SAPIENZA
UNIVERSITÀ DI ROMA

University of Rome "La Sapienza"
Department of Information Engineering, Electronics, and Telecommunications

Ph.D. degree in Information and Communication Technologies
XXXI Cycle

Ph.D. Thesis

Design and Implementation of Machine Learning Techniques for Modeling and Managing Battery Energy Storage Systems

Candidate

Massimiliano Luzi
ID number 1146457

Thesis Advisor

Antonello Rizzi

Co-Advisor

Fabio Massimo Frattale Mascioli

February 2019

Thesis defended on February 18th, 2019
in front of a Board of Examiners composed of:

Prof. Antonio D'ALESSANDRO (chairman)
Prof. Francesco RIGANTI FULGINEI
Prof. Alessandro BUSACCA
Dott. Alessandra BUDILLON

This thesis has been reviewed by:

Prof. Giorgio RIZZONI
Prof. Federico BARONTI

Design and Implementation of Machine Learning Techniques for Modeling and Managing Battery Energy Storage Systems
Ph.D. thesis. University of Rome "La Sapienza"

Author's email: massimiliano.luzi@uniroma1.it

UNIVERSITY OF ROME "LA SAPIENZA"

Abstract

Faculty of Information Engineering, Informatics, and Statistics
Department of Information Engineering, Electronics, and Telecommunications

Doctor of Philosophy

Design and Implementation of Machine Learning Techniques for Modeling and Managing Battery Energy Storage Systems

by Massimiliano LUZI

The fast technological evolution and industrialization that have interested the humankind since the fifties has caused a progressive and exponential increase of CO_2 emissions and Earth temperature. Therefore, the research community and the political authorities have recognized the need of a deep technological revolution in both the transportation and the energy distribution systems to hinder climate changes. Thus, pure and hybrid electric powertrains, smart grids, and microgrids are key technologies for achieving the expected goals. Nevertheless, the development of the above mentioned technologies require very effective and performing Battery Energy Storage Systems (BESSs), and even more effective Battery Management Systems (BMSs).

Considering the above background, this Ph.D. thesis has focused on the development of an innovative and advanced BMS that involves the use of machine learning techniques for improving the BESS effectiveness and efficiency. Great attention has been paid to the State of Charge (SoC) estimation problem, aiming at investigating solutions for achieving more accurate and reliable estimations. To this aim, the main contribution has concerned the development of accurate and flexible models of electrochemical cells.

Three main modeling requirements have been pursued for ensuring accurate SoC estimations: insight on the cell physics, nonlinear approximation capability, and flexible system identification procedures. Thus, the research activity has aimed at fulfilling these requirements by developing and investigating three different modeling approaches, namely black, white, and gray box techniques. Extreme Learning Machines, Radial Basis Function Neural Networks, and Wavelet Neural Networks were considered among the black box models, but none of them were able to achieve satisfactory SoC estimation performances. The white box Equivalent Circuit Models (ECMs) have achieved better results, proving the benefit that the insight on the cell physics provides to the SoC estimation task. Nevertheless, it has appeared clear that the linearity of ECMs has reduced their effectiveness in the SoC task. Thus, the gray box Neural Networks Ensemble (NNE) and the white box Equivalent Neural Networks Circuit (ENNC) models have been developed aiming at exploiting the neural networks theory in order to achieve accurate models, ensuring at the same time very flexible system identification procedures together with nonlinear approximation capabilities.

The performances of NNE and ENNC have been compelling. In particular, the white box ENNC has reached the most effective performances, achieving accurate SoC estimations, together with a simple architecture and a flexible system identification procedure.

The outcome of this thesis makes it possible the development of an interesting scenario in which a suitable cloud framework provides remote assistance to several BMSs in order to adapt the managing algorithms to the aging of BESSs, even considering different and distinct applications.

Acknowledgements

I would like to express my special thanks of gratitude to Prof. Antonello Rizzi for the fundamental support provided during the past years, starting from the period of the Master's Thesis, until the last days of my Ph.D.. I will never forget his help and his suggestions. My life would be very different without our late-afternoon talks.

I would like to thank also my co-advisor Prof. Fabio Massimo Frattale Mascioli, Luigi Anniballi, and all the people from the POMOS laboratories for their valuable and undiscussed support and the opportunities otherwise I would not be able to enjoy. Special thanks go to Maurizio Paschero, not only for his help during my Ph.D. years, but also for having trusted me as a photographer for his daughter's baptism.

My most sincere gratitude is for Prof. Giorgio Rizzoni for the opportunity he gave me to work at the Center for Automotive Research, for his help, and for our guitar talking and the numerous jam sessions.

A very special thanks go to the "CAR-riola", Fabrizio Donatantonio, Nicola Zanardi, Matteo Galli, and Stefano Leonori, the most crazy Italian gang walking in the street of Ohio. All of you made the life in Columbus easier and happier. In particular, I would like to acknowledge Nicola, for the sincere friendship we have built.

I am also grateful with all my lab-mates, Stefano Leonori, Alessio Martino, Enrico Maiorino, Antonello Rosato, Rosa Altilio, Enrico de Santis, and Antonino Capillo. We spent great time together. In particular, a special mention is for Stefano, who shared with me all the good and bad moments of our Ph.D., supporting each other.

Last but not least, I would like to express my gratitude to all my family, in particular to my parents Agnese Mura and Patrizio Luzi, to my sister Annalisa Luzi and her husband David Camilli, and in particular to my darling nephew Alessandro. Further thanks are for my other family, Riccardo Deiana, Angelo Deiana, and Gabriele Marozzi for the not necessary words. My acknowledgement is also for my band, Davide Ubaldi, Dino Manoni, Daniele Sangiorgi, and Sandro Elia, and for all the numerous members of "La Banda".

Finally, my most important thanks are for Valentina Taschini, to whom this thesis is dedicated, simply for being her and nothing more. The past three years have been difficult for us, being far away to each other, but her sure presence, support, and help never made me feel the distance from her.

Contents

To Valentina

Chapter 1

Introduction

1.1 Background

Climate changes and global warming are some of the most relevant issues of modern era. Indeed, the wide technological evolution that has interested the nineteenth century has caused a progressive and exponential increase of the Earth temperature, as shown in Figure 1.1.

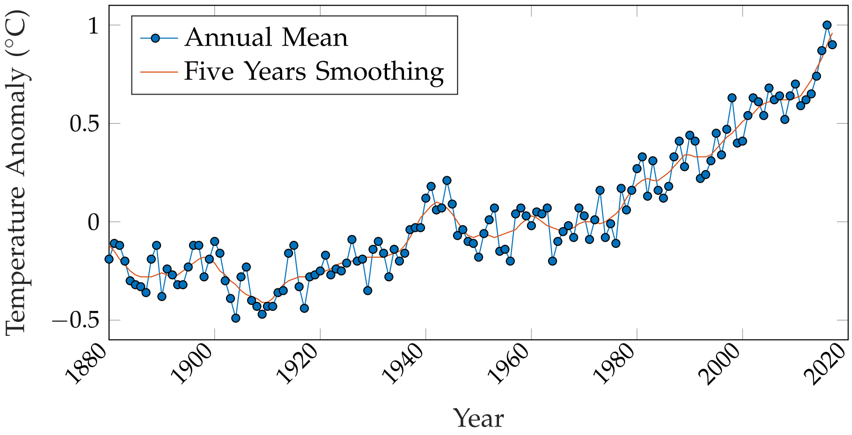


Figure 1.1: Variation of the land-ocean temperature with base period 1951-1980. Data downloaded from `data.giss.nasa.gov/gistemp/graphs`.

One of the main reasons of this increasing trend is related to the uncontrolled emissions of carbon dioxide (CO_2). In general, a suitable concentration of CO_2 and of the other natural gases composing the Earth atmosphere creates a beneficial greenhouse effect that has allowed and still allows a favorable environment for living beings. Nevertheless, any alteration on the concentration of the greenhouse gases results in a significant and long lasting variation of the climatic conditions. Unfortunately, since the Industrial Revolution until now the human activity has caused a progressive and unrestrained increase of the CO_2 concentration. In particular, this phenomenon has become progressively more relevant together with the wide industrialization and the very fast technological evolution that have interested the humankind since the fifties.

Several consequences are related to global warming and to the resulting climate changes. First of all, variations in the biological environment cause the extinction of several animal and plant species. Second, melting of ice is an immediate effect of global warming. Specifically, this results in retreat of glaciers, decline of Arctic sea ice, and rise of the sea levels. Third, global warming affects also the seasonal cycles, with consequent changes in several ecosystems, alterations in the plant and animal life cycles, anomalies in the rainfall rhythms, and extreme weather conditions. Other consequences are the acidification of oceans, expansion of deserts, and droughts. Moreover, all of these phenomena will affect also the human society, with alterations in the crop production and in general in the public health.

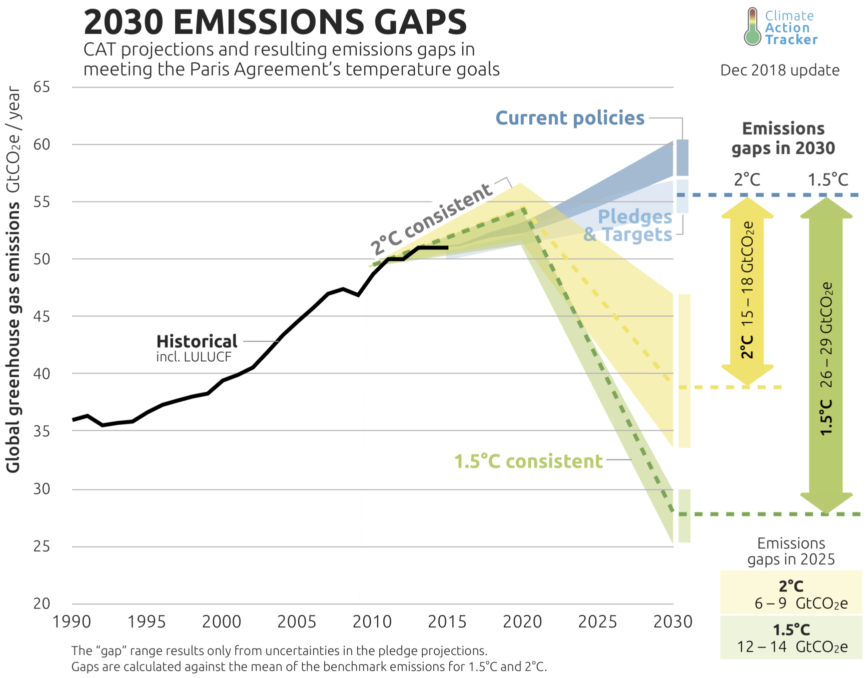
The above discussed negative situation has led both the research community and the political authorities at focusing on the mitigation of global warming. Therefore, 195 countries have attended to the *United Nation Climate Change Conference* held in Paris on December 2015 aiming at discussing about how to hinder climate changes. During this conference, it has been negotiated the *Paris Agreement* [1] in which the signatory countries has pledged to adopt solutions aiming at «holding the increase in the global average temperature to well below 2 °C above pre-industrial levels and pursuing efforts to limit the temperature increase to 1.5 °C above pre-industrial levels, recognizing that this would significantly reduce the risks and impacts of climate change»[1]. The agreement states that each country shall determine autonomously the best contribution for achieving the expected goals. Moreover, each of them has to make its economical system consistent with a low greenhouse gas emissions and climate-resilient environment. Indeed, each country shall make efforts «to reach global peaking of greenhouse gas emissions as soon as possible, [...] and to undertake rapid reductions thereafter in accordance with best available science, so as to achieve a balance between anthropogenic emissions by sources and removals by sinks of greenhouse gases in the second half of this century [...] »[2].

A graphical analysis of the expected goals and a forecast of the greenhouse gas emissions trend by 2030 is shown in Figure 1.2. The negative effect of the human activity held until now is evident. Indeed, from one side

[1] Article 2.1.a from the *Paris Agreement*.

[2] Article 4.1 from the *Paris Agreement*.

the historical data shows a constant increasing trend in the greenhouse gas emissions. On the other hand, the figure highlights that the policies and the pledges adopted so far are not satisfactory for fulfilling the expected goals.



The "gap" range results only from uncertainties in the pledge projections.
Gaps are calculated against the mean of the benchmark emissions for 1.5°C and 2°C.

Figure 1.2: Analysis of the CO_2 emissions and comparison with the *Paris Agreement*'s goals by December, 2018. Figure downloaded at www.climateactiontracker.org.

The main reason of the inadequacy of the adopted solutions is attributable to the ambitious and challenging technological revolution that is necessary for matching the goals of the *Paris Agreement*. From the industrial point of view, radical changes in the manufacturing process must be introduced in order to limit the emissions of the industrial plants. Consequently, many companies must provide huge investments for pursuing the aim of the agreement. The most immediate solution consists in the renovation of the industrial equipment. Nevertheless, often the simple modernization of the plants could not be sufficient for matching the very demanding requirements concerning the greenhouse gas emissions. Therefore, the most compelling solution consists in a deep redesign of the entire manufacturing processes in order to develop plants natively *green*. It is clear that this process is very demanding for many companies, and it requires structural changes and suitable settling times.

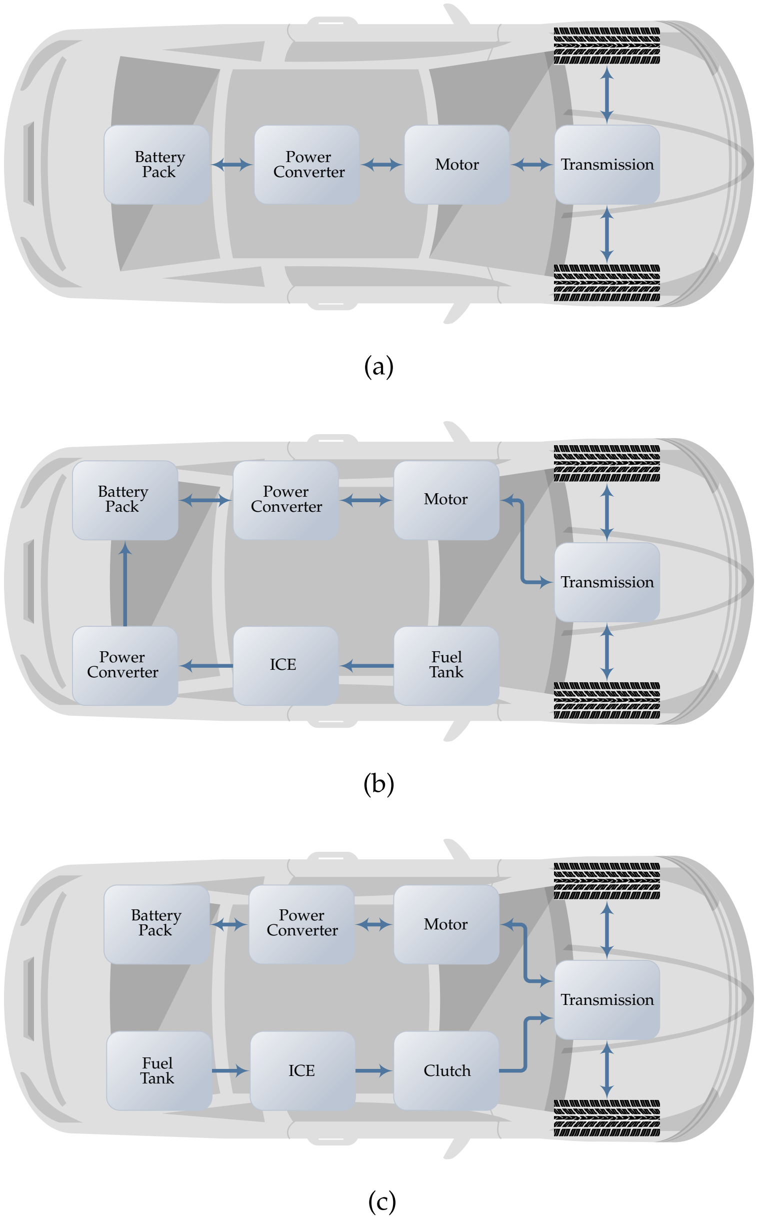
Besides the industrial area, the transportation and the Energy Distribution System (EDS) are widely recognized as the two main fields where to act for hindering global warming.

The transportation system has been historically based on fossil fuel propulsion. The main advantage of this solution has been always recognized in the very high energy density of fuel that ensures very fast refills and long distance ranges. Thus, gasoline-powered vehicles have dominated the transportation technology so far even if electric cars have got a reliable production with Thomas Parker in 1884 [2] about 20 years before of the *Ford Model T*. Nevertheless, traditional Internal Combustion Engines (ICEs) are effective but inefficient machines [3]. Indeed, the average efficiency is around 20 %, meaning that most of the fuel energy is dissipated in heat and in the production of waste gases, in particular CO_2. Therefore, considering also that the total number of vehicles in use worldwide has increased from 126 millions of the sixties to 1.3 billions in 2014, it is straightforward to think about conventional cars as one of the most relevant causes of the increasing trend in the CO_2 concentration.

Several actions have been done aiming at reducing the greenhouse gas emissions of conventional vehicles, such as the introduction of international rules like the European Emission Standards. The adoption of these rules have forced car makers at finding solutions for developing more efficient ICEs and for introducing novel strategies to limit the related emissions. Nevertheless, these solutions alone are far to be sufficient for achieving the goals of the *Paris Agreement*. Moreover, the conventional transportation technology is going to be completely unsustainable considering that the number of cars is going to further increase in the next future. Therefore, many countries and car makers have recognized the need of coming back to the electric propulsion, and to convert the transportation technology to Electric Vehicles (EVs) and Hybrid Electric Vehicles (HEVs) [4], [5].

A simplified block diagram of the EV and HEV architectures is shown in Figure 1.3. The powertrain of an EV is in general very simple. It consists of a battery pack, a power converter, and an electric motor. The power converter is in charge of performing two tasks. The first one consists in converting the DC voltage of the battery pack to the suitable voltage specifications of the electric motor. Very often, the power converter is actually an inverter performing a DC-AC conversion because of the common use of AC induction motors for the wheels traction. The second task is to control the amount of power to be provided to the electric motor accordingly with the power required from the driver.

HEVs have a more complicated architecture because electric and conventional propulsion coexist in the same powertrain. HEVs can be split in two



(a)

(b)

(c)

Figure 1.3: Powertrain architecture of electrified vehicles. (a) EVs. (b) Series-HEVs. (c) Parallel-HEVs.

main categories: series-hybrid and parallel-hybrid. In the series architecture only the electric motor provides traction to the driving wheels, whereas the ICE works as a range extender being connected to a generator in order to recharge the battery pack. Two power converters are required: the first one is connected between the battery pack and the electric motor having the same role seen for EVs. The second power converter is connected between the generator and the battery pack in order to allow the recharging process. Likewise EVs, the required power must be satisfied only by the electric propulsion. In the parallel-hybrid architecture both the ICE and the electric motor can drive the wheels. Therefore, the power required by the user must be satisfied by means of an intelligent control of the powertrain. Indeed, it is possible to recognize at least four different operating modes. The first one is the full electric propulsion, in which only the electric motor is used for driving the wheels. Similarly, only the conventional propulsion is used in the ICE mode. The third is the hybrid mode, in which the required power is satisfied by using the ICE and the electric motor together. Finally, in the regenerative breaking mode the inertia of the car is used to recover energy for recharging the batteries. Moreover, in some architecture it is possible to introduce a further generator that is connected to the ICE in order to create a more complicated series-parallel architecture. In this case, it is possible to have a further mode in which the wheels are driven only by the electric motor, while the ICE is used for recharging the batteries through the related generator.

Besides the zero and the low emissions properties, respectively, EVs and HEVs have the further advantage of a more efficient powertrain with respect to the conventional ones. Indeed, electric motors can achieve efficiency up to more than 90 % on the whole rotational speed range. Moreover, very often electric motors do not need a multi-gear transmission system because they can handle higher rotational speeds with respect to ICEs. Consequently, it is possible to avoid complex transmission gearboxes, further improving the efficiency of the powertrain. Nevertheless, the main drawback of EVs is their limited autonomy and limited flexibility of the refill process resulting in the *range anxiety* commonly perceived by the drivers. HEVs provide a suitable palliative to these drawbacks thanks to the installation of the ICE in their powertrain, and for this reason their commercialization is being more prominent with respect to EVs. Nevertheless, the actual reduction of CO_2 emissions lays widely in a consistent diffusion of EVs, and therefore it is necessary to furtherly improve their performances, as well as to create the most suitable environment for their diffusion, starting from a wide revolution of the EDS.

EDS has been based so far on the concept of centralized power generation and decentralized loads. In this context, the most diffused power plants have relied on fossil fuels, where electric energy is harvested by burning coal, natural gas, or petroleum [6], [7]. It is straightforward that also the above described EDS has contributed in the increasing trend of CO_2 emissions and global warming. Moreover, the traditional grid is also characterized by a very rigid and inefficient operating condition. First, its stability is deeply dependent on the synchronous AC electric machines of the centralized power plants. As a direct consequence, the traditional grid technology itself makes harder the transition to alternative and greener power sources [8]. Second, the centralized architecture implies that the distribution must be organized and operated following a tree architecture in which often the plants are far away from the actual loads. Thus, the energy has to flow among very long power lines, power transformers, transmission lines, transmission substations, distribution lines, distribution substations, and finally to the loads. This means that the entire architecture has an intrinsic lack of efficiency because of the losses unavoidably present in each node of the tree. Moreover, this strict operating architecture has two further implicit drawbacks. First, customers do not have any consciousness of the grid, of the generation processes, of the energy prices, and consequently they have a very limited opportunity of participating to the energy market. Second, it is very hard to control the grid, and it is almost impossible to think about a dynamic reconfiguration of the distribution architecture. This results in a further lack of efficiency and in an implicit weakness of the grid, because possible failures can easily lead to blackouts [9], [10]. For the above discussed reasons the traditional EDS looks widely inadequate for facing the more and more increasing energy demand due to the deep pervasiveness of technology and the incoming diffusion of EVs that will characterize the modern society. Moreover, it is likewise inadequate for reducing CO_2 emissions. Therefore, a deep re-design of the entire EDS is necessary. In particular, the smart grid [7], [11]–[14] and the microgrid [8], [15], [16] concepts have been recognized as the most promising technologies for implementing the next generation EDS.

Smart grid and microgrid aim at creating a suitable environment for converting the rigid, weak and inefficient distribution system based on centralized fossil fuel power plants to a flexible and reliable grid based on decentralized Renewable Energy Sources (RESs) [17]–[19]. From the smart grid perspective, the distribution system must be revolutionized through a deep introduction of Information Technology (IT) tools [7]. The aim is to create a constant two-way communication between suppliers and customers, as well as to equip each node with suitable *smart meters* aiming at monitoring in real-time the operative condition of the grid. The introduction of IT tools allows to increase the flexibility of the distribution system, and to prepare the grid for an effective use of RESs. Indeed, the constant monitoring of the system provides a real-time full knowledge of both the generation and the load status. If this information were unnecessary for the traditional grid since both generation and loads were considered substantially constant along the network, the installation of RESs, Energy Storage Systems (ESSs), DC fast charge station for EVs, as well as the changed habits of customers make generation and load profiles very variable both temporally and spatially. Therefore, a constant monitoring of the entire grid is mandatory for achieving a more effective operation of the system. Moreover, a deep use of sensors makes EDS more reliable and efficient. Indeed, it permits a prediction and a prompt detection of incoming failures, as well as the actuation of remote fixing, whenever applicable. Furthermore, the decentralized architecture allows an adaptive reconfiguration of the grid on the basis of the actual generation-load status, aiming at increasing the efficiency of the entire system.

The constant communication between suppliers and customers is the other foundation concept of smart grid. This communication aims at making customers more active and more conscious of the energy market, being informed in real-time about the energy prices and the grid status. Thus, customers can have an active participation in the distribution system, also because they are encouraged at being local suppliers by means of personal RESs, as well as personal ESSs, Plug-in EVs (PEVs), or Plug-in HEVs (PHEVs) [20]–[22].

The above described foundation concepts imply that next generation smart grids shall be composed of Zero-net Energy Nodes (ZENs) [8], namely part of the grid able to reach a self-sufficient energy balance by taking advantage of local RESs and local ESSs. Microgrids are the most feasible candidates for implementing a ZEN node. A microgrid is an atomic EDS that behaves like a single entity with respect to the main grid [23]. It is composed of interconnected loads, local generation plants, mainly RESs, and ESSs. Thanks to the local generation and the presence of ESSs, a microgrid can be completely self-sufficient with respect to the grid, and therefore it can work in an islanded mode. Nevertheless, the most challenging scenario sees the microgrid still connected to the main grid, such that the customer can acts as a "prosumer", i.e. both an energy producer and consumer at the same time. Therefore, microgrids play a key role in the implementation of the smart grid concept, involving each customer at having an active role in the energy market. Indeed, each microgrid must be equipped with effective Energy Management Systems (EMSs) for achieving a more efficient, more effective, smarter, and more remunerative use of the available energy [24], [25].

It is clear from the above discussion the importance that EVs, HEVs, smart grids, and microgrids have for leading the radical technological and social revolution required by the *Paris Agreement*. Although these technologies belong to different application fields, all of them share ESSs as an invariant device for achieving significant improvements with respect to the conventional technologies.

Concerning EVs and HEVs, the importance of ESSs is straightforward. The availability of effective and reliable battery packs is the first step for improving the vehicle performances, as well as for ensuring the actual commercialization of HEVs and especially EVs. Indeed, most of the restraints on the diffusion of electrified cars comes from ESS related issues, such as range anxiety and high prices. Therefore, it is mandatory to focus on the development of effective and reliable ESSs both for lowering the prices and for improving the vehicle performances in terms of speed and distance range.

From the EDS perspective, the installation of ESSs is mandatory for facing the stochastic nature of RESs and the variable operating condition characterizing the incoming smart grid and microgrid environment [26]. The first important application of ESSs consists in facing the fluctuation of the generation profiles characterizing RESs. Indeed, the energy production of photovoltaic plants and wind turbines is strictly dependent on the actual atmospheric conditions. Therefore, it could happen that RESs have a generation peak when that energy is not required, or conversely the actual generation is not sufficient for satisfying the loads request. Thus, ESSs are useful for implementing an energy buffer that stores the overproduced energy for supplying it when necessary. In addition to this application, ESSs are useful also for facing the increased aleatory nature of load profiles due to the diffusion of EVs and of the related charging stations. Indeed, thinking about a larger and larger fleet of EVs, each car represents an unpredictable load that can be connected to the grid in any time and in any node. Moreover, this problem is even worse considering the incoming diffusion of DC fast charge stations, since each connected EV is a full stochastic and unpredictable load greater than 50 kW [27]–[29]. In this context, ESSs will be key devices for addressing these load peaks that otherwise the traditional grid would not be able to cope with. Besides performing peak shaving of both generation and load profiles, the installation of ESSs allows the implementation of a smarter and more effective management of the energy flows. Indeed, ESSs introduce a further degree of freedom that can be exploited for implementing innovative and beneficial strategies for handling the energy production and loads, especially in the microgrid environment. In this context, it is possible to think about strategies in which the prosumer takes advantage of the local RESs and ESSs in order to make profit. To this aim, it is necessary to decide when it is more advantageous charging or discharging the ESSs, when it is more convenient buying or selling energy from/to the main grid, as well as to schedule the activity of the home appliances. Therefore, it is necessary to implement smart EMSs whose task is to manage the energy flows for optimizing suitable objectives [24]. Herein, machine learning techniques play a key role for developing innovative EMSs, and to this aim several solutions have been already proposed in the literature [15], [16], [30], [31].

Several technologies are available for implementing ESSs. The most diffused ones can be classified in four macro-areas depending on the physical nature of the stored energy, namely mechanical, electrochemical, electrical, and chemical [26], [32]. A summary of the most important ESSs technologies divided by the physical nature of stored energy is shown in Table 1.1.

Table 1.1: Most important ESSs technologies

Mechanical	**Electrochemical**	**Electrical**	**Chemical**
Pumped Hydroelectric Compressed Air Flywheel	Lead-acid Nickel-cadmium Nichel-metal Hydrid Li-ion Sodium-sulfur Flow Battery	Capacitor Supercapacitor Superconducting Magnetic	Fuel Cell

Pumped Hydroelectric Storage (PHS) [33], Compressed Air Energy Storage (CAES) [34], and Flywheel [35] belong to the mechanical category.

PHSs are the most common and mature energy storage technology for EDSs. In PHSs energy is stored and dispatched by means of two water reservoirs placed at two different altitude levels. Specifically, the water is pumped from the lower to the upper reservoir for storing energy, whereas it is released back to the lower reservoir to get the energy back by driving suitable power turbine units. The effectiveness of PHSs is ensured by their high maturity level and their wide use worldwide. However, their actual installation is restrained by environmental issues, location limitations and high costs, as well as their application is limited by the small energy and power densities.

In CAES a reversible motor/generator and a corresponding compressor-turbine unit are used for converting electricity to compressed air and the other way around. During the energy storing phase, the motor drives the compressor that forces air into a suitable natural or artificial reservoir. Conversely, the stored air drives the turbine for getting back energy through the generator. CAESs offer flexible sizes and durability. Nevertheless, their limitation consists in low cycle efficiency (40-70 %), difficulties at finding suitable geographical environments for their installation, high capital costs, and small energy and power densities.

Flywheels are very simple and intuitive ESSs converting electrical energy in the kinetic one. Specifically, they consist of a flywheel and a reversible motor/generator. The motor drives the flywheel accelerating it for storing energy, whereas the flywheel decelerates while its inertia drives the generator for getting the stored energy back. The main advantage of flywheels is their high cycle efficiency (90-95 %), absence of depth-of-discharge effects, and easy maintenance. However, their main limitation consists in a very high self-discharge ratio that implies the impossibility of storing energy for long-term periods.

The electrochemical category is composed of rechargeable and flow batteries, both of them characterized by storing and delivering energy by means of redox reactions [26], [36].

Rechargeable batteries consist in several electrochemical cells that are connected in series and in parallel for achieving the nominal voltage and nominal energy required by the specific application they are used for. Each electrochemical cell consists in three main components: two electrodes (cathode and anode) immersed in one electrolyte. During the discharge phase a redox reaction causes electrons flowing from the anode to the cathode through the external system, and thus providing energy to it. Conversely, during the charge phase the redox reaction is inverted by applying a suitable voltage to the cell.

Several electrochemical technologies are available depending on the specific materials used for the anode, cathode, and electrolyte.

The most mature is the lead-acid technology, in which PbO_2, Pb, and sulfuric acid compose the cathode, the anode, and the electrolyte, respectively [37]. Lead-acid cells offer a good cycle efficiency (60-90 %), as well as small self-discharge ratio and low capital costs. However, their main drawback is the limited life, together with a limited energy density and specific energy, meaning that lead-acid ESSs are in general voluminous and heavy devices.

Nickel-Cadmium (NiCd) cells have nickel hydroxide as anode and metallic cadmium as cathode, whereas the electrolyte is an alkaline solution. They have a higher energy and power densities, as well as higher specific energy with respect to lead-acid cells. Nevertheless, the actual use of NiCd batteries is highly restrained by the presence of toxic metals (nickel and cadmium), and because they suffer from a prominent memory effect that can easily compromise their effectiveness. Therefore, it has been developed the Nichel metal Hydride (NiMH) technology, in which a hydrogen-absorbing alloy is used as cathode in place of cadmium. These cells have similar properties of the NiCd technology, but with the advantage of a reduced memory effect, a higher energy density, and the use of environmental friendly materials. Despite that, NiMH cells suffer from a high self-discharge ratio and a strong dependence on the depth-of-discharge effect that limit their lifetime.

Li-ion electrochemical cells use a lithium metal oxide for the anode, such as $LiCoO_2$, $LiMO_2$, or $LiFePO_2$, and graphite or carbon for the cathode [38]. The electrolyte is an organic liquid containing dissolved lithium salts. The main advantages of this technology are the high cycle efficiency (80-97 %), the high energy and power densities, and the high specific energy and specific power. Moreover, lithium cells do not suffer from the memory effect, and they have a negligible self-discharge ratio. However, the main drawback is the prominent sensibility to the depth-of-discharge effect that requires a very careful management of them for avoiding damages that can compromise their effective use. In addition, lithium cells still have a higher capital cost with respect to other electrochemical technologies.

Sodium-sulfur cells (NaS) have melted sodium and melted sulfur as cathode and anode electrodes, respectively, whereas beta alumina composes the electrolyte [39], [40]. These cells have high energy density, higher rated capacity with respect to other electrochemical technologies, negligible self-discharge ratio, and high pulse power capabilities. Furthermore, sodium and sulfur are cheap, non-toxic, and highly available materials. However, the working temperature must be set between 300 °C and 350 °C in order to ensure both the electrodes are melted. This is the main disadvantage of this technology, implying high operating costs and the presence of a dedicated equipment for ensuring the working temperature.

Flow batteries are electrochemical ESSs storing energy in two external tanks where two redox couples are dissolved in liquid electrolytes [36]. Besides the tanks, a flow battery is composed of two electrodes connected to related compartments separated by a ion selective membrane. When charging, the two electrolytes are pumped to the respective chambers so that one electrolyte is oxidized and the other is reduced. The reverse reaction happens when discharging. Flow batteries are very flexible and scalable ESSs since the stored energy depends only on the size of the electrolyte tanks, whereas the dispatchable power depends on the size of the electrodes. Moreover, they have small self-discharge ratio. Despite that, flow batteries can lose performances depending on non-uniform pressure drops and limitation on the transfer of the electrolytes. Moreover, they have a high manufacturing cost and they make use of toxic materials.

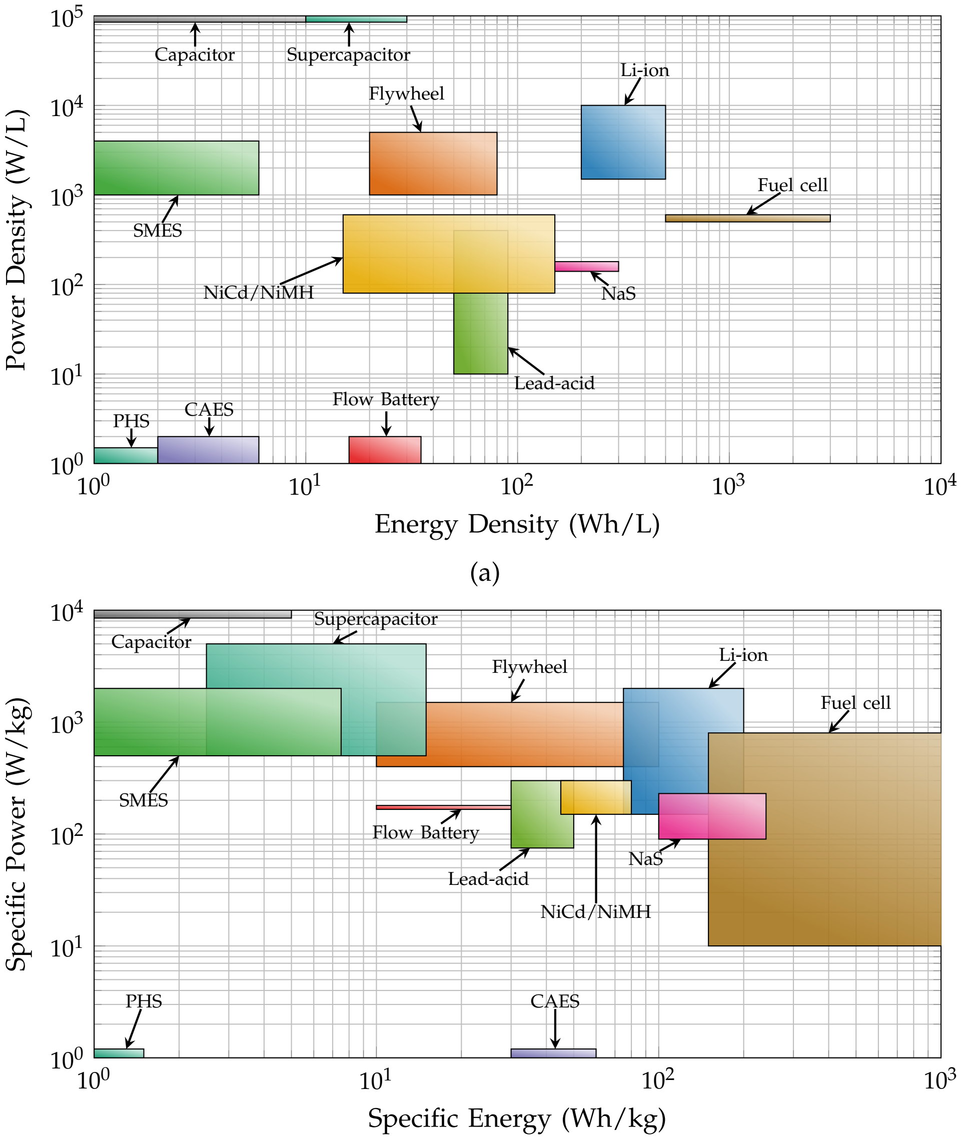
Capacitor, supercapacitor and Superconducting Magnetic Energy Storage (SMES) belong to the electrical category of ESSs.

Capacitors are a basic electrical device composed of two electrodes separated by a very thin dielectric layer. Energy is stored in the static electric field generated by the accumulation of charge in the two electrodes. Supercapacitors are actually standard capacitors characterized by a very high capacitance [41]. Capacitors and supercapacitors are power oriented devices being characterized by a very high power density and specific power. Conversely, both of them are not able to store a large amount of energy. Other advantages are their almost linear behavior, their long life, and the high cycle efficiency (84-97 %). Nevertheless, besides the very low energy density and specific energy, capacitors and supercapacitors are affected by a significant self-discharge phenomenon.

An SMES system stores energy by inducing a DC current in a superconducting coil [42]. The main components of an SMES device are a refrigerator, a power converter, and the superconducting coil. In particular, the refrigerator is necessary for bringing the environment temperature below of the superconducting threshold, that is around 9 °K. The power converter is used for inducing the current to the coil when charging, and to get the energy back when discharging. SMESs have long lifetime, high cycle efficiency (95-98 %), and an almost absent degradation from the depth-of-discharge. However, SMES is a young technology, and therefore it is characterized by a high capital and maintenance cost, as well as a prominent self-discharge ratio and a high sensibility to the temperature divergence.

Fuel cells are chemical ESSs in which energy is stored in high pressure tanks of hydrogen [43]. The storing and the dispatching of energy is performed by two separate and independent processes. For storing energy it is necessary to produce hydrogen, and an example is water electrolysis. On the other hand, the fuel cell device performs the conversion from the stored hydrogen to the electric energy. The advantage of the separate processes make fuel cells very flexible and scalable ESSs, achieving a wide range of power and energy densities. However, their actual use is limited by a very limited efficiency (20-60 %), and by the hazard related to the high pressure tanks used for storing hydrogen.

A graphical comparison of the above described ESS technologies in terms of energy and power densities, specific energy and specific power is shown



(a)

(b)

Figure 1.4: Comparison among different ESS technologies [26], [32]. (a) Energy density vs power density. (b) Specific energy vs specific power.

in Figure 1.4, whereas Table 1.2 shows a comparison in terms of cycle efficiency, self discharge rate, lifetime, number of cycles, power capital cost, energy capital cost.

Each ESS technology has different and specific beneficial properties, as well as corresponding drawbacks. Therefore, it is impossible to determine a dominant technology that suits successfully any kind of application. Nevertheless, the brief overview discussed in the above and the comparisons shown in Figure 1.4 and Table 1.2 highlight as Li-ion ESSs offer compelling performances in a greater number of technical properties with respect to the other ESS technologies. First, lithium cells show the overall best trade-off

Table 1.2: Comparison among different ESSs technologies [26], [32]

Technology	Cycle Efficiency [%]	Daily Self Discharge [%]	Lifetime [years]	Number of Cycles	Power Capital Cost [$/kW]	Energy Capital Cost [$/kWh]
PHS	70-85	∼0	40-60	10000-30000	2500-4300	5-100
CAES	40-70	∼0	20-40	8000-12000	400-1550	2-250
Flywheel	90-95	100	15-20	>20000	250-350	1000-14000
Lead-acid	60-90	0.1-0.3	5-15	200-1800	200-600	50-400
Ni-Ca	60-83	0.2-0.6	3-20	2000-4500	500-1500	400-2400
Ni-MH	60-83	5-20	3-20	2000-4500	500-1500	400-2400
Li-ion	80-97	0.1-0.3	5-15	1000-20000	900-4000	600-3800
NaS	75-90	∼0	10-20	2500-4500	350-3000	300-500
Flow Battery	65-85	∼0	5-20	2000-12000	200-2500	150-1000
Capacitor	60-70	40-100	1-10	>50000	200-400	500-1000
Supercapacitor	84-97	10-40	10-30	>50000	100-450	300-2000
SMES	95-98	10-15	20-30	>20000	200-489	500-72000
Fuel Cell	20-60	∼0	5-20	1000-20000	500-3000	2-15

among energy density, power density, specific energy, and specific power. Therefore, it is possible to develop very compact and light ESSs without losing performances both in terms of stored energy and deliverable power. Second, the negligible self-discharge ratio and the absence of the memory effect allow long periods of inactivity, as well as a frequent charge of the battery pack without jeopardizing the effectiveness of the cells. In addition, Li-ion cells have one of the most high cycle efficiency with respect to all the other technologies [26], [36], they offer a lifetime comparable to the other electrochemical ESSs, and they do not require specific environment conditions for their effective use.

All the previous mentioned advantages make lithium cells the most promising ESS technology in the field of portable electronics, automotive industry, and EDS. Indeed, Li-ion ESSs have become the dominant storing technology so far for notebooks, smartphones, cameras, and so on, as well as it is becoming the main technology also for EVs, HEVs, and microgrid. The main reason of that is because all the above mentioned applications are characterized by very rigid requirements concerning the packaging of the battery pack both in terms of weight and volume. Moreover, they can involve frequent long lasting idle periods, frequent transitions between charging and discharging phases, as well as partial charge/discharge cycles. It is evident that lithium cells are the ESS technology that mostly cover the requirements of these applications, especially the electrified transportation and the microgrid environment. Nevertheless, Li-ion ESSs are very fragile, and they need to be carefully managed for achieving an effective and safe use of them. This is because they are strongly affected by the depth-of-discharge effect, as well as lithium is a very reactive and flammable element. Indeed, the chemical processes being at the basis of the cell functionality can be irreversibly compromised in case of a severe over-discharge condition. Therefore, lithium cells must not be deep discharged in order to prevent any early aging and loss of capacity. Similarly, a severe over-charge can easily result in damages, loss of performances, as well as fire or even explosions. Moreover, although lithium cells can work in a wide temperature range from -20 °C to 60 °C, a high operating temperature can result in an early aging of the cells [44], and it can contribute at making lithium even more prone to fire. Therefore, an accurate management of lithium cells is mandatory and any Li-ion ESSs must be equipped with a dedicated device called Battery Management System (BMS).

Any BMS is an ensemble of hardware and software components aiming at monitoring, managing, and protecting Battery ESSs (BESSs). The main task of any BMS is to safeguard the safety and the health of every cell by ensuring that each of them works in a Safe Operating Area (SOA) concerning the terminal voltage, the flowing current, and the operating temperature. This task is typically performed by limiting the charging/discharging current or at least by disconnecting the entire battery pack in case of severe violations of the SOA boundaries. The second task of BMS is to evaluate the State of Charge (SoC) of each cell, providing information about the residual energy stored in them. Third, it has to perform State of Health (SoH) estimation in order to retrieve information about the residual ability of each cell at storing energy. Finally, the last task is cells balancing, aiming at ensuring that each cell of the BESS works in an operating point as much as possible similar to that of the others. Indeed, even if the battery pack is built by selecting and balancing the cells, the randomness of the manufacturing process and of the aging effect cause a progressive divergence of their operating points.

Besides the fundamental task of ensuring safety and health, BMSs are key devices for significantly improving the performances of BESSs, in particular the lithium ones.

The first benefit comes from an effective balancing of the battery pack. Indeed, the undesired divergence of the cells operating points is one of the main reasons of the gradual loss of performances characterizing BESSs. This is because an unbalanced battery pack has a reduced operating range since the most charged cell can easily violate the upper boundaries of the SOA, as well as the most discharged one can violate the lower boundaries simultaneously. Therefore, a balanced BESS is able to better exploit the stored energy, allowing to span a wider range of SoC, resulting in an increased cycle duration. Moreover, since all the cells work in similar operating points, also their aging will be similar.

Concerning SoH, an accurate determination of this quantity is critical for tracking the aging of the BESS, and specifically that of each cell. Accurate SoH estimations are beneficial for two reasons. First, considering that BESSs are considered to become unfeasible when they lose more than 20 % of the original capacity, any error in the SoH estimation can result in a early disposal of healthy battery packs. Second, just because of the above mentioned rigid health requirements, it is possible to think about second life applications of dismissed BESSs in order to amortize the capital cost of the first installation. Therefore, it will be very beneficial an accurate SoH estimation in order to get an exact knowledge of the residual capacity of second life battery packs.

Among the tasks belonging to BMS, SoC estimation is surely the most critical one. Indeed, an accurate knowledge of the SoC value is of crucial importance for maximizing the effectiveness not only of the BESS, but also of the whole energy system. This is because the energy management algorithms running on any EMS deeply involve the accurate knowledge of the current SoC in order to improve the overall efficiency of the energy flows [31], [45]. It is clear that any estimation error can result in a loss of effectiveness of the entire energy system. Moreover, a reliable and accurate SoC estimation is helpful for performing all the other tasks of BMS, allowing to monitoring in real-time the actual working point of the cells, the over-charge and over-discharge conditions, to provide useful information for cells balancing, as well as to track the loss of capacity.

1.2 Objective of the Research

Considering the strategic role that BMSs have in the incoming technological revolution concerning both the automotive and the energetic distribution areas, this Ph.D. thesis has focused on the development of an innovative, reliable, and advanced BMS that largely involves the use of machine learning techniques for improving the BESS effectiveness and efficiency.

The research activity has focused mainly on the software part of BMSs, being most of the improvements concerning this device depending on the introduction of innovative and more effective algorithms. In particular, because of its critical importance, great attention has been paid to the SoC estimation task, aiming at investigating solutions for achieving more accurate, robust, and reliable estimations. To this aim, the most promising methods proposed in the literature have been considered and analyzed. Herein, the BMS performances at estimating SoC are strictly dependent on the availability of accurate models of electrochemical cells. Therefore, the main objective of this thesis has been the study and the design of accurate and flexible models of electrochemical cells to be used for improving the BMS performances at estimating SoC, as well as at performing the other tasks.

The research activity on electrochemical cell models has been conducted aiming at fulfilling three main requirements considered mandatory for ensuring accurate and reliable SoC estimations. First, the model shall provide useful insights on the physics of the cell for exposing information about SoC. Second, it shall track and approximate in the most accurate way all the nonlinearities characterizing electrochemical cells. Third, system identification shall be as flexible as possible. In particular, the first and the second requirements have been considered necessary for improving the performances of the SoC estimator at providing accurate and reliable SoC estimations. On the other hand, the third property has aimed at allowing a frequent update of the model parameters in order to track the aging of the cells. More precisely, a flexible system identification procedure allows the task to be performed on generic data, avoiding then to disassemble the BESS for performing specific, long lasting and expensive offline tests.

Several modeling techniques have been analyzed looking for the solution that best fulfills all the above described requirements. To this aim, machine learning algorithms have been deeply investigated because it has been a firm conviction that their data-driven approach and their capability of dealing with nonlinearities and uncertainties would be very helpful for developing electrochemical cell models being simultaneously effective at performing SoC estimation and characterized by a flexible system identification procedure. Thus, three different approaches have been investigated: black box, gray box, and white box techniques. In particular, black box models offer the most flexible system identification and an easy nonlinear modeling as well, but they provide a limited insight on the cell physics. White box models offer a complete insight on the physics, but they do not allow a flexible system identification procedure, as well as they could have a limited nonlinear approximation capability. Therefore, part of the research activity has focused on the development of suitable methods for making system identification more flexible for white box models. Finally, a promising research field has been investigated by developing customized gray box and white box neural networks models. Indeed, the use of customized neural networks architectures have appeared particularly suitable for modeling electrochemical cells, allowing to keep satisfying insight on the cell physics, together with a flexible system identification and a nonlinear approximation capability.

1.3 Overview on the Research Topic

1.3.1 Battery Management System

Monitoring

The basic task of any BMS is monitoring the status of each cell composing the BESS. This task is straightforward and it is performed by the real-time measurement of the main physical quantities related to the cell activity. Specifically, these quantities are voltage, current, and temperature. To this aim, BMSs are typically organized in a master-slave architecture. The master unit is devolved to the computational tasks and to the consequent analysis of the measured quantities. On the other hand, the slave device is equipped with sensors and actuators. In particular, the sensors perform the required measurements of voltage, current, and temperature, whereas the actuators are the electronic hardware needed for performing cells balancing, as well as for opening the main switch of the battery pack in case of severe hazards. Moreover, often the slave device is also equipped with a very simple monitoring interface. This is implemented with a microcontroller that allows to set the lower and upper SOA boundaries related to voltage and temperature, and to throw a warning flag in case some of the cells violate these limits.

Several commercial slave devices are available in the market performing both the required measurements and cells balancing. Moreover, most of them are equipped with several General Purpose Input Output (GPIO) pins for controlling different actuators, mainly electronic switches. Examples are the products from *Linear Technology* such as *LTC 6804*[3] or *LTC 6813*[4], and the devices from *Texas Instruments* such as *TIDA-00553*[5] or *bq76PL455A-Q1*[6].

All the above mentioned devices are equipped with a suitable communication interface for transmitting the measurements and the warning flags to the master. In particular, besides the other computational tasks, the master

[3] www.analog.com/en/products/ltc6804-1.html

[4] www.analog.com/en/products/ltc6813-1.html

[5] www.ti.com/tool/TIDA-00553

[6] www.ti.com/tool/BQ76PL455EVM

collects these information and performs a second and more accurate monitoring of the working condition of the cells, as well as it processes this data for a more suitable visualization and analysis.

State of Charge

SoC estimation is the most critical task of BMSs. Indeed, besides providing information about the residual stored energy, an accurate and robust determination of SoC helps at maximizing life, efficiency, and effectiveness of each cell and of the whole BESS. Moreover, it makes easier performing the other tasks, such as cells balancing and SoH estimation.

SoC is defined as the ratio between the actual stored charge and the capacity C_n. More precisely, C_n is the total amount of charge that the cell can store between the minimum and the maximum allowed voltage. Consequently, SoC is a number belonging to the range $[0, 1]$, with 0 and 1 meaning that the cell is full discharged and full charged, respectively. Very often, SoC is expressed in a percentage unit by multiplying the ratio by 100. Thus, let $Q(t)$ be the actual stored charge at time t, SoC is defined as follows:

$$\mathrm{SoC}(t) = \frac{Q(t)}{C_n}. \tag{1.1}$$

Although the definition is very simple, SoC estimation is still considered an open problem both from the research community and the industrial field. Indeed, being SoC related to an internal state of the cell, it cannot be directly measured and it has to be inferred from indirect information, specifically the measurements of current, voltage, and temperature [46], [47].

Several methods have been proposed in the literature, and they can be divided in four main categories: coloumb counting, open circuit voltage, machine learning, and state observers [48].

The coloumb counting approach estimates SoC by evaluating the stored charge $Q(t)$ through the integral of the charging/discharging current [49], [50]. Therefore, SoC is evaluated as follows:

$$\mathrm{SoC}(t) = \mathrm{SoC}(t_0) + \frac{1}{C_n}\int_{t_0}^{t} I_{\mathrm{in}}(t)dt \tag{1.2}$$

where I_{in} is the input current of the cell and it is considered positive during charging, t_0 is the initial instant, $\mathrm{SoC}(t_0)$ is the initial value of SoC, and C_n is the cell capacity expressed in Ah. Nevertheless, both the measurements and the data elaboration are performed in discrete time. Therefore, coloumb counting is typically performed with the following discrete expression:

$$\mathrm{SoC}[k+1] = \mathrm{SoC}[k] + \frac{T_s}{C_n} I_{\mathrm{in}}[k] \tag{1.3}$$

where k is the temporal index, and T_s is the sampling time.

Despite the very simple and intuitive approach, coloumb counting is not a suitable method for several reasons. First, being based on the discrete approximation of an integral, it is affected by error accumulation that progressively compromises the estimation accuracy. Second, coloumb counting requires very accurate measurements of the input current I_{in}. Nevertheless, current sensing is not as straightforward as measuring voltage. Consequently, the accuracy of the current measurements is often limited, making even worse the error accumulation problem. Third, it is necessary to know exactly the initial value of SoC. This is because expression (1.1) does not take into account any physical properties of electrochemical cells, that indeed are collapsed into the value $\mathrm{SoC}[0]$ [51]. The problem is that the initial SoC can be retrieved only at the end of a full charging procedure. Therefore, coloumb counting is not able to cope with partial charge/discharge cycles. Finally, it is necessary to know the value of the capacity C_n, but this quantity changes with the aging of the cell, and its value is significantly dependent on the procedure used for evaluating it.

The second approach is more consistent, and it estimates SoC by means of the measured voltage. Indeed, it has been observed a unique correspondence between the amount of stored charge and the Open Circuit Voltage (OCV), with OCV being the steady-state terminal voltage of the cell. In particular, these methods are based on the definition of a lookup-table upon the OCV-SoC curve that determines the correspondence between each SoC value and the related steady state voltage. Thus, SoC is estimated by comparing the terminal voltage V_{out} with this lookup-table [47], [52]. The OCV-SoC curve can be retrieved by applying a sequence of pulse loads to the cell and waiting for equilibrium [53], or through machine learning methods such as genetic algorithms [54]. An example of OCV-SoC curve of a lithium electrochemical cell is shown in Figure 1.5.

The OCV method is easier and more effective than coloumb counting, but it is has several drawbacks as well. First, the voltage response of any electrochemical cell to its input current is affected by parasitic effects. Specifically, the internal resistance and charge redistribution phenomena cause instantaneous voltage drops and transient responses, respectively, that make the

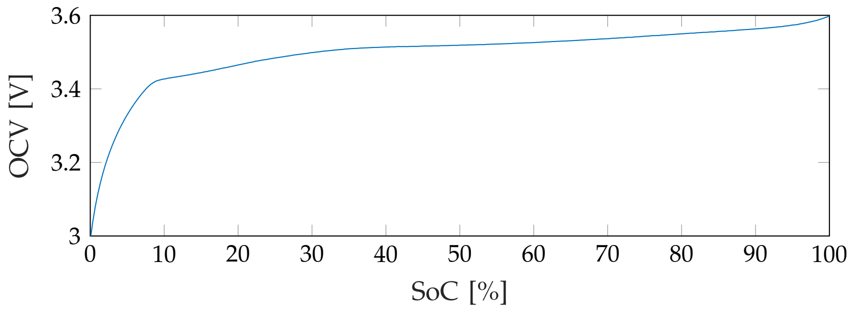


Figure 1.5: OCV-SoC curve of a lithium iron-phosphate cell.

measured voltage not anymore coincident with OCV. Consequently, there is a weak correlation between SoC and the terminal voltage, causing errors in the SoC estimation. Second, as shown in Figure 1.5, the OCV-SoC curve can have a very flat shape for a wide range of SoC values, resulting in a variation of few millivolts in front of high variations of SoC. Consequently, any small measurement error or measurement noise cause a significant erroneous SoC estimation. This phenomenon affects particularly the lithium technology, especially the iron-phosphate one.

Machine learning techniques are data-driven methods aiming at finding suitable correlations between the measured physical quantities and SoC. In particular, SoC estimation is formulated as a black-box function approximation problem, where voltage, current, and temperature are the inputs of the system, and SoC the output. Examples are neural networks [55], [56], Support Vector Machines (SVM) [57], or neuro-fuzzy systems [58]. The main drawback is that these approaches require a huge amount of historical data related to the cell usage in order to train the machine learning tools. Nevertheless, collect this amount of data is not a straightforward task, since it involves expensive testing equipment, such as accurate battery cyclers and climatic chambers, as well as long-term tests that can last tens of days [51] Moreover, these techniques may require further data processing and filtering for avoiding noisy estimations.

In accordance with the latest research results, the most promising SoC estimators are those based on state observer techniques such as Kalman Filters (KFs) [59]. The foundation of these methods is that SoC is a state variable of electrochemical cells, with expression (1.1) being the related state update equation. Therefore, provided there is a suitable model in which expression (1.1) appears in the state form equations, then any state observer is able to estimate SoC together with all the other state variables of the model [59].

Since electrochemical cells are nonlinear devices, the nonlinear versions of KFs need to be considered. Thus, Extended Kalman Filter (EKF) [60]–[62] and Unscented Kalman Filter (UKF) [63]–[65] have been widely used in the literature for performing SoC estimation. EKF copes with the nonlinearities through a piecewise linearization of the nonlinear system equations. Although it provides good estimation performances, the effectiveness of EKF decreases with deep nonlinear systems. Thus, UKF has been proposed in 1997 by Julier et.al [66] as a more suitable technique for dealing with nonlinear systems. To this aim, UKF uses a deterministic sampling, called *unscented transformation*, in order to obtain information about the current state variables and the related state-covariance matrices. This procedure avoids the evaluation of Jabobians, and it allows to obtain an accuracy up to the third order of the Taylor approximation, in front of the first order accuracy of EKF.

The main advantage of state observers is that they are real-time and self-correcting algorithms able to progressively correct a wrong estimation, converging to the actual SoC value. Nevertheless, they are model-based techniques requiring the availability of an accurate and effective model of the system under analysis for a likewise effective estimation of the system state vector. Therefore, the modeling and the related system identification of electrochemical cells is of utmost importance for performing accurate SoC estimations by means of Kalman filters.

In conclusion, a brief overview about the different SoC estimation methods is shown in Table 1.3.

State of Health

Likewise SoC, also SoH is not a directly measurable quantity, and therefore its estimation is based again on the indirect information of voltage, current, and temperature. Moreover, its real-time estimation is still considered an open question from the research community.

One of the first problems concerning SoH estimation is that there is not a unique and standardized definition. From a general point of view, SoH should be an index in the range $[0, 1]$ indicating with 1 a brand new cell, and with 0 the End Of Life (EOL) status. Thus, SoH can be defined in many different ways: some definitions are based on explicit expressions, whereas others provide an SoH index by analyzing the trend that suitable parameters show with respect to cell aging [67], [68]. In the latter case, the definition of SoH is dependent on the specific parameters taken into account, and therefore it cannot be generalized.

Table 1.3: Overview on the SoC estimation methods

Method	Description	Advantages	Drawbacks
Coloumb Counting	Discrete integral of the input current.	Simple and intuitive approach. Computationally efficient.	It requires to know the initial SoC value. It is affected by error accumulation. It does not consider any physical property of the cell. It requires accurate current measurements. It is not able to cope with partial charge/discharge cycles. It requires to know the actual capacity C_n.
OCV	Matching of the terminal voltage with the OCV-SoC lookup-table.	It takes into account physical properties of the cell. Computationally efficient.	Internal resistance and charge redistribution phenomena weak the correlation between voltage and SoC. Flat SoC-OCV curves make the SoC prediction more sensible to measurement noises and errors.
Machine Learning	SoC estimated with a black-box function approximation tool.	Computationally efficient.	It requires a huge amount of historical data for training the tool. Collect the training data involves expensive testing equipment and long-lasting tests. The relationship between voltage, temperature, current, and SoC is hidden. It may require further data processing and filtering.
State Observer	Use of nonlinear KFs for estimating SoC as a state variable of the system.	Real-time and self-correcting method. It can provide information about the estimation accuracy.	Can be computationally demanding. It requires an accurate model of electrochemical cells.

Two explicit expressions of SoH can be found in the literature. The first one is based on the evaluation of the actual cell capacity C_n and it is the most used and the most recognized from the research community. Indeed, even if the manufacturer declares a nominal capacity for a specific cell, C_n progressively fades with aging. Therefore, SoH is defined as follows:

$$\text{SoH}(t) = \frac{C_n(t)}{C_{n_{\text{SOL}}}} \tag{1.4}$$

where $C_n(t)$ is the actual capacity at time t, and $C_{n_{\text{SOL}}}$ is the capacity at the manufacturing moment, or Start of Life (SOL).

The second definition is based on the observation that the internal resistance of electrochemical cells shows an increasing trend with the number of cycles. However, there is not an officially recognized expression, and different authors have proposed their own:

$$\begin{aligned} \text{SoH}(t) &= \frac{R_{\text{EOL}} - R(t)}{R_{\text{EOL}} - R_{\text{SOL}}} && [69] \\ \text{SoH}(t) &= 1 - \frac{R(t) - R_{\text{SOL}}}{R_{\text{SOL}}} && [70] \\ \text{SoH}(t) &= \frac{R(t)}{R_{\text{SOL}}} && [71] \end{aligned} \tag{1.5}$$

where $R(t)$, R_{SOL}, and R_{EOL} are the internal resistance at time t, SOL, and EOL, respectively. It can be noticed that the third definition is not actually a number in the range $[0, 1]$, as well as the expression proposed in reference [69] requires to perform specific tests over aged cells in order to retrieve the R_{EOL} value.

Several methods have been proposed for performing SoH estimation. The most important ones are those based on capacity measurement [50], on internal resistance measurement [72], on Incremental Capacity Analysis (ICA) [73], [74], on Electrochemical Impedance Spectroscopy (EIS) [75], [76], on Kalman filters [70], and on machine learning techniques [77].

Both the capacity and the resistance measurement methods are based on offline tests aiming at estimating the value of $C_n(t)$ and $R(t)$ to be applied in expressions (1.4) and (1.5), respectively. In particular, the capacity test consists in performing a discharge cycle until voltage reaches the minimum allowed value, followed by a charge cycle run until voltage reaches the maximum allowed value. During this test, $C_n(t)$ is evaluated by applying the coloumb counting technique. The resistance test consists in applying a fast current pulse to the cell and in measuring the resulting voltage drop. Thus, let ΔI and ΔV be the current pulse and the related voltage drop, respectively, the estimation of the internal resistance is evaluated as follows:

$$R = \frac{\Delta V}{\Delta I}. \tag{1.6}$$

The main drawback of both the previous approaches is that they allow only an offline SoH estimation. Moreover, it must be noticed that it is necessary to disassemble the entire BESS and to perform specific tests over the single cells in order to get the SoH of each of them.

ICA methods are based on the estimation of the OCV-SoC curve that implicitly encloses information about SoH. In particular, these methods consist in determining the Incremental Capacity (IC) curve from the OCV-SoC one by evaluating the derivative of SoC with respect to OCV. The ICA method is based on the observation that the peaks appearing in the IC curve become smoother and progressively move towards higher value of OCV with the aging of the cell [73]. Therefore, it is possible to define an SoH index by matching the shape of the IC peaks with the health of the cell.

Similarly to the capacity and the internal resistance approaches, also ICA can be performed only offline, it needs to disassemble the battery pack, as well as it requires to perform specific tests in order to retrieve the OCV-SoC curve of the cells. Furthermore, it is necessary to define a suitable method for getting an SoH index from the IC curve.

EIS methods are some of the most promising approaches for performing online SoH estimations. They are based on the EIS analysis that involves the measurement of the terminal impedance of the cell at different frequency values. In particular, the impedance $Z(\omega)$ is evaluated by injecting a small sinusoidal signal at different frequencies, and by measuring the consequent absolute and phase-shift voltage responses. It has been observed that the resulting Nyquist diagram of $Z(\omega)$ changes with the aging of the cell, so that it is possible to define an SoH index similarly to the ICA method [75].

The EIS approach has the advantage of allowing an online SoH estimation. Nevertheless, the currently available EIS hardware is very expensive and voluminous. Therefore, it is not possible to provide a multi-cell BESS with the sufficient amount of EIS devices needed for monitoring all the cells. Moreover, as well as for the ICA approach, also EIS needs the definition of a suitable method for retrieving an SoH index from the EIS curve.

Concerning Kalman filtering approaches, their application to SoH estimation is very similar to that of SoC. Indeed, also in this case it is required the availability of a suitable cell model in order to make Kalman filters track the evolution of the cell capacity C_n and/or of the internal resistance R. Often, dual Kalman filtering techniques can be used in order to estimate both SoC and SoH simultaneously [70].

Also Kalman filter techniques have the advantage of an online and real-time SoH estimation. However, besides the necessity of a suitable model, the main drawback is that it is mandatory a very accurate SoC estimation in order to permit a likewise accurate tracking of the C_n evolution. To make things worse, also the accuracy of the SoC estimator depends on that of SoH, since expression (1.3) use the value of C_n for evaluating SoC. Therefore, it can happen that the Kalman filter estimating SoH tries to correct a wrong SoC estimation, and similarly the SoC estimator tries to correct the erroneous C_n estimation, resulting in an unstable and oscillating monitoring system.

Finally, machine learning approaches aim at finding suitable relationships between the measurements of voltage, current, and temperature with the progressive capacity fade of the cell. Therefore, SoH estimation is formulated as a function approximation problem, similarly to the SoC estimation task. Alternatively, machine learning methods, such as neural networks or SVMs, have been used in the ICA and EIS approaches for retrieving an SoH index from the IC and $Z(\omega)$ curves, respectively [74], [76].

Likewise the SoC estimation, also in this case the main drawback comes from the necessity of a huge amount of historic data in order to allow a suitable and effective training of the machine learning tools.

In conclusion, a brief overview about the different SoH estimation methods is shown in Table 1.4.

Table 1.4: Overview on the SoH estimation methods

Method	Description	Advantages	Drawbacks
Capacity Test	Direct measurement of C_n by performing a controlled charge/discharge test.	Simple and intuitive approach.	Offline procedure. It requires expensive testing equipment. It is necessary to disassemble the BESS.
Internal Resistance Measurement	Direct measurement of the internal resistance by applying a short current pulse to the cell.	Simple and intuitive approach.	Offline procedure. It requires expensive testing equipment. It is necessary to disassemble the BESS.
ICA	Definition of a suitable procedure for retrieving an SoH index from the analysis of the IC curve.	It takes into account physical properties of the cell.	Offline procedure. It requires to retrieve the OCV-SoC curve for evaluating the IC. It may need expensive testing equipment. It is necessary to disassemble the BESS. It is necessary to define a suitable procedure for retrieving the SoH index from the IC curve.
EIS	Definition of a suitable procedure for retrieving an SoH index from the analysis the Nyquist diagram of $Z(\omega)$.	Online procedure. It takes into account physical properties of the cell.	EIS hardware is very expensive and voluminous. It cannot be applied to numerous electrochemical cells. It is necessary to define a suitable procedure for retrieving the SoH index from the EIS curve.
State Observer	Use of a dual nonlinear KF for estimating both SoC and C_n and/or the internal resistance.	Online procedure. Real-time and self-correcting method. It can provide information about the estimation accuracy.	Can be computationally demanding. It requires an accurate model of electrochemical cells. It needs very accurate SoC estimations. It may happen an undesired conflict between the KF estimating SoC and that estimating C_n and/or the internal resistance.
Machine Learning	SoH estimated with a black-box function approximation tool.	Online procedure. Computationally efficient. Machine learning methods may be used in the ICA or EIS techniques.	It requires a huge amount of historical data for training the tool. Collect the training data involves expensive testing equipment and long-lasting tests. The relationship between voltage, temperature, current and SoH is hidden. It may require further data processing and filtering.

Balancing

Cells balancing aims at ensuring that each cell of the BESS works in an operating point as much as possible similar to that of the others. To this aim, two main approaches can be found in the literature, namely active and passive balancing [78]. In the active approach, a programmable circuit composed of capacitors or inductors is used for discharging the most charged cells in the most discharged ones. More precisely, the capacitors/inductors network is used as an electronic bridge aiming at transferring charge between the cells.

Thus, the active balancing is controlled by means of a complex and expensive switching system. Conversely, the passive balancing method consists in discharging the most charged cells on suitable shunt resistors. Therefore, each cell of the battery pack is coupled with a resistor, which activation is controlled by a dedicated MOSFET switch.

The two methods have opposite properties in terms of complexity, cost, and energetic efficiency. The active balancing approach has a widely more complex and more expensive control circuit, but it is able to achieve negligible energy losses during cells balancing. On the opposite, the passive method is way more simple and cheaper, but it implies a systematic loss of energy every time a shunt resistor is activated.

Because of its lower cost and higher simplicity, the passive approach is usually preferred as balancing method. Indeed, most of the slave boards shown in the previous paragraphs are natively equipped with the necessary hardware performing passive balancing. In particular, it is necessary to develop a balancing algorithm aiming at evaluating which is the best configuration of the shunt resistors that progressively brings all the cells working as much as possible in the same operating point. To this aim, the common policy used both in the literature [79], [80] and in the industrial area [81] is based on the leveling of the cells voltage at working time.

1.3.2 Modeling Techniques

In the previous section it has been briefly discussed as Kalman filter techniques show very promising performances in the SoC estimation task, as well as they can be used effectively also for performing SoH estimation. Nevertheless, it has been discussed that state observers need the availability of accurate models for achieving effective results as well. Therefore, great effort of the research community is focusing on the development of effective and accurate models of electrochemical cells.

The most accurate models are the electrochemical ones [82]–[85], based on modeling the chemical processes occurring at the molecular level. Consequently, these models achieve both a complete insight on the cell physics, as well as they model accurately the nonlinearities of the cell. Despite that, electrochemical models are typically highly computational demanding, and often they cannot be used to implement real-time estimators. Moreover, the related system identification is very stiff, being based on specific and long-lasting offline tests. In addition, often it relies on the knowledge of property parameters that are not made available by the manufacturer [60], [86].

A good trade-off between accuracy and complexity is achieved by Equivalent Circuit Models (ECMs) [51], [60], [86]–[88], in which a connection of bipolar lumped elements emulate the electrical behavior of the cell. ECMs have the advantage of keeping a sufficient insight on the cell physics together with a simple and computational efficient architecture. However, also for ECMs system identification is stiff and time consuming, because it requires long-lasting and specific charging and discharging tests to be performed offline in order to complete the task [51].

Machine learning techniques are a valid alternative to the previous approaches, allowing to deal with the strong nonlinear and dynamical behavior of electrochemical cells in a more effective way. Indeed, they are able to achieve great estimation accuracy with a more flexible system identification procedure due to their learning capabilities. Thus, several machine learning approaches have been proposed in the literature addressing the modeling of electrochemical cells. In [89] and [90] the authors have proposed a Radial Basis Function Neural Network for estimating the terminal voltage of a cell receiving as input the present current, the present SoC and a sliding window of the previous output voltages. More recently, an Extreme Learning Machine and a Wavelet Neural Network performing the same task have been proposed in [65] and in [91], respectively. Other approaches based on neuro-fuzzy networks [92] and SVMs [93] estimate the voltage response of the cell avoiding the windowing of the output voltage. All of these approaches perform system identification following a black box scheme and addressing the estimation of the global input-output relationship only. Thus, they do not provide any insight on the cell physics. Furthermore, most machine learning models use a nonlinear autoregressive approach by introducing a sliding window of the previous output voltage in order to model the internal states of the system. However, due to the very slow dynamics typical of any electrochemical cell, these models could easily overfit the delayed output, converging to the trivial prediction model in which the output voltage is almost equal to its previous sample. This unwanted behavior could compromise the dependence of the model output on SoC, jeopardizing an effective use of the model in Kalman estimators.

From the above discussion it is possible to split the modeling techniques in two main categories, namely the white box and the black box ones. White box approaches are based on the definition of modeling equations that explicitly expose information about the physics of the system. Conversely, black box methods aim at building a universal approximator of the overall input-output relationship, without providing any physical interpretation of the model. Therefore, ECMs and electrochemical models belong to the white box technique, whereas the machine learning models belong to the black box one.

White box and black box techniques have opposite properties. Typically, white box models offer a complete insight on the physics of electrochemical cells that is helpful for performing accurate SoC estimations. Nevertheless, they are or too much computational demanding for being used in the SoC estimation task, or system identification is a stiff and time consuming task. Conversely, black box models are able to easily catch the nonlinearities of the cell, but unfortunately they do not provide any insight on their physics.

It is interesting to note that none of the models proposed in the literature are able to fulfill the three modeling requirements discussed in Section 1.2. For this reason this Ph.D. thesis aims at investigating novel modeling solutions with the objective of pursuing all the three requirements in order to improve the model performances in the SoC estimation task.

Chapter 2

Machine Learning Techniques

2.1 Neural Networks

Artificial Neural Networks (ANNs) are powerful processing tools enclosing the ability of learning from experience. From a general point of view ANNs are a data-driven black box technique aiming at learning and modeling the input-output relationship of a given process from the knowledge of a set of input-output measurements only. Thus, ANNs have been applied with very promising performances in several black box modeling tasks, involving classification [94]–[96], function approximation [97]–[99], system identification [100]–[103], and image processing [104], [105].

Likewise other machine learning algorithms, also neural networks are natural inspired techniques. Specifically, ANNs aim at emulating the functionality and the learning capabilities of the animal brain. Thus, they are organized as a network of atomic computational units called *artificial neurons*, each of them performing a very simple and basic processing of its inputs and propagating the resulting output to the other neurons.

The artificial neuron has been firstly theorized by McCulloch and Pitts in reference [106], and a first real implementation of an ANN has been proposed by Rosenblatt in reference [107] with the perceptron. A graphical representation of the artificial neuron is shown in Figure 2.1.

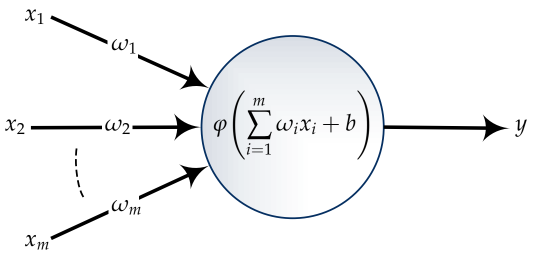


Figure 2.1: Basic functionality of the artificial neuron.

Similarly to the brain cells, an artificial neuron is composed of two main elements: nucleus and axon. The nucleus is the actual processing unit and it is in charge of performing a nonlinear transformation of the weighted sum of the inputs. The basic processing of the neuron is summarized as follows:

$$y = \varphi\left(\sum_{i=1}^{m} x_i \omega_i + b\right) \tag{2.1}$$

where x_i is the i^{th} input, ω_i is the related weight, b is a bias term, and $\varphi(.)$ is a nonlinear transformation called *activation function*. The axon is merely the connection from the neuron to the following one and typically the weighting operation is attributed to it. In general, the activation $\varphi(.)$ can be any kind of nonlinear function. For example, the neuron theorized by McCulloch and Pitts was based on the Heaviside step function and for this reason it is commonly known as Linear Threshold Unit (LTU). Nevertheless, LTU allows only a binary processing. Thus, smoother and continuous functions such as *hyperbolic tangent* or *sigmoid* have been preferred in order to ensure a continuous input-output mapping.

Feed-forward neural networks are the basic architecture of ANNs and the most suitable example is the Multi-Layer Perceptron (MLP) shown in Figure 2.2.

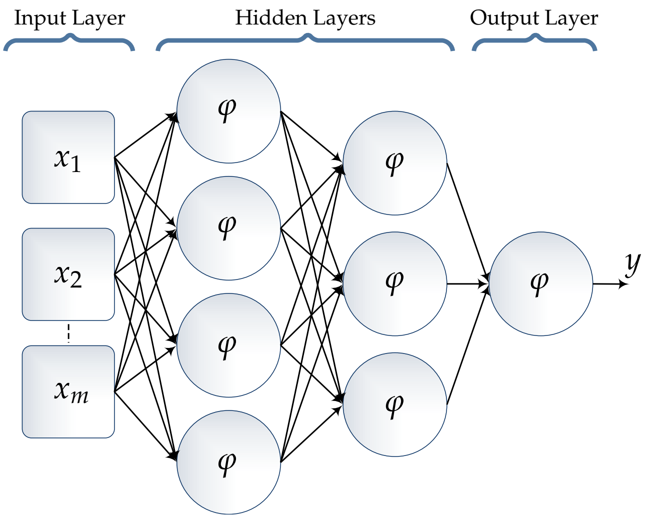


Figure 2.2: Architecture of a feed-forward MLP.

Feed-forward MLPs are organized in a stack of layers in which the output of each neuron is connected to the immediate following layer only. Thus, the inputs are processed and propagated in only one direction moving layer by layer to the output neuron. This means that feed-forward neural networks are pure combinatory and memoryless processing tools in which the current output depends only on the current input. In particular, MLPs are characterized by fully connected layers, meaning that each neuron receives as input the output of all the neurons composing the previous layer and it propagates its output to all the neurons of the following layer. The MLP architecture is organized in three groups: the input layer, one or more hidden layers, and the output layer. In particular, the input layer is not composed of real neurons, but it aims only at feeding the first hidden layer with the overall inputs of the network.

It has been proven with the *universal approximation theorem* [108]–[110] that a simple MLP configured with one hidden layer composed of n neurons having arbitrary activation $\varphi(.)$ can approximate continuous functions on compact subsets of $\mathbb{R}^m$.

Theorem 2.1.1 (Universal Approximation Theorem). *Let $\varphi(.)$ be an arbitrary activation function. Let $\boldsymbol{x} \in X \subseteq \mathbb{R}^m$ and X is compact. The space of continuous functions on X is denoted by $C(X)$. Then, $\forall f \in C(X), \forall \epsilon > 0 : \exists\, n \in \mathbb{N}, \omega_{ij} \in \mathbb{R}$, $b_i \in \mathbb{R}$, $\alpha_i \in \mathbb{R}$, $i \in \{1, \dots, n\}$, $j \in \{1, \dots, m\}$:*

$$\mathfrak{f}(\boldsymbol{x}) = \sum_{i=1}^{n} \alpha_i \varphi \left(\sum_{j=1}^{m} \omega_{ij} x_j + b_i \right)$$

is a universal approximation of $f(.)$ such that:

$$\| f(\boldsymbol{x}) - \mathfrak{f}(\boldsymbol{x}) \| < \epsilon.$$

Moreover, Hornik has showed in reference [98] that the universal approximation property of MLPs comes from the multi-layer architecture itself rather than on the specific activation function of the neurons. Indeed, what an MLP architecture does is to project the input vector $\boldsymbol{x} = \{x_1, x_2, \dots, x_m\}$ to a higher dimensional space where the indistinguishable properties of the original data are made clearer and separable to successfully approximate functions, classify objects, or identify the main properties of systems. Moreover, the multi-layer architecture reiterates this projection layer by layer, furtherly increasing the data abstraction in higher order dimensional spaces. In the end, the output layer is in charge of recollecting and synthesizing the abstract information in order to retrieve the useful knowledge.

The most common used activation functions are *linear*, *hyperbolic tangent*, *sigmoid*, and *rectifier linear unit* (ReLU). The definition and the graphical representation of each of them is shown in Figure 2.3.

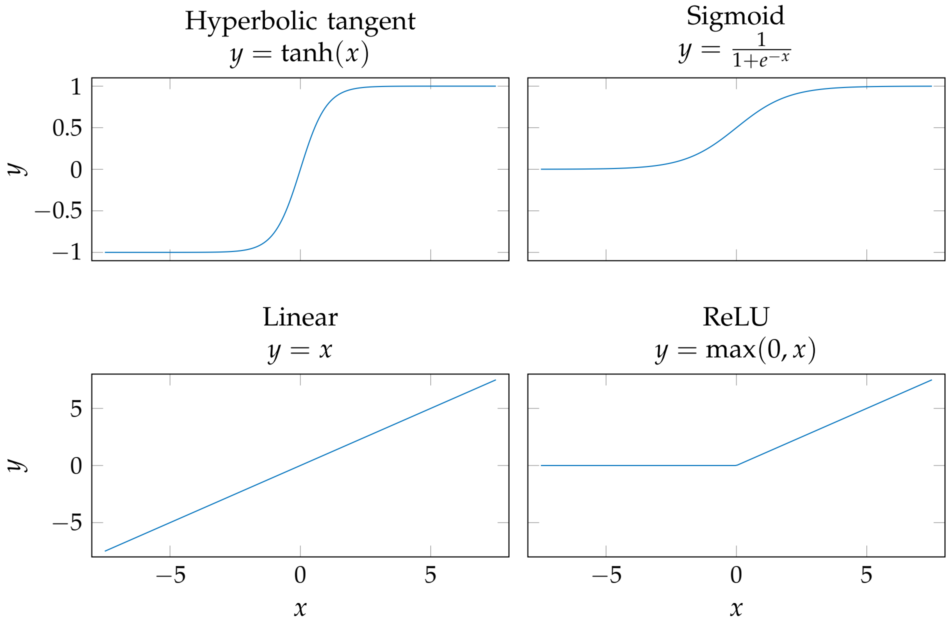


Figure 2.3: Most common activation functions.

Several alternatives to the above mentioned activation functions have been proposed in the literature, such as *sinusoidal* [111], *exponential linear units* [112], or *softsign* [113]. Furthermore, the use of specific activation functions is at the basis of the definition of different and distinct neural networks topologies.

Radial Basis Function Neural Networks (RBF-NN) [114] are feed-forward architectures having only one hidden layer and one output layer. As the name suggest, the activation of the hidden neurons is a radial function. Moreover, the inputs are fed to the hidden layer without any weighting of them. In particular, the most common radial activation is the gaussian function defined as follows:

$$y = e^{-\beta \|\mathbf{x}-\boldsymbol{\mu}\|^2} \tag{2.2}$$

where $\|.\|$ is a distance operator, typically the Euclidean one, β is the shape parameter, and μ is the center of the radial function. In RBF-NNs the learning procedure must train not only the output weights, but also the parameters β and μ of each radial neuron.

Wavelet Neural Networks (WNNs) aim at importing the wavelet analysis theory to neural networks. WNNs have a similar architecture to RBF-NNs, except for the use of mother wavelet functions $\Phi(.)$ as activation of the hidden layer [115], [116]. Specifically, each *wavelon* performs the following elaboration:

$$y = \prod_{j=1}^{m} \Phi\left(\frac{x_j - \delta_j}{\gamma_j}\right) \tag{2.3}$$

where δ_j and γ_j are the *translation* and *dilation* parameters related to the wavelet theory, and j refers to the j^{th} input. Also in this case the learning procedure shall determine not only the output weights, but also the value of the parameters δ_j and γ_j for each wavelon.

Functional Link Neural Networks (FLNNs) [102], [117], [118] differ from MLPs, RBF-NNs, and WNNs for the fact that neurons do not share the same nonlinear transformation. Indeed, the aim of FLNNs is to model the process at hand by considering the superimposition of different functions selected from an orthogonal basis. FLNNs have the same architecture of both RBF-NNs and WNNs, but each neuron of the hidden layer evaluates a specific function of the selected basis. Common choices for the functional reservoir are the standard polynomial basis, Chebyshev polynomials, trigonometric polynomials, or Bernstein polynomials. Furthermore, it is possible to mix different nonlinear representations in order to furtherly increase the nonlinear abstraction of the inputs. Finally, the output neuron mixes all the nonlinear transformations in order to match the output of the process to be modeled.

Besides the memoryless feed-forward architecture, neural networks can also have a recurrent design in order to equip the model with memory. Indeed, Recurrent Neural Networks (RNNs) involve the introduction of feedback connections resulting in the presence of delay units and circular loops. Thus, the output of any RNN depends not only on the current input of the network, but also on the values memorized in each of the delay units.

A typical example of RNN is the Elman network shown in Figure 2.4, in which the output of each hidden neuron is fed back as an input of the same layer. Therefore, each neuron performs the following elaboration:

$$y_i = \varphi\left(\sum_{j=1}^{m} \omega_j^x x_j + \sum_{\ell=1}^{n} \omega_\ell^h h_\ell + b_i\right) \tag{2.4}$$

where m and n are the number of inputs and the number of neurons, respectively, h_ℓ is the state variable related to the ℓ^{th} hidden neuron, ω_j^x and ω_ℓ^h are the weights related to the j^{th} input and the ℓ^{th} hidden state, respectively, and b_i is the bias term.

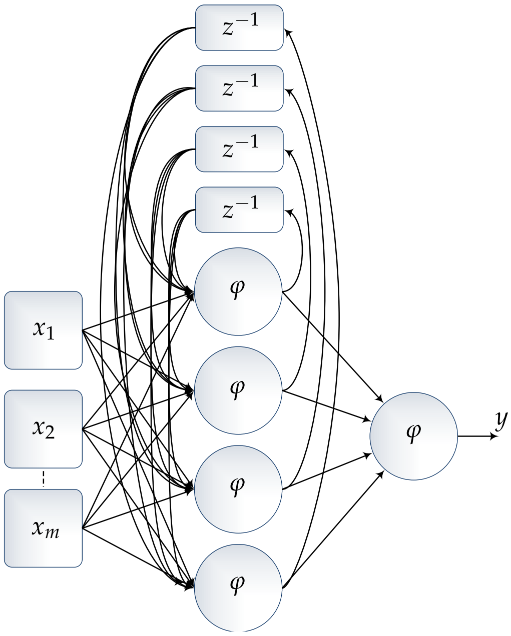


Figure 2.4: Architecture of an Elman RNN.

More complex recurrent architectures have been introduced in order to solve some issues of Elman RNNs concerning the loss of information that is far away in the past [119]. These recurrent networks are based on the introduction of more sophisticated neurons, such as the *Long Short Term Memory* (LSTM) unit [120], [121] and the *Gated Recurrent Unit* (GRU) [122]. In particular, both LSTM and GRU units have an architecture that allows to propagate the information through time, and thus to efficiently learn long-range dependencies.

As explained in the above, ANNs are able to learn the main properties of a given process from a set of input-output samples. To this aim, the most used learning algorithm is the Gradient Descent Backpropagation (GDBP). As the name suggests, GDBP is based on the standard gradient descent optimization technique. It consists in the re-iteration of two main phases, namely the forward and the backward ones. During the forward step, the input samples belonging to the training set are fed to the network and the related outputs are evaluated. During the backward phase, the predicted outputs are compared with the expected ones and the resulting errors are fed back to the network in order to update all the weights. In particular, the latter step is performed by minimizing a suitable error function by means of the gradient descent algorithm. Typically, a differentiable function must be considered, and examples are *mean squared error* or *root mean squared error* that are commonly used in function approximation problems, or *categorical crossentropy* that is used for multi-class classification.

Let $\mathfrak{E}(.)$ be the error function at hand and ω_{ij} be a generic weight of the network. The gradient descent algorithm iteratively updates the value of ω_{ij} by moving towards the direction of the anti-gradient of $\mathfrak{E}(.)$. Thus, the update rule is defined as follows:

$$\omega_{ij}[k] = \omega_{ij}[k-1] - \lambda \frac{\partial \mathfrak{E}}{\partial \omega_{ij}} \tag{2.5}$$

where k is the current iteration and λ is the *learning rate* that defines the step width towards the anti-gradient direction. In particular, the backpropagation algorithm makes use of the derivative chain rule in order to propagate the gradient back to each of the network weights. Thus, let consider as example the application of the GDBP algorithm to the weight ω_{11} of the simple network shown in Figure 2.5.

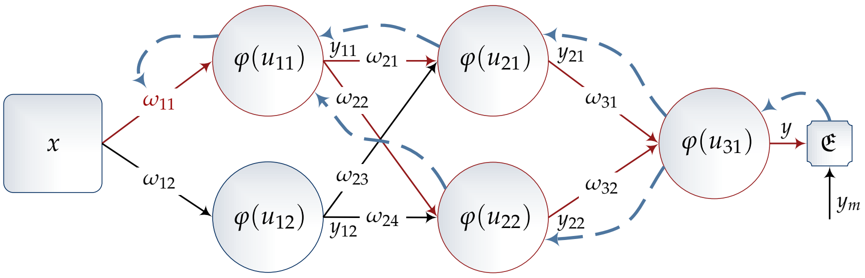


Figure 2.5: Graphical representation of the backpropagation technique to ω_{11}. The backward path of the derivative chain rule is highlighted in the figure.

Considering that $y = \varphi(u_{31})$ and that $u_{31} = \omega_{31}y_{21} + \omega_{32}y_{22} + b_{31}$, it is possible to evaluate the partial derivative of $\mathfrak{E}(.)$ with respect to ω_{11} as follows:

$$\begin{aligned}
\frac{\partial \mathfrak{E}}{\partial \omega_{11}} &= \frac{\partial \mathfrak{E}}{\partial y}\frac{\partial y}{\partial \omega_{11}} \\
&= \frac{\partial \mathfrak{E}}{\partial y}\frac{\partial \varphi(u_{31})}{\partial \omega_{11}} \\
&= \frac{\partial \mathfrak{E}}{\partial y}\frac{\partial \varphi(u_{31})}{\partial u_{31}}\frac{\partial u_{31}}{\partial \omega_{11}} \\
&= \frac{\partial \mathfrak{E}}{\partial y}\frac{\partial \varphi(u_{31})}{\partial u_{31}}\frac{\partial(\omega_{31}y_{21} + \omega_{32}y_{22} + b_{31})}{\partial \omega_{11}} \\
&= \frac{\partial \mathfrak{E}}{\partial y}\frac{\partial \varphi(u_{31})}{\partial u_{31}}\left(\omega_{31}\frac{\partial y_{21}}{\partial \omega_{11}} + \omega_{32}\frac{\partial y_{22}}{\partial \omega_{11}}\right)
\end{aligned} \tag{2.6}$$

where $\partial b_{31}/\partial \omega_{11} = 0$ because the bias term b_{31} is not dependent on ω_{11}. From the above expression, it is possible to apply again the derivative chain rule to $\partial y_{21}/\partial \omega_{11}$ and $\partial y_{22}/\partial \omega_{11}$. Specifically, $\partial y_{21}/\partial \omega_{11}$ is evaluated as follows:

$$\begin{aligned}\frac{\partial y_{21}}{\partial \omega_{11}} &= \frac{\partial \varphi(u_{21})}{\partial \omega_{11}} \\ &= \frac{\partial \varphi(u_{21})}{\partial u_{21}} \frac{\partial u_{21}}{\partial \omega_{11}} \\ &= \frac{\partial \varphi(u_{21})}{\partial u_{21}} \frac{\partial(\omega_{21} y_{11} + \omega_{23} y_{12} + b_{21})}{\partial \omega_{11}} \\ &= \frac{\partial \varphi(u_{21})}{\partial u_{21}} \omega_{21} \frac{\partial y_{11}}{\partial \omega_{11}}\end{aligned} \tag{2.7}$$

where $u_{21} = \omega_{21} y_{11} + \omega_{23} y_{12} + b_{21}$, whereas $\partial y_{12}/\partial \omega_{11}$ and $\partial b_{21}/\partial \omega_{11}$ are both zero because y_{12} and b_{21} are not dependent on ω_{11}. Similarly, $\partial y_{22}/\partial \omega_{11}$ is evaluated as follows:

$$\frac{\partial y_{22}}{\partial \omega_{11}} = \frac{\partial \varphi(u_{22})}{\partial u_{22}} \omega_{22} \frac{\partial y_{11}}{\partial \omega_{11}}. \tag{2.8}$$

Both expressions (2.7) and (2.8) require the evaluation of $\partial y_{11}/\partial \omega_{11}$, that can be performed by applying again the chain rule as follows:

$$\begin{aligned}\frac{\partial y_{11}}{\partial \omega_{11}} &= \frac{\partial \varphi(u_{11})}{\partial \omega_{11}} \\ &= \frac{\partial \varphi(u_{11})}{\partial u_{11}} \frac{\partial u_{11}}{\partial \omega_{11}} \\ &= \frac{\partial \varphi(u_{11})}{\partial u_{11}} \frac{\partial(\omega_{11} x + b_{11})}{\partial \omega_{11}} \\ &= \frac{\partial \varphi(u_{11})}{\partial u_{11}} x.\end{aligned} \tag{2.9}$$

where $u_{11} = \omega_{11} x + b_{11}$ and again $\partial b_{11}/\partial \omega_{11} = 0$ because b_{11} is not dependent on ω_{11}.

In conclusion, the final expression of $\partial \mathfrak{E}/\partial \omega_{11}$ is evaluated by mixing expressions (2.9) with (2.7), (2.8), and (2.6) as follows:

$$\frac{\partial \mathfrak{E}}{\partial \omega_{11}} = \frac{\partial \mathfrak{E}}{\partial y} \frac{\partial \varphi(u_{31})}{\partial u_{31}} \left(\omega_{31} \frac{\partial \varphi(u_{21})}{\partial u_{21}} \omega_{21} + \omega_{32} \frac{\partial \varphi(u_{22})}{\partial u_{22}} \omega_{22} \right) \frac{\partial \varphi(u_{11})}{\partial u_{11}} x. \tag{2.10}$$

The GDBP learning algorithm consists in applying the above described procedure to every weight ω_{ij} and bias b_{ij} of the network for a certain amount of iterations, also known as training *epochs*, or until a certain stop condition is verified.

Besides GDBP, other learning procedures can be considered. For example, evolutionary algorithms have been successfully applied for training different kind of ANNs [117], [123], [124]. Moreover, if the output activation is linear, then RBF-NNs, WNNs, and FLNNs can be trained in a faster way because of the static processing of their hidden layer. Concerning both RBF-NNs and WNNs, it is possible to conceive a two step-learning procedure. During the first step, a suitable clustering algorithm is used for determining the value of the parameters β and μ for RBF-NNs, δ and γ for WNNs. During the second phase, a linear least square is used for training the output weights. Conversely, only the linear least square phase is necessary for training FLNNs because the hidden layer does not have any parameter to be trained. In general, this faster learning approach is possible for any reservoir computing network such as Extreme Learning Machines (ELMs) [125], Random Vector Functional Links (RVFL) [126], and Echo State Networks (ESNs) [127].

2.2 Evolutionary Algorithms

Evolutionary Algorithms (EAs) are a family of meta-heuristics used for solving optimization problems where standard derivative based techniques are ineffective. In order to introduce EAs, let consider a generic optimization problem consisting in minimizing a suitable objective function $\mathfrak{F}(.)$ with respect to the set of parameters $\boldsymbol{\theta} = \{\theta_1, \dots, \theta_n\}$, and subject to a certain set of constraints. The optimization problem can be formalized as follows:

$$\begin{aligned} \boldsymbol{\theta}_{\text{opt}} = & \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\ \mathfrak{F}(\boldsymbol{\theta}) \\ \text{s.t.} \quad & g_i(\boldsymbol{\theta}) \le 0, \quad i = 1, \dots, m \\ & h_j(\boldsymbol{\theta}) = 0, \quad j = 1, \dots, p \end{aligned} \tag{2.11}$$

where $g_i(\boldsymbol{\theta}) \le 0$ and $h_j(\boldsymbol{\theta}) = 0$ are the inequality and equality constraints, respectively. Note that this formulation is valid also for maximization problems, since they are actually coincident with minimizing $-\mathfrak{F}(\boldsymbol{\theta})$.

If certain conditions are verified, the optimization problem (2.11) can be solved in a closed form or at least with an arbitrary precision. In particular, if the objective function $\mathfrak{F}(.)$ is convex and the feasibility region defined by the constraints is convex as well, then the solution is unique and it can be evaluated in a closed form by means of the Lagrange multipliers and the Karush–Kuhn–Tucker methods [128]–[130]. Alternatively, it is possible to use gradient descent techniques that will find a solution arbitrary close to the global optimum because of the convexity of the problem.

Both the Lagrange multipliers and the gradient descent techniques are derivative based optimization methods. Indeed, the former is based on zeroing the gradient of $\mathfrak{F}(.)$, whereas the latter is an iterative algorithm requiring the evaluation of $\nabla_{\boldsymbol{\theta}}\mathfrak{F}(\boldsymbol{\theta})$ at each step. Nevertheless, in some applications it is not possible to evaluate the derivative of $\mathfrak{F}(.)$. First, the parameters to be optimized may not be real numbers, but structured data such as text, audio, images, or graphs, upon which it is not possible to define a differentiation operator. Second, the function $\mathfrak{F}(.)$ may not be known in closed form, so that it is impossible to evaluate $\nabla_{\boldsymbol{\theta}}\mathfrak{F}(\boldsymbol{\theta})$. Moreover, if the problem is not convex, then the solution is not unique anymore and any derivative based approach can be trapped in local optimum solutions.

EAs are general purpose derivative-free methods aiming at solving optimization problems affected by any of the above discussed issues. The main concept of EAs is to emulate natural phenomena that implicitly involve an optimization process, typically the survival of the species, or food supply [131]. In general, EAs are population-based techniques in which a population of individuals evolves looking for the best solution of the problem at hand. More precisely, each individual encodes one possible solution of the optimization problem and it interacts with the other individuals in order to improve its suitability at solving the problem. To this aim, it is necessary to evaluate how much a certain individual is eligible as a solution, or in other words how much it "fits" the environment in which the population lives. This evaluation is performed by means of a suitable function $\mathfrak{F}(.)$, called *fitness* function in the EA context, that includes the objective function, as well as the optimization constraints.

Several EAs algorithms have been proposed in the literature. Examples are Genetic Algorithm (GA) [132], Particle Swarm Optimization (PSO) [133], [134], Ant Colony Optimization (ACO) [135], [136], Artificial Bee Colony (ABC) [137], and many others. Nevertheless, the most used and pioneering ones are GA and PSO.

GAs aim at applying the evolution theory of Darwin [138] for solving optimization problems of the form (2.11). It consists in evolving a population of M individuals by applying the basic concepts of elitism, reproduction, and mutation. In particular, as well as the fitness of a living being to a certain environment is determined by its genetic code, the eligibility of a GA individual to be a solution of (2.11) is determined by the set of parameters $\boldsymbol{\theta}_i$ associated to it. From a different perspective, $\boldsymbol{\theta}_i$ is the genetic code characterizing the i^{th} individual, whereas its fitness $\mathfrak{f}_i$ is determined by the function $\mathfrak{F}(.)$.

Therefore, the entire population is modeled with a list of candidate solutions having genetic code $\boldsymbol{\theta}_i$ and related fitness value $\mathfrak{f}_i$, as shown in Table 2.1.

Table 2.1: Represenation of the population in the GA algorithm

Individual ID	**Genetic Code**	**Fitness**
1	$\boldsymbol{\theta}_1$	$\mathfrak{f}_1$
2	$\boldsymbol{\theta}_2$	$\mathfrak{f}_2$
$\vdots$	$\vdots$	$\vdots$
M	$\boldsymbol{\theta}_M$	$\mathfrak{f}_M$

GA starts from the first generation by initializing a generic population of random individuals. Successively, each of them evolves generation after generation by applying three main operators, namely *elitism*, *crossover*, and *mutation*.

The elitism operator encloses the natural selection theory, so that individuals best adapted to the environment are more likely to survive and reproduce. Therefore, the E individuals having the best fitness survive to the next generation without modifications.

The crossover operator models the reproduction process. It consists in selecting two individuals and mixing their genetic codes for generating two new offsprings. Thus, given the parent individuals $\boldsymbol{\theta}_i = \{\theta_{i1}, \dots, \theta_{in}\}$, and $\boldsymbol{\theta}_j = \{\theta_{j1}, \dots, \theta_{jn}\}$, the two offsprings can be generated by swapping a random set of the parents' genes as follows:

$$\begin{array}{l} \boldsymbol{\theta}_i = \{\theta_{i1}, \;\; \theta_{i2}, \Big| \, \theta_{i3}, \;\; \theta_{i4}, \;\; \theta_{i5}, \Big| \, \theta_{i6}\} \\ \qquad\quad \updownarrow \qquad\qquad\qquad\qquad\quad \updownarrow \\ \boldsymbol{\theta}_j = \{\theta_{j1}, \;\; \theta_{j2}, \Big| \, \theta_{j3}, \;\; \theta_{j4}, \;\; \theta_{j5}, \Big| \, \theta_{j6}\} \end{array} \Rightarrow \begin{array}{l} \boldsymbol{\vartheta}_i = \{\theta_{j1}, \theta_{j2}, \theta_{i3}, \theta_{i4}, \theta_{i5}, \theta_{j6}\} \\ \boldsymbol{\vartheta}_j = \{\theta_{i1}, \theta_{i2}, \theta_{j3}, \theta_{j4}, \theta_{j5}, \theta_{i6}\} \end{array} \tag{2.12}$$

where an example with $n = 6$ genes has been considered for simplicity.

As the name suggest, the mutation operator aims at emulating the effect of a genetic mutation. Typically, it is performed by selecting one individual and by applying a random modification of a random set of its genes. In case of real numbers, it can be implemented through the summation of a random noise, otherwise it is necessary to define a specific and suitable mutation operator accordingly with the semantic of the gene to be muted.

It must be noticed that both the crossover and the mutation operators imply the selection of the individuals to which they must be applied. In particular, this selection process plays a crucial role in the GA performances and therefore it must be designed carefully. The basic idea of the natural selection is that the best adapted individuals have the most interesting genetic properties. Thus, the selection operator must be more sensible to them. Nevertheless, applying the mutation and the crossover operators to the best individuals only will compromise the genetic variability of the population, resulting in a global lack of adaptation to the environment. Therefore, it is necessary to promote the selection of the best individuals, without excluding the worst ones from the selection process. Two of the most used algorithms are *K-tournament* and *roulette wheel*. In K-tournament, K individuals are chosen randomly from the entire population independently of their fitness. Successively, the best individual among those K is selected. In roulette wheel, each individual has a probability to be selected that is directly or inversely proportional to its fitness depending if GA is solving a maximization or a minimization problem, respectively. Thus, both in K-tournament and roulette wheel the best individuals have the higher probability to be selected, but the worst ones are not excluded a priori.

Summarizing, the pseudo code of GA is shown in Algorithm 1, where M is the number of individuals, G is the maximum number of generations, and E is the number of elites.

Algorithm 1 Pseudocode of GA

```
for $i = 1 : M$ do                                            ▷ Initialize population
    $\boldsymbol{\theta}_i[1] = \mathsf{rand}(n, 1)$;
end for
while !StopCondition do                                       ▷ Start GA
    for $g = 1 : G$ do
        for $i = 1 : M$ do                                    ▷ Evaluate fitness
            $\mathfrak{f}_i[g] = \mathfrak{F}(\boldsymbol{\theta}_i[g])$;
        end for
        sort($\boldsymbol{\theta}, \mathfrak{f}$);                                   ▷ Sort population
        for $e = 1 : E$ do                                    ▷ Elite
            $\boldsymbol{\theta}_e[g+1] = \boldsymbol{\theta}_e[g]$;
        end for
        while !CrossoverFinished do
            $i, j = $ selection($\boldsymbol{\theta}$);                          ▷ Selection
            $\boldsymbol{\theta}_i[g+1], \boldsymbol{\theta}_j[g+1] = $ cross($\boldsymbol{\theta}_i[g], \boldsymbol{\theta}_j[g]$);   ▷ Crossover
        end while
        while !MutationFinished do
            $i = $ selection($\boldsymbol{\theta}$);                             ▷ Selection
            $\boldsymbol{\theta}_i[g+1] = $ mutate($\boldsymbol{\theta}_i[g]$);             ▷ Mutation
        end while
    end for
end while
sort($\boldsymbol{\theta}, \mathfrak{f}$);                                           ▷ Sort population
return $\boldsymbol{\theta}_1[g]$
```

PSO is inspired by the behavior that birds show in their organization in swarm. Indeed, it is observable a very strong cooperation between birds in order to achieve a common objective, namely find the best source of food in the environment they live. In the swarm organization, each individual shows a dual behavior. From one side, it is a fully independent entity that behaves and takes decisions by its own, pursuing the best for itself. From the other side, its behavior influences and it is influenced by that of the other individuals. These two attitudes are identified as the *cognitive* and the *social* behaviors. From the cognitive perspective, the individual acts by taking into account only its own experience. In particular, it tries to move toward the best position it has found so far. From the social perspective, the individual chases the success of the others, and therefore it tries to move toward the best position found by the entire swarm.

Similarly to GA, also in PSO each individual, also called *particle*, represents a candidate solution for the problem (3.11). More precisely, the vector $\boldsymbol{\theta}$ belongs typically to $\mathbb{R}^n$ and it models the particle position inside the search space. Again, the performance of each individual is evaluated by means of the fitness function $\mathfrak{F}(.)$, so that also in this case the swarm can be represented by means of the list shown in Table 2.1.

The cognitive and the social behaviors are modeled by means of two attractors called *personal best* and *global best*. Specifically, the personal best $\boldsymbol{\Theta}_{p_i}$ is the best position that the i^{th} particle has found so far, whereas the global best $\boldsymbol{\Theta}_g$ is the best position that the entire swarm has found.

The population is evolved by applying repeatedly two update rules: the *velocity update* and the *position update*. Velocity update consists in evaluating the direction toward which each particle has to move considering the two attractors $\boldsymbol{\Theta}_{p_i}$ and $\boldsymbol{\Theta}_g$. Considering the i^{th} particle, the velocity update rule is evaluated with the following expression:

$$\boldsymbol{v}_i = \omega \boldsymbol{v}_i + c_p \boldsymbol{r_p}(\boldsymbol{\Theta}_{p_i} - \boldsymbol{\theta}_i) + c_g \boldsymbol{r_g}(\boldsymbol{\Theta}_g - \boldsymbol{\theta}_i) \tag{2.13}$$

where $\boldsymbol{v}_i = \{v_{i1}, \ldots, v_{in}\}$ is the velocity vector, ω is the *inertial* coefficient, c_p and c_g are the *acceleration* coefficients related to the cognitive and the social behaviors, respectively. The terms $\boldsymbol{r_p}$ and $\boldsymbol{r_g}$ are called *craziness* coefficients and they are vectors whose elements are randomly chosen from a uniform distribution in the range $[0, 1]$ for providing the particle with a stochastic behavior. In particular, the term $\omega \boldsymbol{v}_i$ aims at preventing a sudden change of direction, the term $c_p \boldsymbol{r_p}(\boldsymbol{\Theta}_{p_i} - \boldsymbol{\theta}_i)$ models the cognitive behavior, whereas

$c_g \boldsymbol{r_g}(\boldsymbol{\Theta}_g - \boldsymbol{\theta}_i)$ is the social component.

The position update is much more simple and it consists in adding the updated velocity to the current position:

$$\boldsymbol{\theta}_i = \boldsymbol{\theta}_i + \boldsymbol{v}_i \tag{2.14}$$

Summarizing, the pseudocode of PSO is shown in Algorithm 2.

Algorithm 2 Pseudocode of PSO

```
for $i = 1 : M$ do
    $\boldsymbol{\theta}_i = \mathsf{rand}(n, 1)$;                                    ▷ Initialize particle
    $\boldsymbol{v}_i = [0.5 - \mathsf{rand}(n, 1)]$;                            ▷ Initialize velocity
    $\mathfrak{f}_i = \mathfrak{F}(\boldsymbol{\theta}_i)$;                                        ▷ Evaluate fitness
    $\boldsymbol{\Theta}_{p_i} = \boldsymbol{\theta}_i$;                                          ▷ Initialize personal best
end for
$\boldsymbol{\Theta}_g = \boldsymbol{\theta}_i : \min_i \mathfrak{F}(\boldsymbol{\theta}_i)$;                              ▷ Initialize global best
while !StopCondition do
    for $i = 1 : M$ do
        $\boldsymbol{v}_i = \omega \boldsymbol{v}_i + c_p \boldsymbol{r_p}(\boldsymbol{\Theta}_{p_i} - \boldsymbol{\theta}_i) + c_g \boldsymbol{r_g}(\boldsymbol{\Theta}_g - \boldsymbol{\theta}_i)$;   ▷ Velocity update
        $\boldsymbol{\theta}_i = \boldsymbol{\theta}_i + \boldsymbol{v}_i$;                                ▷ Position update
        $\mathfrak{f}_i = \mathfrak{F}(\boldsymbol{\theta}_i)$;                                    ▷ Evaluate fitness
        if $\mathfrak{f}_i < \mathfrak{F}(\boldsymbol{\Theta}_{p_i})$ then
            $\boldsymbol{\Theta}_{p_i} = \boldsymbol{\theta}_i$;                              ▷ Update personal best
        end if
    end for
    $\bar{\boldsymbol{\Theta}}_g = \boldsymbol{\theta}_i : \min_i \mathfrak{F}(\boldsymbol{\theta}_i)$;
    if $\mathfrak{F}(\bar{\boldsymbol{\Theta}}_g) < \mathfrak{F}(\boldsymbol{\Theta}_g)$ then
        $\boldsymbol{\Theta}_g = \bar{\boldsymbol{\Theta}}_g$                                 ▷ Update global best
    end if
end while
return $\boldsymbol{\Theta}_g$
```

2.3 Fuzzy Logic

Fuzzy logic has been theorized by Lofti Zadeh [139] and it is a powerful methodology for handling uncertainty in a more effective way with respect to the traditional boolean logic, also called *crisp* logic. The main foundation is to go beyond the Aristotelian principle of the excluded third (tertium non datur), for which any logical proposition can be only completely true or completely false. Conversely, fuzzy logic aims at emulating the human reasoning in which any logical assumption is interpreted with a certain degree of truth, as well as the transition between true and false is continuous. This is because usually humans do not think in terms of mere numbers, but more in terms of abstract concepts, classes, and objects. For example, a 'ball' can be 'small', 'big', as well as 'red', 'blue', 'spherical', and 'oval'. For this reason, fuzzy logic is typically explained talking about linguistic variables.

Let consider the universe of discourse U composed of words, concepts, and related meanings. Let consider the subset $W \subset U$. A fuzzy set F is defined by a proposition p associating an element $x \in U$ to the subset W, and by a suitable function $\mu(x)$ assessing the grade of membership of x to W:

$$F = \{x, p, \mu(x)\} : \begin{cases} p : & x \mapsto W \\ \mu : & U \mapsto [0,1]. \end{cases} \tag{2.15}$$

For example the proposition p:

The 'ball' is 'red'

creates a fuzzy set where 'ball' is an element of U, the concept 'red' is a subset of U, and $\mu(.)$ evaluates the membership of 'ball' to 'red'.

The function $\mu(.)$ has codomain in the range $[0,1]$, with the values 0 and 1 indicating the falsity or the truth of p, respectively. In other words, 0 means that x does not belong to the subset W, and conversely 1 that it belongs completely to W. The membership function $\mu(.)$ can have different shapes. The most diffused ones are *triangular*, *trapezoidal*, *gaussian*, and *bell shaped*. In particular, $\mu(.)$ plays a crucial role for the definition of the fuzzy logical operators.

As well as in the boolean logic, also in the fuzzy one the main operators are *intersection* (AND), *union* (OR), and *complement* (NOT). Let A and B be two fuzzy sets having membership functions $\mu_A(.)$, and $\mu_B(.)$, respectively, the AND, OR, and NOT operators project to other fuzzy sets whose membership functions are evaluated as follows:

$$\begin{array}{lcl} A \text{ OR } B & \longrightarrow & \mu_{A\cap B}(x) = \max\left(\mu_A(x), \mu_B(x)\right) \\ A \text{ AND } B & \longrightarrow & \mu_{A\cup B}(x) = \min\left(\mu_A(x), \mu_B(x)\right) \\ \text{NOT } A & \longrightarrow & \mu_{\bar{A}}(x) = 1 - \mu_A(x). \end{array} \tag{2.16}$$

A graphical example of the fuzzy logical operators is shown in Figure 2.6, where two fuzzy sets characterized respectively by a triangular and trapezoidal membership functions have been considered.

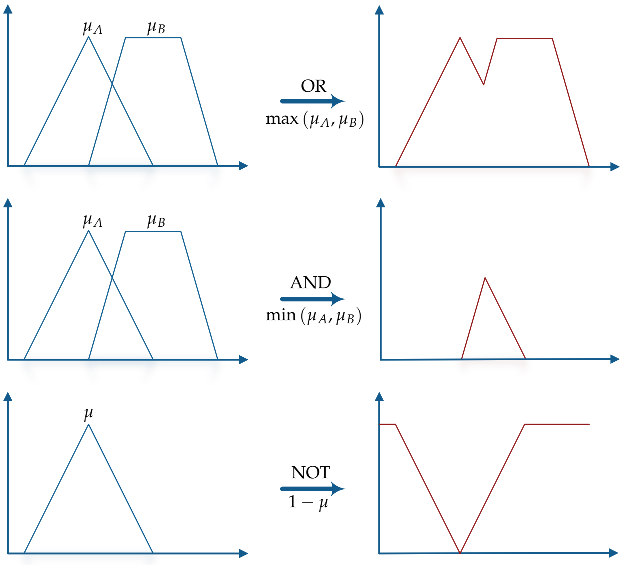


Figure 2.6: Example of fuzzy logical operators.

The AND, OR, and NOT operators are used for defining the logical rules of the fuzzy reasoning. In particular, these rules respect the IF-THEN structure as follows:

$$\textbf{IF} \;\; x \,\text{IS}\, A \;\; \textbf{THEN} \;\; y \,\text{IS}\, B \tag{2.17}$$

or considering two variables:

$$\textbf{IF} \;\; x \,\text{IS}\, A \;\; \text{AND} \; y \,\text{IS}\, B \;\; \textbf{THEN} \;\; z \,\text{IS}\, C. \tag{2.18}$$

In general, any fuzzy rule is characterized by two elements: the antecedent, that is a generic combination of logical conditions on the inputs, and the consequent, that is the fuzzy set inferred by the antecedent. Thus, expressions (2.17) and (2.18) define an implication that associates the antecedents $x \,\text{IS}\, A$, and $x \,\text{IS}\, A$ AND $y \,\text{IS}\, B$ to the corresponding logical consequents $y \,\text{IS}\, B$, and $z \,\text{IS}\, C$, respectively.

From a practical and mathematical point of view, the fuzzy reasoning is implemented by means of Fuzzy Inference Systems (FISs). In particular, FISs are processing tools that receive as input a vector of real numbers and provide as output a real number as well. They have been used in different applications, such as function approximation [31], [140], classification [141], decision making systems [30], and control systems [142], [143].

Any FIS involves four main phases: *fuzzification*, *inference*, *composition*, and *defuzzification*. The fuzzification consists in evaluating the membership of the inputs to the related fuzzy sets appearing in the antecedent of every fuzzy rule. The inference phase involves the evaluation of the *fire strength* of each fuzzy rule, that represents the veracity of the rules themselves. The composition step consists in mixing the rule consequents on the basis of the fire strengths of the related rules. Finally, the defuzzification process evaluates a real value from the fuzzy output of the composition phase.

Two main FIS structures have been proposed in the literature, namely the Mamdani [144] and the Takagi-Sugeno [145] ones.

In the Mamdani FIS, the domain of every input variable and that of the output variable are split in a certain number of fuzzy sets, such that the related membership functions are overlapped and they cover the entire domain. A graphical representation of a Mamdani FIS with two inputs and one output is shown in Figure 2.7.

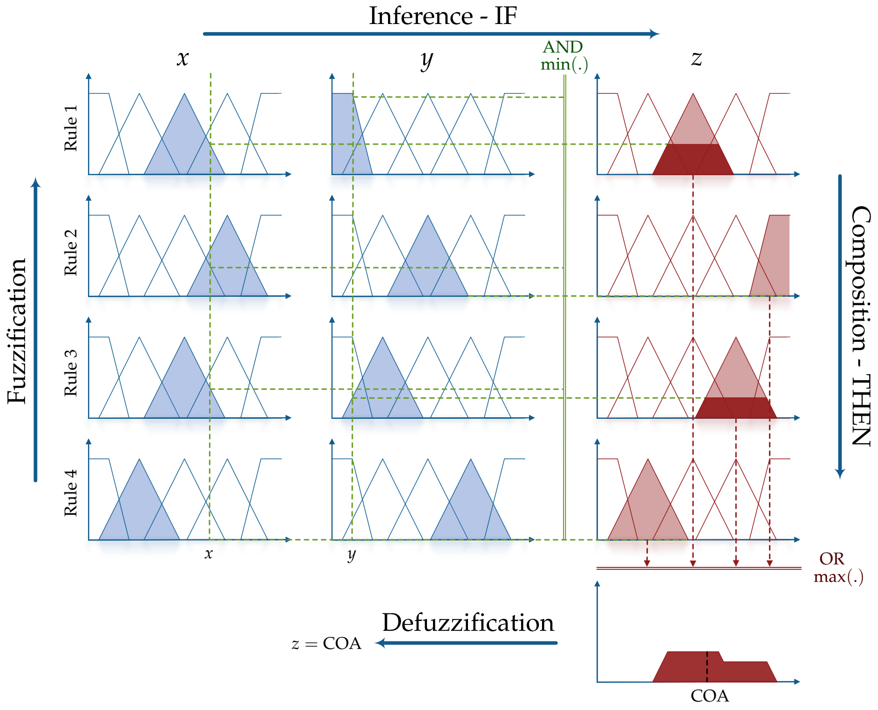


Figure 2.7: Example of a Mamdani FIS with two inputs (x and y) and one output (z). Defuzzification is made by means of COA method.

The IF-THEN reasoning is based on the AND-OR logic. Thus, each rule antecedent consists in the application of the AND operator between fuzzy sets of the inputs, whereas the combination of the rules is made by means of the OR operator. Different options are available for the defuzzification process. The most diffused one is the Center Of Area (COA) consisting in finding the crisp value z where a vertical line would slice the area of the output membership function into two equal parts. Alternatives are the *first of maximum*, *last of maximum*, and *mean of maximum*, in which the defuzzified value is the first edge, the last edge, and the center of the highest output fuzzy set, respectively.

The Takagi-Sugeno FIS is similar to the Mamdani one, except for the definition of rule consequents. Indeed, conversely to Mamdani, they are not fuzzy sets but mathematical expressions of the inputs. Typically, these expressions are polynomials, and FIS is said to be a k^{th} order Takagi-Sugeno, with k being the order of the polynomial used in the rule consequent expressions. In particular, the most used Takagi-Sugeno FIS is the first order one, in which consequents are hyperplanes. An example of a first order Takagi-Sugeno rule is the following:

$$\textbf{IF}\ \ x \in A\ \text{ AND }\ y \in B\ \ \textbf{THEN}\ \ z = ax + by + c \tag{2.19}$$

where A and B are fuzzy sets of the inputs x and y, respectively, whereas a, b, and c are the coefficients of the consequent hyperplane.

Adaptive Neuro-Fuzzy Inference System (ANFIS) [140], [146], [147] is the mathematical implementation of the Takagi-Sugeno FIS. The ANFIS architecture is shown in Figure 2.8, where gaussian membership functions have been considered for the fuzzification of the inputs.

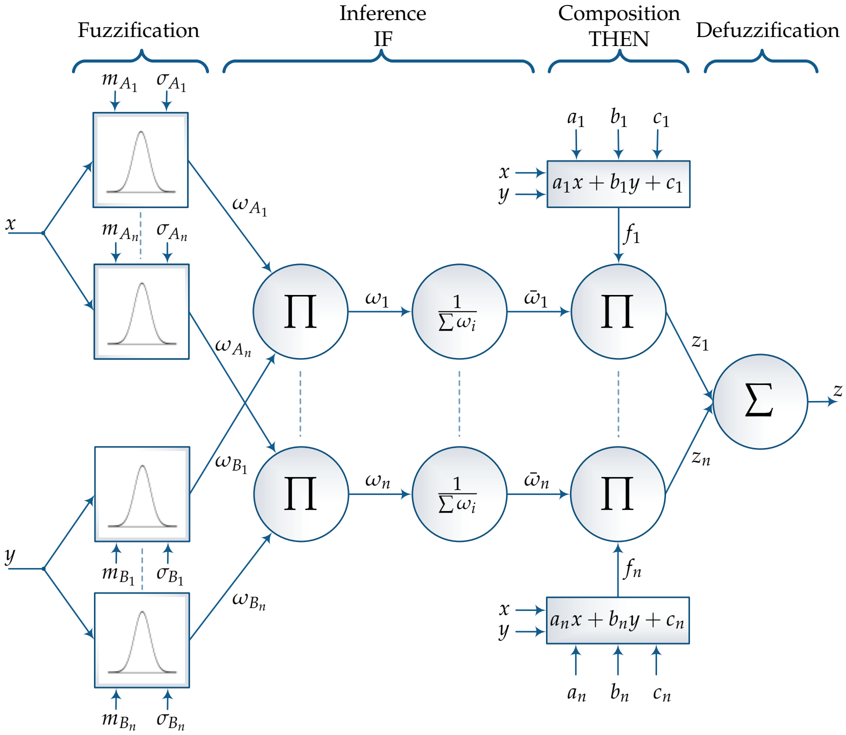


Figure 2.8: Takagi-Sugeno ANFIS network with gaussian membership functions. Parameters m_{X_i} and σ_{X_i} are the mean and standard deviation of the gaussian function related to the fuzzy set X_i.

It is clear that ANFIS inherits the feed-forward architecture from neural networks and this is the reason why it is called neuro-fuzzy. ANFIS is organized in 5 layers, each one performing one phase of the fuzzy reasoning. The first layer is devolved to the fuzzification step. Herein, it is possible to use different membership functions, such as triangular, trapezoidal, or gaussian. The second and the third layers implement the inference phase. Specifically, the second layer evaluates the fire strength of each rule, whereas the third layer performs the normalization of the fire strengths with respect to their sum. It must be noticed that the AND operator is performed by means of the multiplication of the membership functions, instead of the minimum used in the Mamdani FIS. The fourth layer composes the fuzzy rules by multiplying each consequent expression by the normalized fire strength of the related rule. Finally, the fifth layer performs the defuzzification. To this aim, it is possible to consider two different approaches. The weighted average (AVG) consists in evaluating the crisp output z by performing the average of the consequents' expressions by using the terms $\bar{\omega}_i$ as weights. Conversely, the Winner Takes All (WTA) method evaluates z by considering only the consequent expression that has achieved the highest fire strength $\bar{\omega}$:

$$\begin{aligned} \text{AVG} \quad &\longrightarrow \quad z = \frac{\sum_{i=1}^{n} \bar{\omega}_i z_i}{\sum_{i=1}^{n} \bar{\omega}} \\ \text{WTA} \quad &\longrightarrow \quad z = z_i \ : \ \{\bar{\omega}_i = \max_{j=1,\dots,n} \bar{\omega}_j\}. \end{aligned} \tag{2.20}$$

2.4 Multi-Swarm Hybrid Genetic Particle Swarm Optimization

An improved version of PSO, called Multi-Swarm Hybrid Genetic PSO (MSHG-PSO), has been developed for solving several optimization problems concerning the research activity. This version involves the implementation of three main improvements with respect to the vanilla version of PSO with the aim of reducing the stagnation effect, increasing the exploration and exploitation capability, and avoiding the convergence to local minima.

The main improvement has interested the hybridization of PSO with GA. This hybridization has aimed at enhancing the exploitation capability of the algorithm, facilitating it in escaping from local minima. In details, a certain number of new particles are generated at the end of each iteration by applying the genetic operators of crossover and mutation to a set of individuals selected by means of the K-tournament algorithm. Once the new individuals have been generated, these substitute a set of particles randomly chosen from the worst half of the original swarm. Mutation consists in adding a random number to a random set of elements of the position vector. On the other hand, crossover is performed by swapping a random set of elements from the two parent particles.

The second improvement has been the introduction of a multi-swarm heuristic. Herein, the entire swarm is split in a number $G \geq 1$ of sub-swarms, each one characterized by having its own global best. In this topology, each particle belongs to the sub-swarm related to the geometrically closest global best and consequently it updates its position with respect to that global best. The advantage of using a multi-swarm implementation consists in an improved exploration capability that allows to search in different areas of the solution space and to avoid a premature convergence to local minima.

The final improvement was the implementation of the *Guaranteed Convergence* update rule [148]. In this version, the velocity of a particle being close to its global best is updated with expression (2.21) in place of the update rule of the standard PSO:

$$\boldsymbol{v}_i = \omega \boldsymbol{v}_i + \boldsymbol{\Theta}_g - \boldsymbol{\theta}_i + \delta(1 - 2\boldsymbol{r}) \tag{2.21}$$

where $\boldsymbol{r}$ is a random vector drawn from a uniform distribution in $[0, 1]$ and δ is a scaling factor. This update rule introduces a more random behavior for the particles closer to the global best, allowing to avoid stagnation and to improve exploitation around the current global best.

Besides the ability of dealing with real numbers, it has been implemented also a binary version of MSHG-PSO, called MSHG-BPSO, in order to allow its application to optimization problems that involve binary variables. To this aim, it has been necessary to align the velocity and position update rules of MSHG-PSO to those of the Binary PSO (BPSO) [149], [150].

In the BPSO context, the j^{th} element of the velocity vector is interpreted as the probability of the j^{th} bit of the position vector of being 0 or 1. Thus, the velocity update rule is the same of the standard PSO, whereas the position update rule is evaluated by means of a sigmoid transformation of the particle velocity. The update rules of BPSO are summarized as follows:

$$\begin{aligned} \boldsymbol{v}_i &= \omega \boldsymbol{v}_i + c_p \boldsymbol{r_p}(\boldsymbol{\Theta}_{p_i} - \boldsymbol{\theta}_i) + c_g \boldsymbol{r_g}(\boldsymbol{\Theta}_g - \boldsymbol{\theta}_i) \\ \theta_{ij} &= \begin{cases} 0 & \text{if } \frac{1}{1+\exp\left(-v_{ij}\right)} \leq r_{ij} \\ 1 & \text{if } \frac{1}{1+\exp\left(-v_{ij}\right)} > r_{ij} \end{cases} \end{aligned} \tag{2.22}$$

where $\boldsymbol{\theta}_i$ and $\boldsymbol{v}_i$ are the position and the velocity vectors related to the i^{th} particle, respectively. Similarly, θ_{ij} and v_{ij} are the j^{th} element of $\boldsymbol{\theta}_i$ and $\boldsymbol{v}_i$, respectively. Specifically, each element of $\boldsymbol{\theta}_i$, $\boldsymbol{\Theta}_{p_i}$ and $\boldsymbol{\Theta}_g$ is a binary variable. The terms r_{ij} is a random number drawn from a uniform distribution in the range $[0, 1]$. Concerning the genetic hybridization, the mutation operator complements a random set of bits in the original position vector. Similarly to MSHG-PSO, the crossover operator generates two offspring particles by swapping a random set of bits from the two parent individuals.

Chapter 3

BMS Algorithms

3.1 Comparison Between EKF and SR-UKF for SoC Estimation

3.1.1 Nonlinear Kalman Filters

Kalman filter techniques are the most promising methods for performing SoC estimation. In particular, EKF and UKF have been largely used to this purpose. Therefore, a comparison between these two main estimation techniques has been performed aiming at analyzing which of them is the most effective and reliable solution for dealing with the crucial task of SoC estimation.

Let consider a discrete nonlinear system described by the following equations:

$$\begin{aligned} \boldsymbol{x}[k+1] &= \boldsymbol{\mathcal{F}}(\boldsymbol{x}[k], \boldsymbol{u}[k]) + \boldsymbol{q}[k] \\ \boldsymbol{y}[k] &= \boldsymbol{\mathcal{G}}(\boldsymbol{x}[k], \boldsymbol{u}[k]) \ + \boldsymbol{v}[k] \end{aligned} \tag{3.1}$$

where $\boldsymbol{x}$, $\boldsymbol{y}$, and $\boldsymbol{u}$ are the state vector, the output vector, and the input vector, respectively; $\boldsymbol{\mathcal{F}}(.)$ and $\boldsymbol{\mathcal{G}}(.)$ are nonlinear functions defining the state update, and the output equations, respectively; $\boldsymbol{q} = \mathcal{N}(0, \boldsymbol{Q})$ and $\boldsymbol{v} = \mathcal{N}(0, \boldsymbol{V})$ are the state and output noises drawn from a zero mean gaussian distribution having $\boldsymbol{Q}$ and $\boldsymbol{V}$ covariance matrices, respectively.

A Kalman filter is an iterative algorithm using a mixture of model-driven and data-driven techniques aiming at estimating the state vector $\boldsymbol{x}$ that minimizes the error between the output of the real system and that estimated by the model. The block diagram of the algorithm is shown in Figure 3.1.

Kalman filter involves two main phases: *prediction* and *update*. The prediction phase is the model-driven part of the algorithm. Herein, the available model is used to perform a prediction of the next state, of the next output, and of their related statistics, namely the state-covariance matrix $\boldsymbol{P}_{xx}$, the

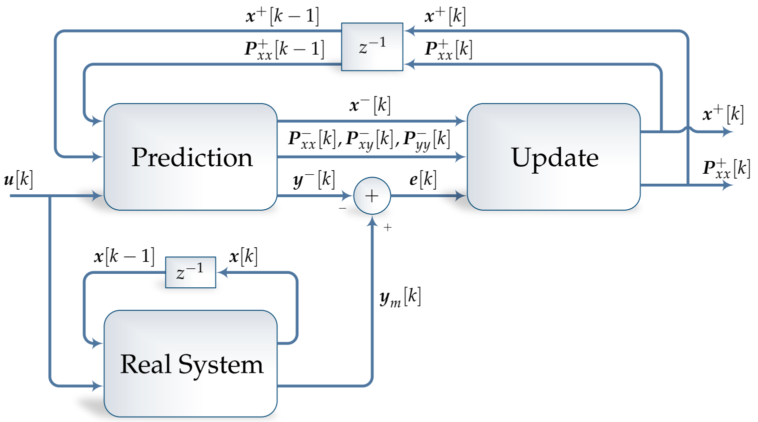


Figure 3.1: Block diagram of a stochastic state observer.

output-covariance matrix $\boldsymbol{P}_{yy}$, and the state/output-covariance matrix $\boldsymbol{P}_{xy}$. The update phase uses a data-driven approach in which the output vector $\boldsymbol{y}_m$ measured on the real system is used for updating and correcting the predicted state vector and the state-covariance matrix.

Although the asymptotic stability of Kalman filters can be established only for linear systems, the same technique can be generalized to the nonlinear ones by means of EKF and UKF.

The core of EKF consists in the linearization of the nonlinear system around the a posteriori estimation of the state vector $\boldsymbol{x}^+[k]$. This operation is performed by computing the Jacobians $\hat{\boldsymbol{A}}[k]$ and $\hat{\boldsymbol{C}}[k]$ related to the state update equation $\boldsymbol{\mathcal{F}}(.)$ and the output equation $\boldsymbol{\mathcal{G}}(.)$, respectively:

$$\hat{\boldsymbol{A}}[k] = \left.\frac{\partial \boldsymbol{\mathcal{F}}(\boldsymbol{x}[k], \boldsymbol{u}[k])}{\partial \boldsymbol{x}[k]}\right|_{\boldsymbol{x}=\boldsymbol{x}^+[k]} \qquad \hat{\boldsymbol{C}}[k] = \left.\frac{\partial \boldsymbol{\mathcal{G}}(\boldsymbol{x}[k], \boldsymbol{u}[k])}{\partial \boldsymbol{x}[k]}\right|_{\boldsymbol{x}=\boldsymbol{x}^+[k].} \tag{3.2}$$

Thanks to the linearization process, EKF is formulated as a linear Kalman filter applied to the linearized system defined by the Jacobians $\hat{\boldsymbol{A}}$ and $\hat{\boldsymbol{C}}$. Thus, the prediction and update phases of EKF are summarized in Algorithm 3, where the initial state vector $\boldsymbol{x}^+[0]$ and the initial state-covariance matrix $\boldsymbol{P}^+_{xx}[0]$ are assumed to be set.

As opposed to EKF, UKF does not apply any linear approximation. It uses a deterministic sampling of the system and a weighted average of the resulting samples in order to predict the current state vector and the current state-covariance matrix. These samples, called *sigma points*, are obtained through the unscented transformation defined by Jeffrey Uhlmann in reference [151] and formalized by Simon Julier for the UKF context in [66].

Let $\boldsymbol{P}^+_{xx}[k-1]$ and $\boldsymbol{x}^+[k-1]$ be the a posteriori predictions of the state vector and the state-covariance matrix at the previous filter step, respectively,

Algorithm 3 Pseudocode of EKF.

```
**Prediction step for EKF**
Evaluate the Jacobians matrices $\hat{\boldsymbol{A}}[k-1]$ and $\hat{\boldsymbol{C}}[k]$.
Predict state and output vectors.
    $\boldsymbol{x}^-[k] = \boldsymbol{\mathcal{F}}\left(\boldsymbol{x}^+[k-1], \boldsymbol{u}[k-1]\right)$
    $\boldsymbol{y}^-[k] = \boldsymbol{\mathcal{G}}\left(\boldsymbol{x}^-[k], \boldsymbol{u}[k]\right)$
Predict covariance matrices.
    $\boldsymbol{P}^-_{xx}[k] = \hat{\boldsymbol{A}}[k-1]\boldsymbol{P}^+_{xx}[k-1]\hat{\boldsymbol{A}}^T[k-1] + \boldsymbol{Q}$
    $\boldsymbol{P}^-_{yy}[k] = \hat{\boldsymbol{C}}^T[k]\left(\hat{\boldsymbol{C}}[k]\boldsymbol{P}^-_{xx}[k]\hat{\boldsymbol{C}}^T[k] + \boldsymbol{V}\right)$
    $\boldsymbol{P}^-_{xy}[k] = \boldsymbol{P}^-_{xx}[k]\hat{\boldsymbol{C}}^T[k]$
**Update step for EKF**
    $\boldsymbol{K}[k] = \boldsymbol{P}^-_{xy}[k](\boldsymbol{P}^-_{yy}[k])^{-1}$
    $\boldsymbol{x}^+[k+1] = \boldsymbol{x}^-[k+1] + \boldsymbol{K}[k]\left(\boldsymbol{y}_m[k] - \boldsymbol{y}^-[k]\right)$
    $\boldsymbol{P}^+_{xx}[k+1] = \boldsymbol{P}^-_{xx}[k] - \boldsymbol{K}[k]\hat{\boldsymbol{C}}[k]\boldsymbol{P}^-_{xx}[k]$
```

the sigma points $\boldsymbol{\chi}$ and the related weights w are evaluated as follows:

$$\begin{aligned}
&\boldsymbol{Z} = \left\{ +\sqrt{\boldsymbol{P}^+_{xx}[k-1]} \quad \middle| \; -\sqrt{\boldsymbol{P}^+_{xx}[k-1]} \right\} = \{\boldsymbol{z}_1|\ldots|\boldsymbol{z}_{2D}\} \\
&\boldsymbol{\chi}^-_0[k] = \boldsymbol{x}^+[k-1], \\
&\boldsymbol{\chi}^-_i[k] = \boldsymbol{x}^+[k-1] + \sqrt{\frac{L}{\lambda}}\,\boldsymbol{z}_i, \quad i = 1, \ldots, 2D \\
&w^{(m)}_0 = 1 - \lambda, \quad w^{(c)}_0 = 2 - \lambda - \alpha^2 + \beta, \quad w = \frac{\lambda}{2D}
\end{aligned}$$

where D is the number of state variables, $\lambda = L/(\alpha^2(L+\kappa))$ is a scaling factor, and $\boldsymbol{z}_i$ is the i^{th} column of the matrix $\boldsymbol{Z}$. The coefficients $\alpha \in (0,1]$ and $\kappa \geq 0$ determine the spread of the sampling around the state $\boldsymbol{x}^+$; β is a coefficient incorporating information about the probability distribution, and $\beta = 2$ is optimal for the gaussian one. The terms $w^{(m)}_0$ and $w^{(c)}_0$ are the weights related to the zeroth sigma point for the evaluation of the mean and the covariance, respectively, and w is the weight related to all the other samples.

The prediction and update steps for the UKF algorithm are described in Algorithm 4, where again the initial state $\boldsymbol{x}^+[0]$ and the initial state-covariance matrix $\boldsymbol{P}^+_{xx}[0]$ are assumed to be set.

The most critical operation in UKF is the evaluation of the square root of the matrix $\boldsymbol{P}_{xx}$, typically performed through a Cholesky decomposition.

Algorithm 4 Pseudocode of UKF.

```
Prediction step for UKF
Evaluate Sigma Points $\chi_i[k]$ considering the previous state $x^+[k-1]$ and the previous square root state-covariance matrix $P^+_{xx}[k-1]$.
Predict state and output vectors.
    $\chi_i^-[k] = \mathcal{F}(\chi_i[k-1], u[k-1]), \quad i = 0, \dots, 2D$
    $v_i^-[k] = \mathcal{G}(\chi_i^-[k], u[k]), \quad i = 0, \dots, 2D$
    $x^-[k] = w_0^{(m)} \chi_0^-[k] + w \sum_{i=1}^{2D} \chi_i^-[k]$
    $y^-[k] = w_0^{(m)} v_0^-[k] + w \sum_{i=1}^{2D} v_i^-[k]$
Predict covariance matrices.
    $P^-_{xx}[k] = w_0^{(c)} (\chi_0^-[k] - x^-[k])(\chi_0^-[k] - x^-[k])^T + w \sum_{i=1}^{2D} (\chi_i^-[k] - x^-[k])(\chi_i^-[k] - x^-[k])^T$
    $P^-_{yy}[k] = w_0^{(c)} (v_0^-[k] - y^-[k])(v_0^-[k] - y^-[k])^T + w \sum_{i=1}^{2D} (v_i^-[k] - y^-[k])(v_i^-[k] - y^-[k])^T$
    $P^-_{xy}[k] = w_0^{(c)} (\chi_0^-[k] - x^-[k])(v_0^-[k] - y^-[k])^T + w \sum_{i=1}^{2D} (\chi_i^-[k] - x^-[k])(v_i^-[k] - y^-[k])^T$
Update step for UKF
    $K[k] = P^-_{xy}[k](P^-_{yy}[k])^{-1}$
    $x^+[k] = x^-[k] + K[k](y_m[k] - y^-[k])$
    $P^+_{xx}[k] = P^-_{xx}[k] - K[k] P^-_{yy}[k] K[k]^T$
```

Unfortunately, P_{xx} can lose its positive definite property during the UKF iterations, so that it is not anymore possible to evaluate its Cholesky decomposition. To avoid this problem, a more numerically stable implementation of UKF, known as Square Root UKF (SR-UKF), has been proposed in reference [152]. In this implementation, the square root of the state-covariance matrix $S_{xx} = \sqrt{P_{xx}}$ is directly propagated iteration by iteration, avoiding to repeatedly apply the Cholesky decomposition to the matrix P_{xx}. In this way, the positive definite property of S_{xx} is guaranteed and the numerical stability of the algorithm is improved.

SR-UKF involves two fundamental linear algebra techniques: QR decomposition and Cholesky rank one update/downdate (see [152] for details about these techniques). In the following, the function $\mathrm{qr}\,(.)$ refers to the operator performing QR decomposition and returning the upper triangular part of the matrix R. Similarly, the function $\mathrm{cholupdate}\,(S, x, \mu)$ refers to the rank one update/downdate of S by means of the vector $\sqrt{|\mu|}x$; the update happens if $\mu > 0$ and the downdate if $\mu < 0$.

Algorithm 5 Pseudocode of SR-UKF.

```
    **Prediction step for SR-UKF**
    Evaluate Sigma Points $\boldsymbol{\chi}_i[k]$ considering the previous state $\boldsymbol{x}^+[k-1]$ and the previous square root state-covariance matrix $\boldsymbol{S}_{xx}^+[k-1]$.
    Predict state and output vectors.
        $\boldsymbol{\chi}_i^-[k] = \boldsymbol{\mathcal{F}}(\boldsymbol{\chi}_i[k-1], \boldsymbol{u}[k-1]), \quad i = 0, \dots, 2D$
        $\boldsymbol{v}_i^-[k] = \boldsymbol{\mathcal{G}}(\boldsymbol{\chi}_i^-[k], \boldsymbol{u}[k]), \quad i = 0, \dots, 2D$
        $\boldsymbol{x}^-[k] = w_0^{(m)} \boldsymbol{\chi}_0^-[k] + w \sum_{i=1}^{2D} \boldsymbol{\chi}_i^-[k]$
        $\boldsymbol{y}^-[k] = w_0^{(m)} \boldsymbol{v}_0^-[k] + w \sum_{i=1}^{2D} \boldsymbol{v}_i^-[k]$
    Predict covariance matrices.
        $\begin{cases} \boldsymbol{X}[k] = \{\boldsymbol{\chi}_1^-[k] - \boldsymbol{x}^-[k] \mid \dots \mid \boldsymbol{\chi}_{2D}^-[k] - \boldsymbol{x}^-[k]\} \\ \boldsymbol{S}_{xx}^*[k] = \mathrm{qr}\left(\left\{ \sqrt{w}\boldsymbol{X}[k] \mid \sqrt{\boldsymbol{Q}} \right\}^T\right) \\ \boldsymbol{S}_{xx}^-[k] = \mathrm{cholupdate}\left(\boldsymbol{S}^*[k], \boldsymbol{\chi}_0^-[k], w_0^{(c)}\right) \end{cases}$
        $\begin{cases} \boldsymbol{Y}[k] = \{\boldsymbol{v}_1^-[k] - \boldsymbol{y}^-[k] \mid \dots \mid \boldsymbol{v}_{2D}^-[k] - \boldsymbol{y}^-[k]\} \\ \boldsymbol{S}_{yy}^*[k] = \mathrm{qr}\left(\left\{ \sqrt{w}\boldsymbol{Y}[k] \mid \sqrt{\boldsymbol{V}} \right\}^T\right) \\ \boldsymbol{S}_{yy}^-[k] = \mathrm{cholupdate}\left(\boldsymbol{S}_{yy}^*[k], \boldsymbol{v}_0^-[k], w_0^{(c)}\right) \end{cases}$
    $\boldsymbol{P}_{xy}^-[k] = w_0^{(c)} (\boldsymbol{\chi}_0^-[k] - \boldsymbol{x}^-[k])(\boldsymbol{v}_0^-[k] - \boldsymbol{y}^-[k])^T + w \sum_{i=1}^{2D} (\boldsymbol{\chi}_i^-[k] - \boldsymbol{x}^-[k])(\boldsymbol{v}_i^-[k] - \boldsymbol{y}^-[k])^T$
    **Update step for SR-UKF**
        $\boldsymbol{K}[k] = \boldsymbol{P}_{xy}^-[k](\boldsymbol{S}_{yy}^-[k]\boldsymbol{S}_{yy}^-[k]^T)^{-1}$
        $\boldsymbol{x}^+[k] = \boldsymbol{x}^-[k] + \boldsymbol{K}[k](\boldsymbol{y}_m[k] - \boldsymbol{y}^-[k])$
        $\boldsymbol{S}_{xx}^+[k] = \mathrm{cholupdate}\left(\boldsymbol{S}_{xx}^-[k], \boldsymbol{K}[k]\boldsymbol{S}_{yy}^-[k], -1\right)$
```

The prediction and update phases of SR-UKF are described in Algorithm 5. Once again, the initial state $\boldsymbol{x}^+[0]$ and the initial square root state-covariance matrix $\boldsymbol{S}_{xx}^+[0] = \sqrt{\boldsymbol{P}_{xx}^+[0]}$ are assumed to be set.

3.1.2 Hardware in The Loop Comparison

Experimental Setup

Because of the more numerical stability of SR-UKF with respect to UKF, the comparison in the SoC estimation task has been performed upon EKF and SR-UKF.

Both the SoC estimators have been implemented in an Hardware in the Loop (HIL) architecture in order to get more realistic results. To this aim, an *Arduino Due* board and a *Matlab* user interface have been configured for emulating a BMS device managing a single cell BESS. Specifically, the *Arduino Due* board works as the master device running the SoC estimation algorithms, whereas the *Matlab* interface emulates the slave device by sending the values of voltage and current measurements to the master.

Voltage, current, and SoC data has been collected through a previous acquisition campaign performed on one *A123 Nanophosphate AHR23113M1Ultra-B* cell[1]. In particular, this was a lithium iron-phosphate cell characterized by a capacity of 2.4 Ah. A realistic and highly dynamic current profile derived from the US06 driving cycle from the *Federal Test Procedure* set has been considered for better analyzing the accuracy and the reliability of EKF and SR-UKF. Specifically, the test has been composed of one hour of rest at zero current followed by six US06 cycles for a total of two hours of measurements. The real SoC sequence has been evaluated by applying the coloumb counting algorithm related to expression (1.3) initialized with the correct SoC value, i.e. $\text{SoC}[0] = 100\,\%$, and considering the real capacity $C_n = 2.4\,\text{Ah}$. Data has been collected at a sampling time of 1 s, that is fast enough both for tracking the dynamics of electrochemical cells and for providing a timely SoC estimation for most of the BESS applications. Moreover, this sampling time together with the duration of the test ensure a limited accumulation errors for expression (1.3), so that the obtained SoC sequence can be considered close to the real SoC of the cell. The current profile, the voltage response, and the related SoC sequences are shown in Figure 3.2.

Both SR-UKF and EKF have been implemented on the *Arduino Due* board with the aim of investigating which of them offers the most robust and the most effective SoC estimation upon a limited hardware that will be similar to that of a real BMS. In order to perform a rigorous comparison between EKF and SR-UKF, the two algorithms have been tuned with the same model, the same noise-covariance matrices, and the same initialization of the state-covariance matrix. Specifically, the considered cell model is the nonlinear ECM that will be discussed in Section 4.2.1.

[1] The measurement campaign has been performed by Dr. Gian Luca Storti from the *Center for Automotive Research* (CAR), *The Ohio State University*, Columbus, Ohio, USA.

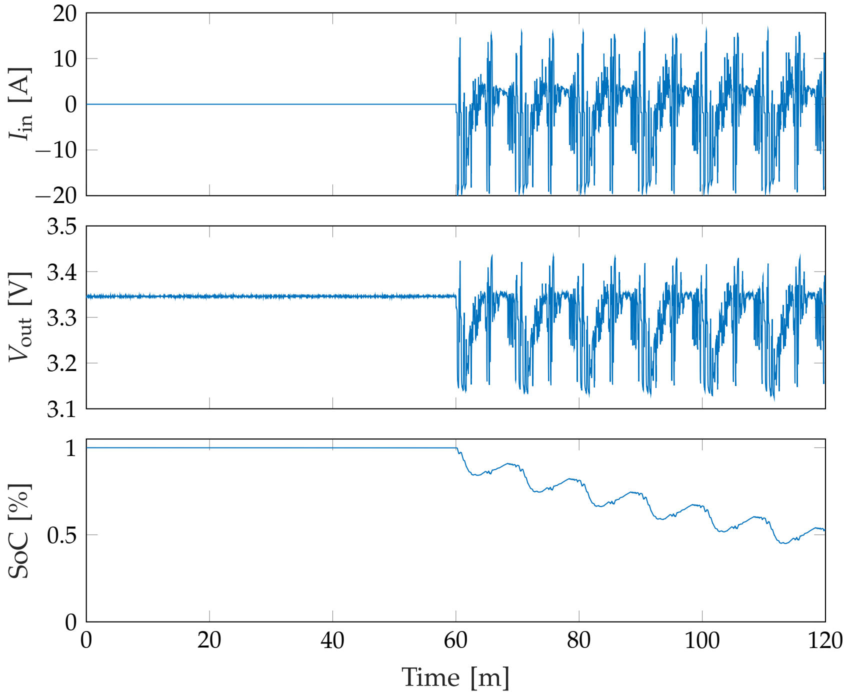


Figure 3.2: Current profile, voltage response and SoC evolution related to the US06 driving cycle.

The convergence capability of the algorithms has been investigated considering four different initialization values for SoC. The adopted performance metric is the percentage error evaluated as follows:

$$\text{Error}\,[\%] = 100\frac{\left|\text{SoC} - \tilde{\text{SoC}}\right|}{\text{SoC}} \tag{3.3}$$

where SoC and $\tilde{\text{SoC}}$ are the real and the estimated SoC, respectively. The configuration of EKF and SR-UKF are summarized as follows:

$$\boldsymbol{P}^{+}_{xx}[0] = \begin{pmatrix} 10^{-1} & 0 \\ 0 & 10^{-4} \end{pmatrix}, \quad \boldsymbol{x}^{+}[0] = \begin{pmatrix} \text{SoC}[0] \\ 0 \end{pmatrix},$$
$$\boldsymbol{Q} = \begin{pmatrix} 10^{-11} & 0 \\ 0 & 10^{-11} \end{pmatrix}, \quad \boldsymbol{V} = 10^{-6},$$
$$\alpha = 0.5, \quad \beta = 2, \quad \kappa = 0,$$
$$\text{Model: Nonlinear ECM}^{2}.$$

[2] Refer to Section 4.2.1 for more details about this cell model and the related system identification procedure.

Experimental Results

The results obtained by setting SoC[0] = 100 % (i.e. the true value) are shown in Figure 3.3. It can be noticed that both the algorithms perform a good SoC estimation. In particular, EKF outperforms SR-UKF keeping an error almost always lower than 0.5 %. Conversely, after losing temporarily the convergence, SR-UKF keeps an error lower than 2 %.

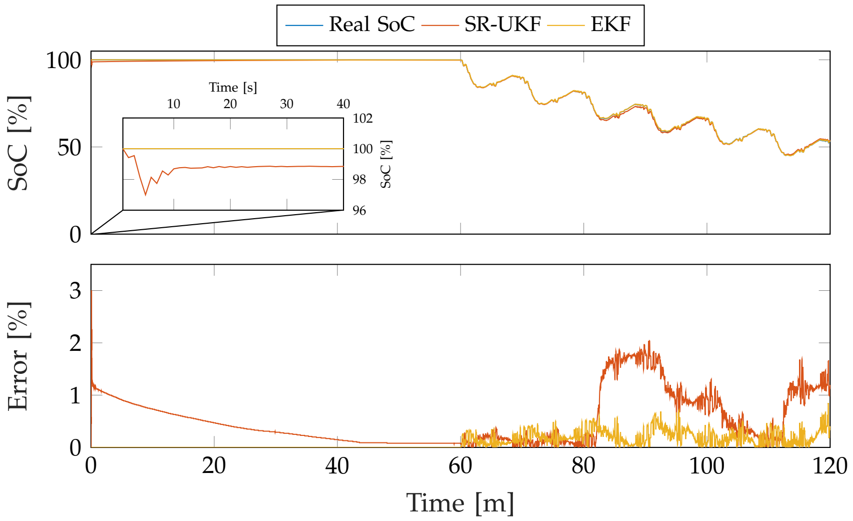


Figure 3.3: SoC estimation and related error performed by setting SoC[0] = 100 %.

The results obtained considering a 25 % error in the initialization of SoC (i.e. SoC[0] = 75 %) are shown in Figure 3.4.

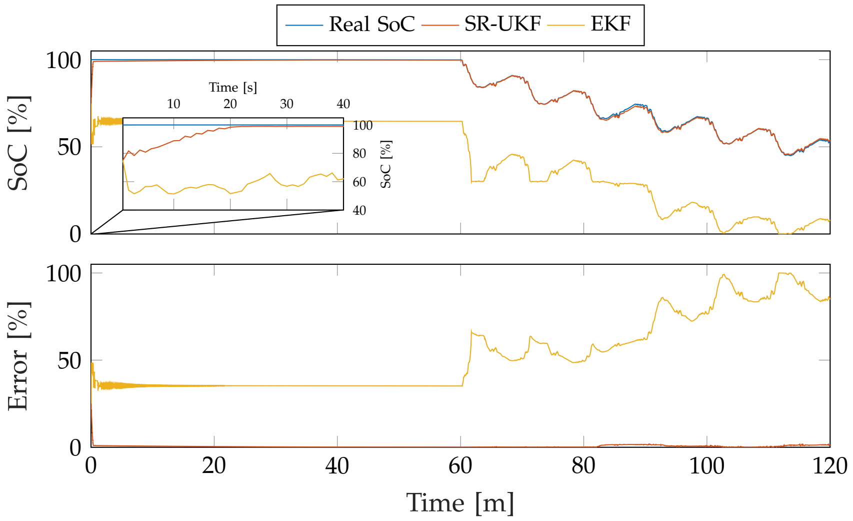


Figure 3.4: SoC estimation and related error performed by setting SoC[0] = 75 %.

It is clear that EKF does not converge anymore, whereas SR-UKF keeps the same performances seen before. In particular, it converges in about 20 s and it keeps again an error lower than 2 % after reaching the convergence.

In order to further investigate the convergence capability of SR-UKF over higher initialization errors, two more tests have been performed. In particular, it has been considered an initial SoC of 50 % and 25 %, corresponding to an initial error of 50 % and 75 %, respectively. EKF has not been considered since it has not reached the convergence with these initialization values. The results are shown in Figure 3.5.

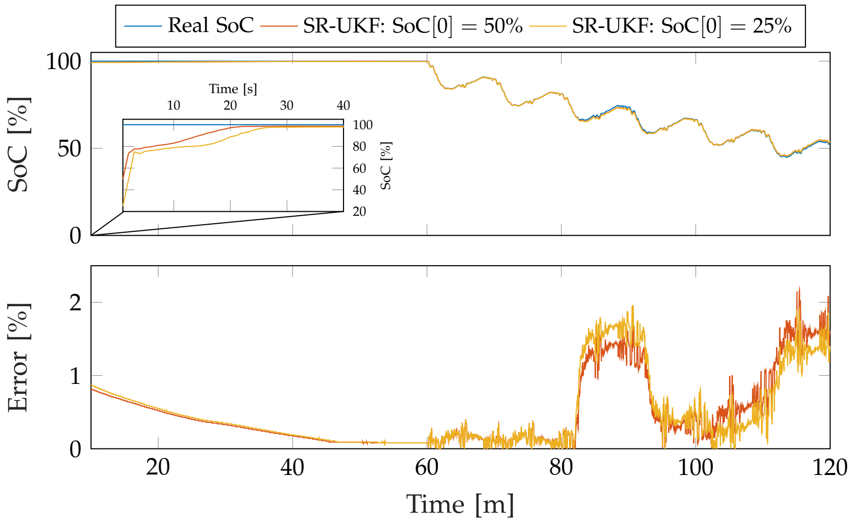


Figure 3.5: SoC estimation and related error performed by SR-UKF by setting the initial SoC to 50 % and to 25 %.

It can be seen that SR-UKF succeeds again at reaching the convergence in about 20 s when the initial SoC has been set to 50 %, and in about 25 s when the initial SoC was 25 %. Furthermore, once again the error committed is almost always lower than 2 % for both the initialization values once the convergence has been reached.

The above results highlight that SR-UKF is the most suitable choice for developing an effective and reliable SoC estimation. Indeed, except the case when the correct initialization of SoC has been considered, SR-UKF performs a more robust and a more accurate SoC estimation with respect to EKF. Specifically, EKF achieves a better estimation accuracy only with the correct initialization, but it loses the convergence capability as the initial SoC is set with an erroneous value of more than 25 %. Conversely, SR-UKF shows a more stable performance, reaching the convergence with all the initialization values, and keeping an error almost always lower than 2 %. This conclusion is reinforced also by considering the comparable computational cost of the algorithms, with both of them requiring about 8 ms for performing one estimation step.

3.2 Real Time Optimal Balancing with a Binary PSO

3.2.1 Passive Balancing Architecture

The most used technique for performing cells balancing is the passive approach, in which the most charged cells are discharged on suitable shunt resistors. In this context, BMS is in charge of determining in real-time which is the optimal configuration of the shunt resistors in order to maximize the effectiveness of cells balancing.

An example of BESS equipped with a passive balancing hardware is shown in Figure 3.6. The battery pack is composed of the series of N cells ideally identical to each other. A shunt resistor R_s is connected in parallel to each cell and its activation is managed by means of a related MOSFET switch. Also the transistors and the shunt resistors are identical to each other for every cell. The activation of the switches is controlled through the voltage applied to the gate terminal of the transistors. Indeed, if the gate is connected to the low digital value, the transistor is interdicted and the switch is *OFF*. Conversely, when the gate is connected to the high digital value the transistor is *ON* and it behaves like a short circuit. Therefore, the state of the i^{th} switch can be modeled by means of a binary variable, namely s_i.

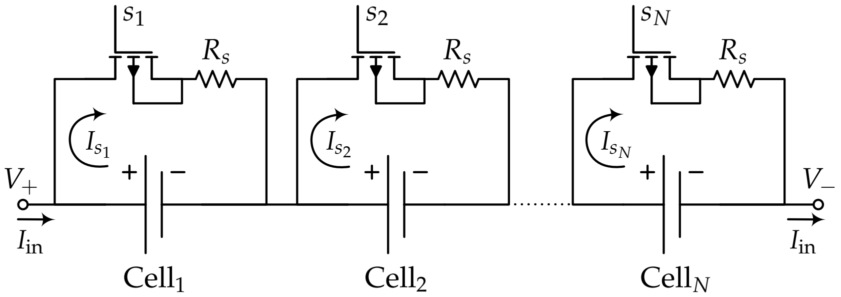


Figure 3.6: BESS equipped with a passive balancing system.

The circuit related to the balancing of the i^{th} cell is shown in Figure 3.7. Herein, I_{in} is the overall input current of the BESS, V_{out_i} is the terminal voltage of the cell, R_s is the shunt resistor, and s_i is the binary variable indicating with 1 and 0 the activation and the deactivation of the MOSFET, respectively.

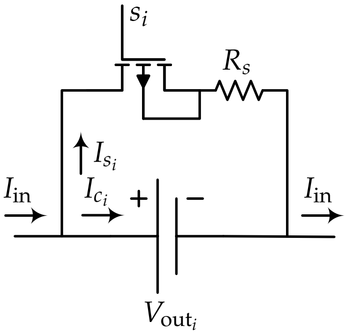


Figure 3.7: Balancing circuit related to the i^{th} cell.

The input current is split in two components, I_{c_i} that flows through the cell and I_{s_i} flowing through the shunt resistor R_s:

$$I_{\text{in}} = I_{c_i} + I_{s_i}. \tag{3.4}$$

When $s_i = 0$ the switch is *OFF* and the shunt resistor is electrically disconnected from the cell, resulting in $I_{s_i} = 0$ and $I_{c_i} = I_{\text{in}}$. Otherwise, when $s_i = 1$ the MOSFET is *ON* and R_s is electrically connected in parallel to the cell. Assuming that the voltage drop across the MOSFET is negligible, the current flowing through the resistor is $I_{s_i} = V_{\text{out}_i}/R_s$. Thus, the current I_{c_i} flowing through the cell can be expressed as a function of the binary variable s_i as follows:

$$I_{c_i} = I_{\text{in}} - \frac{V_{\text{out}_i}}{R_s} s_i. \tag{3.5}$$

It can be noticed that the balancing is performed by subtracting the current I_{s_i} from the cell. This means that an implicit loss of energy happens every time the shunt resistor is activated.

The aim of the balancing algorithm is to evaluate online which is the binary vector $\boldsymbol{s} = \{s_1, \dots, s_N\}$ that progressively brings all the cells working as much as possible in the same operating point. Moreover, the development of the balancing system needs to take into account also the implicit energy loss related to the activation of the shunt resistors. For this reason, it is desirable that the balancing task is performed by limiting the number of the active shunts in order to avoid excessive energy losses.

3.2.2 Proposed Balancing Algorithm

In the literature, cells balancing is typically performed by considering only the terminal voltage V_{out} and by leveling it among all the cells [79]–[81]. However, the terminal voltage is affected by several resistive and transient phenomena that weaken the correlation between the voltage of the cell and its operating point. Indeed, the internal resistance causes instantaneous voltage drop as a response to the input current and the internal charge redistribution results in the typical low pass transient response that any electrochemical cell exhibits in front of sharp variations of the input current. Moreover, many lithium cells show a flat OCV-SoC curve, resulting in a limited dependence of voltage on SoC in a wide range of operating points. For this reason, voltage based balancing algorithms can work only when the cells are in the high SoC range, such that the contribution of the OCV response is prevalent with respect to the resistive and transient phenomena. It is clear that both of the previous drawbacks can significantly reduce the effectiveness of the balancing algorithm. Moreover, because of the not optimal approach, typically voltage balancing is performed only during the charging phases (i.e. when $I_{\text{in}} > 0$) in order to limit the loss of dispatchable energy.

Aiming at overcoming the above discussed drawbacks, it has been developed a novel approach for determining in real-time the optimal configuration of the passive balancing hardware. Specifically, instead of considering the terminal voltage, the proposed method aims at leveling directly the operating points of the cells represented by their SoC. In order to do that, the balancing algorithm has been formulated as an optimization problem aiming at progressively minimizing the variance of the SoC of the cells. Thus, this approach is not affected by the parasitic voltage responses and it can be performed independently of the actual cells' SoC. Moreover, because of the optimal solution, the proposed approach can be performed both during the charging and discharging phases without facing an excessive loss of dispatchable energy thanks to a fast and optimal leveling of the cells.

The block diagram of the BMS including the proposed balancing system is shown in Figure 3.8. The architecture is composed of three main units: the slave device, the SoC estimator, and the balancer.

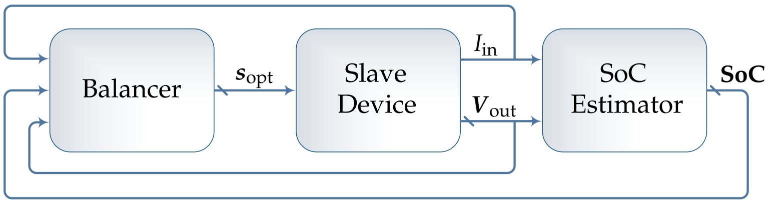


Figure 3.8: Block diagram of the BMS including the balancing system.

Slave device: It is the slave board of the BMS architecture including the electronic hardware devoted to the measurements of the cells' voltage

$\boldsymbol{V}_{\text{out}} = \{V_{\text{out}_1}, \ldots, V_{\text{out}_N}\}$ and of the input current I_{in}, as well as to the actuation of passive balancing.

SoC estimator: It performs the SoC estimation for each cell by means of the SR-UKF algorithm. The SoC estimator receives as input the measurements of the current I_{in} and of the cells' voltage $\boldsymbol{V}_{\text{out}}$, and it provides as output the SoC of each cell $\mathbf{SoC} = \{\text{SoC}_1, \ldots, \text{SoC}_N\}$.

Balancer: It evaluates at each time-step the optimal configuration of the switches $\boldsymbol{s}_{\text{opt}} = \{s_1, \ldots, s_N\}$ resulting in the best achievable leveling of the cells' SoC. The balancer receives as inputs the measurements of I_{in}, $\boldsymbol{V}_{\text{out}}$, and the vector of the estimated state of charges $\mathbf{SoC}$. It provides as output the best configuration $\boldsymbol{s}_{\text{opt}}$.

The balancing algorithm aims at evaluating at each instant k which is the optimal configuration $\boldsymbol{s}_{\text{opt}}$ resulting in the maximum achievable minimization of the SoC variance in the present time-step. In order to perform this task, it has been necessary to evaluate an $\boldsymbol{s}$ dependent prediction of the future SoC. Given the measurements $I_{\text{in}}[k]$ and $\boldsymbol{V}_{\text{out}}[k]$, as well as the SoC estimations $\mathbf{SoC}[k]$ provided by the SoC estimator, the predicted $\mathbf{SoC}^*[k+1]$ can be evaluated by applying the coloumb counting expression. Thus, the predicted SoC for the i^{th} cell can be written as follows:

$$\text{SoC}_i^*[k+1] = \text{SoC}_i[k] + \frac{T_s}{C_n} I_{c_i}[k]. \tag{3.6}$$

Expression (3.5) can be substituted in (3.6), obtaining:

$$\text{SoC}_i^*[k+1] = \text{SoC}_i[k] + \frac{T_s}{C_n}\left(I_{\text{in}}[k] - \frac{V_{\text{out}_i}[k]}{R_s} s_i\right). \tag{3.7}$$

Two terms can be identified in the above formula:

$$\begin{cases} a_i = \text{SoC}_i[k] + \dfrac{T_s}{C_n} I_{\text{in}}[k] \\ b_i = \dfrac{T_s}{C_n} \dfrac{V_{\text{out}_i}[k]}{R_s} s_i \end{cases} \tag{3.8}$$

where a_i is coincident with the future SoC without the presence of the shunt resistor, whereas b_i is the amount of charge subtracted from the cell due to the connection of R_s.

By assigning $\boldsymbol{a} = \{a_1, \ldots, a_N\}$ and B equal to the diagonal matrix whose i^{th} element is b_i, the predicted SoC can be rewritten as follows:

$$\mathbf{SoC}^*(\boldsymbol{s}) = a_i - B\boldsymbol{s} \tag{3.9}$$

where the term $\mathbf{SoC}^*[k+1]$ has been expressed as a function of the binary variables $\boldsymbol{s}$ and the temporal dependence has been neglected for simplicity.

Let $\mu^*(\boldsymbol{s})$ be the average of $\mathbf{SoC}^*[k+1]$, the unbiased variance of the predicted SoCs can be evaluated as follows:

$$\sigma^*_{\text{SoC}}(\boldsymbol{s}) = \frac{\left(\mathbf{SoC}^*(\boldsymbol{s}) - \mu^*(\boldsymbol{s})\right)^T \left(\mathbf{SoC}^*(\boldsymbol{s}) - \mu^*(\boldsymbol{s})\right)}{N-1} \tag{3.10}$$

where again the temporal dependence has been neglected for simplicity. Consequently, at each time-step k the balancer has to solve the following optimization problem in order to progressively minimize σ^*_{SoC}:

$$\begin{aligned} \boldsymbol{s}_{\text{opt}} = & \underset{\boldsymbol{s}}{\operatorname{argmin}}\ \sigma^*_{\text{SoC}}(\boldsymbol{s}) \\ \text{s.t.} \quad & \boldsymbol{s} \in \{0,1\}\,. \end{aligned} \tag{3.11}$$

The problem defined in (3.11) performs cells balancing without taking into account the amount of energy lost in the activation of the shunt resistors. Indeed, the best solution should achieve the maximum leveling of the cells by activating the minimum number of switches at the present time-step. This condition can be formalized by forcing the solution vector $\boldsymbol{s}_{\text{opt}}$ to be sparse. To this purpose, the ℓ_1 regularization [153] is a well known method for achieving sparse solutions from an optimization problem. Therefore, the ℓ_1 regularization has been included in the problem (3.11) in order to make the balancing algorithm more energetically efficient. The resulting ℓ_1 regularized version of problem (3.11) is formalized as follows:

$$\begin{aligned} \boldsymbol{s}_{\text{opt}} = & \underset{\boldsymbol{s}}{\operatorname{argmin}}\ \sigma^*_{\text{SoC}}(\boldsymbol{s}) + \lambda \sum |s_i| \\ \text{s.t.} \quad & \boldsymbol{s} \in \{0,1\} \end{aligned} \tag{3.12}$$

where λ is the regularization coefficient and the $|.|$ operator evaluates the absolute value.

Both the optimization problems defined in (3.11) and (3.12) belong to the zero-one integer programming. Therefore, it has been necessary to use optimization techniques able to deal with binary variables. In particular, being both (3.11) and (3.12) non-convex problems, heuristic methods must be considered. Thus, both of them have been solved by means of the MSHG-BPSO algorithm explained in Section 2.4.

3.2.3 Tests and Results

Experimental Design

The developed balancing algorithm has been validated by investigating its effectiveness at dealing with the main causes of an unbalanced BESS. The first source of a systematic unbalancing is related to the stochastic nature of the manufacturing process. Indeed, despite a specific electrochemical cell is declared to have a certain nominal capacity, each manufactured one is characterized by a slightly different C_n. Consequently, every cell composing the BESS is systematically dissimilar from the others and it has a likewise different charging/discharging rate in front of the same input current. Second, every cell is characterized by a different capacity fade related to the aging process. Thus, even if all the cells start from a quite close value of C_n, the aging effect causes a progressive divergence of their real capacity. Also in this case, the consequence is a different charging/discharging rate in front of the same input current. Finally, it is possible that the cells start working with different initial SoC values.

In order to allow a flexible testing of the proposed method in front of all the three unbalancing causes, the BESS has been simulated by using a suitable model of electrochemical cells. Specifically, the ECM discussed in Section 4.2.2 has been considered for this task. The simulated BESS has been implemented by considering the series connection of N models. Each of them was characterized by both a random value of the initial capacity and a random linear fading of it. These choices have aimed at emulating the randomness of the manufacturing process and of the aging effect, respectively. The initial capacity of each cell has been drawn from a normal distribution whose average value is the plate capacity C_n and the standard deviation is γ Ah. Moreover, the aging effect has been emulated with the following expression:

$$\begin{cases} \tilde{C}_{n_i}[0] = \mathcal{N}(C_n, \gamma) \\ \tilde{C}_{n_i}[k] = \tilde{C}_{n_i}[k-1] - \delta_i k \end{cases} \tag{3.13}$$

where $\tilde{C}_{n_i}[k]$ is the actual capacity of the i^{th} cell at the k^{th} time-step, $\mathcal{N}(.)$ is the normal distribution, and the fading rate δ_i is a constant value drawn from a uniform distribution. Furthermore, the SoC of each cell has been initialized with a random value drawn from a uniform distribution as well.

The balancing algorithm has been tested by simulating the use of the BESS in the powertrain of a pure electric vehicle. To this aim, the software *Advanced Vehicle Simulator* (ADVISOR) [154] has been considered for retrieving the charging/discharging current interesting the BESS during a simulated journey of the vehicle. In particular, ADVISOR has been set for emulating an electric vehicle similar to the *Nissan Leaf*, whose specifications are shown in Table 3.1.

Table 3.1: Vehicle configuration in ADVISOR

Powertrain	**Weight**	**Motor**	**ESS Energy**	**ESS Capacity**
Electric	2068 Kg	AC 83 KWh	30.6 KWh	85 Ah

The journey has been built by mixing urban and highway driving cycles from the *Federal Test Procedure* (FTP) repository with rests and battery charging phases. The obtained current profile has been firstly normalized with respect to the capacity of the BESS used in ADVISOR and successively re-scaled to the nominal capacity of the cell simulated with ECM. The list of the driving cycles composing the journey is shown in Table 3.2, whereas the resulting speed and charging profiles are shown in Figure 3.9.

Table 3.2: List of the driving cycles composing the journey

Name[*]	**Duration [s]**	**Distance [km]**
Rest	600	0
NREL2VAIL	4320	105.2
Rest	600	0
Charge	7200	0
Rest	600	0
UDDS	1370	12
US06	600	12.89
ARB02	1640	31.91
Rest	3600	0
Tot.	20530	162.00

[*] The acronyms refer to the driving cycles belonging to the FTP repository.

Experimental Results

The validation of the proposed balancing approach has been performed on a BESS counting $N = 10$ cells. In particular, ECM simulates the Li-ion cell model *18650* having nominal capacity of 2.1 Ah related to the *Randomized*

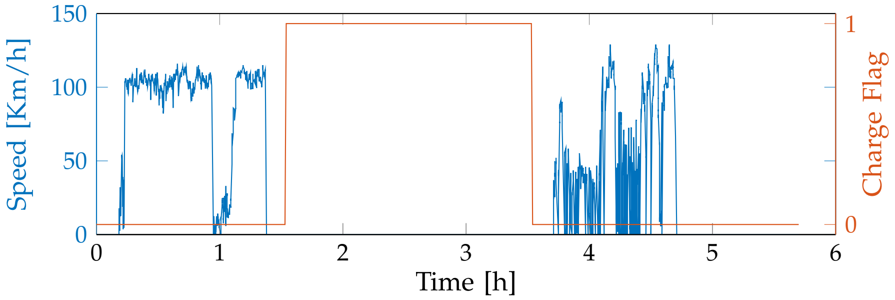


Figure 3.9: Speed and charging profiles used for testing the balancing algorithm.

Battery Usage Data Set collected by the *NASA Ames Research Center* [155]. ECM has been configured with 3 RC dipoles and the sampling time has been set to 1 s. Refer to Section 4.2.2 for more details about this cell model and the related system identification procedure.

The real capacities of the ten cells have been evaluated by applying expression (3.13) with $C_n = 2.1\,\text{Ah}$ and $\gamma = 0.1\,\text{Ah}$. The fading coefficient δ_i has been chosen randomly from the range $[0, 1e\text{-}5]$ in order to achieve a maximum capacity fade of 0.2 Ah along the 20530 time samples composing the simulated journey. The resulting capacities are shown in Figure 3.10. Moreover, the SoC of each cell has been randomly initialized in the range $[75\,\%, 100\,\%]$.

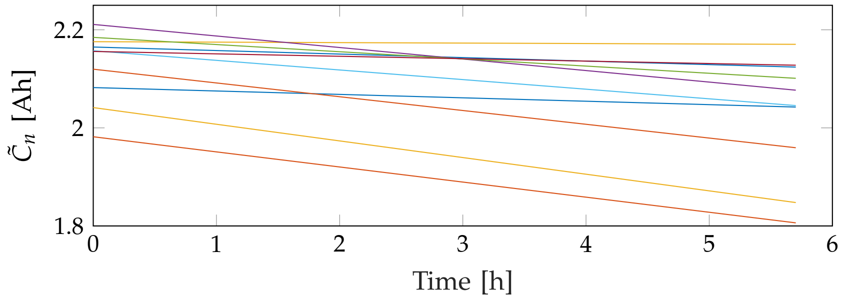


Figure 3.10: Simulated capacity fade for the ten cells composing the BESS.

SoC estimation has been performed with SR-UKF. In particular, ten state observers have been instantiated, each one performing SoC estimation upon a single cell of the BESS. All of the SoC estimators have been initialized with an SoC value of 50 %. In particular, the choice of initializing the SR-UKFs with a wrong SoC value has aimed at investigating if the effectiveness of the developed balancing algorithm is affected by erroneous SoC estimations, as well as if the influence of the balancer unit can affect the convergence capability of the SR-UKFs. Moreover, it is worth noting that both the SoC estimator and the balancer unit consider the plate capacity C_n for performing their tasks. This is because it is not possible to known online the real value of the cells capacity until specific offline tests are performed on them. Therefore, the only reasonable choice has been to consider the nominal capacity declared by the manufactured for performing both the SoC estimation and the balancing algorithms. As a consequence, SR-UKF can temporary lose the convergence from the real SoC, providing then erroneous estimations, as well as also the prediction of the future SoC performed by the balancer unit can be wrong. Thus, it will be interesting to verify the robustness of the proposed balancing algorithm with respect to the previous discussed issues.

The configuration of MSHG-BPSO is shown in Table 3.3, in which N_{pop}, N_{mut}, N_{crs}, G, and N_{itr} refer to the number of individuals, the number of mutations, the number of crossovers, the number of sub-swarms, and the maximum number of iterations, respectively. Specifically, the number of individuals and iterations have been chosen looking for the best trade-off between effectiveness and speed.

Table 3.3: MSHG-BPSO Configuration

N_{pop}	ω	c_p	c_g	N_{mut}	N_{crs}	K	G	N_{itr}
30	0.7298	1.4962	1.4962	3	4	6	1	30

The proposed balancing algorithm has been compared with a standard voltage based approach. More precisely, the procedure explained in reference [81] has been considered. In this balancing algorithm, only the cells having the terminal voltage V_{out_i} greater than a suitable threshold V_{th} are considered for being balanced. This is because voltage based methods can operate only with cells having a high SoC value, and the voltage threshold aims at selecting only these cells. Successively, the actual activation of the switches is conditioned to the voltage difference between the selected cells and those having a lower voltage value with respect to them. More precisely, the switch is activated only when the maximum of this voltage difference is greater than another suitable threshold ΔV_{th}. This procedure aims at ensuring a faster balancing of the BESS by activating the switches only for those cells having an operating point farther from the others. Summarizing, the voltage based balancing is formalized with the following logic condition:

$$s_i = V_{\text{out}_i} > V_{\text{th}} \quad \wedge \quad \max_{V_{\text{out}_j} < V_{\text{out}_i}} \left| V_{\text{out}_i} - V_{\text{out}_j} \right| > \Delta V_{\text{th}}. \tag{3.14}$$

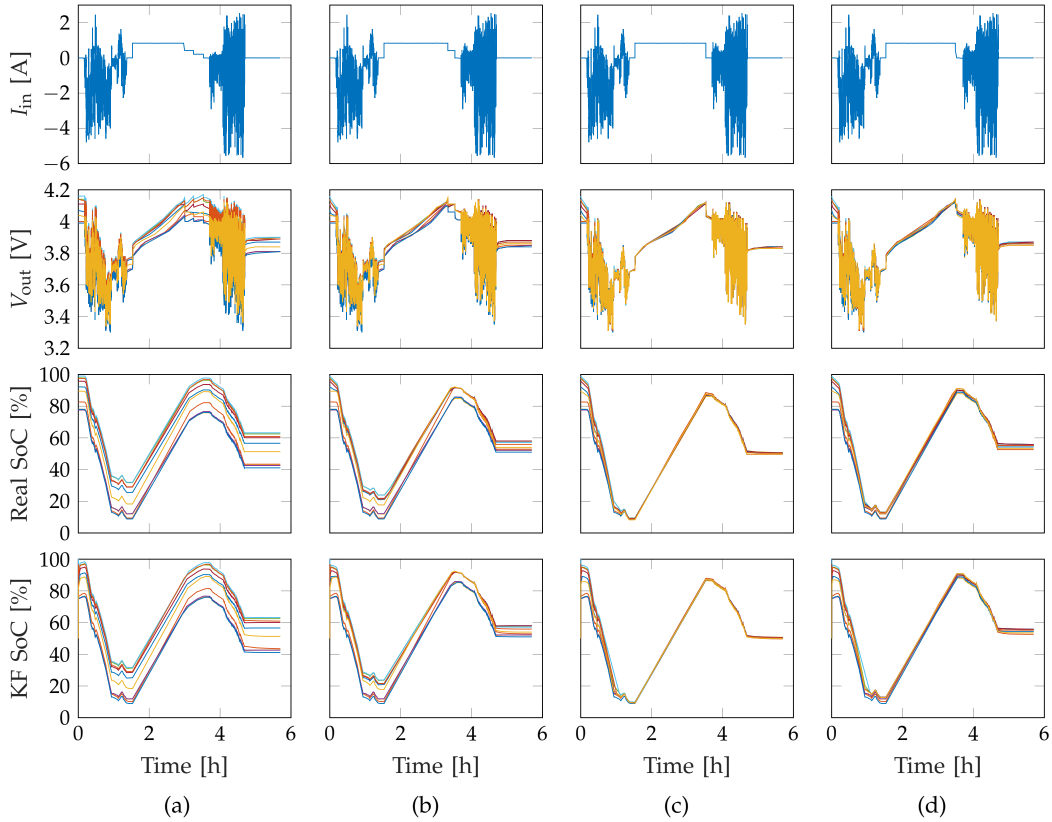


Figure 3.11: Actual applied current, terminal voltage, real and estimated SoC for different balancing algorithms. (a) No balancing. (b) Voltage balancing. (c) MSHG-BPSO balancing. (d) MSHG-BPSO balancing with ℓ_1 regularization.

Even if the considered voltage based approach should be performed only during the battery charging phases, the procedure has been applied independently of the sign of the input current in order to achieve a fair comparison with the proposed balancing approach.

The results achieved by four distinct balancing schemes are shown in Figure 3.11. Part (a) refers to a BMS without any kind of balancing system. Part (b) is related to the voltage based algorithm configured with $V_{\text{th}} = 3.8\,\text{V}$ and $\Delta V_{\text{th}} = 0.05\,\text{V}$. Parts (c) and (d) refer to the MSHG-BPSO algorithm solving the baseline problem (3.11) and the regularized one (3.12), respectively. With regards to the regularized approach, the coefficient λ has been set equal to $1e$-6. This value has proved to offer the best trade-off between an effective balancing of the cells and the energetic efficiency of the procedure. Indeed, the introduction of the ℓ_1 regularization limits the number of cells being balanced at the same time. Therefore, a too high value of λ will compromise the effectiveness of the balancing, whereas a too small value reduces the energetic efficiency of the procedure.

The results without the balancing system highlights the importance of performing an effective cells balancing. Indeed, it can be seen in Figure 3.11(a) that BMS was forced to limit the total charging current at the end of the charging phase. This is because some of the cells were close to the maximum allowed voltage (4.2 V) and it was necessary to limit the charging current for avoiding damages to them. Consequently, the entire BESS has received less charge than the expected one, despite that only few of the cells were close to the upper boundaries.

With regards to the balancing approaches, the MSHG-BPSO methods have achieved a more effective balancing with respect to the voltage based one. Indeed, MSHG-BPSO has succeeded at leveling all the cells in less than two hours considering both the baseline and the regularized problems. Moreover, the proposed approach was able to keep balanced the battery pack despite the effect of the random aging. It is remarkable that the effectiveness of the proposed balancing approach has not been affected neither by the wrong initial SoC values, nor by the erroneous SoC estimations caused by the difference between the actual cells' capacity and that used in SR-UKF. Furthermore, none of the SR-UKFs has shown convergence issues related to the effect of cells balancing. Conversely, the voltage based algorithm has reached a good balancing only for 7 of the 10 cells, as well as this method was not able to maintain the achieved balancing for the remainder of the test. These observations are confirmed also by analyzing the initial, final, and average differences between the highest and the lowest SoC values among the cells, as shown in Table 3.4.

Table 3.4: Initial, final, and average difference between the maximum and minimum SoC values among the cells

	No Bal.	**Voltage**	**MSHG-BPSO**	**MSHG-BPSO ℓ_1 Reg.**
Initial	21.4 %	21.4 %	21.4 %	21.4 %
Final	22.1 %	7.0 %	1.2 %	3.2 %
Average	22.2 %	10.7 %	3.8 %	5.2 %

Three metrics have been used for comparing the balancing performances. The first one is the unbiased variance of the actual state of charge $\tilde{\mathbf{SoC}}$ evaluated by means of expression (3.6) and considering the real capacity $\tilde{C}_{n_i}$ of the cells. The second metric is the unbiased variance of the cells' voltage. Finally, the third metric is the energy lost for the activation of the shunt resistors. The three metrics are formalized in the following expressions:

$$\sigma_{\text{SoC}}[k] = \frac{\left(\tilde{\mathbf{SoC}}[k] - \tilde{\mu}[k]\right)^T \left(\tilde{\mathbf{SoC}}[k] - \tilde{\mu}[k]\right)}{N-1} \tag{3.15}$$

$$\sigma_{V_{\text{out}}}[k] = \frac{\left(\boldsymbol{V}_{\text{out}}[k] - \nu[k]\right)^T \left(\boldsymbol{V}_{\text{out}}[k] - \nu[k]\right)}{N-1} \tag{3.16}$$

$$\mathcal{L}[k+1] = \mathcal{L}[k] + \sum_{i=1}^{N} s_i[k] \frac{V_{\text{out}_i}[k]^2}{R_s} T_s \tag{3.17}$$

where $\tilde{\mu}[k]$ and $\nu[k]$ are the average of the actual SoC and of the cells voltage evaluated at the k^{th} time-step, respectively. The energy loss $\mathcal{L}$ has been evaluated through the discrete integral of the instantaneous power dissipated upon the active shunt resistors. The three metrics evaluated for all the balancing schemes are shown in Figure 3.12.

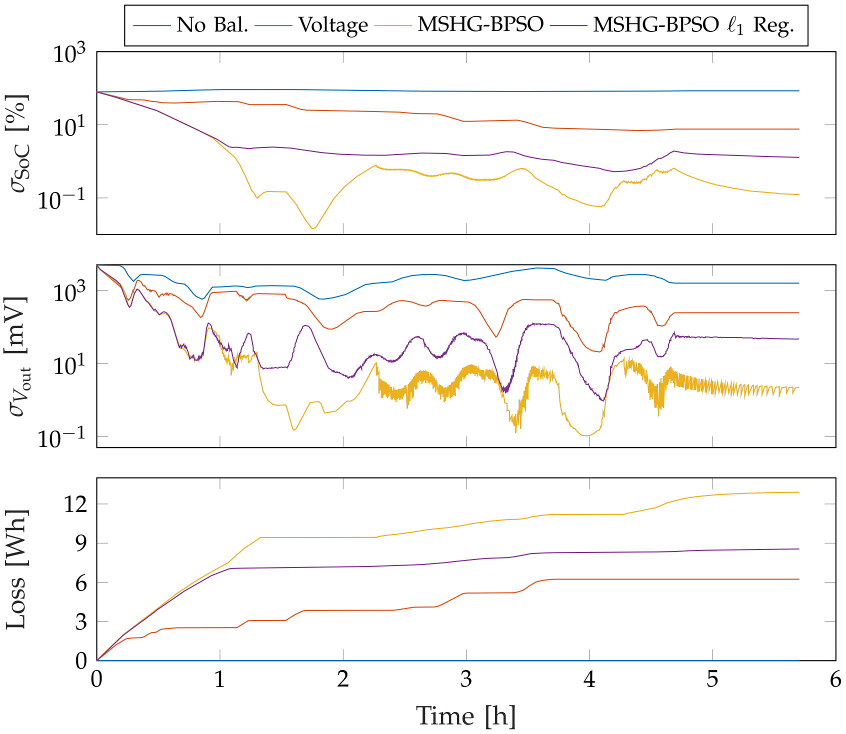


Figure 3.12: Comparison between the balancing algorithms in terms of SoC variance, voltage variance and total energy loss.

The MSHG-BPSO algorithm solving the baseline problem has achieved the best performances both in the SoC variance and voltage variance. Nevertheless, the better balancing of the BESS is counterbalanced by the highest energy losses. This is because the baseline problem (3.11) does not take into account the energy dissipated from the activation of the switches. Conversely, the performances of MSHG-BPSO applied to the regularized problem are remarkable. As expected, the introduction of the ℓ_1 regularization has reduced the balancing performances in front of a better energetic efficiency with respect to the baseline problem. Despite that, this method has achieved a better balancing with respect to the voltage based approach with a slightly higher and still comparable energy losses. More precisely, the total amount of energy dissipated by the activation of the shunt resistors is 6.24 Wh, 12.89 Wh, and 8.55 Wh for the voltage based balancing, the baseline MSHG-BPSO, and the regularized MSHG-BPSO, respectively.

The proposed approach is clearly more computationally demanding with respect to the voltage based balancing. Indeed, it solves an optimization problem at each time-step, whereas the voltage based method requires the evaluation of a single logic condition. Thus, it will be necessary to evaluate its suitability for being applied in real-time tasks.

Even if the MSHG-BPSO algorithm has achieved very interesting and effective results, in particular when solving the regularized problem, the proposed procedure will be clearly more advantageous considering an active balancing hardware. Indeed, in this case the entire system will be not affected by the energy losses due to the activation of the shunt resistors, and the MSHG-BPSO algorithm shall determine which cells must be charged, instead of discharged. Thus, future works will focus on the application of the proposed procedure to an active balancing system in order to further improve the effectiveness and efficiency of BESSs.

Chapter 4

Modeling Electrochemical Cells

4.1 Modeling Background

Any electrochemical cell is a nonlinear dynamical system whose behavior is affected by several physical quantities, namely the input current, the terminal voltage, the cell temperature, and the amount of stored charge. As any nonlinear dynamical system, also electrochemical cells can be described through the following discrete time state form:

$$\begin{aligned} \boldsymbol{x}[k+1] &= \mathcal{F}(\boldsymbol{x}[k], \boldsymbol{u}[k]) \\ \boldsymbol{y}[k] &= \mathcal{G}(\boldsymbol{x}[k], \boldsymbol{u}[k]) \end{aligned} \tag{4.1}$$

where $\boldsymbol{x}$, $\boldsymbol{u}$, and $\boldsymbol{y}$ are the internal states, the input variables, and the output variables vectors, respectively, $\mathcal{F}(.)$ and $\mathcal{G}(.)$ are the nonlinear functions defining the state update and the output equations, respectively.

In accordance with reference [51], the typical voltage response of any electrochemical cell due to a given input current can be thought as the superimposition of three main nonlinear contributions. Each of these distinct voltage responses is related to a different timescale, namely the instantaneous, the dynamic, and the quasi-stationary ones. The instantaneous response V_{ist} tracks the memoryless relationship between the input current and the terminal voltage due to the internal resistance. The dynamic contribution V_{dyn} tracks the voltage transient response related to the internal charge redistribution which exhibits a low pass behavior with respect to the input current. Finally, the quasi-stationary term V_{qst} is coincident with the OCV and it is the voltage contribution related to the amount of charge actually stored in the cell.

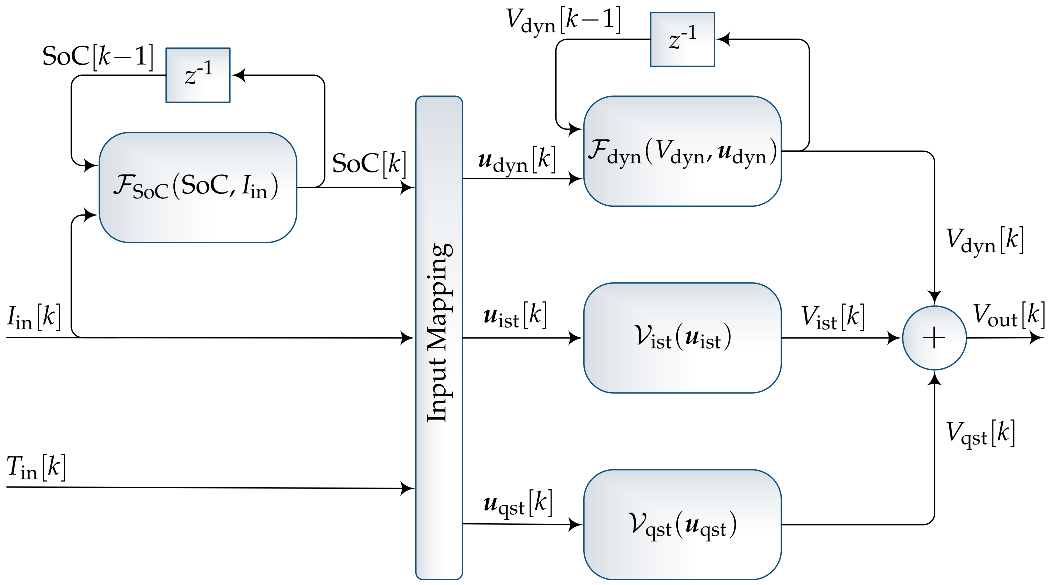
Because of the nonlinear electrochemical processes occurring during the charging/discharging phases, each of the internal voltage responses can have a potential nonlinear dependence on the current I_{in}, SoC, and the cell temperature T_{in}. Therefore, the canonical state form equations of electrochemical cells can be formalized as follows:

$$
\begin{aligned}
\text{SoC}[k+1] &= \mathcal{F}_{\text{SoC}}\left(\text{SoC}[k], I_{\text{in}}[k]\right) \\
V_{\text{dyn}}[k+1] &= \mathcal{F}_{\text{dyn}}\left(V_{\text{dyn}}[k], \text{SoC}[k], I_{\text{in}}[k], T_{\text{in}}[k]\right) \\
V_{\text{out}}[k] &= \mathcal{V}_{\text{qst}}\left(\text{SoC}[k], I_{\text{in}}[k], T_{\text{in}}[k]\right) + V_{\text{dyn}}[k] + \\
&\quad + \mathcal{V}_{\text{ist}}\left(\text{SoC}[k], I_{\text{in}}[k], T_{\text{in}}[k]\right)
\end{aligned}
\tag{4.2}
$$

where k is the temporal index, the output variable is the terminal voltage V_{out}, the state variables are SoC and V_{dyn}, and $\mathcal{F}_{\text{dyn}}(.)$, $\mathcal{V}_{\text{qst}}(.)$, and $\mathcal{V}_{\text{ist}}(.)$ are nonlinear functions.

The state update function $\mathcal{F}_{\text{SoC}}(.)$ is well defined by the coloumb counting algorithm. Therefore, $\mathcal{F}_{\text{SoC}}(.)$ is coincident with expression (1.3), so that it is dependent only on the previous SoC and the input current I_{in}. Thus, the temperature quantity has not been considered as input variable of $\mathcal{F}_{\text{SoC}}(.)$.

The block diagram corresponding to (4.2) is shown in Figure 4.1, where $\boldsymbol{u}_{\text{ist}}$, $\boldsymbol{u}_{\text{dyn}}$, and $\boldsymbol{u}_{\text{qst}}$ refer to the inputs related to the instantaneous, dynamic, and quasi-stationary responses, respectively. From a general point of view, these inputs are tuples composed of the physical quantities I_{in}, T_{in}, and SoC.



$\text{SoC}[k-1]$
z^{-1}
$\mathcal{F}_{\text{SoC}}(\text{SoC}, I_{\text{in}})$
$\text{SoC}[k]$
$I_{\text{in}}[k]$
$T_{\text{in}}[k]$
Input Mapping
$V_{\text{dyn}}[k-1]$
z^{-1}
$\boldsymbol{u}_{\text{dyn}}[k]$
$\mathcal{F}_{\text{dyn}}(V_{\text{dyn}}, \boldsymbol{u}_{\text{dyn}})$
$V_{\text{dyn}}[k]$
$\boldsymbol{u}_{\text{ist}}[k]$
$\mathcal{V}_{\text{ist}}(\boldsymbol{u}_{\text{ist}})$
$V_{\text{ist}}[k]$
+
$V_{\text{out}}[k]$
$V_{\text{qst}}[k]$
$\boldsymbol{u}_{\text{qst}}[k]$
$\mathcal{V}_{\text{qst}}(\boldsymbol{u}_{\text{qst}})$

Figure 4.1: Block diagram of the discrete system modeling electrochemical cells.

It has been largely discussed in reference [51] the importance of accurately identifying and separating the three main distinct voltage responses of electrochemical cells. Indeed, an effective separation and approximation of V_{ist},

V_{dyn}, and V_{qst} is critical for improving the model performances and accuracy. Moreover, it allows a physical interpretation of each distinct voltage response that is helpful for exposing the insights on the cell physics which importance has been discussed in Section 1.2.

4.2 White Box Technique

4.2.1 Nonlinear Equivalent Circuit Model

Model Background

The first developed model takes inspiration from the nonlinear ECM proposed in reference [51] and illustrated in Figure 4.2.

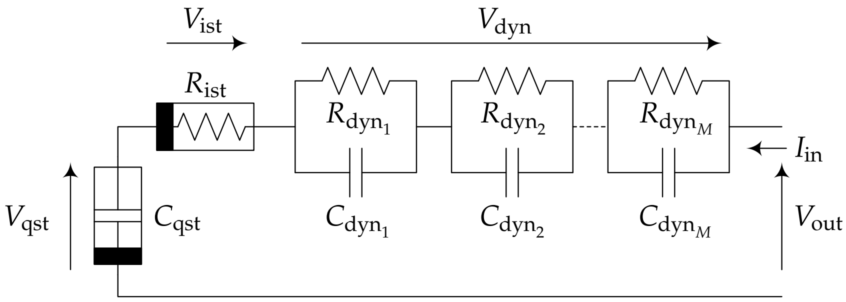


Figure 4.2: Nonlinear ECM proposed in [51].

This circuit aims at providing an ECM able to reflect the nonlinear physical behaviors of electrochemical cells. Thus, both the quasi-stationary and the instantaneous voltage responses are modeled by means of nonlinear electric components, respectively a nonlinear capacitor and a nonlinear resistor. In particular, the electrical behavior of C_{qst} is implemented by means of the OCV-SoC curve, whereas R_{ist} emulates the different resistive behavior that electrochemical cells expose in front of different current values, especially when opposite in sign. Conversely, the low-pass transient response related to V_{dyn} is modeled by means of the series connection of M parallel Resistor-Capacitors (RC) dipoles. The state form system equations of the model are expressed as follows:

$$
\begin{aligned}
\mathrm{SoC}[k+1] &= \mathrm{SoC}[k] + \frac{T_s}{C_n} I_{\mathrm{in}}[k] \\
V_{\mathrm{dyn}_1}[k+1] &= V_{\mathrm{dyn}_1}[k] e^{\frac{-T_s}{\tau_{\mathrm{dyn}_1}}} + R_{\mathrm{dyn}_1} I_{\mathrm{in}}[k] \left(1 - e^{\frac{-T_s}{\tau_{\mathrm{dyn}_1}}}\right) \\
\vdots \quad &= \quad \vdots \\
V_{\mathrm{dyn}_M}[k+1] &= V_{\mathrm{dyn}_M}[k] e^{\frac{-T_s}{\tau_{\mathrm{dyn}_M}}} + R_{\mathrm{dyn}_M} I_{\mathrm{in}}[k] \left(1 - e^{\frac{-T_s}{\tau_{\mathrm{dyn}_M}}}\right) \\
V_{\mathrm{out}}[k] &= \mathcal{V}_{qst}(\mathrm{SoC}[k]) + \sum_{i=1}^{M} V_{\mathrm{dyn}_i}[k] + \mathcal{R}_{\mathrm{ist}}(I_{\mathrm{in}}[k]) I_{\mathrm{in}}[k]
\end{aligned}
\tag{4.3}
$$

where the nonlinear function $\mathcal{V}_{\mathrm{qst}}(.)$ is coincident with the OCV-SoC curve used for modeling the nonlinear capacitor C_{qst}, whereas $\mathcal{R}_{\mathrm{ist}}(.)$ is the nonlinear function modeling the nonlinear resistor R_{ist}.

The system identification procedure is based on the analysis of the voltage response of the cell to the current profile shown in Figure 4.3.

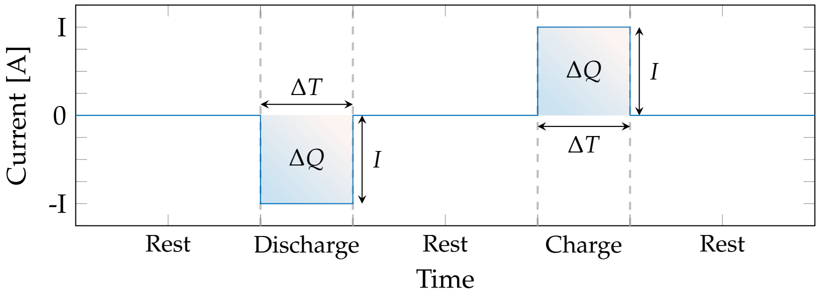


Figure 4.3: Current profile used for performing system identification upon the nonlinear ECM.

This current profile aims at exciting and exposing each of the distinct voltage responses in order to allow their proper identification. In particular, starting from a full charged cell, the procedure consists in a constant current discharge phase followed by a corresponding constant current charge. Both the discharge and the charge phases involve the application of a current pulse of the same width I and duration ΔT, so that the transferred charge is ΔQ. Moreover, rest periods are present at the beginning of the test, between the discharge and charge pulses, and at the end of the test in order to allow the transient responses to decay.

The identification procedure consists in three subsequent steps performed by analyzing the voltage response to the pulse edges, the rest periods, and the charging/discharging periods, respectively.

Edges: The first step consists in analyzing the voltage response to the edges of the current pulses in order to identify the instantaneous contribution. Indeed, the effect of the internal resistance causes a sudden voltage drop as a response to the current step. Therefore, it is possible to evaluate $\mathcal{R}_{\text{ist}}(I)$ as follows:

$$\mathcal{R}_{\text{ist}}(I) = \frac{\Delta V}{I} \tag{4.4}$$

where ΔV is the width of the voltage drop and I is the width of the current pulse. Moreover, it is possible to track the variation of $\mathcal{R}_{\text{ist}}(.)$ with respect to the input current by applying the same procedure using different current rates.

Rest Periods: Since $I_{\text{in}} = 0$ during the rest periods, then V_{qst} keeps constant and V_{ist} is zero as well. Therefore, only the dynamic contribution is active and it is possible to identify the components R_{dyn_i} and C_{dyn_i} as the parameters resulting in the best fitting of the transient voltage responses.

Discharge/Charge Pulses: All the three voltage responses are active during the charging/discharging periods. Nevertheless, it is possible to retrieve V_{qst} by difference, since both the instantaneous and the dynamic contributions have been estimated in the previous steps:

$$V_{\text{qst}}(t) = V_{\text{out}}(t) - V_{\text{dyn}}(t) - V_{\text{ist}}(t). \tag{4.5}$$

This procedure allows to estimate the OCV-SoC curve for both the charging and the discharging distinct processes. A unique $\mathcal{V}_{\text{qst}}(.)$ function is evaluated considering the average curve between the charging and discharging OCV-SoC curves.

Nonlinear RC Dipole for Modeling the Low-Pass Transient Response

The dynamic contribution plays a fundamental role at determining the global effectiveness of the whole cell model. Indeed, according to expression (4.5) V_{qst} can be accurately estimated only if the model provides an accurate approximation of the transient phenomena.

In the literature, the dynamic response is commonly modeled through the series connection of a suitable number of RC parallel dipoles [51], [86]–[88].

This solution is justified by the similarity between the measured transient response of the cell and the transient response of a low-pass filter. However, this assumption could be inadequate due to the intrinsic nonlinear nature of the device. In addition, there is not a systematic procedure to choose the number of RC dipoles. In fact, this choice is strictly dependent on the designer experience and it is performed looking for a good trade-off between complexity and accuracy. For example, in [88], [86], [156], [51], and [87] a single, two, two, three, and eight RC dipoles have been considered, respectively. In general, the model accuracy enhances with the increasing of the filter order, but this implies an increasing model complexity, as well as an increased state space dimension that results in an impracticable SoC estimation by means of Kalman filters. Furthermore, there may exist several sets of couples $R_{\text{dyn}_i}, C_{\text{dyn}_i}$ producing a good fitting of the same transient response with the same filter order M.

An alternative modeling of the transient response has been proposed aiming at solving the previous issues. In particular, the dynamic contribution has been modeled by means of a single nonlinear RC dipole, where it has been chosen to further use nonlinear electrical components for achieving a model closer to the nonlinear physical behavior of electrochemical cells. The proposed model is shown in Figure 4.4.

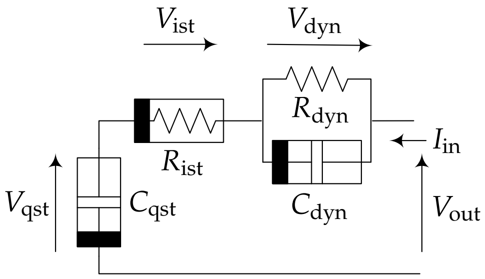


Figure 4.4: Proposed nonlinear ECM.

The modeling of V_{qst} and V_{ist} is coincident with that of Figure 4.2. Conversely, V_{dyn} is modeled by means of the parallel connection of one standard linear resistor and one nonlinear voltage driven capacitor. This solution allows to overcome the uncertainty in the number of RC filters because the model uses only one nonlinear dipole. As a consequence, the state space dimension decreases, allowing to reduce the computational cost of the SoC estimator. Moreover, the presence of only one dipole helps in making more robust the identification of R_{dyn} and C_{dyn}, since most likely there will be less local optimum solutions.

System Identification Procedure for the Nonlinear RC Dipole

Let Q_c and V_c be the stored charge and the terminal voltage of a nonlinear capacitor, respectively, the related constitutive equation is expressed as follows:

$$Q_c(t) = f(V_c(t)) \tag{4.6}$$

where t is the time variable and $f(.)$ is a generic nonlinear function. By evaluating the derivative with respect to time, expression (4.6) becomes:

$$\frac{dQ_c(t)}{dt} = \frac{df(V_c(t))}{dt} = \frac{df(V_c(t))}{dV_c}\frac{dV_c}{dt}. \tag{4.7}$$

Being $dQ_c(t)/dt$ coincident with the current I_c flowing through the capacitor and indicating with $\dot{V}_c(t)$ the time derivative of V_c, expression (4.7) can be rewritten as follows:

$$I_c(t) = \frac{df(V_c(t))}{dV_c}\dot{V}_c(t). \tag{4.8}$$

By comparing the above expression with the constitutive equation of a linear capacitor $I = C\dot{V}$, it is possible to define the nonlinear capacitor as follows:

$$\mathcal{C}(V_c(t)) = \frac{df(V_c(t))}{dV_c} \tag{4.9}$$

where $\mathcal{C}(.)$ is the value of the capacity expressed as a nonlinear function of the terminal voltage V_c.

The voltage response of the nonlinear RC dipole can be retrieved by mixing the constitutive equation of the nonlinear capacitor with that of the linear resistor. Let I_c, I_r, I_{in}, and V_{dyn} be the current flowing through C_{dyn}, the current flowing through R_{dyn}, the input current of the cell, and the voltage measured on the RC dipole, respectively, then:

$$\begin{aligned} I_{\text{in}}(t) &= I_c(t) + I_r(t) \\ I_{\text{in}}(t) &= \mathcal{C}_{\text{dyn}}(V_{\text{dyn}}(t))\dot{V}_{\text{dyn}}(t) + \frac{V_{\text{dyn}}(t)}{R_{\text{dyn}}} \\ \Rightarrow \dot{V}_{\text{dyn}}(t) &= \frac{I_{\text{in}}(t)}{\mathcal{C}_{\text{dyn}}(V_{\text{dyn}}(t))} - \frac{V_{\text{dyn}}(t)}{R_{\text{dyn}}\mathcal{C}_{\text{dyn}}(V_{\text{dyn}}(t))} \end{aligned} \tag{4.10}$$

and defining the characteristic time function of the nonlinear capacitor as follows:

$$\mathcal{T}_{\text{dyn}}\left(V_{\text{dyn}}(t)\right) = R_{\text{dyn}}\mathcal{C}_{\text{dyn}}\left(V_{\text{dyn}}(t)\right) \tag{4.11}$$

the governing equation of the nonlinear RC dipole is:

$$\dot{V}_{\text{dyn}}(t) = \frac{1}{\mathcal{T}_{\text{dyn}}\left(V_{\text{dyn}}(t)\right)}\left(R_{\text{dyn}} I_{\text{in}}(t) - V_{\text{dyn}}(t)\right) \tag{4.12}$$

whose discrete time version can be expressed as follows:

$$V_{\text{dyn}}[k+1] = V_{\text{dyn}}[k] e^{\frac{-T_s}{\mathcal{T}_{\text{dyn}}\left(V_{\text{dyn}}[k]\right)}} + R_{\text{dyn}} I_{\text{in}}[k]\left(1 - e^{\frac{-T_s}{\mathcal{T}_{\text{dyn}}\left(V_{\text{dyn}}[k]\right)}}\right). \tag{4.13}$$

According to expressions (4.12) and (4.13), the transient model is totally defined by the value of R_{dyn} and by the nonlinear function $\mathcal{T}_{\text{dyn}}(.)$. In order to estimate these parameters, the current profile shown in Figure 4.3 is applied to the cell and the rest periods are considered, as discussed in Section 4.2.1.

The value of R_{dyn} can be simply estimated as follows:

$$R_{\text{dyn}} = \frac{\Delta V_{\text{dyn}}}{I} \tag{4.14}$$

where $\Delta V_{\text{dyn}} = V_{\text{dyn}}^{\infty} - V_{\text{dyn}}^{0}$ is the total voltage drop of the rest period, with V_{dyn}^{0} and V_{dyn}^{∞} being the voltage at the beginning and at the end of the transient response, respectively.

The identification of the characteristic time function $\mathcal{T}_{\text{dyn}}(.)$ has been performed by applying a sampling and interpolating approach. In particular, this method uses a limited number L of samples equally distributed in the function domain as representatives of the nonlinear function to identify. Then, the L samples are interpolated with a cubic polynomial in order to recover the overall shape of the function. Thus, the identification procedure is in charge of finding only the values of the L samples, searching for the best sampling representation of $\mathcal{T}_{\text{dyn}}(.)$. A graphic example of the procedure is illustrated in Figure 4.5.

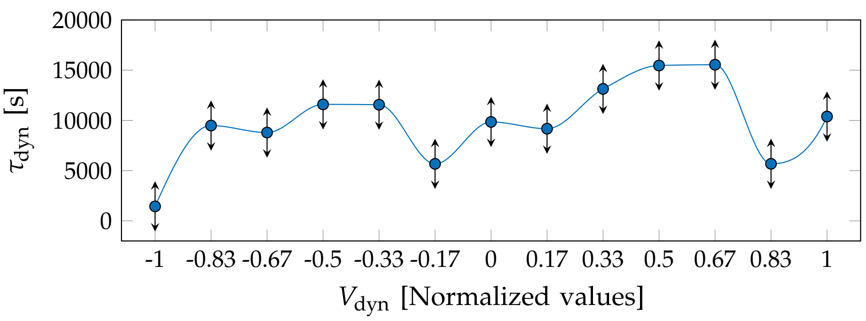


Figure 4.5: Sampling and interpolating approach for the identification of the nonlinear function $\mathcal{T}_{\text{dyn}}(.)$.

The identification of the L samples has been performed by means of the MSHG-PSO algorithm presented in Section 2.4. Herein, the position of each particle $\boldsymbol{\theta}$ is the vector having as components the samples of $\mathcal{T}_{\text{dyn}}(.)$:

$$\boldsymbol{\theta} = \left\{\tau_{\text{dyn}_1}, \ldots, \tau_{\text{dyn}_L}\right\} \tag{4.15}$$

where τ_{dyn_i} is the i^{th} sample of the $\mathcal{T}_{\text{dyn}}(.)$ function.

The fitness of each particle has been evaluated by first interpolating the samples τ_{dyn_i} and successively by evaluating the voltage transient response of the RC dipole by means of expression (4.13). More precisely, the fitness $\mathfrak{F}$ has been defined as a convex linear combination of two terms, representing respectively the estimation error ε and a penalty function ν:

$$\mathfrak{F}(\boldsymbol{\theta}) = \gamma\varepsilon + (1-\gamma)\nu \tag{4.16}$$

where $\gamma \in [0,1]$ sets the weight between ε and ν.

The estimation error ε has been evaluated by means of the Root Mean Squared Error (RMSE) between the estimated voltage response $\tilde{V}_{\text{dyn}}$ and the measured one V_{dyn}. RMSE has been evaluated both for the positive and the negative transient responses and the average value has been considered as follows:

$$\begin{aligned} RMSE^{+} &= \sqrt{\frac{1}{K}\sum_{k=1}^{K^{+}}\left(V_{\text{dyn}}^{+}[k] - \tilde{V}_{\text{dyn}}^{+}[k]\right)^{2}} \\ RMSE^{-} &= \sqrt{\frac{1}{K}\sum_{k=1}^{K^{-}}\left(V_{\text{dyn}}^{-}[k] - \tilde{V}_{\text{dyn}}^{-}[k]\right)^{2}} \\ \varepsilon &= \frac{RMSE^{+} + RMSE^{-}}{2} \end{aligned} \tag{4.17}$$

where the $+$ superscript refers to the positive transient, the $-$ superscript to the negative transient, and K is the number of time samples composing the transient responses.

The penalty function ν has been added in order to avoid that any component of the vector approximating the function $\mathcal{T}_{\text{dyn}}(.)$ assumes some inconsistent value. In particular, ν has been evaluated as follows:

$$\nu_i = \begin{cases} \mathrm{lb} - \tau_{\mathrm{dyn}_i} & \text{if} \quad \tau_{\mathrm{dyn}_i} < \mathrm{lb}, \\ \tau_{\mathrm{dyn}_i} - \mathrm{Ub} & \text{if} \quad \tau_{\mathrm{dyn}_i} > \mathrm{Ub}, \\ 0 & \text{otherwise} \end{cases} \tag{4.18}$$

$$\nu = \sum_{i=1}^{L} \nu_i$$

where lb and Ub are the minimum and the maximum admissible value for each samples τ_{dyn_i}, respectively. In particular, the lower bound lb was necessary because no value can be less or equal to zero. In fact, a negative or a zero value of $\mathcal{T}_{\mathrm{dyn}}(.)$ corresponds to an unstable system. On the other hand, the higher bound Ub has forced the characteristic time function under a reasonable value.

Performance Analysis

The proposed nonlinear ECM and the related system identification procedure have been tested on the same *A123 Nanophosphate AHR23113M1Ultra-B* cell used in Section 3.1. To this aim, the current profile of Figure 4.3 has been set with a charging/discharging pulse of width $I = 0.4\,\mathrm{A}$ and duration $\Delta T = 6\,\mathrm{h}$, resulting in a total transferred charge $\Delta Q = 2.4\,\mathrm{Ah}$. The voltage response of the cell is shown in Figure 4.6, where the value ΔV_{dyn} used for identifying R_{dyn} has been highlighted.

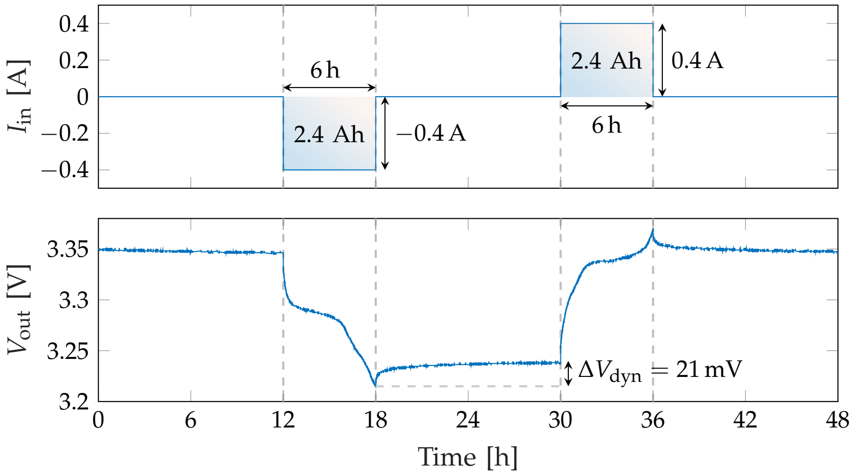


Figure 4.6: Voltage response of the *A123 Nanophosphate AHR23113M1Ultra-B* cell to the current profile used for the system identification procedure.

Before performing the estimation of $\mathcal{T}_{\text{dyn}}(.)$, both the positive and negative transient responses have been normalized with respect to ΔV_{dyn}:

$$\hat{V}_{\text{dyn}} = \frac{V_{\text{dyn}} - V^0_{\text{dyn}}}{\Delta V_{\text{dyn}}}. \tag{4.19}$$

This normalization has allowed to simplify and generalize the identification procedure, because it automatically transforms the negative transient in a positive one and it makes possible to apply the same procedure to different electrochemical technologies.

The configuration of the MSHG-PSO algorithm is summarized in Table 4.1, where it has been noted also the configuration of the parameters γ, lb, and Ub related to the evaluation of the fitness function.

Table 4.1: MSHG-PSO Configuration

N_{pop}	ω	c_p	c_g	N_{mut}	N_{crs}	K	G	N_{itr}	γ	**lb**	**Ub**
100	0.7298	1.4962	1.4962	3	4	10	5	1000	0.8	10	4e4

The identification procedure has been repeated considering different L values in order to retrieve the most suitable number of samples needed for approximating $\mathcal{T}_{\text{dyn}}(.)$. Precisely, the values 7, 9, 11, 13, 15, 17, 19, and 21 have been tested. The obtained functions $\mathcal{T}_{\text{dyn}}(.)$ are shown in Figure 4.7.

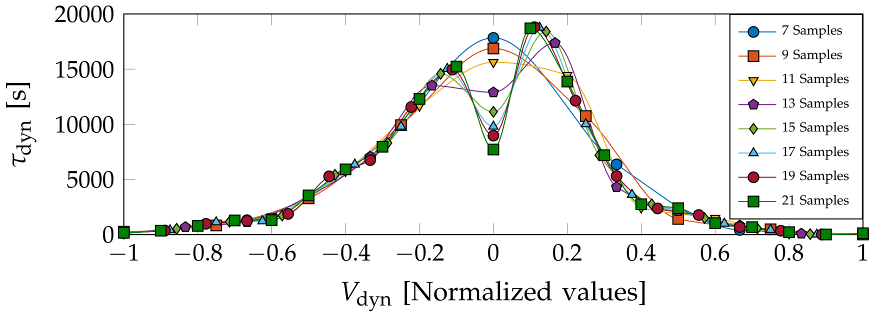


Figure 4.7: $\mathcal{T}_{\text{dyn}}(V_{\text{dyn}})$ function obtained considering different number of samples L.

Although the different resolution, the resulting functions are very similar to each other and all of them show an asymmetrical shape, meaning that the cell has a different response among the positive and the negative transients. The main differences occur at $V_{\text{dyn}} = 0$, most probably because the acquisition noise is more relevant for voltage values close to zero, where the signal to noise ratio becomes excessively small. Table 4.2 summarizes the best fitness achieved with the different number of samples and the related number of iterations to get convergence. The contiguity of the fitness values highlights that the differences between the several $\mathcal{T}_{\text{dyn}}(.)$ functions around $V_{\text{dyn}} = 0$ are not so significant for the final estimation accuracy. Therefore, it has been concluded that the best configuration counts $L = 13$ samples, because it has achieved the best trade-off between accuracy and time to convergence. The resulting $\mathcal{T}_{\text{dyn}}(.)$ function obtained with 13 samples is shown in Figure 4.8.

Table 4.2: Results obtained varying the number of samples

L	**Convergence at Iter**	$\mathfrak{F}$
7	160	2.15*e*-2
9	163	2.12*e*-2
11	197	2.07*e*-2
13	223	2.04*e*-2
15	422	2.03*e*-2
17	454	2.03*e*-2
19	727	2.02*e*-2
21	646	2.02*e*-2

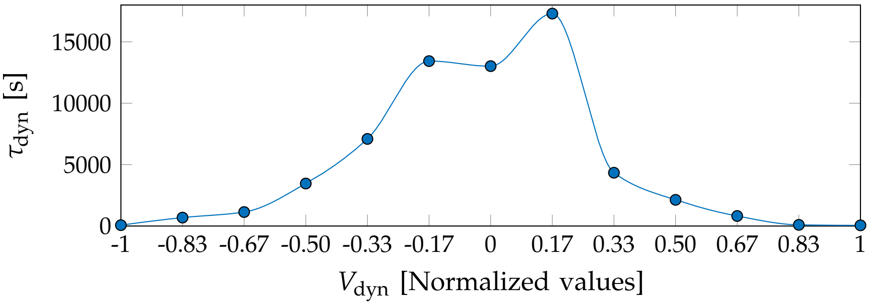


Figure 4.8: $\mathcal{T}_{\text{dyn}}(V_{\text{dyn}})$ function obtained by applying the identification procedure with $L = 13$ samples.

The performances of the proposed model have been compared with that of the ECM shown in Figure 4.2 in which V_{dyn} has been approximated considering 3 linear RC dipoles. In particular, the values of the resistors and capacitors have been retrieved by using again the MSHG-PSO algorithm. The obtained values are summarized in Table 4.3.

Table 4.3: Identified R_{dyn} and C_{dyn} values for the reference ECM

Group	R_{dyn} **[Ω]**	C_{dyn} **[F]**	τ_{dyn} **[s]**
First	1.97*e*-2	5.33*e*3	1.05*e*2
Second	1.66*e*-2	9.30*e*4	1.54*e*3
Third	1.17*e*-2	1.02*e*6	1.78*e*4

A first comparison between the linear and the nonlinear modeling of the dynamic contribution can be seen in Figure 4.9. Herein, the fitting performances of the two models during the rest periods are compared.

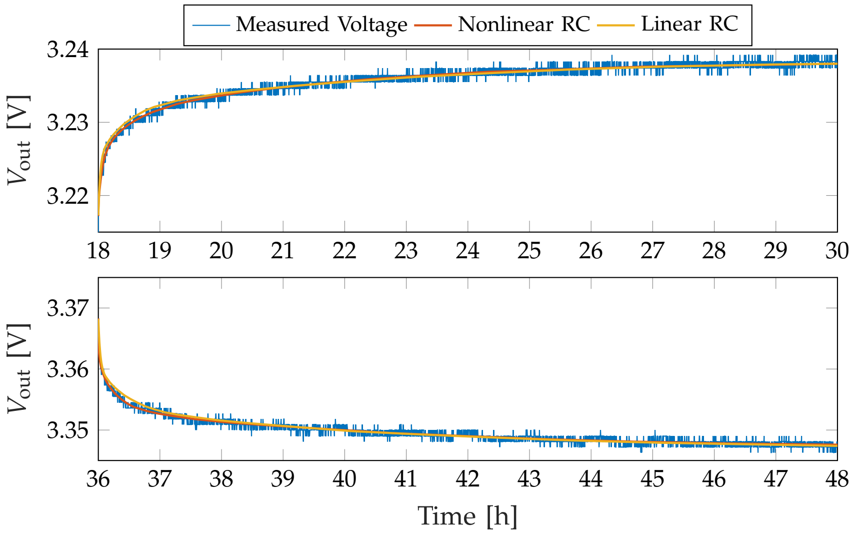


Figure 4.9: Comparison between the measured voltage and the voltage estimated by the linear and the nonlinear RC models.

It can be seen that the linear RC model is too fast in the first part of the positive transient and it is too slow in the first part of the negative transient. Conversely, the proposed nonlinear RC approach shows a better fitting in both the positive and the negative transient responses.

In order to consider a more realistic testing condition, the model performances have been analyzed upon the current profile shown in Figure 4.10 and derived from the US06 driving cycle test. This driving cycle has been chosen because it provides a highly dynamic current profile that widely excites the transient response of the dynamic contribution.

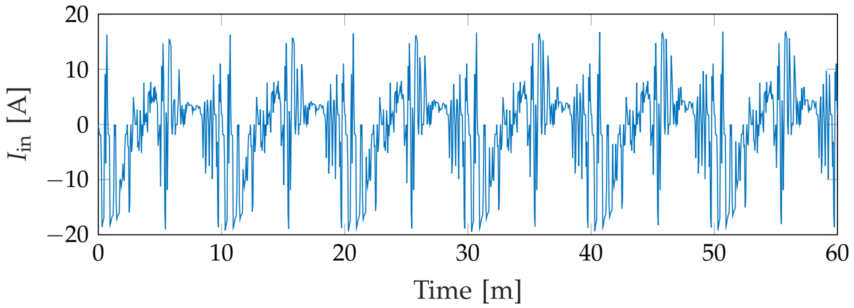


Figure 4.10: Current profile related to the US06 driving cycle.

Again the performance of the proposed ECM equipped with the nonlinear RC dipole has been compared with that of the ECM equipped with 3 linear RC groups. In particular, the percentage error committed at estimating the output voltage has been considered as performance metric and it has been evaluated as follows:

$$\text{Error}\,[\%] = 100\frac{\tilde{V}_{\text{out}} - V_{\text{out}}}{V_{\text{out}}} \tag{4.20}$$

where V_{out} is the measured voltage and $\tilde{V}_{\text{out}}$ is the estimated one.

The performances of the two models have been analyzed considering the application of the US06 driving cycle when the cell has two different initial conditions, specifically $\text{SoC}[0] = 60\,\%$ and $\text{SoC}[0] = 100\,\%$[1]. The obtained results are shown in Figure 4.11.

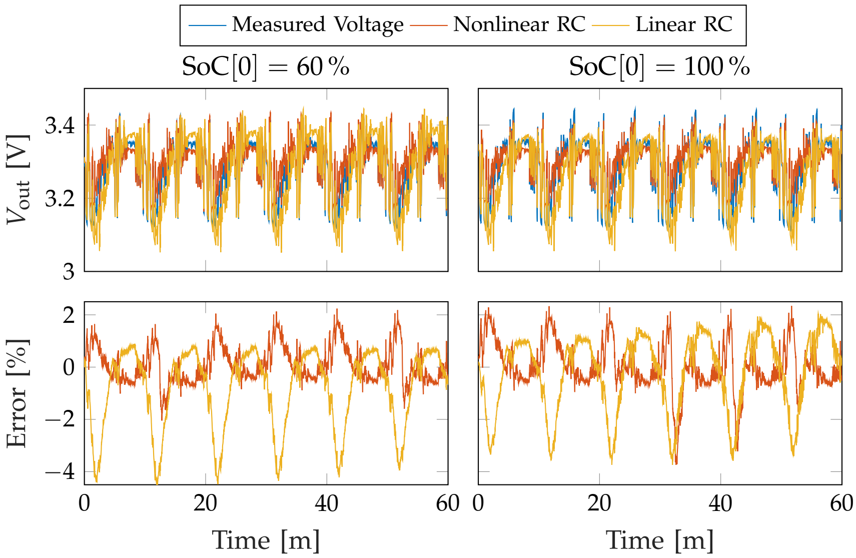


Figure 4.11: Voltage estimation accuracy of the ECMs equipped with the nonlinear RC dipole and the three linear RC dipoles.

The proposed model has obtained the best results considering both the initial SoC values. Indeed, it has kept a percentage error generally limited in the range of $[-1\,\%, 2\,\%]$, with only two error peaks at $-4\,\%$ for the test with $\text{SoC}[0] = 100\,\%$. Conversely, the ECM with linear RC dipoles has shown a repetitive error peak over the $-3\,\%$. The same conclusions can be drawn by analyzing the overall RMSE performed by the two models and summarized in Table 4.4.

[1] The measurement campaign has been performed by Dr. Gian Luca Storti from the *Center for Automotive Research* (CAR), *The Ohio State University*, Columbus, Ohio, USA.

Table 4.4: Comparison between the resulting RMSE

Model	**RMSE** $\mathrm{SoC}[0] = 60\,\%$	**RMSE** $\mathrm{SoC}[0] = 100\,\%$
3 Linear RC Dipole	3.74*e*-2	4.44*e*-2
Nonlinear RC Dipole	2.99*e*-2	2.89*e*-2

4.2.2 Linear Equivalent Circuit Model

Flexible Identification Procedure

The nonlinear ECM described in the previous section has shown very promising performances in terms of model accuracy and SoC estimation (see Section 3.1), together with a simple and efficient architecture and the ability of tracking the nonlinearities of electrochemical cells. Despite that, the main drawback is the complex and rigid system identification procedure. Indeed, it implies to test the cell with a very specific and long lasting current profile. As a consequence, the entire procedure can last some days in order to allow the identification of the parameters related to the slowest transient responses. Moreover, it is necessary to repeat the test for different current rates and different temperatures aiming at characterizing all the operating conditions of the cell. Thus, this identification procedure is very rigid, time consuming and expensive to perform, because it requires to disassemble the cell from the BESS and to test it offline using expensive testing equipment, such as climatic chambers and high precision sensors.

From the above discussion, it is clear that ECMs can be suitable models for performing SoC estimation, but it is necessary to make system identification simpler and more flexible. To this aim, some alternative solutions have been proposed in references [157] and [158], where system identification has been performed by fitting the datasheet curves of the cell by means of a Gray Wolf Optimization (GWO) [159] and a PSO, respectively. Nevertheless, these approaches can be ineffective since most likely electrochemical cells will work differently with respect to their datasheet curves because of manufacturing deviations or aging. Therefore, it has been developed a novel flexible identification procedure for ECMs. Similarly to [157] and [158], system identification has been formulated as a fitting problem solved by means of swarm intelligence algorithms. However, the proposed method searches for the best parameters set resulting in the best fitting of the voltage response measured during the use of the cell, instead of considering the generic information of the datasheet curves. Consequently, this approach results in a more flexible identification procedure that does not require any specific test and that is tailored on the specific cell upon which the procedure is applied, bringing to a more accurate modeling of its functionality.

In order to achieve this improved flexibility, it has been necessary to linearize most of the ECM discussed in Section 4.2.1. First, it has been considered again the architecture of Figure 4.2 in which V_{dyn} is modeled by means of the series of M RC linear dipoles. Second, the modeling of V_{ist} has been simplified by using a linear resistor in place of the nonlinear one. The resulting ECM is shown in Figure 4.12 and the related system equations are summarized as follows:

$$\begin{aligned}
\text{SoC}[k+1] &= \text{SoC}[k] + \frac{T_s}{C_n} I_{\text{in}}[k] \\
V_{\text{dyn}_1}[k+1] &= V_{\text{dyn}_1}[k] e^{\frac{-T_s}{\tau_{\text{dyn}_1}}} + R_{\text{dyn}_1} I_{\text{in}}[k] \left(1 - e^{\frac{-T_s}{\tau_{\text{dyn}_1}}}\right) \\
\vdots \quad &= \quad \vdots \\
V_{\text{dyn}_M}[k+1] &= V_{\text{dyn}_M}[k] e^{\frac{-T_s}{\tau_{\text{dyn}_M}}} + R_{\text{dyn}_M} I_{\text{in}}[k] \left(1 - e^{\frac{-T_s}{\tau_{\text{dyn}_M}}}\right) \\
V_{\text{out}}[k] &= \mathcal{V}_{\text{qst}}(\text{SoC}[k]) + \sum_{i=1}^{M} V_{\text{dyn}_i}[k] + R_{\text{ist}} I_{\text{in}}[k]
\end{aligned} \tag{4.21}$$

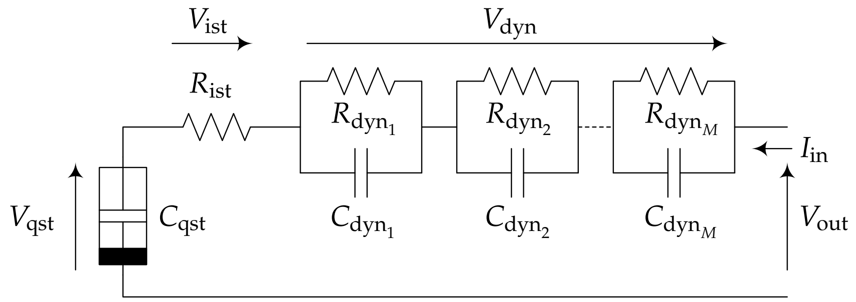


Figure 4.12: Circuit diagram of the linearized ECM.

Because of the linearization process, all the model parameters are real numbers, except for those of the quasi-stationary contribution. In fact, the identification of the OCV-SoC curve is more complicated since it requires to retrieve the shape of the nonlinear function $\mathcal{V}_{\text{qst}}(.)$. To this purpose, it has been considered again the sampling and interpolating procedure discussed in Section 4.2.1 that has proved to achieve effective results. Thus, the identification procedure is in charge of finding the parameters set $\boldsymbol{\theta}$ being coincident with the following vector:

$$\boldsymbol{\theta} = \left\{ R_{\mathrm{ist}}, \tau_{\mathrm{dyn}_1}, R_{\mathrm{dyn}_1}, \ldots, \tau_{\mathrm{dyn}_M}, R_{\mathrm{dyn}_M}, V_{\mathrm{qst}_1}, \ldots, V_{\mathrm{qst}_L} \right\} \tag{4.22}$$

where V_{qst_i} is the i^{th} sample of the OCV-SoC curve representation.

As explained in the above, system identification has been formulated as a fitting problem to be applied on generic data measured during the use of the cell. Thus, let V_{out} and I_{in} be any sequence of the measured voltage and current, respectively, the identification algorithm aims at finding the optimal parameters set $\boldsymbol{\theta}_{\mathrm{opt}}$ that minimizes the error between the estimated voltage and the measured one. Consequently, the identification procedure consists in solving the following optimization problem:

$$\boldsymbol{\theta}_{\mathrm{opt}} = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \left\{ \mathrm{RMSE}\left(V_{\mathrm{out}}, \tilde{V}_{\mathrm{out}}^{\boldsymbol{\theta}} \right) \right\} \tag{4.23}$$

where $\tilde{V}_{\mathrm{out}}^{\boldsymbol{\theta}}$ is the estimated voltage given the parameters set $\boldsymbol{\theta}$, and $\mathrm{RMSE}(.)$ is the root mean squared error function evaluated between V_{out} and $\tilde{V}_{\mathrm{out}}^{\boldsymbol{\theta}}$. Again, the MSHG-PSO algorithm has been used for solving the optimization problem (4.23).

Performance Analysis

The proposed method has been validated by applying the defined system identification procedure upon a real Li-ion cell. In order to highlight the achieved flexibility, the *Randomized Battery Usage Data Set* [155] collected by the *NASA Ames Research Center* has been considered for building the Training Set and the Test Set sequences. This data set is composed of the measurements performed over a 2.1 Ah Li-ion cell model *18650* tested with a randomly generated sequences of charging/discharging pulses. Starting form a full charged cell in a stationary condition (i.e. SoC = 100 % and $V_{\mathrm{dyn}} = 0$), each pulse of the sequence is randomly selected from the set $\{-4.5\,\mathrm{A}, -3.75\,\mathrm{A}, -3\,\mathrm{A}, -2.25\,\mathrm{A}, -1.5\,\mathrm{A}, -0.75\,\mathrm{A}, 0.75\,\mathrm{A}, 1.5\,\mathrm{A}, 2.25\,\mathrm{A}, 3\,\mathrm{A}, 3.75\,\mathrm{A}, 4.5\,\mathrm{A}\}$, and it is applied to the cell until either the terminal voltage goes outside the range $[3.2\,\mathrm{V}, 4.2\,\mathrm{V}]$ or 5 minutes have passed. In particular, the input current I_{in}, the terminal voltage V_{out}, and the surface temperature T_{in} have been measured during the test with a sampling time of 1 s. The SoC sequence has been retrieved by applying the coloumb counting algorithm considering $C_n = 2.1$ Ah. The Training Set includes the first 100 pulses of the first random cycling test performed on the cell ID *RW9*. Similarly, the Test Set is composed of the first 100 pulses related to the second random cycling test of the same cell. The sequences of I_{in}, V_{out}, T_{in}, and SoC of the NASA data set are shown in Figure 4.13. Since the simplified ECM does not consider the effect of the temperature, T_{in} has been ignored in this work.

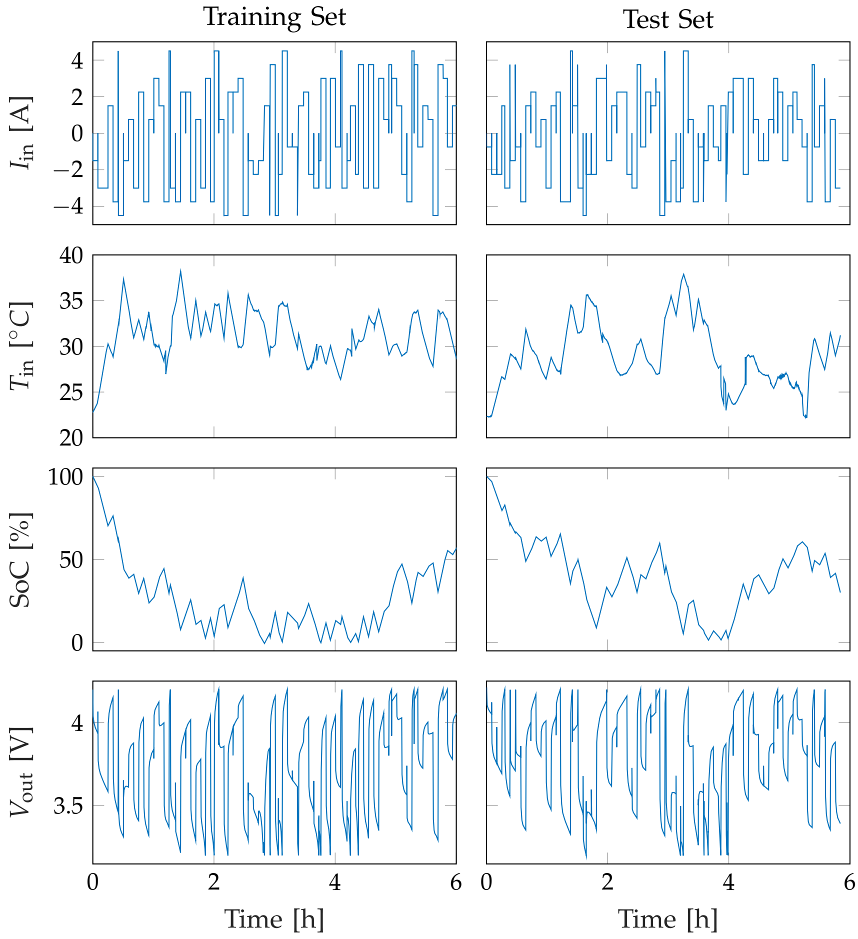


Figure 4.13: Data set related to the*Randomized Battery Usage Data Set*.

In order to analyze the robustness of the proposed identification procedure, the entire process has been performed 100 times considering a different random initialization of MSHG-PSO at each run. Then, a statistical analysis of the obtained results has been performed considering the values of the 1st, 2nd, and 3rd quartiles; note that the 2nd quartile coincides with the median value. The ECM model has been configured with $M = 3$ RC dipoles, and the identification of the nonlinear function $\mathcal{V}_{\text{qst}}(.)$ has been performed considering $L = 15$ and a total of 1000 interpolation points. The configuration of MSHG-PSO is summarized in Table 4.5.

Table 4.6 collects the results and the related statistical analysis concerning the RMSEs performed at estimating the output voltage of both the Training and the Test Sets sequences, as well as the results related to the identified

Table 4.5: MSHG-PSO Configuration

N_{pop}	ω	c_p	c_g	N_{mut}	N_{crs}	K	G	N_{itr}
50	0.7298	1.4962	1.4962	5	4	12	2	1000

Table 4.6: System identification results and statistical analysis

	Median	**Q1**	**Q3**	**Q3-Q1**
RMSE Training	1.56*e*-2	1.56*e*-2	1.56*e*-2	2.70*e*-5
RMSE Test	2.76*e*-2	2.76*e*-2	2.76*e*-2	7.91*e*-5
R_{ist} **[mΩ]**	79.09	78.99	79.16	0.16
R_{dyn_1} **[mΩ]**	20.36	20.25	20.49	0.24
τ_{dyn_1} **[h]**	4.93*e*-3	4.82*e*-3	5.05*e*-3	2.28*e*-4
R_{dyn_2} **[mΩ]**	21.22	21.13	21.31	0.18
τ_{dyn_2} **[h]**	8.35*e*-2	8.23*e*-2	8.57*e*-2	3.45*e*-3
R_{dyn_3} **[mΩ]**	24.00	13.75	75.20	61.46
τ_{dyn_3} **[mΩ]**	6.62	4.16	10.94	6.78

Q1: 1st quartile; Q3: 3rd quartile; Q3-Q1: Difference between 3rd and 1st quartile

values of R_{ist}, R_{dyn_i} and τ_{dyn_i}. The retrieved $\mathcal{V}_{\text{qst}}(.)$ function is shown in Figure 4.14.

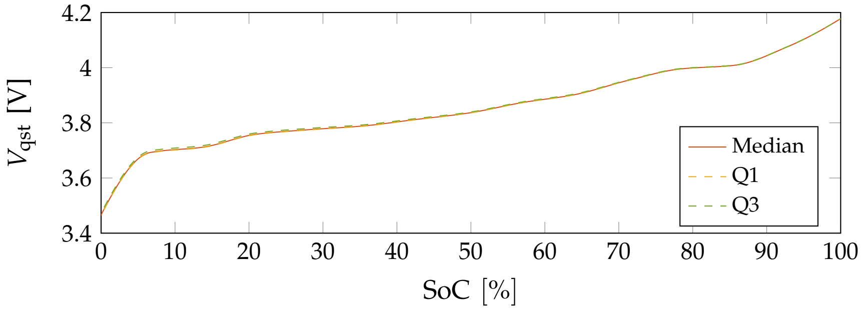


Figure 4.14: Function $\mathcal{V}_{\text{qst}}(.)$ identified by MSHG-PSO.

The effectiveness of the proposed flexible system identification procedure is largely proved by the obtained results. Indeed, ECM has achieved a great accuracy, performing a median RMSE of 1.56*e*-2 and 2.76*e*-2 for the Training Set and the Test Set, respectively. Moreover, the similar performances between Training and Test sets proves the good generalization capability of the proposed identification method. It is clear also the robustness of the procedure. Indeed, the difference between the 3rd and the 1st quartiles is always at least one order of magnitude lower than the median value for almost any of the circuital parameters, meaning that MSHG-PSO converges every time to very close solutions. The only exception is the third RC dipole, most probably because the time constant τ_{dyn_3} is comparable with the entire length of the Training Set sequence, making more uncertain its identification.

In addition to the robustness analysis, all the 100 identified models have been implemented in an SR-UKF for performing SoC estimation upon the Test Set data. In order to avoid a trivial test in which the dynamical states of the cell are known, SoC estimation has been started with a temporal offset of one hour. This way, when SR-UKF is started the cell is in a non-stationary and unknown dynamical condition. The obtained estimated sequences of SoC and V_{out} are shown in Figure 4.15, where SR-UKF has been initialized considering a generic state with SoC $= 50\,\%$ and $V_{\text{dyn}_i} = 0$ for each RC dipole.

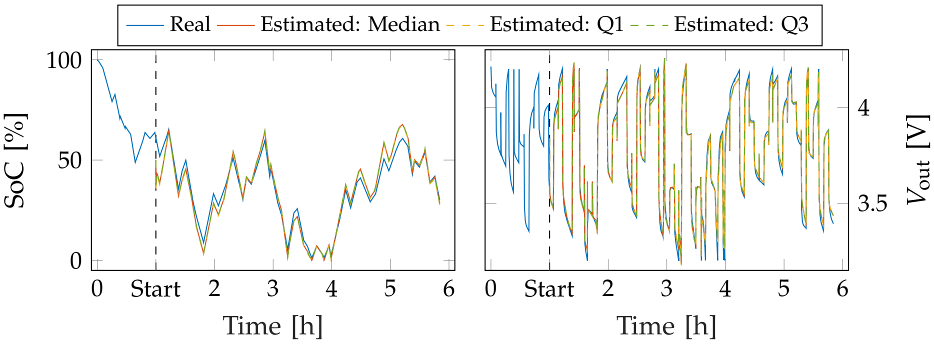


Figure 4.15: SoC and V_{out} estimations performed by the models identified with MSHG-PSO. SoC estimation has been performed by means of SR-UKF.

It can be seen that SR-UKF succeeds at correcting the state estimation in few minutes, achieving then an accurate SoC estimation along the entire length of the Test Set sequence. In particular, it has performed a median RMSE between the real SoC and the estimated one equal to 4.06*e*-2. Moreover, it is noticeable also the robustness of the estimation with the median almost always overlapped with both the 1^{st} and the 3^{rd} quartiles evaluated on the 100 estimated SoC sequences.

4.2.3 Mechanical Inspired Equivalent Circuit Model

Mechanical Analogy

Both the linear and the nonlinear ECMs discussed in the previous sections use coloumb counting as SoC state update equation, as well as the OCV-SoC curve is used for retrieving the electric response of the nonlinear capacitor C_{qst}. Nevertheless, an alternative approach has been proposed in reference [51] by introducing a mechanical analogy between storing of charge in electrochemical cells and storing of liquid in tanks.

From an engineering point of view, both electrochemical cells and liquid tanks can be seen as a suitable reservoir characterized by two main measurable quantities. The first one is related to the amount of charge/liquid flowing inside or outside the reservoir, whereas the second quantity is related to the amount of charge/liquid stored in it. Specifically, the flowing quantity is the input current for electrochemical cells, that is equivalent to the liquid flow rate for tanks. The quantity related to the amount of stored charge/liquid is given by the terminal voltage for cells and the liquid level for tanks. More precisely, the liquid level is typically measured through the pressure measured at the bottom of the tank.

The mechanical analogy concerns also the decomposition of the terminal voltage V_{out} in the distinct voltage responses V_{ist}, V_{dyn}, and V_{qst}. Indeed, as the instantaneous and the dynamic contributions are parasitic responses of the cell to the input current, similarly the total liquid pressure P_{out} is affected by parasitic effects related to the presence of waves on the liquid surface. Thus, P_{out} can be split in a quasi-stationary term P_{qst} related to the actual stored liquid, and a dynamic term P_{dyn} related to the waves effect. A graphical representation of the mechanical analogy is shown in Figure 4.16.

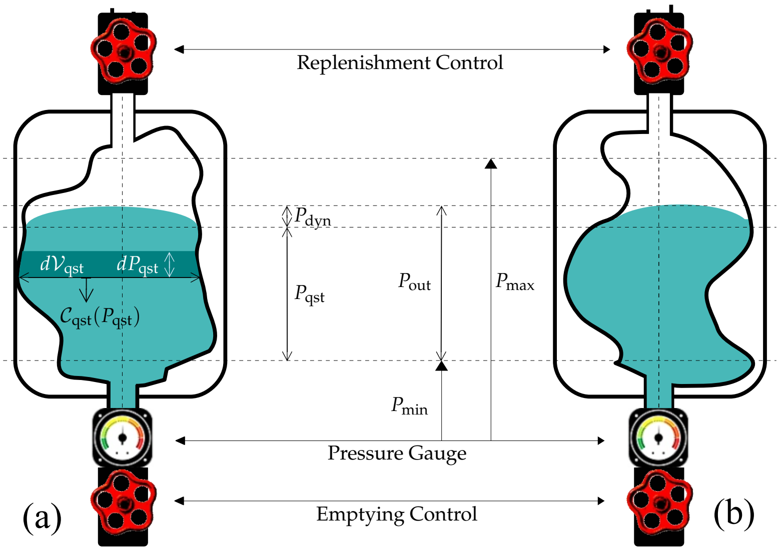


Figure 4.16: Mechanical analogy of electrochemical cells with a liquid reservoir.

It is clear that the shape of the reservoir must be known in order to estimate the amount of stored quantity. Indeed, as shown in Figure 4.16, two reservoirs having a different shape can store a different amount of liquid even if their measured quantities are the same. For this reason, usually liquid tanks are manufactured with a known and regular shape (typically cylindrical), so that it is straightforward to relate the liquid level to the stored one. Unfortunately, the same process cannot be applied for electrochemical cells. Indeed, the nonlinear chemical processes cannot be controlled to produce regular and perfectly known reservoir shapes. Therefore, it is necessary to determine how the amount of liquid/charge can be related to the measured pressure/voltage when the reservoir has a generic and irregular shape.

With regards to Figure 4.16(a), it is possible to define the shape of the tank by means of a nonlinear function $\mathcal{C}_{\text{qst}}(.)$ of the quasi-stationary pressure. Indicating with $\mathcal{V}(P_{\text{qst}})$ the liquid volume expressed as a function of P_{qst}, the shape of the tank can be evaluated as the derivative of $\mathcal{V}(.)$ with respect to the quasi-stationary pressure:

$$\mathcal{C}_{\text{qst}}(P_{\text{qst}}) = \frac{d\mathcal{V}(P_{\text{qst}})}{dP_{\text{qst}}}. \tag{4.24}$$

Consequently, the amount of stored liquid can be retrieved from the generic shape $\mathcal{C}(.)$ by evaluating:

$$\mathcal{V}(P_{\text{qst}}) = \int_{P_{\text{min}}}^{P_{\text{qst}}} \mathcal{C}_{\text{qst}}(p)dp, \quad P_{\text{qst}} \in [P_{\text{min}}, P_{\text{max}}] \tag{4.25}$$

where P_{max} and P_{min} are the maximum and the minimum allowed stationary pressure in the considered tank, respectively.

The previous discussion can be extended to any electrochemical cell independently of the manufacturing technology. Remembering that the liquid pressure and the liquid volume are equivalent to the cell voltage and the amount of stored charge, respectively, the shape of the charge reservoir is modeled with the following expression:

$$\mathcal{C}_{\text{qst}}(V_{\text{qst}}) = \frac{d\mathcal{Q}(V_{\text{qst}})}{dV_{\text{qst}}} \tag{4.26}$$

where $\mathcal{Q}(.)$ is the actual stored charge expressed as a nonlinear function of the quasi-stationary voltage V_{qst}. Thus, if the reservoir shape $\mathcal{C}_{\text{qst}}(.)$ is known, it is possible to evaluate the residual stored charge by means of the following integral:

$$\mathcal{Q}(V_{\text{qst}}) = \int_{V_{\text{min}}}^{V_{\text{qst}}} \mathcal{C}_{\text{qst}}(v)dv, \quad V_{\text{qst}} \in [V_{\text{min}}, V_{\text{max}}] \tag{4.27}$$

where V_{max} and V_{min} are the maximum and the minimum allowed stationary voltage, respectively.

Thanks to the mechanical analogy, it is possible to introduce an alternative definition of SoC with respect to the coloumb counting one. Indeed, SoC can be obtained by normalizing expression (4.27) with respect to the total amount of charge that the cell can store. Specifically, this quantity is now well defined and it is evaluated as follows:

$$Q_{\text{max}} = \mathcal{Q}(V_{\text{qst}})\Big|_{V_{\text{max}}} = \int_{V_{\text{min}}}^{V_{\text{max}}} \mathcal{C}_{\text{qst}}(v)dv. \tag{4.28}$$

Therefore, SoC is evaluated by means of the following expression:

$$\text{SoC}(V_{\text{qst}}) = \frac{1}{Q_{\text{max}}} \int_{V_{\text{min}}}^{V_{\text{qst}}} \mathcal{C}_{\text{qst}}(v)dv. \tag{4.29}$$

This new definition overcomes the drawbacks related to the coloumb counting approach. Indeed, expression (4.29) takes into account a physical characteristic of the cell, namely the shape of the reservoir defined by the nonlinear function $\mathcal{C}_{\text{qst}}(.)$. Consequently, it does not require the knowledge of the initial SoC value for being applied correctly. Moreover, this formulation is not anymore based on current measurements, resulting in an easier and more accurate definition. Finally, all the parameters Q_{max}, V_{max}, and V_{min} are well defined.

By comparing expression (4.26) with expression (4.9) it can be noticed that both of them model the same physical phenomenon, being coincident with the electric response of a nonlinear capacitor. Therefore, it is possible to use the mechanical analogy for defining a linear ECM like that of Figure 4.12, in which the electrical behavior of the nonlinear capacitor C_{qst} is modeled by means of the reservoir shape function $\mathcal{C}_{\text{qst}}(.)$, instead of the OCV-SoC curve. Thus, the state form equations of the Mechanical inspired ECM (M-ECM) are formalized as follows:

$$
\begin{aligned}
V_{\text{qst}}[k+1] &= V_{\text{qst}}[k] + T_s \frac{I_{\text{in}}[k]}{\mathcal{C}_{\text{qst}}(V_{\text{qst}}[k])} \\
V_{\text{dyn}_1}[k+1] &= V_{\text{dyn}_1}[k] e^{\frac{-T_s}{\tau_{\text{dyn}_1}}} + R_{\text{dyn}_1} I_{\text{in}}[k] \left(1 - e^{\frac{-T_s}{\tau_{\text{dyn}_1}}}\right) \\
\vdots \quad &= \quad \vdots \\
V_{\text{dyn}_M}[k+1] &= V_{\text{dyn}_M}[k] e^{\frac{-T_s}{\tau_{\text{dyn}_M}}} + R_{\text{dyn}_M} I_{\text{in}}[k] \left(1 - e^{\frac{-T_s}{\tau_{\text{dyn}_M}}}\right) \\
V_{\text{out}}[k] &= V_{\text{qst}} + \sum_{i=1}^{M} V_{\text{dyn}_i}[k] + R_{\text{ist}} I_{\text{in}}[k]
\end{aligned}
\tag{4.30}
$$

where it can be noticed that the coloumb counting equation has been substituted with the state update equation related to the nonlinear capacitor.

System Identification

The same flexible system identification procedure discussed in Section 4.2.2 has been maintained also for M-ECM. Thus, the procedure has been formulated as a fitting problem to be performed on generic sequences of measured current and voltage. In particular, system identification aims again at identifying the values of R_{ist}, R_{dyn_i}, and τ_{dyn_i} for $i = 1, \ldots, M$, as well as at retrieving the shape of the nonlinear function $\mathcal{C}_{\text{qst}}(.)$.

If the representation of R_{ist}, R_{dyn_i}, and τ_{dyn_i} is straightforward because they are all real numbers, modeling the nonlinear function $\mathcal{C}_{\text{qst}}(.)$ is more complicated and it requires the implementation of a suitable function estimator. To this aim, it has been used an ANFIS approximator (Section 2.3), whose block diagram for the specific application is shown in Figure 4.17.

The architecture counts one input and one output, corresponding to the quasi-stationary voltage V_{qst} and the value of the nonlinear capacitor C_{qst}, respectively. The fuzzy reasoning has been implemented by means of a first order *Takagi-Sugeno* FIS. In particular, each rule antecedent has been implemented by gaussian membership functions, whereas the overall output of the FIS is computed with the AVG method. Therefore, the aim is to estimate the reservoir shape $\mathcal{C}_{\text{qst}}(.)$ by means of a fuzzy piecewise linear approximation.

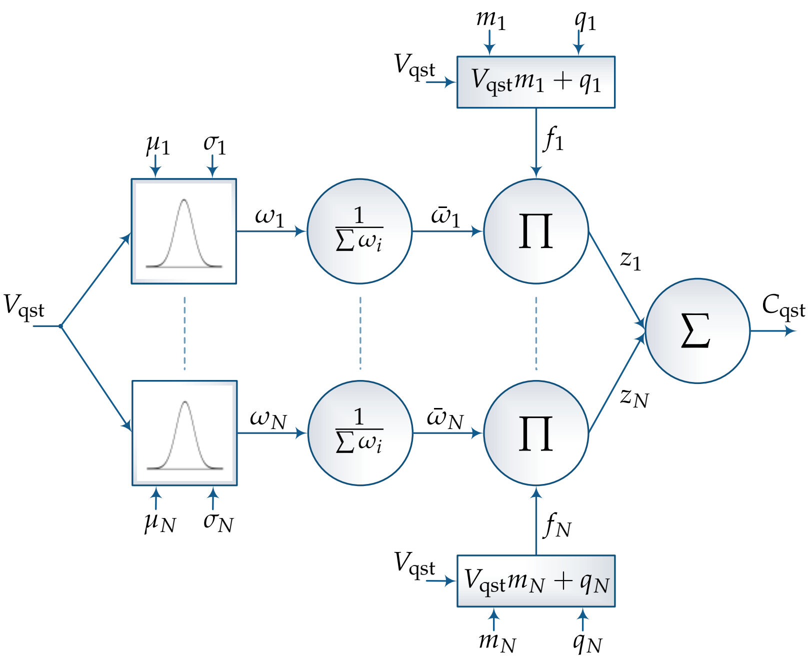


Figure 4.17: Architecture of the ANFIS used for approximating the reservoir shape $\mathcal{C}_{\text{qst}}(.)$.

The ANFIS is characterized by a total of N rules defined as follows:

$$\begin{array}{llll} \text{if} & V_{\text{qst}} \in A_1 & \text{then} & C_{\text{qst}} = f_1 = V_{\text{qst}}m_1 + q_1 \\ \text{if} & V_{\text{qst}} \in A_2 & \text{then} & C_{\text{qst}} = f_2 = V_{\text{qst}}m_2 + q_2 \\ & \vdots & & \vdots \\ \text{if} & V_{\text{qst}} \in A_N & \text{then} & C_{\text{qst}} = f_N = V_{\text{qst}}m_N + q_N \end{array} \tag{4.31}$$

where A_i is the i^{th} fuzzy set defined by the univariate gaussian function centered in μ_i and characterized by a σ_i deviation. In conclusion, let ω_i be the fire strength of the i^{th} rule, the output of the ANFIS is given by the following expression:

$$\mathcal{C}_{\text{qst}}(V_{\text{qst}}) = \sum_{i=1}^{N} \bar{\omega}_i \left(V_{\text{qst}}m_i + q_i \right) \tag{4.32}$$

where $\bar{\omega}_i = \omega_i / \sum \omega_j$.

The model is therefore represented by $4N + 2M + 1$ real numbers given by the following vector:

$$\boldsymbol{\theta} = \left\{ R_{\text{ist}}, \boldsymbol{R}_{\text{dyn}}, \boldsymbol{\tau}_{\text{dyn}}, \boldsymbol{\mu}_{\text{qst}}, \boldsymbol{\sigma}_{\text{qst}}, \boldsymbol{m}_{\text{qst}}, \boldsymbol{q}_{\text{qst}} \right\} \tag{4.33}$$

where $\boldsymbol{R}_{\text{dyn}}$ and $\boldsymbol{\tau}_{\text{dyn}}$ are the vectors of the M dynamic resistances and dynamic time constants, respectively, whereas $\boldsymbol{\mu}_{\text{qst}}$, $\boldsymbol{\sigma}_{\text{qst}}$, $\boldsymbol{m}_{\text{qst}}$, and $\boldsymbol{q}_{\text{qst}}$ are the vectors of the mean values, standard deviations, slopes, and intercepts related to the N rules of the ANFIS, respectively. In particular, system identification has been performed again by solving the optimization problem (4.23) by means of the MSHG-PSO algorithm.

It is clear that MSHG-PSO is in charge not only of retrieving the parameters R_{ist}, $\boldsymbol{R}_{\mathrm{dyn}}$, and $\boldsymbol{\tau}_{\mathrm{dyn}}$, but also of training the ANFIS for estimating the charge reservoir shape. However, it should be noticed that this training procedure is unsupervised since the function $\mathcal{C}_{\mathrm{qst}}(.)$ is not known at training time. Moreover, also the input of the ANFIS is unknown because the quasi-stationary contribution V_{qst} cannot be measured during the cell activity, being one of the internal dynamical states of the system. Despite that, the ANFIS can be trained in an effective way because the dynamical nature of the state update equation related to V_{qst} allows to retrieve the evolution of the quasi-stationary voltage response starting from the knowledge of the initial V_{qst} value. The advantage with respect to the coloumb counting equation is that the initial value $V_{\mathrm{qst}}[0]$ can be retrieved in an easier and more accurate way with respect to $\mathrm{SoC}[0]$. Indeed, the quasi-stationary contribution V_{qst} is by definition coincident with the terminal voltage when all the transient phenomena have decayed. Thus, it is sufficient to rest the cell at $I_{\mathrm{in}} = 0$ for a suitable long period in order to measure accurately the initial V_{qst} value, independently of the actual SoC of the cell.

Performance Analysis

Experimental Setup The performances of the M-ECM model and of the related system identification procedure have been analyzed again considering the *Randomized Battery Usage Data Set* shown in Figure 4.13. Herein, both I_{in} and V_{out} have been normalized in order to facilitate MSHG-PSO at performing the training of the ANFIS, as well as at identifying the other model parameters. Specifically, I_{in} has been normalized with respect to its maximum absolute value measured in the Training Set sequence. This is because it is necessary to keep unmodified the sign and the zero of the input current for preserving the information about the internal resistance and the transient phenomena. The voltage V_{out} has been scaled in the range $[0, 1]$ by normalizing it with respect to its maximum and minimum values measured in the Training Set sequence. The maximum allowed V_{qst} has been set to 4.2 V because this is the typical maximum value used in the Constant Current-Constant Voltage charging procedure for lithium-ion cells. The minimum

V_{qst} has been set to 3.3 V for having a reasonable margin with respect to the lower threshold of the procedure used for building the data set.

Concerning the number of RC dipoles, the model has been configured again with $M = 3$. Conversely, some preliminary tests have been performed for tuning the ANFIS estimator modeling $\mathcal{C}_{\text{qst}}(.)$. The best solution has been achieved by using $N = 15$ fuzzy rules. Moreover, it has been imposed a maximum admissible value for the deviation σ_i in order to avoid an excessive overlapping of the fuzzy sets. In particular, this constraint has been imposed by modifying the fitness function introducing a suitable penalty function as follows:

$$\begin{gathered}\nu_j = \begin{cases} \sigma_j - \sigma_{\max}, & \text{if } \sigma_j > \sigma_{\max} \\ 0, & \text{otherwise} \end{cases}, \quad j = 1, \dots, N \\ \mathfrak{F}(\boldsymbol{\theta}) = RMSE(V_{\text{out}}^{\boldsymbol{\theta}}, V_{\text{out}}) + \sum_{j=1}^{N} \nu_j \end{gathered} \tag{4.34}$$

where ν_j is a penalty term and $\sigma_{\max}$ is the maximum allowed standard deviation. Specifically, a maximum value of 0.5 has been considered. Finally, the complete configuration of MSHG-PSO is shown in Table 4.7.

Table 4.7: MSHG-PSO Configuration

N_{pop}	ω	c_p	c_g	N_{mut}	N_{crs}	K	G	N_{itr}
75	0.7298	1.4962	1.4962	7	4	18	2	1000

All the model parameters have been initialized with a random uniform distribution evaluated in the range $[0, 1]$, except for those related to the standard deviations, that have been initialized in the range $[0, 0.5]$, and to the slopes $\boldsymbol{m}_{\text{qst}}$ and intercepts $\boldsymbol{q}_{\text{qst}}$, that have been initialized with a gaussian distribution having zero mean and unitary standard deviation.

System Identification Results The properties of the identified charge reservoir are shown in Figure 4.18.

The effectiveness of the ANFIS training is proved by the comparison between the estimated and the real OCV-SoC curves shown in Figure 4.18(c). Specifically, the real curve has been obtained from a dedicated experimental measurement performed over the cell. It can be seen that MSHG-PSO succeeds at finding a suitable set of the parameters $\boldsymbol{\mu}_{\text{qst}}$, $\boldsymbol{\sigma}_{\text{qst}}$, $\boldsymbol{m}_{\text{qst}}$, and $\boldsymbol{q}_{\text{qst}}$ resulting in an almost perfect overlapping between the estimated OCV-SoC curve and the real one. A further effectiveness proof of the ANFIS training can be found in Figure 4.18(b). This curve is coincident with the function

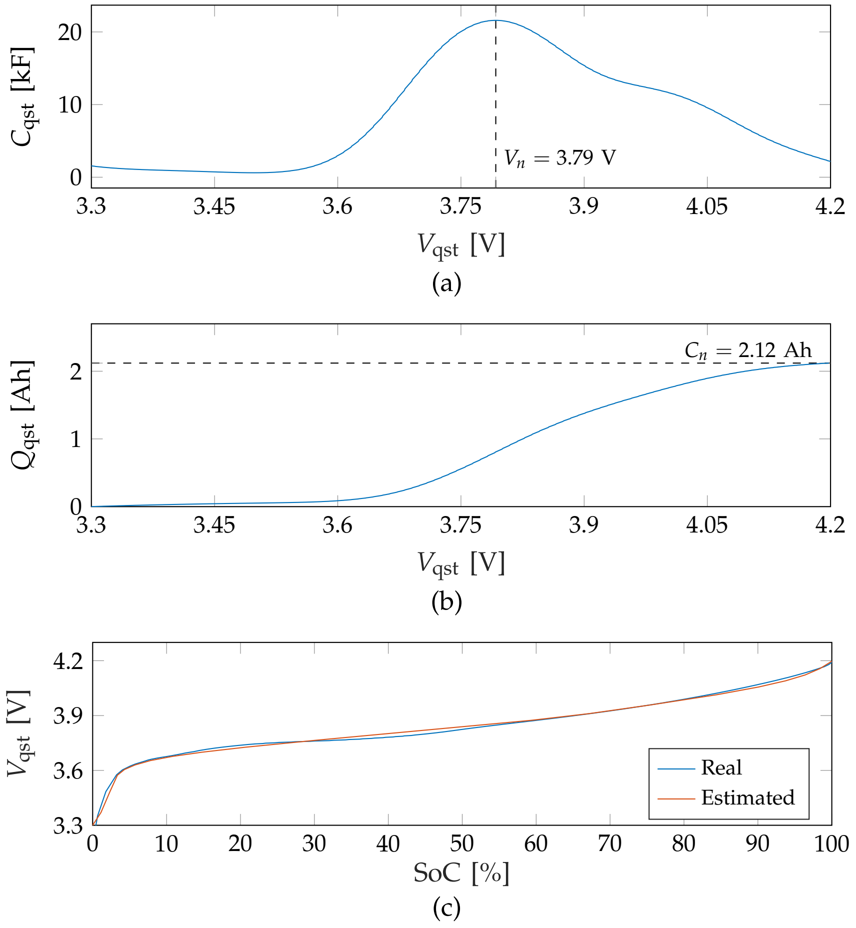


Figure 4.18: Identified Reservoir Properties. (a) Reservoir shape $\mathcal{C}_{qst}(V_{qst})$. (b) Charge-Voltage function $\mathcal{Q}(V_{qst})$. (c) OCV-SoC curve.

$\mathcal{Q}(.)$ and it has been evaluated by means of expression (4.27). In accordance with expression (4.28), the total storable charge Q_{max} is equal to 2.12 Ah, that is almost equivalent to the declared cell capacity of 2.1 Ah. This result proves the effectiveness of the proposed reservoir approach at evaluating the actual charge capacity, as well as the effectiveness of the ANFIS training itself. Moreover, the defined modeling procedure allows also to introduce a physical definition of the nominal voltage of electrochemical cells. Indeed, it can be seen in Figure 4.18(a) that the reservoir shape shows only one global peak, corresponding to the voltage value in which it is stored most of the charge. Therefore, accordingly with [51], it is possible to define the nominal voltage V_n of electrochemical cells as the quasi-stationary voltage V_{qst} related to the maximum of the reservoir shape $\mathcal{C}_{qst}(.)$. In this case, it has been obtained a value of 3.79 V with respect to the 3.7 V declared by the manufacturer.

The model performances at estimating the total output voltage considering both the Training Set and the Test Set sequences are shown in Figure 4.19, where it is shown also the estimation error evaluated as $V_{out} - \tilde{V}_{out}$.

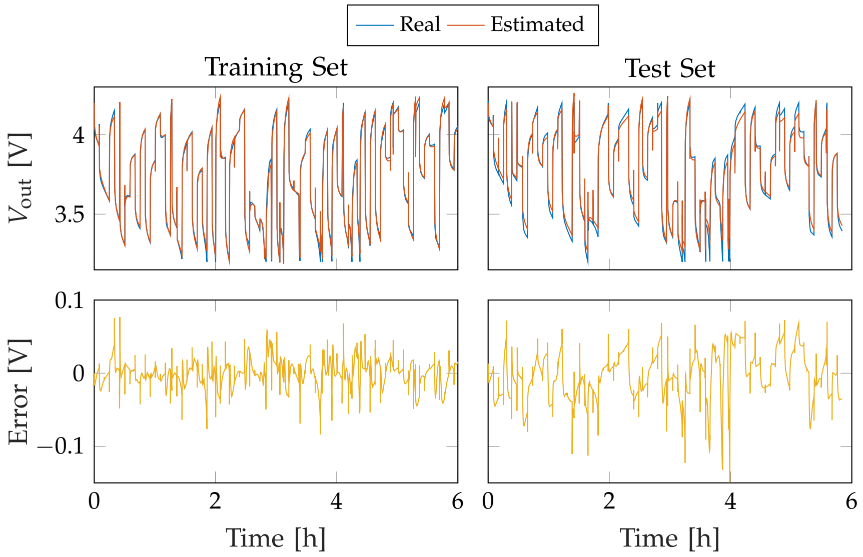


Figure 4.19: Estimation of the terminal voltage in the Training and Test Sets data.

It can be seen that the proposed model is effective also in the overall V_{out} estimation. Indeed, the absolute error is almost always lower than $|0.1|$ V, whereas the performed RMSE in the Training Set and in the Test Set are 1.46*e*-2 and 2.84*e*-2, respectively. The slight difference among the performed RMSEs proves the generalization capability of the model, as well as the effectiveness of the designed training procedure, which avoids to overfit the training data.

SoC Estimation Results A further test has been performed to measure the model performances in the SoC estimation task. To this aim, the same procedure discussed in Section 4.2.2 has been considered. Specifically, the trained model has been implemented in an SR-UKF state observer and SoC estimation has been performed over the Test Set sequence considering a temporal offset of one hour. Also in this case the temporal offset has aimed at testing the convergence of SR-UKF when the cell is in a non-stationary condition, in which the dynamical states are unknown.

SR-UKF is in charge of estimating four state variables, namely the quasi-stationary voltage V_{qst} and the three dynamic voltages V_{dyn_i} related to the RC dipoles. SoC is retrieved from the V_{qst} estimation by applying expression (4.29). The resulting SoC sequence has been compared with that obtained from the coloumb counting approach. In particular, coloumb counting has been started from the beginning of the Test Set sequence because it is known that the cell starts from a fully charged condition. Therefore, expression (1.3) has been initialized with $\text{SoC}[0] = 1$. Moreover, the sequence evaluated with coloumb counting can be reasonably considered equivalent to the real SoC of the cell since $C_n \simeq Q_{\max}$, and the overall sequence is short enough to assume the accumulated errors negligible. The state vector of SR-UKF has been initialized by considering V_{qst} coincident with the measured nominal voltage V_n, whereas all the dynamic terms V_{dyn_i} have been set equal to zero. The estimated V_{qst}, SoC, and V_{out} and the related comparison with the real SoC and the measured voltage are shown in Figure 4.20.

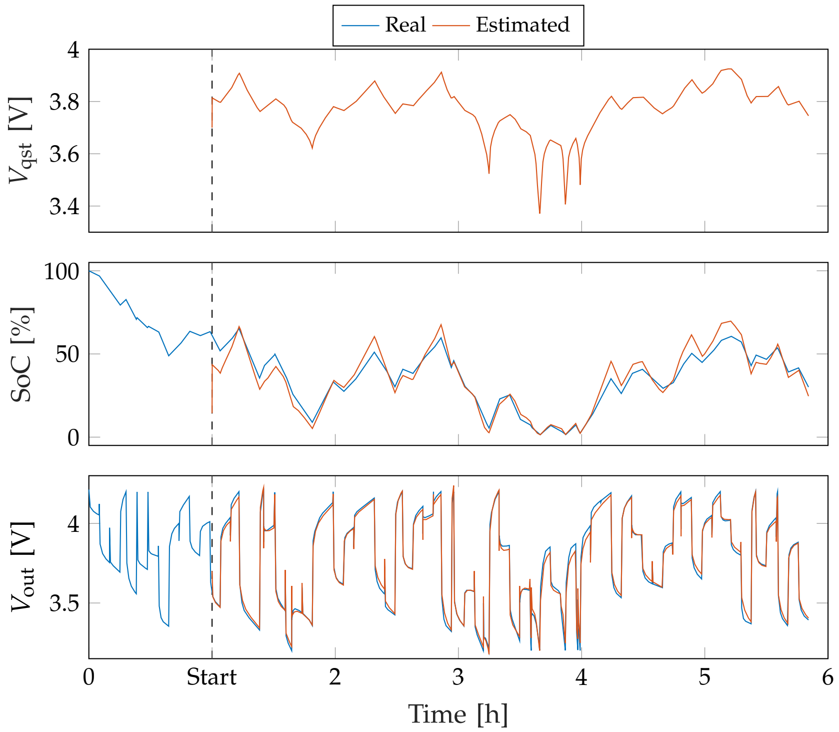


Figure 4.20: Performances of the model in the SoC estimation task.

It is clear that the proposed reservoir modeling and the related SoC definition are effective not only at estimating the output voltage, but also for performing SoC estimation, that is the main reason for which it has been developed. Indeed, SR-UKF converges in few minutes to a V_{qst} value that is related to an SoC close to the real one, obtaining a total RMSE between the real and estimated SoC of 5.04e-2, slightly worse of the 4.06e-2 performed by the linear ECM.

Besides a more proper definition of SoC, the main advantage of M-ECM is that the related system identification procedure can be applied independently of the actual SoC of the cell. In fact, the identification procedure requires the knowledge of the initial value of V_{qst} instead of that of SoC. The benefit is that it is always possible to measure accurately the value of V_{qst}

after a sufficiently long rest of the cell, whereas the value of SoC can be retrieved only after a full charging procedure. Nevertheless, the main drawback of this approach is that SoC estimation is based on the evaluation of the integral of the reservoir shape function $\mathcal{C}_{\text{qst}}(.)$. Therefore, any kind of estimation error of $\mathcal{C}_{\text{qst}}(.)$ can easily result in a significant drop of performances.

4.3 Black Box Technique

The most important black box modeling approaches are those based on ELM [65], RBF-NN [89], [90], and WNN [91]. Accordingly with the black box theory, these models address only the input-output relationship and therefore they are not able to provide any information about the physical properties of electrochemical cells. Moreover, they need the implementation of an autoregressive feedback of the previous output voltage samples in order to model the dynamical response of the cells. The architecture of the ELM, RBF-NN, and WNN models is very similar to each other and it can be generalized in the neural network shown in Figure 4.21.

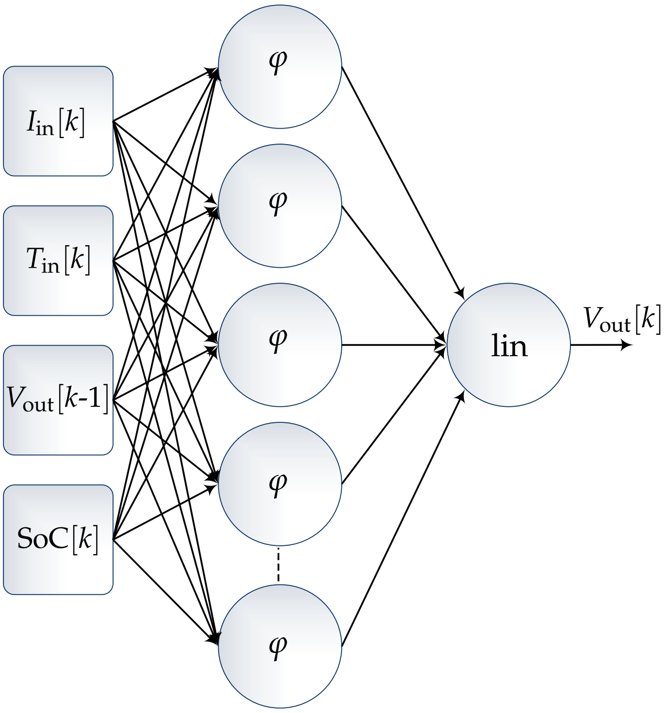


Figure 4.21: Architecture of the neural network black box models.

All the models are feedforward neural networks characterized by only one hidden layer. As explained in Section 2.1, the main differences between ELM, RBF-NN, and WNN concern the activation function used in the hidden neurons. ELM uses an hyperbolic tangent, RBF-NN uses a gaussian function, whereas WNN uses a mother wavelet function. The elaboration of the neurons related to the three models can be summarized as follows:

$$\varphi = \begin{cases} \tanh(\boldsymbol{W}\boldsymbol{x} + b), & \text{ELM} \\ e^{-\beta\|\boldsymbol{x}-\boldsymbol{\mu}\|^2}, & \text{RBF-NN} \\ \prod_{j=1}^{m} \Phi\left(\frac{x_j-\delta_j}{\gamma_j}\right), & \text{WNN} \end{cases} \tag{4.35}$$

where $\boldsymbol{x}$ is the generic input vector of the neuron; $\boldsymbol{W}$ and b are the weights matrix and the bias value related to a neuron of ELM; $\boldsymbol{\mu}$ and β are the centroids and the shape parameters of the gaussian function related to RBF-NN; $\Phi(.)$ is a mother wavelet function, whereas δ_j and γ_j are the translation and dilation terms related to the j^{th} input of a wavelon.

As introduced above, black box models need the feedback of a sliding window of the previous output voltage samples for modeling the dynamical response of the cells. In particular, the sliding window is typically set to one backward step since electrochemical cells are characterized by a markovian dynamical behavior. Thus, the terminal voltage $V_{\text{out}}[k]$ is approximated considering the tuple $\{I_{\text{in}}[k], T_{\text{in}}[k], V_{\text{out}}[k-1], \text{SoC}[k]\}$ as input. Alternatively, the tuple $\{I_{\text{in}}[k], V_{\text{out}}[k-1], \text{SoC}[k]\}$ is considered if temperature measurements are not available.

A comprehensive analysis of the performances of ELM, RBF-NN, and WNN both in the estimation of V_{out} and SoC is discussed Section 4.4.3. In particular, it is shown as black box models are highly ineffective in the SoC estimation task despite they achieve very accurate V_{out} estimations. The main reason of that is attributed to the feedback of the output voltage. Indeed, the sliding window of V_{out} brings all the black box models at overfitting the previous voltage sample, converging to the trivial prediction model for which $V_{\text{out}}[k] \simeq V_{\text{out}}[k-1]$. This fact reduces the dependence of the model output on SoC, resulting in a consequent inability of SR-UKF at correcting a wrong SoC estimation. A more detailed discussion can be found in Section 4.4.3.

4.4 Gray Box Technique

4.4.1 Neural Networks Ensemble Model

White box and black box modeling techniques are characterized by opposite properties. White box models offer a complete insight on the physics of electrochemical cells that is helpful for performing accurate SoC estimations. Nevertheless, the most accurate electrochemical models are too much computational demanding for being applied in a real-time SoC estimator. Conversely, it has been shown that ECMs are effective and efficient models offering promising performances at estimating both the output voltage and SoC. Nevertheless, it has been discussed in Section 4.2.1 that even if the nonlinear ECM is able to track the nonlinearities of electrochemical cells, the related system identification is a stiff, time consuming, and expensive task. Thus, it has been shown in Sections 4.2.2 and 4.2.3 that system identification can be made more flexible for ECMs, as long as the model is largely simplified so that it does not consider anymore most of the nonlinearities of the cells.

As opposite to the white box technique, black box models are able to easily catch the nonlinearities of the cells together with a very flexible system identification procedure. Unfortunately, they address the estimation of the global input-output relationship only, so that black box models do not provide any insight on the cell physics. Furthermore, most of them need a nonlinear autoregressive approach by introducing a sliding window of the previous output voltage in order to model the dynamical response of the system. However, due to the very slow dynamics typical of any electrochemical cell, these models could easily overfit the delayed output, converging to the trivial prediction model in which the output voltage is almost equal to its previous sample. This unwanted behavior could compromise the dependence of the model output on SoC, jeopardizing its effective use for performing SoC estimation.

From the previous analysis, it is clear that neither the white box ECMs, nor the black box machine learning methods are able to fulfill all the three main modeling requirements discussed in Section 1.2. Indeed, ECMs provide a complete insight on the cell physics, but it is necessary to neglect most of the nonlinearities for achieving a flexible system identification procedure. Conversely, black box models offer nonlinear approximation capabilities and flexible system identification procedures, but they do not provide any insight on the cell physics. Thus, a novel gray box neural network model has been developed aiming at achieving a more satisfying compliance with the three main modeling requirements.

The main novelty consists in the application of the gray box technique to neural networks, aiming at identifying not only the overall input-output relationship, but also the internal responses of electrochemical cells. This choice is supported by two reasons. First, it has been discussed that a key aspect for improving the model accuracy consists in the ability of accurately identifying and separating the three main voltage contributions composing the cells' response. Second, an accurate identification of the internal behaviors allows a physical interpretation of them that helps in achieving a more effective SoC estimation by means of state observers.

In order to implement this gray box model, the network architecture has been designed by taking advantage of the a priori knowledge of the system at hand. The architecture consists in a Neural Networks Ensemble (NNE) in which distinct neural networks are used for directly implementing the state form representation of the system. More precisely, all the unknown nonlinear functions appearing in the system equations are approximated with a distinct and specialized neural network. However, the NNE architecture itself does not allow the automatic identification of the internal behaviors of the system, since it still requires a supervised learning of each internal network. Unfortunately, often only the inputs and outputs of the system can be measured, meaning that only a global supervised learning can be performed. Consequently, each internal network should be trained with an unsupervised scheme. To overcome this problem, the inputs and the topology of each internal neural network must be specialized to the specific behavior the network is modeling, so that it is possible to apply an effective globally supervised-locally unsupervised learning of the model.

It must be noticed that the defined gray box neural network approach can be applied to any dynamical system and it is a novel method for applying gray box modeling to nonlinear dynamical systems. Indeed, usually this problem is faced with a black box approach that does not provide any useful insight nor a physical interpretation of the internal behaviors of the system. The only similar gray box method has been found in reference [160], in which the authors use a distinct neural network for each nonlinear function appearing in the state form equations, likewise the proposed approach. However, in that work it is supposed that the internal quantities are measurable and all the related networks are trained with a supervised learning. Conversely, the proposed gray box model is trained considering the supervision of the global output only, whereas ground truth output values of each internal network are actually unknown.

The NNE designed for modeling electrochemical cells is a faithful implementation of the system equations (4.2) and it considers the current I_{in} as input, the terminal voltage V_{out} as output, whereas the state variables are SoC and the dynamic voltage contribution V_{dyn}. Three unknown nonlinear functions can be identified in (4.2), specifically $\mathcal{V}_{\text{ist}}(.)$, $\mathcal{F}_{\text{dyn}}(.)$, and $\mathcal{V}_{\text{qst}}(.)$. As explained in the above sections, $\mathcal{V}_{\text{ist}}(.)$ models the instantaneous contribution, $\mathcal{F}_{\text{dyn}}(.)$ defines the state update equation related to the dynamic voltage response, and $\mathcal{V}_{\text{qst}}(.)$ is coincident with the OCV-SoC curve. In particular, the function $\mathcal{F}_{\text{SoC}}(.)$ has been implemented again considering the coloumb counting algorithm and it is coincident with expression (1.3).

The resulting NNE model is shown in Figure 4.22, where the unknown functions $\mathcal{V}_{\text{ist}}(.)$, $\mathcal{F}_{\text{dyn}}(.)$, and $\mathcal{V}_{\text{qst}}(.)$ have been modeled with an MLP, a Leaky Recurrent Neural Network (Leaky-RNN) [127], and an FLNN, respectively. The choice of each network topology has been guided by the a priori knowledge of each specific behavior and it will be discussed in details in the next paragraphs.

It is worth noting that the proposed NNE model estimates the dynamic contribution taking only the current I_{in} as input. In fact, the internal state related to V_{dyn} is modeled by means of the feedback loops of the Leaky-RNN. Conversely, the black box models ELM, RBF-NN, and WNN need to include the previous sample of the output voltage $V_{\text{out}}[k-1]$ in the input tuple for approximating $V_{\text{out}}[k]$. However, due to the very slow dynamics characterizing the cells (hundreds or even thousands of seconds) and because the sampling time is usually set close to one second, black box models can easily converge to the trivial prediction model in which $V_{\text{out}}[k] \simeq V_{\text{out}}[k-1]$. This fact could jeopardize the effectiveness of the models in the SoC estimation task since it widely compromises the dependence of V_{out} on SoC. Indeed, as any state observer, SR-UKF requires that all the state variables must affect the output of the model in order to allow their effective estimation. If the model converges to $V_{\text{out}}[k] \simeq V_{\text{out}}[k-1]$, the influence of SoC in the estimation of V_{out} is reduced or even nullified, compromising its estimation. The proposed neural model overcomes the drawback discussed above. In fact, it avoids the introduction of a sliding windowing of V_{out}, preserving the influence of SoC in the model output. In addition, it has the further advantage of estimating separately the unknown distinct voltage responses, providing then information about the physical behavior of electrochemical cells.

In the following a detailed description of the three neural networks composing the NNE model is presented.

Instantaneous Timescale Modeling

The instantaneous contribution, which represents the nonlinear resistive behavior of the cell, is modeled by $\mathcal{V}_{\text{ist}}(.)$. In general, it can be thought as a

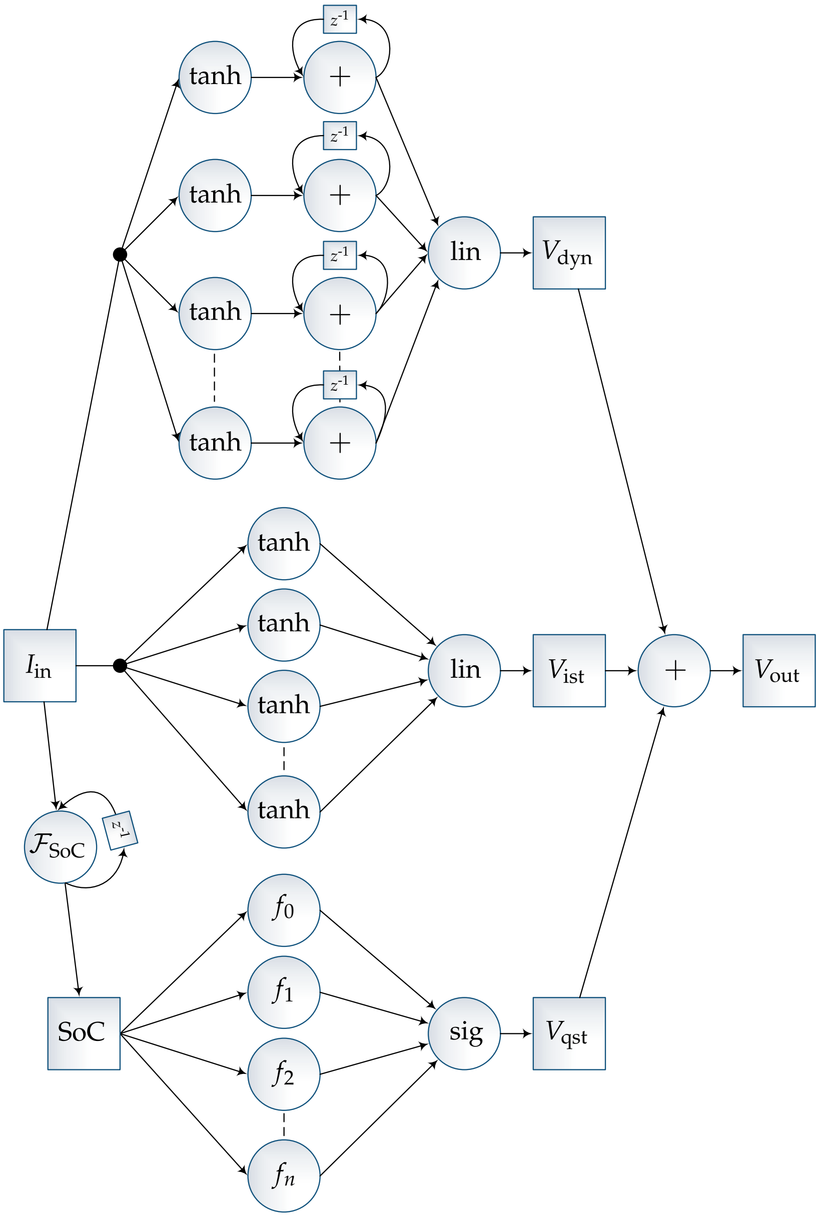


Figure 4.22: Diagram of the NNE model. Top, middle, and bottom networks model the dynamic, the instantaneous, and the quasi-stationary contributions, respectively. The weights of the connections are neglected in the figure, whereas tanh, lin, and sig refer to the hyperbolic tangent, linear, and sigmoid activation functions, respectively.

memoryless nonlinear function of the input current and it is constrained to be zero when the current is zero because of its resistive nature. Thus, any kind of non-recurrent neural network can be used for modeling $\mathcal{V}_{\text{ist}}(.)$. MLP has been chosen among other topologies because it has offered the best trade-off between effectiveness, complexity, and simplicity in forcing the resistive constraint. Indeed, the weak nonlinearity of the instantaneous contribution did not justify the use of other nonlinear approximation models, such as RBF-NN, WNN, and FLNN. Furthermore, all of them have required tricky solutions for ensuring $V_{\text{ist}} = 0$ when $I_{\text{in}} = 0$. ELM was a potential network candidate, but it has required a greater number of hidden neurons for reaching the same performances of MLP.

The network keeps the current I_{in} as input and it counts one hidden layer composed of N_{ist} neurons and one output layer composed of one neuron. The activation function of the hidden neurons is the hyperbolic tangent, whereas that of the output neuron is linear. No bias term has been considered neither for the neurons of the hidden layer nor for that of the output layer. This choice, together with the chosen activation functions, ensures the validity of the resistive constraint. In conclusion, the instantaneous contribution is approximated by means of the following equation:

$$V_{\text{ist}}[k] = \sum_{n=1}^{N_{\text{ist}}} \omega_{\text{ist}_n}^{\text{out}} \tanh\left(\omega_{\text{ist}_n}^{\text{in}} I_{\text{in}}[k]\right) \tag{4.36}$$

where $\omega_{\text{ist}_n}^{\text{in}}$ and $\omega_{\text{ist}_n}^{\text{out}}$ are respectively the input and output weights of the n^{th} neuron.

Dynamic Timescale Modeling

It can be observed from experimental data that electrochemical cells show a typical low pass behavior characterizing their transient response to sharp variations in the input current I_{in}. In general, this kind of low pass behavior has a nonlinear nature and for this reason it is modeled with the nonlinear state update function $\mathcal{F}_{\text{dyn}}(.)$. Furthermore, this voltage response is constrained to convergence to zero when all the transient phenomena have decayed due to its stable nature. Since $\mathcal{F}_{\text{dyn}}(.)$ is a state update function, it has been necessary to implement a memory equipped model and recurrent architectures must have been considered. However, both Elman RNN or nonlinear auto-regressive architectures could not ensure the low pass behavior because of possible positive feedbacks. Alternatively, more complex architectures such as LSTM layers were too much computationally demanding for the aim of the model. For this reason, the dynamic contribution has been modeled with a Leaky-RNN, in which the hidden layer is composed of leaky integrator neurons [127]. This choice has allowed to automatically force the low pass behavior in the network, preserving its computational efficiency.

The dynamic timescale network takes the current I_{in} as input, and it counts one hidden layer with N_{dyn} leaky integrator neurons with hyperbolic tangent activation and one output layer composed of one neuron with linear activation. The network has no bias neither in the hidden layer nor in the output layer in order to reproduce the convergence to zero in absence of the input current. Thus, the output of the dynamic scale network is given by the following equations:

$$\begin{cases} y_n^H[k] = \alpha_n \tanh\left(\omega_{\text{dyn}_n}^{\text{in}} I_{\text{in}}[k]\right) + (1-\alpha_n) y_n^H[k-1] \\ V_{\text{dyn}}[k] = \sum_{n=1}^{N_{\text{dyn}}} \omega_{\text{dyn}_n}^{\text{out}} y_n^H[k] \end{cases} \tag{4.37}$$

where $\omega_{\text{dyn}_n}^{\text{in}}$, $\omega_{\text{dyn}_n}^{\text{out}}$, $\alpha_n \in [0,1]$, and y_n^H refer to the input weight, the output weight, the leak coefficient, and the output of the n^{th} leaky integrator neuron, respectively.

Quasi-Stationary Timescale Modeling

The quasi-stationary voltage contribution is related to the amount of charge actually stored in the cell. It is a nonlinear function of the state variable SoC and it is modeled by $\mathcal{V}_{\text{qst}}(.)$. Even if this contribution is related to a state variable, a recurrent architecture was not necessary in this case because the function $\mathcal{F}_{\text{SoC}}(.)$ is known and well defined by expression (1.3). Consequently, the quasi-stationary response could be modeled with a non-recurrent architecture having SoC as input and V_{qst} as output. Considering the results obtained in [54] and [60], in which the OCV-SoC curve was approximated in an effective way by using a mixture of several nonlinear functions, FLNN is appeared to be the most suitable option for the quasi-stationary network. Indeed, RBF-NN, WNN, MLP, and ELM are resulted not able to deal effectively with the nonlinearity characterizing this response, unless the number of layers, the number of neurons, and the training time were significantly increased.

The functional reservoir is composed of Chebyshev polynomials of the first kind from the zeroth to the $N_{\text{qst}}^{\text{cheth}}$ degree and of trigonometric polynomials from the 1^{th} to the $N_{\text{qst}}^{\text{trgth}}$ degree. The output layer is composed of one neuron with sigmoid activation function. In this case, the sigmoid is preferred since the term V_{qst} has to be non-negative and it has an implicit upper bound given by the full charge condition of the cell. Moreover, the bias term of the output neuron is needed since the quasi-stationary contribution has a natural offset that can be interpreted as the nominal voltage of the cell. Despite that, the bias has not been included in the output neuron since it is implicitly implemented by the zeroth-degree Chebyshev polynomial. Thus, the V_{qst} contribution is approximated by the following expression:

$$\begin{split} V_{\text{qst}}[k] = \text{sig}\left(\sum_{n=0}^{N_{\text{qst}}^{\text{che}}} \omega_{\text{qst}_n}^{\text{che}} \text{p}_n^{\text{che}}(\text{SoC}[k]) + \right. \\ \left. + \sum_{n=1}^{N_{\text{qst}}^{\text{trg}}} \left(\omega_{\text{qst}_n}^{\text{sin}} \sin(n\text{SoC}[k]) + \omega_{\text{qst}_n}^{\text{cos}} \cos(n\text{SoC}[k]) \right) \right) \end{split} \tag{4.38}$$

where $\text{p}_n^{\text{che}}(.)$ is the n^{th} degree Chebyshev polynomial of the first kind, $\omega_{\text{qst}_n}^{\text{che}}$, $\omega_{\text{qst}_n}^{\text{sin}}$, and $\omega_{\text{qst}_n}^{\text{cos}}$ are the weights of the Chebyshev, sinusoidal, and cosinusoidal polynomials of the n^{th} order, respectively.

4.4.2 Model Training and Testing

Data Sets

Three different data sets have been considered for validating the effectiveness of the proposed model. Two of them have been synthetically generated for evaluating the ability of the proposed gray box NNE at identifying automatically the internal contributions of the cell. In fact, only with a synthetic data set it has been possible to get ground truth sequences of V_{ist}, V_{dyn}, and V_{qst} and to compare them with the estimated ones. The third data set is the *Randomized Battery Usage Data Set* collected by the *NASA Ames Research Center*, already introduced in Section 4.2.2 and shown in Figure 4.13. Specifically, the NASA data set has been used for validating the model both in the system identification and the SoC estimation tasks upon real data related to a generic use of a real cell.

The two synthetic data sets have been built by using the linear ECM discussed in Section 4.2.2 as data generator. Similarly to Section 3.2.3, the input current I_{in} has been generated by simulating the use of a BESS in a pure electric vehicle, synthesizing current and voltage sequences as close as possible to those measurable in a real cell during its activity. Also in this case the software ADVISOR has been considered for retrieving the charging/discharging current of the cell by simulating an electric vehicle similar to the *Nissan Leaf*.

Both the Training Set and the Test Set have been generated by simulating a typical daily journey. Thus, they are composed of a sequence of urban, extra-urban, and highway driving cycles alternated with rests and battery charging phases. Again, the selected driving cycles belong to the FTP set. The Training Set simulates a journey of about six hours for a total of 162 km of driving, whereas the Test Set simulates a journey of about five hours for a total of 157.06 km of driving. The data has been collected considering a sampling time $T_s = 1\,\text{s}$, value that is in line with the sampling time of the NASA data set, and that is fast enough both for tracking the dynamics of electrochemical cells and for providing a timely SoC estimation for most of the BESS applications. The list of the cycles composing both the Training Set and the Test Set are shown in Table 4.8.

Table 4.8: List of the driving cycles composing the data sets

Training Set			**Test Set**		
Name*	Duration [s]	Distance [km]	Name*	Duration [s]	Distance [km]
Rest	600	0	Rest	60	0
NREL2VAIL	4320	105.2	VAIL2RAIL	5915	139.29
Rest	600	0	FTP-75	2478	17.77
Full Charge	7200	0	Rest	3600	0
Rest	600	0	Partial Charge	3600	0
UDDS	1370	12	Rest	3600	0
US06	600	12.89			
ARB02	1640	31.91			
Rest	3600	0			
Tot.	20530	162.00	Tot.	19253	157.06

*The acronyms refer to the driving cycles belonging to the FTP set.

In order to generate the sequences of V_{out}, V_{ist}, V_{dyn}, and V_{qst}, the current profiles obtained from ADVISOR have been firstly scaled to the capacity of the cell to be simulated and then applied to the related ECM. In particular, the circuit model has been initialized considering a full charged cell in a stationary condition, i.e. the state variables SoC and V_{dyn} have been initialized to one and zero, respectively. Furthermore, the inaccuracy of the voltage and current sensors has been emulated by adding a zero mean gaussian noise to the input current I_{in} and to the terminal voltage V_{out}. The gaussian noise added to V_{out} has a standard deviation equal to $1e\text{-}4\,\text{V}$, whereas the one of the input current has a standard deviation equal to $3e\text{-}3\,\text{A}$. These values match the standard accuracy of real voltage and current sensors.

The procedure discussed above has been applied to two different ECMs in order to generate the two data sets. The main difference between them is related to the lithium technology of the considered cells, whose specifications are shown in Table 4.9.

Table 4.9: Specifications of the A123 and ePLB cells

Parameter	**A123**	**ePLB**
Technology	Lithium iron-phosphate	Lithium polymer
Capacity	2.4 Ah	15 Ah
Nominal Voltage	3.3 V	3.6 V

The first data set has been generated by applying the current I_{in} to the ECM of an *A123 Nanophosphate AHR23113M1Ultra-B*, whereas the second one refers to an *ePLB C020*. The former cell belongs to the lithium iron-phosphate technology and the latter is a lithium polymer cell. Besides the differences in terms of total capacity, nominal voltage, internal resistance, and different low pass behaviors, the main discrepancy between the two technologies is related to the quasi-stationary contribution. In fact, iron-phosphate cells usually show an almost flat OCV-SoC curve for mid-range SoC values, whereas lithium polymer ones have a steeper curve in the same region of SoC. The synthetic data sets related to the A123 and ePLB cells are shown in Figure 4.23 parts (a) and (b), respectively.

To the purpose of this work the temperature influence has not been taken into account neither in the synthetic nor in the NASA data sets.

Training Procedure

Only the measurements of the input current I_{in} and of the output voltage V_{out} are available at training time, whereas the internal contributions V_{ist}, V_{dyn}, and V_{qst} are unknown. Consequently, although the learning of the whole NNE model is globally supervised, that of the internal networks is locally unsupervised. Thus, the network has been trained only on the output voltage V_{out} by means of GDBP aiming at minimizing the RMSE between the measured and the estimated V_{out}. Despite the local unsupervised learning, the effectiveness of each internal network is guaranteed by the architectural constraints imposed to reproduce the specific dynamical behaviors of the system.

The NNE model has been implemented in the *Python* framework *Keras* running on top of the backend *Theano*, and both the training and the testing have been performed on a laptop featuring an *Intel i5 3230M* and 8GB RAM. GDBP has been performed with the optimizer *Nadam* that has proved to be the most effective with respect to other alternatives such as *Adam*, *Adadelta*, or *RMSprop*. All the network weights have been initialized with the *Glorot*

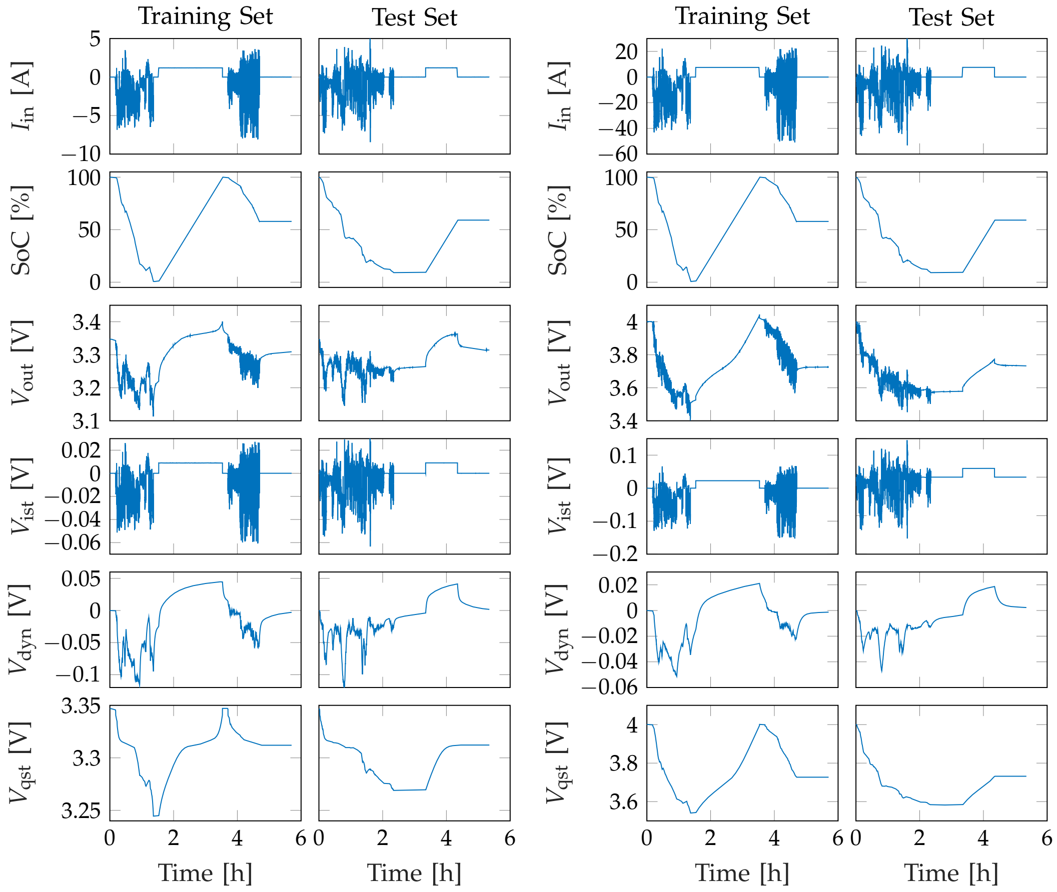


Figure 4.23: Generated Training and Test Sets, I_{in}, SoC, V_{out}, V_{ist}, V_{dyn}, V_{qst}. (a) A123 cell, (b) ePLB cell.

normal initialization, except for those of the FLNN output layer. In this case, a zero initialization has been preferred for avoiding initial oscillations due to a casual mixing of the functional reservoir. The leak coefficients $\boldsymbol{\alpha}$ have been initialized imposing a logarithmic distribution of the leaky integrator time constants in the range between 100 s and 20000 s, in order to cover the characteristic decay times of electrochemical cells [51].

The structural complexities of the three internal neural networks have been preliminary selected by performing a grid search upon the parameters N^{ist}, N^{dyn}, $N^{\text{che}}_{\text{qst}}$, and $N^{\text{trg}}_{\text{qst}}$ for each data set. The best achieved configurations are shown in Table 4.10.

Table 4.10: Configuration of the internal networks of NNE

Data set	N_{ist}	N_{dyn}	$N^{\text{che}}_{\text{qst}}$	$N^{\text{trg}}_{\text{qst}}$
A123	10	5	20	0
ePLB	10	5	20	10
NASA	10	5	20	10

Two different GDBP training schemes have been investigated for the gray box model. The first one is a standard GDBP algorithm, in which all the network weights are updated together during each epoch. The second method performs an alternated training of the internal networks aiming at improving the identification of the unknown behaviors. During each epoch, the three networks are trained in sequence, one at a time, for a certain number of sub-epochs. In particular, when one of the networks is trained, the other two are frozen. In this way, during each global epoch the training procedure updates first only the weights of the instantaneous network, second only those of the dynamic network, and finally only the weights of the quasi-stationary network. The pseudo-code of the Alternated GDBP (A-GDBP) procedure is shown in Algorithm 6.

Algorithm 6 Pseudo-code of the A-GDBP procedure

```
Initialize all the weights.
for $epoch = 1 : N_G$ do
    Freeze dynamic and quasi-stationary networks.
    for $subepoch = 1 : N_S$ do                ▷ Update instantaneous network
        full_network.update($I_{\text{in}}, V_{\text{out}}$)
    end for
    Freeze instantaneous and quasi-stationary networks.
    for $subepoch = 1 : N_S$ do                ▷ Update dynamic network
        full_network.update($I_{\text{in}}, V_{\text{out}}$)
    end for
    Freeze instantaneous and dynamic networks.
    for $subepoch = 1 : N_S$ do                ▷ Update quasi-stationary network
        full_network.update($I_{\text{in}}, V_{\text{out}}$)
    end for
end for
```

It has been noticed that the effectiveness of A-GDBP depends on the number of sub-epochs N_S. Indeed, a small value did not allow a significant improvement of the internal networks, whereas a big value could result in an oscillation of the training algorithm. For this specific case, a number of 10 subepochs has provided the best trade-off between learning speed and undesired oscillations.

The effectiveness of the two learning approaches has been analyzed by executing both of them without any limitation on the training time. In particular, a significant high number of epochs has been considered with the aim of ensuring that both the learning schemes would reach the convergence. With regards to the A-GDBP method, a training epoch has been considered coincident with each local update of the networks' weights. Thus, A-GDBP has been run for 334 global epochs and 10 subepochs for each subnetwork, resulting in a total of $334 \times 10 \times 3 = 10020$ updates. For the sake of a fair comparison, also the GDBP method has been performed with 10020 epochs. The evolution of the training loss of the two learning schemes considering all the data sets is shown in Figure 4.24.

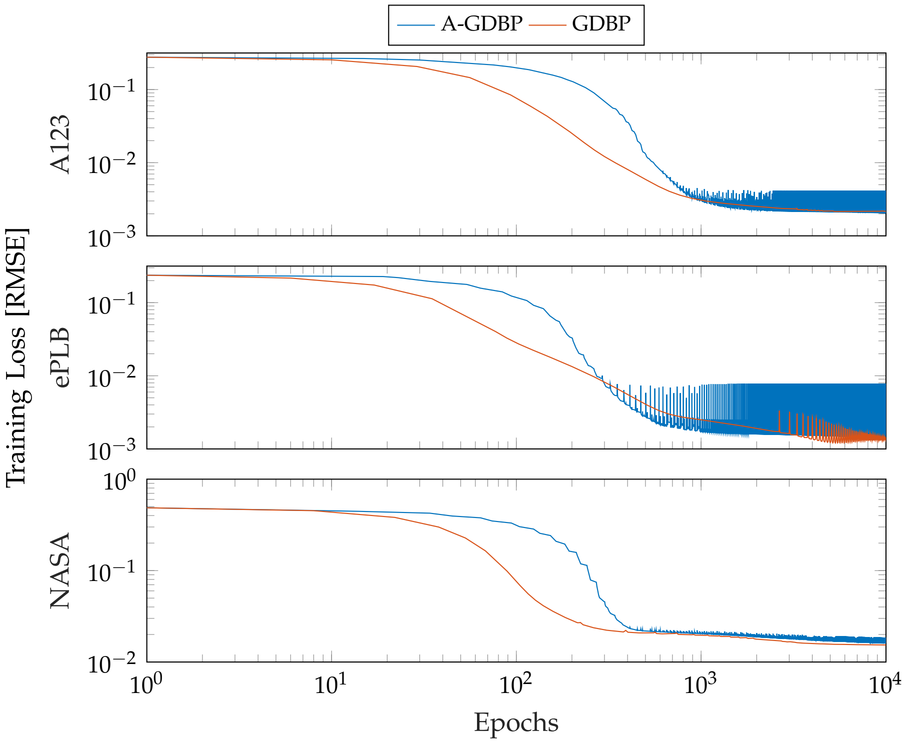


Figure 4.24: Evolution of the training loss for A-GDBP and GDBP training methods.

Both A-GDBP and GDBP have reached a similar loss value at the end of the training epochs for any of the three data sets. In particular, GDBP has showed a more stable and efficient convergence characterized by fewer oscillations and a faster decreasing of the training loss. Conversely, A-GDBP has introduced a systematic oscillation beginning from between the 200th and the 1000th epoch. These oscillations are related to the switching between the learning of the internal networks. Indeed, as the learning of NNE advances, a temporary increase of the training loss is observed at the beginning of each local training. Despite this phenomenon, A-GDBP is characterized by an overall improvement of the training loss resulting in a final performance similar to that of the GDBP algorithm.

The training times and the overall RMSE performed by the two learning schemes at estimating V_{out}, V_{ist}, V_{dyn}, and V_{qst} for both the Training and the Test Sets are listed in Table 4.11.

The performances of the two learning schemes are very similar and both

Table 4.11: RMSE Performances of A-GDBP and GDBP

Training Set

	A-GDBP			**GDBP**		
	A123	ePLB	NASA	A123	ePLB	NASA
Time	3366 s	4332 s	5035 s	6821 s	7964 s	8840 s
V_{out}	2.14e-3	1.42e-3	1.62e-2	2.15e-3	1.45e-3	1.54e-2
V_{ist}	7.33e-4	1.48e-4	N/A	7.30e-4	2.64e-4	N/A
V_{dyn}	5.02e-3	4.00e-4	N/A	4.51e-3	3.46e-4	N/A
V_{qst}	5.25e-3	1.28e-3	N/A	4.57e-3	1.37e-3	N/A

Test Set

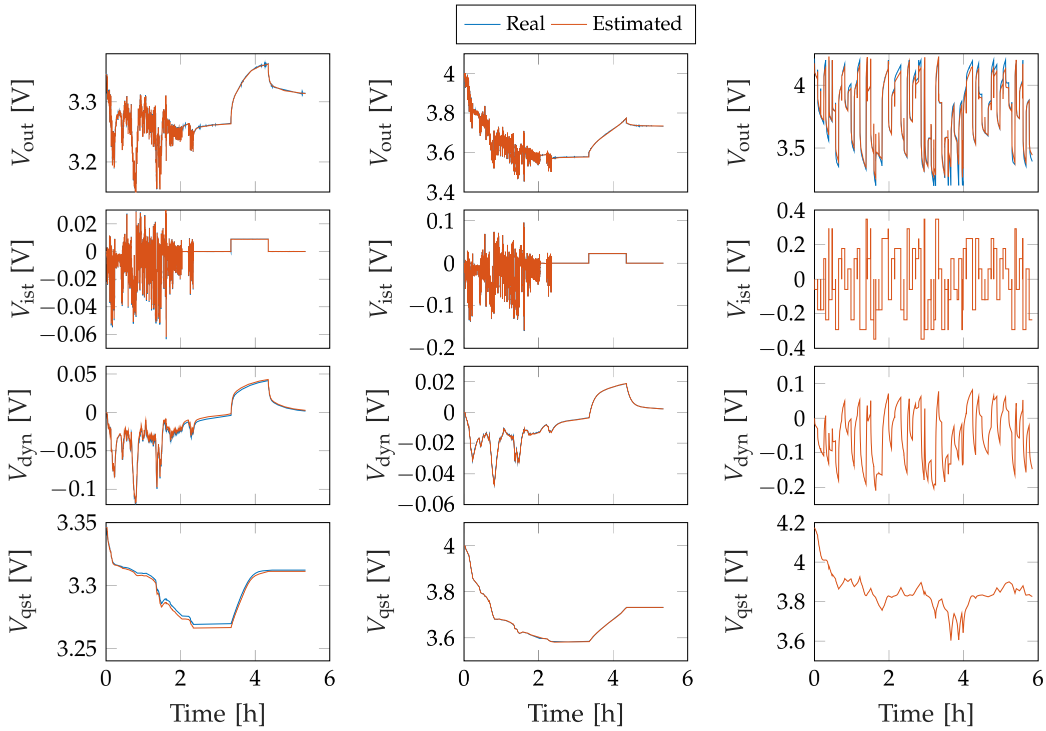
	A-GDBP			**GDBP**		
	A123	ePLB	NASA	A123	ePLB	NASA
V_{out}	1.84e-3	1.37e-3	2.71e-2	1.86e-3	1.28e-3	2.80e-2
V_{ist}	6.01e-4	1.23e-4	N/A	5.97e-4	1.89e-4	N/A
V_{dyn}	6.34e-3	3.75e-4	N/A	5.80e-3	3.74e-4	N/A
V_{qst}	6.64e-3	1.16e-3	N/A	5.71e-3	1.21e-3	N/A

of them have succeeded at training NNE such that it approximates the output voltage and all the internal contributions in a very effective way. Furthermore, the slight differences between the RMSEs performed in the Training and the Test Sets proves that both the learning algorithms have achieved a good generalization capability, without overfitting the training data. Although the two learning methods have been equivalent in terms of the final approximation accuracy, A-GDBP is appeared significantly better with regards to the training time. Indeed, it has required almost half of the total time of GDBP for completing the training of NNE. Moreover, even if the total number of epochs is the same, A-GDBP has updated each network weight only $334 \times 10 = 3340$ times during the entire training procedure. Consequently, A-GDBP has proved to be the most effective learning approach, reaching the same final performances with a lower time and a lower number of weights update. In conclusion, the estimation of V_{out}, V_{ist}, V_{dyn}, and V_{qst} performed by NNE trained with A-GDBP over the Test Set of the A123, the ePLB, and the NASA data sets are shown in Figure 4.25 parts (a), (b), and (c), respectively.

4.4.3 Model Performances

System Identification Task

The performances of the proposed gray box NNE model have been compared with those of the black box models ELM, RBF-NN, and WNN.



(a) (b) (c)

Figure 4.25: Estimation of V_{out}, V_{ist}, V_{dyn}, and V_{qst} on the Test Set: (a) A123 cell, (b) ePLB cell, (c) NASA cell.

The ELM model has been setup with the same configuration of [65], counting ten hidden neurons with hyperbolic tangent activation function and one output neuron with linear activation. In accordance with the ELM technique, the weights of the hidden layer have been randomly initialized and the training of the output weights has been performed by solving a linear least square problem.

The RBF-NN model has been configured with ten hidden neurons and one output neuron with linear activation function. In particular, the centroids of the hidden layer have been initialized with a K-means clustering. The training procedure updates both the centroids and the standard deviations of the gaussian function as well as the weights of the output neuron.

The WNN model has been configured with ten hidden wavelons and one output neuron with linear activation function. In accordance with [91], the Morlet wavelet has been used as activation function of the wavelons. A K-means clustering has been used in order to initialize the translation and the dilatation parameters of the wavelets. In particular, the translations have been initialized to the centroids of the detected clusters and the dilatation to their radius. The training procedure updates the translation and the dilatation of the wavelons and the weights of the output neuron.

Also the competing models have been implemented with *Keras* and all the weights have been initialized with a *Glorot* normal initialization. Furthermore, all of them keep the tuple $\{I_{\text{in}}[k], \text{SoC}[k], V_{\text{out}}[k-1]\}$ as input and estimate the terminal voltage $V_{\text{out}}[k]$. The training procedure has been performed again with the *Nadam* optimizer and both the RBF-NN and the WNN models have been trained for 10020 epochs for allowing a fair comparison with the NNE model.

Three performance metrics have been considered in order to compare the models. First, the RMSE between the measured and estimated output voltage has been adopted in order to measure the average estimation accuracy. Second, the maximum percentage error $E_{\%}^{\text{max}}$ and the maximum absolute error $E_{\text{abs}}^{\text{max}}$ expressed in mV have been evaluated with the aim of measuring the overall spread of the estimation error. The results for all the models are listed in Table 4.12, including the temporal information related to the training and testing times[2].

The proposed NNE has achieved the best performances with RMSEs up to two order of magnitude better than those of the competing models in both the Training and the Test Sets of the A123 and the ePLB data sets. Moreover, it has achieved also the best performances for the maximum percentage error $E_{\%}^{\text{max}}$ and the maximum absolute error $E_{\text{abs}}^{\text{max}}$. In fact, the maximum values of these performance metrics computed on the Test Set of the two synthetic data sets are 0.15 % and 4.95 mV, respectively. Considering the real data of the NASA data set, NNE has shown a slight worse RMSE with respect to the competing models. On the other hand, it has achieved again the best $E_{\%}^{\text{max}}$ and $E_{\text{abs}}^{\text{max}}$. In particular, the network has performed a $E_{\%}^{\text{max}}$ of 5.22 % and a $E_{\text{abs}}^{\text{max}}$ of 167.29 mV for the Test Set.

As expected, the greater complexity of NNE is resulted in a likewise greater computational demand both in the training and in the execution of the model for all the three data sets. Despite that, both the higher computational demand and the slight worse accuracy performed in the NASA data set are largely counterbalanced by the introduction of the gray box approach. Indeed, this method has allowed to obtain the identification of the internal behaviors of the cell together with the best performances in the A123 and ePLB data sets, and worst but still comparable estimation accuracy in the NASA data set. In particular, the former aspect makes the NNE model more suitable for its application in the SoC estimation task.

[2] Both training and testing of every model have been performed on a laptop featuring an *Intel i5 3230M* and 8GB RAM.

Table 4.12: Comparison among NNE, ELM, RBF-NN and WNN

A123 Data Set

	Training Set			**Test Set**			**Time**	
Model	RMSE	$E_{\%}^{\max}$ [%]	$E_{\text{abs}}^{\max}$ [mV]	RMSE	$E_{\%}^{\max}$ [%]	$E_{\text{abs}}^{\max}$ [mV]	Train [s]	Test [s]
NNE	**2.14*e*-3**	**0.16**	**5.38**	**1.84*e*-3**	**0.15**	**4.95**	3366	3.45
ELM	1.12*e*-2	1.82	59.45	8.55*e*-3	1.47	47.76	**0.62**	**0.61**
RBF-NN	1.06*e*-4	1.47	48.18	8.86*e*-3	1.48	48.09	2851	0.72
WNN	9.77*e*-3	1.69	55.24	1.17*e*-2	1.48	48.03	3651	0.75

ePLB Data Set

	Training Set			**Test Set**			**Time**	
Model	RMSE	$E_{\%}^{\max}$ [%]	$E_{\text{abs}}^{\max}$ [mV]	RMSE	$E_{\%}^{\max}$ [%]	$E_{\text{abs}}^{\max}$ [mV]	Train [s]	Test [s]
NNE	**1.42*e*-3**	**0.17**	**6.79**	**1.37*e*-3**	**0.14**	**5.52**	4332	3.86
ELM	9.91*e*-3	3.21	119.94	1.07*e*-2	2.77	98.77	**0.61**	**0.63**
RBF-NN	6.91*e*-3	1.31	51.78	8.77*e*-3	1.72	61.56	2840	0.74
WNN	6.45*e*-3	1.28	48.23	1.02*e*-2	1.70	60.77	3683	0.79

NASA Data Set

	Training Set			**Test Set**			**Time**	
Model	RMSE	$E_{\%}^{\max}$ [%]	$E_{\text{abs}}^{\max}$ [mV]	RMSE	$E_{\%}^{\max}$ [%]	$E_{\text{abs}}^{\max}$ [mV]	Train [s]	Test [s]
NNE	1.62*e*-2	**4.85**	**155.10**	2.71*e*-2	**5.22**	**167.29**	5035	3.89
ELM	1.53*e*-2	12.74	460.83	1.64*e*-2	7.84	276.02	**0.60**	**0.65**
RBF-NN	8.99*e*-3	6.89	255.42	**9.99*e*-3**	6.37	245.02	3393	0.73
WNN	**8.25*e*-3**	6.98	273.51	1.03*e*-2	5.82	216.47	4252	0.79

SoC Estimation Task

Two further tests have been performed for investigating the effectiveness of all the considered models in the SoC estimation task.

The first one has aimed at analyzing if SoC affects enough the estimation of V_{out}. Indeed, as explained in Section 4.4.1, the output of a model to be used in the SoC estimation task should be SoC dependent to ensure the effectiveness of the state observer estimator. Thus, NNE, ELM, RBF-NN, and WNN have been tested again on the NASA Test Set imposing an intentional 50 % error in the initialization of expression (1.3). Therefore, it is expected that the resulting erroneous SoC sequence should compromise significantly the estimation of V_{out}. The obtained results are shown in Figure 4.26.

Concerning NNE, the wrong SoC values are resulted in the expected loss of accuracy in the V_{out} estimation. Consequently, the proposed model is able to provide information to the state observer in order to correct a wrong

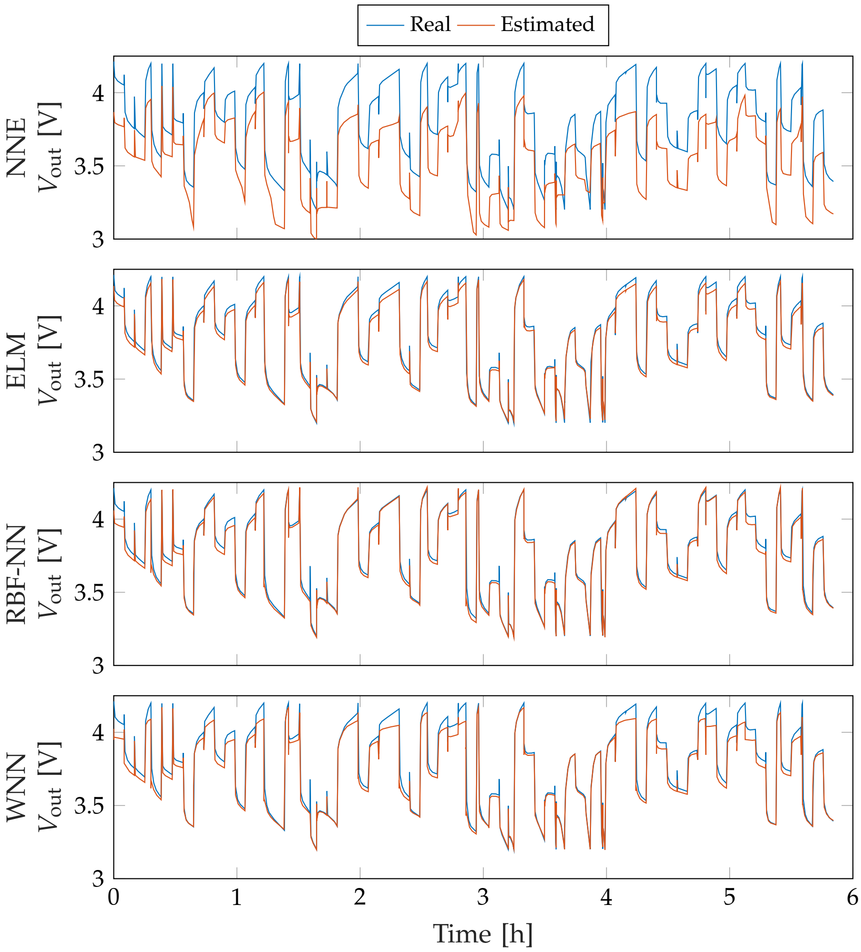


Figure 4.26: Estimation of V_{out} for the Test Set of the NASA cell considering a wrong SoC sequence as input. A likewise wrong V_{out} estimation is expected.

SoC estimation and to converge to its real value. Conversely, all the competing models have kept acceptable V_{out} estimations despite of the wrong SoC sequence. These results suggest that the voltage estimation is slightly affected by SoC in the ELM, RBF-NN, and WNN models. Therefore, the state observer could not have sufficient information for converging to the true SoC. The reason of this undesirable behavior is related to the inclusion of $V_{\text{out}}[k-1]$ in the input tuple. In fact, the windowing of V_{out} is resulted in the overfitting of $V_{\text{out}}[k-1]$ caused by the very slow dynamics of electrochemical cells.

The performances of the considered models in the SoC estimation task have been directly compared in the second test. In particular, all the models have been implemented in the SR-UKF state observer and SoC estimation has been performed again on the NASA Test Set. Furthermore, similarly to the white box models, SR-UKF has been started again with a temporal offset of one hour in order to perform the task when the cell is in a nonstationary condition in which all the internal states are unknown. The SR-UKF related to the NNE model is characterized by a 6-D state space dimension. The first element of the state vector is SoC, whereas the last five are the output of each leaky integrator neuron of the dynamic network. In accordance with [65], [89]–[91], all the competing models share the same SR-UKF configuration and the state vector is composed of SoC and of the output voltage V_{out}. The results are shown in Figure 4.27.

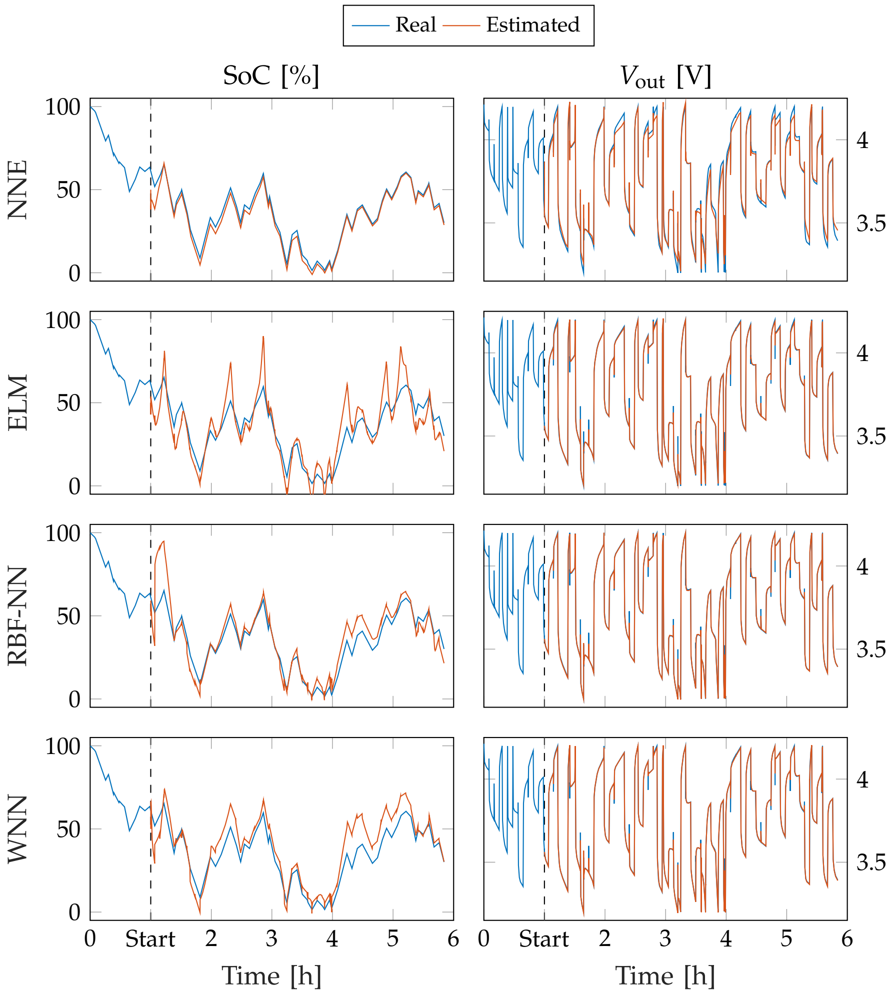


Figure 4.27: SoC and V_{out} estimations performed by the SR-UKF.

Despite the greater state vector dimension, the proposed NNE is much more effective than the competing models. Indeed, it has provided a more accurate and robust estimation with a total RMSE between the real and the estimated SoC of 3.12*e*-2. In more details, SR-UKF has converged to the true SoC in few minutes and then it has kept an accurate estimation during the entire sequence length. Conversely, all the competing models have been characterized by temporary losses of convergence that have compromised their estimation accuracy. The ELM, RBF-NN, and WNN models have reached a total RMSE between the real and the estimated SoC of 9.43*e*-2, 8.46*e*-2, and 8.78*e*-2, respectively. These worst performances are due to the slight dependence of the V_{out} estimation on SoC. Indeed, it can be noticed that SR-UKF has reached a very good estimation of the output voltage for all the competing models, but often it was not able to find the true value of SoC.

4.5 A Neural White Box Model: Equivalent Neural Network Circuit

4.5.1 Equivalent Neural Network Circuit

Model Architecture

In the previous sections it has been shown that black box models are not able to provide satisfactory SoC estimation despite their good model accuracy and their flexible system identification procedure. The main reason of that has been recognized in the overfitting of the sliding window of the output voltage that compromises the dependence of the model output on SoC. Conversely, the white box ECM and gray box NNE have proved to be very promising models for being used in the SoC estimation task.

ECM and NNE counterbalance themselves in terms of complexity, flexibility, and generalization capability. The former is a very simple and almost linear model. Consequently, ECM is very fast both in the voltage prediction and in the SoC estimation task. Furthermore, its white box nature allows a complete understanding of the cell behavior. Nevertheless, ECM does not take into account most of the nonlinearities affecting the relationships between the input physical quantities and each voltage contribution. NNE is definitively more complex and more computationally demanding than ECM. Despite that, it easily takes into account the nonlinear behaviors of the cell, thanks to its architecture and to the learning capability of neural networks. As a result, NNE is more accurate and it performs better in the SoC estimation task. Moreover, its gray box nature still allows a good understanding of the cell physics. Nevertheless, NNE is still not as flexible to coverage all the possible nonlinear relationships between each voltage contribution and all the input physical quantities. This is because the model must respect some architectural constraints in order to ensure its effectiveness at separating and identifying the three main voltage responses of the cell.

A novel Equivalent Neural Network Circuit (ENNC) model being an evolution of NNE has been developed with the purpose of designing a model able to provide a complete insight on the cell physics, a complete approximation of the nonlinear responses, as well as a flexible system identification procedure. In particular, the architecture of ENNC aims at taking advantage of the strengths of both the white box and the gray box techniques by means of an hybridization between NNE and ECM. More precisely, ENNC inherits from the linear ECM of Figure 4.12 the overall model architecture and its white box nature based on the connection of lumped elements. On the other hand, it inherits from NNE the neural networks ensemble design, the flexible system identification based on the learning procedure, and the nonlinear approximation capability typical of neural networks. The block diagram of ENNC is shown in Figure 4.28.

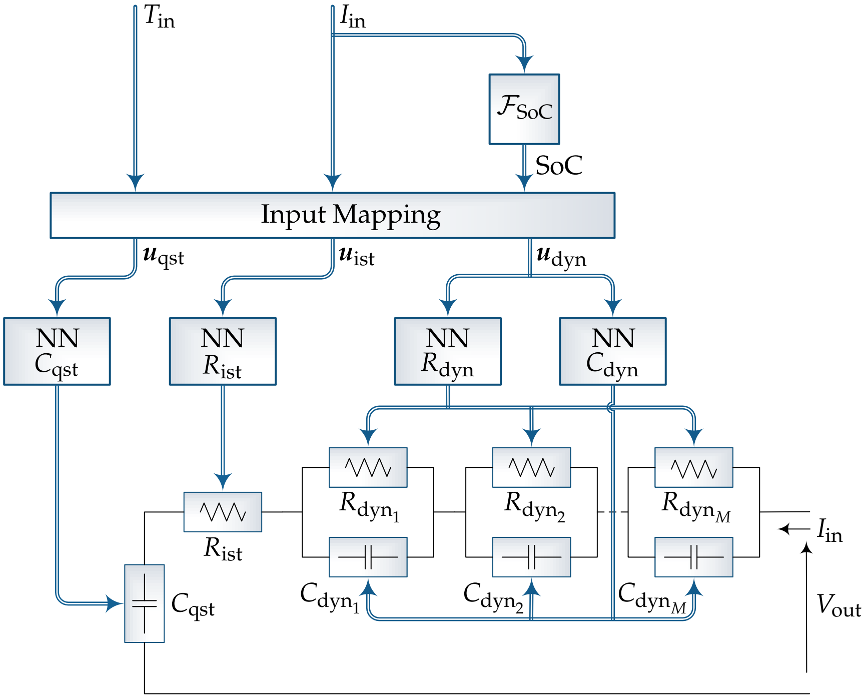


Figure 4.28: Block diagram of the ENNC model.

The overall architecture is actually coincident with that of ECM. Nevertheless, the elements used in the proposed model are not properly electrical components. Indeed, they can be thought as generic dipoles generalizing the constitutive equations of the original electrical devices. Thus, each resistor and capacitor still imposes $V_R = RI_R$ and $I_C = \int V_C/C\,dt$, respectively, but the values of the resistance and the capacitance are defined as generic nonlinear functions of the main physical quantities I_{in}, T_{in}, and SoC. More precisely, each of these nonlinear functions has been approximated with a dedicated neural network, whose inputs are the tuples $\boldsymbol{u}_{\text{ist}}$, $\boldsymbol{u}_{\text{dyn}}$, and $\boldsymbol{u}_{\text{qst}}$, referring to the instantaneous, dynamic, and quasi-stationary elements, respectively. Therefore, the circuit architecture of ECM has been used for imposing the physical equations of the circuit elements, whereas the neural networks have allowed to generalize each resistance and capacitance as a nonlinear function of the input physical quantities. Considering the above discussion, the system equations can be written as follows:

$$\begin{gathered}\begin{bmatrix}\text{SoC}[k+1]\\ V_{\text{dyn}_1}[k+1]\\ \vdots\\ V_{\text{dyn}_M}[k+1]\end{bmatrix} = \mathcal{A}(\boldsymbol{u}_{\text{dyn}}[k])\begin{bmatrix}\text{SoC}[k]\\ V_{\text{dyn}_1}[k]\\ \vdots\\ V_{\text{dyn}_M}[k]\end{bmatrix} + \mathcal{B}(\boldsymbol{u}_{\text{dyn}}[k])I_{\text{in}}[k]\\ V_{\text{out}}[k] = \mathcal{V}_{\text{qst}}(\boldsymbol{u}_{\text{qst}}[k]) + \sum_{i=1}^{M}\omega_{\text{dyn}_i}V_{\text{dyn}_i}[k] + \mathcal{R}_{\text{ist}}(\boldsymbol{u}_{\text{ist}}[k])I_{\text{in}}[k]\end{gathered} \tag{4.39}$$

where M is the number of RC dipoles modeling the dynamic response V_{dyn}, $D = M+1$ is the state space dimension of the model, $\mathcal{A}(\boldsymbol{u}_{\text{dyn}}[k])$ is a $D \times D$ diagonal matrix, and $\mathcal{B}(\boldsymbol{u}_{\text{dyn}}[k])$ is a $D \times 1$ matrix; V_{dyn_i} and ω_{dyn_i} are the output and the weight of the i^{th} dynamic dipole, respectively; $\mathcal{R}_{\text{ist}}(.)$ is the nonlinear function related to the instantaneous resistor, and the function $\mathcal{V}_{\text{qst}}(.)$ is again coincident with the OCV-SoC curve. The matrices $\mathcal{A}(\boldsymbol{u}_{\text{dyn}}[k])$ and $\mathcal{B}(\boldsymbol{u}_{\text{dyn}}[k])$ are defined as follows:

$$\mathcal{A}(\boldsymbol{u}_{\text{dyn}}[k]) = \begin{bmatrix}1 & & & 0\\ & e^{\frac{-T_s}{\mathcal{T}_{\text{dyn}_1}(\boldsymbol{u}_{\text{dyn}}[k])}} & & \\ & & \ddots & \\ 0 & & & e^{\frac{-T_s}{\mathcal{T}_{\text{dyn}_M}(\boldsymbol{u}_{\text{dyn}}[k])}}\end{bmatrix} \tag{4.40}$$

$$\mathcal{B}(\boldsymbol{u}_{\text{dyn}}[k]) = \begin{bmatrix}\frac{T_s}{C_n}\\ \left(1 - e^{\frac{-T_s}{\mathcal{T}_{\text{dyn}_1}(\boldsymbol{u}_{\text{dyn}}[k])}}\right)\mathcal{R}_{\text{dyn}_1}(\boldsymbol{u}_{\text{dyn}}[k])\\ \vdots\\ \left(1 - e^{\frac{-T_s}{\mathcal{T}_{\text{dyn}_M}(\boldsymbol{u}_{\text{dyn}}[k])}}\right)\mathcal{R}_{\text{dyn}_M}(\boldsymbol{u}_{\text{dyn}}[k])\end{bmatrix} \tag{4.41}$$

where $\mathcal{R}_{\text{dyn}_i}(.)$ is the nonlinear function modeling the i^{th} dynamic resistor, whereas the i^{th} dynamic capacitor has been modeled by means of the characteristic time constant, whose value is defined by the nonlinear function $\mathcal{T}_{\text{dyn}_i}(.)$.

Neural Network Implementation

The neural network implementation of ENNC is shown in Figure 4.29. The model is composed of four main neural networks, each one modeling one of the circuit elements shown in Figure 4.28.

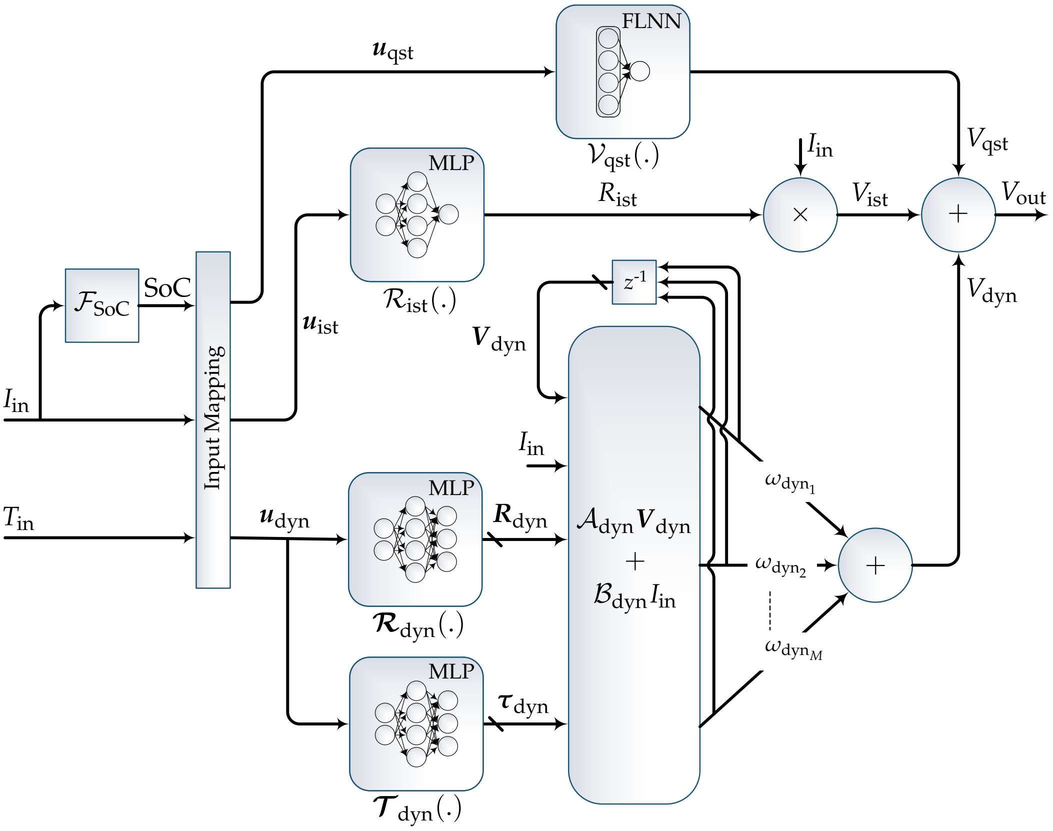

$\boldsymbol{u}_{\text{qst}}$
FLNN
$\mathcal{V}_{\text{qst}}(.)$
V_{qst}
I_{in}
MLP
R_{ist}
×
V_{ist}
+
V_{out}
$\mathcal{R}_{\text{ist}}(.)$
z^{-1}
V_{dyn}
$\mathcal{F}_{\text{SoC}}$
SoC
$\boldsymbol{u}_{\text{ist}}$
$\boldsymbol{V}_{\text{dyn}}$
I_{in}
Input Mapping
I_{in}
ω_{dyn_1}
T_{in}
$\boldsymbol{u}_{\text{dyn}}$
MLP
$\boldsymbol{R}_{\text{dyn}}$
$\mathcal{A}_{\text{dyn}}\boldsymbol{V}_{\text{dyn}}$
+
$\mathcal{B}_{\text{dyn}}I_{\text{in}}$
ω_{dyn_2}
+
ω_{dyn_M}
$\boldsymbol{\mathcal{R}}_{\text{dyn}}(.)$
MLP
$\boldsymbol{\tau}_{\text{dyn}}$
$\boldsymbol{\mathcal{T}}_{\text{dyn}}(.)$

Figure 4.29: Neural Network implementation of the ENNC model.

Distinct and dedicated MLPs have been used for modeling the instantaneous resistor R_{ist}, the dynamic resistors $\boldsymbol{R}_{\text{dyn}} = \{R_{\text{dyn}_1}, \ldots, R_{\text{dyn}_M}\}$, and the dynamic time constants $\boldsymbol{\tau}_{\text{dyn}} = \{\tau_{\text{dyn}_1}, \ldots, \tau_{\text{dyn}_M}\}$. In particular, the MLP related to R_{ist} has only one output neuron because the model has only one instantaneous resistor. Conversely, the MLPs related to $\boldsymbol{R}_{\text{dyn}}$ and $\boldsymbol{\tau}_{\text{dyn}}$ count M output neurons, with M equal to the number of RC dipoles composing the overall dynamic response of the model. With regards to the input tuples, both the instantaneous and the dynamic electric components can be potentially affected by all of the physical quantities. Consequently, $\boldsymbol{u}_{\text{ist}}$ and $\boldsymbol{u}_{\text{dyn}}$ can include any kind of combination of I_{in}, SoC, and T_{in}.

In accordance with the modeling techniques discussed in Sections 4.2.2 and 4.4.1, the capacitor C_{qst} has been modeled by means of the nonlinear OCV-SoC curve, that provides directly the value of the quasi-stationary voltage V_{qst} as a function of SoC. Likewise NNE, this nonlinear curve has been approximated through an FLNN taking as input the tuple $\boldsymbol{u}_{\text{qst}}$, and providing as output the value of V_{qst}. Indeed, it has been already shown in Section 4.4.1 that FLNN offers the best approximation performances of the OCV-SoC curve with respect to other neural network architectures. In accordance with the OCV-SoC representation, SoC is a mandatory input of FLNN and it must be always included in $\boldsymbol{u}_{\text{qst}}$. Although V_{qst} can be affected also by I_{in}, only the temperature T_{in} can be considered as a further input of FLNN. This is the only constraint of the model and it has been necessary for avoiding any kind of conflict between the MLP modeling R_{ist} and the FLNN modeling V_{qst}. Indeed, since both the networks have a memoryless architecture, the inclusion of I_{in} in the tuple $\boldsymbol{u}_{\text{qst}}$ will cause the FLNN trying to approximate part of the instantaneous voltage V_{ist}, compromising the effectiveness of the MLP related to R_{ist}.

The other blocks composing ENNC perform the remaining calculations for implementing the system equations (4.39). Thus, the value of R_{ist} is multiplied by I_{in} to get the instantaneous contribution V_{ist}. Similarly, the state update equations related to each RC dipole are applied for obtaining the values of every V_{dyn_i}. Moreover, the final value of the dynamic contribution V_{dyn} has been obtained as the weighted sum of the single responses V_{dyn_i}. Consequently, the learning procedure is in charge also of training the weights ω_{dyn_i}.

4.5.2 Experimental Setup

Data Sets

Two distinct data sets have been considered for testing the proposed ENNC. Both of them have been chosen in order to test the model upon measurements related to generic or realistic applications of the cell. Indeed, one of the main advantages of the proposed approach is its greater flexibility that allows to perform system identification upon generic data, avoiding specific and time consuming tests.

The first set is again the *Randomized Battery Usage Data Set* chosen for its generic use of the cell. In particular, also the T_{in} sequence has been considered in this case because ENNC has the ability of dealing with the temperature input conversely to the other models discussed so far.

The second data set has been collected autonomously by means of a self-developed programmable battery cycler whose details are discussed in Section 5.2. More precisely, a lithium polymer cell model *ePLB C020* characterized by an effective capacity of 15 Ah has been tested by simulating its use during a journey of an electric vehicle similar to the *Nissan Leaf*. Two different trips have been considered for building the Training and the Test Sets. In order to consider a realistic environment, both the trips have been composed by mixing urban, extra-urban, and highway driving cycles with rests and battery charging phases. The considered driving cycles have been selected from the FTP repository. The complete list of the cycles composing both the Training and the Test Sets are shown in Table 4.13. The Training Set consists in a trip of 277.64 km for a total of about 12 h of data, whereas the Test Set counts 163.24 km and about 7 h of data. The measurements of I_{in}, V_{out}, and SoC have been performed with a sampling time of 1 s and the resulting sequences are shown in Figure 4.30. The SoC sequence has been evaluated by means of expression (1.3) considering $C_n = 15$ Ah. Unfortunately, the battery cycler has not still provided with temperature sensing and T_{in} has not been considered for the ePLB data set.

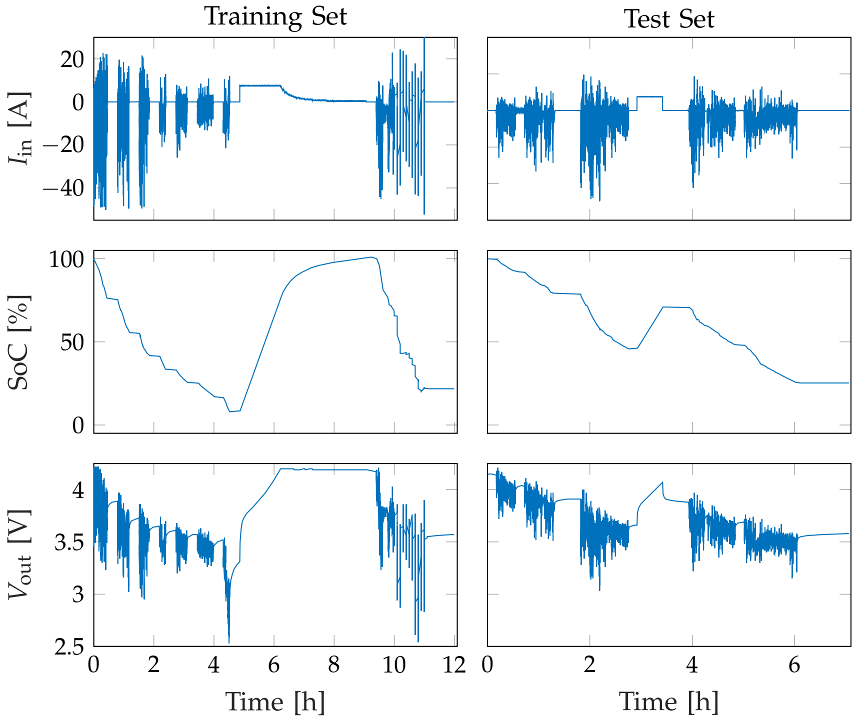


Figure 4.30: Sequences of I_{in}, SoC, and V_{out} composing the ePLB data set.

Table 4.13: List of the driving cycles composing the ePLB data set

Training Set			Test Set		
Name*	Duration [s]	Distance [km]	Name*	Duration [s]	Distance [km]
ARB02	1639	31.91	Rest	600	0
Rest	1200	0	FTP	2478	17.77
REP05	1400	32.25	SC03	600	5.76
Rest	1200	0	UDDSHDV	1060	8.94
US06	600	12.89	Rest	1800	0
SC03	600	5.76	REP05	1400	32.25
Rest	1200	0	SC03	600	17.77
HWFET	765	16.51	UDDS	1369	11.99
Rest	1200	0	Rest	600	0
UDDS	1369	11.99	Partial Charge	1800	0
Rest	1200	0	Rest	1800	0
OCC	1909	10.53	UNIF01	1930	22.05
Rest	1200	0	FTP	2478	17.77
HWFETMNT	765	16.51	HWFET	765	16.51
Rest	1200	0	OCC	1909	10.53
Full Charge	15193	0	NYCC	598	1.9
Rest	1200	0	Rest	3600	0
VEIL2NREL	5915	139.29			
Rest	3600	0			
Tot.	43318	277.64	Tot.	25387	163.24

*The acronyms refer to the driving cycles belonging to the FTP repository.

ENNC Implementation and Training

The ENNC model has been developed in *Python* by means of the framework *Keras* running on top of the backend *Theano*. In particular, the whole architecture has been implemented as a unique computational graph. Thus, it has been possible to train ENNC through the direct minimization of the RMSE between the measured and the estimated output voltage V_{out}. The training procedure has been performed with the GDBP algorithm that has been run with the optimizer *Nadam*. Indeed, preliminary tests have proved that this optimizer reaches the most effective results with respect to other alternatives such as *Adam*, *Adadelta*, or *RMSprop*. Both the training procedure and all the performed tests have been run on a laptop featuring a quad-core *Intel i7* 7770HQ 2.8 GHz and 16 GB RAM. Moreover, GDBP has been run for 2000 epochs. The *Glorot* normal procedure has been considered for initializing all the weights of ENNC, except for those related to the FLNN network. Indeed, likewise NNE, also in this case it has been preferred to initialize the weights to zero in order to avoid oscillations due to a random mixing of the functional reservoir.

It is worth noting that also in this case each internal network is actually trained in an unsupervised way even if the whole model is trained with a supervised approach. Indeed, ground truth sequences of the internal resistance R_{ist}, of the dynamic resistances $\boldsymbol{R}_{\text{dyn}}$, of the dynamic time constants $\boldsymbol{\tau}_{\text{dyn}}$, and of the quasi-stationary voltage V_{qst} are not available at training time. Consequently, the outputs of all the MLPs and FLNN are unknown, so that their learning procedure relies necessary on an unsupervised scheme. Despite that, the effectiveness of GDBP is ensured by the white box nature of the model. Indeed, the system equations defined in (4.39) force each internal network at modeling only the specific electrical component to which they are devolved, making unnecessary the knowledge of ground truth outputs.

ENNC Configurations

Several meta-parameters need to be set in order to get the most suitable configuration of ENNC. First, it is necessary to select which quantities among I_{in}, SoC, and T_{in} should compose the input tuples $\boldsymbol{u}_{\text{ist}}$, $\boldsymbol{u}_{\text{dyn}}$, and $\boldsymbol{u}_{\text{qst}}$. Second, the number of hidden layers, the number of neurons per layer, and the activation functions must be set for each of the MLPs modeling R_{ist}, $\boldsymbol{R}_{\text{dyn}}$, and $\boldsymbol{\tau}_{\text{dyn}}$. Finally, it is necessary to configure the functional reservoir of the FLNN approximating the voltage response V_{qst}. It is clear that a complete grid search of all the possible configurations of ENNC was impracticable. Therefore, only a reasonable and suitable subset of them has been analyzed, looking for the best model setup.

With regards to the input tuples, all the possible combinations of the quantities I_{in}, SoC, and T_{in} have been investigated. However, it has been imposed the same configurations for $\boldsymbol{u}_{\text{ist}}$ and $\boldsymbol{u}_{\text{dyn}}$ in order to limit the number of combinations to be tested. Consequently, every MLP share the same input tuple. Regarding FLNN, only two input configurations were available. Indeed, as explained in Section 4.5.1, the tuple $\boldsymbol{u}_{\text{qst}}$ can be composed of only SoC or it can include SoC and the temperature T_{in}. In particular, the inclusion of the temperature has been aligned with those of the inputs $\boldsymbol{u}_{\text{ist}}$ and $\boldsymbol{u}_{\text{dyn}}$.

All the MLPs share the same setup concerning the number of hidden layers, the number of neurons per layer, and the activation functions. In particular, all the hidden layers have been configured with a fixed value of 15 neurons. Conversely, the number of output neurons depends on the specific component the MLP is approximating, as explained in Section 4.5.1. Therefore, the MLP related to R_{ist} counts one output neuron, whereas those modeling $\boldsymbol{R}_{\text{dyn}}$ and $\boldsymbol{\tau}_{\text{dyn}}$ have M output neurons. Specifically, the model has been configured with $M = 2$.

Starting from this baseline configuration, a grid of eight architectures has been analyzed. More precisely, two different numbers of hidden layers and two different hidden activation functions have been tested for all the MLPs. Furthermore, two output activation functions have been analyzed for the MLPs modeling R_{ist} and $\boldsymbol{R}_{\text{dyn}}$.

First, the impact of using one or two hidden layers has been investigated. In particular, the aim was to state if a higher approximation capability were beneficial for modeling the circuit elements of ENNC. Second, besides the most common hyperbolic tangent (tanh) activation, also the more computationally efficient rectifier linear unit (ReLU) function has been investigated for the hidden layers.

The setup of the output activation functions deserves a more detailed argumentation. Concerning R_{ist} and $\boldsymbol{R}_{\text{dyn}}$, there were no particular constraints on their effective modeling and two different output functions have been tested. The first one is the standard linear (lin) activation that is commonly used in function approximation problems. However, the value of a resistance should be not negative in order to provide a physical interpretation to its functionality. Therefore, it has been chosen to force the non-negativity of R_{ist} and $\boldsymbol{R}_{\text{dyn}}$ by using a sigmoid (sig) output activation function. Conversely, the lin activation was impractible for the output layer of the MLP modeling $\boldsymbol{\tau}_{\text{dyn}}$. Therefore, only sig has been considered in this case. This choice was forced by the need of ensuring the stability of the model and of its training procedure. Indeed, the value of any time constant cannot be in any case lower or equal to zero. Otherwise, the model is unstable and the overall output can easily result in a *Not a Number* that compromises the success of the training process. Moreover, the value range of the time constants of any electrochemical cell is very wide and it can go from tens to thousands of seconds [51]. Consequently, the training procedure can be ineffective at finding a suitable modeling of the $\boldsymbol{\tau}_{\text{dyn}}$ vector. The use of the sig activation has solved both the problems. Indeed, it guarantees always a positive value for each τ_{dyn_i}, ensuring the stability of the model. Moreover, the output value range is limited from 0 to 1 and it can be easily denormalized for getting the desired actual range. Specifically, the denormalization of each τ_{dyn_i} has been implemented with the following expression:

$$\tau_{\text{dyn}_i} = \left[\tau_{\text{min}} + \tau_{\text{MLP}_i}\left(\tau_{\text{max}} - \tau_{\text{min}}\right)\right]\gamma_i \tag{4.42}$$

where τ_{dyn_i} and τ_{MLP_i} are the time constant of the i^{th} RC dipole and the i^{th}

output of the MLP, respectively; τ_{max} and τ_{min} are the maximum and minimum admissible values for τ_{dyn_i}. Moreover, it has been introduced the tuning factor γ_i in order to introduce a suitable tolerance in the denormalization process. Specifically, τ_{max} and τ_{min} have been set to 10000 s and 10 s, respectively, whereas the coefficients γ_i are tuned by the training procedure.

The configuration of the FLNN modeling the voltage response of C_{qst} has been already analyzed in Section 4.4.2 with regards to the NNE model. Indeed, it has been observed that the best results were obtained by considering a functional reservoir composed of a mixture of Chebyshev polynomials of the first kind from the zeroth to the 20th degree, and of trigonometric polynomials from the 1st to the 10th degree. However, Chebyshev polynomials are characterized by suffering significantly from the Runge's phenomenon and by having a high computational load. For this reason, it has been chosen to substitute them with Bernstein polynomials. Indeed, they allow to avoid the Runge's effect and their evaluation is much more computationally efficient with respect to Chebyshev polynomials. Therefore, FLNN has been configured with a functional reservoir composed of 21 neurons evaluating the Bernstein polynomials from the zeroth to the 20th degree, plus 20 neurons evaluating both sinusoidal and cosinusoidal polynomials from the 1st to the 10th degree. Preliminary tests have been conducted and this configuration has proved to offer a better model accuracy and a faster evaluation with respect to the Chebyshev polynomials. Similarly to NNE, the sig activation function has been considered for the output layer of FLNN. Summarizing, the overview of the considered model configurations is shown in Table 4.14.

Table 4.14: Overview on the model configurations

	R_{ist}	R_{dyn}	τ_{dyn}	C_{qst}
Network	MLP	MLP	MLP	FLNN
Input Tuple	I_{in} | SoC | T_{in}	I_{in} | SoC | T_{in}	I_{in} | SoC | T_{in}	SoC | T_{in}
Hidden Layers	1 | 2	1 | 2	1 | 2	1
Hidden Neurons	15	15	15	$21 + 20$
Hidden Activation	tanh | ReLU	tanh | ReLU	tanh | ReLU	bern + trig
Output Neurons	1	2	2	1
Output Activation	lin | sig	lin | sig	sig	sig

4.5.3 Performances Analysis

Performance Metrics

The performances have been evaluated by means of three metrics. The first one evaluates the model accuracy $\Phi_{V_{\text{out}}}$, the second performance metric Φ_{SoC}

tracks the effectiveness in the SoC estimation task, whereas the third one is a suitable fitness coefficient $\mathfrak{F}$ that summarizes the model performances jointly with its computational cost.

The first performance metric $\Phi_{V_{\text{out}}}$ aims at analyzing the overall model accuracy. Its definition is straightforward since it is automatically represented by the RMSE performed at estimating the output voltage V_{out}. Specifically, $\Phi_{V_{\text{out}}}$ has been evaluated only on Test Set sequences in order to verify the generalization capability of the model.

The effectiveness in the SoC estimation task Φ_{SoC} has been analyzed by means of a suitable SoC estimator. To this aim, SoC estimation has been performed with the SR-UKF state estimator. Specifically, the state vector has been always initialized considering a 50 % SoC and $V_{\text{dyn}} = 0$. Again, the SoC estimation task has been performed on Test Set sequences only. In order to avoid straightforward estimations, SR-UKF has been started with a temporal offset with respect to the first sample of the sequences, ensuring that the cell is in an unknown non-stationary working point at the beginning of the SoC estimation task. Two different temporal offsets have been considered aiming at analyzing not only the mere SoC estimation accuracy, but also its robustness. Indeed, this procedure has allowed to analyze the capability of SR-UKF, jointly with the considered model, at correcting the initial wrong state vector estimation starting from two different initial errors. Specifically, the temporal offsets have been set to 30 min and 60 min, respectively. The accuracy of the SoC estimation has been evaluated by means of the RMSE between the real and the estimated SoC[3]. Successively, the overall metric Φ_{SoC} has been defined as the average of the performed RMSEs among the two temporal offsets.

The definition of the model fitness $\mathfrak{F}$ has required the evaluation of the computational cost as well as of the physical memory occupancy of the models under analysis. With regards to the memory occupancy, this value has been measured by counting the total number of floating point parameters N_{par} to be memorized for executing the model.

The evaluation of the computational cost has been based on the estimation of the model latency $\mathfrak{L}$, i.e. the total number of processor clock cycles needed for propagating the inputs to the model output. In order to do that, it has been counted the overall number of additions, multiplications, comparisons, and nonlinear transformations interesting each execution of the model.

[3] The real SoC sequence has been evaluated by applying the coloumb counting algorithm starting from the beginning of the Test Set sequence and initializing expression (1.3) with the correct SoC value, i.e. $\text{SoC}[0] = 100\,\%$ for both the data sets.

Successively, the latency has been evaluated through the summation of these values weighted by the expected number of clock cycles required for executing each basic operation. Therefore, $\mathfrak{L}$ has been expressed as follows:

$$\mathfrak{L} = N_{\text{add}}\mathfrak{L}_{\text{add}} + N_{\text{mul}}\mathfrak{L}_{\text{mul}} + N_{\text{cmp}}\mathfrak{L}_{\text{cmp}} + N_{\text{nlt}}\mathfrak{L}_{\text{nlt}} \tag{4.43}$$

where N_{add}, N_{mul}, N_{cmp}, and N_{nlt} are the number of additions, multiplications, comparisons, and nonlinear transformations, respectively, whereas $\mathfrak{L}_{\text{add}}$, $\mathfrak{L}_{\text{mul}}$, $\mathfrak{L}_{\text{cmp}}$, and $\mathfrak{L}_{\text{nlt}}$ are the number of clock cycles related to each basic operation. The latter values have been set accordingly with the standard CPU latency and they are shown in Table 4.15.

Table 4.15: Clock cycles per basic operation

$\mathfrak{L}_{\text{add}}$	$\mathfrak{L}_{\text{mul}}$	$\mathfrak{L}_{\text{cmp}}$	$\mathfrak{L}_{\text{nlt}}$
2	4	2	20

Since the model approximates a dynamical system, it has been considered also its state space dimension D for a more accurate determination of the computational cost. To this aim, it has been exploited the knowledge that the model must be executed $2D + 1$ times during the prediction phase of SR-UKF. Therefore, the overall computational cost $\mathfrak{C}$ has been evaluated by multiplying the model latency $\mathfrak{L}$ by $2D + 1$:

$$\mathfrak{C} = (2D + 1)\mathfrak{L}. \tag{4.44}$$

The model fitness $\mathfrak{F}$ has been defined by taking into account $\Phi_{V_{\text{out}}}$, Φ_{SoC}, $\mathfrak{C}$, and N_{par}. However, all of these coefficients belong to a different value range. Therefore, it has been necessary to make them comparable. To this aim, each term has been scaled in the range $[0, 1]$ by applying a normalization with respect to the maximum and minimum values achieved among all the configurations of ENNC discussed in Section 4.5.2. Indicating with $\tilde{\Phi}_{V_{\text{out}}}$, $\tilde{\Phi}_{\text{SoC}}$, $\tilde{\mathfrak{C}}$, and $\tilde{N}_{\text{par}}$ the normalized coefficients, the fitness has been defined with the following expression:

$$\mathfrak{F} = \alpha_1\tilde{\Phi}_{V_{\text{out}}} + \alpha_2\tilde{\Phi}_{\text{SoC}} + \alpha_3\tilde{\mathfrak{C}} + \alpha_4\tilde{N}_{\text{par}} \tag{4.45}$$

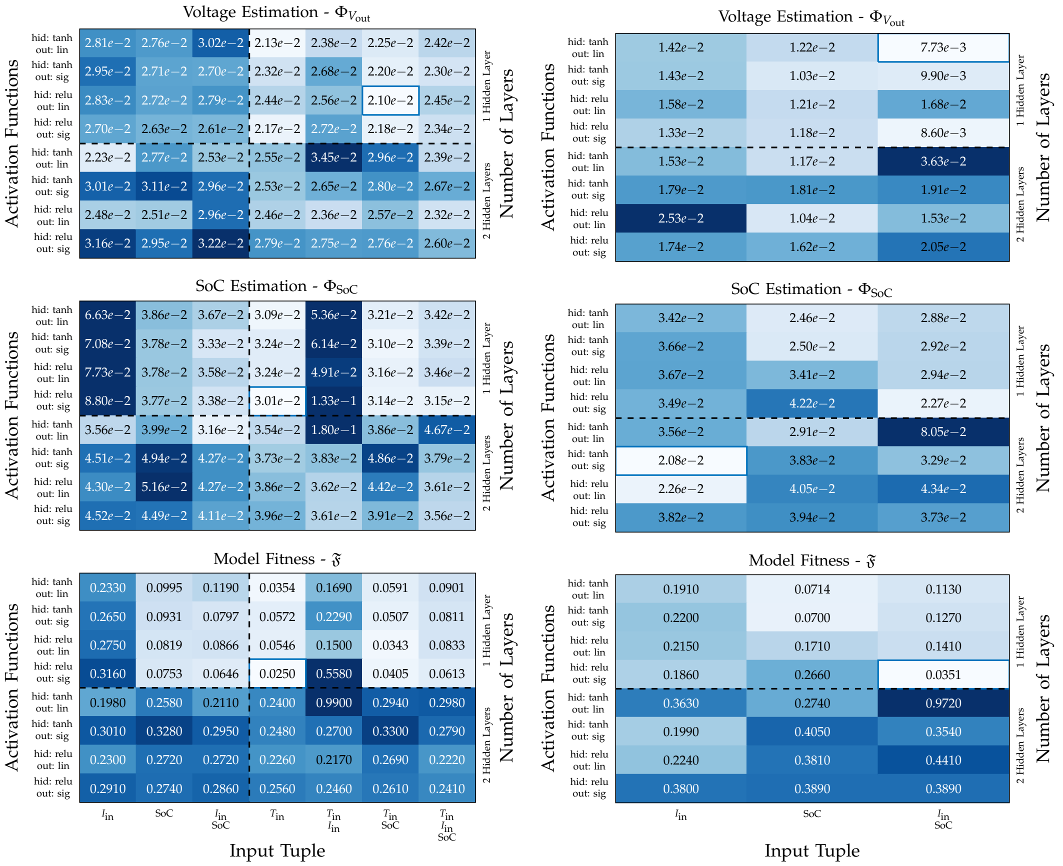
where α_1, α_2, α_3, and α_4 are suitable weighting factors subject to $\sum \alpha_i = 1$. More precisely, the previous constraint, together with the applied normalization, ensures that $\mathfrak{F}$ belongs to the same range $[0, 1]$. The weighting factors have been set as shown in Table 4.16. Herein, it has been chosen to give a greater importance to the SoC estimation accuracy since it is the most important performance being the model developed for performing this task.

Table 4.16: Weighting coefficients for the Model Fitness

α_1	α_2	α_3	α_4
0.1	0.7	0.1	0.1

Discussion on the ENNC Configurations

The value of $\Phi_{V_{\text{out}}}$, Φ_{SoC}, and $\mathfrak{F}$ performed by the considered configurations of ENNC above the NASA and the ePLB data sets are shown in Figure 4.31 part (a) and (b), respectively.



(a) (b)

Figure 4.31: Heatmap of the model performances $\Phi_{V_{\text{out}}}$, Φ_{SoC}, and $\mathfrak{F}$. (a) NASA data set. (b) ePLB data set. Brighter colors refer to a better performance.

The flexibility of the ENNC architecture is surely the first remark to draw. Indeed, several model configurations have achieved effective results concerning both the model accuracy $\Phi_{V_{\mathrm{out}}}$ and the performance in the SoC estimation task Φ_{SoC}. Thus, it can be argued that ENNC can be effectively used in a wide range of applications. Indeed, it allows to pursue the maximum performances according to the available measurements and to the specific requirements concerning model accuracy and its computational cost. Only two input configurations have appeared to be unsuitable for ENNC. The first one considers only the current I_{in} as an input of the MLPs, whereas the second configuration uses both I_{in} and T_{in} as input quantities. Indeed, ENNC has performed a significant worst Φ_{SoC} in the NASA data set with those input configurations, even considering different MLP architectures. The same conclusion can be drawn by analyzing the results performed in the ePLB data set concerning the configuration with only I_{in} as an input.

By analyzing the results achieved on the NASA data set, the model configurations can be split in four categories. Specifically, those including or not the temperature T_{in} in the input tuples $\boldsymbol{u}_{\mathrm{ist}}$, $\boldsymbol{u}_{\mathrm{dyn}}$, and $\boldsymbol{u}_{\mathrm{qst}}$, and those having one or two hidden layers in the MLPs.

The first observation is that the use of two hidden layers has caused a general loss of performances in all of the three metrics. In particular, the higher computational cost due to the presence of two hidden layers have caused a significant worse fitness $\mathfrak{F}$ even in those few cases where the two layers resulted in a better $\Phi_{V_{\mathrm{out}}}$ and/or Φ_{SoC}. Moreover, it has been also observed that including dropout layers did not outcome in any relevant or systematic performance improvements, allowing to exclude that ENNC overfits the training data. Most likely, the loss of performances of the two hidden layers configuration is related to an increased probability of GDBP to be stuck in local minima due to the higher number of weights and to the unsupervised learning that characterizes each internal MLP. Therefore, the use of only one hidden layer is preferred.

As expected, the inclusion of temperature in the input tuples is resulted in a general improvement of the model performances, even when two hidden layers have been considered. Indeed, temperature is one of the physical quantities mostly affecting the voltage response of electrochemical cells [84]. Therefore, the inclusion of T_{in} in the input tuples is almost always advantageous whenever this measurement is available. The remarkable point of the achieved results is that the temperature information has been exploited without performing specific and time consuming tests over the cell. Indeed, the most common procedure for investigating the temperature influence consists in performing several acquisition campaigns by keeping the temperature constant through climatic chambers, and by repeating the test swiping the temperature over suitable ranges. Consequently, these acquisition campaigns are very time consuming and they require accurate and expensive testing equipment. Conversely, thanks to its neural networks architecture, the proposed ENNC is able to track the temperature influence on each electrical component by considering only data acquired during the cell activity. This fact results in a more flexible, more practical, and cheaper modeling technique.

The results achieved in the ePLB data set confirm the previous observations concerning the use of one or two hidden layers. Indeed, also in this case it is evident a loss of performances when two hidden layers have been considered. Again, the use of dropout layers did not result in a performance improvement, suggesting the same tendency of GDBP at being stuck in local minima. Unfortunately, temperature measurements were not available for this data set.

Concerning the configuration of both the hidden and output activation functions, it can be seen that their choice is not affecting significantly the final model performances. Except few cases, the three performance metrics $\Phi_{V_{\text{out}}}$, Φ_{SoC}, and $\mathfrak{F}$ are close to each other in both the data sets although different activation functions have been used. Consequently, the selection among tanh|ReLU and sig|lin must be guided by the requirements of the specific application in terms of real-time constraints and model effectiveness.

With the aim of a deeper analysis, the model configurations performing the best fitness for each input tuple are shown in Table 4.17. Herein, the configurations with the overall best fitness along each data set are highlighted in bold. Table 4.17 provides a useful insight helping at discerning the performances of the activation functions. Indeed, it can be seen that the use of ReLU for the hidden layers and sig for the output layers appear to be the preferred setup. In fact, this configuration has achieved the overall best results in both the data sets, as well as it has reached most frequently the best model fitness $\mathfrak{F}$ among the different input tuples.

Table 4.17: Best model configuration per input tuple

Data	Input Tuple	Hidden Layers	Hidden Act.	Output Act.	$\Phi_{V_{\text{out}}}$	Φ_{SoC}	$\mathfrak{F}$
NASA	I_{in}	2	tanh	lin	2.23e-2	3.56e-2	0.1984
	SoC	1	ReLU	sig	2.63e-2	3.77e-2	0.0753
	I_{in}, SoC	1	ReLU	sig	2.61e-2	3.38e-2	0.0646
	$\boldsymbol{T_{\text{in}}}$	**1**	**ReLU**	**sig**	**2.17e-2**	**3.01e-2**	**0.0250**
	I_{in}, T_{in}	1	ReLU	lin	2.56e-2	4.91e-2	0.1503
	SoC, T_{in}	1	ReLU	lin	2.10e-2	3.16e-2	0.0343
	I_{in}, SoC, T_{in}	1	ReLU	sig	2.34e-2	3.15e-2	0.0613
ePLB	I_{in}	1	ReLU	sig	1.33e-2	3.49e-2	0.1859
	SoC	1	tanh	sig	1.03e-2	2.50e-2	0.0700
	$\boldsymbol{I_{\text{in}}}$, **SoC**	**1**	**ReLU**	**sig**	**8.60e-3**	**2.27e-2**	**0.0351**

4.6 Comparison Among White Box, Black Box, and Gray Box Models

The performances of all the white box, black box, and gray box models discussed in the previous sections have been compared aiming at analyzing more thoroughly which of them is the most suitable modeling technique for performing SoC estimation and more generally for being used in BMSs. To this aim, the comparison has been performed considering the performance metrics $\Phi_{V_{\text{out}}}$, Φ_{SoC}, and $\mathfrak{F}$ presented in Section 4.5.3. In particular, all the models have been tested upon the NASA and the self-collected ePLB data sets introduced in Section 4.2.2 and Section 4.5.2, respectively.

Concerning the white box models, ECM and M-ECM have been configured in accordance with what discussed in Sections 4.2.2 and 4.2.3, respectively. Thus, a total of 3 RC dipoles have been used for modeling the dynamic voltage response V_{dyn}, whereas system identification has been performed by means of MSHG-PSO. With regards to ENNC, the comparison has concerned the best model configurations shown in Table 4.17.

Some modifications have been applied concerning the implementation of the gray box NNE. First, the functional reservoir of the FLNN modeling V_{qst} has been setup in accordance with that used in ENNC in order to avoid the Runge's phenomenon and to reduce the computational cost of the model. Consequently, Bernstein polynomials have been preferred to the Chebyshev ones. Second, all the applicable input configurations have been tested also for NNE. However, only four possibilities were available for this kind of model because of the architectural constraints characterizing NNE. More precisely, the current I_{in} needs to be always included in the input tuples due to the specific architecture of the model.

Concerning the black box models, ELM, RBF-NN, and WNN have been configured and trained in accordance with what discussed in Section 4.4.3. Moreover, concerning the NASA data set it has been analyzed also their performances including the temperature T_{in} in the input tuple.

System identification has been performed considering 2000 iterations of the respective learning procedures for any of the considered models aiming at ensuring a fair comparison among them. Moreover, the values $\Phi_{V_{\text{out}}}$, Φ_{SoC}, $\mathfrak{C}$, and N_{par} have been normalized with respect to the maximum and the minimum values performed among all the models in order to get comparable values for the fitness $\mathfrak{F}$. Furthermore, the average training time and the average time for completing one SR-UKF step have been considered as further terms of comparison. Specifically, the training time has been evaluated on the laptop featuring a quad-core *Intel i7* 7770HQ 2.8 GHz and 16 GB RAM where all the models have been trained. Conversely, the SR-UKF step has been evaluated on an *Arduino Due* board by means of the same HIL architecture discussed in Section 3.1.2. In this way, all the models have been compared considering their actual temporal load for performing SoC estimation over a board with computational power closer to that of a real BMS. The obtained results are shown in Table 4.18.

The performances of the black box models confirm the conclusions drawn in Section 4.4.3. Indeed, ELM, RBF-NN, and WNN are widely resulted the worst models concerning the SoC estimation task both in the NASA and in the ePLB datasets, despite they have achieved very accurate voltage estimations. Moreover, also their fitness $\mathfrak{F}$ is resulted the worst among all the models, meaning that their simple architecture is not sufficient for counterbalancing the inadequacy at performing SoC estimation. The main reason of that has to be attributed to the overfitting of the previous output voltage sample included in the input tuple. Indeed, as explained in Section 4.4.3, the overfitting of $V_{\text{out}}[k-1]$ makes ELM, RBF-NN, and WNN very accurate models at estimating V_{out}, but it compromises the dependence of the model output on SoC, as well as it nullifies the possibility of getting any physical information about the cell behavior. As a consequence, black box models are not able to provide satisfactory results in the SoC estimation task.

A general improvement can be observed with the white box ECM. Indeed, it has achieved a better Φ_{SoC} with respect to the black box models both in the NASA and the ePLB data sets, as well as a better fitness in the NASA data set. This fact proves the importance of retrieving information about the physical

Table 4.18: Comparison among Black, White, and Gray box models

NASA data set

	Model	**Input Tuple**	$\mathbf{\Phi}_{V_{\text{out}}}$	$\mathbf{\Phi}_{\text{SoC}}$	$\mathfrak{F}$	**Training Time [min]**	**SR-UKF step [ms]**
Black Box	ELM	I_{in}, SoC, V_{out}	1.59e-2	7.87e-2	0.2508	**0.10**	**6.30**
		I_{in}, T_{in}, SoC, V_{out}	1.53e-2	6.50e-2	0.1841	**0.10**	7.00
	RBF-NN	I_{in}, SoC, V_{out}	1.22e-2	6.82e-2	0.1877	8.28	7.98
		I_{in}, T_{in}, SoC, V_{out}	1.23e-2	7.06e-2	0.1993	8.53	8.00
	WNN	I_{in}, SoC, V_{out}	**1.01e-2**	1.02e-1	0.3367	12.04	20.00
		I_{in}, T_{in}, SoC, V_{out}	1.14e-2	7.93e-2	0.2359	12.89	26.00
White Box	ECM	N/A	2.76e-2	4.01e-2	0.1783	137.90	10.01
	M-ECM	N/A	2.91e-2	1.05e-1	0.3739	159.38	18.15
	ENNC	I_{in} SoC I_{in}, SoC	2.23e-2 2.63e-2 2.61e-2	3.56e-2 3.77e-2 3.38e-2	0.2029 0.1281 0.1144	19.30	135.38 75.68 75.62
		T_{in} I_{in}, T_{in} SoC, T_{in} I_{in}, SoC, T_{in}	2.17e-2 2.56e-2 2.10e-2 2.34e-2	**3.01e-2** 4.91e-2 3.16e-2 3.15e-2	**0.0877** 0.1968 0.0960 0.1113	22.12	132.48 131.77 132.26 134.80
Gray Box	NNE	I_{in} I_{in}, SoC	2.65e-2 2.67e-2	3.82e-2 3.96e-2	0.1403 0.1513	10.94	131.81 144.11
		I_{in}, T_{in} I_{in}, SoC, T_{in}	2.58e-2 2.67e-2	4.36e-2 8.12e-2	0.1896 0.3728	20.78	249.72 250.86

ePLB data set

	Model	**Input Tuple**	$\mathbf{\Phi}_{V_{\text{out}}}$	$\mathbf{\Phi}_{\text{SoC}}$	$\mathfrak{F}$	**Training Time [min]**	**SR-UKF step [ms]**
Black Box	ELM	I_{in}, SoC, V_{out}	1.28e-2	6.83e-2	0.1542	**0.10**	**6.92**
	RBF-NN	I_{in}, SoC, V_{out}	1.00e-2	2.17e-1	0.5730	14.30	7.71
	WNN	I_{in}, SoC, V_{out}	1.08e-2	2.65e-1	0.7111	21.62	20.00
White Box	ECM	N/A	1.66e-2	5.86e-2	0.2399	235.48	10.22
	M-ECM	N/A	1.20e-2	8.25e-2	0.2011	307.27	19.00
	ENNC	I_{in} SoC I_{in}, SoC	1.33e-2 1.03e-2 **8.60e-3**	3.49e-2 2.50e-2 **2.27e-2**	0.0866 0.0565 **0.0403**	32.91	74.95 99.91 74.59
Gray Box	NNE	I_{in} I_{in}, SoC	1.62e-2 1.66e-2	4.28e-2 2.80e-2	0.1278 0.0905	18.49	140.96 143.29

behaviors of the cell. Indeed, even if ECM is not resulted the best model at estimating V_{out}, the complete insight on the cell physics, as well as the separation and identification of the voltage contributions V_{ist}, V_{dyn}, and V_{qst} have been beneficial for improving the SoC estimation accuracy.

The same conclusions cannot be drawn for M-ECM. Indeed, this model has not achieved satisfying results neither in the NASA and in the ePLB data sets. As explained in Section 4.2.3, the main reason of that has to be attributed to the high sensibility of the model performances to the accurate identification of the nonlinear function $\mathcal{C}_{\text{qst}}(.)$, so that any minimal error compromises the effectiveness of M-ECM.

The advantage of fulfilling all the three modeling properties discussed in Section 1.2 is proved by the performances achieved by ENNC and NNE. Indeed, they are the only models able to provide insight on the cell physics, to take into account the nonlinear responses of electrochemical cells, as well as to perform system identification in a flexible way all together. Specifically, it can be seen that, except for few model configurations, ENNC and NNE have achieved the best results both in the SoC estimation task and in the fitness $\mathfrak{F}$ for both the data sets. Moreover, by comparing their performances with ECM, it is clear the importance of achieving an effective modeling of the nonlinearities characterizing electrochemical cells. Indeed, although ECM provides insight on the cell physics, its effectiveness is limited by its almost linear nature. Conversely, ENNC and NNE have achieved a significant improvement in the SoC estimation accuracy thanks to the nonlinear modeling capability along with their white box and gray box properties, respectively.

Among all the models, the performances of ENNC are remarkable. Indeed, it is largely resulted the best model both in the NASA and the ePLB data sets concerning both Φ_{SoC} and $\mathfrak{F}$. In particular, ENNC has achieved the overall best Φ_{SoC} and $\mathfrak{F}$ in the NASA data set with the input tuple $\{T_{\text{in}}\}$, and it is resulted the overall best model in all the three performance metrics in the ePLB data set with the input tuple $\{I_{\text{in}}, \text{SoC}\}$. The most remarkable result is that all the model configurations have got very similar performances to each other. Moreover, ENNC has outperformed all the other models with almost every input configuration. Concerning the NASA data set, the only exception is for the input tuples $\{I_{\text{in}}\}$ and $\{I_{\text{in}}, T_{\text{in}}\}$, whereas all the configurations of ENNC have got the best results in the ePLB data set. It is remarkable to notice that ENNC is the only model that offer a complete fulfilling of all the three modeling requirements discussed in Section 1.2.

With regards to the temporal performances, as expected the black box models have got the best results concerning both the training time and the average time per SR-UKF step. In particular, ELM is resulted the overall best model in both the temporal performances thanks to its fast training procedure and its simple architecture. Conversely, ECM and M-ECM are characterized by the highest training time, being one order of magnitude greater with respect to that of the other models. Concerning NNE and ENNC, they have achieved comparable training times with respect to RBF-NN and WNN. Indeed, NNE has required about the same time, whereas the training of ENNC has lasted about twice the time than NNE, RBF-NN, and WNN, but it is still in the order of tens of minutes.

Concerning the SR-UKF step, all the simplest models, namely ELM, RBF-NN, WNN, ECM, and M-ECM, have required less than 30 ms for completing one estimation step, resulting the best models among the others. In particular, the performances of ELM, RBF-NN, and ECM are remarkable, with all of them requiring less than 10 ms per SR-UKF step. Conversely, NNE was largely penalized by its 6-D state space dimension and it has required from 131.81 ms to 250.86 ms per SR-UKF step, depending if temperature was or was not considered as model input. Also in this case ENNC has reached the best trade-off, since it has required from 74.59 ms to 144.11 ms per SR-UKF step, together with a significant better SoC estimation accuracy.

From Table 4.18 it can be confirmed that the inclusion of temperature in the input tuples is almost always beneficial, as proved by the general improvements observed in the performance metrics $\Phi_{V_{\text{out}}}$, Φ_{SoC}, and $\mathfrak{F}$. As expected, the presence of a further input has caused an increased training time and SR-UKF step, especially for ENNC and NNE. Indeed, it can be observed that the temporal performances of ELM and RBF-NN have been slightly affected by the inclusion of temperature in the input tuples both considering the training time and the SR-UKF step, whereas WNN has got similar training times, but it has shown an increased SR-UKF step of 6 ms. Conversely, the training times of ENNC and NNE are increased of about 3 and 10 min, respectively, whereas the SR-UKF step has passed from about 75 ms to 130 ms for ENNC, and from about 130 ms to 250 ms for NNE.

The above results show that the inclusion of temperature can be considered negligible with respect to the training times. Indeed, in the worst case the presence of temperature has caused only an increased training time of 10 minutes. Once again, the best trade-off with respect to the SR-UKF step has been achieved with the ENNC model. Indeed, the time for completing one SR-UKF step is about 130 ms, that allows to achieve an accurate real-time SoC estimation of 7 cells with a sampling time of 1 s. In order to increase the number of cells, it will be necessary to consider either a higher sampling time, or to use more powerful boards for the master device, considering also the hardware acceleration of the BMS algorithms by means of Field Programmable Gate Array (FPGA) devices.

Chapter 5

Hardware Implementations

5.1 BMS Prototype

The research activity has aimed at developing an advanced BMS device and related algorithms. Therefore, it has been designed a suitable BMS prototype where to implement the algorithms and the models object of the research activity. The architecture of the prototype is shown in Figure 5.1.

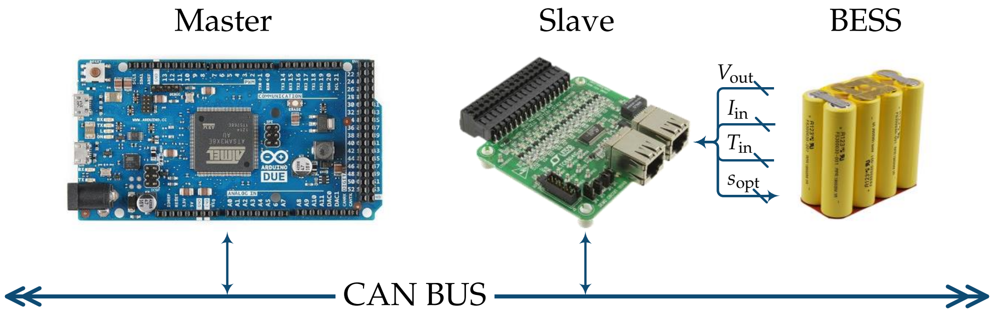


Figure 5.1: Block diagram of the BMS architecture: V_{out}, I_{in}, T_{in}, and s_{opt} refer to the terminal voltage, flowing current, working temperature, and balancing configuration, respectively.

In accordance with what explained in Section 1.3.1, the BMS prototype respects a master-slave architecture. The master device is the system core and it is in charge of performing all the computational tasks belonging to BMS. In particular, it has to estimate SoC, to determine which is the optimal configuration for cells balancing, to evaluate SoH, as well as it has to ensure each cell respects the SOA boundaries. It is clear that all of these tasks are computationally demanding and consequently the master has to provide a sufficient processing power. The main task of the slave device is to perform the required measurements over the battery pack, specifically terminal voltage, flowing current, and working temperature of each cell. Moreover, it is in charge of actuating cells balancing by applying the optimal configuration determined by the master device.

It is clear that the integration of a suitable communication protocol is mandatory for coordinating the activity of master and slave. To this aim, it has been chosen to use the CAN BUS protocol. This choice has aimed at aligning the BMS prototype to one of the most used, flexible, and reliable communication protocol, especially for the automotive industry.

It must be noticed that, depending on the specific hardware used for implementing the master and slave devices, it is possible to adapt the BMS prototype to different application fields like electric vehicles, smart grids or microgrids, as well as to different electrochemical technology.

The implementation of the BMS prototype is shown in Figure 5.2.

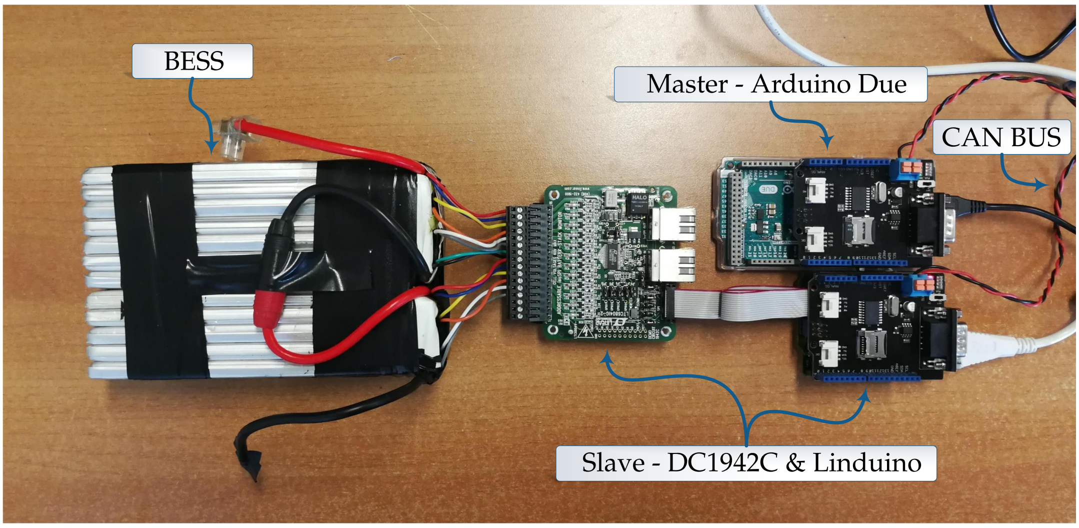


Figure 5.2: Hardware implementation of the BMS prototype.

The master device has been implemented with the *Arduino Due*[1] board, featuring the CPU *Atmel SAM3X8E ARM Cortex-M3*. It is a 32-bit *ARM* microcontroller working at a clock of 84 MHz. The board has 96 kB of SRAM and 512 kB of flash memory. Furthermore, it provides 54 general purpose I/O pins, 4 UARTs, 1 USB OTG capable connection, 2 DAC and 1 CAN interface. Moreover, the non-volatile memory can be extended through a microSD card. These specifications guarantee sufficient processing power and flexible communication capabilities for managing a battery module of about twelve series cells by executing the developed models and algorithms. If more cells must be managed, it will be necessary to use more *Arduino Due* boards or to consider more powerful devices. Moreover, for applications counting hundreds of cells could be necessary to implement the master unit by means of hardware acceleration devices such as FPGA boards.

[1] store.arduino.cc/arduino-due

The slave device has been implemented with the *DC1942C*[2] and the *Linduino One*[3] boards both from *Linear Technology*. In particular, *Linduino One* is necessary for interfacing and configuring the *DC1942C* board, that is the main element of the slave device. The *DC1942C* board features the *LTC6804-2* integrated circuit, able to monitor up to 12 cells. This capability can be expanded for monitoring a greater number of cells by connecting multiple *DC1942C* through an ISO-SPI bus. The terminal voltage of the cells is measured through two 16 bits Sigma-Delta ADCs providing a maximum absolute error of 1.2 mV. A single board requires 290 μs to measure all the twelve cells it can manage. Moreover, it is equipped with a full programmable passive balancing hardware that can achieve up to 120 mA of discharging current on the shunt resistors array. Current sensing is performed by means of the hall effect sensor *LEM CAB 300 C/SP3*[4], that is natively equipped with a CAN interface for the output reading. Temperature sensing has not been implemented yet.

As explained before, it has been chosen to implement all the communications between the master device, the slave boards, and the external system by means of the CAN-BUS protocol. This choice was guided by the aim of taking advantage of the robustness of the CAN protocol, together with uniforming the prototype to the standard communication messages related to the commercial BMSs. For this reason, this protocol has been preferred to the ISO-SPI that is built-in the slave board. In particular, the *CAN-BUS Shield V2*[5] from *Seeed Studio* has been used for equipping both the master and the slave boards with a CAN interface.

5.2 Battery cycler

Besides the BMS prototype, it has been also developed a programmable battery cycler to be used for testing and data acquisition purposes. The block diagram is shown in Figure 5.3.

The cycler is composed of three main components: generator, electronic

[2] `www.analog.com/en/design-center/evaluation-hardware-and-software/evaluation-boards-kits/dc1942c.html`

[3] `www.analog.com/en/technical-articles/meet-linduino.html`

[4] `www.lem.com/en/cab-300csp3004`

[5] `www.seeedstudio.com/CAN-BUS-Shield-V2-p-2921.html`

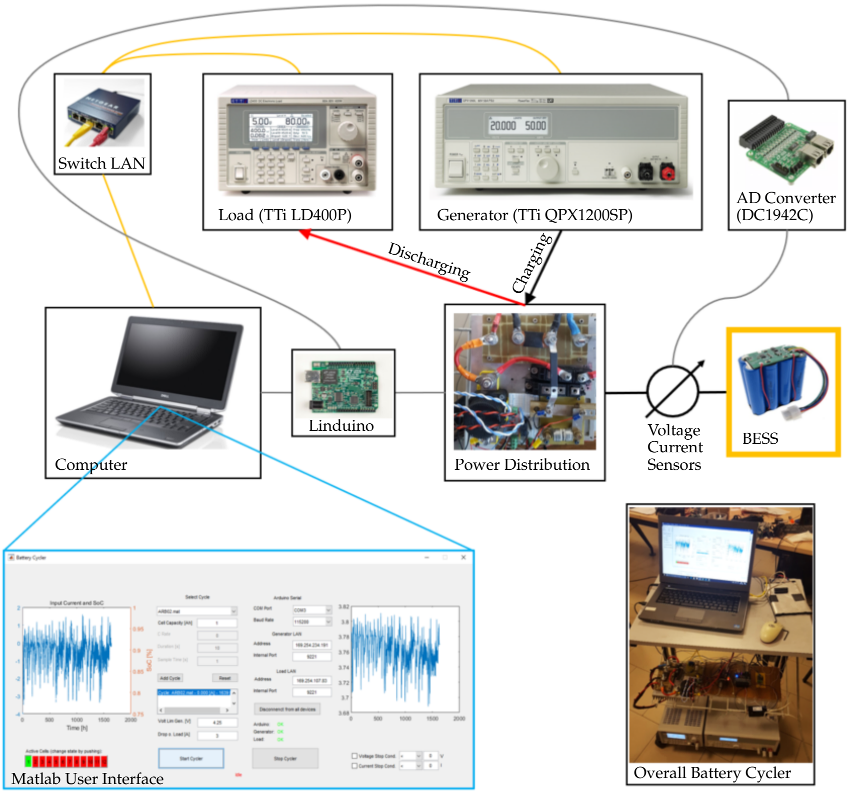


Figure 5.3: Block diagram of the self-developed programmable battery cycler.

load, and sensors. The generator and the load are used for charging and discharging the BESS, respectively, whereas the sensors are necessary for measuring voltage, current, and temperature of every cell. In particular, the generator has been implemented with the *QPX1200SP*[6] and the load with the *LD400P*[7], both from *Aim-TTi*.

The aim was to develop a testing equipment that emulates the use of BESSs in the realistic environment of an electric vehicle. Therefore, the *LD400P* has been used for emulating the current drained by the electric motor, whereas the *QPX1200SP* reproduces the regenerative breaking and the battery charging procedure. Both the load and the generator can be remotely controlled by means of an ethernet connection. Thus, the main computer, the *LD400P*, and the *QPX1200SP* have been connected through a dedicated LAN switch. The voltage and current measurements have been performed by means of the slave board *DC1942C* and the hall sensor *LEM LTS25-NP*[8], respectively. Specifically, the output voltage of the *LEM LTS25-NP* is sensed

[6] www.aimtti.com/product-category/dc-power-supplies/aim-qpxseries

[7] www.aimtti.com/product-category/electronic-loads/aim-ld400series

[8] www.lem.com/en/lts-25np

by one of the auxiliary inputs of the *DC1942C* board. Similarly to the BMS prototype, also in this case the *Linduino One* board was necessary for interfacing and configuring the *DC1942C*. Besides this task, *Linduino One* must collect the voltage and current measurements and send them to the main computer through a USB serial communication. As for the BMS prototype, also the battery cycler is still not provided with temperature sensing.

A dedicated Graphic User Interface (GUI) has been developed for the *Matlab* IDE. The GUI allows the user to easily setup the desired test and to perform it. Specifically, the interface is in charge of controlling the load and the generator in order to apply the desired current to the BESS, as well as to retrieve the measurements from the *Linduino* board.

The developed battery cycler can test BESSs having up to 12 series cells for a maximum charging and discharging power of 1200 W and 600 W, respectively. It is possible to expand the power capability by adding more *QPX1200SP* and *LD400P* devices.

5.3 Hardware in The Loop Equipment

An HIL testing equipment has been designed and implemented during the research period abroad at the *Center for Automotive Research*[9] of *The Ohio State University* with the aim of developing a suitable system for validating the BMS algorithms at managing real BESSs. In particular, the design of this testing equipment has been focused at recreating the environment conditions in which a BMS works while managing the BESS of a real electric vehicle. To this aim, the HIL equipment has needed the integration of two main devices: a suitable and powerful hardware simulator for emulating the real-time electrical and dynamical behavior of electric vehicles, and a powerful programmable battery cycler able to supply and draw an amount of power close to that interesting the electric powertrain. Moreover, the HIL equipment has been designed in order to be scalable to the size of the BESS, allowing then to test either a single cell, a battery module, or an entire BESS.

The architecture of the developed HIL equipment is shown in Figure 5.4.

The entire system is composed of four main elements: the BESS, the *dSPACE mid-size*[10] simulator, the *AV900*[11] programmable battery cycler, and the *Wilson Scott* slave board.

[9] *The Ohio State University*, Columbus, Ohio, USA.

[10] www.dspace.com/en/inc/home/products/hw/simulator_hardware/dspace_simulator_mid_size.cfm

[11] www.avtestsystems.com/av-900

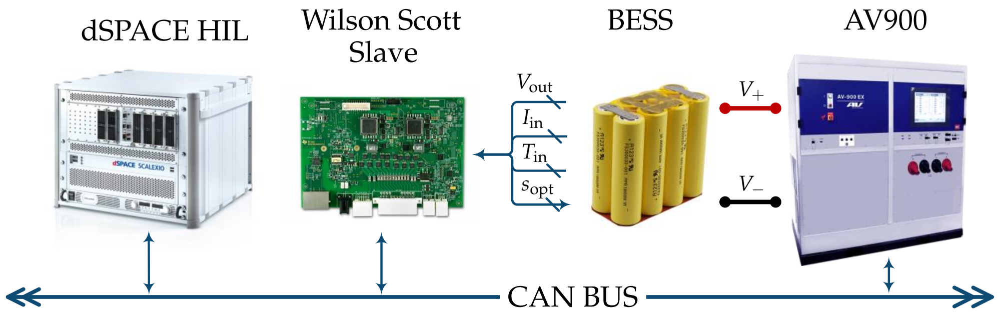


Figure 5.4: Block diagram of the HIL architecture.

The *dSPACE* simulator is a very versatile and powerful device consisting in a processing board and of an ensemble of I/O ports and communication protocols specifically developed for simulating automotive environments. In particular, the *dSPACE* device is in charge of simulating every aspect of the electric vehicle except those related to the BESS, since the HIL system is equipped with a real battery pack. Thus, *dSPACE* simulates the vehicle dynamics, the electric powertrain, as well as the control strategy. Moreover, being the HIL equipment focused on the validation of the BMS algorithms, the *dSPACE* simulator implements also the master device of BMS, including all the related managing algorithms to be validated.

The *dSPACE* device evaluates also in real-time the amount of charging/discharging power interesting the battery pack. In particular, the programmable battery cycler is in charge of providing this charging/discharging power to the BESS. Thus, the *dSPACE* simulator is connected by means of a CAN BUS protocol to the *AV900* in order to communicate the required power to be applied to the BESS at each time-step. In particular, the *AV900* device ensures a sufficient scalability to the size of BESSs, being able to supply a maximum of 900 V DC and to reach a maximum charging/discharging power of 250 kW.

As explained in the above, the HIL involves the use of a real BESS equipped with a suitable slave device. To this aim, it has been used a *Wilson Scott* board featuring 18 AD converters devolved to voltage and temperature measurements, and an array of shunt resistors for performing passive balancing. In particular, this board is directly equipped with a CAN transceiver by which it is possible to retrieve the voltage and temperature measurements, as well as to configure cells balancing. Finally, current sensing has been performed by means of a suitable shunt resistor.

The physical realization of the entire HIL testing equipment is shown in Figure 5.5.

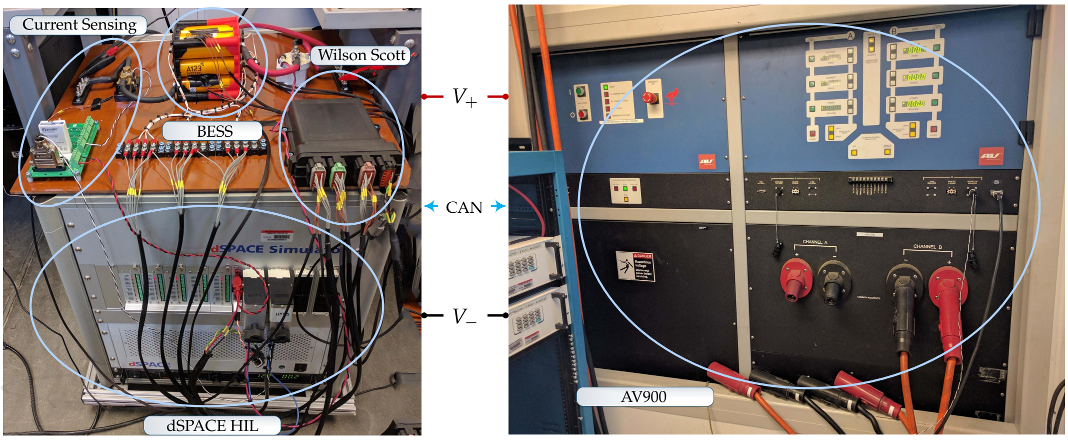


Figure 5.5: Implementation of the HIL testing equipment.

As discussed above, at the present stage the HIL equipment has been developed for validating only the BMS algorithms, so that the master device is implemented in the *dSPACE* simulator. Future works will aim at modifying the actual architecture in order to consider a real master board in order to perform a real-time validation of the entire BMS device, considering both its software and hardware components.

Chapter 6

Conclusions

Lithium BESSs are widely recognized as one of the most promising technologies for hindering climate changes and global warming because of their critical application in the automotive industry and EDS. This is because only an effective use of BESSs makes the transition from conventional propulsion to electrified powertrains possible, as well as from centralized fossil fuel EDSs to the smart grid and microgrid environment based on a wide use of RESs. For this reason, research community and companies are paying great attention to the development of effective BESSs and to the related managing systems. Considering this great research interest, this Ph.D. thesis has been focused on the design and implementation of an effective and innovative BMS device aiming at increasing the effectiveness and reliability of BESSs by means of a rigorous and effective management of them.

The research activity has involved the design of both the hardware and the software components of BMSs. From the hardware perspective, it has been developed a BMS prototype adopting a typical master-slave architecture. Specifically, the master device is in charge of performing all the computational tasks, whereas the slave board performs the required measurements and actuates cells balancing. The BMS prototype is able to monitor and manage lithium BESSs composed of up to twelve cells, but it is ready for managing a greater number of cells by using multiple slave devices. Besides the BMS prototype, two dedicated testing equipment have been developed. The first one is a programmable battery cycler that has been used for performing measurement campaigns over small BESSs by simulating the realistic environment of electric cars. The second equipment is an HIL interface that has been developed for testing and validating the performances of the BMS algorithms at managing a real BESS.

Most of the research efforts have been dedicated to the software aspects of BMS, being its effectiveness mostly depending on the algorithms running on the device. In particular, it has been chosen to deeply investigate the use of machine learning techniques because it has been a firm conviction that their data-driven nature, together with their ability of dealing with nonlinear and uncertain phenomena, would be very helpful for improving the effectiveness of BMSs.

Among the tasks of BMS, SoC estimation is surely the most critical one. Thus, it has been performed an analysis of the state observer techniques, being the most promising methods for performing SoC estimation. To this aim, it has been developed and validated an SR-UKF SoC estimator and its performances have been compared with that of the most common EKF algorithm. In particular, it has been shown that the proposed SR-UKF is a very performing algorithm that achieves not only accurate SoC estimations, but also reliable and robust results. Indeed, SR-UKF has proved to converge to the true SoC value even in front of different and consistent initialization errors. Conversely, EKF has been able to converge only if a small initial error was made.

The availability of an accurate and robust SoC estimation has permitted to develop a novel strategy for performing cells balancing. Indeed, the most common techniques used in the literature and in the industrial field evaluate the optimal configuration of the balancing circuit aiming at leveling the voltage of the cells. However, it has been discussed as this approach can be ineffective because of the lack of correlation between the terminal voltage and the working point of electrochemical cells due to parasitic voltage responses and flat OCV-SoC curves. Therefore, it has been developed a novel strategy that takes advantage of an accurate SoC estimator for performing cells balancing by leveling directly their SoC instead of their terminal voltage. To this aim, cells balancing has been formulated as a zero-one integer programming problem solved at each time-step by means of the MSHG-BPSO algorithm aiming at progressively minimizing the variance of the cells' SoC. The advantage of the proposed approach is that it is not affected by the parasitic voltage responses of the cells, allowing then to get a more effective and efficient balancing of BESSs.

It is clear that performing an accurate SoC estimation is critical not only for retrieving information about the residual energy stored in BESSs, but also for performing the other tasks of BMS, such as cells balancing. For this reason, the research activity has focused on further improving the effectiveness of SR-UKF at estimating SoC. To this aim, most of the efforts have been devolved to the development of effective and reliable models of electrochemical cells, since the accuracy of SR-UKF relies deeply on the performances of the model used in the prediction step.

The research about modeling electrochemical cells has been guided considering three main requirements that a model shall fulfill for ensuring accurate SoC estimations. First, the model shall provide useful insight on the physics of electrochemical cells in order to ensure SR-UKF has sufficient information for converging to the actual SoC value. Second, the model shall approximate as accurately as possible the nonlinearities of electrochemical cells for improving the model accuracy. Third, system identification shall be as flexible as possible for ensuring a straightforward update of the model parameters with the aging of the cells.

Black box, white box, and gray box modeling techniques have been analyzed looking for a model able to successfully fulfill all the three requirements discussed above. Concerning the black box ELM, RBF-NN, and WNN models, none of them were able to achieve satisfactory results in the SoC estimation task, even if they were characterized by a very simple and fast system identification procedure, as well as by the innate ability of modeling the nonlinearities of the cells. The main reason of this unsuitability has to be attributed to their inability at providing information about the physical behaviors of electrochemical cells. Indeed, black box models address only the overall input-output relationship of the system and they need to use a sliding windowing of the output voltage for modeling the dynamical response of the cells. This modeling architecture is easily resulted in the overfitting of the delayed output voltage, compromising the effectiveness of black box models in the SoC estimation task.

Concerning the white box ECMs, they are characterized by providing a complete insight on the cell physics. Nevertheless, the related system identification procedure is very stiff and rigid, requiring to perform very specific and long lasting tests. Therefore, the research activity has focused on the design of a more flexible system identification for ECMs that can be performed over generic data measured during the activity of the cell. In order to retrieve this result, it has been necessary to apply a wide linearization of the model, resulting in the inability of tracking most of the nonlinear behaviors of electrochemical cells. Specifically, only the nonlinear OCV-SoC relationship has been preserved. Two ECMs have been developed following this approach and for both of them system identification has been formulated as a generic fitting problem solved by means of the MSHG-PSO optimization algorithm. In particular, in the linear ECM the OCV-SoC curve has been approximated with a sampling and interpolating approach, whereas in the M-ECM it has been used a mechanical analogy and an ANFIS approximator for the same purpose. It has been observed that the insight on the physical behaviors has been helpful for improving the SoC estimation accuracy. Nevertheless, it has appeared clear also that the wide linearization of ECMs has reduced their effectiveness in the SoC estimation task.

The gray box NNE and the white box ENNC models have been developed aiming at fulfilling the three modeling requirements discussed above. Both of these approaches have aimed at taking advantage of the neural networks theory in order to achieve a very flexible system identification procedure together with nonlinear approximation capabilities. In the NNE model, three distinct and dedicated neural networks have been used for modeling and separating the three main voltage contributions of electrochemical cells, namely V_{ist}, V_{dyn}, and V_{qst}. Specifically, each neural network has been devolved and specialized to model only one of the distinct voltage responses. Conversely, ENNC has aimed at taking advantage of the strengths of both the white box and the gray box techniques by means of an hybridization between NNE and ECM. Thus, ENNC uses the ECM system equations for forcing the white box property to the model, whereas each electrical component has been modeled as a generic nonlinear device whose electrical response is approximated by a dedicated neural network.

Thanks to their architectures, both NNE and ENNC are the only models able to fulfill all the three modeling requirements. The main difference between NNE and ENNC concerns their flexibility at modeling the nonlinearities of electrochemical cells. Indeed, because of its gray box nature, NNE is subject to architectural constraints needed for allowing an effective identification and separation of V_{ist}, V_{dyn}, and V_{qst}. Unfortunately, these constraints partly limit the physical quantities that can be included in the input tuples of the neural networks, resulting in the impossibility of NNE at tracking all the possible nonlinear relationships affecting electrochemical cells. Conversely, ENNC ensures natively the separation and identification of the distinct voltage responses because of the white box system equations inherited from ECMs. As a consequence, ENNC is way more flexible than NNE, allowing also to reduce the architectural complexity of the model.

It is interesting to note that both NNE and ENNC are resulted way more accurate at performing SoC estimation with respect to all the other models, especially in the realistic ePLB data set. Moreover, ENNC is largely the overall most effective and flexible model, achieving the best performances at estimating SoC, as well as at performing the best model fitness both in the NASA and the ePLB data sets considering different model configurations. It is important to remark that these outstanding performances have been achieved without increasing excessively the computational cost of the model, and with acceptable training times. These results confirm and validate the importance of the three requirements that have been pursued for developing electrochemical cell models, being NNE and ENNC the only ones that fulfill all of them. In particular, the performances of ENNC makes particularly clear that the best model for BMSs shall show white box properties together with strong nonlinear approximation capabilities.

It has been discussed that flexible system identification has been considered a mandatory requirement of the developed models for allowing a more frequent and more flexible update of the model parameters. Indeed, the most common system identification techniques for electrochemical cells are very rigid and time consuming tasks, requiring to disassemble the BESS and to execute specific, long lasting, and expensive tests. Therefore, those procedures cannot be applied frequently, resulting in a progressive loss of effectiveness of the BMS algorithms with the aging of BESSs. Thus, it is straightforward to recognize a clear benefit in the developed flexible system identification procedures. Indeed, they allow to identify the model parameters by using data collected during the ordinary use of the battery pack, resulting in a more simple and cheap procedure that is tailored on the specific cell and its specific application, and that avoids to disassemble the BESS and to perform specific tests. This advantage is even more prominent with ENNC, because of the very promising performances and flexibility it has shown with respect to the other models.

Besides the above discussed benefits, the flexible system identification procedure and the performances of ENNC make possible to develop a very interesting scenario, consisting in a remote assistance framework for BMSs, whose conceptual diagram is shown in Figure 6.1.

The framework aims at providing remote assistance to any application involving the use of BESSs, such as electrified vehicles, microgrids, smart grids, and other portable electronic applications. It is composed of three main components: a set of services to be assisted, a cloud interface, and a remote server equipped with machine learning algorithms devolved to perform system identification of electrochemical cell models. Each of the assisted services is able to collect, communicate, and save in cloud the historic data concerning the specific workload performed by its BESS. On the other side, the server is in charge of collecting these data and to perform system identification over

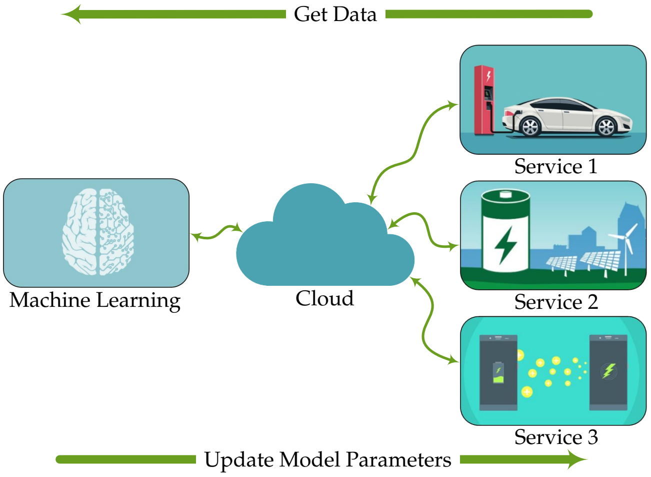


Figure 6.1: Diagram of the remote assisted BMS framework.

them in order to update the model parameters of each specific service. Once the new model parameters are ready, they are sent back to the service in order to update the model to the current status of the related BESS.

From the BMS side, the remote assistance framework will require only the implementation of a suitable logging system for allowing the memorization of the operating voltage, current, and temperature of the cells composing the BESSs. In particular, considering the data sets used for training and validating all the models discussed in this thesis and considering also the compelling performances shown by the ENNC model, it can be argued that a successful implementation of the framework can be achieved by considering a sampling time of 1 s and training sequences of about 6-12 h. Considering the above configuration, the training data of each cell can be stored in files of few MBs, so that the logging system can be implemented by means of suitable and cheap microSD cards. With regards to the communication with the cloud interface, it is possible to think about two scenarios. The first one implies that BMSs are equipped with an internet connection in order to communicate directly with the cloud interface in order to send the training data and receive back the updated model parameters. Otherwise, the communication with the cloud can be devolved to external devices being part of the application infrastructure, such as the charging stations related to the EVs context.

The remote assisted BMS framework provides several advantages. The first and most important benefit is the capability of adapting the models used in BMSs to the aging of BESSs. This fact allows to keep as high as possible the effectiveness of the BMS algorithms, as well as to increase the life of the entire

BESS. Second, the framework allows to generalize the whole BMS, since the same devices and the same framework can be used effectively independently of the specific application, size of the BESS, and electrochemical technology. Indeed, it must be noticed that all the developed models are not technology dependent, and they can be easily applied to other electrochemical cells, such as lead-acid or NiMH. Third, the framework allows to perform system identification on data measured on the exact BESS installed in the specific service. As a consequence, the identified models are tailored on the actual electrochemical cells of the BESS, as well as to the actual measurement sensors used in the specific application. Conversely, offline data acquisition campaigns are often performed on cells that are not actually used in the final BESS, as well as by using measurement sensors that are different from that of the actual application. Finally, the remote assistance framework allows to tailor the model also to the specific workload performed by the user with his own BESS.

Besides the above discussed advantages, an implicit corollary to the remote assistance framework concerns SoH estimation and second life use of automotive BESSs. Indeed, as explained before, the frequent update of the model parameters implies that the models used in BMSs are adapted to the actual aging status of electrochemical cells. Thus, it is clear that this adaptation can be exploited for retrieving accurate information about the actual SoH of BESSs. The immediate consequence is that the remote framework is a very helpful tool for tracking the SoH of several BESSs related to a wide variety of different applications. In particular, considering that automotive BESSs are considered unfeasible when they lose about 20 % of their original capacity, it is expected and encouraged a second life reallocation of dismissed automotive BESSs in different and less critical applications. In this context, the remote assistance framework will allow not only to accurately tracking the SoH of each BESS, but also to easily reallocate them to different services keeping information about their actual status and historic usage. Therefore, the framework allows not only an effective management of single BESSs related to a specific service, but also a more effective and smart management of the entire ensemble of BESSs included in the remote assistance framework.

List of Publications

- M. Luzi, F. M. Frattale Mascioli, M. Paschero, and A. Rizzi, “A white box equivalent neural network circuit model for SoC estimation of electrochemical cells”, *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–12, 2019, Accepted for publication.

- M. Luzi, M. Paschero, A. Rizzi, E. Maiorino, and F. M. Frattale Mascioli, “A novel neural networks ensemble approach for modeling electrochemical cells”, *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 2, pp. 343–354, 2019, ISSN: 2162-237X. DOI: 10.1109/TNNLS.2018.2827307.

- M. Luzi, M. Paschero, A. Rizzi, and F. M. Frattale Mascioli, “An improved PSO for flexible parameters identification of lithium cells equivalent circuit models”, in *Neural Advances in Processing Nonlinear Dynamic Signals*, A. Esposito, M. Faundez-Zanuy, F. C. Morabito, and E. Pasero, Eds., Springer International Publishing, 2019, pp. 229–238, ISBN: 978-3-319-95098-3. DOI: 10.1007/978-3-319-95098-3_21.

- A. Martino, M. Giampieri, M. Luzi, and A. Rizzi, “Data mining by evolving agents for clusters discovery and metric learning”, in *Neural Advances in Processing Nonlinear Dynamic Signals*, A. Esposito, M. Faundez-Zanuy, F. C. Morabito, and E. Pasero, Eds., Springer International Publishing, 2019, pp. 23–35, ISBN: 978-3-319-95098-3. DOI: 10.1007/978-3-319-95098-3_3.

- M. Luzi, M. Paschero, A. Rizzi, and F. M. Frattale Mascioli, “An ANFIS based system identification procedure for modeling electrochemical cells”, in *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018, pp. 1–8. DOI: 10.1109/IJCNN.2018.8489250.

- M. Luzi, M. Paschero, A. Rizzi, and F. M. Frattale Mascioli, “A binary PSO approach for real time optimal balancing of electrochemical cells”, in *2018 International Joint Conference on Neural Networks (IJCNN)*, 2018, pp. 1–8. DOI: 10.1109/IJCNN.2018.8489699.

- A. Capillo, M. Luzi, M. Paschero, A. Rizzi, and F. M. Frattale Mascioli, "Energy transduction optimization of a wave energy converter by evolutionary algorithms", in *2018 International Joint Conference on Neural Networks (IJCNN)*, Preprint, 2018, pp. 1–8. DOI: 10.1109/IJCNN.2018.8489129.

- A. Rizzi, M. Antonelli, and M. Luzi, "Instrument learning and sparse NMD for automatic polyphonic music transcription", *IEEE Transactions on Multimedia*, vol. 19, no. 7, pp. 1405–1415, 2017, ISSN: 1520-9210. DOI: 10.1109/TMM.2017.2674603.

- M. Luzi, M. Paschero, A. Rizzi, and F. M. Frattale Mascioli, "A PSO algorithm for transient dynamic modeling of lithium cells through a nonlinear RC filter", in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 279–286. DOI: 10.1109/CEC.2016.7743806.

- M. Luzi, M. Paschero, A. Rossini, A. Rizzi, and F. M. Frattale Mascioli, "Comparison between two nonlinear Kalman filters for reliable SoC estimation on a prototypal BMS", in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 5501–5506. DOI: 10.1109/IECON.2016.7794054.

Bibliography

[1] "The paris agreement", in *United Nation Climate Change Conference*, 2015. [Online]. Available: unfccc.int/process-and-meetings/the-paris-agreement/the-paris-agreement.

[2] M. Guarnieri, "Looking back to electric cars", in *2012 Third IEEE HISTory of ELectro-technology CONference (HISTELCON)*, 2012, pp. 1–6. DOI: 10.1109/HISTELCON.2012.6487583.

[3] J. Heywood, "Internal combustion engine fundamentals", McGraw-Hill, Ed., 1988.

[4] M. Ehsani, Y. Gao, and A. Emadi, *Modern electric, hybrid electric, and fuel cell vehicles: fundamentals, theory, and design*. CRC press, 2009.

[5] S. Xie, W. Zhong, K. Xie, R. Yu, and Y. Zhang, "Fair energy scheduling for vehicle-to-grid networks using adaptive dynamic programming", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1697–1707, Aug. 2016, ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2526615.

[6] P. Kalkal and V. K. Garg, "Transition from conventional to modern grids: Modern grid include microgrid and smartgrid", in *2017 4th International Conference on Signal Processing, Computing and Control (ISPCC)*, 2017, pp. 223–228. DOI: 10.1109/ISPCC.2017.8269679.

[7] A. R. Khattak, S. A. Mahmud, and G. M. Khan, "The power to deliver: Trends in smart grid solutions", *IEEE Power and Energy Magazine*, vol. 10, no. 4, pp. 56–64, 2012, ISSN: 1540-7977. DOI: 10.1109/MPE.2012.2196336.

[8] B. T. Patterson, "Dc, come home: Dc microgrids and the birth of the "enernet"", *IEEE Power and Energy Magazine*, vol. 10, no. 6, pp. 60–69, 2012, ISSN: 1540-7977. DOI: 10.1109/MPE.2012.2212610.

[9] A. Faizan, "Difference between traditional power grid and smart grid", in *Electrical Academia*, 2017. [Online]. Available: electricalacademia.com/electric-power/difference-traditional-power-grid-smart-grid/.

[10] J. Zhou, L. He, C. Li, Y. Cao, X. Liu, and Y. Geng, "What's the difference between traditional power grid and smart grid? — from dispatching perspective", in *2013 IEEE PES Asia-Pacific Power and Energy Engineering Conference (APPEEC)*, 2013, pp. 1–6. DOI: 10.1109/APPEEC.2013.6837107.

[11] Y. Ma, X. Tong, X. Zhou, and Z. Gao, "The review of smart distribution grid", in *2016 IEEE International Conference on Mechatronics and Automation*, 2016, pp. 154–158. DOI: 10.1109/ICMA.2016.7558552.

[12] X. Yu, C. Cecati, T. Dillon, and M. Simões, "The new frontier of smart grids", *IEEE Industrial Electronics Magazine*, vol. 5, no. 3, pp. 49–63, 2011, ISSN: 1932-4529. DOI: 10.1109/MIE.2011.942176.

[13] S. N. Kulkarni and P. Shingare, "A review on smart grid architecture and implementation challenges", in *2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT)*, 2016, pp. 3285–3290. DOI: 10.1109/ICEEOT.2016.7755313.

[14] G. L. Storti, M. Paschero, A. Rizzi, and F. M. Frattale Mascioli, "Comparison between time-constrained and time-unconstrained optimization for power losses minimization in smart grids using genetic algorithms", *Neurocomputing*, vol. 170, pp. 353–367, 2015, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2015.02.088.

[15] G. K. Venayagamoorthy, R. K. Sharma, P. K. Gautam, and A. Ahmadi, "Dynamic energy management system for a smart microgrid", *IEEE Transactions on Neural Networks and Learning Systems*, vol. 27, no. 8, pp. 1643–1656, 2016, ISSN: 2162-237X. DOI: 10.1109/TNNLS.2016.2514358.

[16] A. Ahmadi, G. K. Venayagamoorthy, and R. Sharma, "Performance of a smart microgrid with battery energy storage system's size and state of charge", in *2014 IEEE Symposium on Computational Intelligence Applications in Smart Grid (CIASG)*, 2014, pp. 1–7. DOI: 10.1109/CIASG.2014.7127250.

[17] O. Ellabban, H. Abu-Rub, and F. Blaabjerg, "Renewable energy resources: Current status, future prospects and their enabling technology", *Renewable and Sustainable Energy Reviews*, vol. 39, pp. 748–764, 2014, ISSN: 1364-0321. DOI: 10.1016/j.rser.2014.07.113.

[18] N. Panwar, S. Kaushik, and S. Kothari, "Role of renewable energy-sources in environmental protection: A review", *Renewable and Sustainable Energy Reviews*, vol. 15, no. 3, pp. 1513–1524, 2011, ISSN: 1364-0321. DOI: 10.1016/j.rser.2010.11.037.

[19] N. Johnstone, I. Haščič, and D. Popp, "Renewable energy policies and technological innovation: Evidence based on patent counts", *Environmental and Resource Economics*, vol. 45, no. 1, pp. 133–155, 2010, ISSN: 1573-1502. DOI: 10.1007/s10640-009-9309-1.

[20] S. Stockar, V. Marano, M. Canova, G. Rizzoni, and L. Guzzella, "Energy-optimal control of plug-in hybrid electric vehicles for real-world driving cycles", *IEEE Transactions on Vehicular Technology*, vol. 60, no. 7, pp. 2949–2962, 2011, ISSN: 0018-9545. DOI: 10.1109/TVT.2011.2158565.

[21] L. Tang, G. Rizzoni, and A. Cordoba-Arenas, "Battery life extending charging strategy for plug-in hybrid electric vehicles and battery electric vehicles**", *8th IFAC Symposium on Advances in Automotive Control AAC 2016*, vol. 49, no. 11, pp. 70–76, 2016, This work was supported by Honda R&D Co., Ltd., ISSN: 2405-8963. DOI: 10.1016/j.ifacol.2016.08.011.

[22] J. Aghaei, A. E. Nezhad, A. Rabiee, and E. Rahimi, "Contribution of plug-in hybrid electric vehicles in power system uncertainty management", *Renewable and Sustainable Energy Reviews*, vol. 59, pp. 450–458, 2016, ISSN: 1364-0321. DOI: 10.1016/j.rser.2015.12.207.

[23] D. T. Ton and M. A. Smith, "The u.s. department of energy's microgrid initiative", English, *The Electricity Journal*, vol. 25, no. 8, pp. 84–94, 2012. DOI: 10.1016/j.tej.2012.09.013.

[24] S. Leonori, E. De Santis, A. Rizzi, and F. M. Frattale Mascioli, "Multi objective optimization of a fuzzy logic controller for energy management in microgrids", in *2016 IEEE Congress on Evolutionary Computation (CEC)*, 2016, pp. 319–326. DOI: 10.1109/CEC.2016.7743811.

[25] S. Leonori, E. De Santis, A. Rizzi, and F. M. Frattale Mascioli, "Optimization of a microgrid energy management system based on a fuzzy logic controller", in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 6615–6620. DOI: 10.1109/IECON.2016.7793965.

[26] X. Luo, J. Wang, M. Dooner, and J. Clarke, "Overview of current development in electrical energy storage technologies and the application potential in power system operation", *Applied Energy*, vol. 137, pp. 511–536, 2015, ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2014.09.081.

[27] M. Paschero, L. Anniballi, G. D. Vescovo, G. Fabbri, and F. M. F. Mascioli, "Design and implementation of a fast recharge station for electric vehicles", in *2013 IEEE International Symposium on Industrial Electronics*, 2013, pp. 1–6. DOI: 10.1109/ISIE.2013.6563862.

[28] M. Ahmadi, N. Mithulananthan, and R. Sharma, "A review on topologies for fast charging stations for electric vehicles", in *2016 IEEE International Conference on Power System Technology (POWERCON)*, 2016, pp. 1–6. DOI: 10.1109/POWERCON.2016.7753886.

[29] D. Steen and L. A. Tuan, "Fast charging of electric buses in distribution systems", in *2017 IEEE Manchester PowerTech*, 2017, pp. 1–6. DOI: 10.1109/PTC.2017.7981167.

[30] S. Leonori, M. Paschero, A. Rizzi, and F. M. F. Mascioli, "An optimized microgrid energy management system based on FIS-MO-GA paradigm", in *2017 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*, 2017, pp. 1–6. DOI: 10.1109/FUZZ-IEEE.2017.8015438.

[31] S. Leonori, M. Paschero, A. Rizzi, and F. M. F. Mascioli, "Fis synthesis by clustering for microgrid energy management systems", in *Neural Advances in Processing Nonlinear Dynamic Signals*, A. Esposito, M. Faundez-Zanuy, F. C. Morabito, and E. Pasero, Eds., Springer International Publishing, 2019, pp. 61–71, ISBN: 978-3-319-95098-3. DOI: 10.1007/978-3-319-95098-3_6.

[32] H. L. Ferreira, R. Garde, G. Fulli, W. Kling, and J. P. Lopes, "Characterisation of electrical energy storage technologies", *Energy*, vol. 53, pp. 288–298, 2013, ISSN: 0360-5442. DOI: 10.1016/j.energy.2013.02.037.

[33] E. Barbour, I. G. Wilson, J. Radcliffe, Y. Ding, and Y. Li, "A review of pumped hydro energy storage development in significant international electricity markets", *Renewable and Sustainable Energy Reviews*, vol. 61, pp. 421–432, 2016, ISSN: 1364-0321. DOI: 10.1016/j.rser.2016.04.019.

[34] X. Luo, J. Wang, M. Dooner, J. Clarke, and C. Krupke, "Overview of current development in compressed air energy storage technology", *Energy Procedia*, vol. 62, pp. 603–611, 2014, 6th International Conference on Sustainability in Energy and Buildings, SEB-14, ISSN: 1876-6102. DOI: `10.1016/j.egypro.2014.12.423`.

[35] M. A. Awadallah and B. Venkatesh, "Energy storage in flywheels: An overview", *Canadian Journal of Electrical and Computer Engineering*, vol. 38, no. 2, pp. 183–193, 2015, ISSN: 0840-8688. DOI: `10.1109/CJECE.2015.2420995`.

[36] P. Alotto, M. Guarnieri, and F. Moro, "Redox flow batteries for the storage of renewable energy: A review", *Renewable and Sustainable Energy Reviews*, vol. 29, pp. 325–335, 2014, ISSN: 1364-0321. DOI: `10.1016/j.rser.2013.08.001`.

[37] G. J. May, A. Davidson, and B. Monahov, "Lead batteries for utility energy storage: A review", *Journal of Energy Storage*, vol. 15, pp. 145–157, 2018, ISSN: 2352-152X. DOI: `10.1016/j.est.2017.11.008`.

[38] N. Nitta, F. Wu, J. T. Lee, and G. Yushin, "Li-ion battery materials: Present and future", *Materials Today*, vol. 18, no. 5, pp. 252–264, 2015, ISSN: 1369-7021. DOI: `10.1016/j.mattod.2014.10.040`.

[39] W. Fischer, "State of development and prospects of sodium/sulfur batteries", *Solid State Ionics*, vol. 3-4, pp. 413–424, 1981, ISSN: 0167-2738. DOI: `10.1016/0167-2738(81)90124-7`.

[40] B. L. Ellis and L. F. Nazar, "Sodium and sodium-ion energy storage batteries", *Current Opinion in Solid State and Materials Science*, vol. 16, no. 4, pp. 168–177, 2012, Emerging Opportunities in Energy Storage, ISSN: 1359-0286. DOI: `10.1016/j.cossms.2012.04.002`.

[41] A. González, E. Goikolea, J. A. Barrena, and R. Mysyk, "Review on supercapacitors: Technologies and materials", *Renewable and Sustainable Energy Reviews*, vol. 58, pp. 1189–1206, 2016, ISSN: 1364-0321. DOI: `10.1016/j.rser.2015.12.249`.

[42] W. Buckles and W. V. Hassenzahl, "Superconducting magnetic energy storage", *IEEE Power Engineering Review*, vol. 20, no. 5, pp. 16–20, 2000, ISSN: 0272-1724. DOI: `10.1109/39.841345`.

[43] A. Kirubakaran, S. Jain, and R. Nema, "A review on fuel cell technologies and power electronic interface", *Renewable and Sustainable Energy Reviews*, vol. 13, no. 9, pp. 2430–2440, 2009, ISSN: 1364-0321. DOI: 10.1016/j.rser.2009.04.004.

[44] F. Leng, C. M. Tan, and M. Pecht, "Effect of temperature on the aging rate of li ion battery operating above room temperature", *Scientific reports*, vol. 5, p. 12 967, 2015.

[45] E. D. Santis, A. Rizzi, A. Sadeghiany, and F. M. F. Mascioli, "Genetic optimization of a fuzzy control system for energy flow management in micro-grids", in *2013 Joint IFSA World Congress and NAFIPS Annual Meeting (IFSA/NAFIPS)*, 2013, pp. 418–423. DOI: 10.1109/IFSA-NAFIPS.2013.6608437.

[46] W. Liu, *Hybrid Vehicle System Modeling and Control*. John Wiley & Sons, 2013, ISBN: 978-1-119-27932-7.

[47] J. Chiasson and B. Vairamohan, "Estimating the state of charge of a battery", *IEEE Trans. on Control Systems Technology*, vol. 13, no. 3, pp. 465–470, 2005, ISSN: 1063-6536. DOI: 10.1109/TCST.2004.839571.

[48] W.-Y. Chang, "The state of charge estimating methods for battery: A review", *ISRN Applied Mathematics*, 2013. DOI: 10.1155/2013/953792.

[49] K. S. Ng, Y. F. Huang, C. S. Moo, and Y. C. Hsieh, "An enhanced coulomb counting method for estimating state-of-charge and state-of-health of lead-acid batteries", in *INTELEC 2009 - 31st International Telecommunications Energy Conference*, 2009, pp. 1–5. DOI: 10.1109/INTLEC.2009.5351796.

[50] K. S. Ng, C.-S. Moo, Y.-P. Chen, and Y.-C. Hsieh, "Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries", *Applied Energy*, vol. 86, no. 9, pp. 1506–1511, 2009, ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2008.11.021.

[51] M. Paschero, G. L. Storti, A. Rizzi, F. M. Frattale Mascioli, and G. Rizzoni, "A novel mechanical analogy-based battery model for soc estimation using a multicell ekf", *IEEE Transactions on Sustainable Energy*, vol. 7, no. 4, pp. 1695–1702, 2016, ISSN: 1949-3029. DOI: 10.1109/TSTE.2016.2574755.

[52] S. Lee, J. Kim, J. Lee, and B. Cho, "State-of-charge and capacity estimation of lithium-ion battery using a new open-circuit voltage versus state-of-charge", *Journal of Power Sources*, vol. 185, no. 2, pp. 1367–1373, 2008, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2008.08.103.

[53] S. Abu-Sharkh and D. Doerffel, "Rapid test and non-linear model characterisation of solid-state lithium-ion batteries", *Journal of Power Sources*, vol. 130, no. 1–2, pp. 266–274, 2004, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2003.12.001.

[54] M. Paschero, V. Di Giacomo, G. Del Vescovo, A. Rizzi, and F. M. Frattale Mascioli, "Estimation of Lithium Polymer cell charachteristic parameters through genetic algorithms", in *Proc. of ICEM 2010 - International Conference on Electrical Machines*, 2010. DOI: 10.1109/ICELMACH.2010.5608060.

[55] W. He, N. Williard, C. Chen, and M. Pecht, "State of charge estimation for li-ion batteries using neural network modeling and unscented kalman filter-based error cancellation", *International Journal of Electrical Power & Energy Systems*, vol. 62, pp. 783–791, 2014, ISSN: 0142-0615. DOI: 10.1016/j.ijepes.2014.04.059.

[56] M. A. Hannan, M. S. H. Lipu, A. Hussain, M. H. Saad, and A. Ayob, "Neural network approach for estimating state of charge of lithium-ion battery using backtracking search algorithm", *IEEE Access*, vol. 6, pp. 10 069–10 079, 2018, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2018.2797976.

[57] J. C. A. Antón, P. J. G. Nieto, C. B. Viejo, and J. A. V. Vilán, "Support vector machines used to estimate the battery state of charge", *IEEE Transactions on Power Electronics*, vol. 28, no. 12, pp. 5919–5926, 2013, ISSN: 0885-8993. DOI: 10.1109/TPEL.2013.2243918.

[58] Y. Lee, W. Wang, and T. Kuo, "Soft computing for battery state-of-charge (bsoc) estimation in battery string systems", *IEEE Transactions on Industrial Electronics*, vol. 55, no. 1, pp. 229–239, 2008, ISSN: 0278-0046. DOI: 10.1109/TIE.2007.896496.

[59] H. Musoff and P. Zarchan, *Fundamentals of Kalman filtering: a practical approach*. American Institute of Aeronautics and Astronautics, 2009.

[60] G. L. Plett, "Extended kalman filtering for battery management systems of LiPB-based HEV battery packs: Part 1-2-3", *Journal of Power*

Sources, vol. 134, no. 2, pp. 252–292, 2004, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2004.02.031.

[61] O. Barbarisi, F. Vasca, and L. Glielmo, "State of charge kalman filter estimator for automotive batteries", *Control Engineering Practice*, vol. 14, no. 3, pp. 267–275, 2006, Advances in Automotive Control (AC'04), ISSN: 0967-0661. DOI: doi.org/10.1016/j.conengprac.2005.03.027.

[62] W. Yan, G. Niu, S. Tang, and B. Zhang, "State-of-charge estimation of lithium-ion batteries by lebesgue sampling-based ekf method", in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 3233–3238. DOI: 10.1109/IECON.2017.8216546.

[63] F. Sun, X. Hu, Y. Zou, and S. Li, "Adaptive unscented kalman filtering for state of charge estimation of a lithium-ion battery for electric vehicles", *Energy*, vol. 36, no. 5, pp. 3531–3540, 2011, ISSN: 0360-5442. DOI: 10.1016/j.energy.2011.03.059.

[64] J. Zhang and C. Xia, "State-of-charge estimation of valve regulated lead acid battery based on multi-state unscented kalman filter", *Journal of Electrical Power and Energy Systems*, vol. 33, no. 3, pp. 472–476, 2011, ISSN: 0142-0615. DOI: 10.1016/j.ijepes.2010.10.010.

[65] J. Du, Z. Liu, and Y. Wang, "State of charge estimation for li-ion battery based on model from extreme learning machine", *Control Engineering Practice*, vol. 26, pp. 11–19, 2014, ISSN: 0967-0661. DOI: 10.1016/j.conengprac.2013.12.014.

[66] S. J. Julier and J. K. Uhlmann, "A new extension of the kalman filter to nonlinear systems", in *Proc. of AeroSense: The 11th Int. Symp. on Aerospace/Defence Sensing, Simulation and Controls*, 1997.

[67] L. Ungurean, G. Cârstoiu, M. V. Micea, and V. Groza, "Battery state of health estimation: A structured review of models, methods and commercial devices", *International Journal of Energy Research*, vol. 41, no. 2, pp. 151–181, 2016. DOI: 10.1002/er.3598.

[68] M. Berecibar, I. Gandiaga, I. Villarreal, N. Omar, J. V. Mierlo, and P. V. den Bossche, "Critical review of state of health estimation methods of li-ion batteries for real applications", *Renewable and Sustainable Energy Reviews*, vol. 56, pp. 572–587, 2016, ISSN: 1364-0321. DOI: 10.1016/j.rser.2015.11.042.

[69] D. Haifeng, W. Xuezhe, and S. Zechang, "A new soh prediction concept for the power lithium-ion battery used on hevs", in *2009 IEEE Vehicle Power and Propulsion Conference*, 2009, pp. 1649–1653. DOI: 10.1109/VPPC.2009.5289654.

[70] S. Nejad, D. T. Gladwin, and D. A. Stone, "On-chip implementation of extended kalman filter for adaptive battery states monitoring", in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 5513–5518. DOI: 10.1109/IECON.2016.7793527.

[71] J. Wang, Z. Li, X. Li, and Y. Zhao, "A novel soh prediction framework for the lithium-ion battery using echo state network", in *Neural Information Processing*, C. K. Loo, K. S. Yap, K. W. Wong, A. Teoh, and K. Huang, Eds., Springer International Publishing, 2014, pp. 438–445, ISBN: 978-3-319-12637-1.

[72] J. Remmlinger, M. Buchholz, M. Meiler, P. Bernreuter, and K. Dietmayer, "State-of-health monitoring of lithium-ion batteries in electric vehicles by on-board internal resistance estimation", *Journal of Power Sources*, vol. 196, no. 12, pp. 5357–5363, 2011, Selected papers presented at the 12th Ulm ElectroChemical Talks (UECT):2015 Technologies on Batteries and Fuel Cells, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2010.08.035.

[73] C. Weng, J. Sun, and H. Peng, "A unified open-circuit-voltage model of lithium-ion batteries for state-of-charge estimation and state-of-health monitoring", *Journal of Power Sources*, vol. 258, pp. 228–237, 2014, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2014.02.026.

[74] C. Weng, Y. Cui, J. Sun, and H. Peng, "On-board state of health monitoring of lithium-ion batteries using incremental capacity analysis with support vector regression", *Journal of Power Sources*, vol. 235, pp. 36–44, 2013, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2013.02.012.

[75] B. G. Carkhuff, P. A. Demirev, and R. Srinivasan, "Impedance-based battery management system for safety monitoring of lithium-ion batteries", *IEEE Transactions on Industrial Electronics*, vol. 65, pp. 6497–6504, 2018.

[76] A. Eddahech, O. Briat, N. Bertrand, J.-Y. Delétage, and J.-M. Vinassa, "Behavior and state-of-health monitoring of li-ion batteries using impedance spectroscopy and recurrent neural networks", *International Journal of Electrical Power & Energy Systems*, vol. 42, no. 1,

pp. 487–494, 2012, ISSN: 0142-0615. DOI: 10.1016/j.ijepes.2012.04.050.

[77] D. Andre, A. Nuhic, T. Soczka-Guth, and D. Sauer, "Comparative study of a structured neural network and an extended kalman filter for state of health determination of lithium-ion batteries in hybrid electricvehicles", *Engineering Applications of Artificial Intelligence*, vol. 26, no. 3, pp. 951–961, 2013, ISSN: 0952-1976. DOI: 10.1016/j.engappai.2012.09.013.

[78] Y. Barsukov, "Battery cell balancing: What to balance and how", Texas Instrument, Tech. Rep., 2009.

[79] Y.-S. Lee and M.-W. Cheng, "Intelligent control battery equalization for series connected lithium-ion battery strings", *IEEE Transactions on Industrial Electronics*, vol. 52, no. 5, pp. 1297–1307, 2005, ISSN: 0278-0046. DOI: 10.1109/TIE.2005.855673.

[80] J. Cao, N. Schofield, and A. Emadi, "Battery balancing methods: A comprehensive review", in *2008 IEEE Vehicle Power and Propulsion Conference*, 2008, pp. 1–6. DOI: 10.1109/VPPC.2008.4677669.

[81] C. Solis, "External cell balancing in bq3060", Texas Instrument, Tech. Rep., 2012.

[82] D. W. Dees, V. S. Battaglia, and A. Bélanger, "Electrochemical modeling of lithium polymer batteries", *Journal of Power Sources*, vol. 110, no. 2, pp. 310–320, 2002, ISSN: 0378-7753. DOI: 10.1016/S0378-7753(02)00193-3.

[83] W. B. Gu and C. Y. Wang, "Thermal-electrochemical modeling of battery systems", *Journal of The Electrochemical Society*, vol. 147, no. 8, pp. 2910–2922, 2000. DOI: 10.1149/1.1393625.

[84] G. Fan, K. Pan, G. L. Storti, M. Canova, J. Marcicki, and X. G. Yang, "A reduced-order multi-scale, multi-dimensional model for performance prediction of large-format li-ion cells", *Journal of The Electrochemical Society*, vol. 164, no. 2, A252–A264, 2017. DOI: 10.1149/2.0791702jes.

[85] K. A. Smith, C. D. Rahn, and C.-Y. Wang, "Control oriented 1d electrochemical model of lithium ion battery", *Energy Conversion and Management*, vol. 48, no. 9, pp. 2565–2578, 2007, ISSN: 0196-8904. DOI: 10.1016/j.enconman.2007.03.015.

[86] M. Chen and G. Rincon-Mora, “Accurate electrical battery model capable of predicting runtime and i-v performance”, *IEEE Trans. on Energy Conversion*, vol. 21, no. 2, pp. 504–511, 2006, ISSN: 0885-8969. DOI: 10.1109/TEC.2006.874229.

[87] T. Tanaka, S. Ito, M. Muramatsu, T. Yamada, H. Kamiko, N. Kakimoto, and Y. Inui, “Accurate and versatile simulation of transient voltage profile of lithium-ion secondary battery employing internal equivalent electric circuit”, *Applied Energy*, vol. 143, pp. 200–210, 2015, ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2015.01.028.

[88] M. A. Roscher and D. U. Sauer, “Dynamic electric behavior and open-circuit-voltage modeling of lifepo4-based lithium ion secondary batteries”, *Journal of Power Sources*, vol. 196, no. 1, pp. 331–336, 2011, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2010.06.098.

[89] M. Charkhgard and M. Farrokhi, “State-of-charge estimation for lithium-ion batteries using neural networks and ekf”, *IEEE Transactions on Industrial Electronics*, vol. 57, no. 12, pp. 4178–4187, 2010, ISSN: 0278-0046. DOI: 10.1109/TIE.2010.2043035.

[90] Y. Shen, “Adaptive online state-of-charge determination based on neuro-controller and neural network”, *Energy Conversion and Management*, vol. 51, no. 5, pp. 1093–1098, 2010, ISSN: 0196-8904. DOI: 10.1016/j.enconman.2009.12.015.

[91] G. Dong, X. Zhang, C. Zhang, and Z. Chen, “A method for state of energy estimation of lithium-ion batteries based on neural network model”, *Energy*, vol. 90, Part 1, pp. 879–888, 2015, ISSN: 0360-5442. DOI: 10.1016/j.energy.2015.07.120.

[92] L. Xu, J. Wang, and Q. Chen, “Kalman filtering state of charge estimation for battery management system based on a stochastic fuzzy neural network battery model”, *Energy Conversion and Management*, vol. 53, no. 1, pp. 33–39, 2012, ISSN: 0196-8904. DOI: 10.1016/j.enconman.2011.06.003.

[93] J. Meng, G. Luo, and F. Gao, “Lithium polymer battery state-of-charge estimation based on adaptive unscented kalman filter and support vector machine”, *IEEE Transactions on Power Electronics*, vol. 31, no. 3, pp. 2226–2238, 2016, ISSN: 0885-8993. DOI: 10.1109/TPEL.2015.2439578.

[94] G. P. Zhang, "Neural networks for classification: A survey", *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 30, no. 4, pp. 451–462, 2000, ISSN: 1094-6977. DOI: 10.1109/5326.897072.

[95] G. Aoki and Y. Sakakibara, "Convolutional neural networks for classification of alignments of non-coding rna sequences", *Bioinformatics*, vol. 34, no. 13, pp. i237–i244, 2018. DOI: 10.1093/bioinformatics/bty228.

[96] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks", in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., Curran Associates, Inc., 2012, pp. 1097–1105.

[97] S. Yang, T. Ting, K. Man, and S.-U. Guan, "Investigation of neural networks for function approximation", *Procedia Computer Science*, vol. 17, pp. 586–594, 2013, First International Conference on Information Technology and Quantitative Management, ISSN: 1877-0509. DOI: 10.1016/j.procs.2013.05.076.

[98] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989, ISSN: 0893-6080. DOI: 10.1016/0893-6080(89)90020-8.

[99] T. Nguyen-Thien and T. Tran-Cong, "Approximation of functions and their derivatives: A neural network implementation with applications", *Applied Mathematical Modelling*, vol. 23, no. 9, pp. 687–704, 1999, ISSN: 0307-904X. DOI: 10.1016/S0307-904X(99)00006-2.

[100] S. Lu and T. Basar, "Robust nonlinear system identification using neural-network models", *IEEE Transactions on Neural Networks*, vol. 9, no. 3, pp. 407–429, 1998, ISSN: 1045-9227. DOI: 10.1109/72.668883.

[101] S. A. Billings and H.-L. Wei, "A new class of wavelet networks for nonlinear system identification", *IEEE Transactions on Neural Networks*, vol. 16, no. 4, pp. 862–874, 2005, ISSN: 1045-9227. DOI: 10.1109/TNN.2005.849842.

[102] S. Swayamsiddha, S. Mondal, and H. P. Thethi, "Identification of nonlinear dynamic systems using differential evolution based update algorithms and chebyshev functional link artificial neural network", in

Third International Conference on Computational Intelligence and Information Technology (CIIT 2013), 2013, pp. 508–513. DOI: 10.1049/cp.2013.2637.

[103] X. M. Ren and A. B. Rad, "Identification of nonlinear systems with unknown time delay based on time-delay neural networks", *IEEE Transactions on Neural Networks*, vol. 18, no. 5, pp. 1536–1541, 2007, ISSN: 1045-9227. DOI: 10.1109/TNN.2007.899702.

[104] M. Egmont-Petersen, D. de Ridder, and H. Handels, "Image processing with neural networks—a review", *Pattern Recognition*, vol. 35, no. 10, pp. 2279–2301, 2002, ISSN: 0031-3203. DOI: 10.1016/S0031-3203(01)00178-9.

[105] A. A. M. Al-Saffar, H. Tao, and M. A. Talab, "Review of deep convolution neural network in image classification", in *2017 International Conference on Radar, Antenna, Microwave, Electronics, and Telecommunications (ICRAMET)*, 2017, pp. 26–31. DOI: 10.1109/ICRAMET.2017.8253139.

[106] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity", *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943, ISSN: 1522-9602. DOI: 10.1007/BF02478259.

[107] F. Rosenblatt, "The perceptron: A probabilistic model for information storage and organization in the brain.", *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[108] G. Cybenko, "Approximation by superpositions of a sigmoidal function", *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989, ISSN: 1435-568X. DOI: 10.1007/BF02551274.

[109] K. Hornik, "Approximation capabilities of multilayer feedforward networks", *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991, ISSN: 0893-6080. DOI: 10.1016/0893-6080(91)90009-T.

[110] B. C. Csáji, "Approximation with artificial neural networks", MSc Thesis, Faculty of Sciences, Etvs Lornd University, Hungary, 2001.

[111] M. S. Gashler and S. C. Ashmore, "Training deep fourier neural networks to fit time-series data", in *Intelligent Computing in Bioinformatics*, D.-S. Huang, K. Han, and M. Gromiha, Eds., Springer International Publishing, 2014, pp. 48–55, ISBN: 978-3-319-09330-7.

[112] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)", *arXiv preprint arXiv:1511.07289*, 2015.

[113] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks", in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, Y. W. Teh and M. Titterington, Eds., ser. Proceedings of Machine Learning Research, vol. 9, Chia Laguna Resort, Sardinia, Italy: PMLR, 2010, pp. 249–256.

[114] J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks", *Neural Computation*, vol. 3, no. 2, pp. 246–257, 1991, ISSN: 0899-7667. DOI: 10.1162/neco.1991.3.2.246.

[115] A. K. Alexandridis and A. D. Zapranis, "Wavelet neural networks: A practical guide", *Neural Networks*, vol. 42, pp. 1–27, 2013, ISSN: 0893-6080. DOI: 10.1016/j.neunet.2013.01.008.

[116] Q. Zhang and A. Benveniste, "Wavelet networks", *IEEE Transactions on Neural Networks*, vol. 3, no. 6, pp. 889–898, 1992, ISSN: 1045-9227. DOI: 10.1109/72.165591.

[117] S. Dehuri and S.-B. Cho, "A comprehensive survey on functional link neural networks and an adaptive pso-bp learning for cflnn", *Neural Computing and Applications*, vol. 19, no. 2, pp. 187–205, 2010, ISSN: 1433-3058. DOI: 10.1007/s00521-009-0288-5.

[118] Y. H. Pao and Y. Takefuji, "Functional-link net computing: Theory, system architecture, and functionalities", *Computer*, vol. 25, no. 5, pp. 76–79, 1992, ISSN: 0018-9162. DOI: 10.1109/2.144401.

[119] S. Hochreiter, Y. Bengio, P. Frasconi, J. Schmidhuber, *et al.*, *Gradient flow in recurrent nets: The difficulty of learning long-term dependencies*, 2001.

[120] S. Hochreiter and J. Schmidhuber, "Long short-term memory", *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997. DOI: 10.1162/neco.1997.9.8.1735.

[121] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with lstm", *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000. DOI: 10.1162/089976600300015015.

[122] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, “Learning phrase representations using rnn encoder-decoder for statistical machine translation”, *arXiv preprint arXiv:1406.1078*, 2014.

[123] D. J. Montana and L. Davis, “Training feedforward neural networks using genetic algorithms”, in *Proceedings of the 11th International Joint Conference on Artificial Intelligence - Volume 1*, ser. IJCAI’89, Detroit, Michigan: Morgan Kaufmann Publishers Inc., 1989, pp. 762–767.

[124] J. Ilonen, J.-K. Kamarainen, and J. Lampinen, “Differential evolution training algorithm for feed-forward neural networks”, *Neural Processing Letters*, vol. 17, no. 1, pp. 93–105, 2003, ISSN: 1573-773X. DOI: 10.1023/A:1022995128597.

[125] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: Theory and applications”, *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006, Neural Networks, ISSN: 0925-2312. DOI: 10.1016/j.neucom.2005.12.126.

[126] A. Rosato, R. Altilio, and M. Panella, “Finite precision implementation of random vector functional-link networks”, in *2017 22nd International Conference on Digital Signal Processing (DSP)*, 2017, pp. 1–5. DOI: 10.1109/ICDSP.2017.8096056.

[127] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert, “Optimization and applications of echo state networks with leaky- integrator neurons”, *Neural Networks*, vol. 20, no. 3, pp. 335–352, 2007, Echo State Networks and Liquid State Machines, ISSN: 0893-6080. DOI: 10.1016/j.neunet.2007.04.016.

[128] I. Vapnyarskii, in *Encyclopaedia of Mathematics (set)*, M. Hazewinkel, Ed. Springer Netherlands, 1994.

[129] H. W. Kuhn and A. W. Tucker, “Nonlinear programming”, in *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*, Berkeley, Calif.: University of California Press, 1951, pp. 481–492.

[130] W. Karush, “Minima of functions of several variables with inequalities as side conditions”, in *Traces and Emergence of Nonlinear Programming*, G. Giorgi and T. H. Kjeldsen, Eds. Basel: Springer Basel, 2014, pp. 217–245, ISBN: 978-3-0348-0439-4. DOI: 10.1007/978-3-0348-0439-4_10.

[131] I. Zelinka, "A survey on evolutionary algorithms dynamics and its complexity – mutual relations, past, present and future", *Swarm and Evolutionary Computation*, vol. 25, pp. 2–14, 2015, SI: RAMONA, ISSN: 2210-6502. DOI: 10.1016/j.swevo.2015.06.002.

[132] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey", *Computer*, vol. 27, no. 6, pp. 17–26, 1994, ISSN: 0018-9162. DOI: 10.1109/2.294849.

[133] J. Kennedy and R. Eberhart, "Particle swarm optimization", in *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, 1995, pp. 1942–1948. DOI: 10.1109/ICNN.1995.488968.

[134] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization", *Swarm Intelligence*, vol. 1, no. 1, pp. 33–57, 2007, ISSN: 1935-3820. DOI: 10.1007/s11721-007-0002-0.

[135] M. Dorigo, V. Maniezzo, and A. Colorni, "Ant system: Optimization by a colony of cooperating agents", *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 26, no. 1, pp. 29–41, 1996, ISSN: 1083-4419. DOI: 10.1109/3477.484436.

[136] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization", *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006, ISSN: 1556-603X. DOI: 10.1109/MCI.2006.329691.

[137] D. Karaboga and B. Basturk, "A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (abc) algorithm", *Journal of Global Optimization*, vol. 39, no. 3, pp. 459–471, 2007, ISSN: 1573-2916. DOI: 10.1007/s10898-007-9149-x.

[138] C. Darwin, *On the Origin of Species by Means of Natural Selection, or the Preservation of Favoured Races in the Struggle for Life*, J. Murray, Ed. 1859.

[139] L. Zadeh, "Fuzzy sets", *Information and Control*, vol. 8, no. 3, pp. 338–353, 1965, ISSN: 0019-9958. DOI: 10.1016/S0019-9958(65)90241-X.

[140] J. R. Jang, "Anfis: Adaptive-network-based fuzzy inference system", *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 3, pp. 665–685, 1993, ISSN: 0018-9472. DOI: 10.1109/21.256541.

[141] A. Rizzi, M. Panella, and F. M. Frattale Mascioli, "Adaptive resolution min-max classifiers", *IEEE Transactions on Neural Networks*, vol. 13, no. 2, pp. 402–414, 2002, ISSN: 1045-9227. DOI: 10.1109/72.991426.

[142] G. Gerla, “Fuzzy logic programming and fuzzy control”, *Studia Logica*, vol. 79, no. 2, pp. 231–254, 2005, ISSN: 1572-8730. DOI: 10.1007/s11225-005-2977-0.

[143] E. H. Mamdani, “Application of fuzzy algorithms for control of simple dynamic plant”, *Proceedings of the Institution of Electrical Engineers*, vol. 121, no. 12, pp. 1585–1588, 1974, ISSN: 0020-3270. DOI: 10.1049/piee.1974.0328.

[144] E. Mamdani and S. Assilian, “An experiment in linguistic synthesis with a fuzzy logic controller”, *International Journal of Man-Machine Studies*, vol. 7, no. 1, pp. 1–13, 1975, ISSN: 0020-7373. DOI: 10.1016/S0020-7373(75)80002-2.

[145] M. Sugeno, *Industrial Applications of Fuzzy Control*. New York, NY, USA: Elsevier Science Inc., 1985, ISBN: 0444878297.

[146] A. Rizzi, F. M. F. Mascioli, and G. Martinelli, “Automatic training of anfis networks”, in *FUZZ-IEEE’99. 1999 IEEE International Fuzzy Systems. Conference Proceedings (Cat. No.99CH36315)*, vol. 3, 1999, pp. 1655–1660. DOI: 10.1109/FUZZY.1999.790153.

[147] M. Panella, A. Rizzi, F. M. F. Mascioli, and G. Martinelli, “Anfis synthesis by hyperplane clustering”, in *Proceedings Joint 9th IFSA World Congress and 20th NAFIPS International Conference (Cat. No. 01TH8569)*, vol. 1, 2001, pp. 340–345. DOI: 10.1109/NAFIPS.2001.944275.

[148] E. S. Peer, F. van den Bergh, and A. P. Engelbrecht, “Using neighbourhoods with the guaranteed convergence pso”, in *Swarm Intelligence Symposium, 2003. SIS ’03. Proceedings of the 2003 IEEE*, 2003, pp. 235–242. DOI: 10.1109/SIS.2003.1202274.

[149] J. Kennedy and R. C. Eberhart, “A discrete binary version of the particle swarm algorithm”, in *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 5, 1997, pp. 4104–4108. DOI: 10.1109/ICSMC.1997.637339.

[150] S. Lee, S. Soak, S. Oh, W. Pedrycz, and M. Jeon, “Modified binary particle swarm optimization”, *Progress in Natural Science*, vol. 18, no. 9, pp. 1161–1166, 2008, ISSN: 1002-0071. DOI: 10.1016/j.pnsc.2008.03.018.

[151] J. K. Uhlmann, “Dynamic map building and localization: New theoretical foundations”, PhD thesis, University of Oxford Oxford, 1995.

[152] R. V. der Merwe and E. A. Wan, “The square-root unscented kalman filter for state and parameter-estimation”, in *2001 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings (Cat. No.01CH37221)*, vol. 6, 2001, pp. 3461–3464. DOI: 10.1109/ICASSP.2001.940586.

[153] R. Tibshirani, “Regression shrinkage and selection via the lasso”, *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996, ISSN: 00359246.

[154] T Markel, A Brooker, T Hendricks, V Johnson, K Kelly, B Kramer, M O’Keefe, S Sprik, and K Wipke, “Advisor: A systems analysis tool for advanced vehicle modeling”, *Journal of Power Sources*, vol. 110, no. 2, pp. 255–266, 2002, ISSN: 0378-7753. DOI: 10.1016/S0378-7753(02)00189-1.

[155] B. Bole, C. Kulkarni, and M. Daigle, “Randomized battery usage data set”, NASA Ames Prognostics Data Repository, NASA Ames Research Center, Moffett Field, CA, Tech. Rep., 2014. [Online]. Available: ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository.

[156] T. Huria, G. Ludovici, and G. Lutzemberger, “State of charge estimation of high power lithium iron phosphate cells”, *Journal of Power Sources*, vol. 249, pp. 92–102, 2014, ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2013.10.079.

[157] V. Sangwan, R. Kumar, and A. K. Rathore, “Estimation of battery parameters of the equivalent circuit model using grey wolf optimization”, in *2016 IEEE 6th International Conference on Power Systems (ICPS)*, 2016, pp. 1–6. DOI: 10.1109/ICPES.2016.7584086.

[158] Y. Wang and L. Li, “Li-ion battery dynamics model parameter estimation using datasheets and particle swarm optimization”, *International Journal of Energy Research*, vol. 40, no. 8, pp. 1050–1061, 2016, ER-15-5937.R2, ISSN: 1099-114X. DOI: 10.1002/er.3497.

[159] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey wolf optimizer”, *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014, ISSN: 0965-9978. DOI: 10.1016/j.advengsoft.2013.12.007.

[160] Z. Cen, J. Wei, and R. Jiang, “A grey-box neural network based identification model for nonlinear dynamic systems”, in *The Fourth International Workshop on Advanced Computational Intelligence*, 2011, pp. 300–307. DOI: 10.1109/IWACI.2011.6160021.