Research Work

Predictive Controller for Load Management on Isolated Photovoltaic Power System

Submitted by
Eng. Daniel Eduardo Zamudio Espinosa

Directed by
Eng. Diego Alejandro Patiño Phd

School of Engineering
Electronics Department
2018-1

Contents

List of Figures

1 Introduction

1 Problem Description and Context

According to Colombian Mining and Energy Ministry, for the year of 2015[2], approximately $460,000$ families in Colombia do not have access to electric energy. That means 3.04% of the population is not covered by the national energy network. Despite it is low percentage, the country is behind in electric energy coverage, in comparison with other regional countries such as Brazil, Costa Rica and Chile, which achieve a network coverage above 99%[3] . On the other hand, rural population suffers a 12.17% of deficit against a 0.28% of urban population, which demonstrate that rural population are the ones been affected by the coverage deficit.

In order to counteract this situation, Cundinamarca's region in association with Pontificia Universidad Javeriana, started a project whose main objective is the electrification of rural schools, using a photovoltaic system designed by a research team at the university (Figure 1). In figure 2 a block diagram of the system is shown. A DC-DC converter is used to maintain a positive and negative DC voltage rail and to charge the battery bank using the power coming from the solar panels. The DC-AC converter produce a 120 V_{RMS}, 60 Hz sine voltage wave used by the loads inside the schools.

Figure 1: PV system on preliminary tests

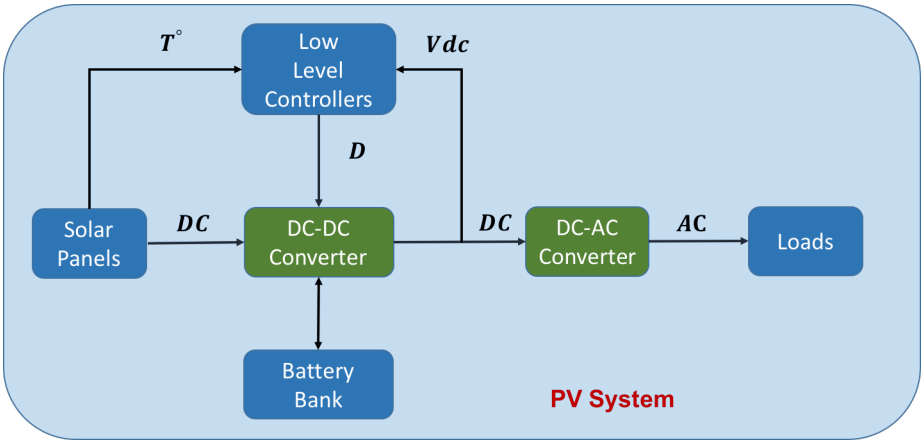

Figure 2: PV system block diagram

Since the power system is isolated and depends exclusively of solar energy and due to the unpredictable nature of the solar irradiation and power consumption, the charge of the batteries could be totally consumed, therefore, a control system that manages the power consumption inside the school is needed, mainly to maintain a minimum charge[1] in the battery bank, so the lifetime of the bank is not compromised[2] and energy is reserved for priority loads. It is important to mention that, the control system must be easily implementable in a low-cost micro-controller, which means, the algorithms must not required big computation times and be flexible enough so it can be applied when other conditions are given (different schools for example). This requirements were taken into account for the design of the management system.

1 Objectives

To solve the problem exposed in the previous section the following general and specific objectives were defined:

- General: Design a predictive control system, implementable in an embedded system, that manage the enable and disable of the loads.
- Construct and validate a model of the power system that allow the estimation of the input and output power.
- Implement an algorithm to estimate the SoC of the batteries whose RMS[3] error is below a $5\ \%$.
- Design a predictive control system that manage the enable and disable of the loads inside the school, so a minimum SoC is guarantee.
- Implement the controller in an embedded system, and validates its functioning using an emulation system so the minimum SoC of the battery is maintain.

1 Problem Solution and Control Architecture

In figure 3 the general control architecture is shown. The signals available from the PV system (Process), are the voltage and current of the battery bank ($V_b(k)$, $I_b(k)$), the DC input voltage and current (V_{in}, I_{in}) and the output voltage and current (V_{out}, I_{in}). Using both input and output signals power generated (P_{in}) and load consumption (Q_l) are calculated to be used in the management system.

The Estimator block used the battery voltage and current, to estimate the SoC of the battery bank, blocks P_{inest} and Q_{lest} used past data of P_{in} and Q_l respectively, to estimate the next N samples of power. The control signal $U_{load}(k)$, is an n_{load}[4] logic signal which enable or disable the loads inside the school. Finally the controller is a simplified MPC that chooses, based on the actual SoC and the power predictions, which loads should be disable in order to maintain a minimum SoC.

It should be noted that a current sensor to calculate the individual consumption of each load could be used, nevertheless, costs of implementation can be significantly increased due to the high costs of AC current sensor as well as the loss of flexibility in the implementation of the control.

The book is organized as follows: In chapter 2 the battery model is explained, and the SoC estimator is derived based on the battery model. In chapter 3 the PV system model is explained, as well as the algorithm to estimate the horizon of power generated and power consumed. With all the auxiliary signals the MPC controller is derived. Finally the hardware in the loop environment is described and the control results are shown.

[1] Also known as state of charge [%], from now on written as SoC

[2] Lead-acid batteries lifetime is inversely proportional to the depth of discharge

[3] root mean square

[4] number of groups of loads

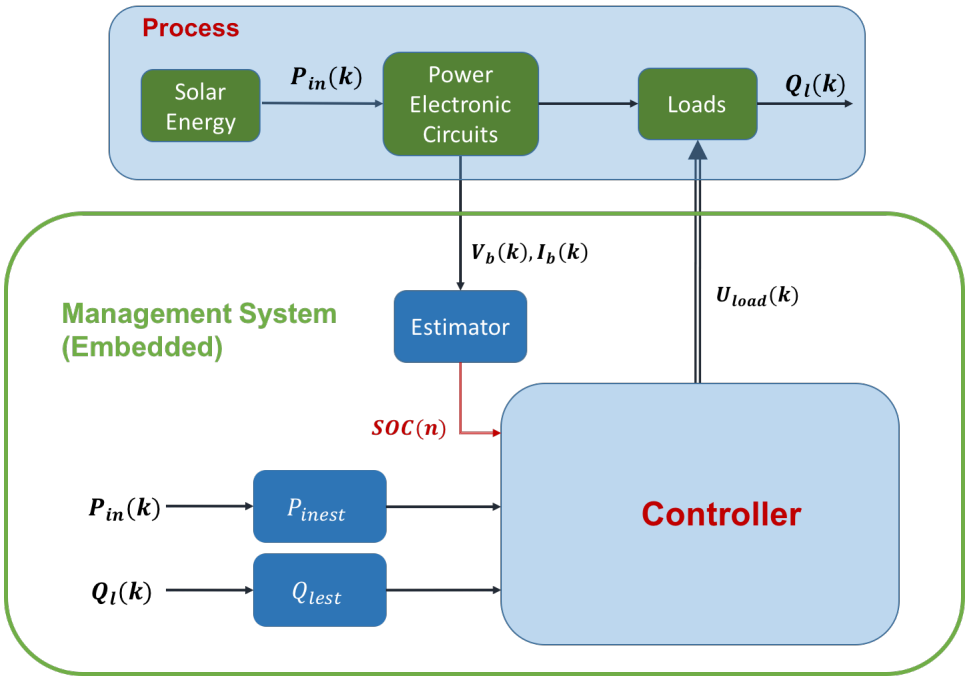

Figure 3: General control architecture

2 Battery Bank Modelling and State of Charge Estimation

This chapter describes the mathematical model used for the battery bank and the SoC estimation algorithm. An equivalent circuit was chosen ([4], [5]), since it enables the use of continuous difference equation; therefore a Kalman filter can be apply to estimate the SoC[6]. Other types of models like switched and hybrid models do not permit this, due to numerical discontinuities that occur when a transition between different states is made.

2 Battery equivalent circuit

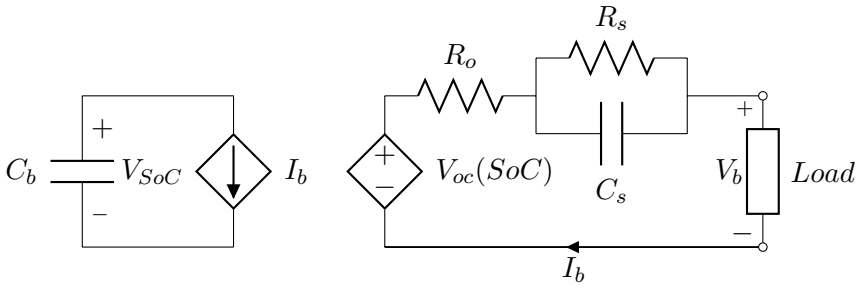

Figure 4: Battery equivalent circuit model

An equivalent circuit model of a battery is taken from figure 4. The capacitor C_b, models the rated capacity of the battery, the resistor R_o, represents the battery opposition to energy flow and the RC network (R_s and C_s) models the dynamic response of the battery. Finally the voltage controlled source $V_{oc}(SoC)$, models the nonlinear relationship between the state of charge, and the open-circuit voltage. Applying voltage, current and element laws for the circuit shown in figure 4, the following equations were obtain:

$$V_b = V_{oc}(SoC) - I_b \cdot R_o - V_s \tag{1}$$

$$I_b = \frac{V_s}{R_s} + Cs \cdot \dot{V_s} \tag{2}$$

$$I_b = -C_b \cdot \dot{SoC} \tag{3}$$

Rearranging terms:

$$\dot{V_s} = \frac{V_s}{R_s} + C_s \cdot R_s \tag{4}$$

$$\dot{SoC} = -\frac{I_b}{C_b} \tag{5}$$

In order to discretize the model, Euler approximation is used:

$$\frac{df}{dt} = \frac{f(k+1) - f(k)}{\Delta t} = \dot{f}(t) \tag{6}$$

$$f(k+1) = \dot{f}(t) \cdot T_s + f(k) \tag{7}$$

Using equation 7 to discretize equations 4 and 5 the following is obtain:

$$V_s(k+1) = \frac{T_s}{C_s} \cdot I_b(k) + \left(1 - \frac{T_s}{R_s \cdot C_s}\right) \cdot V_s(k) \tag{8}$$

$$SoC(k+1) = SoC(k) - \frac{T_s}{Q_T \cdot 36} \cdot I_b(k) \tag{9}$$

Notice that the capacitance C_b, is replaced in equation 9, in terms of the rated capacity of the battery (Q_T $[Ah]$), so the following state space is defined:

$$\begin{bmatrix} SoC(k) \\ V_s(k) \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & \left(1 - \frac{T_s}{R_s \cdot C_s}\right) \end{bmatrix} \cdot \begin{bmatrix} SoC(k) \\ V_s(k) \end{bmatrix} + \begin{bmatrix} -\frac{T_s}{Q_T \cdot 36} \\ \frac{T_s}{C_s} \end{bmatrix} \cdot I_b(k) \tag{10}$$

$$y = V_b(k) = V_{oc}(SoC(k)) - I_b(k) \cdot R_o - V_s(k) \tag{11}$$

To simplify the notation the state space is written as:

$$x(k+1) = A_b \cdot x(k) + B_b \cdot I_b(k) \tag{12}$$

$$V_b(k) = V_{oc}(k, SoC) - I_b(k) \cdot R_o - V_s(k) \tag{13}$$

2 Open circuit map and hysteresis modelling

The $SoC - V_{oc}$ relationship in lead-acid batteries is not only non-linear, but also presents a high hysteresis phenomenon (figure 5). Note that (figure 4), when the battery current is equal to zero (load = ∞) and the capacitor C_s is completely discharge, the battery voltage V_b is equal to V_{oc}. So to obtain the outer curves of figure 5, a map could be constructed by applying a current pulse to the battery, and enough time is waited so the voltage is stable and $V_b = V_{oc}$. The state of charge variation due to the applied current pulse is obtained by equation 9. Finally a linear regression can be used to find a function that describes the charging (upper boundary U_{ub}) and discharging (lower boundary U_{lb}) curves.

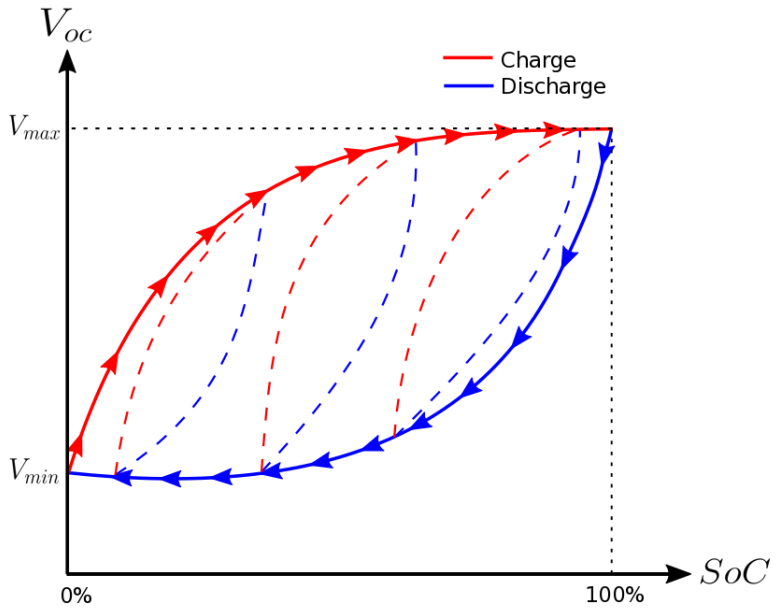

Figure 5: Voc v.s SoC

The procedure described above, provides the curves when no variation of current direction occur from 0% to 100% of the SoC and vice versa. However, inside curves produced by partial charge and discharge, remain unknown. The following mathematical modeling is based on Thele hysteresis model [7].

The $V_{oc}(k)$ can be expressed as[5]:

$$V_{oc}(k) = U_{lb}(SoC) + U_{hyst}(k) \tag{14}$$

[5] $V_{oc}(k, SoC)$ dependency of SoC will be ignored in the notation from now on

Where U_{hyst} is an hysteresis voltage produced by a partial charge or discharge of the battery. Note that V_{oc} is bounded by U_{lb} and U_{ub}, so:

$$U_{hyst}\epsilon[0, U_{hyst-max}] \tag{15}$$
$$U_{hyst-max} = U_{ub}(SoC) - U_{lb}(SoC) \tag{16}$$

On the other hand, the current sum in an inner hysteresis cycle is named Q_{hyst}, which is the charge in $[Ah]$ given or received by the battery, while the V_{oc} is evolving inside the lower and upper boundaries shown in figure 5 (dotted trajectories). Likewise U_{hyst}, Q_{hyst} only change when inner hysteresis occur, therefore it is bounded by $Q_{hyst} = 0$, and $Q_{hyst} = Q_{hyst}^{max}$, which is the amount of charge necessary for V_{oc} to pass from U_{lb} to U_{ub}. The following equations resumed the stated above:

$$Q_{hyst}(k) = Q_{hyst}(k) - \frac{I_b \cdot T_s}{3600} \tag{17}$$
$$Q_{hyst}(k)\epsilon[0, Q_{hyst}^{max}] \tag{18}$$

Note that, when $Q_{hyst}(k) = 0$, charge is not given by an inner hysteresis cycle so, $U_{hyst}(k) = 0$, therefore $V_{oc}(k) = U_{lb}(SoC)$. On the other hand if $Q_{hyst} = Q_{hyst}^{max}$, charge given by an hysteresis cycle is saturated, so $U_{hyst} = U_{hyst}^{max}$ and $V_{oc} = U_{lb}(SoC) + U_{hyst}^{max} = U_{ub}(SoC)$. Consequently, U_{hyst} boundaries are already known, however, how it evolves between those boundaries is left to determine. In figure 6 a full inner hysteresis cycle[6] is shown and two additional parameters are introduced $\Delta U_{hyst-max}^{char}$ and $\Delta U_{hyst-max}^{dis}$, which are the maximum voltage deviation from the center of the hysteresis curve. Notice that the maximum deviation occurred when $Q_{hyst} = \frac{Q_{hyst}^{max}}{2}$.

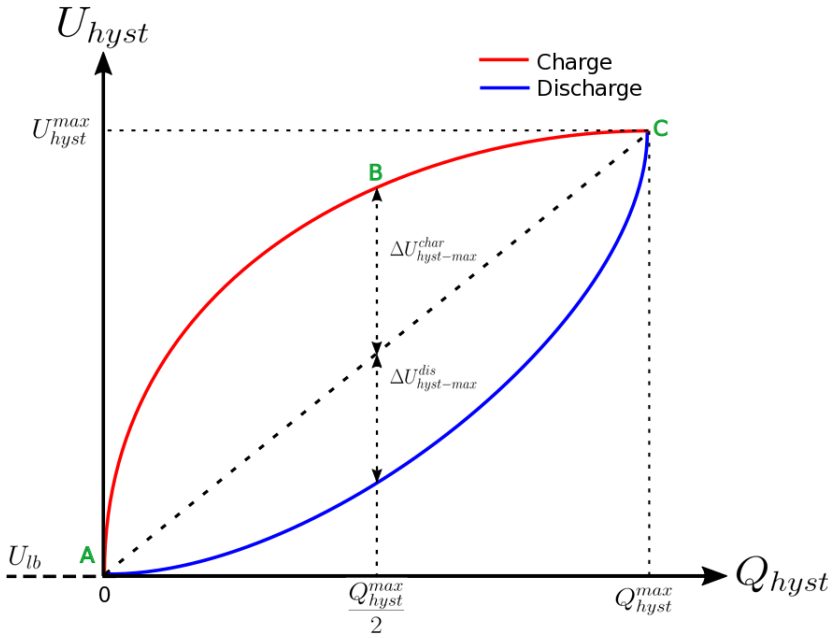

Figure 6: U_{hyst} v.s Q_{hyst}

Assuming that all the hysteresis parameters mentioned are known, coordinates A, B and C of figure 6 can be calculated, therefore a second order polynomial regression can be applied to find the function that describes U_{hyst} evolution. The remaining problem, regards in how inner partial hysteresis behave. To illustrate this issue, figure 7 shows two possible inner partial hysteresis curves. The sequence starts in A

[6]By full, it is meant that a complete transition from U_{lb} to U_{ub} or vice versa is made

with $U_{hyst} = 0$ and $Q_{hyst} = 0$. Since the battery is charging, then the voltage increases following A-C-B trajectory. When charging is finished at C, then the voltage decreases following C-D'-A trajectory, until it reaches E (current direction change again). From E, charging occur again, and the voltage follows the E-F-B trajectory.

According to Thele [] experimentation proved that inner curves are linearly related with the outer curves by a factor dependent of Q_{hyst}. This fact, enable to calculate the intermediate points (D, D' and F, F'), Therefore, a second order lineal regression could be used to describe the actual behavior of U_{hyst}. Note that every inner charge or discharge, tend to finish in A or B respectively. Applying some basic geometrical facts the following equations could be find

$$\Delta U^{dis}_{hyst} = \Delta U^{dis}_{hyst-max} \cdot \frac{Q_{hyst}}{Q^{max}_{hyst}} \tag{19}$$

$$\Delta U^{char}_{hyst} = \Delta U^{char}_{hyst-max} \cdot \frac{Q^{max}_{hyst} - Q_{hyst}}{Q^{max}_{hyst}} \tag{20}$$

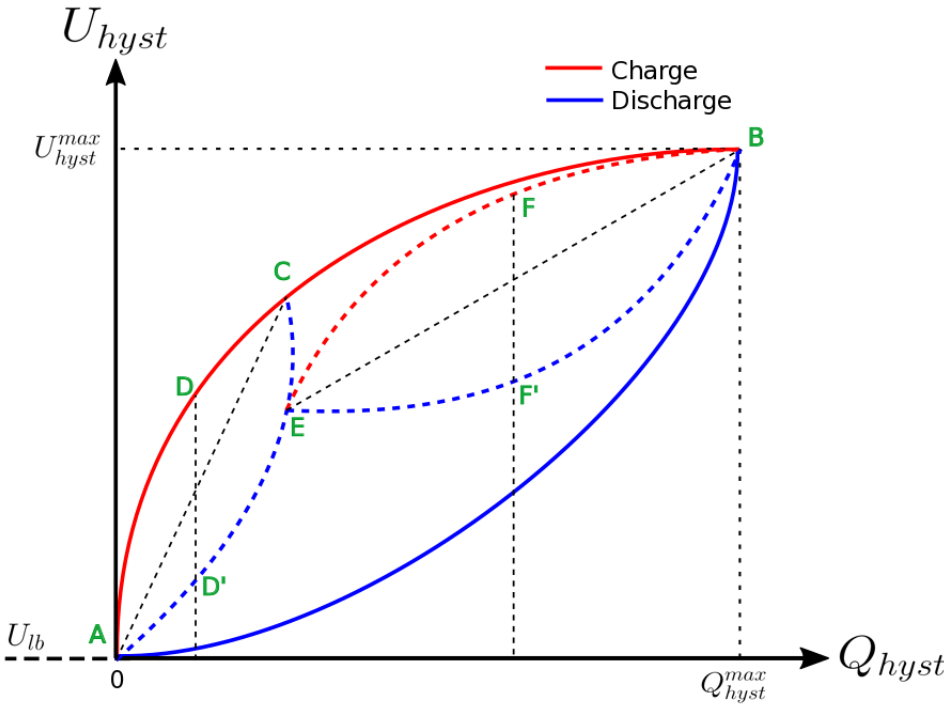

Figure 7: U_{hyst} v.s Q_{hyst} 2

To clarify the previous model, the following algorithm describes how U_{hyst} is calculated as Q_{hyst} is changing due to the charging and discharging of the battery in the sequence shown in figure 7:

1. The sign of current is identified so the battery is charging, then the final point (reference) will be $B = [Q^{max}_{hyst}, U^{max}_{hyst}]$. The middle point M is calculated using ΔU^{dis}_{hyst}, that in this case is $\Delta U^{char}_{hyst-max}$ (because $Q_{hyst} = 0$).
2. Knowing A, B and M coordinates, $f(Q_{hyst})$ is found, using a linear regression as describes previously.
3. As long as the current direction does not change, U_{hyst} is calculated from $f(Q_{hyst})$ found in the previous step.

4. When a change of direction occur in C, a new function needs to be calculated, C is the starting point, $A = [0, 0]$ the reference point and D' is the middle point which is calculated using equation [].

5. Finally in point E a change of current direction is sensed, then the previous procedure is repeated using B as reference and F' as middle point.

2 Battery parameter identification and model validation

The remaining electrical parameters of the battery can be obtained as follows:

1. R_o: Find the instant variation of the battery voltage ΔV_b from an instant current variation ΔI_b. Then compute $R_o = \frac{\Delta V_b}{\Delta I_b}$.

2. R_s and C_s: The RC network define a non-observable state, so it must be identified using time constants. For an instant current variation ΔI_b , a voltage variation will occur. The instant variation, is determine by the series resistor R_o, then the transient response, correspond to the RC network ($\tau = 5 \cdot R \cdot C$).

Therefore, the electrical parameters could be found with the experiment described to obtain the SoC v.s V_{oc} map. In order to validate the model, a computational benchmark of a battery (Matlab) is used. This benchmark include different real behaviors such as internal resistance and rated capacity variation due to current rate and temperature. In figure 8 the voltage response to a $4\ A$ current pulse is shown. Note that as soon as the current is zero, an instant voltage variation occur, due to R_o. The dynamic response (voltage stabilization) correspond to the RC network.

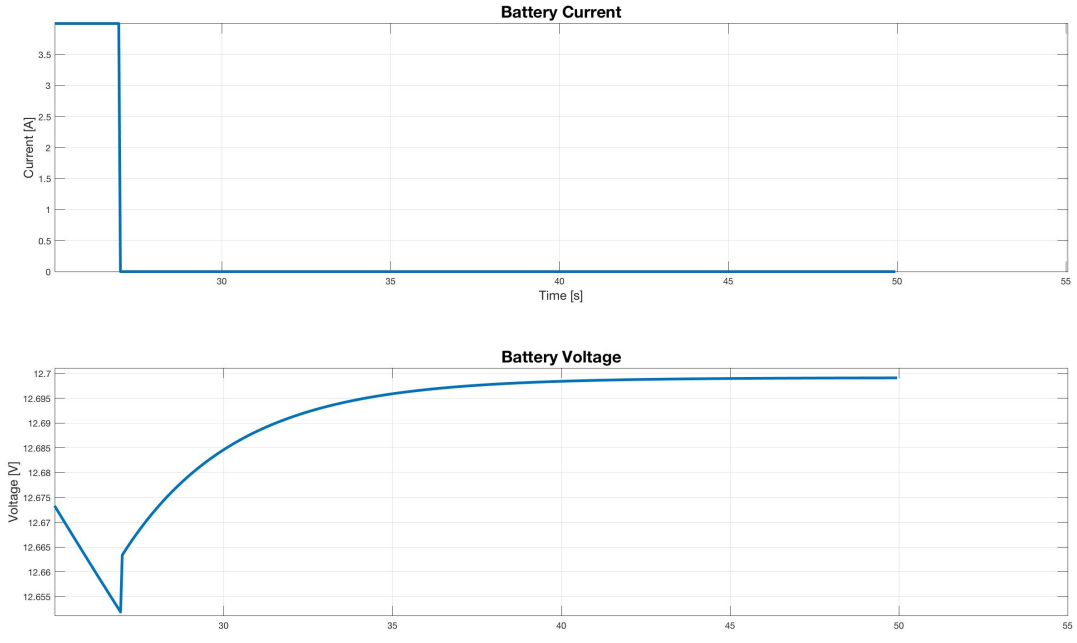

Figure 8: Battery Voltage Response

In figure 9 a series of pulses were made so the battery is completely discharge from a 100 % of SoC, and charged again, to obtain the map of figure 10. With this points a linear regression was made using a truncated Fourier 6th-order function:

$$
\begin{aligned}
U(SoC) = a_0 &+ a_1 \cdot cos(\omega \cdot SoC) + b_1 \cdot cos(\omega \cdot SoC) \\
&+ a_2 \cdot cos(2 \cdot \omega \cdot SoC) + b_2 \cdot cos(2 \cdot \omega \cdot SoC) \\
&+ a_3 \cdot cos(3 \cdot \omega \cdot SoC) + b_3 \cdot cos(3 \cdot \omega \cdot SoC) \\
&+ a_4 \cdot cos(4 \cdot \omega \cdot SoC) + b_4 \cdot cos(4 \cdot \omega \cdot SoC) \\
&+ a_5 \cdot cos(5 \cdot \omega \cdot SoC) + b_5 \cdot cos(5 \cdot \omega \cdot SoC) \\
&+ a_6 \cdot cos(6 \cdot \omega \cdot SoC) + b_6 \cdot cos(6 \cdot \omega \cdot SoC)
\end{aligned}
$$

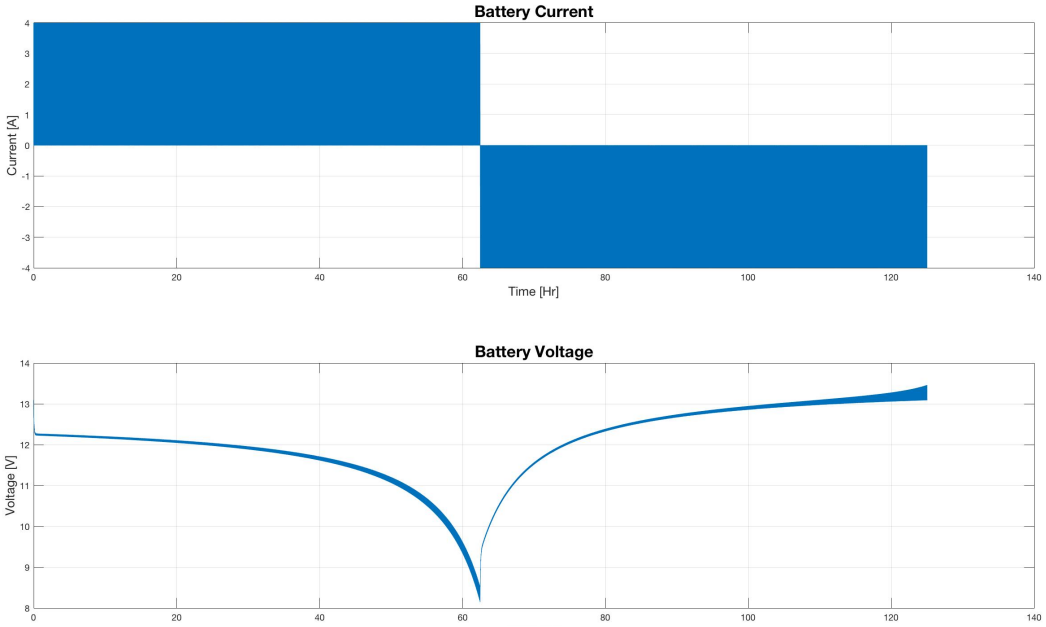

Figure 9: Battery Voltage Response

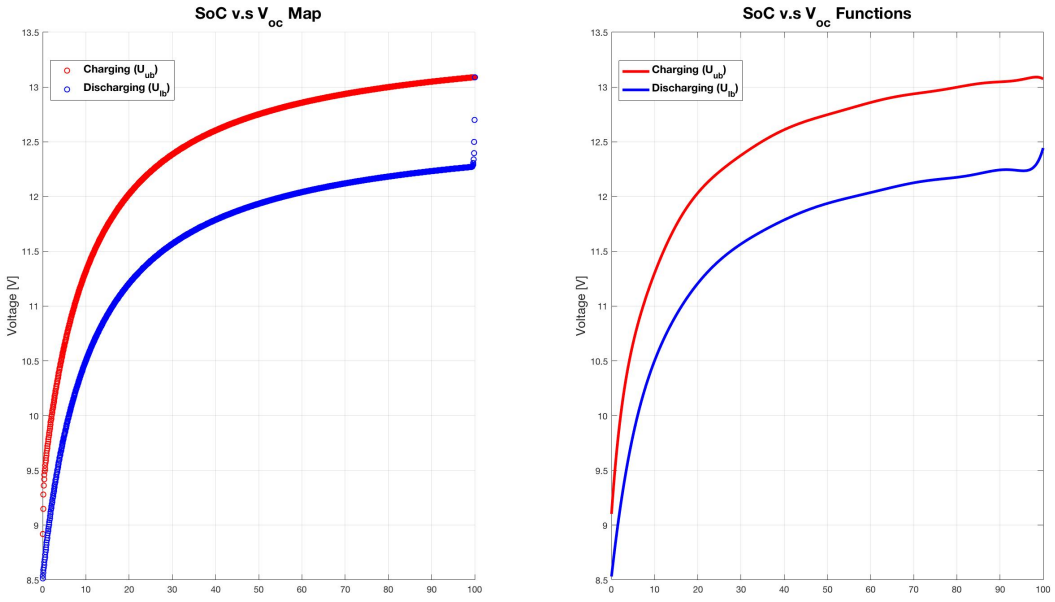

Figure 10: SoC v.s V_{oc}

To obtain the hysteresis parameters, four inner full hysteresis trajectories were found, as shown in figure 11. Traslading to Q_{hyst}-U_{hyst} space (figure 12) the parameters of figure 13 were obtain. Notice that there is a considerable difference between the parameter found for $10\ \%$ and the others, and since the batteries are not meant to work on that range, the final parameters were obtain by taking the mean of 85%, 60% and 35% (figure 14). Another option, if considerable differences were obtain for all SoC ranges, is to interpolate the parameters as a function of SoC.

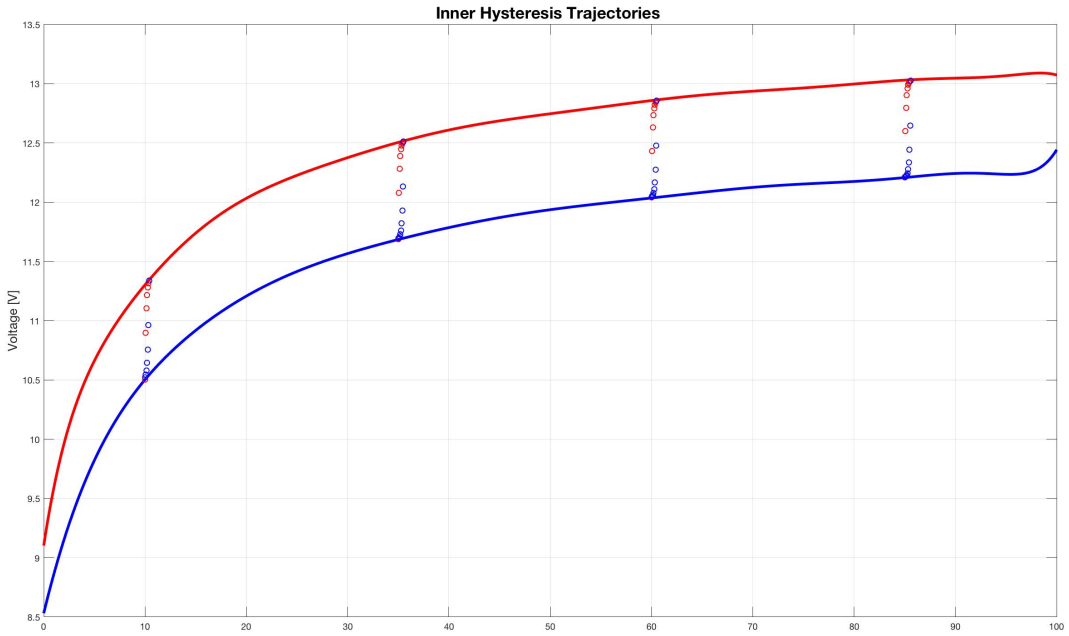

Figure 11: Inner Hysteresis Trajectories

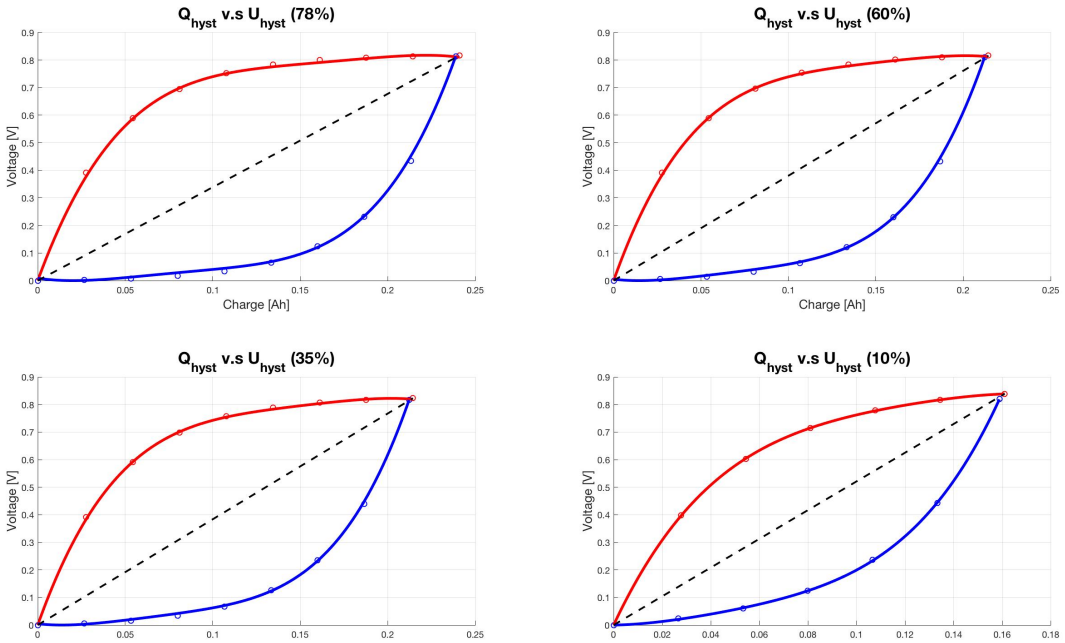

Figure 12: Uhyst Full Hysteresis Trajectories

Parameters 85%		Parameters 60%	
Parameter	Value	Parameter	Value
Q_{hyst_max}	0.2411 Ah	Q_{hyst_max}	0.2144 Ah
V_{hyst_max}	0.8166 V	V_{hyst_max}	0.8159 V
$\Delta V^{Char}_{hyst\_max}$	0.3567 V	$\Delta V^{Char}_{hyst\_max}$	0.3422 V
$\Delta V^{Dis}_{hyst\_max}$	0.3591 V	$\Delta V^{Dis}_{hyst\_max}$	0.3388 V
Parameters 35%		**Parameters 10%**	
Parameter	Value	Parameter	Value
Q_{hyst_max}	0.2144 Ah	Q_{hyst_max}	0.1611 Ah
V_{hyst_max}	0.8226 V	V_{hyst_max}	0.8389 V
$\Delta V^{Char}_{hyst\_max}$	0.3423 V	$\Delta V^{Char}_{hyst\_max}$	0.2938 V
$\Delta V^{Dis}_{hyst\_max}$	0.3387 V	$\Delta V^{Dis}_{hyst\_max}$	0.2741 V

Figure 13: Hysteresis parameters obtain for different state of charge ranges

Mean Parameters	
Parameter	Value
Q_{hyst_max}	0.2233 Ah
V_{hyst_max}	0.8184 V
$\Delta V^{Char}_{hyst\_max}$	0.3471 V
$\Delta V^{Dis}_{hyst\_max}$	0.3455 V

Figure 14: Hysteresis parameters

Finally to validate the hysteresis model, a test of current pulses was made (Figure 15), obtaining a battery voltage RMS error of $0.03\ V$, and a SoC error of $0.04\ \%$.

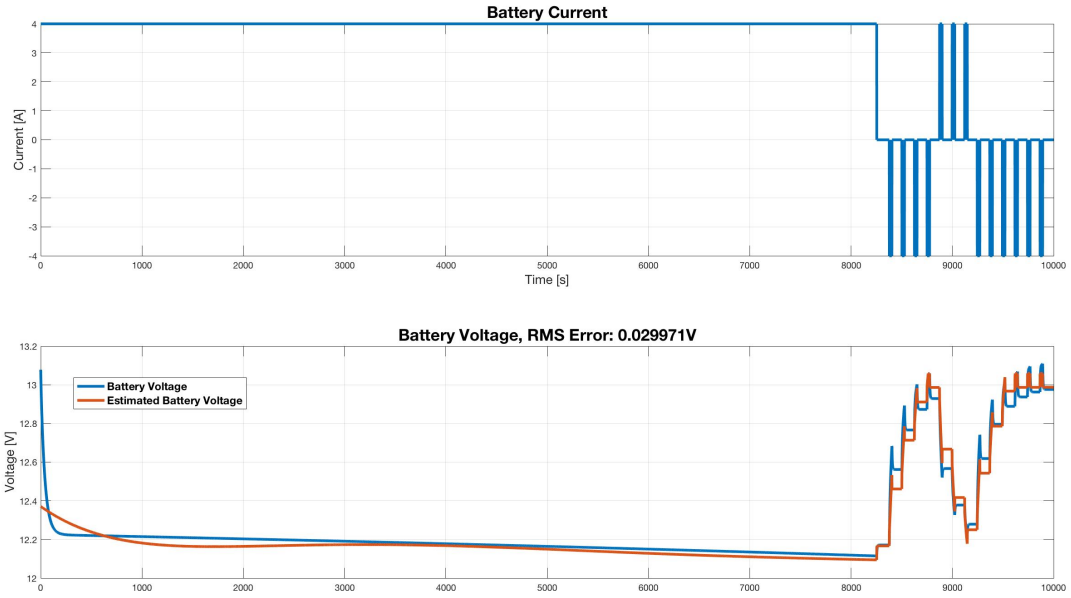

Figure 15: Test Voltage Model

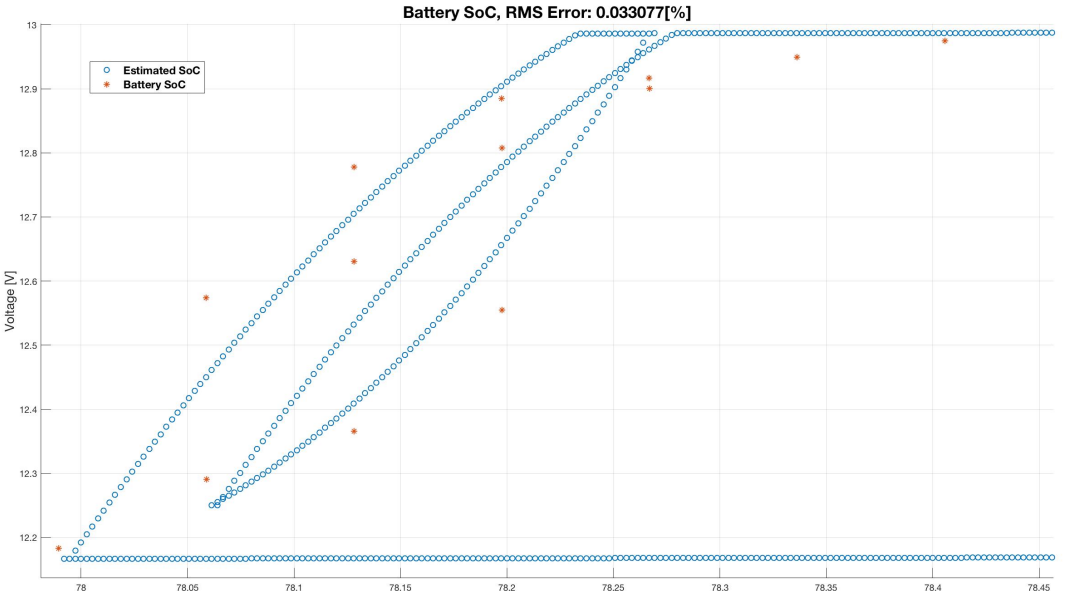

Figure 16: Test SoC Model

2 State of Charge estimator

This section is organized as follows. The SoC estimation algorithm is derived based on an extended Kalman filter and the single battery model of equations 12, 13 and the hysteresis model explained in the previous section. Then, applying a current profile, the state of charge tracking is validated for a single battery. It is important to mention that, the hysteresis model defines a new state U_{hyst}, which defines a non-linear and time-variant model. However, it is a parameter that depends considerably from the SoC and that is commonly saturated, therefore it is treated as a parameter that is calculated online.

Retaking the circuit model equations:

$$x(k+1) = A_b \cdot x(k) + B_b \cdot I_b(k) \tag{21}$$

$$V_b(k) = V_{oc}(SoC) - I_b(k) \cdot R_o - V_s(k) \tag{22}$$

Note that the state equations are linear (without considering saturation). On the other hand, the output equation contain a nonlinear term (V_{oc}), so linearization of the output equation is needed:

$$V_b(k) = V_{oc}(SoC) - V_s - R_o \cdot I_b(k) : F(SoC, V_s) \tag{23}$$

$$\bar{V_b}(k) = C \cdot x(k) + D \cdot u(k) \tag{24}$$

Then

$$D = \left[\frac{\partial F(SoC, V_s)}{\partial I_b}\right] = -R_o \tag{25}$$

$$C = \left[\frac{\partial F}{\partial SoC}, \frac{\partial F}{\partial V_s}\right] = \left[\frac{\partial F}{\partial SoC}, -1\right] \tag{26}$$

$$C = \left[\frac{\partial V_{oc}(SoC)}{\partial SoC}, -1\right] \tag{27}$$

Decomposing $V_{oc}(SoC)$:

$$V_{oc}(SoC) = U_{lb}(SoC) + U_{hyst}(Qhyst) \tag{28}$$

Notice that, when U_{hyst} is saturated on its top boundary $V_{oc}(SoC) = U_{ub}(SoC)$. when U_{hyst} is zero, then $V_{oc}(SoC) = U_{lb}$. On the other hand, note that a variation of SoC produce a proportional variation of Q_{hyst}, as long as it is not saturated, so we can define:

$$\frac{\partial V_{oc}}{\partial SoC} = \begin{cases} \frac{\partial U_{lb}(SoC)}{\partial SoC} & \text{if } Q_{hyst} = 0 \\ \frac{\partial U_{ub}(SoC)}{\partial SoC} & \text{if } Q_{hyst} = Q_{hyst}^{max} \\ \frac{\partial U_{hyst}(SoC)}{\partial SoC} + \frac{\partial U_{lb}(SoC)}{\partial SoC} & \text{if } 0 < Q_{hyst} < Q_{hyst}^{max} \end{cases} \tag{29}$$

Therefore, the SoC estimation algorithm can be resumed as follows:

1. Initialize algorithm with initial guesses for states and additional parameters:
   $SoC = SoC_{Ini}$
   $V_s = V_{s-Ini}$
   $Q_{hyst} = Q_{hyst-Ini}$
   $U_{hyst} = U_{hyst-Ini}$
   $Q = Q_{Ini}$
   $R = R$
   $I_b = 0$
   $C_{par} = [0, 0, 0]^T$: Coefficients of 2-nd order polynomial (Inner hysteresis function).

2. Read actual current I_b and voltage V_b. Determine if the direction of the current has change $I_{dir} = 1$ or not $I_{dir} = 0$.

3. Using I_b, predict the states and Q_{hyst} evolution:

   $x(k+1) = A_b \cdot x(k) + B_b \cdot I_b(k)$

   $Q_{hyst}(k+1) = Q_{hyst}(k) - \frac{I_b(k) \cdot T_s}{3600}$

   Saturate $SoC(k+1)$ and $Q_{hyst}(k+1)$ if previous defined boundaries are exceeded.

4. Project the co-variance matrix:

   $Pp(k+1) = A_b * Pp(k) * A_b^T + Q$

5. If Q_{hyst} is saturated pass to step 9. If not, calculate the hysteresis parameters:

   $\Delta U_{hyst}^{dis} = \Delta U_{hyst-max}^{dis} \cdot \frac{Q_{hyst}}{Q_{hyst}^{max}}$

   $\Delta U_{hyst}^{char} = \Delta U_{hyst-max}^{char} \cdot \frac{Q_{hyst}^{max} - Q_{hyst}}{Q_{hyst}^{max}}$

   $U_{hyst}^{max}(SoC + Q_{hyst}^{max}) = U_{ub}(SoC + Q_{hyst}^{max}) - U_{lb}(SoC + Q_{hyst}^{max})$

6. If current direction does not change ($I_{dir} = 0$), then the function for U_{hyst} has not changed so pass to step 8. If current direction change ($I_{dir} = 1$) occur pass calculate the new coefficients as follows:

   If the battery is charging ($I_b < 0$):

   $P_{ref} = \left[Q_{hyst}^{max}, U_{hyst}^{max}\right]$

   $P_{ini} = [0, 0]$

   $m = \frac{U_{hyst}^{max} - U_{hyst}}{Q_{hyst}^{max} - Q_{hyst}}$

   $b = U_{hyst} - Q_{hyst} \cdot m$

   $Q_{mid} = Q_{hyst} + \frac{(Q_{hyst}^{max} - Q_{hyst})}{2}$

   $U_{mid} = m \cdot Q_{mid} + b + \Delta U_{hyst}^{char}$

   $P_{mid} = [Q_{mid}, U_{mid}]$

   $C_{par} = fitpoly2$

   If the battery is discharging ($I_b >= 0$):

   $P_{ref} = [0, 0]$

   $P_{Ini} = \left[Q_{hyst}^{max}, U_{hyst}^{max}\right]$

   $m = \frac{U_{hyst}}{Q_{hyst}}$

   $b = 0$

   $Q_{mid} = \frac{Q_{hyst}}{2}$

   $U_{mid} = m \cdot Q_{mid} - \Delta U_{hyst}^{dis}$

   $P_{mid} = [Q_{mid}, U_{mid}]$

   $C_{par} = fitpoly2$

7. Calculate U_{hyst} and $\frac{\partial V_{oc}(SoC)}{\partial SoC}$ as follows:

   $U_{hyst} = C_{par-1} \cdot Q_{hyst}^2 + C_{par-2} \cdot Q_{hyst} + C_{par-3}$

   $\frac{\partial V_{oc}(SoC)}{\partial SoC} = \frac{\partial U_{lb}(SoC)}{\partial SoC} + 2 \cdot C_{par-1} \cdot Q_{hyst} + C_{par-2}$

8. Calculate U_{hyst} and $\frac{\partial V_{oc}(SoC)}{\partial SoC}$ as follows:

   $$U_{hyst} = \begin{cases} 0 & \text{if } Q_{hyst} = 0 \\ U_{hyst}^{max} & \text{if } Q_{hyst} = Q_{hyst}^{max} \end{cases}$$

$$\frac{\partial V_{oc}(SoC)}{\partial SoC} = \begin{cases} \frac{\partial U_{lb}(SoC)}{\partial SoC} & \text{if } Q_{hyst} = 0 \\ \frac{\partial U_{ub}(SoC)}{\partial SoC} & \text{if } Q_{hyst} = Q_{hyst}^{max} \end{cases}$$

9. Modify C matrix as follows:
   $C = \left[\frac{\partial V_{oc}(SoC)}{\partial SoC}, -1\right]$
10. Estimate the battery voltage as follows:
    $V_{oc} = U_{hyst} + U_{lb}$
    $V_b^{est} = V_{oc} - I_b \cdot R_o - x_p[2]$
11. States and covariance matrix are corrected, using measured V_b:
    $K = P_p \cdot C^T \cdot \left(C \cdot P_p \cdot C^T + R\right)^{-1}$
    $x_p = x_p + K \cdot \left(V_b - V_b^{est}\right)$
    $P_p = (I - K \cdot C) \cdot P_p$
12. Go to step 2.

Applying a current profile, the state of charge was estimated using the previous algorithm (Figure 17), with a white measure noise with a variance of $\sigma_V = 150mV$ and $\sigma_I = 10mA$ for the battery voltage and current respectively, obtaining an RMS tracking error of $3.52\ \%$ for a 5.5 day simulation

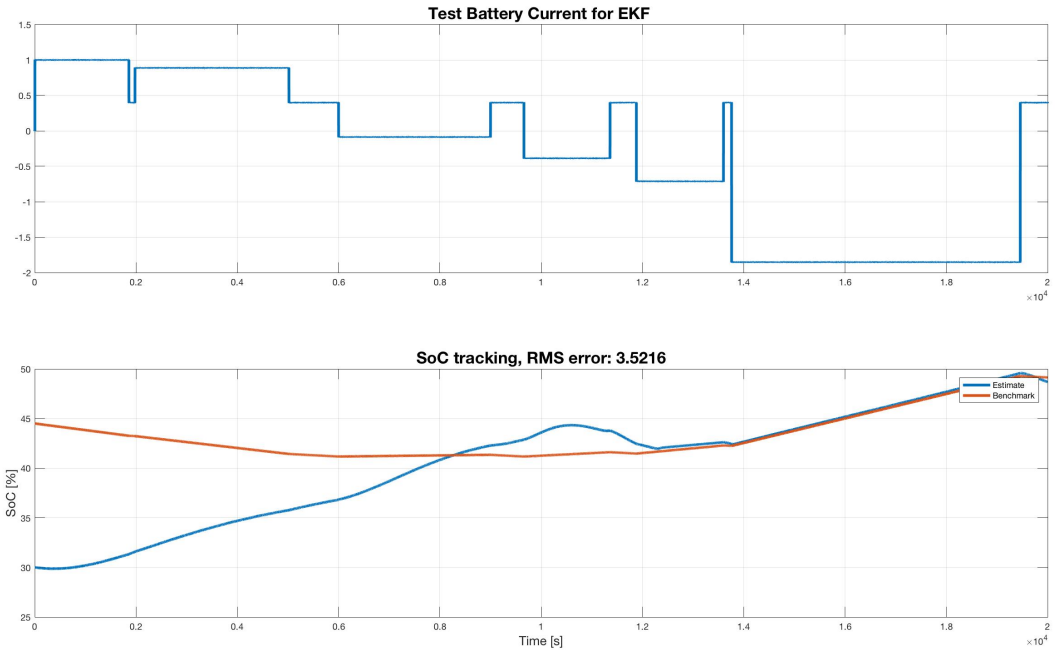

Figure 17: SoC estimation for single battery

3 PV System Model and Model Predictive Controller

3 PV system model and power estimation

To model the photovoltaic system, a correct functioning of the low level controllers is assumed, so a linear balance of energy is obtain. The MPPT[7] controller, guarantee a maximum power P_{max} is being taken from the solar panels, with an efficiency η_{DC-DC}. The P_{max} parameter is specified in STC[8], therefore it must be normalized, so the power generated by the PV system is:

$$P_{gen} = \frac{P_{max} \cdot N_{panels} \cdot \eta_{DC-DC}}{1000} \cdot R(t) \tag{30}$$

Taking into account the DC-AC converter efficiency:

$$P_{bat}(t) = \frac{Q(t)}{\eta_{DC-AC}} - P_{gen} \tag{31}$$

$$P_{bat}(t) = \frac{Q(t)}{\eta_{DC-AC}} - \frac{P_{max} \cdot N_{panels} \cdot \eta_{DC-DC}}{1000} \cdot R(t) \tag{32}$$

To estimate the power horizon (consumption and generation), a persistent[9] model is used. In figure 18 the prediction algorithm is illustrated. Notice that the power prediction of the second horizon (N_p to $2N_p$), is equal to the real power of the first (0 to N_p). Also notice that for the first N_p time samples, there is no control action.

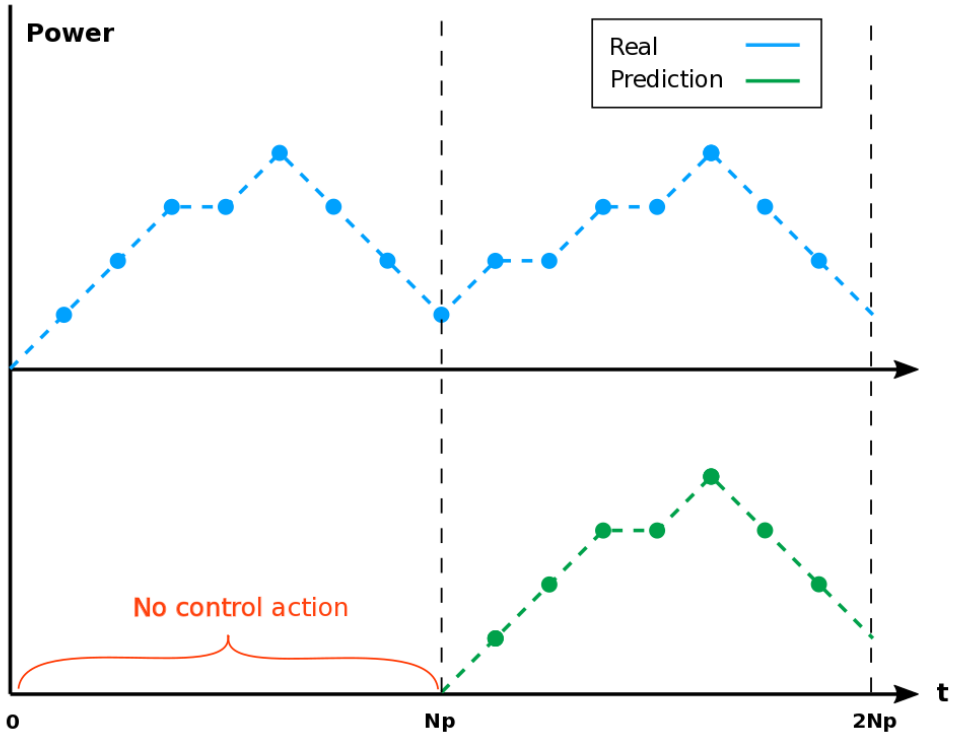

Figure 18: Power prediction

[7]Maximum Power Point Tracking
[8]Standard Test Conditions
[9]A persistent model is a model that maintains the previous version of itself

3 Model predictive control algorithm

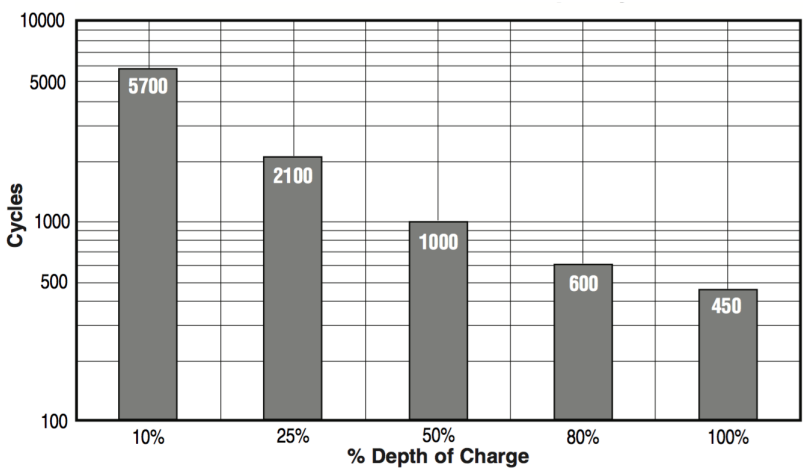

Figure 19: Depth of Discharge Curve (image taken from [1])

Battery lifetime could be measure by the number of charge-discharge cycles, can be done until a significant deterioration of the battery is achieved. In figure 19 the depth of discharge versus lifetime is shown for a 8G40 Deka Solar gel battery. This behavior defines the importance of maintaining a minimum quantity of charge in the battery, so a minimum lifetime is guarantee. Therefore, it is necessary to reduced the consumption inside the school, which can be accomplish by disabling a group of loads[8], affecting the user comfort. Taking into account this facts, the controller's objective is to maintain a minimum $SoC = SoC_{min}$ while minimizing the load disabling inside the school.

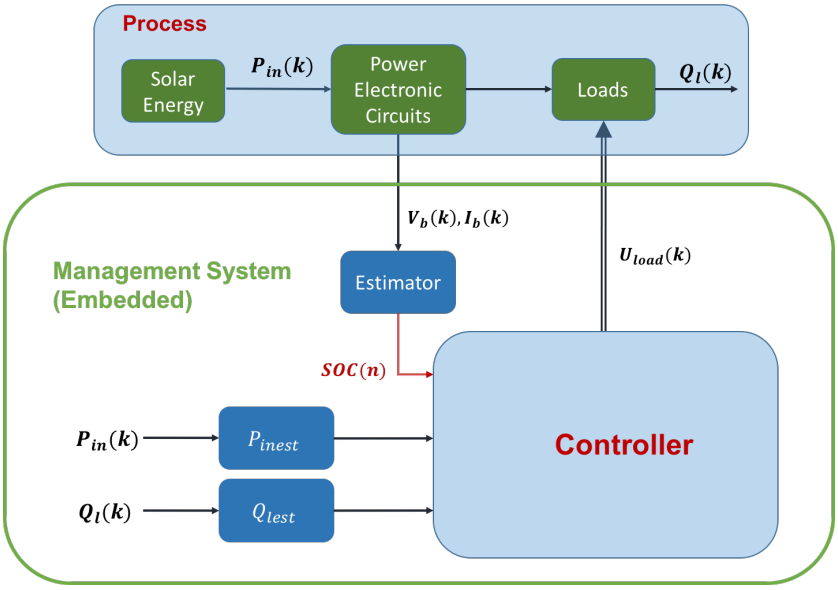

Figure 20: General control architecture

Recalling the control architecture shown previously (figure 20), note that the control signals (U_{load}) produce a possible consumption reduction $Q_r(U_{load}(k))$, then the optimal control problem is defined:

$$\underset{U_{load}(k)}{\text{minimize}} \quad J = \sum_{k=1}^{N_p} \alpha \cdot Q_r(U_{load}(k))^2 \tag{33}$$

subject to:

$$SoC(k) \leq SoC_{min} \tag{34}$$

$$0 \leq Q_r(k) \leq Q_r^{max}(k) \tag{35}$$

$$SoC(k+1) = F(U_{load}(k), k) \tag{36}$$

Where $Q_r^{max}(k)$ is equal to the consumption prediction for the day Q_{lest}. To simplify the problem, the SoC restriction is replaced for a penalization term, inside the objective function as shown in equation 37

$$\underset{U_{load}(k)}{\text{minimize}} \quad J = \sum_{k=1}^{N_p} \alpha \cdot Q_r(U_{load}(k))^2 + S(SoC(U_{load})) \tag{37}$$

subject to:

$$SoC(k+1) = F(U_{load}(k), k) \tag{38}$$

Since the problem is an integer quadratic problem (binary), moreover the need to implement simple algorithms, a simplified optimal control problem is defined:

$$\underset{Q_r(k)}{\text{minimize}} \quad J = \sum_{k=1}^{N_p} \alpha \cdot Q_r(k)^2 + S(SoC(k)) \tag{39}$$

subject to:

$$SoC(k+1) = F(Q_r(k), k) \tag{40}$$

Notice that the binary control signals $U_{load}(k)$, are replaced for a direct consumption reduction $Q_r(k)$. With the optimal signal $Q_r^*(k)$ an additional algorithm is used to choose which $U_{load}(k)$ produce a consumption reduction near to the desired $Q_r^*(k)$. In figure 21, the modified control architecture of the MPC is shown. The MPC block calculates the necessary consumption Q_r to minimize the objective function of equation 40 and the "Load Selection" block, chooses which loads must be disabled or enabled.

An additional simplification is made, in order to guarantee a low computation time. The optimum solution $Q_r^*(k)$ is found by solving $SoC(k+1) = F(Q_r(k), k)$ for several signals $Q_r^i(k)$ (for N_p sample times). With the different pair of signals $\left[Q_r^i(k), SoC(k)^i\right]$ the objective function is evaluated and the minimum solution is found. The $Q_r^i(k)$ signals are choose by scaling the power prediction as follows:

$$Q_r^i(k) = \begin{bmatrix} Q_{lest} \\ 0.9 \cdot Q_{lest} \\ \vdots \\ 0.1 \cdot Q_{lest} \\ 0 \cdot Q_{lest} \end{bmatrix}$$

So a total of 11 predictions are made to find the optimal response, in each control computation.

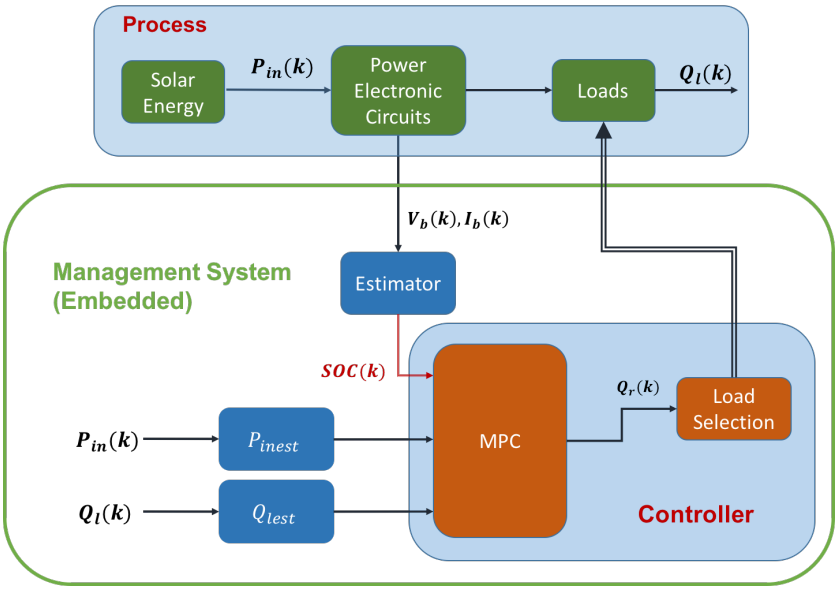

Figure 21: Simplified control architecture

To penalize the SoC restriction an inverse SoC response to Q_r is needes so the following function is defined:

$$S(SoC) = \beta \cdot (SoC - 100)^2 + S_{lin}(SoC) \tag{41}$$

$$S_{lin}(SoC) = \begin{cases} \cdot m_{soc} \cdot SoC + b_{soc} & \text{if } SoC \leq SoC_{corner} \\ 0 & \text{if } SoC \leq SoC_{corner} \end{cases} \tag{42}$$

$$\text{where: } m_{soc} = \frac{-\gamma}{SoC_{corner} - SoC_{min}} \tag{43}$$

$$b_{soc} = -m_{soc} \cdot SoC_{corner} \tag{44}$$

Where γ is a weight parameter. Note that the quadratic term does not include the desired SoC_{min}, so a linear function is used as a barrier; when $SoC(k)$ is less or equal to SoC_{corner} the barrier is activated. On the other hand γ acts as a tuning parameter, with a higher γ the barrier is more restrictive and the penalization is higher . In figure 22 the shape of the $S(SoC)$ terms are shown.

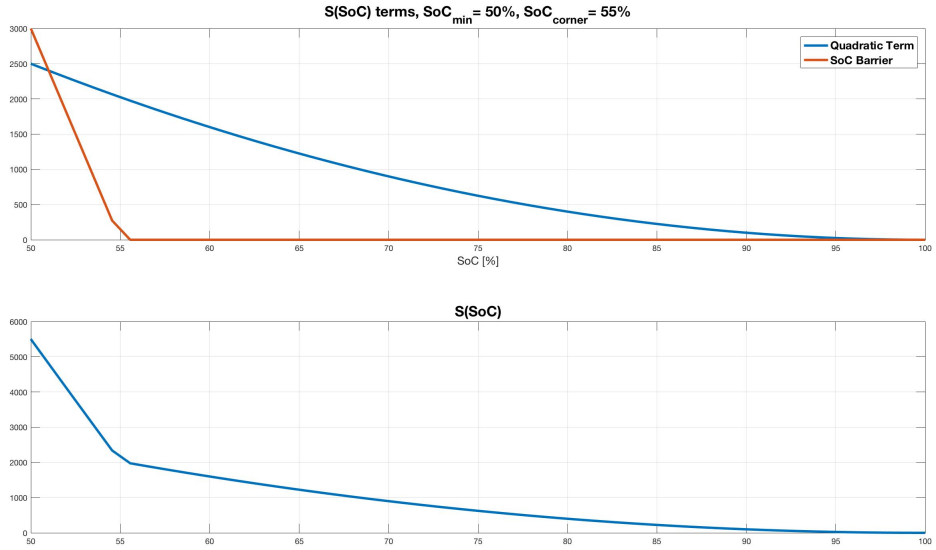

Figure 22: $S(SoC)$

The load selection block use the following procedure to choose the loads that must be disable (When $Q_r > 0$):

1. The information for each group of loads is organized as follows:

$$\begin{bmatrix} n^1_{prio} & Q^1_n & U^1_{load} & ID^1_{load} \\ n^2_{prio} & Q^2_n & U^2_{load} & ID^2_{load} \\ \vdots & \vdots & \vdots & \vdots \\ n^{n_{loads}}_{prio} & Q^{n_{loads}}_n & U^{n_{loads}}_{load} & ID^{n_{loads}}_l \end{bmatrix}$$

   Where n_{prio} is an integer that indicates the priority of the loads. Q_n is the nominal consumption for the group of loads. U_{load} indicates the actual state of the control signal for the group of loads (enable (1) or disable (0)). ID_l, is the number assigned to identified the group of loads.

2. The disabled loads are excluded from the list.
3. The matrix rows are sorted by priority (n_{prio}).
4. A column with the accumulated consumption for the loads and a column with the power deviation from Q^*_r are added.
5. The loads whose consumption deviation is smallest are selected and the control signal U_{load} is updated.

A similar procedure is followed when $Q^*_r = 0$ and loads must be enabled. To clarify the procedure an example is presented with 5 loads and $Q^*_r = 310\ [W]$:

1. The information matrix is constructed $[n_{prio}, Q_n, U_{load}, ID_{load}]$:

$$\begin{bmatrix} 5 & 60 & 0 & 1 \\ 1 & 120 & 1 & 2 \\ 4 & 180 & 1 & 3 \\ 3 & 120 & 0 & 4 \\ 2 & 100 & 1 & 5 \end{bmatrix}$$

2. Loads that are already disabled by the control system are excluded:

$$\begin{bmatrix} 1 & 120 & 1 & 2 \\ 4 & 180 & 1 & 3 \\ 2 & 100 & 1 & 5 \end{bmatrix}$$

3. Sort the matrix by load priority:

$$\begin{bmatrix} 4 & 180 & 1 & 3 \\ 2 & 100 & 1 & 5 \\ 1 & 120 & 1 & 2 \end{bmatrix}$$

4. Add an extra column with the accumulated nominal consumption and consumption reduction deviation ($Q_r^* - Q_n^i$):

$$\begin{bmatrix} 4 & 180 & 1 & 3 & 180 & 130 \\ 2 & 100 & 1 & 5 & 280 & 30 \\ 1 & 120 & 1 & 2 & 400 & 90 \end{bmatrix}$$

5. Because the minimum consumption deviation is obtained by disabling loads 3 and 5, then:

$$U_{load} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

The MPC algorithm can be resumed as follows:

1. Using the power consumption prediction Q_{lest}, calculate the signals:

$$Q_r^i(k) = \begin{bmatrix} Q_{lest} \\ 0.9 \cdot Q_{lest} \\ \vdots \\ 0.1 \cdot Q_{lest} \\ 0 \cdot Q_{lest} \end{bmatrix}$$

2. For every consumption reduction $Q_r^i(k)$ calculate the $SoC(k)^i$ response (using P_{inest}, Q_{lest} and the actual estimated SoC as well).

3. For every pair of signals, $\left[Q_r^i(k), SoC(k)^i\right]$ evaluate the objective function J and choose the pair $[Q_r^*(k), SoC(k)^*]$ that minimize J.

4. Using the optimal signal $Q_r^*(k)$, choose the load control signal U_{load} as shown previously.

The result of the MPC algorithm and the SoC estimation for the battery bank is shown in figure 23

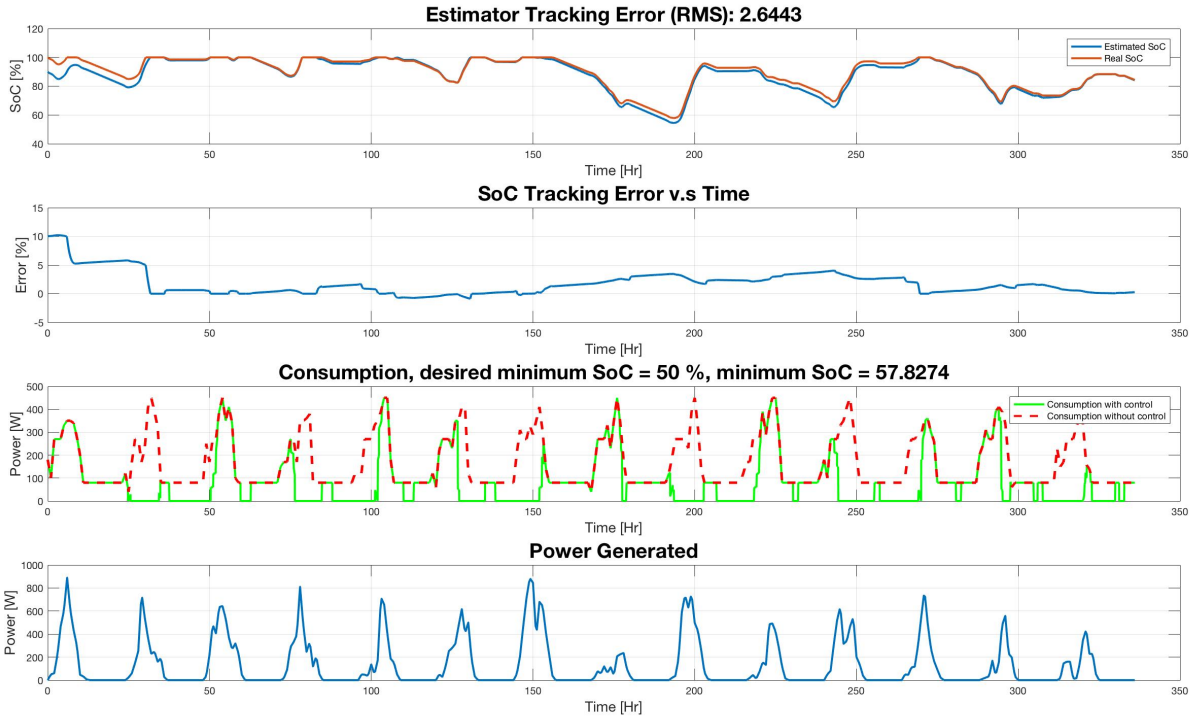

Figure 23: Control Results

4 Implementation

This chapter describes the implementation process. Before evaluating the strategy on the real plant, an emulator is developed, in order to test the control algorithm in a hardware in the loop (HIL) architecture. Through the first section a brief description of the emulation set-up is explained, as well as the implementation procedure. Finally time compression to speed up the emulation is shown along with the control system results.

4 Emulation and Hardware in the loop

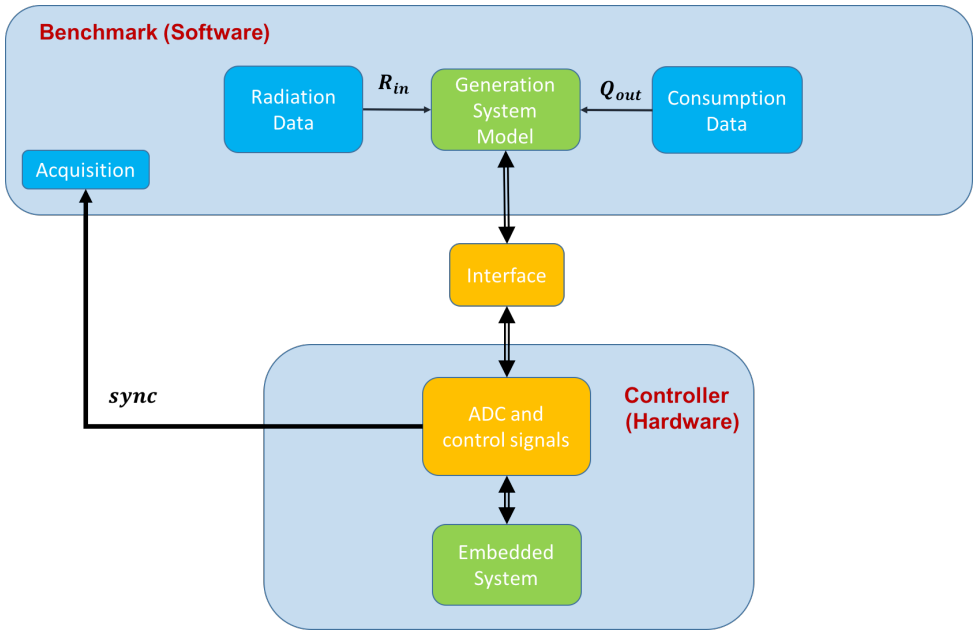

Figure 24: Hardware in the loop set up (block diagram)

In figure 24 the HIL set-up to test the controller implementation is shown. For the emulation environment Matlab/Simulink software was used. To interface the benchmark with the controller, a Q8-USB Quanser data acquisition device was used. Data from IDEAM serves as input data for the radiation profile. The consumption data was based on the school loads list, that was obtained from [8] and with the direct interaction with one of the school teachers. A randomize function was created, so the consumption varies from day to day.

The controller was implemented on a Raspberry Pi 3 model B+ using Python programming language. Due to the lack of an analog-digital converter on the Raspberry an additional circuit was used with an MCP3208 ADC, that communicates with the Raspberry using an SPI protocol. In order to synchronize the data acquisition between the controller and the benchmark, a $sync$ signal that activates every time data is saved in memory, is send to the benchmark.

On the other hand, to speed up the emulation, a time compression is used so the control system could be evaluated and adjusted faster. In order to decide the compression time, is it necessary to evaluate the computation time of the algorithms. Starting with the plant prediction and retaking the state space model defined in chapter 2, the A matrix for the battery model is defined as:

$$A = \begin{bmatrix} 1 & 0 \\ 0 & \left(1 - \frac{T_s}{R_s \cdot C_s}\right) \end{bmatrix} \tag{45}$$

Using the eigenvalues, it could be proof that the model is stable with a sample time $T_s \leq 1s$. Therefore to find the SoC response for one day (time horizon) $3600 \cdot 24 = 86400$ computations are required (3600 seconds per hour times 24 hours per day). Taking into account that for every control calculation, the system must be predicted 11 times (chapter 3), then a total of $11 \cdot 3600 \cdot 24 = 950400$ computations are required. To simplify the algorithm so a bigger time compression could be made, the following circuit simplification is proposed:

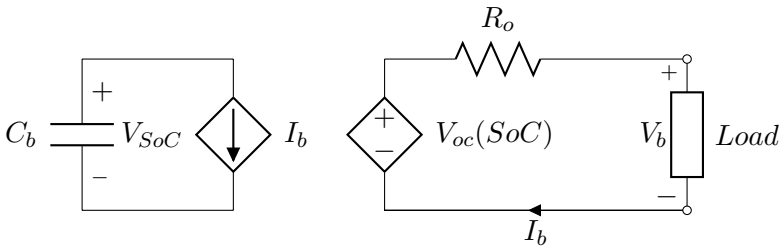

Figure 25: Simplified battery equivalent circuit model

Therefore the unstable state is assumed to be near to zero. The advantage of this simplified model, is that fewer matrix operations are required and the sample time could be increase further than $1s$ without destabilizing the controller. Taking $T_s = 12 \cdot 60\ s$ the number of computations are reduced to $\frac{24 \cdot 3600}{24 \cdot 60} = 60$ computations (which make the algorithm at least 15840 times faster). In figure 26 both SoC responses are shown, with an RMS error of 0.12%.

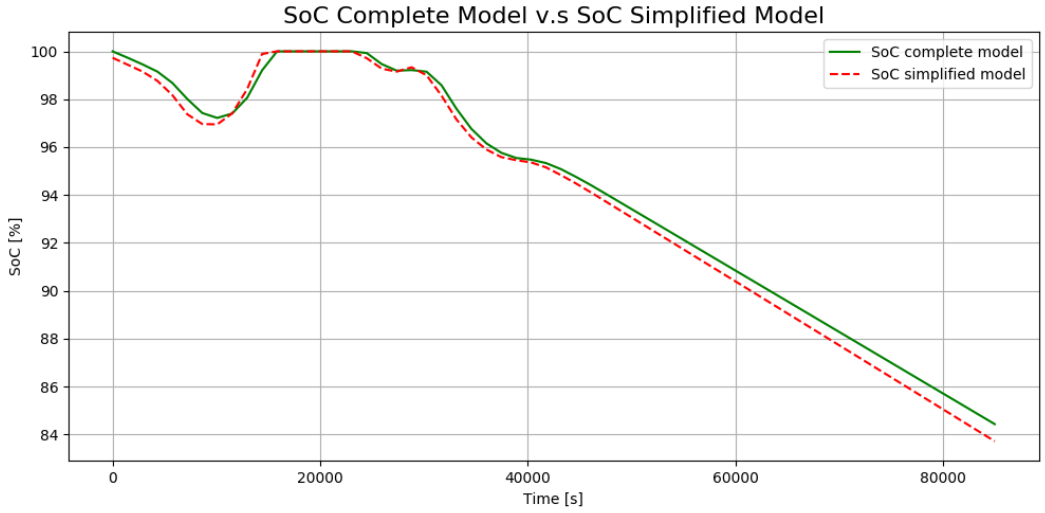

Figure 26: SoC response for simplified model v.s complete model

In figure 27 computation times for each single algorithm are shown. Taking into account the processing capability of the Raspberry Pi and the benchmark, the time values shown in figure 28 were chosen. Notice that a $1.8s$ is used for every control calculation, which is more than twice the computation time found previously. Therefore, to emulate 6 days, a total of 10.8 hours are needed, which gives a compression factor of 13.33.

Algorithm	Max Computation time $[s]$	# of Computations per cycle	Total Computation time $[s]$
Filter calculation	0.078	1	0.078
Prediction	0.068	11	0.7480
Objective function	0.00012	11	0.0013
		Total	0.8273

Figure 27: Computation times for algorithms running on the Raspberry Pi 3

Time Constant	Value$[s]$	Description
dt^{em}	0.006	Emulation time step
T_s^{em}	0.08	Equivalent time sample for every dt^{em}
dt	1.8	Sample time for the Raspberry Pi
T_s	24	Equivalent sample time for every dt

Figure 28: Time values for the HIL set-up

4 Implementation Results

To evaluate the control system, it is necessary to synchronize the benchmark acquisition, with the control system sampling. It is important to mention that the signal is being processed online by identifying the rising edge. The signals inside the benchmark are sampled using a fixed step. So to obtained the synchronized signals, only the values from the data are taken when $sync = 1$, after the emulation is finished (Matlab doesn't permit asynchronous sampling). In figure 29 benchmark and Raspberry pi synchronized battery signals are shown.

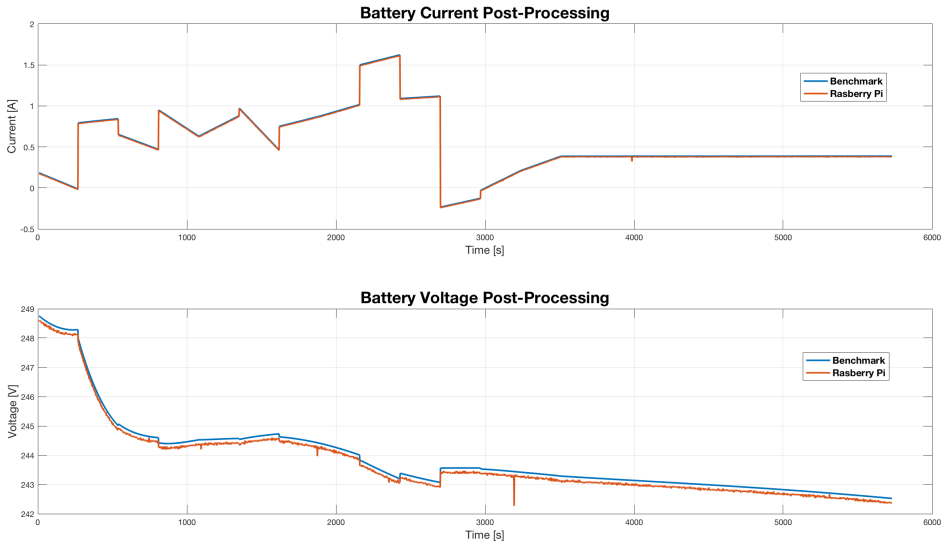

Figure 29: Signal samples obtained by syncronizing the sampling

Finally using the values of figure 30 to tune the controller and the Kalman filter the controller is test as shown in figure 31

Tuning Parameter	Value
Q	$\begin{bmatrix} 0.35 & 0 \\ 0 & 0.005 \end{bmatrix}$
R	4100
α	1.5
β	1.2
γ	9500

Figure 30: Tuning parameters

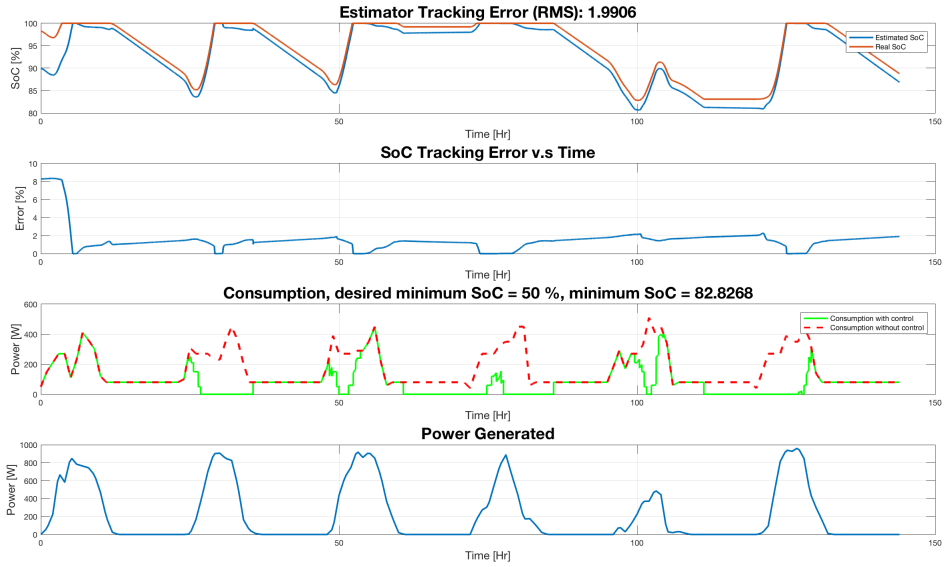

Figure 31: Controller test in the benchmark

5 Conclusions and Future Work

The battery model used in the control system, permits a SoC estimation with an error less than $5\ \%$, even when high hysteresis is present in the battery behavior. The controller, solves a very complex problem, using simple algorithms and low computation times, which lead to the possibility of implementing the controller in a low cost embedded system.

On the other hand the Raspberry Pi, is probably not the best choice to implement the controller for a field implementation. Mainly because it doesn't have an available hardware timer, therefore every timing procedure is made by software, which introduce a timing error in the sampling. In long term emulations (or real implementation) an external RTC clock could be used to correct that issue[10], however, since it is a software based micro-controller, the computation times are quite unpredictable, which can generated other implementation problems sush as RAM saturation.

The controller response depends strongly from the power estimations. Therefore a very aggressive response in the loads enabling is commonly obtained when big errors of estimation occurred (on/off controller). This could be corrected by adding a feed-forward law. On the other hand, the user consumption in real implementation will evolved; in other words, the user will modified he's behavior to minimize the load enabling, which could reduced any occasional aggressive response.

A very practical HIL set-up was design so the control algorithm could be tested without using the real plant and reducing substantially the time needed to test any modification in the controller/filter or the PV system. Future work, may center their efforts in real implementation issues. For instance an actuator design that use a latch relay to reduced considerably the power consumption could be tested. Finally depending on the battery choose for the PV system, the model could be obtained by using an electronic load and a voltage power source to produce a series of current pulses as shown in chapter 2.

[10]There are hardware interruptions available in the GPIO ports of the Raspberry

References

[1] Deka Solar. Photovoltaic Batteries. 2012.

[2] Ministerio de Minas y Energía. Decreto 1623. pages 1–12, 2015.

[3] UPME. Sector Eléctrico Nacional. *Informe Sectorial Sobre La Evolución De La Distribución Y Comercialización De Energía Eléctrica En Colombia*, pages 20–41, 2010.

[4] F Adinolfi, F Conte, S Massucco, M Saviozzi, F Silvestro, and Politecnico Milano. Performance Evaluation of Algorithms for the State of Charge Estimation of Storage Devices in Microgrid Operation.

[5] Van Huan Duong, Ngoc Tham Tran, Woojin Choi, and Dae Wook Kim. State estimation technique for VRLA batteries for automotive applications. *Journal of Power Electronics*, 16(1):238–248, 2016.

[6] Shiqi Qiu, Zhihang Chen, M a Masrur, and Y L Murphey. Battery hysteresis modeling for state of charge estimation based on Extended Kalman Filter. *Industrial Electronics and Applications (ICIEA), 2011 6th IEEE Conference on*, pages 184–189, 2011.

[7] Marc Thele, Oliver Bohlen, Dirk Uwe Sauer, and Eckhard Karden. Development of a voltage-behavior model for NiMH batteries using an impedance-based modeling concept. *Journal of Power Sources*, 175:635–643, 2008.

[8] Pontificia Universidad Javeriana Juan Manuel Mejia. Sistema de Administracion de Energia en una Microred Electrica para una Escuela rural del Departamento de Cundinamarca. Technical report, Pontificia Universidad Javeriana.