Optimal Battery Aging : an Adaptive Weights Dynamic Programming Algorithm

Benjamin Heymann, Pierre Martinon

JOTA manuscript No.
(will be inserted by the editor)

Optimal Battery Aging: an Adaptive Weights Dynamic Programming Algorithm

Benjamin Heymann · Pierre Martinon

Communicated by Lars Grüne.

Received: date / Accepted: date

Abstract We present an algorithm to handle the optimization over a long horizon of an electric microgrid including a battery energy storage system. While the battery is an important and costly component of the microgrid, its aging process is often not taken into account by the Energy Management System, mostly because of modeling and computing challenges. We address the computing aspect by a new approach combining dynamic programming, decomposition and relaxation techniques. We illustrate this 'adaptive weight' method with numerical simulations for a toy microgrid model. Compared to a straightforward resolution by dynamic programming, our algorithm decreases the computing time by more than one order of magnitude, can be parallelized, and allows for online implementations. We believe that this approach can be used for other applications presenting fast and slow variables.

Keywords Control · Aging · Dynamic Programming · Energy Management System

Mathematics Subject Classification (2000) 93A13, 93C15, 90C39, 49L20, 49M27, 49M29

Benjamin Heymann (corresponding author)
CMAP, Inria, Ecole polytechnique, CNRS, Université Paris-Saclay, 91128, Palaiseau, France.
CMM, Universidad de Chile, Santiago, Chile.
E-mail: benjamin.heymann@polytechnique.edu

Pierre Martinon
CMAP, Inria, Ecole polytechnique, CNRS, Université Paris-Saclay, 91128, Palaiseau, France.
E-mail: pierre.martinon@inria.fr

1 Introduction

A microgrid is an electric system that includes some electricity generation units and a battery to store the energy for later use. The energy management system typically enforces the optimum balance between the different energy sources, in order to minimize some global operating cost. This leads to solving an optimization problem with a short time step (say 15 minutes) over a very long time-frame (such as 10 years). The long time-frame means that the aging of the battery becomes non-negligible, both due to the gradual loss of efficiency and the high cost of the battery. However, battery aging is hard to take into account in the optimization, leading to sub-optimal policies. Indeed, there is some trade-off between the short and long term costs: a policy making heavy use of the battery may be better in terms of daily operating costs, but causes a faster aging of the battery, thus leading to a higher global cost. The two main difficulties are the profusion and complexity of aging models, as well as the numerical cost of solving a long horizon optimization problem.

The modeling and optimization of the battery aging is an active interdisciplinary field of research. In [1], Haessig et al. propose a simulation with an aging model to perform a cost analysis, but without optimization. In [2], Riffonneau et al. use a discrete time dynamic programming approach to solve an optimal power flow problem. The battery aging is proportional to the discharge of the battery and linearly decreases the capacity of the battery. In [3], Palma et al. integrate the battery aging in a rolling horizon strategy model. The aging is taken into account using a working zones approach as proposed in [4], with the battery cost as the penalty parameter.

To our knowledge there are basically three approaches in the literature to handle battery aging in an optimization model:

- Some constraints on the control and state variables (for instance, to avoid extreme State of Charge values): The main difficulty consists in choosing the constraints, which can be too conservative or not enough. Also, the aging process is not directly taken into account in the optimization program.
- A penalization of the aging: This requires choosing an aging model and a penalty parameter.
- An aging constraint: This requires choosing an aging model and the aging level.

In the present work, we introduce a variation on the penalty approach. We focus on an efficient way to solve the optimization problem over a long horizon with a short time step, as we would like the optimization to be fast enough for an actual *online* implementation. We propose a new 'adaptive weight' method based on dynamic programming and a coupling of decomposition and relaxation techniques. The approach consists in two parts, the first being an *offline* optimization that first solves a family of short term problems associated to a discretized set of battery ages and aging penalty weights, and then store the resulting costs and age increments. The second part is an *online* optimization that uses this data to compute the daily optimal policy. This decomposition makes both the *online* and *offline* problems one-dimensional in state, allowing to solve them efficiently with dynamic programming.

We illustrate the adaptive weight algorithm (AWA) on a simple microgrid model. Section 2 recalls the optimal control problem for the energy management of the microgrid. Section 3 introduces the adaptive weight algorithm to solve the offline and online optimizations. In Section 4, we compare the numerical results of AWA with a brute force solving of the original problem by dynamic programming.

2 Motivating Application

2.1 Microgrid

We consider a simpler version of the microgrid problem studied in [3,5,6]. The electricity is produced by some non dispatchable units (solar panels) and a dispatchable unit (diesel generator). At anytime, there is an instantaneous (uncontrollable) demand for electricity. A battery storage system can either store excess energy or provide part (or all) of the demand. The battery is not a perfect storage and it has an efficiency strictly lower than 1. The aim is to satisfy the demand at all times while minimizing the overall cost associated with the dispatchable unit (diesel).

We assume the power load (demand) $P_L(t)$ and the solar production $P_S(t)$ to be given and T-periodic, where T is the duration of a day. The diesel generator can produce a power $u \in$

$[u_{min}, u_{max}]$. We assume the resulting cost to follow a quadratic relation:

$$\ell(u) = \beta u^2, \quad \beta > 0. \tag{1}$$

At all times, the power equilibrium is satisfied and allows one to express the battery power P_b (with the convention $P_b > 0$ for discharge)

$$P_b(u,t) = -u - P_s(t) + P_l(t). \tag{2}$$

The battery properties can be summarized by three parameters: total capacity C, charge efficiency ρ_i and discharge efficiency ρ_o. The state of charge (SOC) of the battery $c \in [0,1]$ is defined as the quantity of energy stored in the battery divided by the capacity, and follows the dynamics

$$\dot{c}(t) = F_c(u,t) = \frac{1}{C}(\rho_i (P_b(u,t))^- - (P_b(u,t))^+/\rho_o). \tag{3}$$

In the rest of the following, we impose c to have the same value at the beginning and at the end of the day. This is often done in practice to avoid the 'end of the world effect': with no final constraint, the optimization would typically empty the battery at the end of the day. We point out that the methodology presented thereafter also applies without this periodicity condition.

2.2 Battery Aging

We define the aging of the battery over time with a scalar $a \in [0,1]$, with the convention $a = 0$ for a brand new battery, and $a = 1$ when the battery is dead. We choose a severity factor model, adapted from [7], in which the aging evolves depending on the state of charge and charging power:

$$\dot{a}(t) = F_a(c,u,t) = \frac{(-4c^2+5)}{5}\frac{(P_b(u,t))^-}{K}, \quad K > 0. \tag{4}$$

The main idea behind this model is that charging a nearly full battery will damage it more than at a low state of charge. We model the aging impact on the performance by a simple linear decrease of the charge efficiency ratio

$$\rho_i(a) = (1-a)\rho_i^0$$

where ρ_i^0 is the value for a brand new battery. The state of charge dynamics (3) thus becomes:

$$F_c(a,u,t) = \frac{1}{C}(\rho_i(a)(P_b(u,t))^- - (P_b(u,t))^+/\rho_o). \tag{5}$$

2.3 The Long Term Problem

We want to minimize the operating cost $\int \ell(u(t))dt$ over a long horizon NT, and we add a penalty function ψ to account for the value of the battery at the end of the time horizon. The optimization problem thus writes

$$\inf_u \int_{t_0}^{NT} \ell(u(t))\mathrm{d}t + \psi(a_{NT}), \tag{6}$$

subject to (5) and (4). In what follows, we denote by $V(a_0,t_0)$ the value of this problem for an initial time t_0 and an initial condition $a(t_0)=a_0$. Note that (6) has a 2-dimensional state: state of charge and age. Combined with the long time horizon and short time step, this makes the use of global optimization such as dynamic programming rather costly.

3 The Adaptive Weight Algorithm (AWA)

3.1 Key Ideas

A first observation is that we can use a bi-level formulation to decompose the problem, using an aging constraint at the short term scale. We consider an age increment δ, and introduce the family of constraints

$$a(T) \le a^i + \delta, \tag{7}$$

and the corresponding 'micro' problems (P^μ) over one period T:

$$(P^\mu) \quad \inf_u \int_0^T \ell(u(t))\mathrm{d}t, \tag{8}$$

subject to (4), (5) and (7). We denote by $V^{\mu}(a^i, \delta)$ the associated short term value. The long term value V can be computed via dynamic programming according to

$$V(a^i, kT) = \inf_{\delta} V^{\mu}(a^i, \delta) + V(a^i + \delta, (k+1)T). \tag{9}$$

Another key point is that aging is a slow process, therefore we can neglect the age variation over each micro problem. This reduces the number of state variables to 1 and allows for fast solving. However we still need to keep track of the age increment δ over the long time-frame.

We tackle this issue with a relaxation trick: given some weight α, we introduce the relaxed micro problems (P_r^{μ})

$$(P_r^{\mu}) \quad \inf_{u} \int_0^T [\ell(u(t)) + \alpha F_a(c(t), u(t), t)] \mathrm{d}t, \tag{10}$$

and denote $V_r^{\mu}(a^i, \alpha)$ the associated relaxed short term value.

Now we present the link between the original micro problem and the relaxed micro problem, introducing some notations. Let us fix an initial age a^i. For a control u over a period T, we denote by $\Delta(a^i, u)$ the associated age increment and $\mathcal{L}(a^i, u)$ the operational cost. Then for a given weight α, we note $\Delta^r(a^i, \alpha)$ the age increment cost and $\mathcal{L}^r(a^i, \alpha_k)$ the generator-associated in the relaxed micro problem (10). Let us consider an optimal control $\bar{u}$ for the micro problem (8), with the associated age increment $\Delta(a^i, \bar{u})$. Let us assume there exists a weight $\bar{\alpha}$ leading to the same age increment in the relaxed micro problem, i.e. $\Delta^r(a^i, \bar{\alpha}) = \Delta(a^i, \bar{u})$. Then the generator-associated cost in the relaxed micro problem is actually the same as the cost of the original micro problem:

$$V^{\mu}(a^i, \Delta(a^i, u)) = V_r^{\mu}(a^i, \alpha) - \alpha \Delta(a^i, u) = \mathcal{L}^r(a^i, \alpha). \tag{11}$$

This can be proven by noting that an optimal control for one problem is admissible for the other. We refer the reader to [8] for more technical details. Following this idea, our proposal consists in reducing the long term problem (6) to

$$\inf_{\alpha} \sum_{k} \mathcal{L}^r(a_k, \alpha_k) + \psi(a_{NT}), \tag{12}$$

with (a_k) the sequence of ages defined by $a_{k+1} = a_k + \Delta^r(a_k, \alpha_k)$. We can see (3.2) as a change of variable in (9): instead of optimizing over the one-period age increments, we optimize over the corresponding weights. Such a change of variable can be done safely in the absence of duality jumps. To our knowledge, there is no general condition for the absence of duality jumps, and we refer readers interested in a technical discussion on this decomposition (and the generality of the approach) to [8].

Unlike (9), the new formulation (3.2) does not need to keep track of the age increment δ within the day. Therefore, the relaxed micro problems have only the state of charge as state variable, and can be solved efficiently by dynamic programming. The daily age variation can be computed simply by integrating its dynamics with the optimal control of the relaxed micro problems.

3.2 Algorithm

The adaptive weight algorithm thus solves an approximation of (6) by combining (a) slow/fast state variable separation, (b) constraint relaxation and (c) dynamic programming. The full resolution scheme is composed of an offline part (Algorithm 1) and an online part (Algorithm 2).

The **offline** algorithm computes and saves the values of the generator-associated cost $\mathcal{L}^r$ and age increment Δ^r for a family of relaxed micro problems, over a discretized set of the initial age a^i and weight α. For each relaxed micro problem, the dynamics use a constant age $a(t) = a^i, \forall t \in [0, T]$. Note that all problems are independent, so this offline step could be massively parallelized.

The **online** algorithm uses these stored values to solve the long term problem by dynamic programming. Here the age a is the state and the weight α acts as the control. This part uses an interpolation of V, noted $\hat{V}$.

```
Result: $\Delta^r$ , $\mathcal{L}^r$
for $(a^i, \alpha)$ in the discretization set do
    Compute an optimal control $u$ for the relaxed micro problem with parameters $(a^i, \alpha)$ ;
    $\Delta^r_{a^i,\alpha} \leftarrow \Delta(u)$ and $\mathcal{L}^r_{a^i,\alpha} \leftarrow \mathcal{L}(u)$ ;
end
```

Algorithm 1: AWA algorithm: OFFLINE component

```
Data: initial age $a^i$, day index $k_0$, $\Delta^r$ and $\mathcal{L}^r$ (output of offline part)
Result: optimal control for the next day $u^*$
$V_{a,N} \leftarrow \psi(a)$ for all $a$ ;
for $k \leftarrow N-1$ to $k_0+1$ do
    for $a$ in the discretization set do
        $V_{a,k} \leftarrow \min_\alpha \{\mathcal{L}^r_{\alpha,a} + \hat{V}(a + \Delta^r_{\alpha,a}, k+1)\}$ ;
    end
end
$\alpha^* \leftarrow \mathrm{argmin}_\alpha \{\mathcal{L}^r_{\alpha,a} - \alpha\Delta^r_{\alpha,a^i} + \hat{V}(a^i + \Delta^r_{\alpha,a^i}, k_0+1)\}$;
Compute an optimal control $u^*$ for the relaxed micro problem with parameters $(\alpha^*, a^i)$ ;
```

Algorithm 2: AWA Algorithm: ONLINE component

4 Numerical Simulations

4.1 Parameters and Implementation

We solve the problem with a time horizon of $N = 600$ days. The dynamic programming in dimension 1 and 2 both use a time step of 15 minutes, while the online part of AWA uses a time step of 1 day. For the state, control and weight we take the ranges $SOC \in [0.1, 1]$, $a \in [0, 500]$ (we rescale from $[0, 1]$ to $[0, 1000]$), $u \in [-5, 15]$ and $\alpha \in [0, 250] \bigcup \{10^9\}$. We test three levels of discretization: coarse, medium and fine, detailed in Table 1. The range for α was set by simple trial and error, since simulations only take a few seconds for the coarse discretization. We also set the parameters $\beta = 0.5$, $\rho_i = 0.95$, $\rho_o = 0.95$, $K = 12.5$, and the functions for the solar power $P_S(t) = \max(0, 13 - 0.3(4t - 48)^2)$ and power load $P_L(t) = 3 + 3e^{-0.1(4t-32)^2} + 12e^{-0.03(4t-74)^2}$. We also set $\psi(a) = +\infty$ for $a > 0.5$.

Both the brute-force approach and the micro problems of the AWA method are solved via dynamic programming with the optimal control toolbox BocopHJB[1] ([9], [10]). The AWA algorithm itself is implemented in the R scripting language, keeping in mind that most of the

[1] http://www.bocop.org

Table 1 Discretizations for the brute force and AWA method.

	Brute force			AWA (offline)		AWA (online)	
Grid	SOC	Age	Control	SOC	Control	Age	Weight
coarse	25	25	20	25	20	25	10
medium	50	50	40	50	40	50	25
fine	100	100	100	100	100	100	50

computational cost lies in solving the micro problems. All simulations are run on a standard laptop.

4.2 Results

We perform two batches of simulations, for $a_0 = 0$ and $a_0 = 150$. The objective values, recomputed along the optimal trajectories, as well as the CPU times (in seconds) are summarized in Tables 2 and 3. The CPU time ratio is indicated, as well as the objective 'gap', i.e the relative error in objective compared to the brute force solution. As expected, the online part of the AWA method is extremely fast. It should be noted that the offline part is highly parallelizable, since each micro problem is independent from the others.

The increase in CPU time is consistent with the increased number in discretization steps.

Fig. 1 shows the age evolution for both methods with the fine discretization, with the limit $a \leq 500$ touched for $a_0 = 150$. The values of the weights α are displayed on Fig. 2, and we see that the optimal weight is indeed not constant over time. Observe that the weights are higher when we start with an older battery, and the weight decreases for the $a_0 = 0$ trajectory, as the age limit becomes less stringent.

Finally, Fig. 3 illustrates the change of variable of the AWA: instead of optimizing over the one-period age increments Δa, we optimize over the corresponding weights α. The graphs indicate that Δa as a function of α is non-increasing, as expected. We observe that the age increments are lower for an old battery, indicating that the older and less efficient the battery becomes, the less it is used. On a side note, these graphs could be used to estimate some upper bound for the error in the method in case of duality jumps (see [8]).

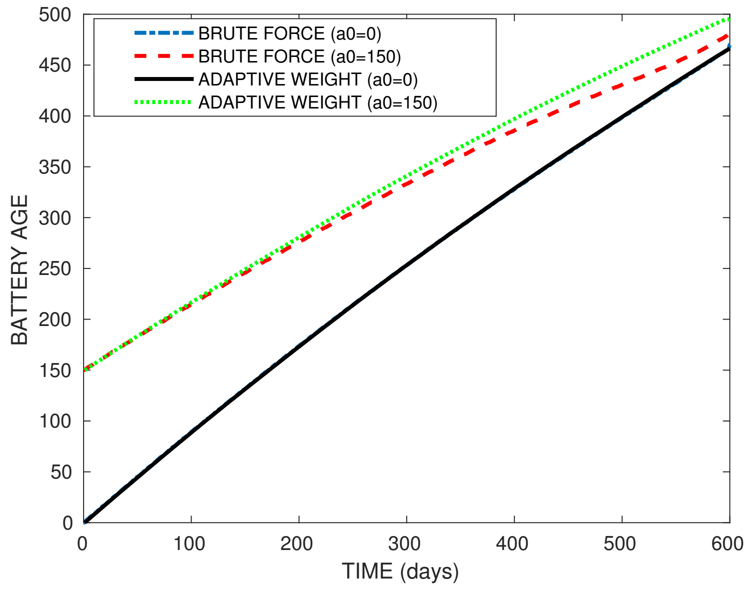

Fig. 1 Age evolution for both methods - initial age: 0 and 150 - fine discretization.

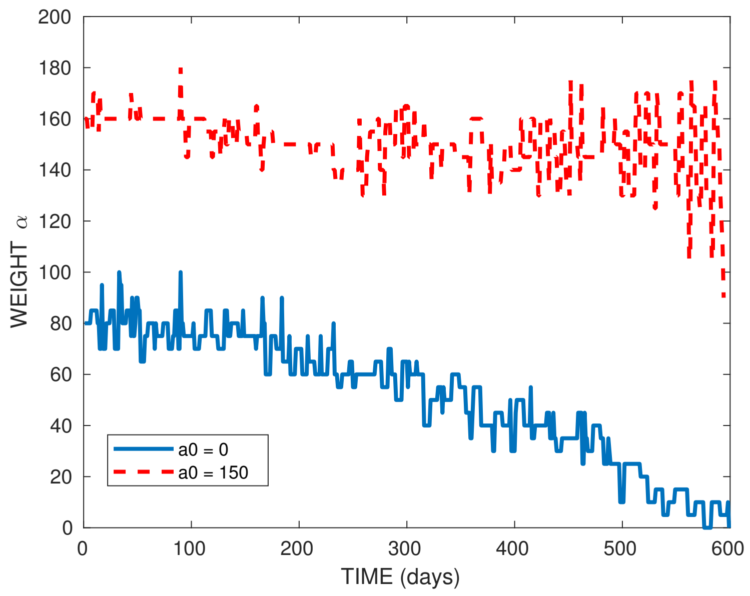

Fig. 2 Weight α evolution - initial age: 0 and 150 - fine discretization.

4.3 Initial age $a_0 = 0$

For the fine discretization, the relative error in objective for AWA is only 0.5%, with a factor 15 decrease of CPU time. We observe on Fig. 1 that the age profiles are almost identical, which confirms that AWA computes an accurate approximation of the solution.

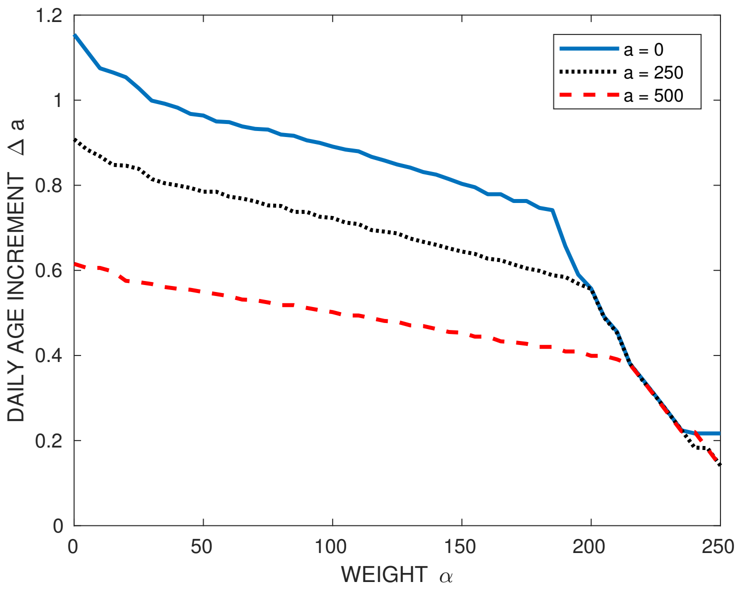

Fig. 3 Daily age increments as function of weight α.

Table 2 Objective values and CPU times (for AWA: offline+online) - initial age $a_0 = 0$

	Objective			CPU time (s)		
Grid	BF	AWA	gap	BF	AWA (offline+online)	BF/AWA
coarse	121408	127271	4.8%	343	7.8+0.11	43.4
medium	111326	107394	3.5%	2362	94+0.33	25.0
fine	106179	106670	0.5%	22063	1432+0.87	15.4

4.4 Initial age $a_0 = 150$

We take a larger initial age in order to illustrate the effect of the state constraint $a \leq 500$. An infeasible state is in practice penalized by a very large value function. Due to the interpolations performed at each time step, some numerical diffusion of these large values occurs. This phenomenon tends to deform optimal trajectories that would come close to the boundary, as seen on Fig. 1 for the brute force (BF) method. The online part of AWA method uses a much larger time step (1 day versus 15 minute), which strongly reduces the diffusion and allows to compute a trajectory close to the boundary. Concerning Fig. 2, at the five last days the age reaches the limit $a \leq 500$, and the corresponding weight α takes the fixed value 10^9, which is not drawn on the graph.

Table 3 Objective values and CPU times (for AWA: offline+online) - initial age $a_0 = 150$.

	Objective			CPU time (s)		
Grid	BF	AWA	gap	BF	AWA	BF/AWA
coarse	156474	165655	5.9%	330	8.0+0.12	40.6
medium	136942	125234	8.5%	2371	90+0.29	26.3
fine	126980	122520	3.5%	22289	1477+0.78	15.1

5 Perspectives and Open Problems

We have identified several follow-up for this work. Increasing the degree of sophistication of the model raises modeling and mathematical questions. Possible extensions include:

- **Replacement of the Battery:** In many applications, it is possible to buy spare parts to replace worn components. Hence we may want to introduce the possibility to buy a replacement battery in the optimization problem, probably with an impulse control.
- **Periodicity:** We can include seasonality by having different kinds of periods. For instance, we could model winter and summer days. In this case, one needs to perform an offline preprocessing for each kind of day. Likewise we could have week days and weekend, or even sunny and cloudy days (using a Markov Chains).
- **Short Term Randomness:** With Markovian dynamics and final constraints on the age increment we may want to see whether the same arguments apply on the daily average age increment.
- **Infinite Horizon:** If we add a discount rate, the arguments should apply for infinite horizon. Then one needs to replace the macro dynamic programing algorithm by either a policy iteration algorithm or a value iteration algorithm.

We also hope to see this approach tried on other application domains.

6 Conclusions

We present the adaptive weight dynamic programming algorithm (AWA), a decomposition technique for problems with periodic data, a fast state and a slow state. We illustrate the method on a toy micro grid problem, and observe that the trajectories and value functions from AWA are close to those from a brute-force approach (straightforward dynamic programming). The computing times are significantly smaller, the online part taking less than 1 second. Moreover, the offline part of the algorithm can be parallelized. We think this would make AWA suitable for a real-time optimization of the microgrid, as well as other dynamical systems with a slow state.

Acknowledgements The first author BH acknowledges support from the Siebel Scholar Program. The authors wish to thank Frédéric Bonnans for his advice and comments on the paper. Thanks also to Hanyun Huang for her English suggestions.

References

1. Haessig, P., Multon, B., Ben Ahmed, H., Lascaud, S., Jamy, L.: Aging-aware NaS battery model in a stochastic wind-storage simulation framework. In: PowerTech (POWERTECH), 2013 IEEE Grenoble, pp. 1–6. IEEE (2013)
2. Riffonneau, Y., Bacha, S., Barruel, F., Ploix, S.: Optimal power flow management for grid connected PV systems with batteries. Sustainable Energy, IEEE Transactions on **2**(3), 309–320
3. Palma-Behnke, R., Benavides, C., Lanas, F., Severino, B., Reyes, L., Llanos, J., Sáez, D.: A microgrid energy management system based on the rolling horizon strategy. Smart Grid, IEEE Transactions on **4**(2), 996–1006 (2013)
4. Guasch, D., Silvestre, S.: Dynamic battery model for photovoltaic applications. Progress in Photovoltaics: Research and applications **11**(3), 193–206 (2003)
5. Heymann, B., Bonnans, J.F., Martinon, P., Silva, F., Lanas, F., Jimenez, G.: Continuous Optimal Control Approaches to Microgrid Energy Management. Energy Systems (2016)
6. Heymann, B., Bonnans, J.F., Silva, F., Jimenez, G.: A Stochastic Continuous Time Model for Microgrid Energy Management. In: ECC2016. Aalborg, Denmark (2016)
7. V. Svoboda et al: Operating conditions of batteries in off-grid renewable energy systems. Solar Energy **81**(11), 1409–1425 (2007).
8. Heymann, B.: Mathematical contributions for the optimization and regulation of electricity production. PhD Thesis (2016). URL https://hal.archives-ouvertes.fr/tel-01416404
9. Bonnans, J.F., Giorgi, D., Heymann, B., Martinon, P., Tissot, O.: BocopHJB 1.0.1 – User Guide. Technical Report RT-0467, INRIA (2015). URL https://hal.inria.fr/hal-01192610
10. Bonnans, J.F., Martinon, P., Giorgi, D., Grélard, V., Heymann, B., Jinyan, L., Maindrault, S., Tissot, O.: Bocop - a collection of examples. Tech. rep. (2016). URL http://bocop.saclay.inria.fr/