
- TODO:
 - Write down the logic used for the implementation of the controller
 - Implement your logic in the microgrid simulator

3) Optimization-based controller (OPC)

- Using a mathematical programming formulation (Linear Programming), compute decisions with a rolling look-ahead horizon of 24 hours making a perfect information assumption.
- The goal is the minimization of the operational cost over the rolling horizon of 24 hours.
- TODO
 - Formulate your optimization problem (objective, variables and constraints)
 - Implement your formulation in the microgrid simulator using pyomo.

TODO

- Complete the files:
 - ◆ microgrid/control/rule_based_controller.py
 - ◆ microgrid/control/optimization_based_controller.py
- Use a time step of 1 hour (already set in the .json file)
- Compare the above operation policies (idle, rule-based, optimization-based) on the case provided on different time intervals.
- First consider the case without peak cost ($= 0 \text{ €/kW}$)
- Extend then your formulation (in the OPC) to consider the peak penalty. Test your formulation with a peak cost $= 25 \text{ €/kW}$
- Expected results in term of total cost :
 - ◆ Idle \gg RBC \geq OPC (without peak) , Idle \gg RBC $>$ OPC (with peak)