Maximisation of self-consumption of PV power

Pol Boudou Perez

October 2019

1 Introduction

2 Problem statement

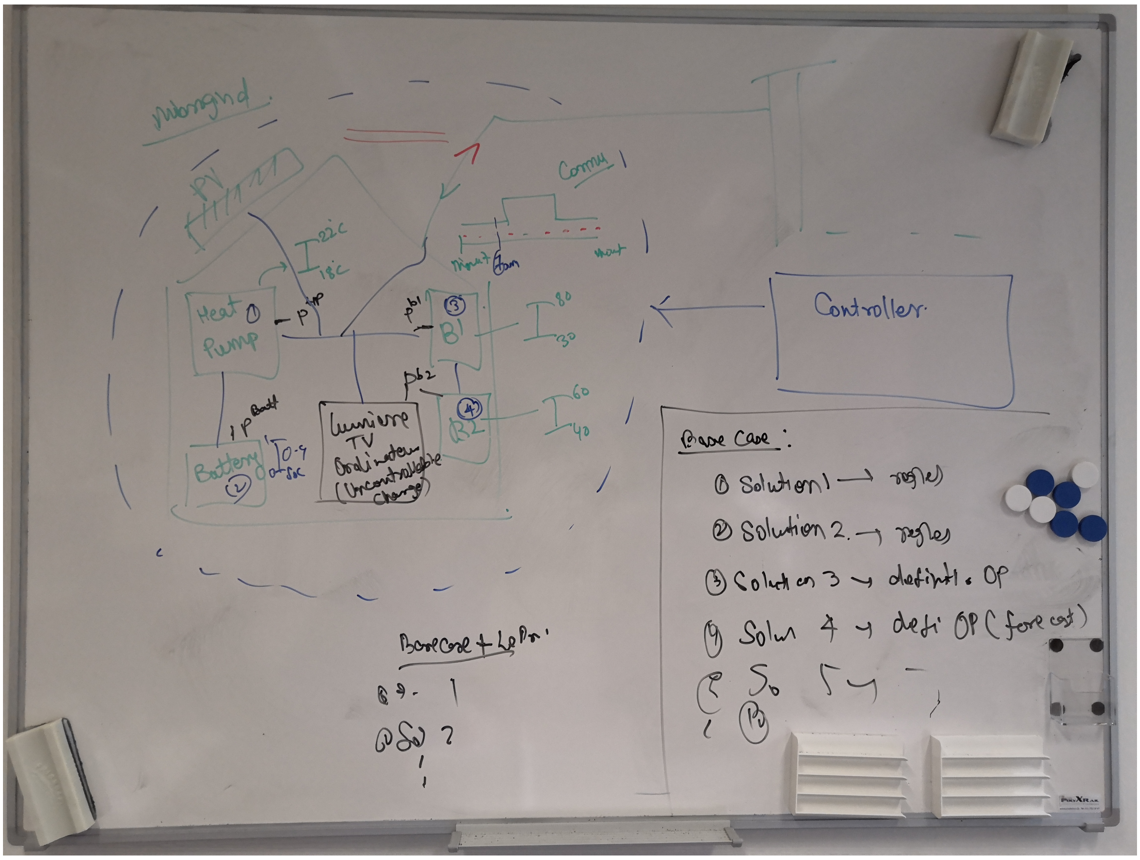

Figure 1: Microgrid sketch (to be changed afterwards)

2.1 System description

The object of the study is a microgrid connected to the distribution grid. In the present case, we consider a simple microgrid that can be seen as a domestic household equipped with a PV system. Because of its power generation capacity, the microgrid can inject excess power into the grid via a single two-ways connection.

As for the energy consumption of the household, this one can be divided in two well defined classes: controllable and non-controllable loads. The latter describes all appliances that are human-switched (TV, lights, coffee machine, etc.). The former makes reference to those flexible loads that are going to be controlled in order to consume whenever there is PV production. For instance, to ensure thermal comfort, the household is equipped with two boilers (B1 and B2) and a heat pump (HP). The storage capacity from those thermo-electric units allows operating them with certain freedom. They can be employed under certain coordination to actively respond to the power system fluctuations. Similarly, an energy storage battery system brings more flexibility to the microgrid.

The objective of the energy management strategy is to control the flexible loads in order to reduce electricity costs by maximising self-consumption of PV power.

2.2 Modelling of the various components and syntax

Control variables are named u, whereas measurement variables will be named p. Capital letters will designate constant parameters. Indexes will refer to specific energy systems. Time value of the variables will be noted with $[h]$, where h designates the current discrete time period, and goes from 0 to H.

PV system

p_{PV}: PV power (kW) (positive variable)

Energy storage system

u_b: target battery power (kW). Positive if battery is charging, negative if discharging.
p_b: actual battery power. Same sign as u_b
x_b: Battery state of charge (kWh)

$\overline{C}_b$: Max. battery capacity (kWh)
$\underline{C}_b$: Min. battery capacity (kWh)
$\overline{P}_b$: Max. charging power of the battery (kW)
$\underline{P}_b$: Max. discharging power of the battery (kW)

Non-controllable loads

p_{nc}: power demanded by non-controllable loads (kW) (positive)

Thermo-electric loads

$u_{B,k}$: target power of electric boiler k (kW) (positive)
$p_{B,k}$: actual power of electric boiler k (kW)

$\overline{P}_{B,k}$: Max. power of electric boiler k (kW)

u_{HP}: target power of electric heat pump (kW)
p_{HP}: actual power of electric heat pump (kW)
$\overline{P}_{HP}$: Max. power of electric heat pump (kW)

Building comfort constraint must be respected and indoor temperature $T_{in}[h]$ must remain within a band around temperature bounds ($\underline{T}_{in}$, $\overline{T}_{in}$).
As for the boilers, flexibility is again achieved by imposing upper and lower limits on hot-water temperature ($\underline{T}_{B,k}$, $\overline{T}_{B,k}$).

3 Control strategy

3.1 Premise

The microgrid automatically balances power supply and demand. We assume any PV power not consumed by the building is absorbed by the grid without the necessity of any control action. Similarly, any power demand from loads that cannot be supplied by the PV system is supplied by the grid.

3.2 Strategy 1

For the moment, such strategy does not look at grid's electricity price.

This being the first and most basic strategy, it does not demand models for thermo-electric units. In a straightforward manner, the charging of those flexible loads is done whenever temperature approaches lower limits. Defining an certain temperature tolerance ϵ_{Tk} for each of the thermo-electric units will ensure (not in a robust manner) that such limits are not attained. Measurements of power inputs p_{B1}, p_{B2} and p_{HP} are therefore not needed in this approach, and temperatures are the only variables used to determine the adequate control action.

Let's define the number of <u>controllable loads</u> as N_c, which equals 3 in the presented system. Because its ability to supply power, the battery is not considered as a controllable *load*. In order to simplify the algorithm syntax, all N_c loads are regrouped in a vector. For instance, $\hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$ regroups the <u>state</u> of such loads, which is a direct measure of their temperature. From a controller device perspective, it is assumed that a pre-processing unit is handling thermal sensors outputs and normalizing them according to the upper and lower temperature limits of each load. In this a way, all $\hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$ elements are bonded between 0 and 1.

3.2.1 Summary of the strategy

The controller is configured by establishing rules determined by all possible system operating scenarios.

For feeding uncontrollable loads p_{nc}, PV power is the prior source, followed by battery storage and finally the main grid.

The variable $p_x = p_{PV} - p_{nc}$ refers to the <u>excess</u> PV power in the microgrid available to feed flexible loads. The dealing of such excess power follows a set of priority rules:

1. If around lower limits, thermo-electric units are prioritized.
2. Battery charging has priority until a certain SOC threshold is reached (80% for example). A binary variable $s_b[h]$ decides if battery has priority over thermal loads ($s_b[h] = 1$) or if thermal loads should be supplied before battery ($s_b[h] = 0$).
3. When loading thermo-electric units, those with the lower energy states $\hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$ are prioritized.

When excess power is negative, charging such controllable loads demands electricity from the main grid.

3.2.2 Algorithm

Algorithm 1: Strategy 1

Executed every time-step h
Inputs: $p_b[h], x_b[h], s_b[h], s_b[h-1], \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h], p_{PV}[h], p_{nc}[h]$
Control variables: $u_b, \hat{\boldsymbol{u}}^{\boldsymbol{c}}$
Initialize: $\mathrm{u}_b \leftarrow 0, \hat{\boldsymbol{u}}^{\boldsymbol{c}} \leftarrow 0$
.
if $x_b[h] > 80\% SOC$ **then**
    $s_b[h] = 0$
end
if $x_b[h] \leq 70\% SOC$ **then**
    $s_b[h] = 1$
else
    $s_b[h] = s_b[h-1]$
end
$p_x = p_{PV}[h] - p_{nc}[h]$
call LowerConstraintHandling($p_x, \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$)
call TreatExcessPower($p_x, p_b[h], x_b[h], s_b[h], \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$)
update Control variables
end

.

Algorithm 2: Lower constraint handling

Control variables: $\hat{\boldsymbol{u}}^{\boldsymbol{c}}[h]$
function *LowerConstraintHandling* ($\mathrm{p}_x, \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$) :
for each energy system k state in $\hat{\boldsymbol{x}}^{\boldsymbol{c}}$*[h]* **do**
    if $x_k^c[h] \in [0; \epsilon_k^T]$ **then**
        if $p_x > 0$ **then**
            $u_k^c \leftarrow min(p_x, \overline{P}_k)$
            if $(p_x > u_k^c)$ **then**
                $p_x = p_x - \overline{P}_k$
            end
        else
            $u_k^c \leftarrow \overline{P}_k$
        end
    end
end

Algorithm 3: Treatment of microgrid's excess power

Control variables: $u_b, \hat{\boldsymbol{u}}^{\mathbf{c}}$
function *TreatExcessPower* $(\mathrm{p}_x, p_b[h], x_b[h], s_b[h], \hat{\boldsymbol{x}}^{\mathbf{c}}[h])$:
$diff = p_x - p_b[h]$
if $p_x > 0$ **then**
  if $(x_b[h] < 80\% SOC)$ **and** $(s_b[h] == 1)$ **then**
    $u_b \leftarrow \min(u_b[h-1] + diff, \overline{P}_b, \frac{1}{dt}(\overline{C}_b - x_b[h]))$
    (unconstrained, power constrained, energy constrained)
    $p_x = p_x - u_b$
    if $p_x > 0$ **then**
      call *PriorityChargingFlexibleLoads(*$p_x, \hat{\boldsymbol{x}}^{\mathbf{c}}[h]$*)*
      break
    end
  else
    call *PriorityChargingFlexibleLoads(*$p_x, \hat{\boldsymbol{x}}^{\mathbf{c}}[h]$*)*
    if returned $p_x > 0$ **then**
      $u_b \leftarrow \min(u_b[h-1] + diff, \overline{P}_b, \frac{1}{dt}(\overline{C}_b - x_b[h]))$
    end
  end
end
else
  $u_b \leftarrow max(u_b[h-1] + diff, \underline{P}_b, \frac{1}{dt}(\overline{C}_b - x_b[h]))$
  (unconstrained, power constrained, energy constrained)
end

Algorithm 4: Controllable loads priority rules

Control variables: $\hat{\boldsymbol{u}}^{\mathbf{c}}$
function *PriorityChargingFlexibleLoads* $(p_x, \hat{\boldsymbol{x}}^{\mathbf{c}}[h])$:
sort $\hat{\boldsymbol{x}}^{\mathbf{c}}[h]$ in increasing order
for k *in* $\hat{\boldsymbol{x}}^{\mathbf{c}}[h]$ **do**
  if $x_k^c \in [1 - \epsilon_k^T; 1]$ **then**
    return(p_x)
  end
  $u_k^c \leftarrow \min(p_x, \overline{P}_k)$
  $p_x = p_x - u_k^c$
  if $p_x \leq 0$ **then**
    return(p_x)
  end
end

3.3 Limitations

Several limitations to the proposed strategy 1 can be found. For instance, the controller action to match battery power to microgrid's excess power is far from perfect. As a broader limitations, the strategy does not make use of models for thermal loads and does not foresee grid's electricity price fluctuations, nor u takes before noon. Let's discuss each of those 3 ways of improvement:

Modelling of thermo-electric units

Firstly, the control of thermal loads could be improved significantly by incorporating models of such loads. This way, changes in temperature could be determined as a function of changes in power injected.

We therefore introduce a simple model for electric boilers that assumes that the water tank has a uniform temperature [**?**].

$$p_{elec} - \dot{m}C_p(T_w - T_{inlet}) + UA_{wh}(T_{amb} - T_w) = C_w \frac{dT_w}{dt} \tag{1}$$

where p_b is the heating capacity of the resistor in the boiler, in Kw, $\dot{m}$ is the hot water flow rate, in $l/hour$, C_p is the thermal capacitance, in $J/(l * °\mathrm{C}$, T_w is the water temperature, in °C, etc. etc.
I will keep updating this and find a model for HP.
The control of such thermo-electric units could be re-designed as follows:

PID control

Secondly, the control of the battery intends to follow the excess power. However, because the control action stays constant between two time intervals and excess power is not, a mismatch happens. In case of excess power decreasing, the battery can be charged using grid's electricity, which would translate into unnecessary costs. In case of excess power increasing, some of it would be absorbed by the grid, missing a storage opportunity. The use of a proportional–integral–derivative (PID) controller on the error $e[h] = u[h-1] - p[h]$ would partially solve that issue.

$$u[h] \leftarrow K_p e[h] + K_i(K_i + e[h]\Delta t) + K_d \frac{e[h] - e[h-1]}{\Delta t}$$

Accounting for grid's electricity price variations

This should be explained way before, when describing the system: Regarding pricing of grid's electricity, several scenarios will be studied. As a first approach, it is assumed that a Time-Of-Use (TOU) pricing option is in place. During a certain period of time around mid-day, price gets relatively higher. It is also assumed that the system is aware of when such price step increase will happen during the day.

In order to take advantage of such price variations, the idea is to "charge" all energy systems of the microgrid prior to the price step increase, during what we will call the "get ready" phase. Afterwards, when price is high, controller configuration will be back to normal. Algorithm 5 illustrates such approach.

It is therefore necessary to define the maximum time t^c needed to charge all flexible elements of the microgrid from their lowest energy state to the highest.

$$t_c = max(t_b^c, t_{B,k}^c, t_{HP}^c)$$

$(t_b^c = \frac{\overline{C}_b - \underline{C}_b}{\overline{P}_b})$

Algorithm 5: Strategy 1 + get ready phase

Executed every time-step h
Inputs: $p_b[h], x_b[h], s_b[h], s_b[h-1], \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h], p_{PV}[h], p_{nc}[h]$
Control variables: $u_b, \hat{\boldsymbol{u}}^{\boldsymbol{c}}$
Initialize: $\mathrm{u}_b \leftarrow 0, \hat{\boldsymbol{u}}^{\boldsymbol{c}} \leftarrow 0$
.
if $h \in [t_1 - t_c; t_1]$ **then**
    $u_b \leftarrow \overline{P}_b$
    $u_k \leftarrow \overline{P}_k$ **for** k in $(B1, B2, HP)$
end
else
    if $x_b[h] > 80\% SOC$ **then**
        $s_b[h] = 0$
    end
    if $x_b[h] \leq 70\% SOC$ **then**
        $s_b[h] = 1$
    else
        $s_b[h] = s_b[h-1]$
    end
    $p_x = p_{PV}[h] - p_{nc}[h]$
    call LowerConstraintHandling($p_x, \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$)
    call TreatExcessPower($p_x, p_b[h], x_b[h], s_b[h], \hat{\boldsymbol{x}}^{\boldsymbol{c}}[h]$)
end
update Control variables
end