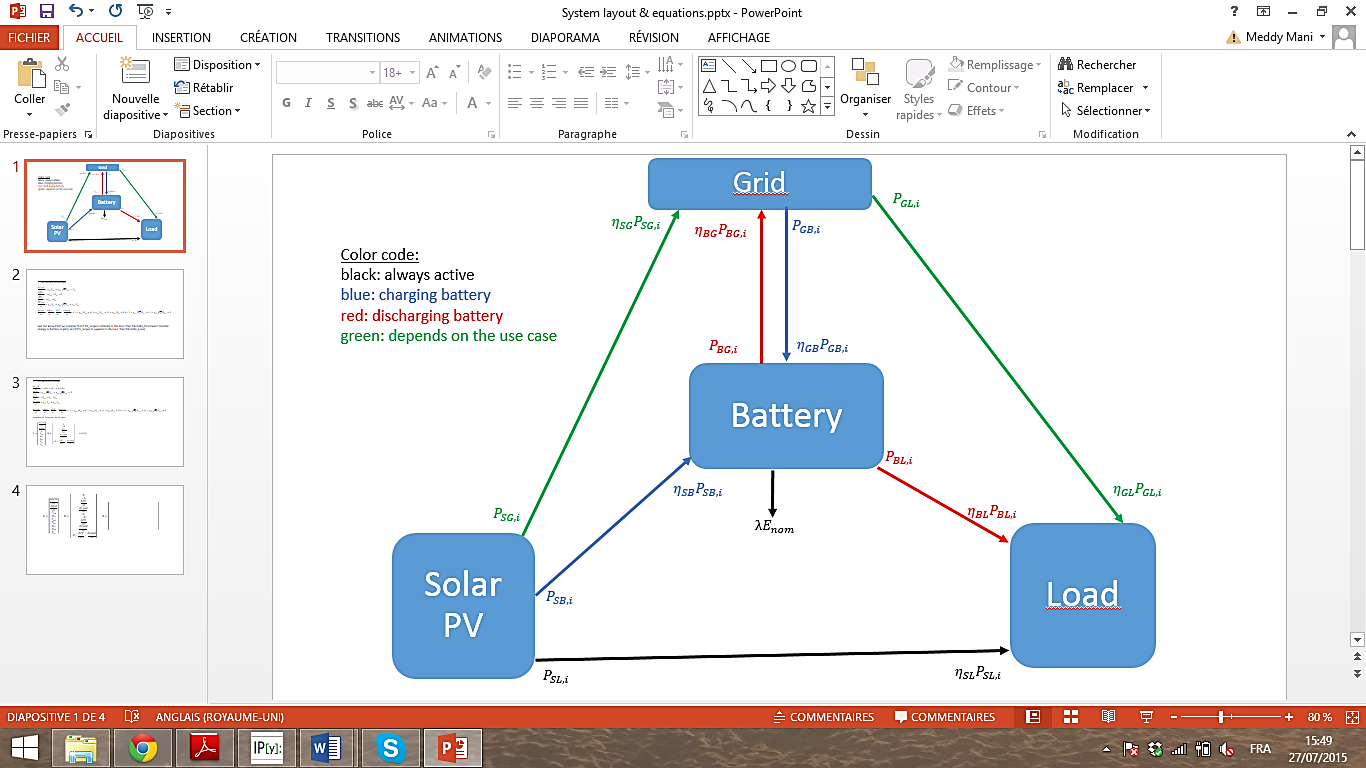
**NRBee**

Our system is connected to the Utility Grid and consists of a Battery, Solar PV panels and the customer/building electricity Load managed by a Controller connected to the Cloud. It is summarised below:

**System Layout**

At time



The NRBee cloud-based real-time operating system is connected to the controller and runs an optimization algorithm every time-steps, using data collected and transmitted by the controller. The optimization algorithm provides a list of instructions (electricity purchase or sale orders) in order to minimize the long-term (to be defined; likely over a 24h period or more) net electricity cost. The optimization has to take into account revenues from electricity trading, as well as battery degradation costs or other revenues from auxiliary services (peak shaving, frequency regulation).

**Controller & Cloud**

To make sure that the grid energy sale or purchase instructions received by the controller are always feasible in real conditions, a regular refresh of these instructions is needed. Every, the controller asks the cloud for an update, according to the following scheme: it first transmits to the cloud the data package collected during the latest period. After receiving the whole data package, the cloud-based optimization algorithm is launched, its output is a list of forecasted instructions over a period, with one instruction at each time-step. These instructions are then transmitted to the controller, which tries, at each time-step, to respect it while granting that the load is always satisfied and that the battery running constraints are always respected.

However, this process is not instantaneous: a non-optimal timeframe has to be compensated when receiving updated instructions. This issue is presented in the following diagram:

This diagram represents a normal situation, where the data transmission and calculation time is short compared to . This time may eventually be much longer in abnormal situations, as we will see later.

actual instruction

updated instruction

Let’s focus on the normal situation, with a normal data transmission and calculation time. This situation can happen in the regular way, every , or in a provoked way, as we will see later. While waiting for updated instructions, the controller keeps applying the old instructions, which, compared to the ones he’s about to receive, aren’t optimal anymore. Different options are possible in order to correct this non-optimal timeframe. To simplify, we set, and.

* 1st option: The cloud considers that the first instruction of the new list is fixed, as the current period is already running, and it makes an optimization on the 23 other time periods. Hence, the 23 other instructions are calculated so that they compensate the non-optimal first instruction. When receiving these 24 instructions, the controller has to ignore the first one, which is anyway the same that the one it’s currently following, and start the next with the second instruction of the list.
* 2nd option: As represented in the previous diagram, the cloud makes an optimization on the 24 periods, and the controller, when receiving it, compensates by its own the previous non-optimal timeframe.

In this situation, the controller compensates the non-optimal instruction he followed during the current before receiving the updated instructions, and starts following these updated instructions on the next .

This option might not be the good one, as it adds intelligence to the controller, and this intelligence is in fact not really “clever”: the controller sets itself a new compensative instruction, without considering any information about battery’s state of charge. Hence it could easily end with reaching extreme battery’s state of charge.

The first option seems to be safer, so let’s extend it to an abnormal case with a connexion failure between the controller and the cloud. The controller will ask for updated instructions, but won’t receive it in a close future. As the last instructions it received covered a period, it will keep applying these instructions, until the end of this period. At this stage, the controller can start the new period with the same instructions again. Indeed, operating conditions are supposed to be more or less cyclical with a one day period, so the old instructions should still temporarily fit with the reality. While the connexion with the cloud is lost, the controller has to store the data as mentioned before, so that it can transmit it all when reestablishing the connexion.

We will precise below what is this “data” transmitted from the controller to the cloud, but we can already talk about one of its components, which is the instructions the controller followed since the last update. Indeed, in case of a connexion failure, the controller will, for an undefined time, follow non-optimal instructions (compared to instructions it would have received with normal update frequency). After reestablishing the connexion, the cloud will thus receive all the data collected since the last update. Same process than before will happen: the cloud will run a new optimization algorithm, with all the instructions followed since the last update being constraints to compensate with the remaining periods in . Referring to the previous diagram, the “non-optimal timeframe” is now much longer than .

When receiving these new instructions after the connexion failure, the controller will have to detect which instruction corresponds to the next , in order to start this with the adequate instruction. This process is represented in the following diagram.

expected reception time

data

actual reception time

instructions

t=0

**CLOUD**

In this diagram are both represented a “normal situation” (in green), where the data transmission and calculation time is short, and another situation, where the time between the update requirement and the reception of new instructions is abnormally long (in red). In both cases, the controller, when receiving the updated instructions, has to ignore some of them: only the first in a normal situation, and the first 4 (arbitrary number) in an abnormal situation. Indeed, in this abnormal situation, the first 4 instructions concern past or running periods. The controller has to detect, amongst the instructions it received, which one corresponds to the following period. When reaching this period, the controller adopts the new list of instructions, starting from, in this case, the fourth instruction in the list.

The data transmitted from the controller to the cloud cover several types:

* Instructions the controller followed since the last update (as mentioned before)
* PV output, grid energy flow, battery energy flow (charge / discharge), and load amount can be calculated in the cloud by simple summation
* Battery parameters evolution: state of charge (SOC), state of health (SOH), temperature, voltage-current characteristics

Most data concern the battery, as it is the only element we have control over, and which charge and discharge profile we adapt to the rest of the system.

The battery parameters data are recorded by a Battery Management System (BMS), and are available to the controller at all times. This last point can allow us to program provoked instructions update requirements in the controller’s “limited” intelligence. For example, we know that reaching minimum or maximum SOC () is not good for the battery as it increases ageing. We can thus set a limit of maximum acceptable time spent with an extreme battery’s state of charge during, or. When this time limit is reached, the controller asks for an update: it sends the last collected data to the cloud, and waits for the reception of new instructions.

**Controller Operation**

Please note that in a first place, which represent power transmission efficiency between two components, are considered as constants. It is a first simple approximation of the reality, which shall be refined later.

Also consider that we are currently assuming the following 2 conditions (to be probably revised in further developments):

* we never curtail PV
* we never shed some load (demand response)

The following use cases show how the controller will react to an electricity sale or purchase instruction. Each instruction transmitted by the cloud-based operating system is an objective defined on, and takes this form:. If the objective is to sell energy to the grid, is positive, and vice-versa.

A sub-instruction, defined on a time-step, is:

The objective, on each period, is to have:

*whereas:* .

Before we take a look at these use cases, we should precise a subtlety about, the battery roundtrip efficiency. Take a look at the following scheme:

**Battery**

We finally have, for the battery energy balance:

Charging battery:

Discharging battery:

This subtlety about leads to another one about and, the battery maximum charging and discharging power. For example, if you want to charge the battery with, you have to compare with, whereas if you want to discharge it by, you have to compare with.

Before getting into the use cases, please find below general equations for the battery energy balance. We have to keep track of the battery’s state of charge after each time-step, as it defines the controller’s actions on the next time-step.

Between time and:

Hence we have, when entering the following time-step:

We also calculate, on each time-step, the grid energy throughput and compare it with the objective:

**Use cases**

1. **Sell to grid:**

**1.A *Without a battery:***

**1.A.1. *PV output is enough to satisfy the load:***

**G**

**S L**

**1.A.2 *PV output isn’t enough to satisfy the load: -> we have to buy from the grid***

**G**

**S L**

***With a battery:*** (load is the priority, so we might have under particular conditions)

* 1. **PV output is enough to satisfy the load:** 
     1. **Remaining PV output is enough to satisfy the grid energy sale objective:**
        1. **Battery can store the leftover PV output that has not been sold to the grid and used to satisfy the load**
           1. **Without exceeding its maximum charging power -> selling target achieved**
           2. **But is limited by its maximum charging power, so PV gives more to grid -> selling target exceeded**
        2. **Battery cannot store ALL remaining PV output -> selling target exceeded**
           1. **And it can be charged up to , the rest of PV output going to the grid**
           2. **And due to charging power limitation, PV gives maximum charging power, and the rest goes to the grid**

**G**

**B**

**S L**

* + 1. **Remaining PV output is not enough to satisfy the grid energy sale objective:** 
       1. **Battery is enough to satisfy the remaining grid energy sale objective**
          1. **And is not limited by its discharging power -> selling target achieved**
          2. **But is limited by its discharging power -> selling target unreached**
       2. **Battery is not enough to satisfy the remaining grid energy sale objective -> selling target unreached**
          1. **Not limited by its discharging power: discharges into grid up to**
          2. **Limited by its discharging power**

**G**

**B**

**S L**

* 1. **PV output isn’t enough to satisfy the load:** 
     1. **Battery is enough to satisfy the remaining load**
        1. **And is not limited by its discharging power**
           1. **Remaining battery is enough to satisfy the grid energy sale objective**

**And is not limited by its discharging power -> selling target achieved**

**But is limited by its discharging power -> selling target unreached**

* + - * 1. **Remaining battery is not enough to satisfy the grid energy sale objective -> selling target unreached**

**Not limited by its discharging power: discharges into grid up to**

**Limited by its discharging power**

**G**

**B**

**S L**

* + - 1. **But is limited by its discharging power -> we have to buy from the grid**
    1. **Battery isn’t enough to satisfy the remaining load -> we have to buy from the grid**
       1. **Not limited by its discharging power: discharges into load up to**
       2. **Limited by its discharging power**

**G**

**B**

**S L**

1. **Buy from grid:**

**2.A *Without a battery:***

**2.A.1 *PV output is enough to satisfy the load: -> we have to sell to the grid***

**G**

**S L**

**2.A.2 *PV output isn’t enough to satisfy the load:***

**G**

**S L**

***With a battery:***

* 1. **PV output is enough to satisfy the load:** 
     1. **Battery can store the remaining PV output**
        1. **Without exceeding its maximum charging power**
           1. **Remaining battery storage capacity is filled by the grid energy purchase objective**

**Without charging power limitation -> buying target achieved**

**But limited by charging power -> buying target unreached**

* + - * 1. **Remaining battery storage capacity cannot fulfil ALL grid energy purchase objective -> buying target unreached**

**So is charged up to**

**And limited by charging power**

**G**

**B**

**S L**

* + - 1. **But charging power limitation, so PV gives the rest to grid -> we have to sell to the grid**
    1. **Battery can’t store the remaining PV output… -> we have to sell to the grid**
       1. **Charged up to with PV output, and the rest goes to the grid**
       2. **Limited by its charging power**

**G**

**B**

**S L**

* 1. **PV output isn’t enough to satisfy the load:**   
     ,  
     1. **Grid energy purchase objective is enough to satisfy the load:**   
        1. **Battery can store the remaining grid energy purchase objective** 
           1. **And is not limited by charging power -> buying target achieved**
           2. **But is limited by charging power -> buying target unreached**
        2. **Battery can’t store the remaining grid energy purchase objective -> buying target unreached**
           1. **So is charged up to**
           2. **And is limited by charging power**

**G**

**B**

**S L**

* + 1. **Grid energy purchase objective isn’t enough to satisfy the load:** 
       1. **Battery is enough to satisfy the remaining load**
          1. **And is not limited by its discharging power -> buying target achieved**
          2. **But is limited by its discharging power -> buying target exceeded**
       2. **Battery isn’t enough to satisfy the remaining load -> buying target exceeded**
          1. **And is not limited by its discharging power**
          2. **And is limited by its discharging power**

**G**

**B**

**S L**

**Net electricity cost**

Net electricity cost can be divided in sub-costs, on different time scales. The following costs are classified in ascending order of the time scale, according to the scheme below:

t=0

i=0  
j=0  
k=0  
l=0

i=1  
j=0  
k=0  
l=0

i=A  
j=1  
k=0  
l=0

i=B  
j=B/A  
k=1  
l=0

i=C  
j=C/A  
k=C/B  
l=1

i=A+1  
j=1  
k=0  
l=0

i=B+A  
j=B/A+1  
k=1  
l=0

i=D  
m=1

Battery time scale cost:

Ageing (simplified version & energy trading only):

(1)

*whereas: is the battery replacement cost  
 is the Life-time Energy Throughput*

This cost is the cycle-life degradation cost. Every time energy goes through the battery, the latter degrades it and thus reduces its lifetime. To record this flow of energy, we only have to consider the incoming energy (that charges the battery), and not the outgoing energy. We also use, for the moment, a simplified expression of LET (Lifetime Energy Throughput), considering that it is constant. In reality, it depends on the battery charging cycle depth, and the whole ageing effect and associated costs need a more complex analysis of the cycles undergone by the battery. To be developped later.

Energy B & S :

(2)

*whereas: are the buying-from-the-grid and selling-to-the-grid tariff*

This is the cost (or revenue if it is negative) which comes from the energy exchanges between the system and the grid. If we buy from the grid, then is negative, and thus represents a positive cost. If we sell to the grid, is positive, and is a negative cost, also called a revenue.

Cost on the economic time scale:

Energy trade:

(3)

*whereas:*

This is the total cost of energy buy & sell operations over a period, during which energy buy & sell tarrifs are constant. This cost is defined on the economic time scale in the sense that it fits with the energy pricing time period imposed by the market regulation.

Demand charge:

Over-capacity cost:

(4)

*whereas:*

This is the cost of taking from the grid, during , an average power superior to what we are allowed to. Our average limit on is . The extra-power is priced .

Overall capacity charge:

(5)  
*whereas:*

This is the total cost coming from the capacity regulation. It contains all the over-capacity costs that we had to pay during, as well as the cost of our assigned capacity.

Overall net electricity cost:

(6)

*whereas:*

This is the overall net electricity cost, defined on. It contains all the costs listed before, summed on a period.

Conditions :

* B is a multiple of D 🡪 is a multiple of
* C is a multiple of B 🡪 is a multiple of
* C is a multiple of A 🡪 is a multiple of

Remarks:

In reality, this overall net electricity cost contains other terms, related to the reserve. If we take the energy reserve of the battery into account, we first have to consider the fact that the depth of discharge (DOD) of the battery as well as its charging and discharging maximum power may be reduced. This will have an impact on the costs and revenues of energy trading. Cycles made with the reserve energy will also impact the cycle-life degradation of the battery. Finally, additional revenues are granted in the operating reserve market, coming from frequency regulation (spinning reserve), balance between demand and generation (regulation reserve)…

For now, in order to simplify our model, reserve aspects are not taken into account.

Fixed charge of access to the grid isn’t covered either. It is a sunk cost (unless a special fixed charge is applied by the Utility for tariff-plan associated with energy trading. Hence, the differential of the fixed charge shall be taken into account in the economic equation), so it doesn’t appear in our net electricity cost. Same for the sunk shelf-life degradation cost: it doesn’t depend on the use me make of the battery, so it doesn’t appear in our net electricity cost.

**Definitions**

*Indexes:*

*Time scales:*

t[s]: time

[s]: battery control time scale

[s]: time interval during which grid energy buy & sell tariffs and are constant

[s]: time period over which capacity charge and are constant

[s]: time period over which capacity assigned to our system is constant

[s]: time interval over which Power Demand is averaged to be compared to

[s]: time interval between each data transmission from the controller to the cloud

[s]: time interval covered by the optimization algorithm forecasted instructions

[s]: time interval between each grid instruction transmitted from the cloud-based software to the controller

*Costs & tariffs:*

[$/kWh]: battery replacement cost

[$/kWh]: buying-from-the-grid tariff

[$/kWh]: selling-to-the-grid tariff

[$/kW]: capacity charge

[$/kW]: over-capacity power tariff

[$]: cycle-life degradation cost on

[$]: cost (or revenue if it is negative) of the energy trading operations between the system and the grid during.

[$]: total cost, on, of energy buy & sell operations

[$]: over-capacity cost on.

[$]: total cost, on, coming from the capacity regulation

[$]: overall net electricity cost on

*Battery specifications:*

[kWh]: nominal battery capacity

[kWh]: lifetime energy throughput

[kW]: battery maximum charging power

[kW]: battery maximum discharging power

[%]: battery maximum state of charge

[%]: battery minimum state of charge

[%]: battery round-trip efficiency

[%/h]: battery hourly self-discharge, expressed as a percentage of per hour.

*System specifications:*

[%]: transmission efficiency from battery to grid

[%]: transmission efficiency from battery to load

[%]: transmission efficiency from grid to battery

[%]: transmission efficiency from grid to load

[%]: transmission efficiency from solar panel to battery

[%]: transmission efficiency from solar panel to grid

[%]: transmission efficiency from solar panel to load

*All the following variables are defined on a timestep:*

[kW]: power delivered by the battery for the grid

[kW]: power delivered by the battery for the load

[kW]: power delivered by the grid for the battery

[kW]: power delivered by the grid for the load

[kW]: power delivered by the solar panel for the battery

[kW]: power delivered by the solar panel for the grid

[kW]: power delivered by the solar panel for the load

[%]: battery state of charge

[kWh]: battery energy throughput [note: positive when battery charges, negative when it discharges]

[kWh]: objective of grid energy throughput

[kWh]: grid energy throughput [note: positive when we sell to the grid, negative when we buy energy]

[kWh]: load energy throughput [note: always an energy inflow, hence a positive number]

[kWh]: solar panel energy throughput [note: always an energy outflow, hence a negative number]

[%]: battery state of charge variation

*The following variable is defined on a timestep:*

[kW]: objective of grid power transmission