Master Thesis

Real-Time Battery Conditions Estimation

Energetic framework definition and algorithm implementation for the real-time determination of the batteries' SoC and SoH

Stefano Pregnolato

Relatore
Prof. Giuseppe Carlo Calafiore

Final Report for the Thesis in
Mechatronic Engineering Master's Degree

Department of Control and Computer Engineering
Politecnico di Torino
Torino, Italy
October 2019

Real-Time Battery Conditions Estimation

Energetic framework definition and algorithm implementation for the real-time determination of the batteries' SoC and SoH

Stefano Pregnolato

Mentor
Prof. Giuseppe Carlo Calafiore

Supervisor
Ing. Giovanni Guida - brain Technologies

Abstract

This academic work is part of the BAT-MAN research and development industrial project owned by *brain Technologies*, sponsored by the regional contribution POR FESR 2014-2020 (European fund for the regional development) and whose main goal is the realisation of an electronic device capable of detecting and forecasting, in real-time, the working conditions of a Lead-Acid battery. Entering the team as Algorithm and Control Engineer, I've been in charge of analysing the problem, defining experimental campaigns and creating the algorithm for the real-time batteries' states estimation. The work can be divided in three major section:

- Energetic Framework definition
- Battery modelling
- Model-based Solution

The definition of a rigorous Energetic Framework, that mathematically describes the main quantities necessary to define the state of a battery (SoC, SoH, etc.) and the energy exchanges, was the first solid milestone on which building all the reasoning. Then, a suitable battery model were built in order to define strategy for the final model-based algorithm, always balancing between computational effort, robustness, required precision and effectiveness. The final solution, implemented in Mathworks environment (Matlab, Simulink, Simscape, Stateflow) was eventually exported with the automatic code generation and the Software Team has been responsible for the micro-controller integration in the first real prototype.

Contents

1 Introduction **14**
1.1 The Problem . . . 14
1.1.1 Key Trends, Market and Industry Forces . . . 15
1.2 The Project . . . 16
1.2.1 Goal . . . 16
1.2.2 Stakeholders . . . 16
1.2.3 Costs . . . 17

2 Batteries **19**
2.1 Actual Technology . . . 19
2.1.1 Lithium-Ion . . . 19
2.1.2 Lead Acid . . . 20
2.2 Future Technology . . . 21
2.2.1 Solid Electrolite . . . 21

3 State of Art **23**
3.1 Battery Modelling . . . 23
3.1.1 Analytical modelling of electrochemical phenomena . . . 23
3.1.2 Data-Driven modelling (Black-Box) . . . 25
3.1.3 Physical modelling (Electrical equivalent circuit) . . . 27
3.1.4 Ibrid modelling . . . 28
3.2 States Estimation Technology . . . 28
3.2.1 Model-Based . . . 29
3.2.2 Data-Driven . . . 30
3.3 Critiques to the State of Art . . . 31

4 Energetic Framework **34**
4.1 Definitions . . . 34
4.1.1 Capacity . . . 34
4.1.2 State of Charge . . . 36
4.1.3 Depth of Discharge . . . 36
4.1.4 State of Health . . . 36
4.2 Measurable Quantity Analysis . . . 36
4.2.1 Absolute approach . . . 37
4.2.2 Relative approach: C_{real} ignorance . . . 37
4.2.3 Relative approach: relative SoC . . . 38

5 Model Identification **40**
5.1 Design of Experiment . . . 40
5.1.1 DoE theory . . . 40
5.2 BAT-MAN DoE . . . 44
5.2.1 Acquisition test - 0 . . . 44
5.2.2 Acquisition test - 1 . . . 45
5.2.3 Acquisition test - 2 . . . 46
5.2.4 Data Organisation . . . 47
5.2.5 Full BAT-MAN DoE . . . 47
5.3 Fundamental quantities . . . 48
5.3.1 Discharge Profile Analysis . . . 48
5.3.2 Real capacity . . . 49
5.3.3 Open Circuit Voltage Characteristic . . . 49
5.4 Parameters Identification . . . 50
5.4.1 Least Square Estimation Theory . . . 51
5.4.2 R Model . . . 53
5.4.3 Dynamic Model . . . 54
5.4.4 Non-Linear R Model . . . 57

6 Battery Model **63**
6.1 The Structure . . . 63
6.1.1 Voltage Source model block . . . 64
6.1.2 Resistor model block . . . 66
6.2 Model Equations . . . 68
6.2.1 State Equation . . . 69
6.2.2 Output Equation . . . 69
6.2.3 Overview . . . 70
6.3 Performance . . . 70
6.3.1 Calibration . . . 71
6.3.2 Test . . . 71

7 Model-based Solution **75**
7.1 Single Non-Linear State Observer . . . 75
7.1.1 Extended Kalman Filter . . . 75
7.1.2 Implementation . . . 77
7.1.3 Real data performance . . . 78
7.2 Real-Time Batteries' SoC and SoH Estimator . . . 80
7.2.1 Series of ANSE . . . 81
7.2.2 The Logic . . . 83
7.2.3 Algorithm Performances . . . 84

8 Conclusions **95**

A Appendix **97**
A.1 Matlab Code . . . 97
A.1.1 Open Circuit Voltage Characteristic . . . 97
A.1.2 Static non-linear model . . . 100
A.1.3 Real capacity . . . 113
A.1.4 Absolute Interpretation: R, RC, R_{nl} . . . 115

A.2 Simulink Code 124
A.2.1 Model 124
A.2.2 EKF 124
A.2.3 ANSE 124
A.2.4 Final Solution 124

List of Figures

1.1 Basic battery model, [7] 14
1.2 Open circuit voltage characteristics comparison Pc-Ac - Li , [22] . . . 15
1.3 BAT-MAN Project 16

3.1 Double tank analogy - KiBaM model [11] 24
3.2 Radial Basis Function, *Ramraj Chandradevan* 25
3.3 Particle Swarm Optimization, [8] 26
3.4 Electrical equivalent circuit, [24] 27
3.5 Hybrid battery model, [27] 28
3.6 Estimation methods comparison 29
3.7 Model-based estimation technique, [27] 30
3.8 Predictive maintenance: classification algorithm, [30] 31
3.9 Open circuit voltage characteristic: single absolute characteristic with SoC=100% @ full charge 32

4.1 Energetic quantities - graphical representation 35
4.2 Energetic quantities (relative) - graphical representation 38

5.1 Experimental project example [19] 41
5.2 Experimental design process [19] 43
5.3 Runs table for full-factorial experiments, [19] 44
5.4 Acquisition test - 0 45
5.5 Acquisition test - 0, error trade-off 45
5.6 Acquisition test - 1 46
5.7 Acquisition test - 2 47
5.8 DoE Legend 48
5.9 Generic discharge profile 49
5.10 Absolute Interpretation: $V_{ocv}(SoC)$ characteristic interpolation 50
5.11 Generic electric equivalent model 51
5.12 R-model 53
5.13 Identification range - R-model 54
5.14 Identification results - R-model 55
5.15 Dynamic model 55
5.16 Identification range - Dynamic model 56
5.17 Identification results - Dynamic model 56
5.18 Identification results - τ_1 - Dynamic Model 57
5.19 Non-linear model 58
5.20 Identification range - Non linear model 59
5.21 Identification results - Non-linear model 60
5.22 Full BAT-MAN DoE 61

6.1 Equivalent electric circuit 63
6.2 Linear approximation - $V_{ocv}(SoC)$ characteristic interpolation 64
6.3 Convex combination parameters space 65
6.4 Voltage source model block 66
6.5 R_{nl} parabolic interpolation 67
6.6 R_{nl} - Convex combination parameters space 67
6.7 Non- linear Resistor model block 68
6.8 Simulink model - High Level 70
6.9 Model performance - Battery: new 72
6.10 Model performance - Battery: medium 72
6.11 Model performance - Battery: old 73

7.1 Extended Kalman Filter, [17] 75
7.2 Extended Kalman Filter - *Simulink* implementation 77
7.3 Extended Kalman Filter - Performance: New 78
7.4 Extended Kalman Filter - Performance: Medium 79
7.5 Extended Kalman Filter - Performance: Old 79
7.6 High level algorithm description 80
7.7 Series of n ANSE (Augmented Non-linear State Estimators 81
7.8 ANSE (Augmented Non-linear State Estimators) 82
7.9 Signal Conditioning 83
7.10 Comparator 84
7.11 Channel selection, battery: new 85
7.12 Internal state identification, battery: new 86
7.13 Modelling error: terminal voltage V_t, battery: new 87
7.14 Channel selection, battery: med 88
7.15 Internal state identification, battery: med 89
7.16 Modelling error: terminal voltage V_t, battery: med 90
7.17 Channel selection, battery: old 91
7.18 Internal state identification, battery: old 92
7.19 Modelling error: terminal voltage V_t, battery: old 93

A.1 Simulink Implementation: battery model 125
A.2 Simulink Implementation: Extended Kalman Filter 126
A.3 Simulink Implementation: ANSE 127
A.4 Simulink Implementation: Final solution 128

List of Tables

2.1 Li-Ion battery energetic specs 20
2.2 Lead Acid battery energetic specs 21

3.1 Feature Engineering process 30

5.1 Real capacity, FIAMM Titanium L150P, 50Ah, 12V 49
5.2 Open Circuit Voltage Characteristics Coefficients 50

6.1 V_{ocv} model block parameters 71
6.2 R_{nl} model block parameters 71

7.1 EKF parameters 78
7.2 Final Algorithm: initialisation values 84

Chapter 1

Introduction

1.1 The Problem

"Being able to provide, in real-time, an accurate estimate of the state of charge and health condition of a battery"

A battery, regardless of its technology, is a fairly complex electro-chemical device of which we can easily measure the terminal voltage V_t and the drawn current I. Considering the simplest electrical-equivalent model described in [7]:

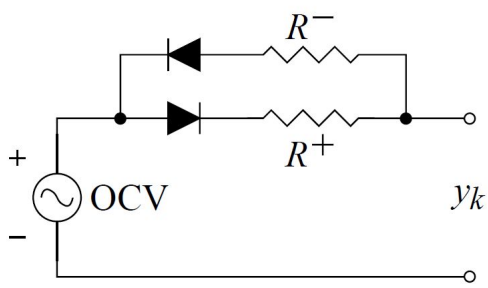


Figure 1.1: Basic battery model, [7]

we can see that it is made up of a voltage source, also called *open circuit voltage* $V_{ocv}(SoC, SoH, T, ...)$ that is strictly related to the energy stored and releasable, and an *internal dynamic impedance* Z_i that takes into account for ageing effects and chemical reactions. For the nature of the system, $V_t = V_{ocv}$ only when $I = 0$ for an amount of time sufficient to let the dynamics be extinguished. Therefore, as shown in [3], measuring state of charge by the terminal voltage V_t is simple, but it can be inaccurate because cell materials, internal impedance and temperature affect the voltage. The most blatant error of the voltage-based SoC measurement occurs when disturbing a battery with a charge or discharge. The resulting agitation distorts the terminal voltage and it no longer represents a correct SoC reference. To get accurate readings, **the battery needs to rest in the open circuit state for at least four hours**. This makes the voltage-based SoC method impractical for real-time application where a battery is in active duty. Moreover each battery chemistry delivers its own unique discharge signature $V_{ocv}(SoC)$. While voltage-based SoC works reasonably well for a lead acid battery that has rested, the flat

discharge curve of nickel- and lithium-based batteries renders the voltage method impracticable (Figure 1.2).

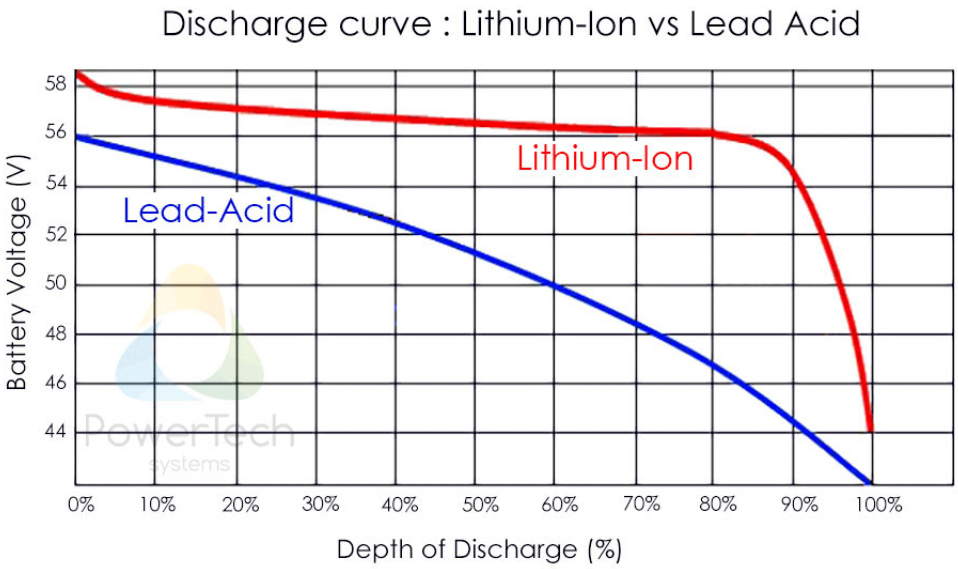


Figure 1.2: Open circuit voltage characteristics comparison Pc-Ac - Li , [22]

There are other several ways of measuring the SoC, such as detecting the specific gravity with an hydrometer, measuring the drawn energy with the coulomb counting and apply parasitic load with the impedance spectroscopy. However, some of them requires a large amount of time for the correct estimation, others discharge the battery and finally, like the coulomb counting, can be affected by reset and initialisation problems. It goes without saying that a state estimator algorithm would be implemented to face this problem.

In this work we'll see first and foremost how to mathematically describe the energy exchanges in an electrochemical device. Then we will create a suitable battery model and we will exploit this model in a model-based estimation algorithm for the joint estimation of the state of charge and state of health of a battery.

1.1.1 Key Trends, Market and Industry Forces

Energy storage is one of the main topic of the next century. Almost everything we use is powered by a battery: mobile-phones, laptops, pace-makers, toothbrushes and so on. According to the Global EV Outlook [9], electric mobility is expanding at a rapid pace. In 2018, the global electric car fleet exceeded 5.1 million, up 2 million from the previous year and almost doubling the number of new electric car sales. Technology advances are delivering substantial cost cuts. Key enablers are developments in battery chemistry and expansion of production capacity in manufacturing plants. In 2030, in the New Policies Scenario, which includes the impact of announced policy ambitions, global electric car sales reach 23 million and the stock exceeds 130 million vehicles. However, for this to happen, as shown in this McKinsey articles [26] and [4], access to charging infrastructure must improve. Although many BEVs are charged at home, public charging is necessary for owners who are travelling or if they don't own homes with garages. *"On-site battery storage*

at an electric -vehicle station can help smooth out load profiles, charging from the grid when no vehicles are present". Storage prices are dropping much faster than anyone expected, due to the growing market for consumer electronics and demand for EVs. Battery-pack costs are down to less than $230 per kilowatt-hour in 2016, compared with almost $1000 per kilowatt-hour in 2010. At today's lower prices, storage is starting to play a broader role in energy markets, moving from niche uses such as grid balancing to broader ones such as replacing conventional power generators for reliability, providing power-quality services, and supporting renewables integration.

1.2 The Project

The BAT-MAN project is part of the contibution *POR FESR 2014-2020 - Azione I.1b.1.2 Poli di Innovazione - Agenda strategica di Ricerca 2016*. (Figure 1.3)

Figure 1.3: BAT-MAN Project

1.2.1 Goal

The main objective of the BAT-MAN project is the realisation of a low-cost device that measures the state of charge *SoC* and the health conditions *SoH* of a battery, capable of recognising critical situations, process data and inform the user through a simple interface such as an app on a smartphone.

1.2.2 Stakeholders

Project Leader

brain Technologies S.r.l.
(`www.brain-tech.it`)

Partner

S.I.V.E. S.p.A.
(`www.siveonline.com`)

Subcontractor

Dipartimento di Elettronica – Laboratorio di Neuronica del Politecnico di Torino
(`https://neuronica.polito.it/`)

1.2.3 Costs

Total Cost: 805.675 €
Total Contribution: 410.231 €

Chapter 2

Batteries

In this chapter we will make a quick review about the most used battery technology (Lithium-Ion), the battery technology used in this work (Lead-Acid) and the future technology that we may expect (Solid Electrolite).

2.1 Actual Technology

2.1.1 Lithium-Ion

"There's Nothing Better Than Lithium-Ion Coming Soon"
David R. Baker, Bloomberg.com, 2019, [5].

A Lithium-Ion battery is made up of an anode, cathode, separator, electrolyte, and two current collectors (positive and negative). The anode and cathode store the lithium. The electrolyte carries positively charged lithium ions from the anode to the cathode and vice versa through the separator. The separator is a very thin sheet of microperforated plastic. As the name implies, it separates the positive and negative electrodes while allowing ions to pass through. The movement of the lithium ions creates free electrons in the anode which creates a charge at the positive current collector. The electrical current then flows from the current collector through a load to the negative current collector. While the battery is discharging and providing an electric current, the anode made of carbon releases lithium ions to the cathode made of Lithium cobalt oxide, or $LiCoO_2$, generating a flow of electrons from one side to the other. In other words, the anode undergoes oxidation, or loss of electrons, and the cathode sees a reduction, or a gain of electrons. When plugging in the device, the opposite happens: Lithium ions are released by the cathode and received by the anode. Li-ion can be considered a low-maintenance battery, an advantage that most other chemistries cannot claim. The battery has no memory and does not need exercising (deliberate full discharge) to keep it in good shape. Self-discharge is less than half that of nickel-based systems and this helps the fuel gauge applications. Average rated values showed in Table 2.1. Such a high specific values (up to 6 time higher than Lead-Acid battery) have made this technology suitable for several field of application, going from mobile devices to electric vehicles. The main drawback concern safety aspects: if the battery gets hot enough to ignite the electrolyte, you are going to get a fire. Moreover, for what concern modelling aspects, the open circuit voltage curve $V_{ocv}(SoC)$ is quite horizontal, making voltage-based SoC

Parameter	Value
Nominal voltage $[V/cell]$	3.6
Specific Energy $[Wh/kg]$	150 - 200
Specific Power $[W/kg]$	300 - 1500

Table 2.1: Li-Ion battery energetic specs

measuring method inapplicable and last but not least, Li-Ion batteries are quite expensive. For these reasons, we decided to use in our work a safer technology, nonetheless bearing in mind the chance to extend the result to any kind of batteries.

Content Sources

David R. Baker, Bloomberg.com, [5];
Battery University, [2];
Wikipedia, [33];
Marshall Brain, Howstuffworks, [13];
Energy.gov, [6];

2.1.2 Lead Acid

The electrical energy produced by a discharging lead–acid battery can be attributed to the energy released when the strong chemical bonds of water (H_2O) molecules are formed from H^+ ions of the acid and O^{2-} ions of PbO_2. Conversely, during charging the battery acts as a water-splitting device, and in the charged state the chemical energy of the battery is stored in the potential difference between the pure lead at the negative side and the PbO_2 on the positive side, plus the Sulphuric Acid in aqueous condition. In the discharged state both the positive and negative plates become lead(II) sulfate $PbSO_4$, and the electrolyte loses much of its dissolved sulfuric acid and becomes primarily water. The discharge process is driven by the pronounced reduction in energy when $2H^+$(aq) (hydrated protons) of the acid react with O^{2-} ions of PbO_2 to form the strong $O-H$ bonds in H_2O. This highly exergonic process also compensates for the energetically unfavorable formation of Pb^{2+}(aq) ions or lead sulfate ($PbSO_4$(s)). Thanks to its ability of withstand high current discharges, this technology is widely used for engine crank. Starter batteries are rated with Ah or RS (reserve capacity) to indicate energy storage capability, as well as CCA (cold cranking amps) to signify the current a battery can deliver at cold temperature. SAE J537 specifies 30 seconds of discharge at –18°C (0°F) at the rated CCA ampere without the battery voltage dropping below 7.2 volts. RC reflects the runtime in minutes at a steady discharge of 25. Lead acid does not lend itself to fast charging and with most types, a full charge takes 14–16 hours. The battery must always be stored at full state-of-charge. Low charge causes sulfation, a condition that robs the battery of performance. Adding carbon on the negative electrode reduces this problem but this lowers the specific energy. Lead acid is heavy and is less durable than nickel- and lithium-based systems when deep cycled. A full discharge causes strain and each discharge/charge cycle permanently robs the battery of a small amount of capacity. This loss is small while the battery is in good operating

condition, but the fading increases once the performance drops to half the nominal capacity.

Parameter	Value
Nominal voltage $[V/cell]$	2.1
Specific Energy $[Wh/kg]$	35 - 40
Specific Power $[W/kg]$	180

Table 2.2: Lead Acid battery energetic specs

Content Sources

Battery University, [1];
Wikipedia, [32];

2.2 Future Technology

2.2.1 Solid Electrolite

"With solid electrolytes, we can realise lithium metal instead of graphite-based anodes",
Dr. Johannes Kasnatscheew, Electrive interview [21]

Battery cells with a solid electrolyte promise high energy densities, in fact the introduction of solid electrolytes could increase gravimetric energy density by 40%, and volumetric energy density by 70%, Dr. Johannes Kasnatscheew of Forschungszentrum Jülich explains in the interview with *electrive*. Reasonable conductivity, high mechanical robustness, but very high contact resistances during charging and discharging characterise **inorganic** solid electrolytes. The current flowing is still too low. **Organic** solid electrolytes, on the other hand, have less contact resistance, but low conductivity. At congresses, scientists continue to discuss the suitability of compounds. At present, sulfid-based inorganic ceramic solid electrolytes are the favourite in terms of conductivity. Solid electrolyte would also drastically improve the use of energy in production and thus the CO2 balance: today, drying is a complex and energy-intensive process. This would at least be superfluous on the anode side when using solid electrolytes because the foil is from lithium metal. This also reduces toxicity. So far, however, there has been no sample of solid electrolyte batteries that could beat current products in terms of their properties. Besides, it first would need to be shown, how pure lithium metal anodes could be produced safely and in mass.

Content Sources

Nora Manthey, electrive.com, [21];

Chapter 3

State of Art

3.1 Battery Modelling

The main approaches used to model battery behavior are:

- Analytical modelling of electrochemical phenomena
- Data-Driven modelling (Black-Box)
- Physical modelling (Electrical equivalent circuit)
- Hybrid modelling

Each method is able to characterise a different detail level of reality, an aspect that directly reflects on the required computational effort. It is therefore necessary to find an optimal compromise taking into account the specific application and the purpose for which it is going to be used. The minimum complexity requirements of the models concern the ability to derive an estimate of the voltage to the battery terminals (V_t), managing and providing the state of charge (SoC) and possibly some measure of the health status (SoH, which takes into account the aging of the battery). Below we will review the main modelling techniques highlighting the key aspects that distinguish them.

3.1.1 Analytical modelling of electrochemical phenomena

This type of technique allows to describe the macroscopic behaviour of the battery by solving complex nonlinear systems of differential equations that characterise the molecular behaviour of the chemical reactions underlying the energy production process. A model of this complexity requires a big experimental effort to identify static parameters, as well as high computing power. The main disadvantage of this approach is that very few applications guarantee time and resources useful for obtaining the necessary parameters and coefficients in real time, not to mention that some of them, such as the specific heat of the electrolyte, are difficult to determine for sealed lead-acid batteries. Moreover, the high level of detail obliges not to neglect the phenomenon of overfitting, which minimizes the robustness of the algorithm and therefore the ability to adapt the model to the entire space of inference.

Kinetic battery model

Among the analytical models, KiBaM is undoubtedly the most widely used since it is based on a simple concept that can be described by the *double tank analogy* (Figure 3.1), it only requires the use of three parameters to model the battery behaviour and has a good precision with regard to the estimate of the battery duration, with average relative error between 2% and 4%. The system of differential

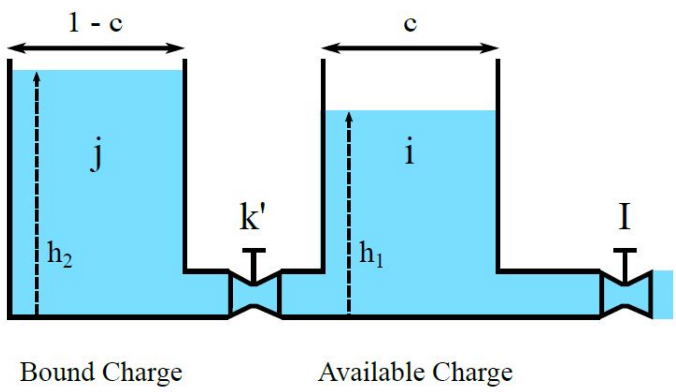


Figure 3.1: Double tank analogy - KiBaM model [11]

equations describing the KiBaM model is the following:

$$\begin{cases} \frac{dC_a}{dt} = -I + k\big(h_2 - h_1\big) \\ \frac{dC_b}{dt} = -k\big(h_2 - h_1\big) \end{cases}$$

where C_a is the available capacity that can be immediately supplied to a load, C_b is the limited capacity that can flow towards the available capacity, regulated by a 'valve' with fixed conductance k and the level indicators h_1 and h_2 are closely linked to the nominal capacity value C_n according to the following relations:

$$h_1 = \frac{C_a}{C_n}$$
$$h_2 = \frac{C_b}{\big(1 - C_n\big)}$$

An extension of this model, called T−KiBaM, introduces the concept of chemical kinetics to model the influence of temperature on the chemical reactions that take place inside the battery exploiting the Arrhenius equation:

$$k' = A\, e^{\left(-\frac{E_a}{R\,T}\right)}$$

and where:

$$k' = \frac{k}{C_n\big(1 - C_n\big)}$$

This model has the great advantage of being able to model two very important effects such as the capacity variation in function of the discharge current intensity (C-rate) and the charge recovery in the inactivity periods due to electro-chemical stabilisation.

Content Sources

Leonardo M. Rodrigues, Carlos Montez, Ricardo Moraes, Paulo Portugal and Francisco Vasques, [12];
Ingemar Kaj, Victorien Konanéb, [10];
NALIN A. CHATURVEDI, REINHARDT KLEIN, JAKE CHRISTENSEN, JASIM AHMED, and ALEKSANDAR KOJIC, [20];
J.F. Manwell, J.G. McGowan, [11];

3.1.2 Data-Driven modelling (Black-Box)

This type of technique circumvents the need for a complete understanding of the complicated physical processes, looking for a data based heuristic approach (Data-Driven) to predict behaviours and battery conditions. To this end, computational intelligence techniques such as Neural Networks, Particle Swarm Optimization and Predictive Maintenance can be applied to experimental data in order to generate a non-linear function able to describe the battery variables of interest.

Neural Networks

In modelling methods with Artificial Neural Networks (ANNs), usually the State of Charge (SoC) is defined as an independent state variable and it is modelled by means of a *Radial Basis Function* (Figure 3.2). Thanks to the Neural Networks

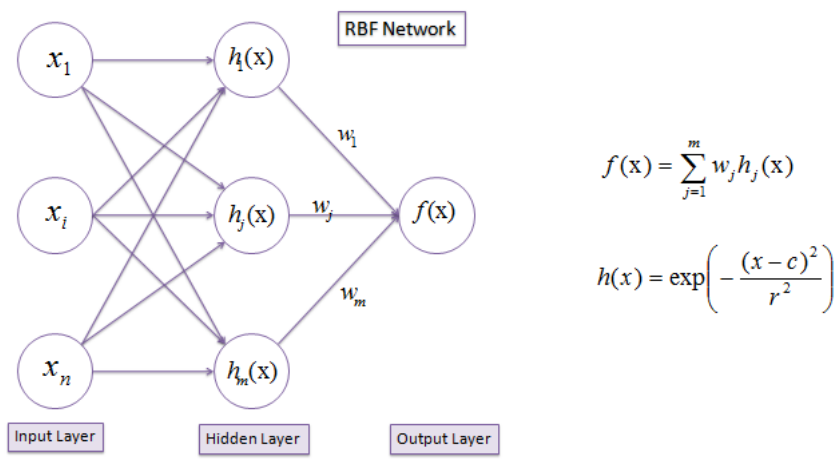


Figure 3.2: Radial Basis Function, *Ramraj Chandradevan*

strong ability to approximate any Non-linear function, the network receives in input:

- terminal voltage at the previous time instant: $V_t(k-1)$
- the actual state of charge: $SoC(k)$
- the actual current: $I(k)$

and by means of back-propagation and least mean square algorithms (training phase)

- the activation function weights of neurons inside the hidden layer

- the radial basis function parameters

are chosen in a way that that a cost function is minimised, yielding:

- the terminal voltage at the next time instant: $V_t(k)$

This technology requires to be trained on a copious amount of experimental data. The acquisition and the selection of this training data is generally the most difficult process in using a NN. Once the training is completed, the performance in terms of robustness outside and inside his training set may vary depending on the design (number of layers and neurons, local weighting function, etc.).

Content Sources

Mohammad Charkhgard and Mohammad Farrokhi, [18];

Particle Swarm Optimization

The PSO is an advanced computational identification technique capable of searching for optimal solutions to:

- non-linear continuous functions
- constrained and unconstrained functions
- multimodal non-differentiable functions

This technique was born from the intuition of the needs of producing a computational intelligence able to represent social interactions as can be that of a flock of birds in search of corn and is able to optimize a problem moving iteratively the position and velocity of the particles within the search space. In each iteration in fact, each particle updates its position and speed in accordance with the own previous optimal solution in order to converge towards the best global solution(Figure 3.3). This type

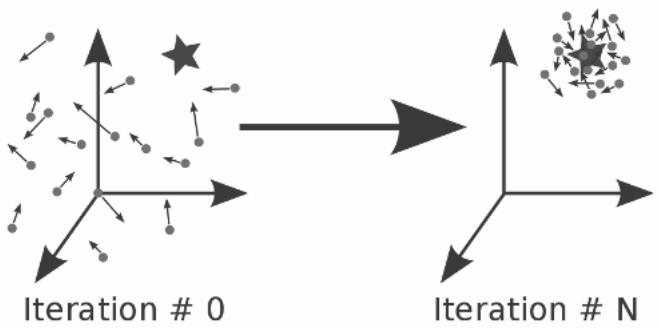


Figure 3.3: Particle Swarm Optimization, [8]

of parametric optimization suffers from local minimum problems and it is therefore necessary to initialise the variables with appropriate values.

Content Sources

Huang Kai, Guo Yong-Fang, Li Zhi-Gang, Lin Hsiung-Cheng and Li Ling-Ling, [8];

3.1.3 Physical modelling (Electrical equivalent circuit)

By means of an electrical equivalent circuit it is possible to empirically approximate the macroscopic behaviour of the physical quantities that are observable from the battery terminals such as current and voltage. Although in this method the cells-chemistry it is not directly described, it is still essential to have a deep knowledge in order to succeed in the description and management of the non-linearity deriving from the experimental tests. The general equivalent circuit scheme is shown in Figure 3.4. The voltage source, denoted by E_m, simulates the open circuit voltage,

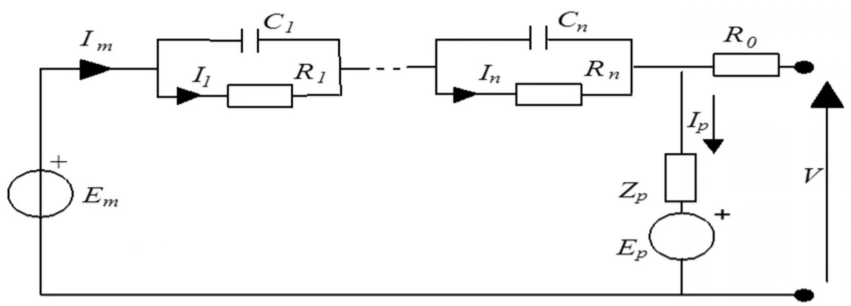


Figure 3.4: Electrical equivalent circuit, [24]

also previously defined as V_{ocv}. It consists of a function $V_{ocv}(SoC)$ experimentally determined with appropriate laboratory tests. At the right top, with the symbol R_0, the internal resistance of the battery is modeled. In the middle, instead, there is a network consisting of one or more $RC - branches$ that take into account the dynamics in terms of time constants and frequency response. To consider the chemical processes that give rise to phenomena like self-discharge or leakage due to eddy currents, a vertical branch is inserted in parallel to the terminals that consist of a generic impedance Z_p and a voltage source E_p. Let's bear in mind that all the parameters of the equivalent circuit are dependent on:

- State of charge: SoC
- State of health: SoH
- Discharge current intensity: $C - rate$
- Temperature
- Battery type: Lead-Acid, li-Ion, Ni-Mh...

At the same time, satisfactory results can be obtained for most real-time applications by neglecting the vertical branch of the parasitic effects and using at most two RC branches to avoid hypoerparametrization. In the **Laplace domain** we can then get the transfer function between the internal impedance voltage $\big(V_{ocv}(s) - Vt(s)\big)$ and the input $I(s)$ as:

$$H(s) = \frac{V_{ocv}(SoC, s) - V_t(s)}{I(s)} = R_0 + \frac{R_1}{s\,C_1} \cdot \frac{1}{\left(R_1 + \frac{1}{s\,C_1}\right)} + ... + \frac{R_n}{s\,C_n} \cdot \frac{1}{\left(R_n + \frac{1}{s\,C_n}\right)}$$

As we will see in chapter 5, you can pass through the **Z-domain** (remember the strong dependency on the sampling time T_s) and eventually convert everything in the **discrete time domain** to perform the parameters identification.

Content Sources

Tarun Huria, Massimo Ceraolo, Javier Gazzarri and Robyn Jackey, [28];
Robyn A. Jackey, [23];
Robyn Jackey, Aubrey da Cunha, Javier Gazzarri, [24];

3.1.4 Ibrid modelling

The hybrid battery model consists of an *Enhanced Coulomb counting* and an *Electric equivalent model*. The first it is based on the KiBaM electrochemical model described extensively above and it is used to estimate the *SoC* of the battery in a robust manner. In this way, effects such as non-linear variation in capacity and voltage self-recovery are modelled too. The latter comes with a double *RC* network to reproduce a wide spectrum of dynamics. In Figure 3.5, the descriptive scheme.

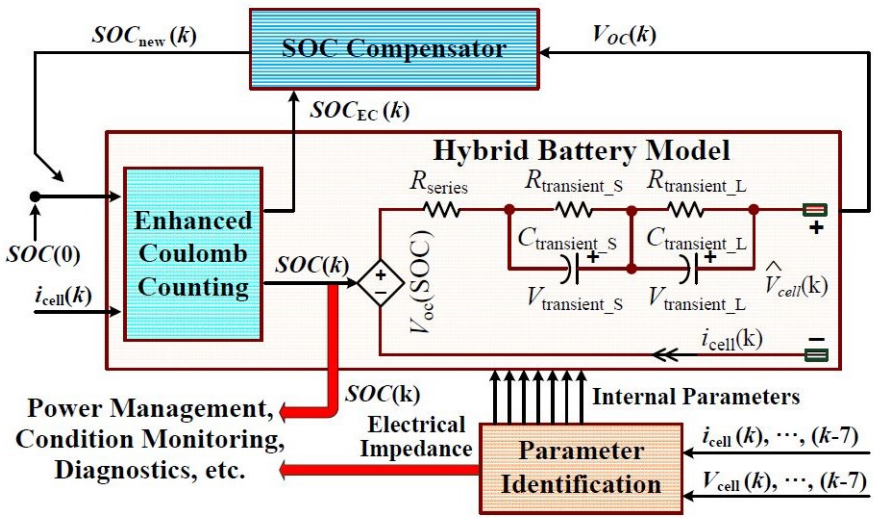


Figure 3.5: Hybrid battery model, [27]

Content Sources

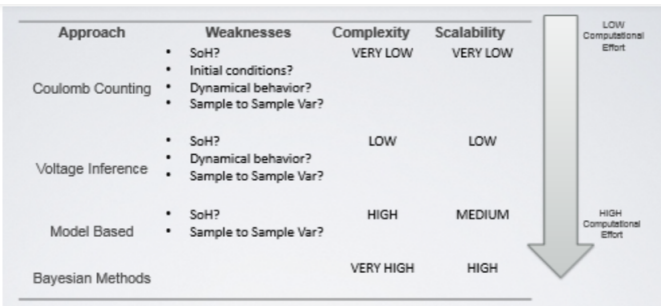
Taesic Kim, Wei Qiao and Liyan Qu, [27];

3.2 States Estimation Technology

In the literature, the state of charge *SoC* is defined as the ratio expressed as a percentage of the energy still available compared to the initial amount of energy. This quantity can be seen as the equivalent of the petrol level indicator in the tank of cars. As shown in [3], there are several ways to measure the SoC:

- Voltage based method: experimental $V_{ocv}(SoC)$ curve
- Hydrometer: specific weight measurement
- Coulomb Counting: $SoC(t) = 1 - \frac{\int_0^t I(\tau)d\tau}{C_n}$
- Impedance Spectroscopy: EIS (Electro-Chemical Impedance Spectroscopy)

Each of these methods, taken individually, is unsuitable for a real-time estimate of the State of Charge. In the one hand, some of them often require ad-hoc experiments or particular working conditions, while on the other some work by exciting the battery or, even worse, they need the battery to be removed from the vehicle. This is the reason why today the model-based estimation solutions are the most used and we can se pros and cons in Figure 3.6.



Approach	Weaknesses	Complexity	Scalability
Coulomb Counting	• SoH? • Initial conditions? • Dynamical behavior? • Sample to Sample Var?	VERY LOW	VERY LOW
Voltage Inference	• SoH? • Dynamical behavior? • Sample to Sample Var?	LOW	LOW
Model Based	• SoH? • Sample to Sample Var?	HIGH	MEDIUM
Bayesian Methods		VERY HIGH	HIGH

Figure 3.6: Estimation methods comparison

3.2.1 Model-Based

This technique allows for the SoC estimation by means of model-based algorithms that exploit the models described in section 3.1, to the simultaneous estimate of both:

- the electric equivalent model parameters
- the state variables

based on real-time input and output observations. In this context the state of charge is promoted to state variable and the observer must therefore solve a problem of estimating the state of a discrete nonlinear dynamic system of the type:

$$\begin{aligned} \mathbf{x}_{k+1} &= f(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{v}_k \\ \mathbf{y}_k &= h(\mathbf{x}_k, \mathbf{u}_k) + \mathbf{n}_k \end{aligned}$$

where $\mathbf{x}$ represent the non-measurable states vector (SoC and $V_p = V_{ocv} - V_t$), $\mathbf{u}$ is the input of the system (current $I[A]$), $\mathbf{v}$ is the process white noise, $\mathbf{n}$ is the measurement white noise and $\mathbf{y}$ is the output of the system. One of the best algorithm for the non-linear system state estimation is **Adaptive extended Kalman Filter - AEKF**, that comparing the real with the estimated output and exploiting the Bayesian theory, allows the internal state estimation instant by instant. This estimate is then eventually used by a **Recursive Least Squares - RLS** algorithm for the on-line model parameters tuning. In Figure 3.7 the working diagram of this best practice.

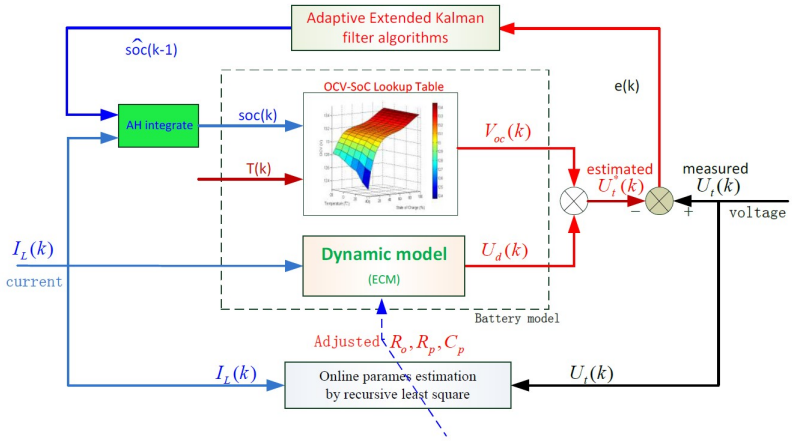


Figure 3.7: Model-based estimation technique, [27]

Content Sources

Taesic Kim, Wei Qiao and Liyan Qu, [27];

3.2.2 Data-Driven

Data-driven modelling is a fairly recent tool that is based on the Feature Engineering theory. The workflow (Table 3.1) starts with the definition and understanding of the problem to which is associated with an important experimental campaign for the acquisition and generation of useful data. These data are then selected, cleaned, transformed and enriched through appropriate processes. When the database is ready you can apply regression, classification and grouping techniques to generate a mathematical model capable of receiving input variables or characteristics and supplying output, such as state of health or charge of a battery.

Problem	- definition - comprehension - hypothesis formulation
Data generation	- specific experimental campaign - preliminary data evaluation
Data Management	- selection - cleaning - transformation - enrichment
Modelling	- features - selection and choice - tuning - test and validation

Table 3.1: Feature Engineering process

Predictive Maintenance

Predictive analytics is the engine of evidence-based decision making. Today there are many opportunities that big data and engineering techniques are bringing to the world of analytics. Predictive maintenance consists in the intelligent device condition monitoring in order to avoid future failures or foresee undesired failures. Predictive models are generated by machine learning algorithms properly identified by analysing large data series (Figure 3.8). For this purpose the data are in fact processed and analysed with the aim of extracting, transforming and selecting peculiar characteristics able to characterise certain operating conditions of the battery and therefore predicting future behaviour. Thanks to tools like semantic segmentation it is possible to automate the process of characterisation of the phenomenon. In

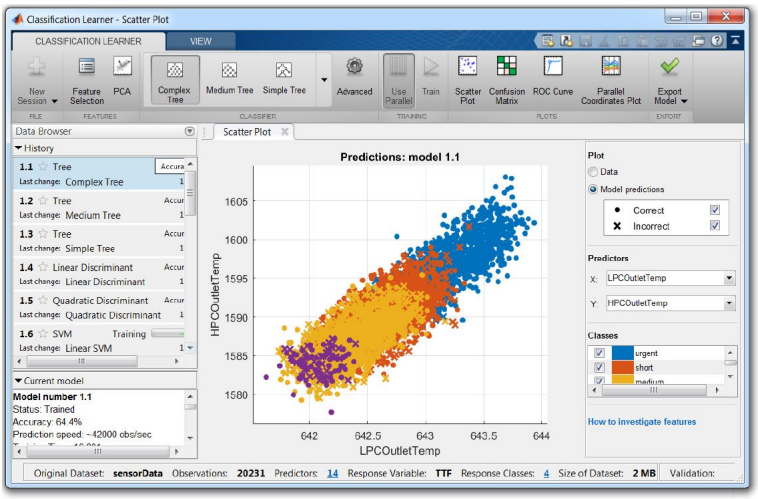


Figure 3.8: Predictive maintenance: classification algorithm, [30]

contrast to preventive maintenance, which follows a set timeline, as we have seen predictive maintenance schedules are determined by analytic algorithms and data from equipment sensors.

Content Sources

The Mathworks, Inc., [31];
The Mathworks, Inc., [30];
The Mathworks, Inc., [29];

3.3 Critiques to the State of Art

Although several sophisticated techniques have already been implemented, during the study of the problem and more, during the data interpretation in the identification section (chapter 5) many doubts and incoherence still raised. For instance, let's imagine two batteries. One is completely new, the latter is almost dead:

- battery n°1: new
- battery n°2: old

After a charging time long enough to consider both fully charged (constant voltage threshold reached), what can we say about the charge condition? They are both fully charged and therefore should they have a $SoC = 100\%$? If so, why can I only extract less than half of the nominal capacity from battery number 2, meaning that it could be exhausted with a $SoC = 60\%$ while the battery n°1 could be considered exhausted at $SoC = 0\%$. Does SoC has still meaning? Should I assign to each fully charged battery a different SoC according to it's "ageing"? Let's have a simple look at one evidence in the lack of strictness. In the following we are going to see the experimental open circuit voltage characteristics $V_{ocv}(SoC)$, where:

- green dots: new battery
- blue dots: medium battery
- red dots: old battery

Let's consider the case in which, when the battery is fully charged we always assign it the $SoC = 100\%$ regardless of its real capacity C_{real} and we use a single absolute characteristic: Figure 3.9. As we can clearly see, except for the new battery, we are

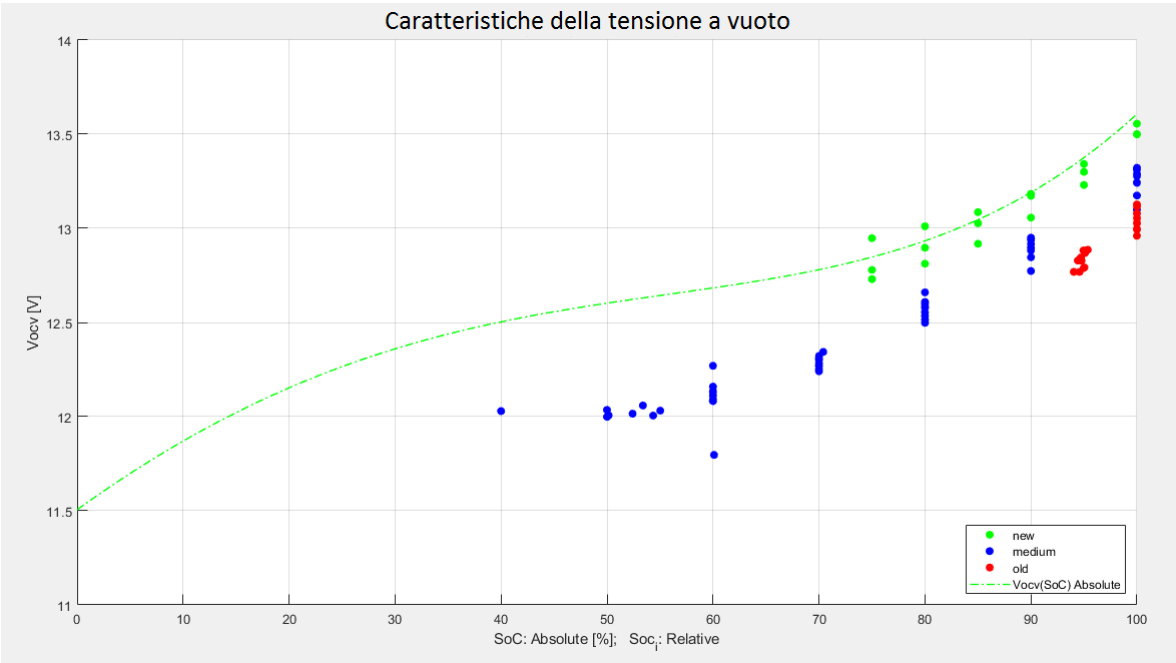


Figure 3.9: Open circuit voltage characteristic: single absolute characteristic with SoC=100% @ full charge

always using the wrong characteristics. *Should we use more than one characteristic? Should we assign a different SoC @ full-charge? etc.*
The aim of this work is exactly to find a rigorous framework in which building a solid solution.

Chapter 4

Energetic Framework

On the basis of the aforementioned doubts about the batteries energetic state description, I went trough [14] and I've tried to enhanced it, with the aim of ending up in a complete and rigorous energetic framework for batteries.

4.1 Definitions

In this section we are going to analyse, define and explain in details the batteries fundamental energetic quantities.

4.1.1 Capacity

Nominal Capacity

It's the ideal value of electric energy potentially deliverable by the battery in standard conditions, expressed in Ah and given as rated value by the manufacturer.

$$C_n$$

Real Capacity

It corresponds to the actual deliverable electrical energy, measured starting from a complete charge and going all the way down to the minimum voltage threshold (1.75 V/cell for Pb-Ac). In general:

$$C_{real} \leq C_n$$

The real capacity value is mainly affected by ageing SoH (state of health), temperature T and discharge rate $c\text{-}rate$ (intensity of discharge).

$$C_{real}\big(SoH, T, I\big)$$

However, along a **single discharge cycle**, ageing effects can be seen as an almost constant factor ($SoH \approx const.$) and for the sake of clarity I'm going to omit the dependency to temperature and discharge intensity since it's a level of detail not necessary in this section.

$$C_{real}(SoH) \approx const.$$

Hence, considering a generic time instant t in a single discharge cycle, the real capacity C_{real} can be expressed as a sum of two complementary terms:

$$C_{real}(SoH) = C_{released}(t) + C_{releasable}(t, SoH) \tag{4.1}$$

where:

$$C_{released}(t) = C_{released}(t_0) + \int_{t_0}^{t} I(\tau)d\tau$$
$$C_{releasable}(t) = \int_{t}^{t_f} I(\tau)d\tau$$

with:
t_f: end discharge instant ($V_t = V_{t\_min}$)
t_0: first data registration instant
t : generic time instant

Lost Capacity

It's the quantity of energy that, due to electrochemical ageing factors, is no longer available from the battery. It corresponds to the exact difference between nominal and real capacity:

$$C_{lost}(SoH) = C_n - C_{real}(SoH) \tag{4.2}$$

Graphical Description

Let's consider a discharge session starting from full-charge, in a generic time instant t, the graphical capacity representation is shown in Figure 4.1.

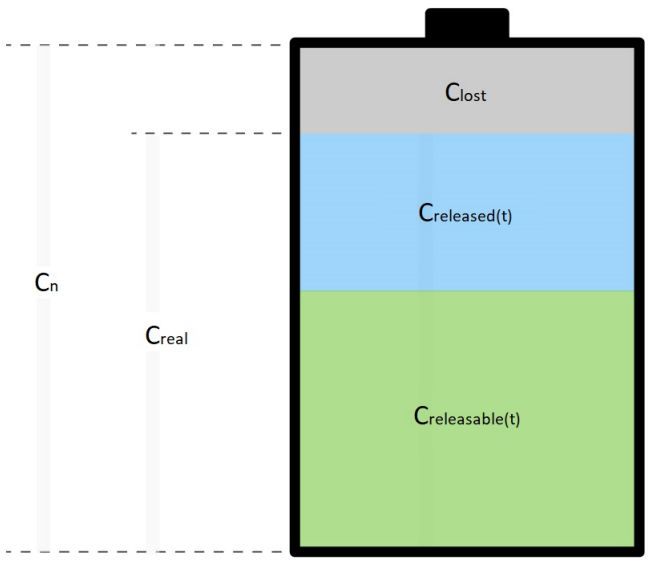


Figure 4.1: Energetic quantities - graphical representation

4.1.2 State of Charge

The State of Charge SoC is the level of **releasable** charge of an electric battery relative to its nominal capacity.

$$SoC(t) = SoC(t_0) - \frac{\int_{t_0}^{t} I(\tau)d\tau}{C_n} \tag{4.3}$$

that in energetic terms corresponds to:

$$SoC(t) = \frac{C_{releasable}(t_0)}{C_n} - \frac{\Big(C_{released}(t) - C_{released}(t_0)\Big)}{C_n}$$

4.1.3 Depth of Discharge

The Depth of Discharge DoD is the level of **released** charge of an electric battery relative to its nominal capacity.

$$DoD(t) = DoD(t_0) + \frac{\int_{t_0}^{t} I(\tau)d\tau}{C_n} \tag{4.4}$$

that in energetic terms corresponds to:

$$DoD(t) = \frac{C_{released}(t_0)}{C_n} + \frac{\Big(C_{released}(t) - C_{released}(t_0)\Big)}{C_n} = \frac{C_{released}(t)}{C_n}$$

4.1.4 State of Health

The State of Health is the measure of the batteries ageing and can analytically interpreted as:

$$SoH = DoD(t) + SoC(t) = \frac{C_{real}}{C_n} \tag{4.5}$$

demonstration:

$$SoH = DoD(t) + SoC(t) = \frac{C_{released}(t_0)}{C_n} + \frac{\Big(C_{released}(t) - C_{released}(t_0)\Big)}{C_n} + ...$$
$$+\frac{C_{releasable}(t_0)}{C_n} - \frac{\Big(C_{released}(t) - C_{released}(t_0)\Big)}{C_n} = \frac{C_{released}(t_0)}{C_n} + \frac{C_{releasable}(t_0)}{C_n} = \frac{C_{real}}{C_n}$$

4.2 Measurable Quantity Analysis

The theoretical framework just described is the basis on which we can build all our reasoning. In this section we will discuss about the unknown quantities, the ways of measuring them and the potential interpretation ways.

4.2.1 Absolute approach

At a generic time instant t we have:

$$DoD(t) = \frac{C_{released}(t_0)}{C_n} + \frac{\Big(C_{released}(t) - C_{released}(t_0)\Big)}{C_n} = \frac{C_{released}(t)}{C_n}$$
$$SoC(t) = \frac{C_{releasable}(t_0)}{C_n} - \frac{\Big(C_{released}(t) - C_{released}(t_0)\Big)}{C_n}$$
$$SoH = DoD(t) + SoC(t) = \frac{C_{real}}{C_n}$$

in which many of the aforementioned variables are unknown in general such as: $C_{released}(t_0)$, C_{real} and $C_{releasable}(t_0)$. Let's now introduce the condition for which at $t = t_0$ the battery is at **full charge**. This implies:

$$C_{released}(t_0) = 0$$
$$C_{released}(t) = \int_{t_0}^{t} I(\tau)d\tau$$
$$C_{releasable}(t_0) = C_{real}$$

yielding to:

$$DoD(t) = \frac{C_{released}(t)}{C_n}$$
$$SoC(t) = \frac{C_{real} - C_{released}(t)}{C_n}$$
$$SoH = DoD(t) + SoC(t) = \frac{C_{real}}{C_n}$$

and as we can see the unique left unknown is the real capacity C_{real}, that in the experimental phase will be suitably retrieved (chapter 5) and in the final algorithm phase (chapter 7) it appears to be the aim of the project.

4.2.2 Relative approach: C_{real} ignorance

We can change the point of view exploiting some "ignorance" assumption, switching to what we call: relative approach. The basic assumption behind the interpretation is the one of considering **unitary state of charge** every time the battery is in the condition of **full charge**. That means that for t=t_0, the $SoC(t_0)$=1 always with the battery fully charged, yielding to:

$$DoD(t) = \frac{C_{released}(t)}{C_n}$$
$$SoC(t) = 1 - \frac{C_{released}(t)}{C_n}$$
$$(SoH = 1)$$

Despite still working in theory, the major drawback of this method is that an old battery can reach, for instance, the the minimum voltage with a state of charge

equal to 70% or even more. This sounds quite strange because how can a battery be exhausted and still have a SoC so high? In this approach, in fact, the SoC looses it's physical meaning and in Figure 4.2 we can see the energetic graphical description. If on the one hand the advantage of this interpretation is that all the quantities

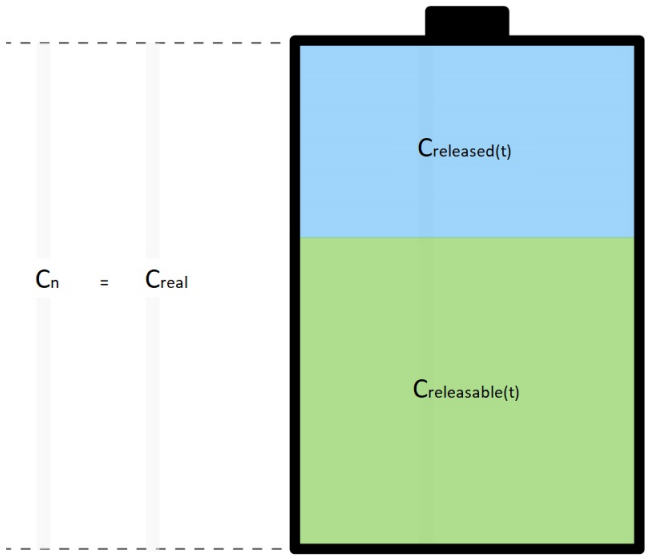


Figure 4.2: Energetic quantities (relative) - graphical representation

are known, on the other it's necessary a voltage-based method to determine the batteries' conditions.

4.2.3 Relative approach: relative SoC

Let's now keep the assumption for which at full charge the battery reaches the unitary state of charge ($SoC(t_0)$=1), but in this case we are now introducing the **relative State of Charge** defined as:

$$SoC_r(t) = SoC_r(t_0) - \frac{\int_{t_0}^{t} I(\tau)d\tau}{C_{real}} \tag{4.6}$$

and respectively, the **relative Depth of Discharge**:

$$DoD_r(t) = DoD_r(t_0) + \frac{\int_{t_0}^{t} I(\tau)d\tau}{C_{real}} \tag{4.7}$$

Exploting the hypothesis for which at $t = t_0$ the battery is fully charged, we get:

$$DoD_r(t) = \frac{C_{released}(t)}{C_{real}}$$
$$SoC_r(t) = 1 - \frac{C_{released}(t)}{C_{real}}$$
$$(SoH = 1)$$

The state of charge regains its physical meaning, spanning the range [0,1] and again the problem is shifted in the determination of the unknown parameters C_{real}.

Chapter 5

Model Identification

5.1 Design of Experiment

In this section we'll see what is the design of the experiment and what has been the shape of the *DoE* for our project.

5.1.1 DoE theory

Introduction

The term experiment is defined as a systematic procedure performed under controlled conditions in order to discover an unknown effect, verify a hypothesis or illustrate a known effect. When analysing a process, experiments are often used to assess which process inputs have a significant impact on the output and what should be the reference threshold of these inputs to obtain the desired result (output). Many experiments can be designed to collect this information in different ways. The Design of Experiments (DoE) is also called Experimental Design. The Design of Experiment technique lowers design costs and accelerates the process either reducing the late design changes or the materials and work complexity. The 'designed experiments' are also powerful tools to achieve savings on production costs by minimising the variation of process, rework, scrap and the need for inspection.

Set-up and Description

To better understand this powerful technology, general knowledge of statistics and analysis is required, such as histograms, statistical process control, regressions, correlations and so on. There are three aspects of the process that are analysed by a designed experiment:

- *Process Inputs (Factors)*
  The factors can be classified as controllable or uncontrollable variables. The controllable variables include everything we can predict, choose or modify a-priori. The uncontrollable variables concern the effects of 'noise' such as the effect of man or exogenous factors, which in an almost unpredictable way cause variability of normal operating conditions. In the design of the experiment only the controllable factors will be considered, while those uncontrollable will be managed by means of randomisation techniques able to highlight and isolate

the effects. Potential factors can be identified and classified using the Fishbone Chart (diagram cause-effect).

- *Levels*
  The levels are the discrete values that a factor can assume.

- *Process Outputs*
  The outputs are the measurable results potentially influenced by the factors and their respective levels. Experimenters often want to avoid optimising the process for one answer at the expense of another. For this reason, important results are measured and analysed in order to determine the factors, and factors related settings, able to provide the best overall result based on the assessments in terms of quality of measurable variables and evaluable attributes.

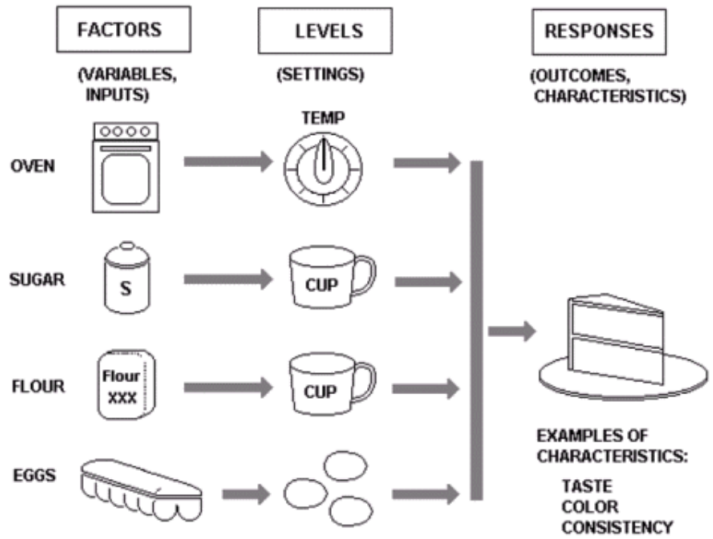


Figure 5.1: Experimental project example [19]

Experimentation Purpose

The designed experiments can find numerous potential applications for process and products improvement, including:

- *Alternatives comparison*
  In the case of the pastry example in Figure 5.1, we may want to compare the results of two different types of flour. If it turns out that the flour, coming from different suppliers, is not a significant factor, we could select the cheapest supplier. If instead the flour was significant, then we would select the best flour. Experiments should make it possible to make a well-considered decision that evaluates both quality and costs.

- *Identification of significant inputs (factors) that influence an output*
  "What are the significant factors besides flour, eggs, sugar and cooking?"

- *Achieving optimal process output*
  "What are the necessary factors and what are the levels of these factors, to obtain the desired response?"

- *Reduction of variability*
  Is it possible to choose the factors that can guarantee the best reproducibility?

- *Minimization, maximization or targeting of an output*
  "How can you make the cake as moist as possible without disintegrating?"

- *Improvement of Robustness - suitability for use in variable conditions*
  "Let's condider the factors and their levels (recipe), can they be modified so that the cake can come out almost similar regardless of the type of used oven?"

- *Balance and optimisation of output quality according to the critical characteristics (CTQC)*
  "How do you produce the best cake with the simplest recipe (the least number of ingredients) and the shorter cooking time?"

Experiment design guidelines

The design of an experiment addresses the questions above outlined by stipulating the following:

- The factors to be tested
- The levels of these factors
- The structure and layout of experimental tests or boundary conditions

A well-designed experiment is as simple as possible and must get the information requested in a cheap and reproducible way. Statistical process control, so to obtain reliable experimental results, is based on two conditions:

- a precise measurement system
- a stable process

If the measurement system contributes to an excessive error, the results of the experiment will be confused. Before conducting the experiment it is necessary to evaluate both the measurement system and the statistical stability the created process. The variation that affects the response must be limited to the random error of the common cause, not to the variation of the special cause deriving from a specific events. In addition to the measurement error (explained above), other sources of error or unexplained changes may obscure the results. Note that the term *error* is not a synonym for *errors*. The error refers to all unexplained variations that arise in the repeated execution of an experiment carried out by fixing the level settings and structure. Properly designed experiments can identify and quantify the error sources. Uncontrollable factors, which induce variations under normal operating conditions, are referred to as *noise factors*. These factors can be incorporated into the experiment so that their variation it is not found in the experiment error. A strength of the designed experiments is the ability of determining factors and settings that minimise the effects of uncontrollable factors. Be careful:

"*Correlation* can often be confused with *Causality*"

Two factors that vary together can be highly correlated without one causing the other, or both can be caused by a third factor. The above highlights the importance of a deep understanding of the operational dynamics during the design of an experiment. Brainstorming exercises and cause/effect diagrams are both excellent techniques for acquiring this operational knowledge during the design phase. The key is to involve people living with the process on a daily basis. The combined effects or interactions between the factors require careful consideration before conducting the experiment. Factors can generate non-linear effects that are not additive, but can only be studied with more complex experiments involving more than 2 level settings. Two levels are defined as linear (two points define a line), three levels are defined as quadratic (three points define a curve), four levels are defined as cubic and so on.

Experiment design process

In Figure 5.2 we can see depicted the experimental design process.

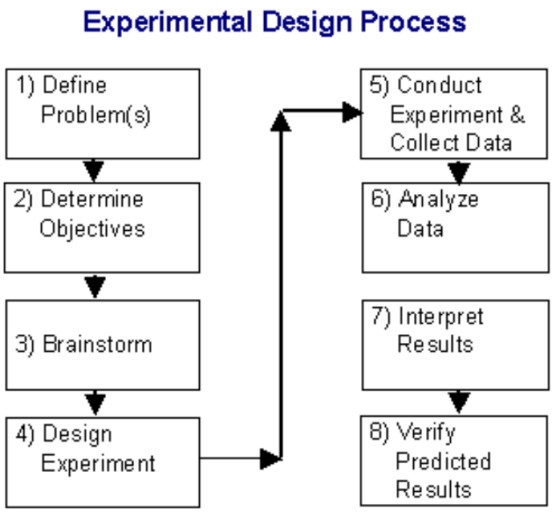


Figure 5.2: Experimental design process [19]

Multi-factor experiments

Multi-factor experiments are designed to evaluate multiple factors set on multiple levels. One approach is called *Full Factorial Experiment*, where every factor is tested with each level in every possible combination with the other factors and their levels. Full Factorial experiments that study all the coupled interactions can be cheap and practical if there are few factors and only two or three levels per factor. The advantage is that all coupled interactions can be studied. However, the number of executions increases exponentially as additional factors gets added. Experiments with many factors can quickly become cumbersome and expensive to execute, as shown in Figure 5.3. To study a larger number of factors and interactions, **Fractional Factor projects** can be used to reduce the number of executions by evaluating

Number of Factors	Number of Levels per Factor	Number of Runs Full Factorial
2	2	4
2	3	9
3	2	8
3	3	27
4	2	16
4	3	81
5	2	32
5	3	243
6	2	64
6	3	729
7	2	128
7	3	2187
8	2	256
8	3	6561

Figure 5.3: Runs table for full-factorial experiments, [19]

only a subset of all the possible factors combinations. These projects are very convenient, but the study of interactions between the factors is limited and therefore the experimental layout must be decided before the experiment execution (during the phase of experiment design). When selecting factor levels for an experiment, understanding the natural variation of the process is essential. Levels close to the process average can hide the meaning of the factor compared to its probable range of values. For the factors that are measured on a variable scale, it is advisable trying to select the levels with +/- three standard deviations from the average value.

5.2 BAT-MAN DoE

The dual goal of the experimental campaign is that of:

- creating a useful battery model for real-time application
- testing the state-estimation algorithm performances

by means of suitable experiments that allow for the highlight of the peculiar features. The *DoE* of our project is made up of three experimental session:

5.2.1 Acquisition test - 0

This preliminary experiment arises from the need to understand the phenomena underlying the dynamics of the battery free response ($I(t) = 0$), in order to optimise subsequent test campaigns. According to technical data sheets, producers suggest a settling time of 24 hours, that is the rest time required to consider $V_t = V_{ocv}$, far too long for a deeper experimental investigation like ours. The test involves a preparation phase:

- full charge
- 24h rest

and a data acquisition phase:

- complete discharge up to the minimum voltage of 10.8V

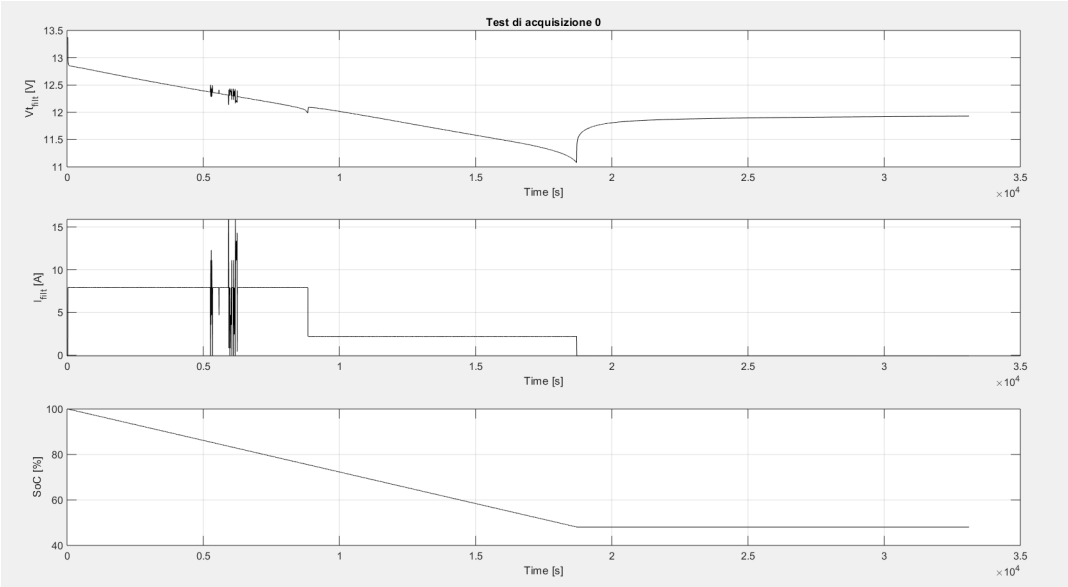


Figure 5.4: Acquisition test - 0

- rest at zero current for 12 hours

Figure 5.4 shows the actual execution highlighting the measured quantities. From the analysis of the open circuit voltage dynamics, we've decided to set a settling time of 3600[s], as an optimal compromise between characterisation of the equilibrium phase and approximation error that amounts to less than 2% (Figure 5.5).

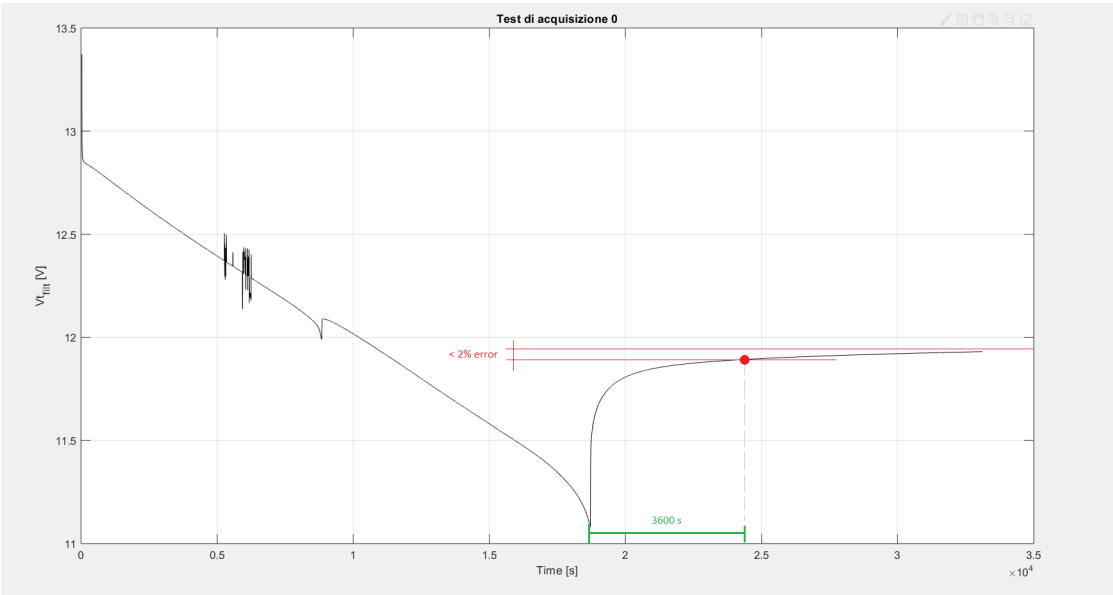


Figure 5.5: Acquisition test - 0, error trade-off

5.2.2 Acquisition test - 1

This fundamental test can be considered the heart of the analysis as it has multiple purposes:

- Experimentally determine the real open circuit voltage characteristic in function of the state of charge: $V_{ocv}(SoC)$
- Experimentally determine the real capacity of the battery: C_real
- Evaluate battery behaviours to identify mathematical models capable of reproducing the discharge phase
- Extract features to introduce self-learning approaches for classification and prediction (predictive maintenance)

The test involves a preparation phase:

- full charge
- 24h rest

and a data acquisition phase:

- discharge at 10% SoC intervals (discharge current: 4 [A])
- rest time at zero current between intervals: 3600 [s]
- STOP condition: $V_t = V_{min} = 10.8$[V]

In Figure 5.6 it's shown the actual execution:

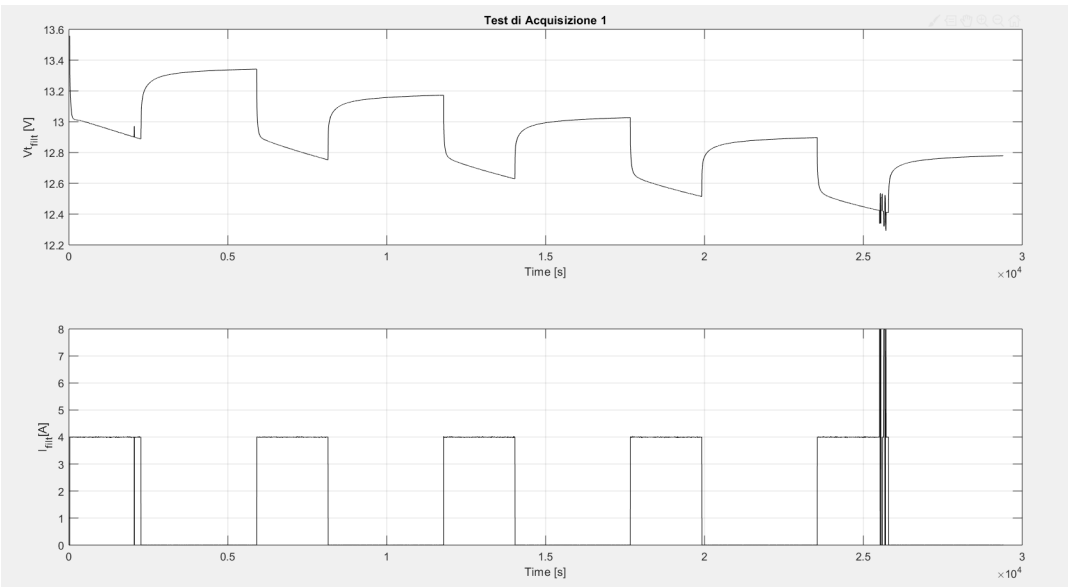


Figure 5.6: Acquisition test - 1

5.2.3 Acquisition test - 2

The objective of this experimental investigation is to reproduce a real use cycle in order to evaluate the effect of current intensity in the discharge phase (discharge at different c-rates) and validate all the done work. The test involves a preparation phase:

- full charge
- 24h rest

and a data acquisition phase:

- 3-levels discharge profile, with 2% SoC loss
- constant discharge, with 8% SoC loss (total interval -10% SoC loss)
- rest at zero current between intervals: 3600 [s]
- STOP condition: $V_t = V_{min} = 10.8$[V]

In Figure 5.7, a discharge example of a three-level profile:

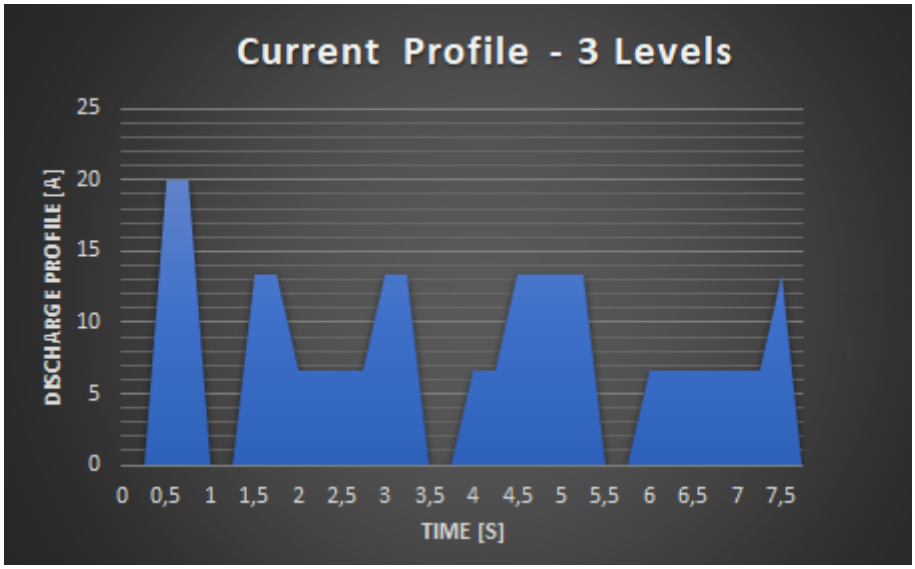


Figure 5.7: Acquisition test - 2

5.2.4 Data Organisation

Each data set is uniquely marked with an identification code based on this factors and values legend: Figure 5.8. For instance, a data set from the second acquisition test, on the medium Bosch battery, started from 70% SoC, repeated for the seventh time at standard temperature (20°), will be named:

$$A2\_M1\_C1\_SoC70\_RE7\_T0$$

5.2.5 Full BAT-MAN DoE

In Figure 5.22 it is depicted the full design of experiment of this work including:

- 3 battery model
- 3 battery states
- 10 experiment repetition
- 3 temperature levels

LEGENDA						
Simbol	**Description**	**Levels**				
A	Acquisition test	0	1	2	3	...
M	Battery Model	1 = Bosch	2 = Fiamm	3 = Energeco		
C	Battery State	0 = New	1 = Medium	2 = Old		
SoC	Initial State of Charge	100%	90%	80%	70%	...
RE	Experiment repetition	1	2	3	4	...
T	Temperature	0 = 20°C	1 = -10°C	2 = +40°C		

Figure 5.8: DoE Legend

5.3 Fundamental quantities

For the correct model identification it's of paramount importance the definition of the two main quantities:

- the Real Capacity: C_{real}
- the Open Circuit Voltage Characteristics: $V_{ocv}(SoC)$

this because with the first we universally define the actual state of health $SoH = \frac{C_{real}}{C_n}$ of the battery and therefore the maximum state of charge that a battery can reach, while the latter defines the behaviour of the internal voltage source. Both can be extracted from the **Acquisition Test 01**, of which we are now going to analyse the discharge profile.

5.3.1 Discharge Profile Analysis

In the **Acquisition Test 01**, the battery is stressed with a constant current at several SoC intervals. However, each interval has similar characteristics and is important to understand its general behaviour in order to correlate the obtained data with the equivalent circuit. With reference to Figure 5.9 we can divide the discharge into three zones:

- **Zone A**: the battery is in a state of rest (I=0 A) for a time longer than one hour (3600s). In this way the terminal voltage can be considered equal to the open circuit voltage: V_t=$V_{ocv}(1)$ and the state of charge does not change: $SoC(t)$=$SoC(1)$.
- **Zone B**: the battery is discharged at constant current until the next state of charge is reached: $SoC(1) > SoC(t) > SoC(2)$.

- **Zone C**: the current is again zero and the battery free response is analysed. It is important to underline that only after an hour (3600s), we consider again the terminal voltage coinciding with the open circuit voltage: $V_t(t) = V_{ocv}(2)$.

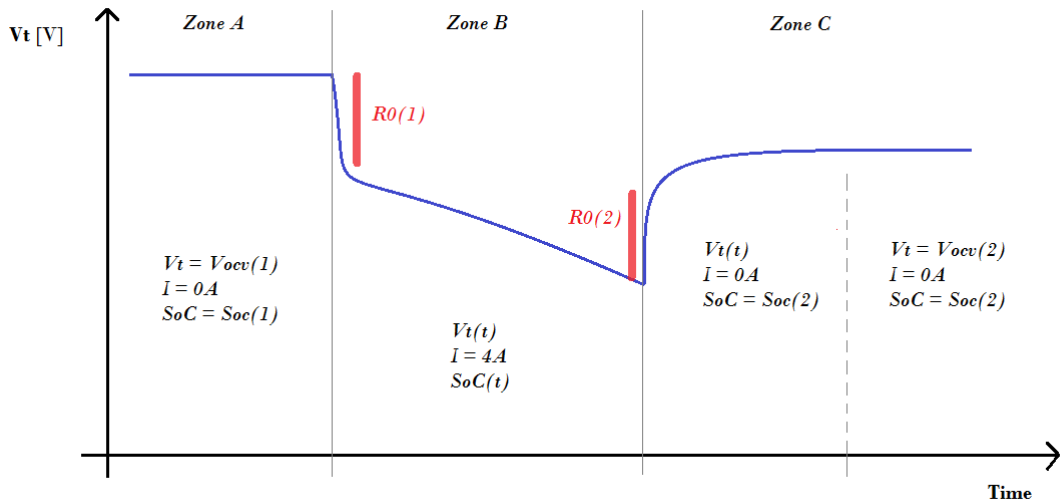


Figure 5.9: Generic discharge profile

5.3.2 Real capacity

This experiment is quite straightforward since we simply integrate over time the extracted current from full-charge to the minimum voltage threshold V_{min}, for each battery state: New, Medium, Old:

$$C_{real} = \sum_{n=0}^{t_{end}} I(n)\, T_s \tag{5.1}$$

yielding to Table 5.1:

C_{real_new}	47.5 Ah
C_{real_med}	25 Ah
C_{real_old}	3 Ah

Table 5.1: Real capacity, FIAMM Titanium L150P, 50Ah, 12V

5.3.3 Open Circuit Voltage Characteristic

A fundamental section of the equivalent electric model consists of the pseudo-voltage generator lead by the state of charge: $V_{ocv}(SoC)$. To analyse this relationship and build the open circuit voltage characteristic, it is necessary to concatenate a series of voltage values in the right no-load voltage range (Zone A), associated with their correspondent states of charge. With a standard interpolation we can eventually build the mathematical continuous relation. According to the **absolute** physical

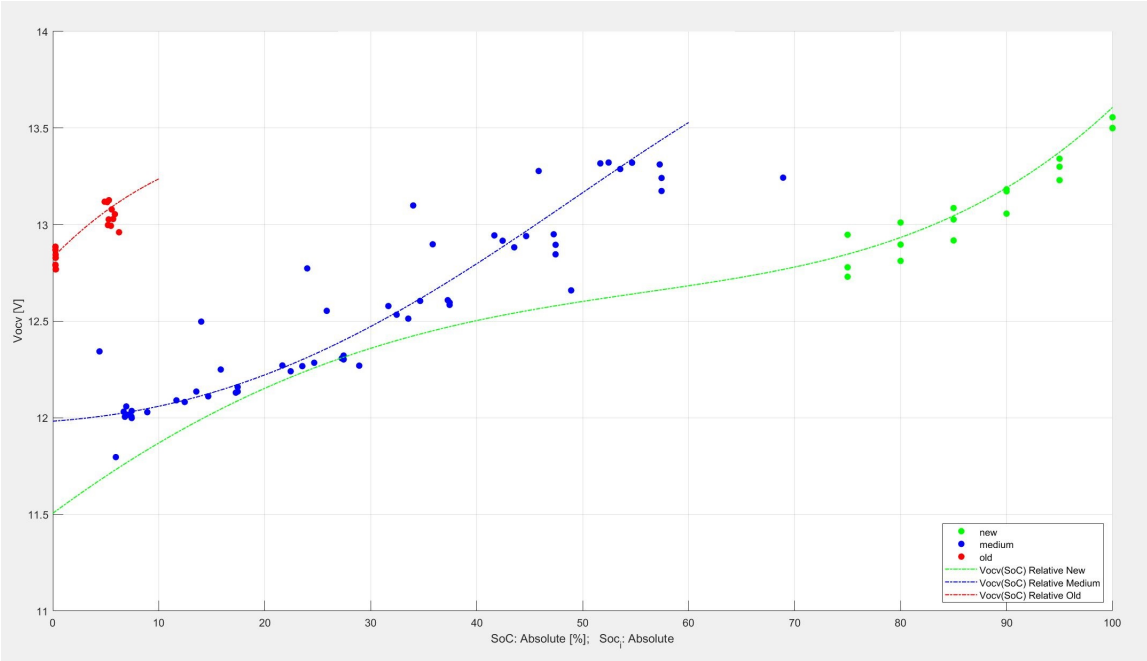


Figure 5.10: Absolute Interpretation: $V_{ocv}(SoC)$ characteristic interpolation

interpretation discussed in section 4.1, we have 3 different $V_{ocv}(SoC)$ curve, one for each battery state (Figure 6.2). The best interpolation curve results to be the 4th order:

$$V_{ocv}(SoC) = a_4\ SoC^4 + a_3\ SoC^3 + a_2\ SoC^2 + a_1\ SoC + a_0$$

whose real values are shown in Table 5.2. It's important to notice that each of these

Parameter	New	Med	Old
a_4	2.962535064794e-08	-7.037522683943e-08	-9.620228489834e-08
a_3	-9.256468042548e-07	4.699901598046e-06;	2.215673138530e-05
a_2	-3.980317908835e-04	3.347037976185e-04	-1.753418892020e-03
a_1	4.044344158681e-02	3.965037927789e-03	5.606582286310e-02
a_0	1.150595067431e+01	1.198283050271e+01	1.283047834034e+01

Table 5.2: Open Circuit Voltage Characteristics Coefficients

three characteristic has it's own working range:

$$SoC_{max} > SoC(t) > 0$$

that it's determined by the absolute interpretation for which:

$$SoC_{max} = SoH \quad \text{@full-charge}$$

5.4 Parameters Identification

Let's remember that The Bat-Man device is a low-cost Embedded System, that should be able to provide the current conditions of a battery in real-time. Starting from this consideration we can immediately exclude complex electrochemical-based

models that require high computational resources and a lot of time for the generation of the result, albeit very detailed and accurate. Focusing now on the electrical equivalent, it is necessary to take into account that each parameter entered is function of:

- Battery type: Pb, Li-Ion, LiPo, NiMh ...
- State of Charge: SoC
- Health Status: SoH
- Temperature: T [°C]
- Discharge current: I [A]

Although in our analysis the field of battery technology has been narrowed to the lead-acid, the space of inference required to analyse all the conditions remains incredibly vast. In this regard, our model selection process is of the Bottom-Up type, that is starting from the simplest model available we gradually increase the complexity keeping track of the improvements obtained. At this stage, where we have universally selected and experimentally defined the main model quantities, we can proceed to the identification of the electric equivalent model parameters and model selection.

5.4.1 Least Square Estimation Theory

Our analysis aims to estimate the parameters of the electrical equivalent model starting from the acquisition of the discrete measurable signals of the quantities of the system:

$$I(k) \;\; ; \;\; V_t(k), \quad with \;\; k = 1, ..., N$$

To this end, it is initially necessary to derive the dynamic equation of the system in the Frequency Domain, also known as Laplace Domain, which in general for this equivalent model (Figure 5.11) has this structure:

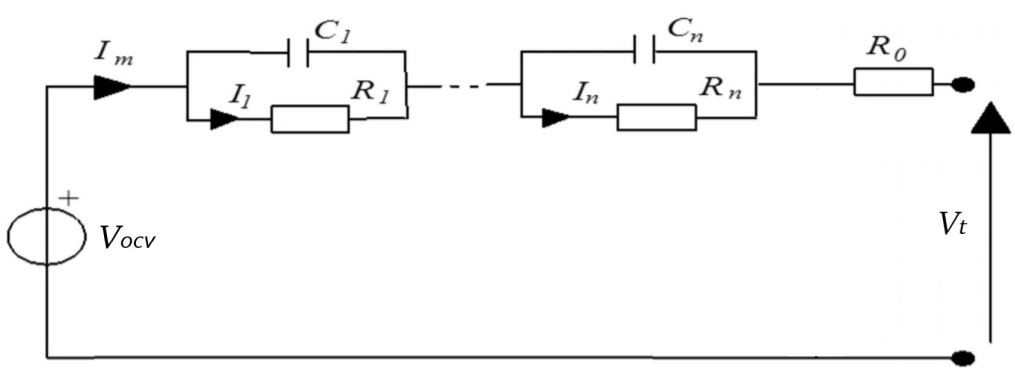


Figure 5.11: Generic electric equivalent model

$$H(s) = \frac{V_{ocv}(SoC, s) - V_t(s)}{I(s)} = R_0 + \frac{R_1}{s\,C_1} \cdot \frac{1}{\left(R_1 + \frac{1}{s\,C_1}\right)} + ... + \frac{R_n}{s\,C_n} \cdot \frac{1}{\left(R_n + \frac{1}{s\,C_n}\right)}$$

Supposing a *Zero Order Hold* conversion system type and an experimental acquisition sampling time of: $T_s = 0.01s$, we pass to the Z-domain, remembering that:

$$G(z) = \frac{z-1}{z}\left\{\frac{G(s)}{s}\right\}_z$$

yielding to:

$$H(z) = \frac{V_{ocv}(SoC, z) - V_t(z)}{I(z)} = R_0 + R_1 + ... + R_n + R_1\left(\frac{1-\alpha_1}{z-\alpha_1}\right) + ... + R_n\left(\frac{1-\alpha_n}{z-\alpha_n}\right)$$

where:

$$\alpha_i = e^{-\frac{T_s}{R_i\,C_i}} = e^{-\frac{T_s}{\tau_i}}$$

We can finally switch to the Discrete-Time domain with the difference equation, defining:

$$w(k) = V_{ocv}k - V_t(k)$$

and:

$$q^{-i} \stackrel{def}{=} \text{backward shift operator}$$

such that:

$$x(k)\ q^{-i} = x(k-i)$$

and therfore we get:

$$w(k) = -a_1\ w(k-1) - ... - a_n\ w(k-n) + b_0\ I(k) + b_1\ I(k-1) + .. + b_n\ I(k-n)$$

For each instant we can manage the noise uncertainty introducing an error term $e(t)$ in "Equation Error" form:

$$D(q^{-i})y(k) = N(q^{-i})I(k) + e(k)$$

that translates in:

$$y(k) = -a_1\ y(k-1) - ... - a_n\ y(k-n) + b_0\ I(k) + b_1\ I(k-1) + .. + b_n\ I(k-n) + e(k)$$

We can now perform a point estimation by means of a least squares estimator. Starting from:

$$b = A\ \theta + e$$

where in our case:

$$b = \begin{pmatrix} y(k+n) \\ \vdots \\ y(N) \end{pmatrix}$$

$$a = \begin{pmatrix} -y(k+n-1) & \cdots & -y(1) & I(k+n) & I(k+n-1) & \cdots & I(1) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ -y(N-1) & \cdots & -y(N-n) & I(N) & I(N-1) & \cdots & I(N-n) \end{pmatrix}$$

$$\theta = \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_n \\ b_0 \\ b_1 \\ \vdots \\ b_n \end{pmatrix}$$

we have to solve an optimisation problem of the type:

$$\hat{\theta_{LS}} = \underset{\theta}{\operatorname{argmin}} \, ||b - A\theta|| = \underset{\theta}{\operatorname{argmin}} \, ||e||$$

It's possible to prove that the optimal solution, under the hypothesis previously described, is:

$$\hat{\theta_{LS}} = \left(A' \, A\right)^{-1} \, A' \, b$$

Considering a first order system ($n = 1$), it is possible to come back to the original physical quantities solving the following system:

$$\begin{cases} a_1 = -\alpha = -e^{-\frac{T_s}{\tau_1}} \\ b_0 = R_0 \\ b_1 = -R_0 \, \alpha + R_1 \, (1 - \alpha) \end{cases}$$

For more complex models it's not always possible to retrieve the physical parameters and may be necessary to work with the discrete time synthetic parameters.

5.4.2 R Model

The single resistance model (Figure 5.12), is made up of an ideal voltage generator which is a function of the state of charge V_{ocv} and a resistance R_0 which is a function of the temperature T, the state of charge SoC, state of health SoH and discharge current I. As shown in subsection 5.3.3, the open circuit voltage characteristic is

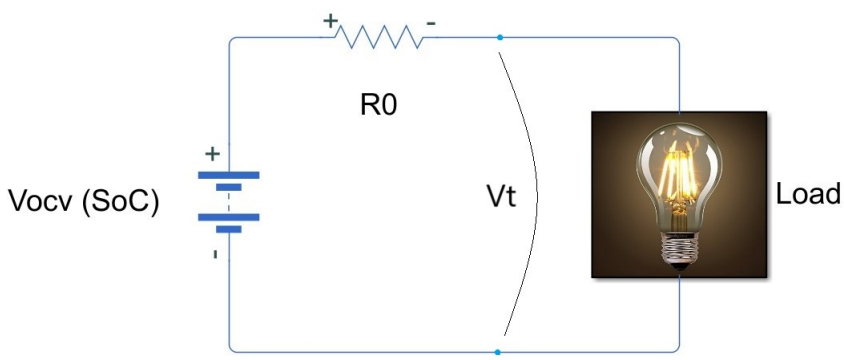


Figure 5.12: R-model

well represented by a 4th order curve but it is even possible to approximate it with a straight line while achieving lower performance. The static equation is:

$$V_{ocv}(SoC) = V_t + R_0 \, I$$

The main advantage of this very simple model lies in the chance of avoiding overfitting problems, and therefore embracing a bigger inference space, accepting a lower fidelity.

Identification interval

To obtain satisfactory results it is advisable to choose the smarter time interval on which performing the identification, exploiting both the knowledge of the physical phenomenon and the limits of the equivalent model that we are using. It would be useless the hope of being able to identify the dynamics of the system with a pseudo-static equivalent model with a single resistance. It is clear that this kind of model is not suitable for high dynamics and it would be useless trying to let the model identify them. That is exactly why we set a *Voltage Threshold* of 500s to avoid the initial dynamics, resulting in the range highlighted in Figure 5.13.

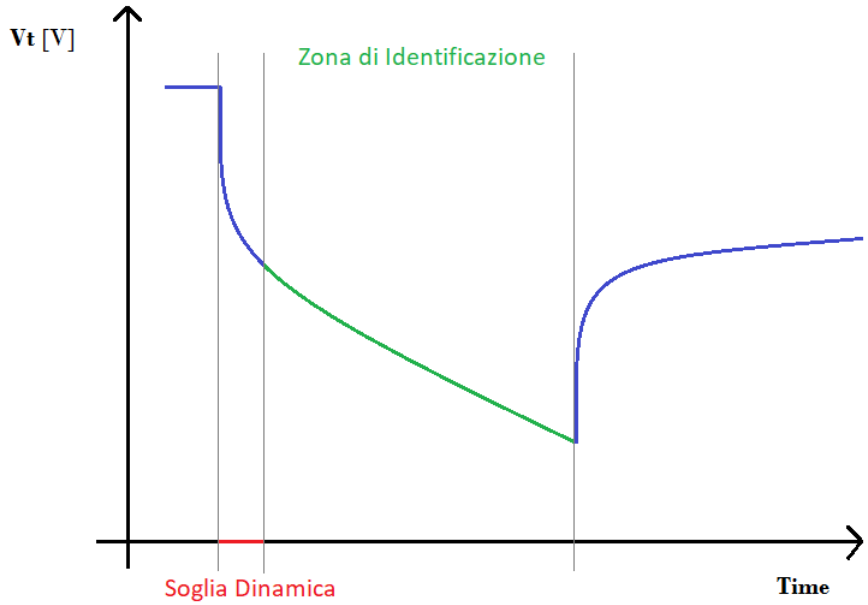


Figure 5.13: Identification range - R-model

Identification results

In Figure 5.14 we can see on the left side the identified parameters while on the right side we have the root mean squared error as a measure of identification goodness, expressed as:

$$RMSe = \sqrt{\left(V_t(k) - \hat{V_t}(k)\right)^2}$$

One of the most blatant evidence is the fact that for low states of charge, where the non-linearity of the real behaviour are stronger, this simple model is working pretty badly, with errors up to four time bigger than the ones for high states of charge.

5.4.3 Dynamic Model

The dynamic model (Figure 5.15) is made up of both an ideal voltage generator which is function of the state of charge and an electrical network composed of a resistance R_0 and an $RC - branch$. In this way is possible to approximately model the dynamics of the system response. Also in this case, as shown in subsection 5.3.3, the open circuit voltage characteristic is well represented by a 4th order curve but it

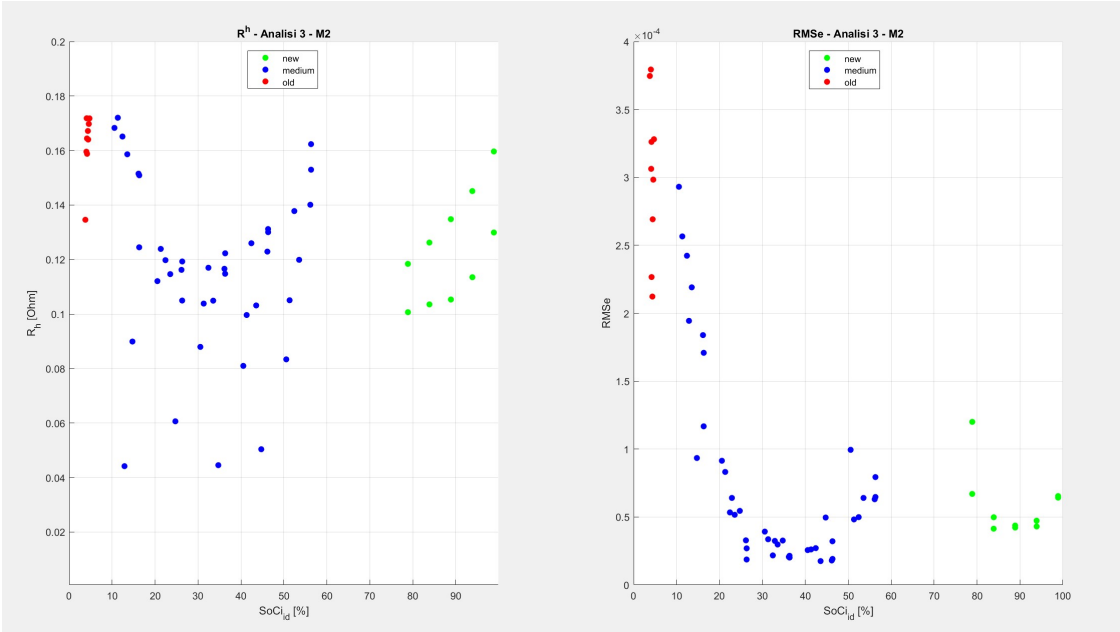


Figure 5.14: Identification results - R-model

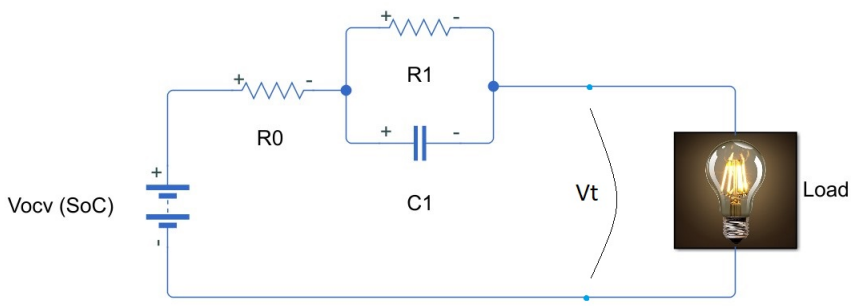


Figure 5.15: Dynamic model

is even possible to approximate it with a straight line while achieving lower performance. The dynamic equation equation in the Laplace Domain is:

$$H(s) = \frac{V_{ocv}(SoC, s) - V_t(s)}{I(s)} = R_0 + \frac{R_1}{s\,C_1} \cdot \frac{1}{\left(R_1 + \frac{1}{s\,C_1}\right)} +$$

Identification interval

To obtain satisfactory results it is advisable to choose the smarter time interval on which performing the identification, exploiting both the knowledge of the physical phenomenon and the limits of the equivalent model that we are using. For this model we have included in the identification interval, all the experiment, except for the free-response which would have required very high order systems. The resulting identification range is highlighted in Figure 5.16.

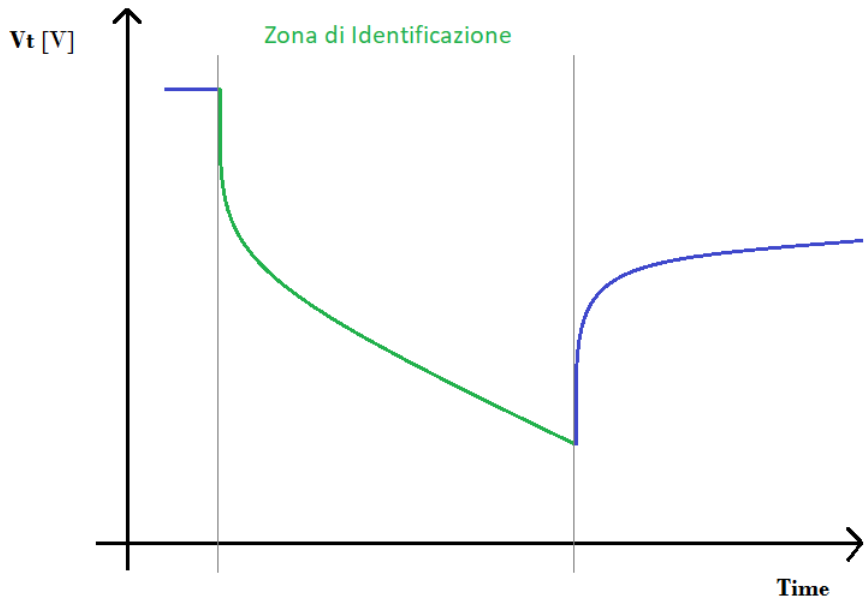


Figure 5.16: Identification range - Dynamic model

Identification results

In Figure 5.21 we can see the identified parameters: R_0,R_1, C_1, while in Figure 5.18 there is a focus on the τ constant:

$$\tau_1 = R_1\ C_1$$

In this case, the R_0 distribution starts to be interesting but the error is still high

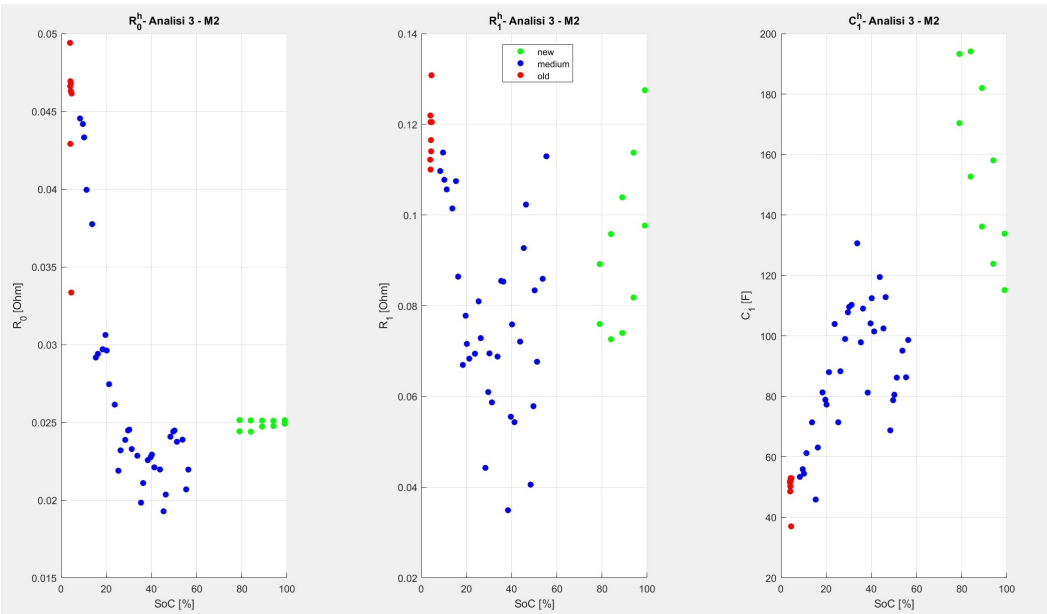


Figure 5.17: Identification results - Dynamic model

because the non-linearity are mixed-up with the dynamic and moreover, in the

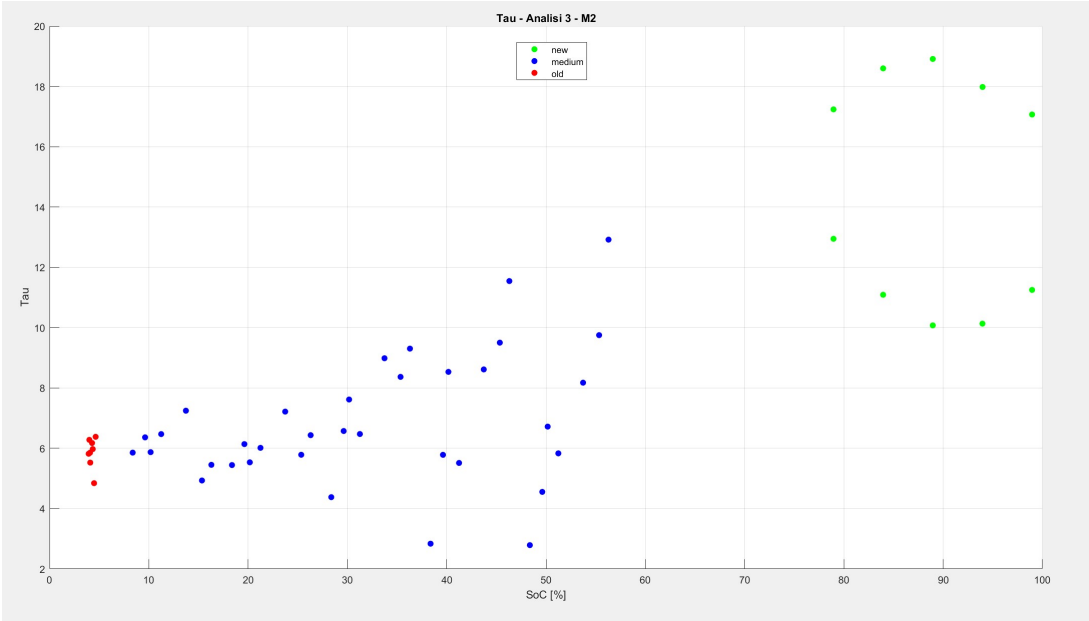


Figure 5.18: Identification results - τ_1 - Dynamic Model

simulation environment, the first order dynamics does not introduce significant improvements in term of modelling effectiveness. **The real dynamics would require higher order system, incurring in a curse of parameter dimension and model complexity, but the problem with the non-linearity will still persist. This is exactly why we ended up with the following non-linear model.**

5.4.4 Non-Linear R Model

This model stems from the consideration that in a constant current discharge phase, the terminal voltage assumes non-linear behaviours depending on the state of charge and the state of health.

$$const. \textbf{ input} \implies non-linear \textbf{ output}$$

Examining the dynamic model described above, if for high SoC and SoH can guarantee decent performance, the same cannot be said in the final phases in which complex non-linearities are completely ignored. And is precisely there that the performance in estimation "goodness" is still low. The problem can be properly managed with the following model: Figure 5.19. It maintains the basics structure but there is the insertion of a non linear term suitably defined. Let's have a look at the mathematical description:

$$V_{ocv} = V_t + (R_0 + R_{nl})\, I \tag{5.2}$$

$$R_{nl} = k_1 \left(\frac{\int_0^t I(\tau) d\tau}{C_{real}} \right)^2 \cdot e^{k_2 \left(\left(\frac{\int_0^t I(\tau) d\tau}{C_{real}} \right)^2 \frac{C_n}{C_{releasable}(t)} \right)} \tag{5.3}$$

where k_1 and k_2 are a couple of constant function of the SoC and SoH.

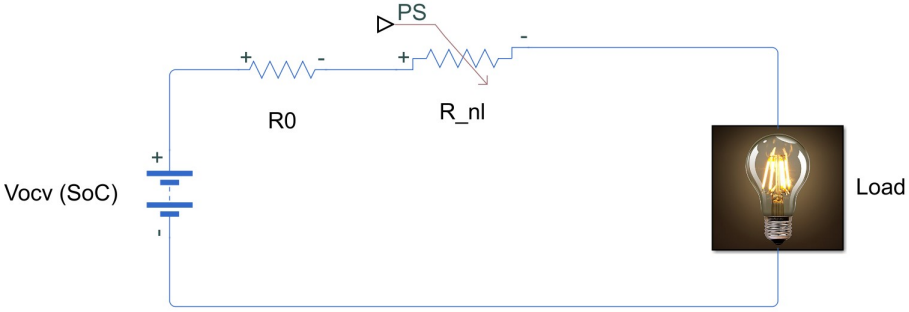


Figure 5.19: Non-linear model

Physical meaning

Let's have a look at the intrinsic meaning of Equation 5.3. The first term:

$$\left(\frac{\int_0^t I(\tau)d\tau}{C_{real}}\right)^2$$

is a quantity that lies in the $[0,1]$ range and for which when $t=0$ $(full-charge)$ it worth 0 and it nulls the effect of the non linearity, while when $t=t_f$ $(full-discharge)$ it worth 1 and it has maximum effect on the non-linearity. It can be seen as modulator and we have squared it to change his effect toward the lower state of charge without affecting the range bounds. The second term:

$$e^{k_2\left(\left(\frac{\int_0^t I(\tau)d\tau}{C_{real}}\right)^2 \frac{C_n}{C_{releasable}(t)}\right)}$$

it's an exponential term that include the previous one and introduce a new quantity $\left(\frac{C_n}{C_{releasable}(t)}\right)$, strictly related to the current status of the battery and its SoH. Indeed it is always greater than one and it has no effect when $C_{releasable}(t)$ is close to the nominal capacity C_n (ratio close to 1), while it has maximum effect when battery is very discharge or it is fully charge but pretty old (ratio $\gg 1$). Finally, we can also switch from the energetic quantities interpretation to the state variables one:

$$R_{nl} = k_1\left(\frac{DoD(t)}{DoD(t)+SoC(t)}\right)^2 \cdot e^{k_2\left(\left(\frac{DoD(t)}{DoD(t)+SoC(t)}\right)^2 \frac{1}{SoC(t)}\right)} \tag{5.4}$$

Proof: remembering that

$$DoD(t) = DoD(t_0) + \frac{\int_{t_0}^t I(\tau)d\tau}{C_n} = \frac{\int_0^{t_0} I(\tau)d\tau}{C_n} + \frac{\int_{t_0}^t I(\tau)d\tau}{C_n}$$
$$SoH = \frac{C_{real}}{C_n} = DoD(t) + SoC(t)$$

we can rewrite the first term, multiplying and dividing by C_n:

$$\frac{\int_0^t I(\tau)d\tau}{C_{real}} = \frac{\int_0^t I(\tau)d\tau}{C_{real}}\frac{C_n}{C_n} = \frac{DoD(t)}{SoH} = \frac{DoD(t)}{DoD(t)+SoC(t)}$$

while the ratio in the exponential can be seen as:

$$\frac{C_n}{C_{releasable}(t)} = \frac{C_n}{C_{real} - \int_0^t I(\tau)d\tau} = \frac{C_n}{C_{real}\frac{1}{1-\frac{\int_0^t I(\tau)d\tau}{C_{real}}}} = \frac{1}{SoH - DoD(t)} = \frac{1}{SoC(t)}$$

yielding to Equation 5.4.

Identification interval

The model is non-linear but it remains static. this is why we have selected smae identification interval of the R-model: Figure 5.20. In this way we can avoid that

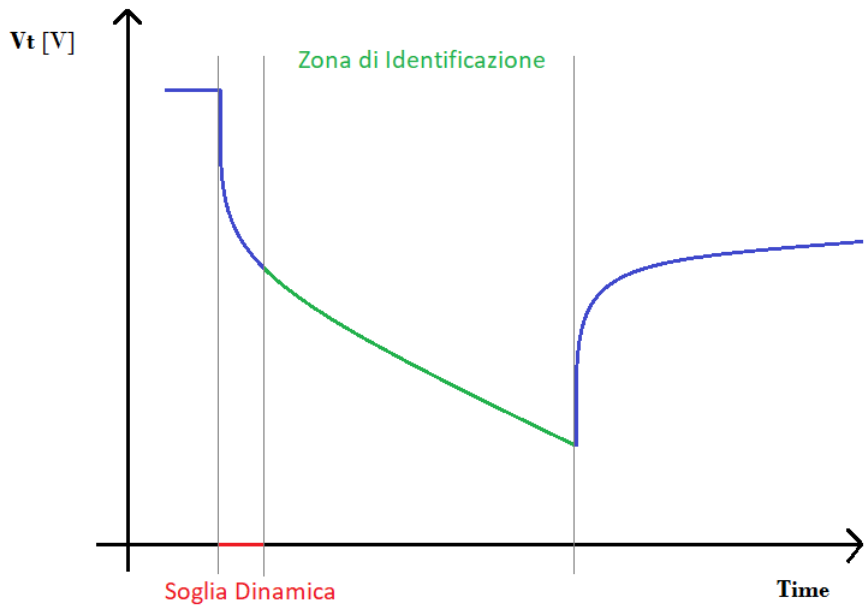


Figure 5.20: Identification range - Non linear model

the initial dynamics affects the estimation performance.

Identification results

In Figure 5.21 we can immediately see the power of this solution. On the left side the identified resistances describe a mathematical shape similar to a parabola and the trend is clearly visible. Moreover, on the right side we can see that the error remains extremely low along all the inference space: $< 1.5e^{-4}$ for each $SoCs$. **Thanks to this model we've been able to correctly identify the model parameters in function of the SoC and the SoH and in chapter 6 we'll see how to create the final model to be used in the model-based final solution.**

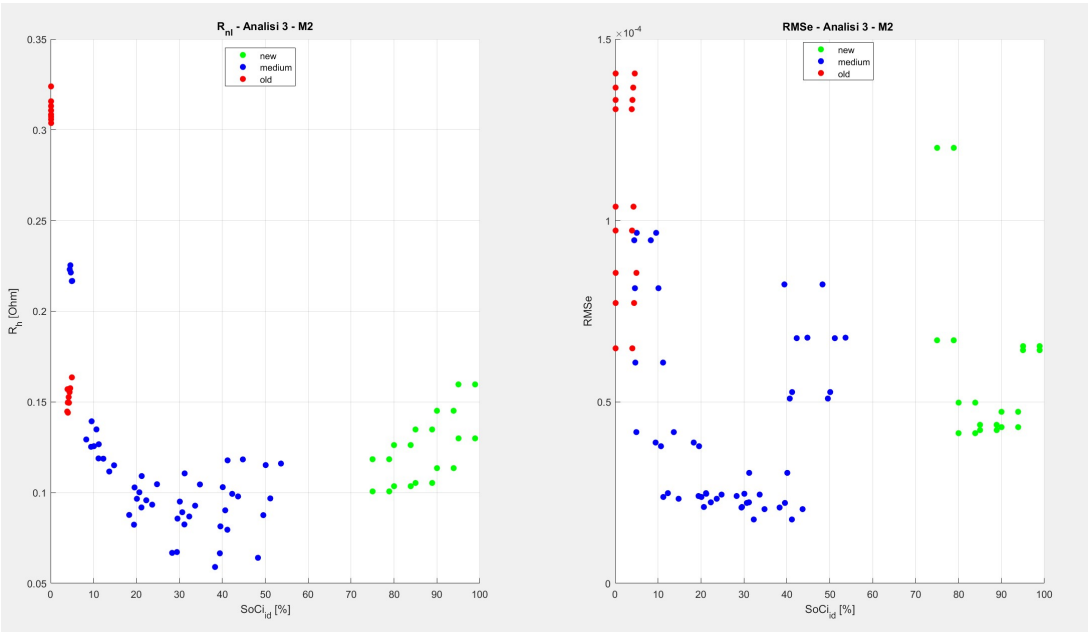


Figure 5.21: Identification results - Non-linear model

Experiment	Factors e Levels	
A0	Battery model	M1
	Battery state	C1
		C2
	Initial State of Charge	SoC100
	Temperature	T0
	Experiment Repetition	RE1

A1	Battery model	M1
		M2
		M3
	Battery state	C0
		C1
		C2
	Initial State of Charge	SoC100
		⋮
		SoCend
	Temperature	T0
		T1
		T2
	Experiment Repetition	RE1
		⋮
		RE10

A2	Battery Model	M1
		M2
		M3
	Battery State	C0
		C1
		C2
	Initial State of Charge	SoC100
		⋮
		SoCend
	Temperature	T0
		T1
		T2
	Experiment Repetition	RE1
		⋮
		RE10

Figure 5.22: Full BAT-MAN DoE

Chapter 6

Battery Model

In this chapter we will exploit the data coming from the identification section to create a complete battery model. The model is made up of a couple of convex combinations that, in function of the provided State of health (SoH), are able to span all the battery behaviour in a discharge session ($0 < SoC(t) < C_{real}$).

6.1 The Structure

Starting from the evidence that during a finite number of charge-discharge cycles the SOH (State of Health) is a quasi-static physical quantity, we can at first assume it constant and consider it as a model parameter. In chapter 7 we'll see how to exploit this structure to build the solution. The model can be briefly represented by the electric equivalent circuit model in Figure 6.1.

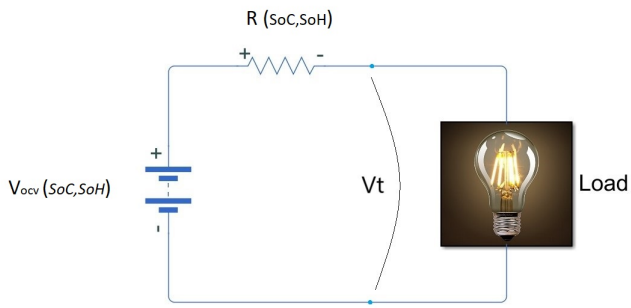


Figure 6.1: Equivalent electric circuit

It consists of a couple of non-linear elements that act like a voltage source $V_{ocv}(SoC, SoH)$ and a resistor $R(SoC, SoH)$. This elements are made up of convex combinations in such a way that they receive as input both the SoH (State of Health) and the SoC (State of Charge) and respectively provide the correspondent value of the open circuit voltage and resistance. The synthetic model description in the discrete-time state-space form is:

$$\begin{cases} DoD(k+1) = DoD(k) + \frac{\Delta T}{C_n}\, I(k) \\ V_t(k) = V_{ocv}\Big(DoD(k), SoH\Big) - R_{nl}\Big(DoD(k), SoH\Big)\, I(k) \end{cases}$$

and since:

$$SoH = SoC(k) + DoD(k) \cong const. \quad \forall k = 1, ..., N$$

we can equivalently reformulate it as:

$$\begin{cases} SoC(k+1) = SoC(k) - \frac{\Delta T}{C_n} I(k) \\ V_t(k) = V_{ocv}\Big(SoC(k), SoH\Big) - R_{nl}\Big(SoC(k), SoH\Big) I(k) \end{cases}$$

In the following we'll see in depth both how V_{ocv} and R_{nl} are built from the experimental data acquisitions and how they work.

6.1.1 Voltage Source model block

The voltage source should represent the behaviour of the open circuit voltage relation:

$$V_{ocv}(SoC, SoH)$$

or equivalently:

$$V_{ocv}(DoD, SoH)$$

According to the analysis performed in chapter 5, where we've dealt with the V_{ocv} curves experimental trends at the three main health conditions (New, Medium, Old), we notice in the first instance that they can be approximated with a straight line that rotates and translates with ageing.

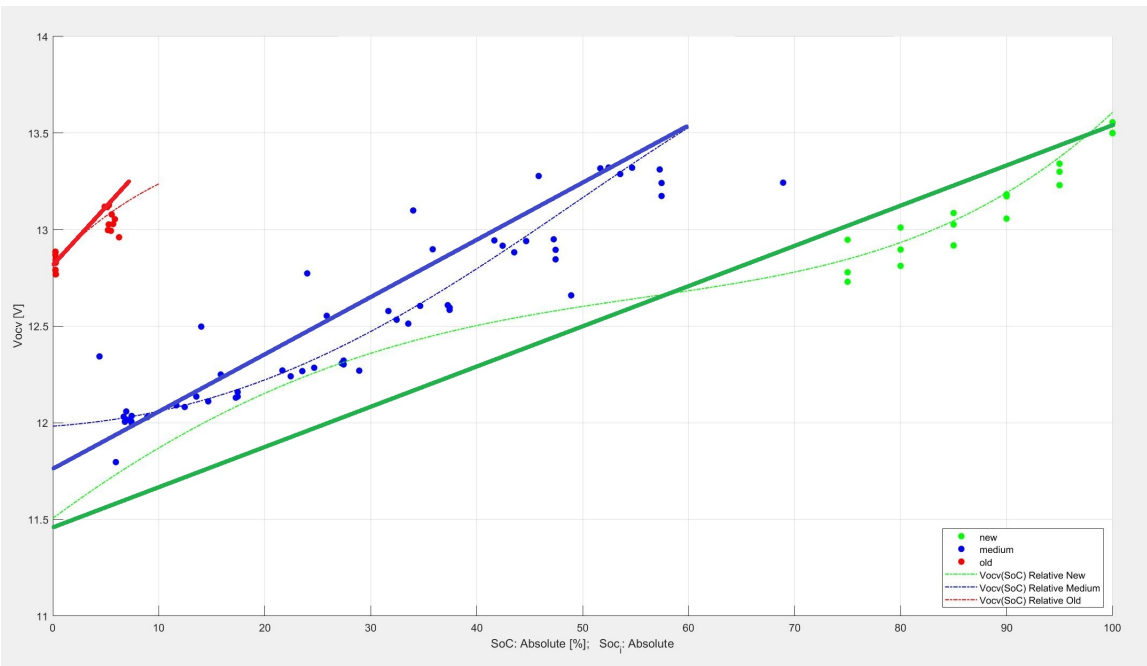


Figure 6.2: Linear approximation - $V_{ocv}(SoC)$ characteristic interpolation

Following this insight we can build a convex combination of straight lines that spans over the SoH range. Defining b_{1_i} the ith angular coefficient and b_{0_i} the ith intercept, we can represent in the parameters space the starting and the final conditions: We call $\mathbf{x}_{new}$ the point represented by the couple (b_{0_new}, b_{1_new}) that

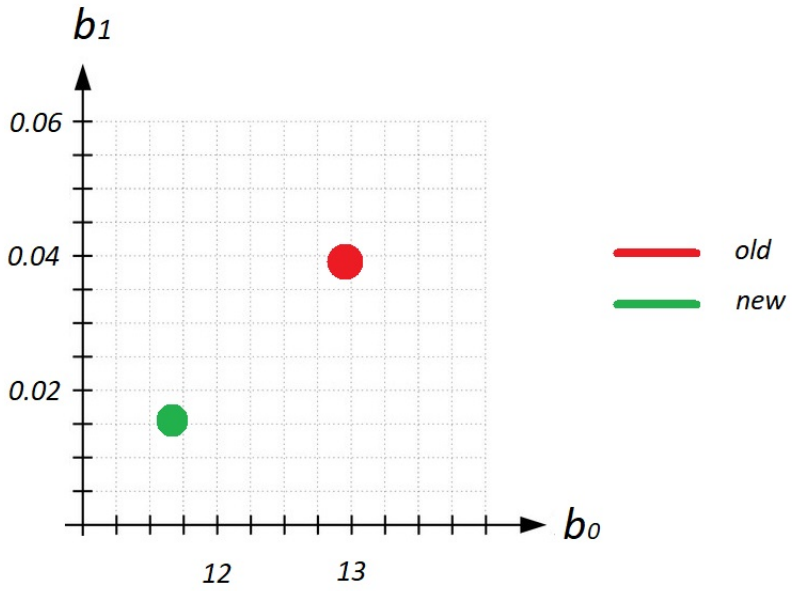


Figure 6.3: Convex combination parameters space

corresponds to the line associated to SoH=1 (new battery) and we call $\mathbf{x}_{old}$ the point represented by the couple (b_{0_old}, b_{1_old}) that corresponds to the line associated to SoH=0 (dead battery). In this way we have defined the limits of the convex combination and the generic conditions x_i is given by:

$$\mathbf{x}_i(SoH) = \mathbf{x}_{new} + (1 - SoH)^k(\mathbf{x}_{old} - \mathbf{x}_{new})$$

$$(SoH = 0 \mapsto \ \mathbf{x}_i = \mathbf{x}_{old} \quad ; \quad SoH = 1 \mapsto \ \mathbf{x}_i = \mathbf{x}_{new})$$

Now that we have the basic structure, we can tune the coefficient k to match the non linearity with whom the trajectory space it's spanned by the empirical data. Indeed, since the base $(1 - SoH)$ lies in the $[0, 1]$ range, the exponent k can be chosen at will without affecting the convex combination boundaries. In our case the choice $(1 - SoH)^4$ gives the optimal fit to experimental data:

$$\mathbf{x}_i(SoH) = \mathbf{x}_{new} + (1 - SoH)^4 \ (\mathbf{x}_{old} - \mathbf{x}_{new})$$

that is:

$$\mathbf{x}_i(SoH) = (b_{0\_i}, b_{1\_i})$$

In this way the SoH uniquely defines the line parameters and a first approximation of the $V_{ocv}(SoC/DoD, SoH)$ is:

$$V_{ocv\_linear\_i}(SoC, SoH) = SoC \ b_{1\_i} + b_{0\_i}$$

Eventually we can add a sinusoidal term that is responsible for the management of the intrinsic s-shaped non-linearity of the real V_{ocv} characteristics:

$$V_{ocv\_i}(SoC, SoH) = SoC\ b_{1\_i} + b_{0\_i} + k_2\ SoH\ sin(2\pi\ (SoH - SoC))$$

or equivalently

$$V_{ocv\_i}(DoD, SoH) = (SoH - DoD)\ b_{1\_i} + b_{0\_i} + k_2\ SoH\ sin(2\pi\ DoD)$$

where k_2 is a model parameter to be experimentally obtained.
In this schematic representation (Figure 6.4) is summarised at high level the structure of this main model block. The blue squares represent the functions, the green

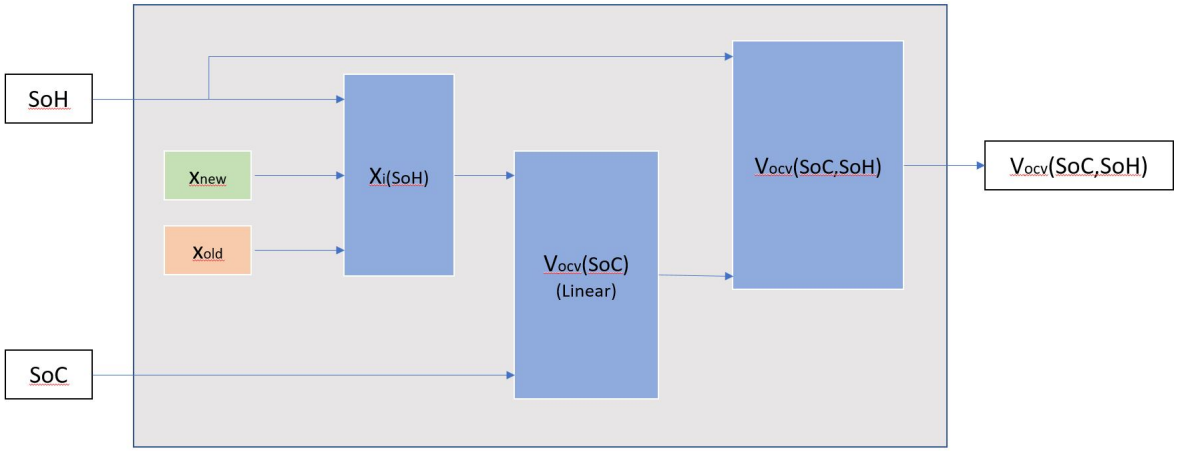


Figure 6.4: Voltage source model block

and red ones represent the limits (parameters) of the convex combination and the white squares are the inputs and outputs of this first model block.

6.1.2 Resistor model block

The resistor should represent the behaviour of the battery internal impedance:

$$R_{nl}(SoC, SoH)$$

or equivalently:

$$R_{nl}(DoD, SoH)$$

According to the analysis performed in chapter 5, where we've dealt with the identification of the resistance distributions at the three main health conditions (New, Medium, Old), we notice in the first instance that they can be approximated with a parabola that shrinks and translates with ageing (Figure 6.5). Following this insight we can build a convex combination of parabolas that spans over the SoH range. Defining c_{0_i}, c_{1_i}, c_{2_i} the parabola's numeric coefficients, we can represent in the parameters space the starting and the final conditions (Figure 6.6). We call $\mathbf{x}_{new}$ the point represented by the triplet (c_{0_new}, c_{1_new}, c_{2_new}) that corresponds to the

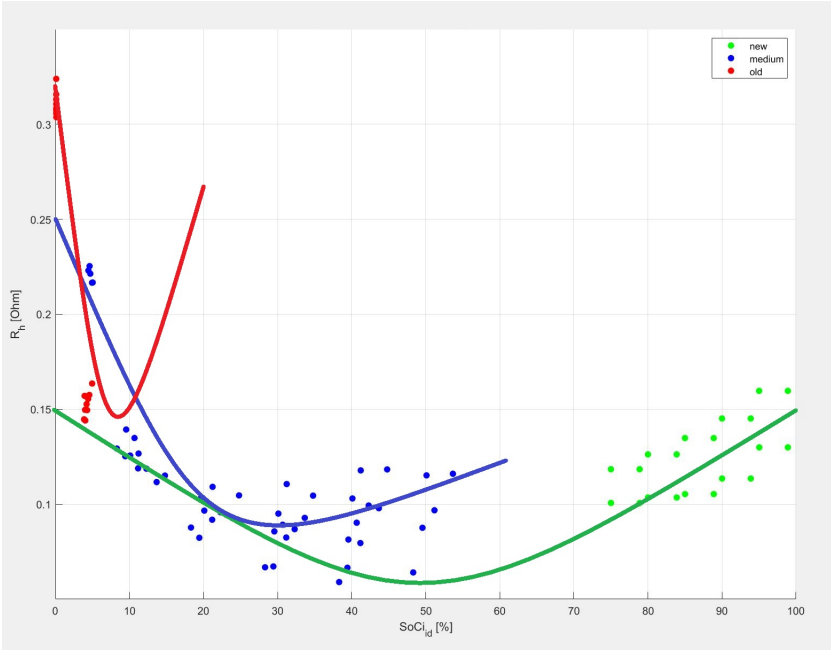


Figure 6.5: R_{nl} parabolic interpolation

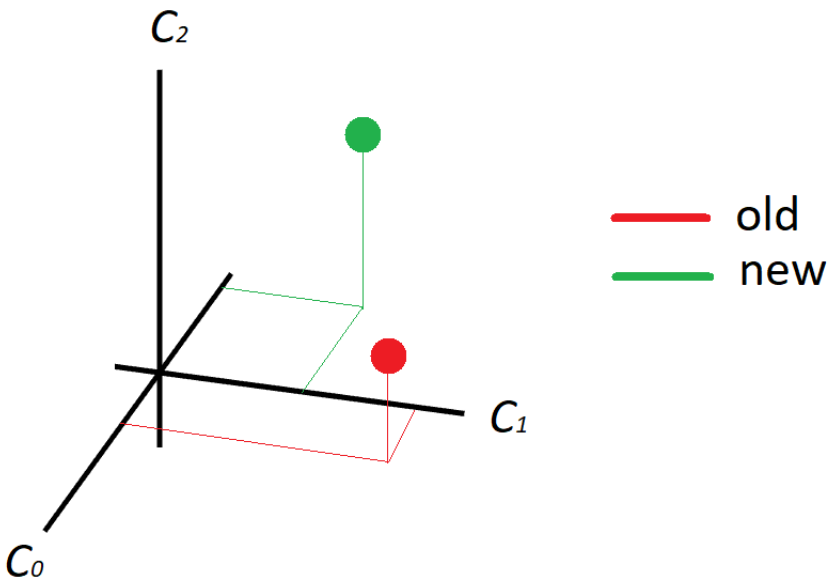


Figure 6.6: R_{nl} - Convex combination parameters space

parabola associated to $SoH = 1$ (new battery) and we call $\mathbf{x}_{old}$ the point represented by the triplet (c_{0_old}, c_{1_old}, c_{2_old}) that corresponds to the parabola associated to $SoH = 0$ (dead battery). In this way we have defined the limits of the convex combination and the generic conditions $\mathbf{x}_i$ is given by:

$$\mathbf{x}_i(SoH) = \mathbf{x}_{new} + (1 - SoH)^k(\mathbf{x}_{old} - \mathbf{x}_{new})$$

$$(SoH = 0 \mapsto \ \mathbf{x}_i = \mathbf{x}_{old} \quad ; \quad SoH = 1 \mapsto \ \mathbf{x}_i = \mathbf{x}_{new})$$

Now that we have the basic structure, we can tune the coefficient k to match the non linearity with whom the trajectory space it's spanned by the empirical data.

Indeed, since the base $(1 - SoH)$ lies in the $[0, 1]$ range, the exponent k can be chosen at will without affecting the convex combination boundaries. In our case the choice $(1 - SoH)^8$ gives the optimal fit to experimental data:

$$\mathbf{x}_i(SoH) = \mathbf{x}_{new} + (1 - SoH)^8 \, (\mathbf{x}_{old} - \mathbf{x}_{new})$$

that is:

$$\mathbf{x}_i(SoH) = (c_{0\_i}, c_{1\_i}, c_{2\_i})$$

In this way the SoH uniquely defines the parabola parameters and a first approximation of the $R_{nl}(SoC/DoD, SoH)$ is:

$$R_{nl\_linear\_i}(SoC, SoH) = SoC^2 \; c_{2\_i} + SoC \; c_{1\_i} + c_{0\_i}$$

Eventually we can add an exponential term that is responsible for the management of the intrinsic non-linearity in the low-charge range:

$$R_{nl\_i}(SoC, SoH) = SoC^2 \; c_{2\_i} + SoC \; c_{1\_i} + c_{0\_i} + k_7 \; e^{\left(k_6 (1 - \frac{SoC}{SoH})\right)^8}$$

or equivalently

$$R_{nl\_i}(DoD, SoH) = (SoH - DoD)^2 \; c_{2\_i} + (SoH - DoD) \; c_{1\_i} + c_{0\_i} + k_7 \; e^{\left(k_6 \frac{DoD}{SoH}\right)^8}$$

where k_6 and k_7 are model parameters to be experimentally obtained.
In this schematic representation (Figure 6.7) is summarised at high level the structure of this main model block. The blue squares represent the functions, the green

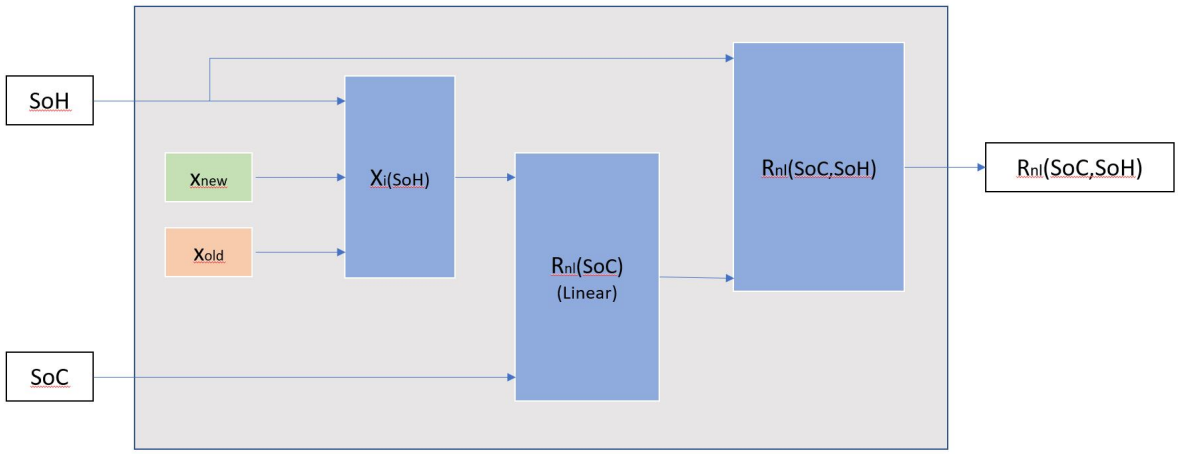


Figure 6.7: Non- linear Resistor model block

and red ones represent the limits (parameters) of the convex combination and the white squares are the inputs and outputs of this second model block.

6.2 Model Equations

With the aim of giving both a rigorous mathematical description and a comprehensive visual overview on which the overall model it's founded, in the following

we'll merge the model blocks to obtain the input/output relationship. According to measurement at our disposal, we consider the current drained or provided to the battery $I(t)$ as the system input $U(t)$ while the terminals voltage $V_t(t)$ is the system output $Y(t)$. At present the SoH (State of Health) enters the problem as a fixed parametric input that fixes the behaviour of the model though in the next chapter we'll see how to exploit this structure to implement the solution.

6.2.1 State Equation

The state of the system can be chosen at will among the SoC (State of Charge) and the DoD (Depth of Discharge) bearing in mind the energetic relation expressed in chapter 4:

$$SoH = SoC(k) + DoD(k) \cong const. \quad \forall k = 1, ..., N$$

IMPORTANT: *Henceforward we'll use the* ***DoD*** *as* ***state*** $X(t)$ ***of the system*** *since both in the experimental environment and in real life it's easier to end up in a "safe" condition in which we can do assumption about the initial conditions, though we are going to discuss more about this in chapter 8.*
The state equation in Discrete Time form is quite simple and is given by:

$$X(k+1) = X(k) + a_1\, U(k) \tag{6.1}$$

where $a_1 = \frac{T_s}{C_n}$.

6.2.2 Output Equation

The output of the system corresponds to the voltage measured at the battery terminals. According to the complete model structure,the output equation in Discrete Time form becomes:

$$\begin{aligned} Y(k) = b_0 + b_1\Big(SoH - X(k)\Big) - k_2\, SoH\, sin\Big(2\pi\, X(k)\Big) + \\ - \left(c_0 + c_1\, \Big(SoH - X(k)\Big) + c_2\, \Big(SoH - X(k)\Big)^2 + k_7\, e^{\left(k_6 \frac{X(k)}{SoH}\right)^8}\right) U(k) \end{aligned} \tag{6.2}$$

and calling:

$$\begin{aligned} b_4 &= b_0 + b_1\, SoH \\ b_5 &= k_2\, SoH \\ c_4 &= c_0 + c_1\, SoH + c_2\, SoH^2 \\ c_5 &= c_1 + 2\, c_2\, SoH \end{aligned}$$

we can reformulate the output equation in a simplified fashion:

$$\begin{aligned} Y(k) = b_4 - b_1\, X(k) - b_5\, sin\Big(2\pi\, X(k)\Big) + \\ - \left(c_4 - c_5 X(k) + c_2\, X(k)^2 + k_7\, e^{\left(k_6 \frac{X(k)}{SoH}\right)^8}\right) U(k) \end{aligned} \tag{6.3}$$

Despite the state equation is linear and quite simple, the model presents strong nonlinearities in the output equation. This aspect has strongly affected the tools that have been exerted in the estimation algorithm generation (chapter 7). We should bear in mind that the SoH, in this phase, still enters the problem as known and given parameter.

6.2.3 Overview

Let's now have a look at the the overall battery model *Simulink* implementation (Figure 6.8) As we can see it's necessary to provide:

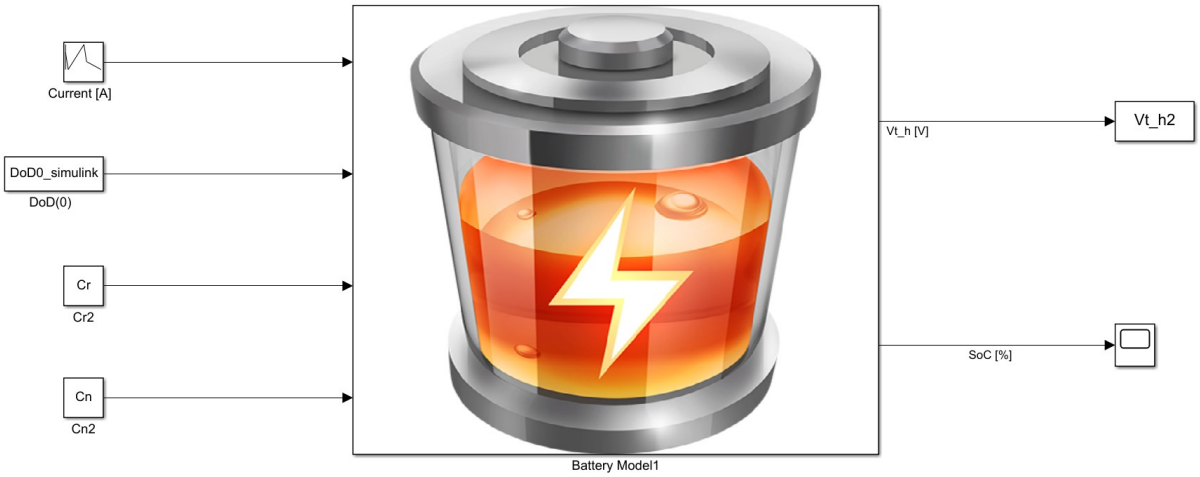


Figure 6.8: Simulink model - High Level

- The Initial State: $DoD(t_0)$ $\Big($or equivalently $SoC(t_0)\Big)$
- The Real Capacity: C_r $\Big($that defines the actual State of Health: $SoH\Big)$
- The Nominal Capacity: C_n

and the system automatically evolves according to the external output.

6.3 Performance

Let's have a closer look to the limits and potentialities of the battery model. In this section are going to be shown the calibration steps and the performance comparison with our experimental benchmark tests.

6.3.1 Calibration

This non-linear model requires the fundamental blocks calibration, in which we set the best-fit parameters to the real data. Starting from the V_{ocv} block, we exploit previous analysis on the "NEW Battery" open circuit voltage curve to set the upper limit and the "OLD Battery" open circuit voltage curve to set the lower limit of the convex combination. Once the main structure is defined we can fine-tune the parameter k_2 to manage the sinusoidal behaviour of the real data and finally perform a check of the model with the "MEDIUM Battery", resulting in: The same procedure

Parameter	**Value**
b_{0_new}	11.70
b_{1_new}	0.018
b_{0_old}	13.05
b_{1_old}	0.040
k_2	0.29

Table 6.1: V_{ocv} model block parameters

holds for the R_{nl} block, resulting in:

Parameter	**Value**
c_{0_new}	0.1300
c_{1_new}	-2.159 e-3
c_{2_new}	2.159 e-5
c_{0_old}	0.4433
c_{1_old}	-8.165 e-2
c_{2_old}	6.8041 e-3
k_6	12
k_7	1 e-4

Table 6.2: R_{nl} model block parameters

In chapter 8 we'll introduce some insight to automate the calibration process exploiting particular initial condition and discharge/charge/rest sessions.

6.3.2 Test

The following plots show the quality of the model for the battery in the three main conditions: New, Medium, Old.
IMPORTANT: *The input signal it's processed by a moving median filter with a window length of 10s in order to avoid both noise and abrupt current variation.*
Thus, we can clearly see that if on one hand the model is working extremely well for this battery in *new* and *medium* condition, on the other it's not perfect in the *old* one, while keeping remarkable performance. This little modelling error can be easily accepted since in real-life condition going so deep with the discharge could result in serious battery damage, mainly for lead-acid batteries.

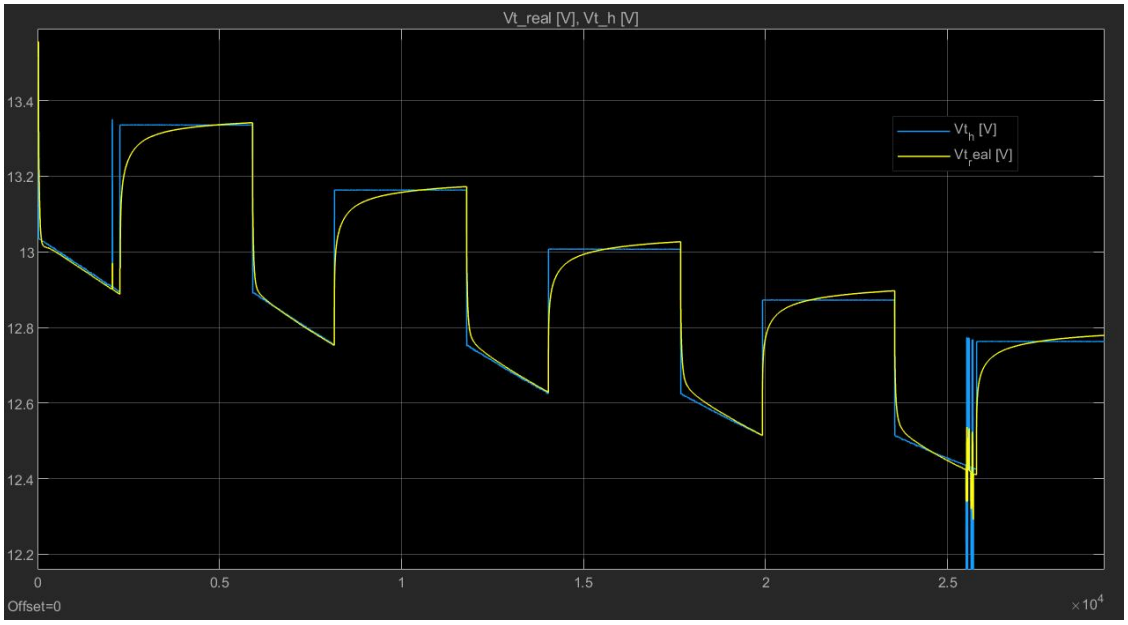


Figure 6.9: Model performance - Battery: new

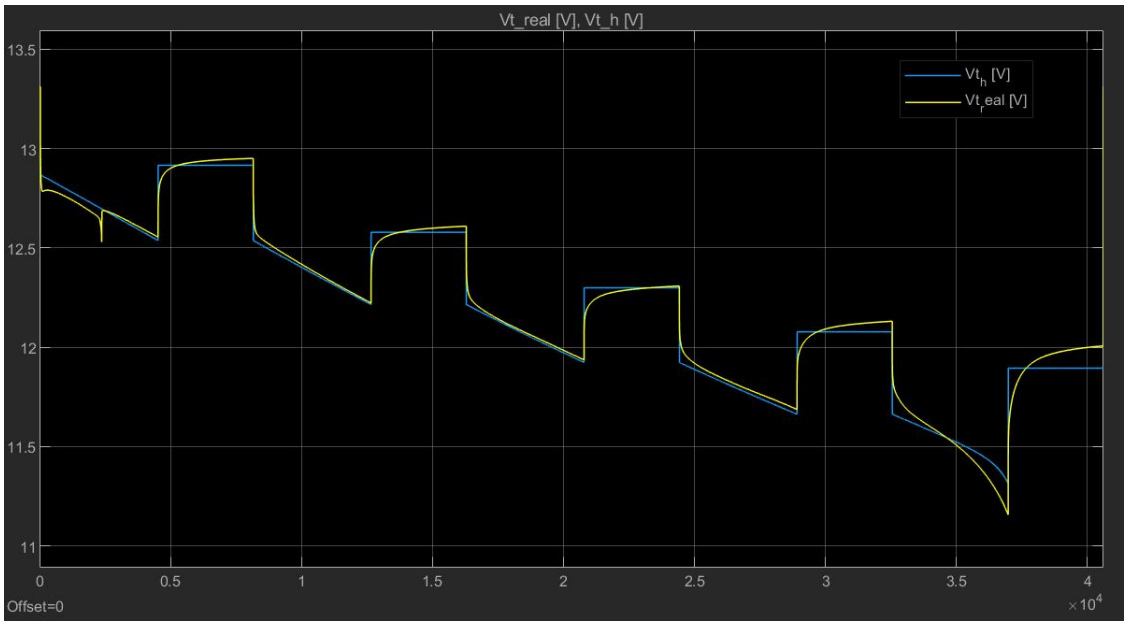


Figure 6.10: Model performance - Battery: medium

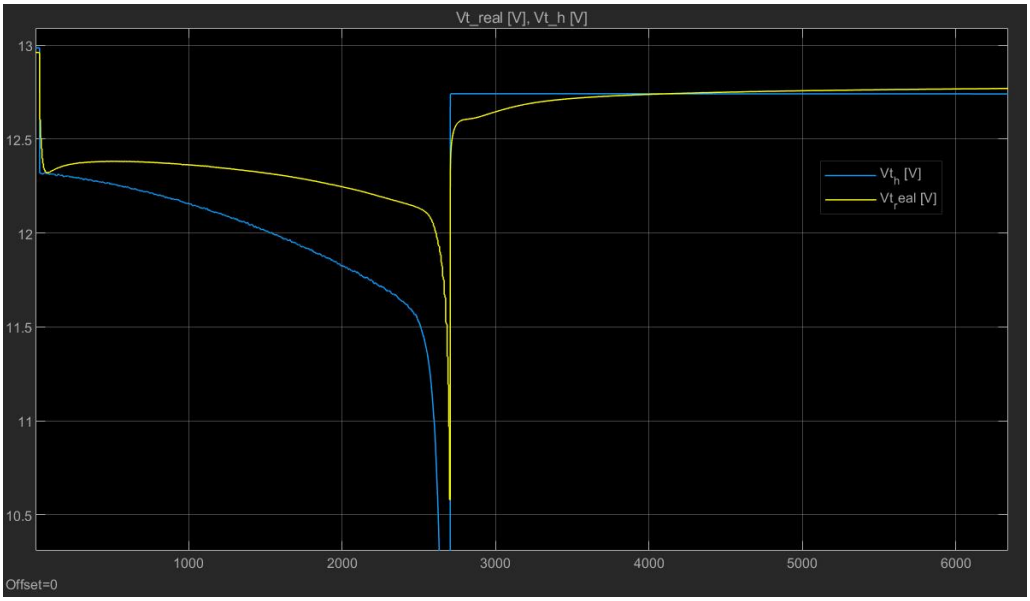


Figure 6.11: Model performance - Battery: old

Chapter 7

Model-based Solution

7.1 Single Non-Linear State Observer

As former step towards the final solution we reckon the Real Capacity C_r to be known and we'll see that this assumption it's plausible in the logic of the complete algorithm. The goal in this phase is to be able to retrieve the state initial conditions by means of a non-linear state observer. Among several algorithm with different complexity, the most robust, used and validated is the Extended Kalman Filter (EKF).

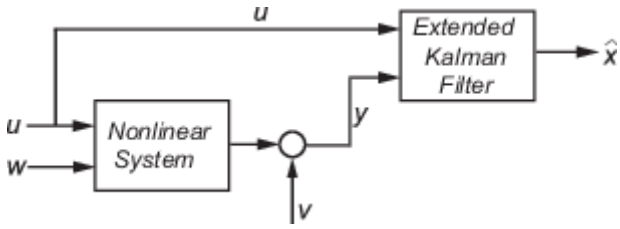


Figure 7.1: Extended Kalman Filter, [17]

7.1.1 Extended Kalman Filter

The Extended Kalman Filter is a model-based non-linear online state estimator. Assuming that the state transition and output equations for a discrete-time nonlinear system have non-additive process and measurement noise terms with zero mean and covariance matrices Q and R, respectively:

$$\begin{cases} x(k+1) = f\Big(x(k), u(k)\Big) + w(k) \\ y(k) = h\Big(x(k), u(k)\Big) + v(k) \end{cases}$$

$$w(k) \approx \Big(0, Q(k)\Big)$$

$$v(k) \approx \Big(0, R(k)\Big)$$

Where $f(\cdot)$ is a nonlinear state transition function that describes the evolution of states x from one time step to the next. The nonlinear measurement function $h(\cdot)$ relates x to the measurements y at time step k. The process and measurement noise are w and v, respectively. The covariance matrices Q and R are to be provided

and act like model tuning parameters. In the following we are going to describe the fundamental steps of this algorithm while more can be found in [25]:

1) Filter Initialisation

We initialise the state $x(0)$ and the state estimation error covariance matrix P:

$$\hat{x}(0|-1) = E[x(0)]$$

$$P(0|-1) = E[\big(x(0) - \hat{x}(0|-1)\big)\big(x(0) - \hat{x}(0|-1)\big)^T]$$

where $\hat{x}(0|-1)$ is the best guess of the state value before you make any measurements.

Then, for each time steps k=0,1,2...

2.1) Correction

Compute the analytical Jacobian of the measurement function h:

$$C(k) = \frac{\partial h}{\partial x}\Big|_{x=\hat{x}(k|k-1)}$$

$$S(k) = \frac{\partial h}{\partial v}\Big|_{x=\hat{x}(k|k-1)} = \mathbf{I}_n$$

update the kalman gain K:

$$K(k) = P(k|k-1)C(k)^T\left(C(k)P(k|k-1)C(k)^T + S(k)R(k)S(k)^T\right)^{-1}$$

and eventually update the state x and state estimation error covariance P using the measured data $y(k)$:

$$\hat{x}(k|k) = \hat{x}(k|k-1) + K(k)\Big(y(k) - \hat{y}(k)\Big)$$

$$\hat{P}(k|k) = P(k|k-1) + K(k)C(k)P(k|k-1)$$

where $\hat{y}(k) = h\Big(\hat{x}(k|k-1), 0, u(k)\Big)$

2.2) Prediction

Compute the analytical Jacobian of the state transition function f:

$$A(k) = \frac{\partial f}{\partial x}\Big|_{x=\hat{x}(k|k)}$$

$$G(k) = \frac{\partial f}{\partial w}\Big|_{x=\hat{x}(k|k)} = \mathbf{I}_n$$

and update the value to be fed back again in the corrector section:

$$P(k+1|k) = A(k)P(k|k)A(k)^T + G(k)Q(k)G(k)^T$$

$$\hat{x}(k+1|k) = f\Big(\hat{x}(k|k), 0, u(k)\Big)$$

7.1.2 Implementation

In the early stage of the prototyping phase we've decided to exploit the Simulink library to rapidly implement and test the Extended Kalman Filter. As shown in the diagram we have the main block that receives the real output data $y(k)$ and estimates the internal state of the system $\hat{x}(k)$, then, we have two simulink functions that respectively represent $\hat{x}(k+1|k) = f\Big(\hat{x}(k|k), 0, u(k)\Big)$ and $\hat{y}(k) = h\Big(\hat{x}(k|k-1), 0, u(k)\Big)$. Eventually we have also added another couple of Simulink functions that represent the jacobians of $J_f(\cdot)$ and $J_h(\cdot)$ so to reduce the computational effort (non strictly mandatory).

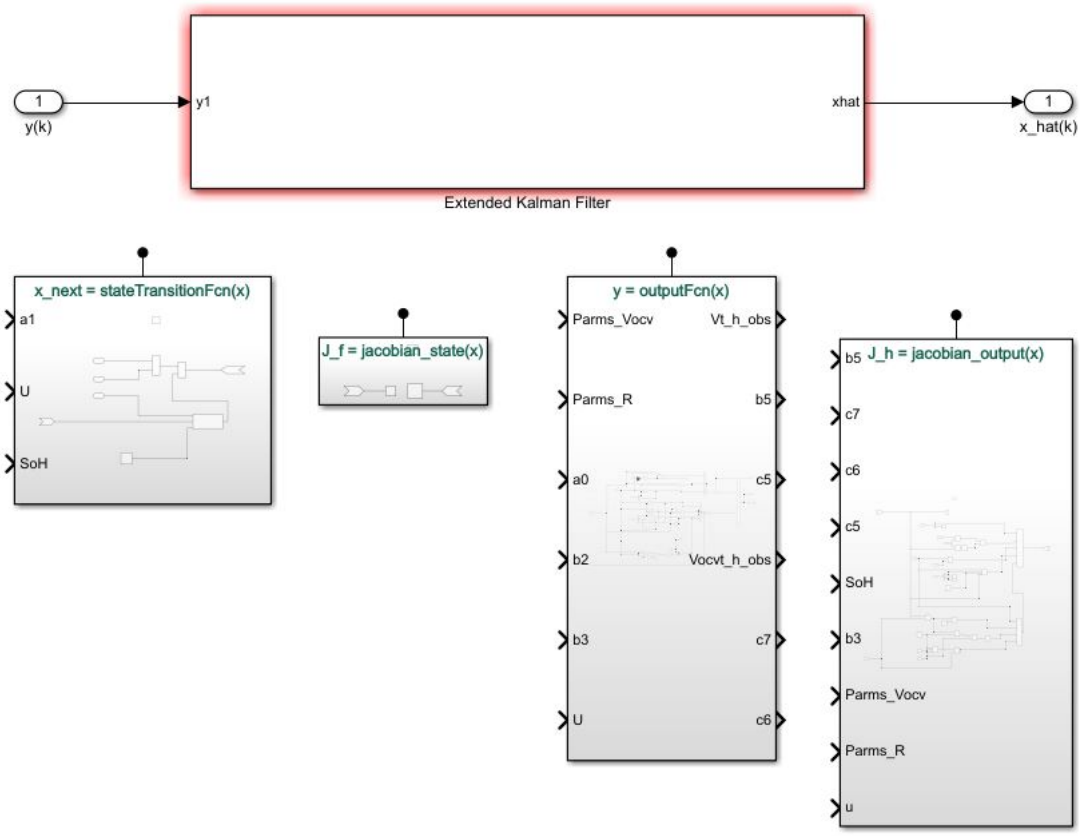


Figure 7.2: Extended Kalman Filter - *Simulink* implementation

The last step includes both the configuration of the initial conditions in terms of:

- initial state $\hat{x}(0)$
- initial covariance $P(0)$

and the setting of the uncertainties:

- model uncertainties covariance $Q(0)$
- measurement uncertainties covariance $R(0)$

That results in:

Parameter	Value
$\hat{x}(0)$	10
$P(0)$	100
$Q(0)$	10
$R(0)$	1

Table 7.1: EKF parameters

Content Sources

Simon, [25];
Mathworks, [16];
Mathworks, [15];
Mathworks, [17];

7.1.3 Real data performance

Let's now have a look at the estimation performances of the implemented EKF. In this section the estimation algorithm is tested on our real data set for the battery in the three main health conditions: new (Figure 7.3), medium (Figure 7.4), old (Figure 7.5).

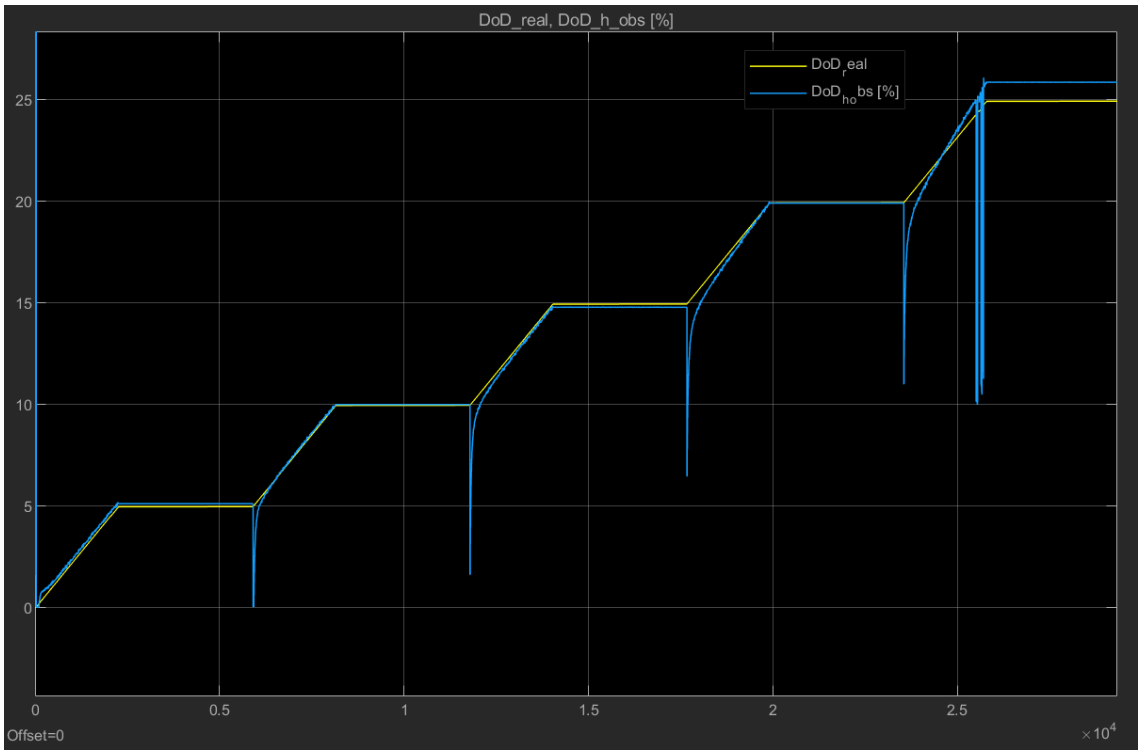


Figure 7.3: Extended Kalman Filter - Performance: New

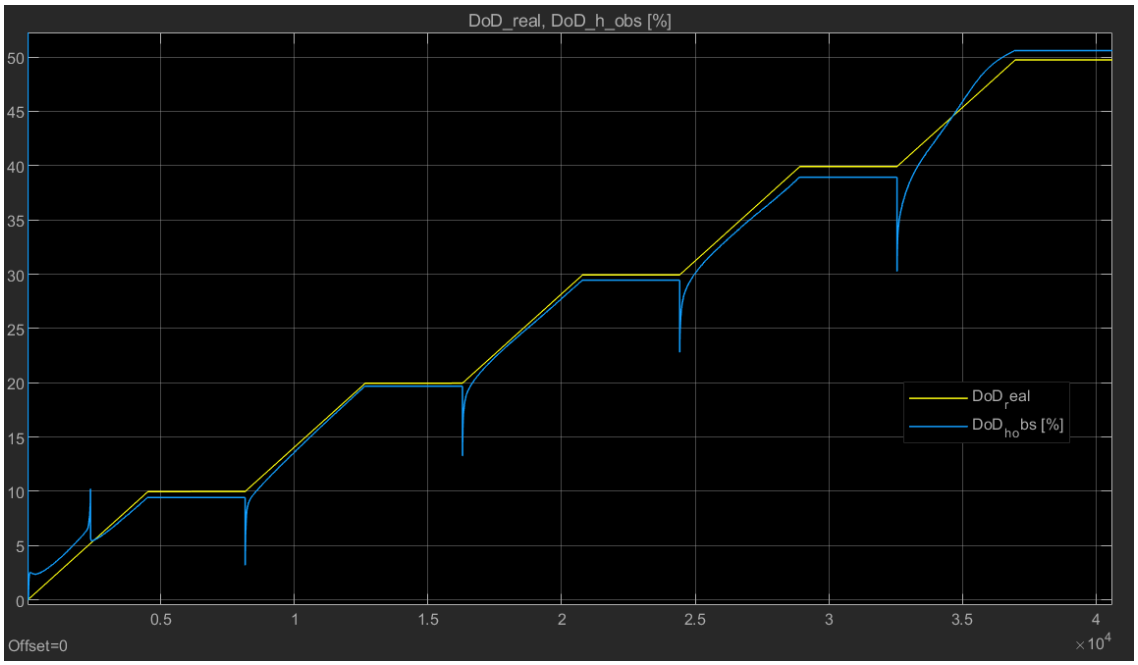


Figure 7.4: Extended Kalman Filter - Performance: Medium

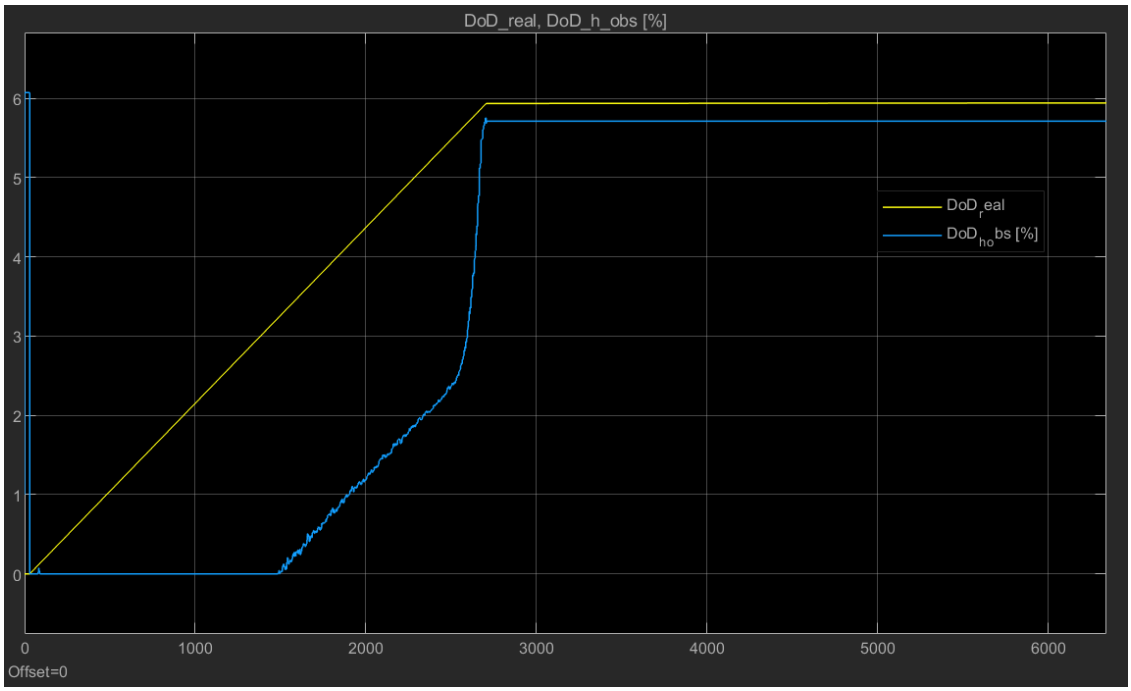


Figure 7.5: Extended Kalman Filter - Performance: Old

In the plots we can see the real battery state $DoD(t)$ (yellow line), and the EKF estimated state $\hat{DoD}(t)$(blue line). This algorithm reaches quite good results, yielding to average absolute estimation errors $\leq 2\%$ with great convergence times for this slow dynamics. The relatively lower performances in the old battery (3% max error) are mainly due to the modelling approximation.

IMPORTANT: *Thanks to the experimental session shown in subsection 5.3.2, in which we have measured the real capacity of the battery, and thanks to the created*

energetic framework (chapter 4), the yellow line can be considered as "absolute" in terms of status. This means that the estimation error with respect to this line can be considered "absolute".

7.2 Real-Time Batteries' SoC and SoH Estimator

At this stage we have just set the basis for the implementation of the final model-based solution. Indeed, as seen in subsection 6.2.3, given:

- the real capacity C_r
- the state initial condition $DoD(0)$

we have built a working approximated battery model.
In subsection 7.1.1, we've exploited this model and given only:

- the real capacity C_r

we have built a model-based non-linear state observer (EKF) to retrieve the state initial condition $DoD(0)$ (or $SoC(0)$ equivalently). Yet the real capacity C_r is unknown during the standard battery life-cycles and is exactly one of the most crucial information to extract. In order to overcome this problem we came out with a more complex solution that at the highest possible level of abstraction can be described as a series of n augmented non-linear state estimators (ANSE), each of which supplied with n possible guess in the span:

$$C_{r\_i} \in [0, C_n] \quad i = 1, ..., n$$

and that generate peculiar error signal that are eventually managed by a logic to find the "optimum", that is the "best" estimated real capacity $\hat{C}_{r\_best}$, and the relative real-time state estimation $\hat{DoD}(t)$ (or equivalently $\hat{SoC}(t)$).

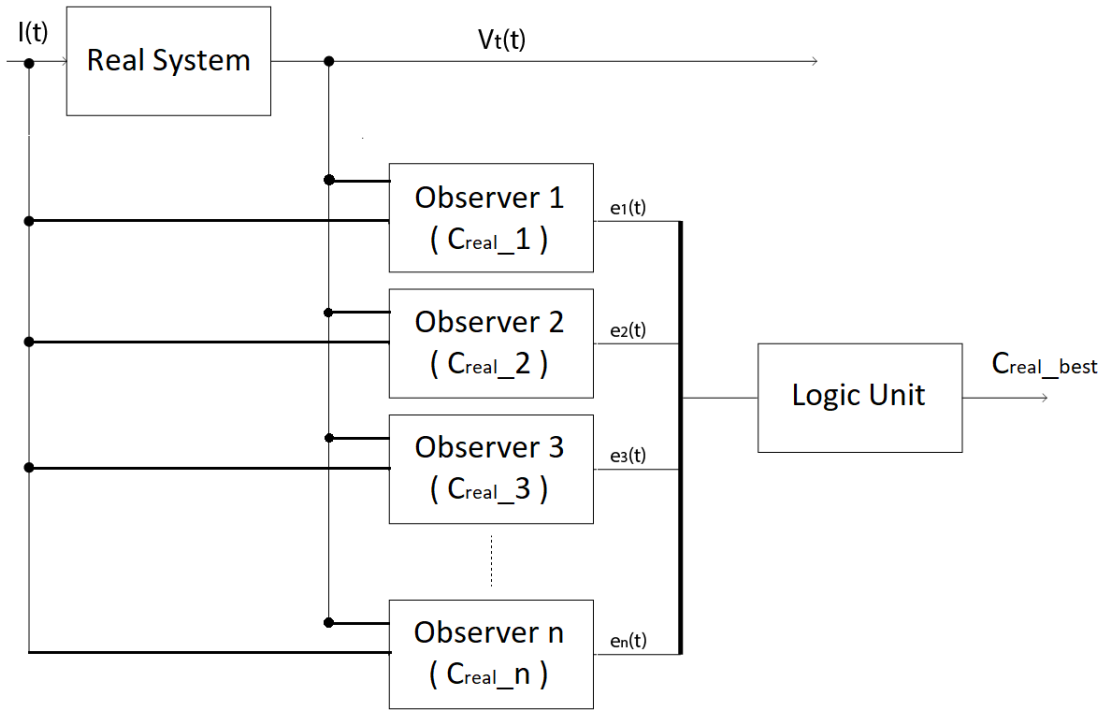


Figure 7.6: High level algorithm description

Hence the final solution is made up of two main macro-blocks:

- a series of n ANSE (Augmented Non-Linear State Estimators)
- a signal logic unit

Let's now have a look at the nitty-gritty of these elements.

7.2.1 Series of ANSE

The acronym ANSE stands for Augmented Non-linear State Estimator and it is a structure that we've ideate to generate useful error signals. It corresponds to the pulsing core of the final solution and the structure is shown in Figure 7.7.

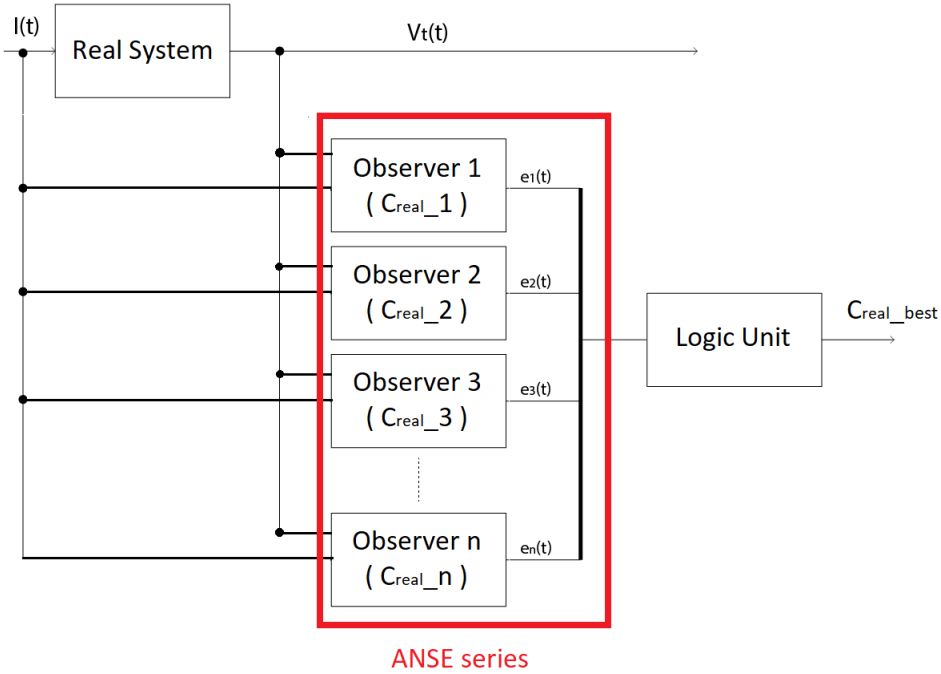


Figure 7.7: Series of n ANSE (Augmented Non-linear State Estimators

The basic idea is that of fixing one of the two unknowns (C_r and $DoD(t)$) and parallelize n discrete possible "solutions" to find the "best". In our case the dynamic of the real capacity C_r is order of magnitudes slower than the one of the $DoD(t)$ and therefore it perfectly lends itself for the algorithm. Each ANSE (Figure 7.8) mainly consists of:

- $n°1$ Non-linear State Observer (EKF in this case)
- $n°1$ Open Loop Battery Model
- $n°2$ Sample and hold

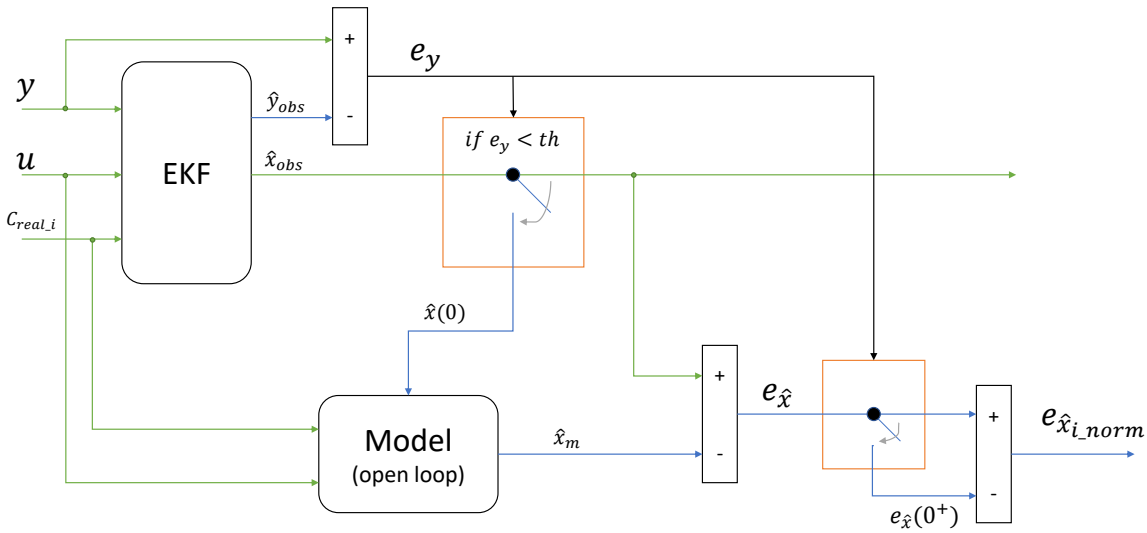


Figure 7.8: ANSE (Augmented Non-linear State Estimators)

Working Principle

The ith observer receives as input:

- the input signal of the system $U(k)$
- the output signal of the system $Y(k)$
- the ith guessed real capacity $\hat{C}_{r\_i}$

and yield:

- the estimated state of the system $\hat{x}_{obs}(k)$
- the estimated output signal of the system $\hat{Y}_{obs}(k)$

We define the *observer output tracking error* as the difference between the real output $Y(k)$ and the estimated output $\hat{Y}_{obs}(k)$:

$$e_{Y\_obs}(k) = Y(k) - \hat{Y}_{obs}(k) \tag{7.1}$$

and when at the generic time instants t_1 the error $e_{Y\_obs}(k)$ goes below a fixed threshold th_{obs}:

$$e_{Y\_obs}(k) < th_{obs} \qquad when \;\; k = t_1$$

the estimated state of the system $\hat{x}_{obs}(t1)$ is sampled, hold and provided as initial condition of the Open Loop Battery Model that from now on will evolve autonomously. The model produces:

- the estimated state of the system $\hat{x}_m(k)$
- the estimated output signal of the system $\hat{Y}_m(k)$

We define the *estimated state tracking error* as the difference between the state estimated from the observer $\hat{x}_{obs}(k)$ and the state estimated from the model $\hat{x}_m(k)$:

$$e_{\hat{x}_i}(k) = \hat{x}_{obs}(k) - \hat{x}_m(k) \tag{7.2}$$

This error defines the difference in the evolution between the initialised open-loop model and the observer. However, what really matter to us is not the absolute value in itself but the relative error with respect to the value assumed in the neighbourhood of the convergence instant t_1:

$$e_{\hat{x}_i}(t_1 + \delta\tau)$$

Therefore, we define the *normalised estimated state tracking error* as the difference between the current *estimated state tracking error* and its value at time t_1:

$$e_{\hat{x}_i\_norm}(k) = e_{\hat{x}_i}(k) - e_{\hat{x}_i}(t_1 + \delta\tau) \tag{7.3}$$

This process let us to generate a series of n comparable signals whose associated physical meaning is that of measuring how much the observer changes "the internal parameters" to follow the output signal. The less the movement the more accurate is the initial guess about $\hat{C}_r$. The algorithm is extendable according to the computational power at your disposal. The higher the number n of ANSE, the more precise is the joined estimation. However, an important increase of estimators could not lead to a correspondent rise of performance due to modelling uncertainties.

7.2.2 The Logic

The logic block aims at deciding which of the n parallel guesses is the optimal one. In order to accomplish this result, this item it's made up of two parts:

- a signal conditioning section (Figure 7.9)
- a comparator (Figure 7.10)

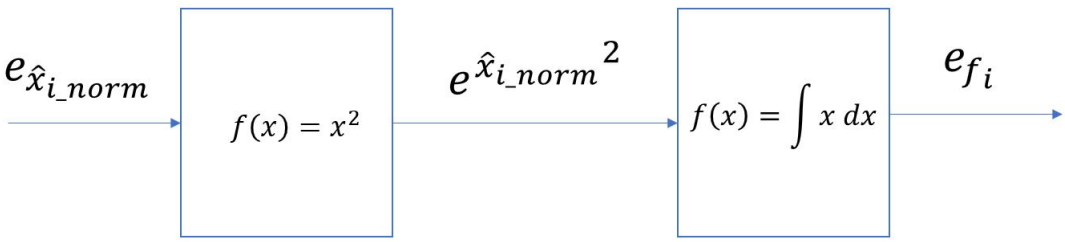


Figure 7.9: Signal Conditioning

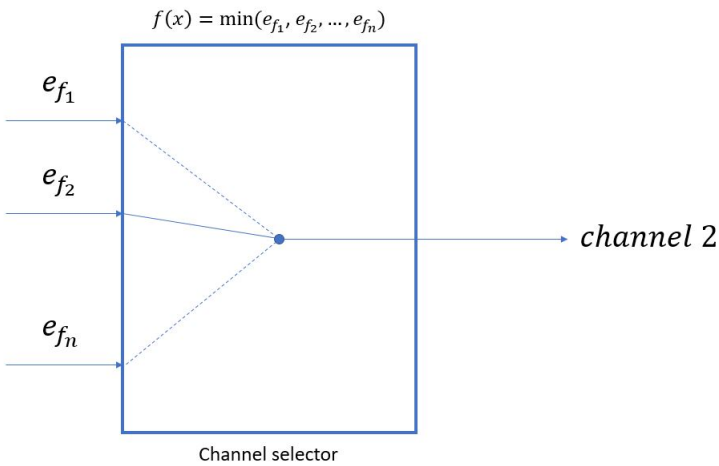


Figure 7.10: Comparator

Working Principle
All the n *normalised estimated state tracking error signal* are squared to avoid sign problems and then integrated over time to include the history:

$$e_{f_i}(k) = \sum_{j=t_2}^{k} e_{\hat{x}_i\_norm}(j)^2 \cdot \delta\tau \tag{7.4}$$

where t_2 represent each instant in which the value of one of the *observer output tracking error* goes below the threshold th_{obs}. This time instants are of paramount importance because also tells when to reset the integral values. The comparator simply takes the minimum of the n error signals and assign to it the "optimal" label.

7.2.3 Algorithm Performances

Let's have a look at the tracking performances.

How to interpret the results

As we have seen this model acts like a switch among n different initial guesses. In this prototyping phase we have decided to make a test with **3 internal parallel models** so initialised (Table 7.2). In channel 0, therefore, we always have the correct

Channel	New	Med	Old
0	$C_{real\_new} \approx 0.95 \cdot C_n$	$C_{real\_med} \approx 0.50 \cdot C_n$	$C_{real\_old} \approx 0.10 \cdot C_n$
1	$0.75 \cdot C_n$	$0.75 \cdot C_n$	$0.75 \cdot C_n$
2	$0.50 \cdot C_n$	$0.25 \cdot C_n$	$0.50 \cdot C_n$

Table 7.2: Final Algorithm: initialisation values

model initialisation, while in the other two channels there are other wrong model choices. The goal of our algorithm would be the one of having the *channel* 0 selected as soon as possible and maintained as long as possible during the discharge session.

If so, we will pick the estimated internal state $\hat{x} = DoD(t)$ coming from the model associated to channel 0 yielding to a **complete joint SoC and SoH estimation**. We will eventually compare the absolute state evolution (experimentally measured in(chapter 4), with the estimated one to find the **absolute error**. The same hold for the output signal V_t.

The results: New battery

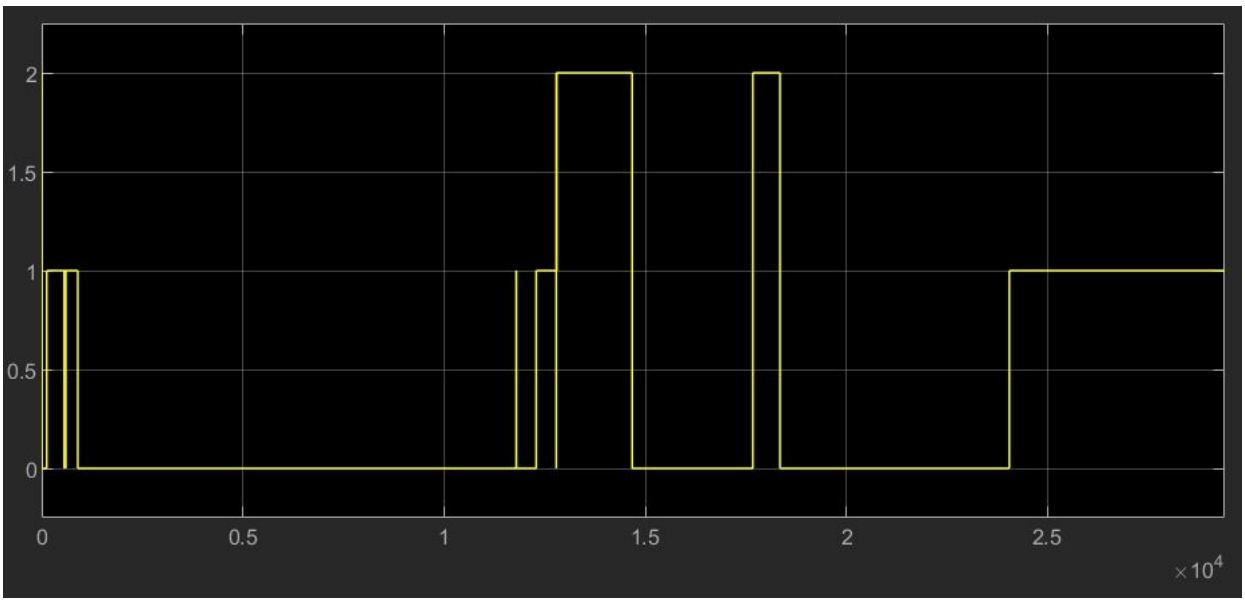


Figure 7.11: Channel selection, battery: new

As we can see, the performance in terms of channel selection for the battery in good condition are pretty interesting: over 70% of the time, the channel 0 remains selected.

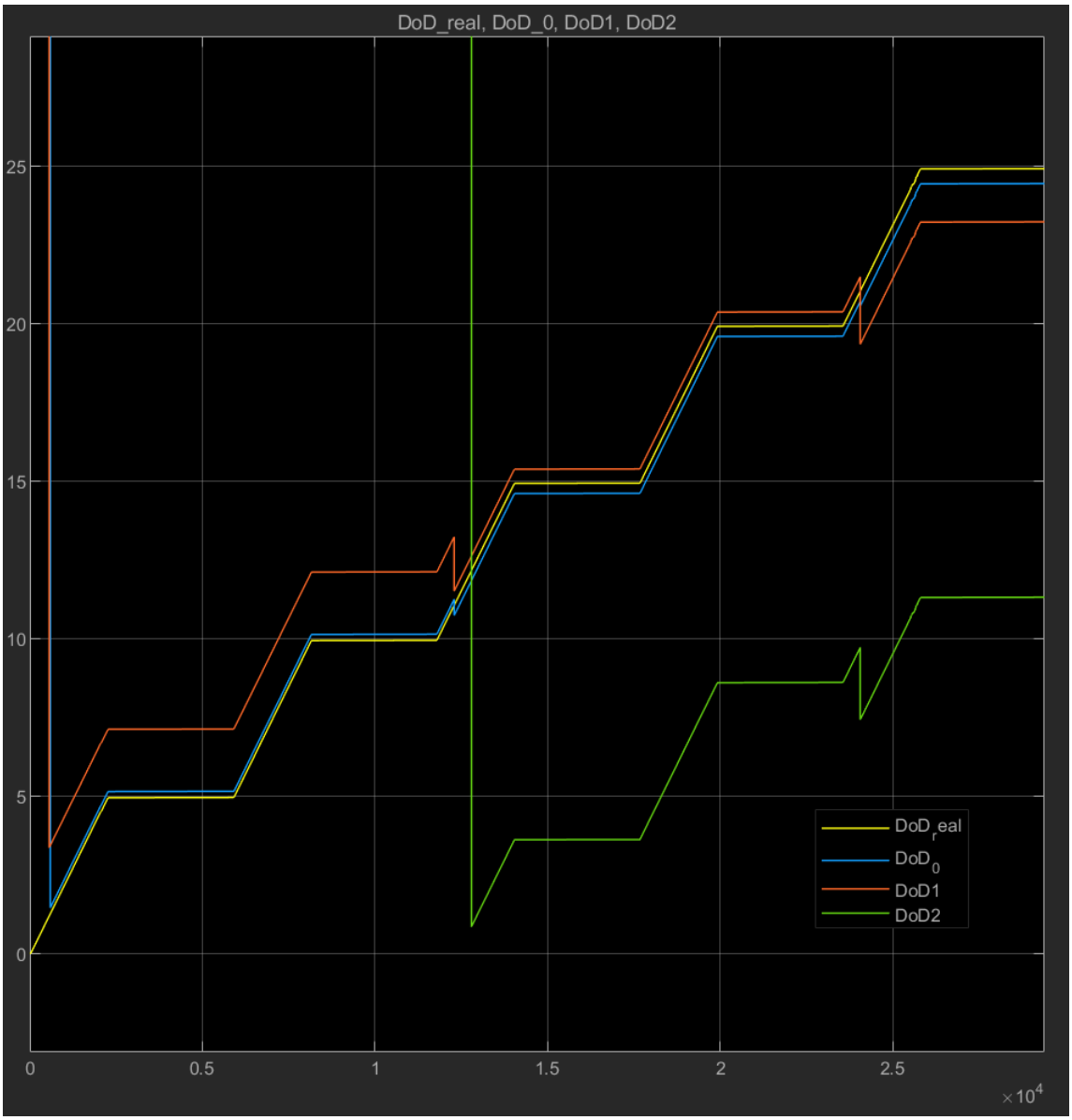


Figure 7.12: Internal state identification, battery: new

This directly translates in an internal state estimation absolute error $\leq$ 1-2 % along all the discharge section. Blue (ch. 0) and yellow (real) lines are almost superimposed. **IMPORTANT:** *Thanks to the experimental session shown in subsection 5.3.2, in which we have measured the real capacity of the battery, and thanks to the created energetic framework (chapter 4), the yellow line can be considered as "absolute" in terms of status. This means that the estimation error with respect to this line can be considered "absolute".*

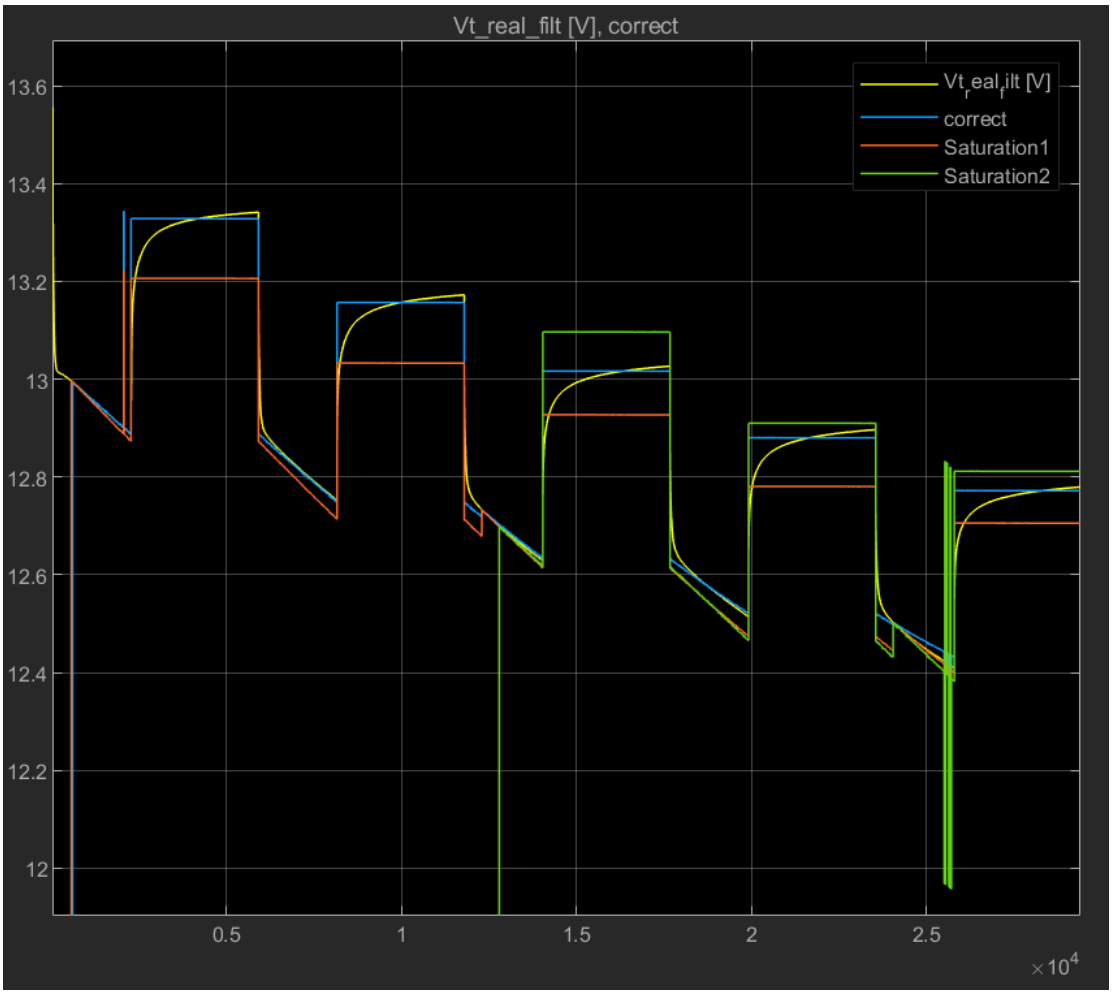


Figure 7.13: Modelling error: terminal voltage V_t, battery: new

As expected, from the modelling side too, the best model is the one associated to channel 0 (blue line).

The results: Medium battery

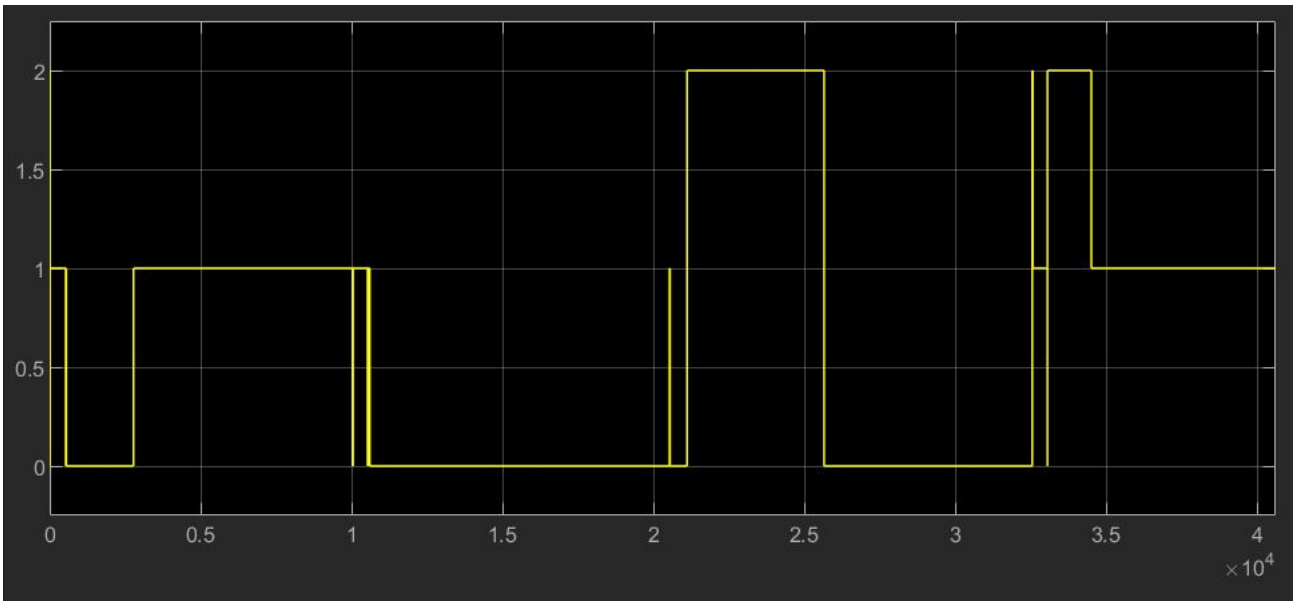


Figure 7.14: Channel selection, battery: med

In this this case, channel selection performance for the battery in medium health condition yields channel 0 to remain selected over 50% of the time. In this case the convergence conditions takes up to 1e4s, but this is mainly due to a non perfect signal (in terms of estimation).

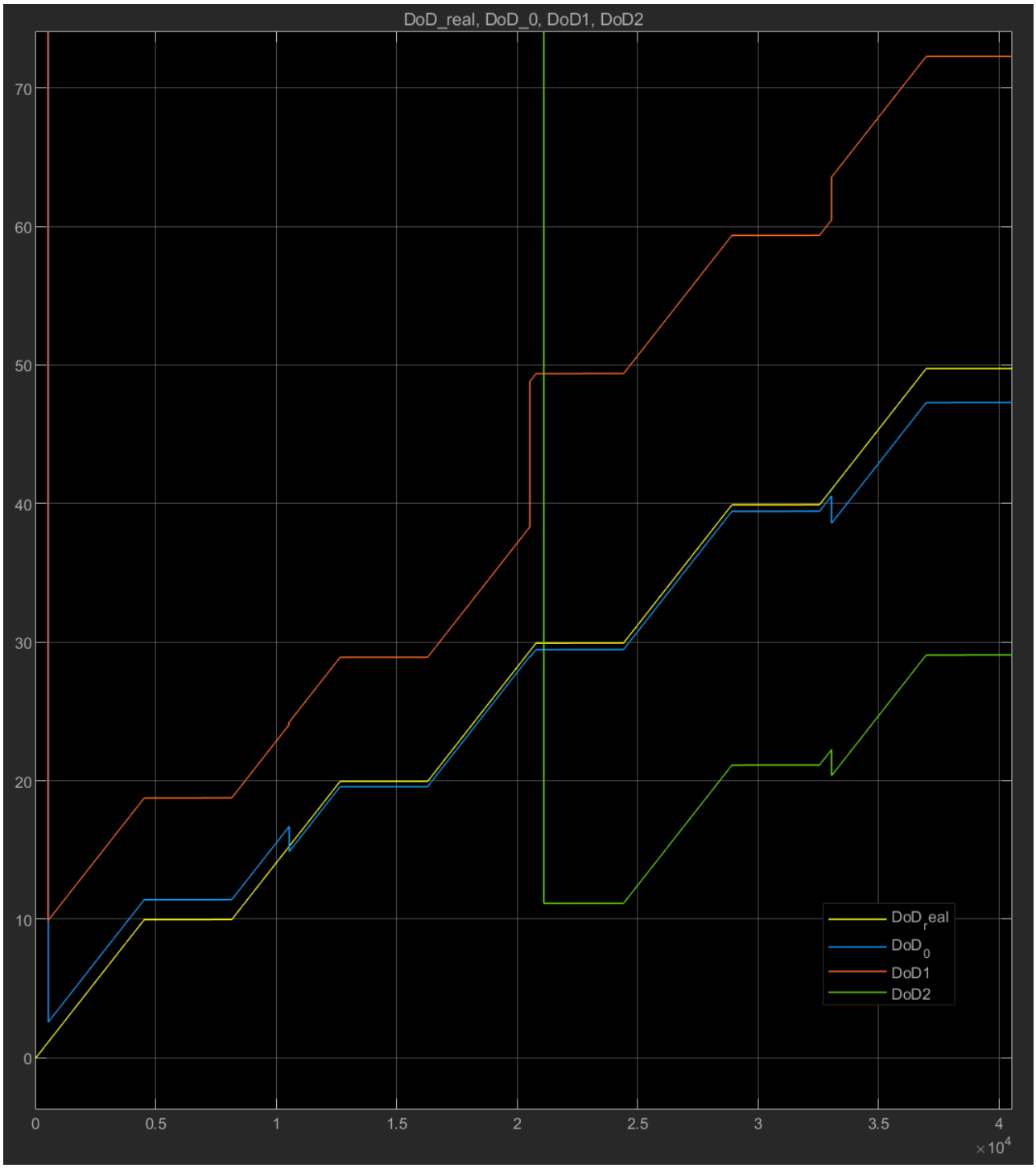


Figure 7.15: Internal state identification, battery: med

This directly translates in an internal state estimation absolute error $\leq$ 2-4 % along all the discharge section. Blue (ch. 0) and yellow (real) lines are almost superimposed. **IMPORTANT:** *Thanks to the experimental session shown in subsection 5.3.2, in which we have measured the real capacity of the battery, and thanks to the created energetic framework (chapter 4), the yellow line can be considered as "absolute" in terms of status. This means that the estimation error with respect to this line can be considered "absolute".*

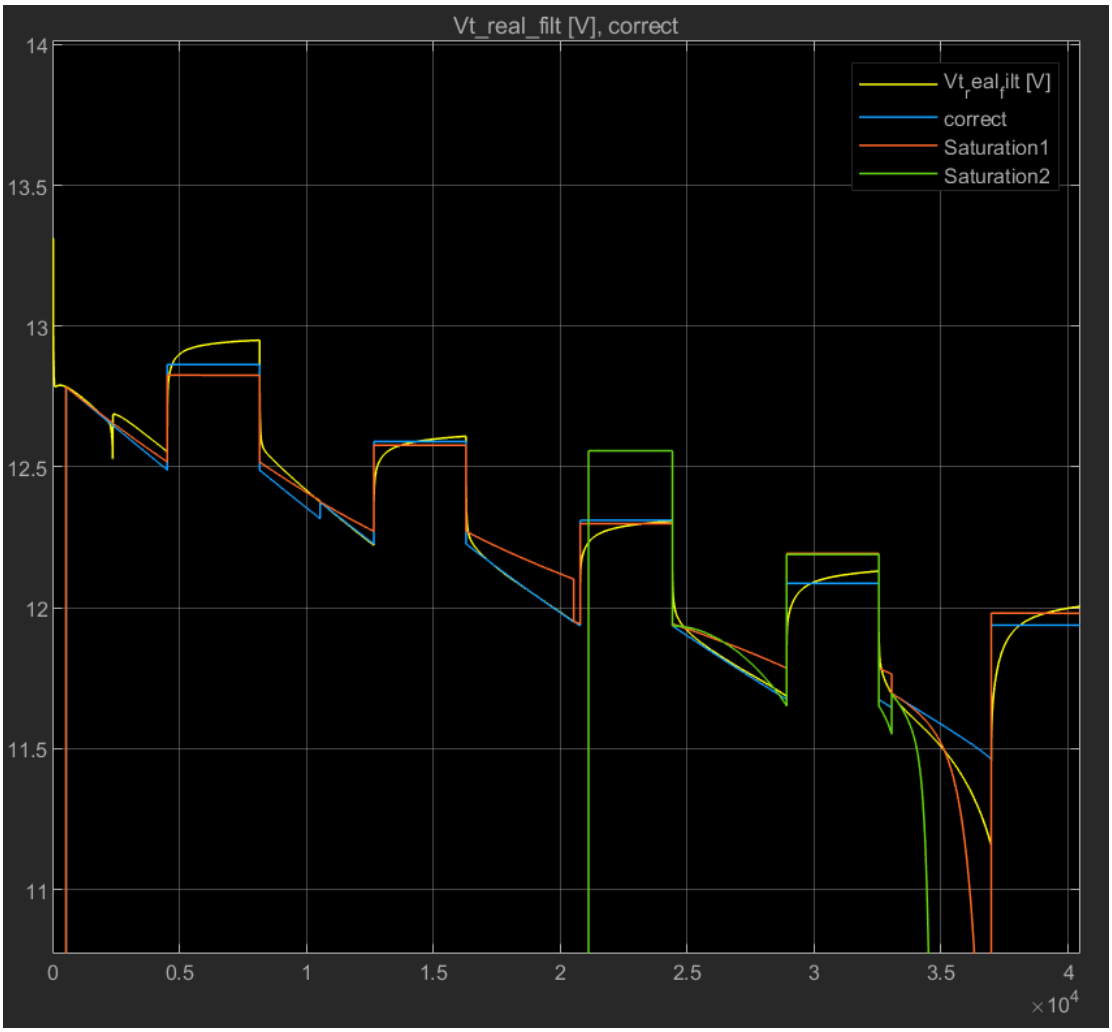


Figure 7.16: Modelling error: terminal voltage V_t, battery: med

As expected, from the modelling side too, the best model is the one associated to channel 0 (blue line).

The results: Old battery

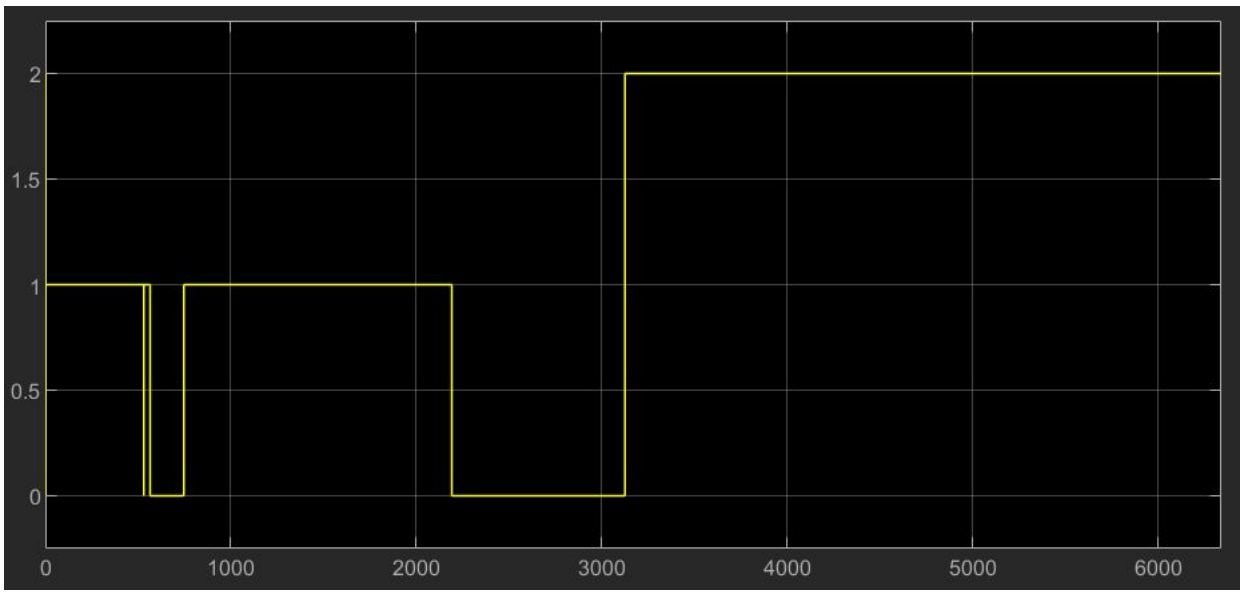


Figure 7.17: Channel selection, battery: old

In this this case, channel selection performance for the battery in old health condition yields channel 0 to remain selected a less than 50% of the time. In this case the convergence conditions takes up to 2e3s, but this is mainly due to a non perfect signal (in terms of estimation). The final jump to channel 2 should not be taken into account because it's a zero input zone where the identification does not work. Also in this case the signal is not perfect for the identification purposes.

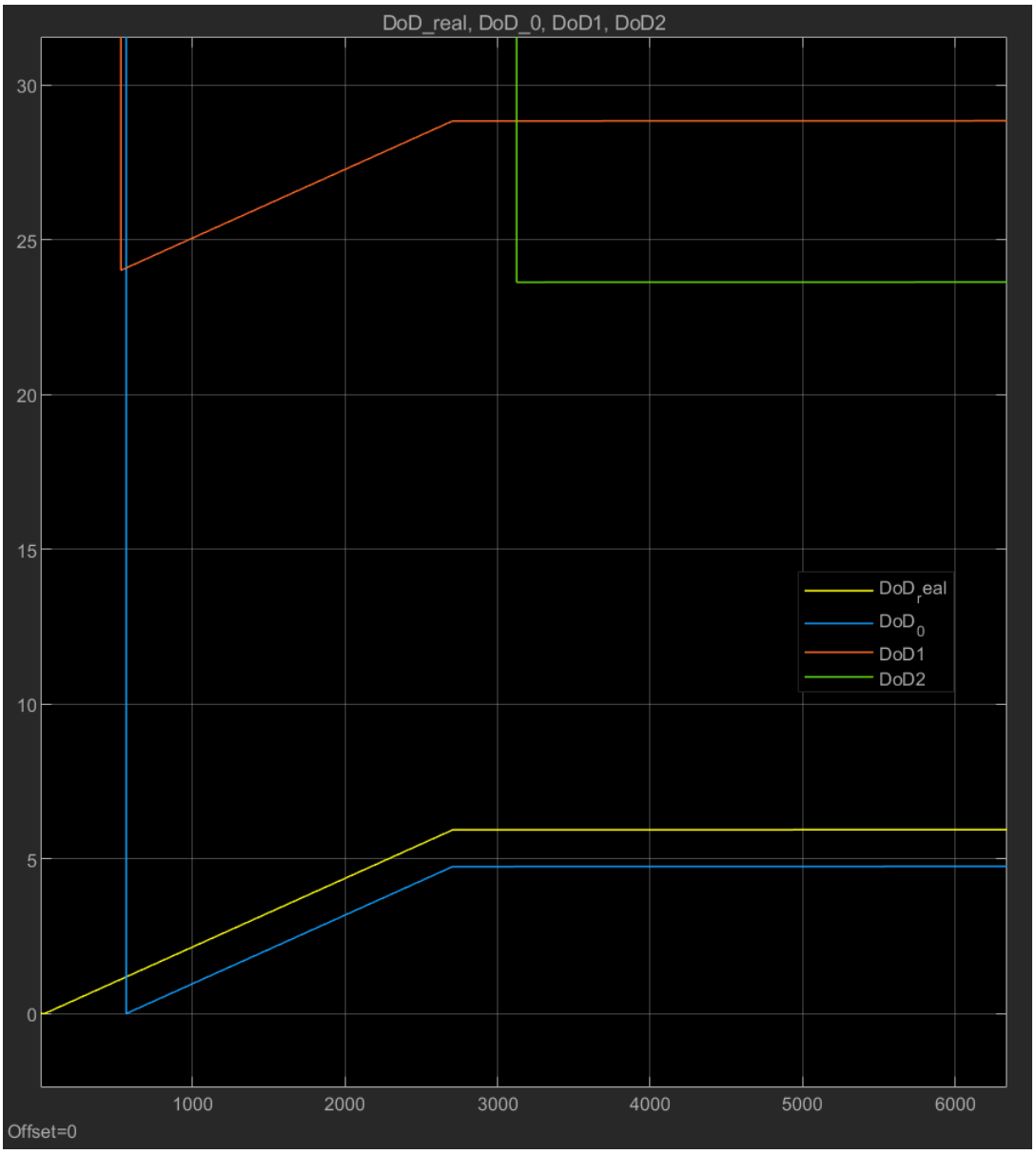


Figure 7.18: Internal state identification, battery: old

This directly translates in an internal state estimation absolute error $\leq$ 3-5 % along all the discharge section. Blue (ch. 0) and yellow (real) lines are almost superimposed. **IMPORTANT:** *Thanks to the experimental session shown in subsection 5.3.2, in which we have measured the real capacity of the battery, and thanks to the created energetic framework (chapter 4), the yellow line can be considered as "absolute" in terms of status. This means that the estimation error with respect to this line can be considered "absolute".*

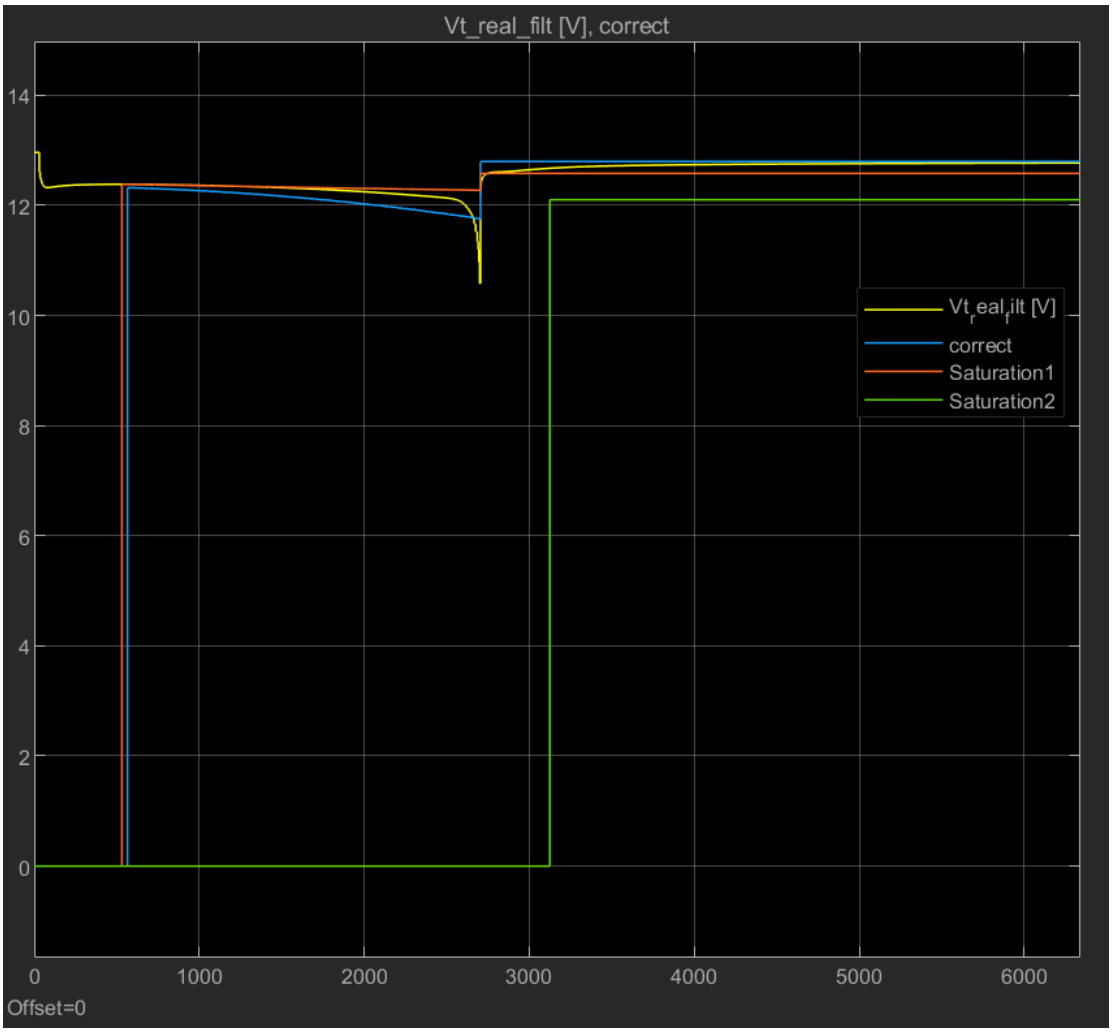


Figure 7.19: Modelling error: terminal voltage V_t, battery: old

As expected, from the modelling side too, the best model is the one associated to channel 0 (blue line).

Chapter 8

Conclusions

The whole work focuses on replying to a question that results as simple as it is complex: *how much is my battery charge?*
While I was conducting the early stage research and managing the first bunch of experimental results, I've experienced a sense of ambiguity in the interpretation of the energetic quantities. What does it means that a battery is fully charged? Which is it's associated *State of charge*? Considering a battery fully charged from which you can extract only the 35% of the nominal capacity, how much is it's SoC? 100%, 35%? etc.
The SOC and SoH definitions that you can find in literature are correct but not complete and, above all, not connected. There was a lack of an energetic framework able to explain with a mathematical description, the macro energetic exchanges and the deterioration of the battery. This forced me to keep open several interpretation as seen in section 4.2, giving rise to the *Measurable Quantity Analysis*. Then fortunately, I came across [14], in which new concepts such as *releasable capacity* where used. Taking a clue from that article I've built the Energetic Framework (chapter 4): a mathematical description of the basic working principle and energetic exchanges of the batteries. With such a solid foundation we've then identified, created and calibrated a battery model that has been exploited from the model-based estimation algorithm to find the real-time joint SoC and SoH estimation.

Future Release

The major limits we can find, consist in the model calibration. However in future works one could exploit limit conditions such as:

- full charge
- full discharge
- new battery: SoH=1

to fix some variables to perform online parameter identification. Moreover the model could be enriched with temperature dependency and so on. Alternatively, the logic could also be replaced by an optimisation algorithm that with data-driven techniques could be trained to choose the best channel of the model algorithm.

Appendix A

Appendix

A.1 Matlab Code

A.1.1 Open Circuit Voltage Characteristic

```
clc, clear all, close all
run('DATA_M2_Bianconiglio');
for i = 1:Numero_Esperimenti_M2_new_new
    isout=0;
    for k = 1:Sezioni_Esperimenti_M2_new_new(i)
            data=['SoCSafeMon-FIAMM-L150P-New+_0' num2str(k)
                '_' num2str(i) '.mat'];
            load(data)

             %Outlier Correction
            if isout == 0 & k == 1
                Chg_err = 13.5-Vbat(1);
                if Chg_err > 0
                    Vbat = Vbat + Chg_err;
                    isout = 1;
                end
            elseif isout == 1
                Vbat = Vbat + Chg_err;
            end

            y_new_new_M2(k,i) = mean(Vbat(10:3000)); %Vocv
            x_new_new_M2(k,i) = SoC(1); %SoC
    end
end

idx=0;
for i = 1:Numero_Esperimenti_M2_new
    isout = 0;
    for k = 1:Sezioni_Esperimenti_M2_new(i)
            data=['SoCMon-FIAMM-L150P-New_0' num2str(k) '_'
               num2str(i) '.mat'];
```

```
            load(data)

            %Outlier Correction
            if isout == 0 & k == 1
                Chg_err = 13.1-Vbat(1);
                if Chg_err > 0
                    Vbat = Vbat + Chg_err;
                    isout = 1;
                end
            elseif isout == 1
                Vbat = Vbat + Chg_err;
            end
            y_new_M2(k,i) = mean(Vbat(10:3000)); %Vocv
            x_new_M2(k,i) = SoC(1); %SoC


            idx=idx+1;
            y_M2(idx) = y_new_M2(k,i);
            x_M2(idx) = x_new_M2(k,i);
    end
end

for i = 1:Numero_Esperimenti_M2_old
    for k = 1:Sezioni_Esperimenti_M2_old(i)
            data=['SoCMon-FIAMM-L150P-Used_0' num2str(k) '_'
                num2str(i) '.mat'];
            load(data)

            y_old_M2(k,i) = mean(Vbat(10:3000)); %Vocv
            x_old_M2(k,i) = SoC(1); %SoC

    end
end

Characteristic computed with the MEDIUM battery and then
   tuned according to the new and old battery:
%linear regression (1st order)
x_M2
y_M2

ao=1; %approximation order
N=length(x_M2);
phi=vander(x_M2);    % Vandermonde Matrix: A(i,j)=x_1(i)^(n
   +1?j)
phi=phi(:,N-ao:end);

```

```
c_M2=phi\y_M2';  %LS -> estimated coefficients vector from y_=phi(x_)*c
a_M2=c_M2(1)
b_M2=c_M2(2)



%4th order
% Vmax = 13.7; %[V]
% Vmin = 9.5; %[V]
%
% I1=25; % SoC[%]
% I2=95; % SoC[%]
% % a_M2_4 = a_M2*0.7;
% % b_M2_4 = b_M2*1.115;
% a_M2_4 = a_M2*0.8;
% b_M2_4 = b_M2*1.08;

Vmax = 13.75; %[V]
Vmin = 9.5; %[V]

I1=25; % SoC[%]
I2=95; % SoC[%]
% a_M2_4 = a_M2*0.7;
% b_M2_4 = b_M2*1.115;
a_M2_4 = a_M2*0.8;
b_M2_4 = b_M2*1.08;
[a0_M2, a1_M2, a2_M2, a3_M2, a4_M2] = Order4_Characteristic(I1,I2, a_M2_4, b_M2_4, Vmax,Vmin)

%Data Visualization

TF_new_new_M2 = x_new_new_M2 > 0;
TF_new_M2 = x_new_M2 > 0;
TF_old_M2 = x_old_M2 > 0;


figure,
scatter(x_new_new_M2(TF_new_new_M2), y_new_new_M2(TF_new_new_M2), 'green','filled'), hold on, grid on
scatter(x_new_M2(TF_new_M2), y_new_M2(TF_new_M2),'blue','filled')
scatter(x_old_M2(TF_old_M2), y_old_M2(TF_old_M2),'red','filled'), xlim([0 100]), ylim([10 14])


x_plot=0:1:100;  %used before: not necessary
y_plot=polyval([a_M2,b_M2],x_plot);
```

```
y_plot4_new=polyval(coeff4_new_new_M2,x_plot);
% y_plot4=polyval([a4_M2,a3_M2, a2_M2, a1_M2, a0_M2],x_plot);
y_plot4_medium=polyval(coeff4_new_M2,x_plot);

y_plot4_old=polyval(coeff4_old_M2,x_plot);
% plot(x_plot,y_plot,'black','LineWidth',1,'LineStyle','--'),hold on
plot(x_plot,y_plot4_new,'green','LineWidth',1,'LineStyle','-.'),hold on
plot(x_plot,y_plot4_medium,'blue','LineWidth',1,'LineStyle','-.'),hold on
plot(x_plot,y_plot4_old,'red','LineWidth',1,'LineStyle','-.'),hold on




%Tuning
a_M2_t=c_M2(1)-c_M2(1)*0.02
b_M2_t=c_M2(2)+0.5

y_plot_t=polyval([a_M2_t,b_M2_t],x_plot);
plot(x_plot,y_plot_t,'black','LineWidth',2),hold on


legend('new','medium','old','Original Characteristic','Tuned Characteristic','location', 'northwest'), title('Vocv(SoC) M2')
```

A.1.2 Static non-linear model

```

clc, clear all, close all
run('DATA_M2_Bianconiglio');

%Slow Dynamic Threshold: delete the first 100 seconds
din_thresh  = 500; %[s]

Ts = 0.01;

NEW
c1 = 0;
c2 = 0;


for i = 1:Numero_Esperimenti_M2_new_new
```

```
    isout = 0;
    if Sezioni_Esperimenti_M2_new_new(i) ~= -1 %esclusione esperimenti sbagliati
        for k = 1:Sezioni_Esperimenti_M2_new_new(i)
                data=['SoCSafeMon-FIAMM-L150P-New+_0' num2str(k) '_' num2str(i) '.mat'];
                load(data)
                SoC = fillmissing(SoC,'previous');  %create the full SoC vector

                %Outlier Correction
                if isout == 0 & k == 1
                    Chg_err = 13.5-Vbat(1);
                    if Chg_err > 0
                        Vbat = Vbat + Chg_err;
                        isout = 1;

                    end
                elseif isout == 1
                    Vbat = Vbat + Chg_err;
                end

                %DoD
                if k == 1

                    DoD0 = 0;
                    DoD = DoD0 + 100 * cumsum(Iload*Ts) / Cn;
                    DoD01 = DoD(end);

                else

                    DoD = DoD01 + 100* cumsum(Iload*Ts) / Cn;
                    DoD01 = DoD(end);

                end



                disp(['M2 newnew ' 'Section:' num2str(k) ' Rep: ' num2str(i)])



                %Simple Model R Estimation
% %                    [R_new_new_M2(k, i), SoCi_R_new_new_M2(k, i), LC_R_new_new_M2(k, i)] = R_Estimate_LS_SoC_Tuning(
```

```
    Vbat, Iload, SoC, Time, coeff4_b4_new_M2 , din_thresh);

                    [R_new_new_M2(1:2,k, i), Rs_new_new_M2(k, i)
                        , SoCi_R_new_new_M2(1:2,k, i),
                        DoDi_R_new_new_M2(1:2,k, i),
                        LC_R_new_new_M2(1:2,k, i)] =
                        R_Est_NL_SoC_Tuning1_V2(Vbat, Iload, SoC,
                         DoD, Time, coeff4_b4_new_M2, din_thresh,
                         c1 ,c2);

            end
        end
end
```

MEDIUM

```
c1 = 0.07; %original
c2 = 0.06;
%% c1 = 0.025; % Mg2
%% c2 = 0.045;
%% c1 = 0.025; % Mg3
%% c2 = 0.165;
%% c1 = 0.005; % Mg4
%% c2 = 0.3;
%
%% c1 = 0;
%% c2 = 0;

% c1=0.3;
% c2=1;


for i = 1:Numero_Esperimenti_M2_new
    isout = 0;
    if Sezioni_Esperimenti_M2_new(i) ~= -1 %esclusione
        esperimenti sbagliati
        for k = 1:Sezioni_Esperimenti_M2_new(i)
                data=['SoCMon-FIAMM-L150P-New_0' num2str(k)
                    '_' num2str(i) '.mat'];
                load(data)

                %Outlier Correction
                if isout == 0 & k == 1
                    Chg_err = 13.1-Vbat(1);
```

```
                    if Chg_err > 0
                        Vbat = Vbat + Chg_err;
                        isout = 1;
                    end
                elseif isout == 1
                    Vbat = Vbat + Chg_err;
                end


                % SoC recasting: SoC(0) computed with the Ctrue/Cn ratio but still defined wrt Cn (speed according to Cn)


                if k == 1

%                   DVorig = 0.4;
%                   [SoC_err] = SoC_recasting_4(Vbat, Iload, SoC, Time, coeff4_M2, din_thresh,DVorig )
%
                    SoC_truei = Cn_true_new(i)/Cn*100;
                    SoC_err = 100 - SoC_truei;
                    if SoC_err > 0
                        SoC_err;
                        SoC = SoC - SoC_err;
                    else
                        SoC_err = 0;
                    end

                    DoD0 = 0;
                    DoD = DoD0 + 100 * cumsum(Iload*Ts) / Cn;
                    DoD01 = DoD(end);

                else
                    SoC = SoC - SoC_err;

                    DoD = DoD01 + 100 * cumsum(Iload*Ts) / Cn;
                    DoD01 = DoD(end);

                end

                disp(['M2 new ' 'Section:' num2str(k) ' Rep: ' num2str(i)])

                %Simple Model R Estimation
```

```
    %                    [R_new_new_M2(k, i), SoCi_R_new_new_M2(k, i), LC_R_new_new_M2(k, i)] = R_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_med_M2 , din_thresh);


                    [R_new_M2(1:2,k, i), Rs_new_M2(k, i), SoCi_R_new_M2(1:2,k, i), DoDi_R_new_M2(1:2,k, i),  LC_R_new_M2(1:2,k, i)] = R_Est_NL_SoC_Tuning1_V2(Vbat, Iload, SoC, DoD, Time, coeff4_b4_med_M2, din_thresh, c1 ,c2);

        end
    end
end


OLD
c1 = 0.022;
c2 = 0.004;

% c1 = 0.0025; %Mg2
% c2 = 0.0003;

% c1 = 0.04; % Mg3
% c2 = 0.045;

% c1 = 0.04; % Mg4
% c2 = 0.045;

% c1 = 0;
% c2 = 0;


for i = 1:Numero_Esperimenti_M2_old
    if Sezioni_Esperimenti_M2_old(i) ~= -1 %esclusione esperimenti sbagliati
        for k = 1:Sezioni_Esperimenti_M2_old(i)
                    data=['SoCMon-FIAMM-L150P-Used_0' num2str(k) '_' num2str(i) '.mat'];
                    load(data)

                    % SoC recasting: SoC(0) computed with the Ctrue/Cn ratio but still defined wrt Cn (speed according to Cn)
                    if k == 1
                        k
```

```
                    i
%                     DVorig = 0.4;
%                     [SoC_err] = SoC_recasting_4(Vbat, Iload, SoC, Time, coeff4_M2, din_thresh,DVorig )
%
                    SoC_truei = Cn_true_old(i)/Cn*100;
                    SoC_err = 100 - SoC_truei;
                    if SoC_err > 0
                        SoC_err;
                        SoC = SoC - SoC_err;
                    else
                        SoC_err = 0;
                    end

                    DoD0 = 0;
                    DoD = DoD0 + 100 * cumsum(Iload*Ts) / Cn;
                    DoD01 = DoD(end);

                else
                    SoC = SoC - SoC_err;

                    DoD = DoD01 + 100 * cumsum(Iload*Ts) / Cn;
                    DoD01 = DoD(end);

                end

                disp(['M2 old ' 'Section:' num2str(k) ' Rep:' num2str(i)])


                %Simple Model R Estimation
%                     [R_new_new_M2(k, i), SoCi_R_new_new_M2(k, i), LC_R_new_new_M2(k, i)] = R_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_old_M2 , din_thresh);

                [R_old_M2(1:2,k, i), Rs_old_M2(k, i), SoCi_R_old_M2(1:2,k, i), DoDi_R_old_M2(1:2,k, i), LC_R_old_M2(1:2,k, i)] = R_Est_NL_SoC_Tuning1_V2(Vbat, Iload, SoC, DoD, Time, coeff4_b4_old_M2, din_thresh, c1 ,c2);

            end
        end
end
```

```
%% result display


% R_new_new_M2
%
% R1_new_new_M2
% R2_new_new_M2
% C2_new_new_M2
%
% SoCi_R_new_new_M2
% SoCi_RRC_new_new_M2
%
%
% R_new_M2
% R1_new_M2
% R2_new_M2
% C2_new_M2
%
% SoCi_R_new_M2
% SoCi_RRC_new_M2
%
%
% R_old_M2
% R1_old_M2
% R2_old_M2
% C2_old_M2
%
% SoCi_R_old_M2
% SoCi_RRC_old_M2


% R plot
TF_new_new = SoCi_R_new_new_M2 > 0;
TF_new = SoCi_R_new_M2 > 0;
TF_old = SoCi_R_old_M2 > 0;

figure,
subplot(1,2,1)
scatter(SoCi_R_new_new_M2(TF_new_new), R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_R_new_M2(TF_new), R_new_M2(TF_new),'blue','filled'), hold on
scatter(SoCi_R_old_M2(TF_old), R_old_M2(TF_old),'red','filled'), xlabel('SoCi_i_d [%]'), ylabel('R_h [Ohm]')
legend('new','medium','old'), title('R-Estimate - Bianconiglio 4 - M2')
```

```
subplot(1,2,2)
scatter(SoCi_R_new_new_M2(TF_new_new), LC_R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_R_new_M2(TF_new), LC_R_new_M2(TF_new),'blue','filled'),
scatter(SoCi_R_old_M2(TF_old), LC_R_old_M2(TF_old),'red','filled'), xlabel('SoCi_i_d [%]'), ylabel('RMSe')
legend('new','medium','old'), title('Linearity Coefficient - Bianconiglio 4 - M2')

figure,
% subplot(1,2,1)
scatter(DoDi_R_new_new_M2(TF_new_new), R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(DoDi_R_new_M2(TF_new), R_new_M2(TF_new),'blue','filled'), hold on
scatter(DoDi_R_old_M2(TF_old), R_old_M2(TF_old),'red','filled'), xlabel('SoCi_i_d [%]'), ylabel('R_h [Ohm]')
legend('new','medium','old'), title('R-Estimate - Bianconiglio 4 - M2')


figure,
% subplot(1,2,1)
scatter3(SoCi_R_new_new_M2(TF_new_new), 100-DoDi_R_new_new_M2(TF_new_new), R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter3(SoCi_R_new_M2(TF_new), 100-DoDi_R_new_M2(TF_new), R_new_M2(TF_new),'blue','filled'), hold on
scatter3(SoCi_R_old_M2(TF_old), 100-DoDi_R_old_M2(TF_old), R_old_M2(TF_old),'red','filled'), xlabel('SoC [%]'), ylabel('SoC_R_e_l [%]'), zlabel('R_h [Ohm]')
legend('new','medium','old'), title('R-Estimate - Bianconiglio 4 - M2')



Curva 2d - set di parabole
figure,
scatter(SoCi_R_new_new_M2(TF_new_new), R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_R_new_M2(TF_new), R_new_M2(TF_new),'blue','filled'), hold on
scatter(SoCi_R_old_M2(TF_old), R_old_M2(TF_old),'red','filled'), xlabel('SoCi_i_d [%]'), ylabel('R_h [Ohm]')
legend('new','medium','old'), title('R-Estimate - Bianconiglio 4 - M2')
```

```
ax_par_new_new = 50;
ax_par_old = 6;
%
% a_par_i =100000;
% b_par_i = -(2*a_par_i*ax_par_old)
% c_par_i = 50;
%
% x_plot_par = 0:100;
% y_plot_par = polyval([a_par_i b_par_i c_par_i], x_plot_par
    )
%
% plot(x_plot_par,y_plot_par)

%
R_old_t= [ R_old_M2(TF_old)+0.05; fliplr(R_old_M2(TF_old)
    +0.05)];
SoC_old_t = [SoCi_R_old_M2(TF_old)+1; ax_par_old+(ax_par_old
    -SoCi_R_old_M2(TF_old))-1 ] ;

scatter(SoC_old_t, R_old_t)

R_new_new_t= [ R_new_new_M2(TF_new_new); fliplr(R_new_new_M2
    (TF_new_new))];
SoC_new_new_t = [SoCi_R_new_new_M2(TF_new_new);
    ax_par_new_new - (-ax_par_new_new+SoCi_R_new_new_M2(
    TF_new_new)) ] ;

scatter(SoC_new_new_t, R_new_new_t)


% p1_old =  0.003264  ;
% p2_old =    -0.1045   ;
% p3_old =      0.9629 ;

% p1_old = 0.005  ;
% p2_old =   -0.08001  ;
% p3_old =     0.5273;

p1_old = 0.006804  ;
p2_old =    -0.08165  ;
p3_old =    0.4433;




Coeff_par_f = [p1_old p2_old p3_old];
```

```
p1_new_new =  2.159e-05 ;
p2_new_new =  -0.002159 ;
p3_new_new = 0.145   ;

Coeff_par_i = [p1_new_new p2_new_new p3_new_new];



x_plot_par = 0:100;

for SoH = 0:0.05:1

    s = Coeff_par_i + (1-SoH)^8 * [Coeff_par_f-Coeff_par_i];
%     s = Coeff_par_f + (1-SoH)^2 * [Coeff_par_i-Coeff_par_f];
    y_plot_par = polyval(s,x_plot_par);


    if SoH == 0
        plot(x_plot_par,y_plot_par,'red'), hold on

    end
    if SoH == 1
        plot(x_plot_par,y_plot_par,'green'), hold on

    end
    if SoH < 1 && SoH > 0
        plot(x_plot_par,y_plot_par,'blue'), hold on

    end



end

ylim([0.05, 0.4])









% R_old_orig = [R_new_new_M2(TF_new_new); R_new_M2(TF_new); R_old_M2(TF_old)];
% SoC_tot = [SoCi_R_new_new_M2(TF_new_new) ; SoCi_R_new_M2(TF_new) ; SoCi_R_old_M2(TF_old) ] ;
```

```
%
%% Tuning
% R_tot = R_tot_orig;
% R_tot(R_tot>0.2) = R_tot(R_tot>0.2) + 0.3;
%
%
%% coeff_SoC_R = [deg6_fit_R_SoC.p1, deg6_fit_R_SoC.p2, deg6_fit_R_SoC.p3, deg6_fit_R_SoC.p4, deg6_fit_R_SoC.p5, deg6_fit_R_SoC.p6, deg6_fit_R_SoC.p7];
%% figure
%% plot([0:100],polyval(coeff_SoC_R , [0:100]))
%% save('coeff_SoC_R','coeff_SoC_R')
%
% figure
%% coeff_spline=Spline_fit_SoC_R.p
%
% plot([0:100], fnval(coeff_spline, [0:100]))
%
% save('spline_Fit_SoC_R', 'coeff_spline')


 Superfice 3d
%
% [xq,yq] = meshgrid(0:1:100, 0:1:100);
% x = [SoCi_R_new_new_M2(TF_new_new); SoCi_R_new_M2(TF_new); SoCi_R_old_M2(TF_old)];
% y = [100-DoDi_R_new_new_M2(TF_new_new); 100-DoDi_R_new_M2(TF_new); 100-DoDi_R_old_M2(TF_old)];
% v = [R_new_new_M2(TF_new_new); R_new_M2(TF_new); R_old_M2(TF_old) ];
%%  'nearest', 'linear', 'natural', and 'cubic'
% vq = griddata(x,y,v,xq,yq,'natural')
%% vq = fillmissing(vq,'linear',1)
%% vq = fillmissing(vq,'linear',2)
% mesh(xq,yq,vq)
%
% save('Lookup_Table','vq')
%




%% RRC plot
%
% TF_new_new = SoCi_RRC_new_new_M2 > 0;
% TF_new = SoCi_RRC_new_M2 > 0;
% TF_old = SoCi_RRC_old_M2 > 0;
%
% figure,
```

```
% subplot(1,3,1)
% scatter(SoCi_RRC_new_new_M2(TF_new_new), R1_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_M2(TF_new), R1_new_M2(TF_new),'blue','filled'),
% scatter(SoCi_RRC_old_M2(TF_old), R1_old_M2(TF_old),'red','filled'), title('R1-Estimate - Bianconiglio 3 Bis'), xlabel('SoC [%]'), ylabel('R1_h [Ohm]')
% legend('new','medium','old')
%
% subplot(1,3,2)
% scatter(SoCi_RRC_new_new_M2(TF_new_new), R2_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_M2(TF_new), R2_new_M2(TF_new),'blue','filled'),
% scatter(SoCi_RRC_old_M2(TF_old), R2_old_M2(TF_old),'red','filled'), title('R2-Estimate - Bianconiglio 3 Bis'), xlabel('SoC [%]'), ylabel('R2_h [Ohm]')
% legend('new','medium','old')
%
% subplot(1,3,3)
% scatter(SoCi_RRC_new_new_M2(TF_new_new), C2_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_M2(TF_new), C2_new_M2(TF_new),'blue','filled'),
% scatter(SoCi_RRC_old_M2(TF_old), C2_old_M2(TF_old),'red','filled'), title('C2-Estimate - Bianconiglio 3 Bis'), xlabel('SoC [%]'), ylabel('C2_h [F]')
% legend('new','medium','old')

%% RRC SLOW plot
%
% TF_new_new_slow = SoCi_RRC_new_new_slow_M2 > 0;
% TF_new_slow = SoCi_RRC_new_slow_M2 > 0;
% TF_old_slow = SoCi_RRC_old_slow_M2 > 0;
%
% figure,
% subplot(1,3,1)
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), R1_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), R1_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), R1_old_slow_M2(TF_old_slow),'red','filled'), title('R1 slow-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('R1_h [Ohm]')
%
% subplot(1,3,2)
```

```
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), R2_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), R2_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), R2_old_slow_M2(TF_old_slow),'red','filled'), title('R2 slow-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('R2_h [Ohm]')
%
%
% subplot(1,3,3)
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), C2_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), C2_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), C2_old_slow_M2(TF_old_slow),'red','filled'), title('C2 slow-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('C2_h [F]')
% legend('new','medium','old')
%
%
% % tau
% figure,
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), R2_new_new_slow_M2(TF_new_new_slow).*C2_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), R2_new_slow_M2(TF_new_slow).*C2_new_slow_M2(TF_new_slow),'blue','filled'),
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), R2_old_slow_M2(TF_old_slow).*C2_old_slow_M2(TF_old_slow),'red','filled'), title('Tau - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('Tau [1/s]')
% legend('new','medium','old')
%
% % Model Error
% figure
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), LC_RRC_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), LC_RRC_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), LC_RRC_old_slow_M2(TF_old_slow),'red','filled')
%
```

```
% legend('new','medium','old'), title('Linearity Coefficient - Bianconiglio 4 - M2'), xlabel('SoCi_i_d [%]'), ylabel('RMSe')
```

A.1.3 Real capacity

```

V_threshold_min = 11.2 V


clc, clear all, close all
run('DATA_M2_Bianconiglio.m')

format long


Ts = 0.01;
V_th_min = 11.2; % [V]
realC_correction = 1+(V_th_min-10.5)/(13.6-10.5) %to be discussed

%medium battery

Cn_true = zeros(Numero_Esperimenti_M2_new,1);
Cn_extract = zeros(Numero_Esperimenti_M2_new,1);
for i = 1:Numero_Esperimenti_M2_new
    if Sezioni_Esperimenti_M2_new(i) ~= -1 %esclusione esperimenti sbagliati
        for k = 1:Sezioni_Esperimenti_M2_new(i)
            data = ['SoCMon-FIAMM-L150P-New_0' num2str(k) '_' num2str(i) '.mat'];
            load(data)

            Cn_true(i) = Cn_true(i) + sum(Iload*Ts);

            Vbat_ = smoothdata(Vbat,'movmedian',1e3);
            TF = Vbat_ > V_th_min & Iload > 1;
            Cn_extract(i) = Cn_extract(i) + sum(Iload(TF)*Ts);

%               figure,
%               plot(Time,Vbat_), title(['section:' num2str(k) 'rep' num2str(i)])
            min_V(k,i) = min(Vbat_);

            if k == 1
                figure,
```

```
                    plot(Time,Vbat_), title(['section:' num2str(k) 'rep' num2str(i)])
                    Vt_i(i) = mean(Vbat(1:300));
                end
            end
        end
end

Cn_true
Cn_extract
min_V
Vt_i


figure
TF = Cn_true ~= 0;
scatter(Vt_i(TF),Cn_true(TF),'*'), xlabel('Vt(t=0) [V]'), ylabel('C extract to end experiment [A*s]')

figure
TF = Cn_extract ~= 0;
scatter(Vt_i(TF),Cn_extract(TF),'*'), xlabel('Vt(t=0) [V]'), ylabel('C extract @ threshold [A*s]')



clc, clear all, close all
run('DATA_M2_Bianconiglio.m')

format long


Ts = 0.01;
V_th_min = 11.2; % [V]
realC_correction = 1+(V_th_min-10.5)/(13.6-10.5)



%oldbattery

Cn_true_old = zeros(Numero_Esperimenti_M2_old,1);
Cn_extract_old = zeros(Numero_Esperimenti_M2_old,1);
for i = 1:Numero_Esperimenti_M2_old
    if Sezioni_Esperimenti_M2_old(i) ~= -1 %esclusione esperimenti sbagliati
        for k = 1:Sezioni_Esperimenti_M2_old(i)
            data=['SoCMon-FIAMM-L150P-Used_0' num2str(k) '_' num2str(i) '.mat'];
```

```
                load(data)

                Cn_true_old(i) = Cn_true_old(i) + sum(Iload*Ts);

                Vbat_ = smoothdata(Vbat,'movmedian',1e3);
                TF = Vbat_ > V_th_min & Iload > 1;
                Cn_extract_old(i) = Cn_extract_old(i) + sum(Iload(TF)*Ts);

                figure,
                plot(Time,Vbat_), title(['section:' num2str(k) 'rep' num2str(i)])
                min_V(k,i) = min(Vbat_);

                if k == 1
%                   figure,
%                   plot(Time,Vbat_), title(['section:' num2str(k) 'rep' num2str(i)])
                    Vt_i(i) = mean(Vbat(1:300));
                end
            end
        end
end

Cn_true_old
Cn_extract_old
min_V
Vt_i


figure
TF = Cn_true_old ~= 0;
scatter(Vt_i(TF),Cn_true_old(TF),'*'), xlabel('Vt(t=0) [V]'), ylabel('C extract to end experiment [A*s]')

figure
TF = Cn_extract_old ~= 0;
scatter(Vt_i(TF),Cn_extract_old(TF),'*'), xlabel('Vt(t=0) [V]'), ylabel('C extract @ threshold [A*s]')
```

A.1.4 Absolute Interpretation: R, RC, R_{nl}

```
clc, clear all, close all


run('DATA_M2_Bianconiglio');
```

```
%Slow Dynamic Threshold: delete the first 100 seconds
din_thresh  = 500; %[s]



for i = 2:Numero_Esperimenti_M2_new_new
    isout = 0;
    for k = 1:Sezioni_Esperimenti_M2_new_new(i)
            data=['SoCSafeMon-FIAMM-L150P-New+_0' num2str(k) '_' num2str(i) '.mat'];
            load(data)

            %Outlier Correction
            if isout == 0 & k == 1
                Chg_err = 13.5-Vbat(1);
                if Chg_err > 0
                    Vbat = Vbat + Chg_err;
                    isout = 1;

                end
            elseif isout == 1
                Vbat = Vbat + Chg_err;
            end




             disp(['M2 newnew ' 'Section:' num2str(k) ' Rep: ' num2str(i)])

%             %Simple Model R Estimation
% %             [R_new_new_M2(k, i-1), SoCi_R_new_new_M2(k, i-1), LC_R_new_new_M2(k, i-1)] = R_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_new_M2 , din_thresh);
%             n_split = 2;
%             pSoC=10; %slpit percentage
%             [R_new_new_M2(1:n_split, k, i-1), SoCi_R_new_new_M2(1:n_split,k, i-1), LC_R_new_new_M2(1:n_split,k, i-1)] = R_Estimate_LS_SoC_Tuning_split(Vbat, Iload, SoC, Time, coeff4_b4_new_M2, din_thresh, n_split, pSoC);

            %RRC Estimation
            Current_Filter = 0;   % 1 = ON   0 = OFF
%             [R1_new_new_M2(k, i-1), R2_new_new_M2(k, i-1), C2_new_new_M2(k, i-1), SoCi_RRC_new_new_M2(k, i-1)] =
```

```
RRC_Estimate_LS(Vbat, Iload, SoC, Time, a, b, Current_Filter, din_thresh);
            [R1_new_new_M2(k, i-1), R2_new_new_M2(k, i-1), C2_new_new_M2(k, i-1), SoCi_RRC_new_new_M2(k, i-1)] = RRC_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_new_M2, Current_Filter, din_thresh);

%               %RRC Slow Dynamics Estimation
%               [R1_new_new_slow_M2(k, i-1), R2_new_new_slow_M2(k, i-1),C2_new_new_slow_M2(k, i-1), SoCi_RRC_new_new_slow_M2(k, i-1), LC_RRC_new_new_slow_M2(k, i-1)] = RRC_Estimate_SLOW_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_new_M2);
%               [R1_rnew_new_slow_M2(k, i-1), R2_new_new_slow_M2(k, i-1),C2_new_new_slow_M2(k, i-1), SoCi_RRC_new_new_slow_M2(k, i-1), LC_RRC_new_new_slow_M2(k, i-1)] = RRC_Estimate_SLOW_LS_SoC_Tuning_Full(Vbat, Iload, SoC, Time, coeff4_b4_new_M2);

    end
end




for i = 2:Numero_Esperimenti_M2_new
    isout = 0;
    for k = 1:Sezioni_Esperimenti_M2_new(i)
            data=['SoCMon-FIAMM-L150P-New_0' num2str(k) '_' num2str(i) '.mat'];
            load(data)

            %Outlier Correction
            if isout == 0 & k == 1
                Chg_err = 13.1-Vbat(1);
                if Chg_err > 0
                    Vbat = Vbat + Chg_err;
                    isout = 1;
                end
            elseif isout == 1
                Vbat = Vbat + Chg_err;
            end


            % SoC recasting: SoC(0) computed with the Ctrue/Cn ratio but still defined wrt Cn (speed
```

```
                according to Cn)
            if k == 1
                k
                i
%                 DVorig = 0.4;
%                 [SoC_err] = SoC_recasting_4(Vbat, Iload, SoC, Time, coeff4_M2, din_thresh,DVorig )
%
                SoC_truei = Cn_true_new(i)/Cn*100;
                SoC_err = 100 - SoC_truei;
                if SoC_err > 0
                    SoC_err;
                    SoC = SoC - SoC_err;
                else
                    SoC_err = 0;
                end
            else
                SoC = SoC - SoC_err;
            end

            disp(['M2 new ' 'Section:' num2str(k) ' Rep: ' num2str(i)])

% %             %Simple Model R Estimation
% %             [R_new_M2(k, i-1), SoCi_R_new_M2(k, i-1), LC_R_new_M2(k, i-1)] = R_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_med_M2, din_thresh);
%             n_split = 2;
%             pSoC=10;  %slpit percentage
%             [R_new_M2(1:n_split,k, i-1), SoCi_R_new_M2(1:n_split, k, i-1),LC_R_new_M2(1:n_split, k, i-1)] = R_Estimate_LS_SoC_Tuning_split(Vbat, Iload, SoC, Time, coeff4_b4_med_M2, din_thresh, n_split, pSoC);

%             %RRC Estimation
            Current_Filter = 0;   % 1 = ON   0 = OFF
%             [R1_new_M2(k, i-1), R2_new_M2(k, i-1), C2_new_M2(k, i-1), SoCi_RRC_new_M2(k, i-1)] = RRC_Estimate_LS(Vbat, Iload, SoC, Time, a, b, Current_Filter, din_thresh);
            [R1_new_M2(k, i-1), R2_new_M2(k, i-1),C2_new_M2(k, i-1), SoCi_RRC_new_M2(k, i-1)] = RRC_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_med_M2, Current_Filter, din_thresh);


            %RRC Slow Dynamics Estimation
```

```
%                [R1_new_slow_M2(k, i-1), R2_new_slow_M2(k, i-1),C2_new_slow_M2(k, i-1), SoCi_RRC_new_slow_M2(k, i-1), LC_RRC_new_slow_M2(k, i-1)] = RRC_Estimate_SLOW_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_med_M2);
%                [R1_new_slow_M2(k, i-1), R2_new_slow_M2(k, i-1),C2_new_slow_M2(k, i-1), SoCi_RRC_new_slow_M2(k, i-1), LC_RRC_new_slow_M2(k, i-1)] = RRC_Estimate_SLOW_LS_SoC_Tuning_Full(Vbat, Iload, SoC, Time, coeff4_b4_med_M2);
    end
end

for i = 2:Numero_Esperimenti_M2_old
    for k = 1:Sezioni_Esperimenti_M2_old(i)
            data=['SoCMon-FIAMM-L150P-Used_0' num2str(k) '_' num2str(i) '.mat'];
            load(data)

            % SoC recasting: SoC(0) computed with the Ctrue/Cn ratio but still defined wrt Cn (speed according to Cn)
            if k == 1
                k
                i
%                 DVorig = 0.4;
%                 [SoC_err] = SoC_recasting_4(Vbat, Iload, SoC, Time, coeff4_M2, din_thresh,DVorig )
%
                SoC_truei = Cn_true_old(i)/Cn*100;
                SoC_err = 100 - SoC_truei;
                if SoC_err > 0
                    SoC_err;
                    SoC = SoC - SoC_err;
                else
                    SoC_err = 0;
                end
            else
                SoC = SoC - SoC_err;
            end

            disp(['M2 old ' 'Section:' num2str(k) ' Rep: ' num2str(i)])

%               %Simple Model R Estimation
% %               [R_old_M2(k, i-1), SoCi_R_old_M2(k, i-1), LC_R_old_M2(k, i-1)] = R_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_old_M2, din_thresh);
```

```
%                n_split = 2;
%                pSoC=10;  %slpit percentage
%                [R_old_M2(1:n_split,k, i-1), SoCi_R_old_M2(1:n_split,k, i-1),LC_R_old_M2(1:n_split,k, i-1)]    = R_Estimate_LS_SoC_Tuning_split(Vbat, Iload, SoC, Time, coeff4_b4_old_M2, din_thresh,n_split, pSoC);

               %RRC Estimation
              Current_Filter = 0;    % 1 = ON    0 = OFF
%               [R1_old_M2(k, i-1), R2_old_M2(k, i-1), C2_old_M2(k, i-1), SoCi_RRC_old_M2(k, i-1)] = RRC_Estimate_LS(Vbat, Iload, SoC, Time, a, b, Current_Filter, din_thresh);
              [R1_old_M2(k, i-1), R2_old_M2(k, i-1),C2_old_M2(k, i-1), SoCi_RRC_old_M2(k, i-1)] = RRC_Estimate_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_old_M2, Current_Filter, din_thresh);

               %RRC Slow Dynamics Estimation
%               [R1_old_slow_M2(k, i-1), R2_old_slow_M2(k, i-1),C2_old_slow_M2(k, i-1), SoCi_RRC_old_slow_M2(k, i-1), LC_RRC_old_slow_M2(k, i-1)] = RRC_Estimate_SLOW_LS_SoC_Tuning(Vbat, Iload, SoC, Time, coeff4_b4_old_M2);
%               [R1_old_slow_M2(k, i-1), R2_old_slow_M2(k, i-1),C2_old_slow_M2(k, i-1), SoCi_RRC_old_slow_M2(k, i-1), LC_RRC_old_slow_M2(k, i-1)] = RRC_Estimate_SLOW_LS_SoC_Tuning_Full(Vbat, Iload, SoC, Time, coeff4_b4_old_M2);

    end
end


%% result display


% R_new_new_M2
%
% R1_new_new_M2
% R2_new_new_M2
% C2_new_new_M2
%
% SoCi_R_new_new_M2
% SoCi_RRC_new_new_M2
%
%
```

```
% R_new_M2
% R1_new_M2
% R2_new_M2
% C2_new_M2
%
% SoCi_R_new_M2
% SoCi_RRC_new_M2
%
%
% R_old_M2
% R1_old_M2
% R2_old_M2
% C2_old_M2
%
% SoCi_R_old_M2
% SoCi_RRC_old_M2


% % R plot
% TF_new_new = SoCi_R_new_new_M2 > 0;
% TF_new = SoCi_R_new_M2 > 0;
% TF_old = SoCi_R_old_M2 > 0;
%
% figure,
% subplot(1,2,1)
% scatter(SoCi_R_new_new_M2(TF_new_new), R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
% scatter(SoCi_R_new_M2(TF_new), R_new_M2(TF_new),'blue','filled'),
% scatter(SoCi_R_old_M2(TF_old), R_old_M2(TF_old),'red','filled'), xlabel('SoCi_i_d [%]'), ylabel('R_h [Ohm]')
% legend('new','medium','old'), title('R-Estimate - Bianconiglio 4 - M2')
%
% subplot(1,2,2)
% scatter(SoCi_R_new_new_M2(TF_new_new), LC_R_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
% scatter(SoCi_R_new_M2(TF_new), LC_R_new_M2(TF_new),'blue','filled'),
% scatter(SoCi_R_old_M2(TF_old), LC_R_old_M2(TF_old),'red','filled'), xlabel('SoCi_i_d [%]'), ylabel('RMSe')
% legend('new','medium','old'), title('Linearity Coefficient - Bianconiglio 4 - M2')


% RRC plot

TF_new_new = SoCi_RRC_new_new_M2 > 0;
```

```
TF_new = SoCi_RRC_new_M2 > 0;
TF_old = SoCi_RRC_old_M2 > 0;

figure,
subplot(1,3,1)
scatter(SoCi_RRC_new_new_M2(TF_new_new), R1_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_RRC_new_M2(TF_new), R1_new_M2(TF_new),'blue','filled'),
scatter(SoCi_RRC_old_M2(TF_old), R1_old_M2(TF_old),'red','filled'), title('R1-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('R1_h [Ohm]')
legend('new','medium','old')

subplot(1,3,2)
scatter(SoCi_RRC_new_new_M2(TF_new_new), R2_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_RRC_new_M2(TF_new), R2_new_M2(TF_new),'blue','filled'),
scatter(SoCi_RRC_old_M2(TF_old), R2_old_M2(TF_old),'red','filled'), title('R2-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('R2_h [Ohm]')
legend('new','medium','old')

subplot(1,3,3)
scatter(SoCi_RRC_new_new_M2(TF_new_new), C2_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_RRC_new_M2(TF_new), C2_new_M2(TF_new),'blue','filled'),
scatter(SoCi_RRC_old_M2(TF_old), C2_old_M2(TF_old),'red','filled'), title('C2-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('C2_h [F]')
legend('new','medium','old')

figure,
scatter(SoCi_RRC_new_new_M2(TF_new_new),  R2_new_new_M2(TF_new_new).*C2_new_new_M2(TF_new_new),'green','filled'), hold on, grid on
scatter(SoCi_RRC_new_M2(TF_new), R2_new_M2(TF_new).*C2_new_M2(TF_new),'blue','filled'),
scatter(SoCi_RRC_old_M2(TF_old), R2_old_M2(TF_old).*C2_old_M2(TF_old),'red','filled'), title('Tau - Bianconiglio 4 '), xlabel('SoC [%]'), ylabel('Tau')
legend('new','medium','old')

%% RRC SLOW plot
%
% TF_new_new_slow = SoCi_RRC_new_new_slow_M2 > 0;
```

```
% TF_new_slow = SoCi_RRC_new_slow_M2 > 0;
% TF_old_slow = SoCi_RRC_old_slow_M2 > 0;
%
% figure,
% subplot(1,3,1)
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), R1_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), R1_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), R1_old_slow_M2(TF_old_slow),'red','filled'), title('R1 slow-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('R1_h [Ohm]')
%
% subplot(1,3,2)
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), R2_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), R2_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), R2_old_slow_M2(TF_old_slow),'red','filled'), title('R2 slow-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('R2_h [Ohm]')
%
%
% subplot(1,3,3)
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), C2_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), C2_new_slow_M2(TF_new_slow),'blue','filled'),hold on, grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), C2_old_slow_M2(TF_old_slow),'red','filled'), title('C2 slow-Estimate - Bianconiglio 4'), xlabel('SoC [%]'), ylabel('C2_h [F]')
% legend('new','medium','old')
%
%
%% tau
% figure,
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow), R2_new_new_slow_M2(TF_new_new_slow).*C2_new_new_slow_M2(TF_new_new_slow),'green','filled'), hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow), R2_new_slow_M2(TF_new_slow).*C2_new_slow_M2(TF_new_slow),'blue','filled'),
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow), R2_old_slow_M2(TF_old_slow).*C2_old_slow_M2(TF_old_slow),'red','filled'), title('Tau - Bianconiglio 4'), xlabel('SoC [%]'),
```

```
    ylabel('Tau [1/s]')
% legend('new','medium','old')
%
%% Model Error
% figure
% scatter(SoCi_RRC_new_new_slow_M2(TF_new_new_slow),
    LC_RRC_new_new_slow_M2(TF_new_new_slow),'green','filled')
    , hold on, grid on
% scatter(SoCi_RRC_new_slow_M2(TF_new_slow),
    LC_RRC_new_slow_M2(TF_new_slow),'blue','filled'),hold on,
     grid on
% scatter(SoCi_RRC_old_slow_M2(TF_old_slow),
    LC_RRC_old_slow_M2(TF_old_slow),'red','filled')
%
% legend('new','medium','old'), title('Linearity Coefficient
     - Bianconiglio 4 - M2'), xlabel('SoCi_i_d [%]'), ylabel
    ('RMSe')
```

A.2 Simulink Code

A.2.1 Model

Figure A.1

A.2.2 EKF

Figure A.2

A.2.3 ANSE

Figure A.3

A.2.4 Final Solution

Figure A.4

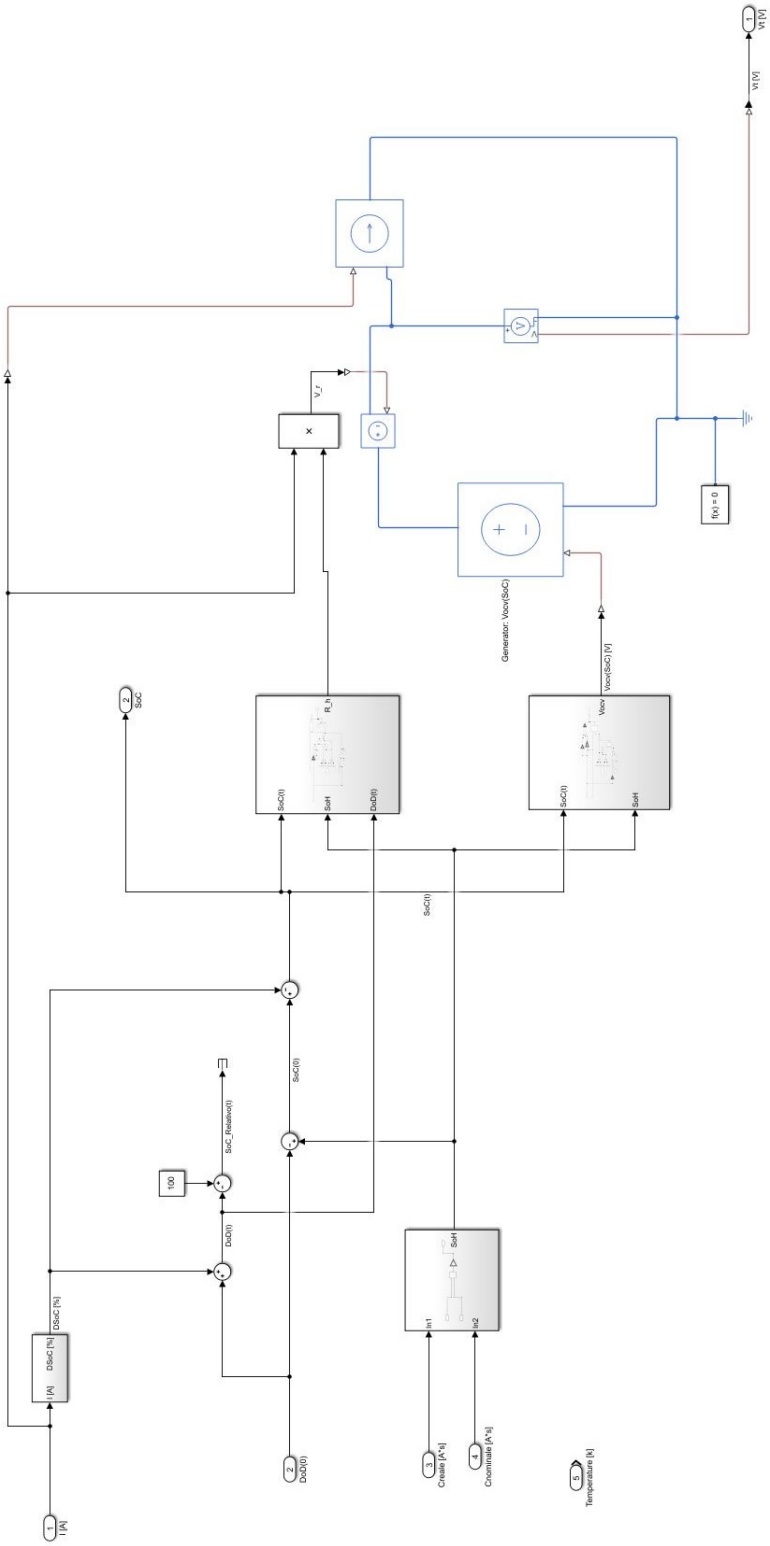


Figure A.1: Simulink Implementation: battery model

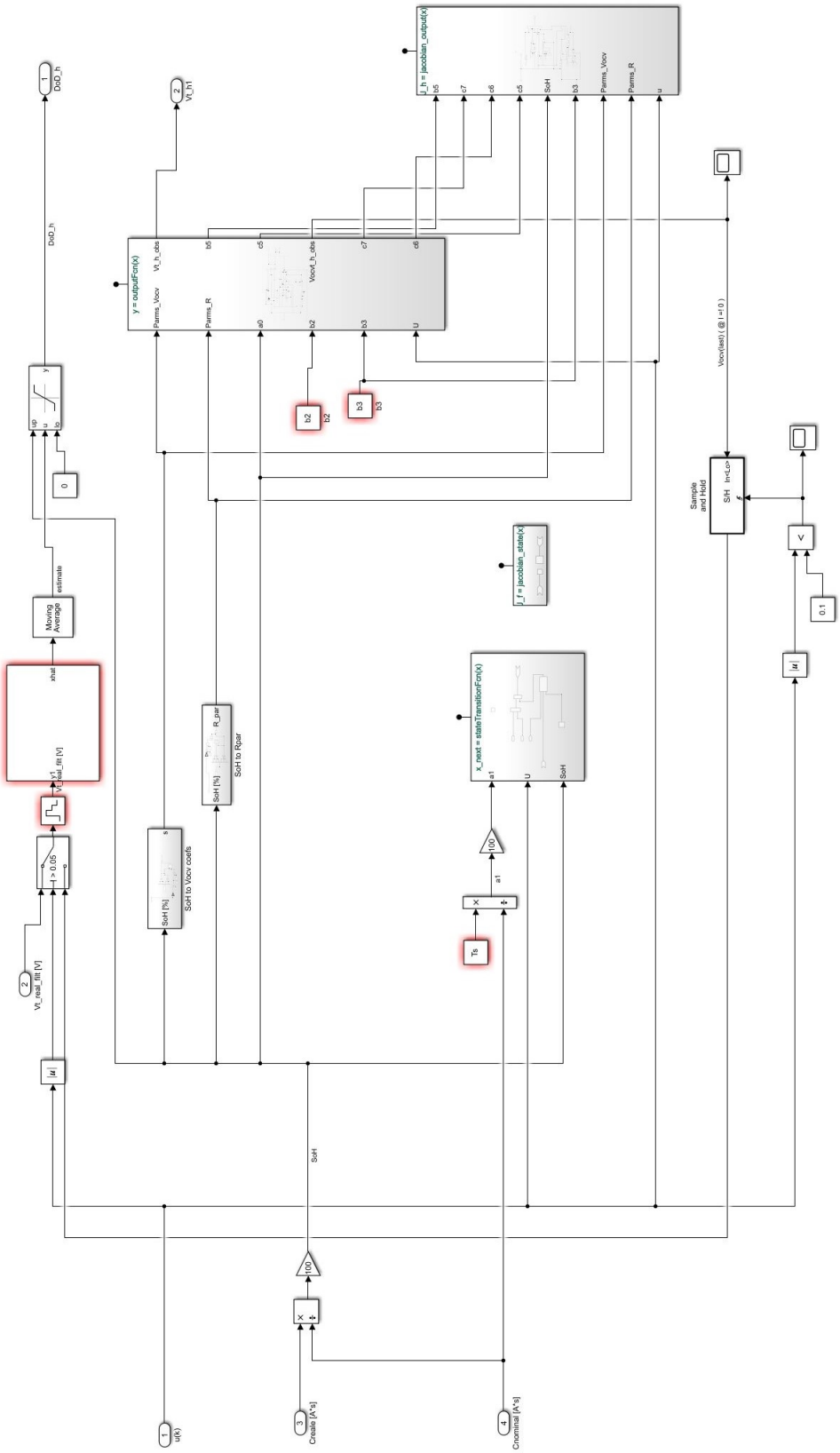


Figure A.2: Simulink Implementation: Extended Kalman Filter

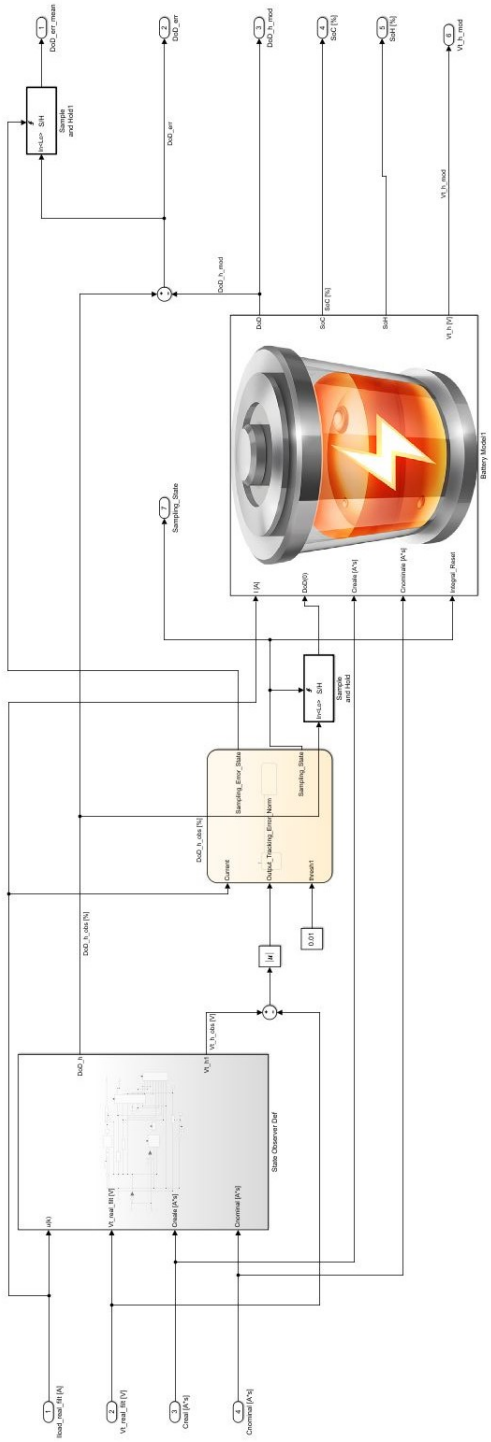


Figure A.3: Simulink Implementation: ANSE

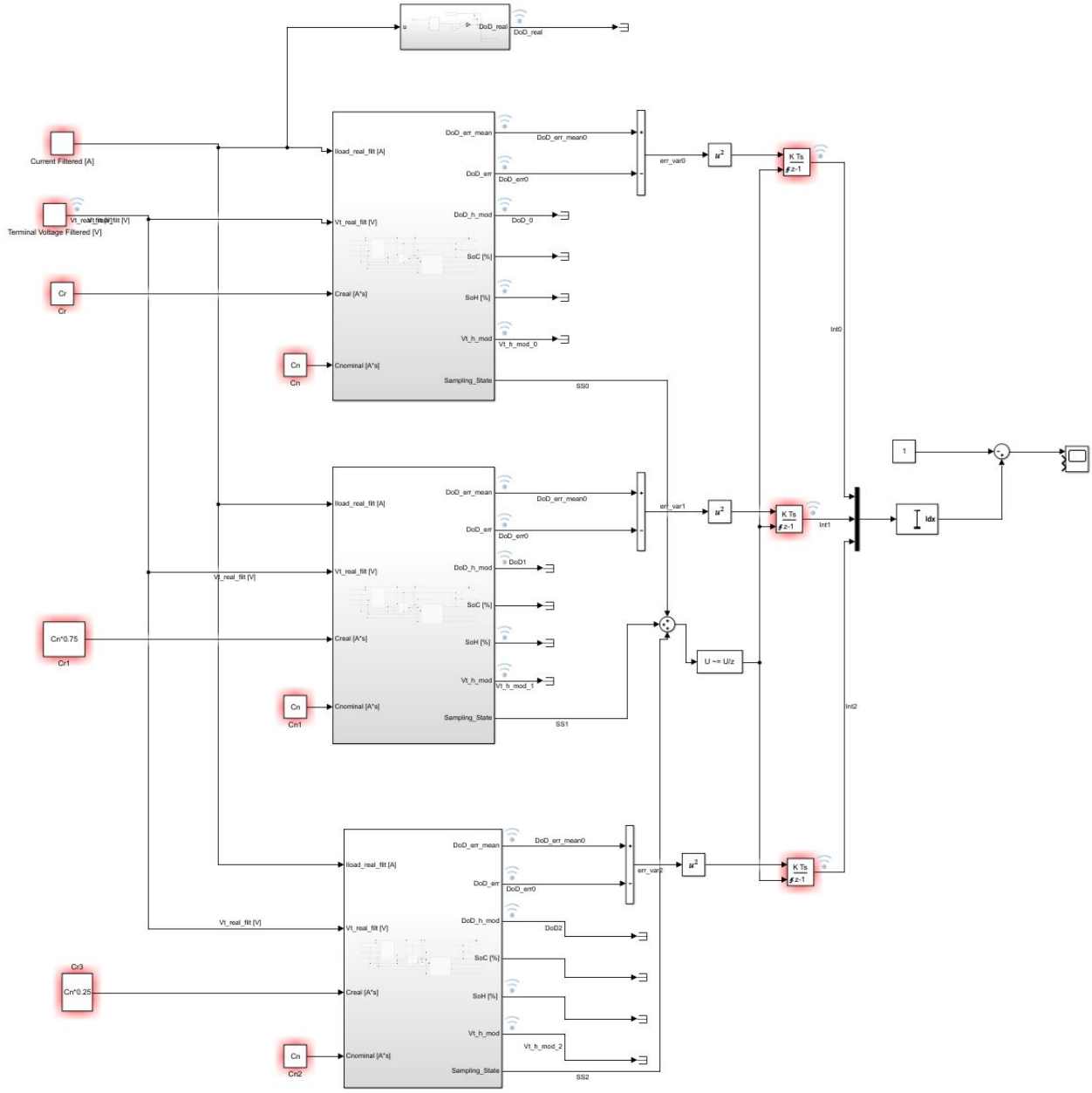


Figure A.4: Simulink Implementation: Final solution

Bibliography

[1] Battery University. BU-201: How does the Lead Acid Battery Work? 2019. URL: https://batteryuniversity.com/learn/article/lead_based_batteries.

[2] Battery University. BU-204: How do Lithium Batteries Work? 2018. URL: https://batteryuniversity.com/learn/article/lithium_based_batteries.

[3] Battery University. BU-903: How to Measure State-of-charge. 2019. URL: https://batteryuniversity.com/learn/article/how_to_measure_state_of_charge.

[4] David Frankel, Amy Wagner. Battery storage: The next disruptive technology in the power sec 2017. URL: https://www.mckinsey.com/business-functions/sustainability/our-insights/battery-storage-the-next-disruptive-technology-in-the-power-sector.

[5] David R. Baker, Bloomberg.com. Battery Reality: There's Nothing Better Than Lithium-Ion C 2019. URL: https://www.bloomberg.com/news/articles/2019-04-03/battery-reality-there-s-nothing-better-than-lithium-ion-coming-soon.

[6] Energy.gov. How Does a Lithium-ion Battery Work? 2017. URL: https://www.energy.gov/eere/articles/how-does-lithium-ion-battery-work.

[7] Gregory L. Plett. "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs Part 2. Modeling and identification". In: Journal of Power Sources 134 (2004) 262–276 (2004).

[8] Huang Kai, Guo Yong-Fang, Li Zhi-Gang, Lin Hsiung-Cheng and Li Ling-Ling. "Development of Accurate Lithium-Ion Battery Model Based on Adaptive Random Disturbance PSO Algorithm". In: Hindawi, Mathematical Problems in Engineering (2018).

[9] IEA, Paris. Global EV Outlook 2019. 2019. URL: www.iea.org/publications/reports/globalevoutlook2019/.

[10] Ingemar Kaj, Victorien Konanéb. "Modeling battery cells under discharge using kinetic and stochastic battery models". In: Elsevier, Applied Mathematical Modelling (2016).

[11] J.F. Manwell, J.G. McGowan. "Lead Acid Battery Storage Model for Hybrid Energy Systems". In: Sol. Energy 1993, 50, 399–405. (1993).

[12] Leonardo M. Rodrigues, Carlos Montez, Ricardo Moraes, Paulo Portugal and Francisco Vasques. "A Temperature-Dependent Battery Model for Wireless Sensor Networks". In: Sensors 2017, 17, 422 (2017).

[13] Marshall Brain, Howstuffworks. How Lithium-ion Batteries Work. 2019. URL: https://electronics.howstuffworks.com/everyday-tech/lithium-ion-battery.htm.

[14] Martin Murnane, Adel Ghazel. "A Closer Look at State of Charge (SOC) and State of Health (SOH) Estimation Techniques for Batteries". In: Analog Devices (2017).

[15] Mathworks. Estimate States of Nonlinear System with Multiple, Multirate Sensors. 2016. URL: https://it.mathworks.com/help/ident/ug/multirate-nonlinear-state-estimation-in-simulink.html.

[16] Mathworks. Extended and Unscented Kalman Filter Algorithms for Online State Estimation. 2018. URL: https://it.mathworks.com/help/ident/ug/extended-and-unscented-kalman-filter-algorithms-for-online-state-estimation.html.

[17] Mathworks. Extended Kalman Filter. 2017. URL: https://it.mathworks.com/help/ident/ref/ekf_block.html#examples.

[18] Mohammad Charkhgard and Mohammad Farrokhi. "State-of-Charge Estimation for Lithium-Ion Batteries Using Neural Networks and EKF". In: IEEE TRANSACTIONS O (2010).

[19] MoreSteam. Design of Experiments (DOE). URL: https://www.moresteam.com/toolbox/design-of-experiments.cfm.

[20] NALIN A. CHATURVEDI, REINHARDT KLEIN, JAKE CHRISTENSEN, JASIM AHMED, and ALEKSANDAR KOJIC. "MODELING, ESTIMATION, AND CONTROL CHALLENGES FOR LITHIUM-ION BATTERIES". In: IEEE CONTROL SYSTEMS MAGAZINE (2010).

[21] Nora Manthey, electrive.com. Solid electrolyte batteries – the next big thing?! 2019. URL: https://www.electrive.com/2019/08/02/solid-electrolyte-batteries-the-next-big-thing/.

[22] Power Tech. Lithium-Ion State of Charge (SoC) measurement. 2019. URL: https://www.powertechsystems.eu/home/tech-corner/lithium-ion-state-of-charge-soc-measurement/.

[23] Robyn A. Jackey. "A Simple, Effective Lead-Acid Battery Modeling Process for Electrical System Component Selection". In: The MathWorks, Inc. (2007).

[24] Robyn Jackey, Aubrey da Cunha, Javier Gazzarri. Automating Battery Model Parameter Estin 2013. URL: https://it.mathworks.com/videos/automating-battery-model-parameter-estimation-using-experimental-data-81987.html?elqsid=1548087779778&potential_use=Student.

[25] Dan Simon. Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches. Cleveland: John Wiley and Sons Inc., 2006.

[26] Stefan Knupfer, Jesse Noffsinger and Shivika Sahdev. How battery storage can help charge the 2018. URL: https://www.mckinsey.com/business-functions/sustainability/our-insights/how-battery-storage-can-help-charge-the-electric-vehicle-market.

[27] Taesic Kim, Wei Qiao and Liyan Qu. "Real-Time State of Charge and Electrical Impedance Estimation for Lithium-ion Batteries Based on a Hybrid Battery Model". In: 978-1-4673-4355-8/13, IEEE (2013).

[28] Tarun Huria, Massimo Ceraolo, Javier Gazzarri and Robyn Jackey. "High Fidelity Electrical Model with Thermal Dependence for Characterization and Simulation of High Power Lithium Battery Cells". In: 71900, IEEE (2012).

[29] The Mathworks, Inc. "Predictive Analytics with MATLAB: Unlocking the Value in Engineering and Business Data". In: Mathworks Whitepaper (2016).

[30] The Mathworks, Inc. "Predictive Maintenance with MATLAB: Avoid costly equipment failures by using sensor data analytics". In: Mathworks Whitepaper (2016).

[31] The Mathworks, Inc. "Risolvere quattro problemi comuni legati alla manutenzione predittiva con MATLAB e Simulink". In: Mathworks Whitepaper (2018).

[32] Wikipedia. Lead–acid battery. 2019. URL: https://en.wikipedia.org/wiki/Lead%5C%E2%5C%80%5C%93acid_battery#cite_note-crompton-7.

[33] Wikipedia. Lithium-ion battery. 2019. URL: https://en.wikipedia.org/wiki/Lithium-ion_battery.