



**UNIVERSITÀ DI PARMA**

**UNIVERSITÀ DEGLI STUDI DI PARMA**

*Dottorato di Ricerca in Tecnologie dell'Informazione*

*XXXVI Ciclo*

**Advanced State of Charge Evaluation Techniques for  
Embedded Battery Management Systems**

Coordinatore:

*Chiar.mo Prof. Marco Locatelli*

Relatore:

*Chiar.ma Prof. Valentina Bianchi*

Tutor:

*Chiar.ma Prof. Ilaria De Munari*

Dottorando: *Mattia Stighezza*

A.A. 2020/2021 - 2022/2023



*Dedicated to  
My Family  
and Friends*



# Summary

<b>Introduction</b>	<b>1</b>
<b>1 Lithium-Ion Cells</b>	<b>5</b>
1.1 Cell Structure . . . . .	6
1.2 Cell Operations . . . . .	7
1.3 State of Charge . . . . .	9
1.4 Key Concepts . . . . .	9
1.4.1 Standard Charging Procedure . . . . .	9
1.4.2 Error metrics . . . . .	11
1.4.3 Drive Cycles . . . . .	11
1.4.4 Cross-Validation . . . . .	13
<b>2 State of the Art</b>	<b>15</b>
2.1 Measure Approaches . . . . .	17
2.1.1 Coulomb Counting . . . . .	17
2.1.2 Look-Up Tables . . . . .	18
2.2 Model-Based Approaches . . . . .	20
2.2.1 Offline parameter identification . . . . .	24
2.2.2 Online parameter identification . . . . .	27
2.3 Filtering Approaches . . . . .	28
2.3.1 Kalman filters . . . . .	28
2.3.2 Particle filter . . . . .	29

---

2.3.3	Recursive Least Squares filter . . . . .	29
2.4	Data-Driven Approaches . . . . .	29
2.5	Hybrid Approaches . . . . .	31
<b>3</b>	<b>Implementation of Model-Based and Data-Driven approaches</b>	<b>33</b>
3.1	Equivalent Circuit Model and Coulomb Counting . . . . .	33
3.2	Support Vector Machines . . . . .	36
3.2.1	Support Vector Regression . . . . .	36
3.2.2	Ant Colony Optimization . . . . .	41
3.2.3	FPGA Implementation . . . . .	42
3.2.4	Results and Discussion . . . . .	43
<b>4</b>	<b>Application-Independent Training for Support Vector Machines</b>	<b>51</b>
4.1	Preliminary Approach Validation . . . . .	52
4.1.1	Data Acquisition Workbench . . . . .	52
4.1.2	Model Training . . . . .	55
4.2	Application Independent Training Data . . . . .	58
4.2.1	Improved Workbench . . . . .	58
4.2.2	SVR Training Phase . . . . .	59
4.2.3	Dataset Downsampling . . . . .	60
4.2.4	Post-Processing Stage . . . . .	61
4.2.5	Testing Phase . . . . .	64
<b>5</b>	<b>Advanced Input Features Investigation</b>	<b>71</b>
5.1	Online Electrochemical Impedance Spectroscopy . . . . .	72
5.2	EIS Dedicated Workbench . . . . .	74
5.3	Impedance Measurements . . . . .	76
5.3.1	Dataset . . . . .	76
5.3.2	Voltage Drift Correction . . . . .	76
5.3.3	SVR Training and Test . . . . .	77
5.3.4	Results and Discussion . . . . .	83

<b>Summary</b>	<b>iii</b>
<hr/>	
<b>6 Conclusions</b>	<b>87</b>
<b>A Vehicle Simulator</b>	<b>91</b>
A.1 Required Force . . . . .	91
A.2 Adherence Coefficient . . . . .	94
A.3 Required Torque . . . . .	95
A.4 Battery Power . . . . .	98
A.5 Current Consumption Profile . . . . .	99
A.6 Range Estimation . . . . .	100
A.7 Results . . . . .	101
<b>B List of Publications</b>	<b>103</b>
<b>Bibliography</b>	<b>105</b>
<b>Acknowledgments</b>	<b>125</b>





# List of Figures

1	Trends in battery demand (a) and materials and battery prices (b) . . .	3
1.1	Cell structure and operating principle . . . . .	7
1.2	Standard Constant Current Constant Voltage charging procedure with 0.5C current rate. . . . .	10
1.3	Vehicle speed as a function of time in the FTP-75 US06 drive cycle.	12
1.4	Example of a 5-fold cross-validation procedure. . . . .	13
1.5	Example of a Hold Out cross-validation. . . . .	14
2.1	State of Charge estimation techniques . . . . .	16
2.2	Example of Look-Up Table storing OCV-SOC data pairs as a func- tion of the temperature T . . . . .	19
2.3	Open Circuit Voltage - State of Charge characteristic at different tem- peratures . . . . .	20
2.4	Electrical Equivalent Circuit Model - Rint model . . . . .	22
2.5	Electrical Equivalent Circuit Model - Thevenin model . . . . .	22
2.6	Dynamic behaviour of cell voltage in response to an input current step	23
2.7	Example of pseudo-linear relationship in a reduced interval. . . . .	26
2.8	Typical voltage curve versus cell State of Charge . . . . .	26
2.9	Kalman Filter variants . . . . .	28
3.1	System overview for a hybrid approach based on Equivalent Circuit Model, Coulomb Counting and online parameter identification. . . . .	34

3.2	Support Vector Machines for binary classification . . . . .	37
3.3	Supervised and Unsupervised Learning . . . . .	38
3.4	Support Vector Machines for regression tasks . . . . .	39
3.5	Proposed Ant-Colony-Optimized SVR system architecture . . . . .	42
3.6	Absolute Error compared with the target US06 SoC for the Vivado fixed-point behavioural simulation and the MATLAB floating-point double precision. . . . .	47
3.7	Comparison of expected SoC with SoC processed data on the FPGA board. . . . .	49
4.1	Data acquisition schematic overview. . . . .	52
4.2	Data acquisition workbench for constant current measurements. . . . .	54
4.3	SVR training and testing workflow. . . . .	56
4.4	(a) Comparison between the reference CC computed SoC and the SVR estimated SoC. (b) Detail of the SoC estimation profile . . . . .	57
4.5	Improved workbench for data acquisition. . . . .	59
4.6	Prefiltered SoC (blue line) compared with the filtered SoC signal (red line). . . . .	62
4.7	SoC value at the output of the system (red line) compared to the reference (blue line). In the box, the absolute error. . . . .	66
4.8	Workbench for parallel data acquisition on multiple cells. . . . .	69
5.1	Typical cell impedance spectrum . . . . .	72
5.2	Sinewave component for the multi-sine signal. . . . .	73
5.3	Schematic overview of the proposed workbench for EIS dataset. . . . .	74
5.4	Detailed EIS workbench. . . . .	75
5.5	Voltage data comparison before (a) and after (c) the drift correction a single acquisition period. . . . .	78
5.6	Voltage FFT spectrum corresponding to a single acquisition period before (a) and after (b) the drift compensation. . . . .	79
5.7	Magnitude of the impedance for cell number 1 discharged at 2 A as a function of the SoC. . . . .	80

---

5.8	Magnitude of the impedance for cell number 1 at different current rates.	81
5.9	State of Charge estimated by SVR trained with (green line) and without (red line) the impedance information at 100 Hz compared with the reference CC SoC. . . . .	84
A.1	Tesla Model S 85 kWh battery pack. . . . .	92
A.2	Schematic representation of the forces acting on a moving vehicle. .	93
A.3	NEDC drive cycle speed setpoints (blue line) and the corresponding cell current (red line). . . . .	102



# List of Tables

3.1	Area Occupation from Synthesis Elaboration . . . . .	35
3.2	Panasonic NCR18650 Electrical Characteristics . . . . .	40
3.3	RMSE Results for Four SVR Kernels Trained on the NN Drive Cycle Dataset and Tested on the US06 Drive Cycle Data. . . . .	44
3.4	RMSE and MAE Results Over 15 Runs of Different Optimization Algorithms. . . . .	45
3.5	Summary of Datasets Used in Each Evaluation Step, for Training and Test. . . . .	46
3.6	Area Occupation Comparison with Other Studies. . . . .	48
3.7	Comparison of Performance in SoC Estimation Between SVR Approach, ECM Approach Either with Constant and SoC-Varying Parameters, and the Literature. . . . .	48
4.1	Digital Multimeter Settings and Specifications . . . . .	53
4.2	SVR Model Error Metrics on Different Discharge Cycles . . . . .	57
4.3	Effects of Data Downsampling on Training Time and Accuracy . . . . .	61
4.4	Test Results with Random Dynamic Profiles in the Range 3A-4A . . . . .	65
4.5	Test Results with Random Dynamic Profiles in Different Current Ranges . . . . .	67
4.6	Comparison with the State of the Art . . . . .	67
5.1	Input Vectors for Each Cell and Each Current Rate. . . . .	81

5.2	Different Input Features for SoC Estimation on the Same Cell . . .	83
A.1	Comparison Between NEDC Ratings and Simulation Results for a Tesla Model S 85kWh. . . . .	101

# List of Acronyms

**ACO** Ant Colony Optimization.

**AE** Absolute Error.

**Ah** Ampere-hour.

**AI** Artificial Intelligence.

**BMS** Battery Management System.

**CC** Coulomb Counting.

**CC-CV** Constant Current Constant Voltage.

**DDM** Data-Driven Model.

**DL** Deep Learning.

**DMM** Digital MultiMeter.

**ECM** Equivalent Circuit Model.

**EIS** Electrochemical Impedance Spectroscopy.

**ESR** Equivalent Series Resistance.

**EV** Electric Vehicle.

**FFT** Fast Fourier Transform.

**FPGA** Field Programmable Gate Array.

**HDL** Hardware Description Language.

**HPPC** Hybrid Pulse Power Characterization.

**KF** Kalman Filter.

**LFP** Lithium iron phosphate.

**Li** Lithium.

**LIB** Lithium-Ion Battery.

**LUT** Look-Up Table.

**MAE** Mean Absolute Error.

**ML** Machine Learning.

**MSE** Mean Squared Error.

**MWLS** Moving Window Least Squares.

**NN** Neural Network.

**OCV** Open Circuit Voltage.

**RLS** Recursive Least Squares.

**RMSE** Root Mean Squared Error.

**SLR** Simple Linear Regression.

**SoC** State of Charge.



**SoH** State of Health.

**SV** Support Vectors.

**SVM** Support Vector Machine.

**SVR** Support Vector Regression.



# Introduction

The rapid evolution of modern society in the last decades has been heavily dependent on the development of portable electronic devices, the electrification of transportation, and the integration of renewable energy sources into the power grid. The demand for energy storage options offering high energy density and fast charge/discharge capabilities has increased over the years. First commercialised in the early 1990s, Lithium-Ion Batteries (LIBs) have emerged as the leading technology among the existing electrochemical energy storage options, e.g., Nickel-Metal Hydride (NiMH) or lead-acid batteries. Their lightweight design and the consistently high energy density over numerous charge and discharge cycles meet the requirements set by recent innovations in different market fields. In portable and consumer electronics, from smartphones and laptops to cordless power tools, they substituted the large and bulky NiMH batteries. The high energy density and fast charging capabilities completely replaced the heavy lead-acid batteries in Electric Vehicles (EVs), providing sufficient energy for long-range driving and revolutionising the automotive industry. Moreover, their long cycle life with minimal capacity degradation and low self-discharge makes them ideal for long-term applications, hence, they found applications in grid energy storage systems. LIBs can manage the intermittency of renewable energy sources, enhancing grid stability and facilitating the transition towards cleaner and more sustainable energy sources.

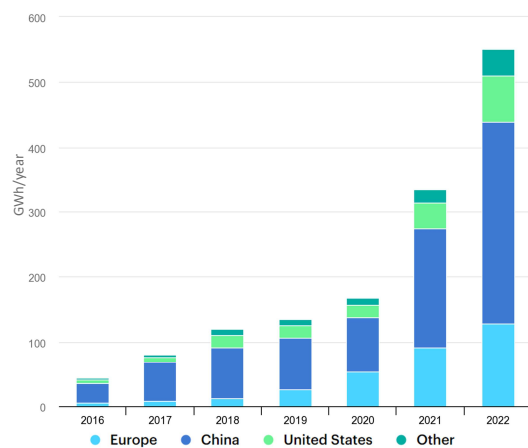
The market for LIBs is growing rapidly, and the cost of batteries is decreasing (as shown in Fig. 1) [1]. This trend is expected to continue, leading to further expansion of the market in the coming years. In fact, the battery market size has already reached

USD 45.0 billion in 2022 and is projected to grow to USD 93.3 billion by 2028 [2].

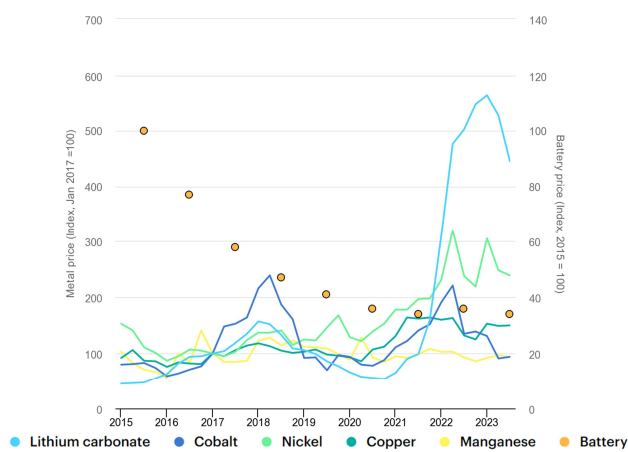
Lithium-ion batteries also have certain limitations and disadvantages. Some environmental concerns have arisen in recent years due to the production and dismantling of batteries. Lithium resources are finite, and the rapid growth in demand for LIBs has raised concerns about the sustainability of lithium mining. In addition, they have a longer cycle life than other battery types, but at the end of their life, they must be recycled appropriately to mitigate environmental impacts. Moreover, the lithium-ion batteries present some safety concerns related to proper usage. When damaged, overcharged or over-discharged, LIBs can occur in a thermal runaway event, leading to safety hazards due to catching fire and potential explosions.

A Battery Management System (BMS) is crucial in monitoring the battery charging and discharging processes. For the safe operation of a Li-ion battery, the BMS can also provide thermal management to prevent thermal damage. Other than for safety purposes, properly operating the battery also maximises the storable charge, prevents early turn-off, and improves the battery cycle life. The BMS algorithms heavily rely on the proper identification of the battery state in order to manage operations accurately and safely. Two main indicators are monitored: the State of Charge (SoC) and the State of Health (SoH). The SoC represents the percentage of available charge compared to the maximum battery capacity, as it is crucial to operate the battery in the nominal range and avoid overcharging and over-discharging. The SoH, on the other hand, represents the remaining available capacity compared to the manufacturer's rated capacity, and it is, therefore, an estimate of battery ageing. Typically, BMSs evaluate the SoC to provide the user with an estimate of the remaining usage time, e.g., the driving range of an EV or the remaining charge in a smartphone. More recently, some high-end devices have introduced the SoH to warn the user of the battery status and suggest a replacement. In most cases, this is still a feature unavailable to the user.

Developing sophisticated BMSs allows for a more accurate battery state estimation. Unfortunately, both indicators are strictly related to complex electrochemical reactions within the battery structure, making direct measurements unfeasible. Therefore, they have to be estimated by a properly designed algorithm that evaluates some



(a)



(b)

**Figure 1:** Trends in battery demand (a) and materials and battery prices (b)

directly measurable quantities, such as voltage, current and temperature.

This dissertation aims to investigate the state of the art of BMS algorithms for the SoC estimation and validate the implementation of new techniques. Battery data for this purpose have been gathered from remotely controlled workbenches specifically designed for this investigation. The obtained datasets have been then employed to evaluate algorithms' effectiveness for embedded systems.

The dissertation is organised as follows:

- Chapter 1 provides some insights on the lithium-ion technology.
- In Chapter 2, the state of the art for the SoC estimation is investigated.
- In Chapter 3, an Equivalent Circuit Model (ECM) and a Machine Learning (ML) algorithm are designed for the SoC estimation to be implemented on an FPGA platform.
- In Chapter 4, the ML approach is extended to an application-independent scenario to improve algorithm generalization.
- In Chapter 5, a novel set of input features is investigated for the training of ML algorithms.
- In Chapter 6, the investigated approaches are summarized and conclusions on this dissertation work are drawn.
- In Appendix A, the fundamentals of vehicle dynamics are delineated and the procedures behind the designed vehicle simulator are detailed.
- In Appendix B, a list of publications based on the proposed approaches is provided.

# Chapter 1

## Lithium-Ion Cells

An introduction to the Lithium-Ion chemistry field follows, outlining the key concepts. A Lithium-Ion battery can comprise several primitive blocks called *cells*. A combination of cells is usually called a *battery pack*, and its voltage and capacity depend on the chosen series and parallel configuration. In this dissertation, the term *cell* will be used to refer to the individual electrochemical unit while referring to the group of connected cells with the term *battery* or *battery pack*.

The Lithium-ion typical nominal voltage is in the magnitude order of 3.6 V. The sum of the series-connected cell voltages will determine the battery pack's voltage. In contrast, the number of parallel-connected cells determines the battery pack capacity. The cell capacity is the storable charge expressed in Ampere-hours (Ah). Li-Ion cell datasheets typically provide information on the maximum discharge current that the cell can handle, whether pulsed or continuous. This current data is usually expressed in terms of a parameter known as the C-rate. The C-rate represents how fast the cell is charged or discharged relative to its nominal capacity. For example, a 1C rate means the cell is charged or discharged in one hour. This implies that a cell with a nominal capacity of 2000 mAh would be discharged with a current of 2000 mA (i.e., 2 A). Similarly, if the same cell had a 0.5C rate, it would take two hours to charge or discharge, with a current of 1000 mA (i.e., 1 A). It's important to note that high C-rates exceeding the datasheet limitations can cause heat and stress, ultimately

reducing the cell's lifespan.

Li-Ion cells are available in various form factors, different from the commonly known AA size. The most common form factors are the 18650 and the 26650. The name convention represents the size of the diameter (18) and the length (65), respectively, in millimetres. Different-sized cells have been recently designed to increase the energy density.

## 1.1 Cell Structure

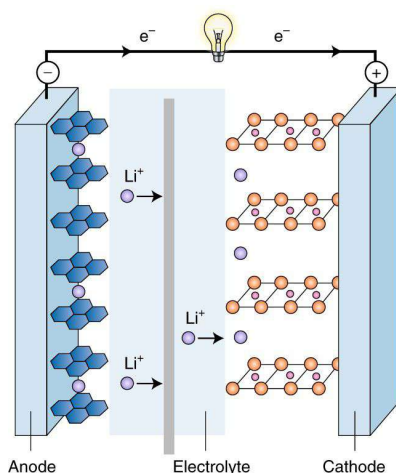
Lithium-ion cells consist of four primary components: a positive electrode (cathode), a negative electrode (anode), a separator and an electrolyte. The operation of LIBs relies on the reversible intercalation of positively charged lithium ions ( $\text{Li}^+$ ) between the anode and cathode materials during charge and discharge cycles. The more lithium ions can intercalate in the material structure, the more capacity the cell can have. Therefore, materials composing both the electrodes must facilitate the intercalation in their crystal structure without being chemically changed by the insertion of the lithium ion.

Much research is being done on finding the optimum materials, but graphite is often used as the anode material (lithiated graphite  $\text{LiC}_6$ ) [3], which has a hexagonal layered structure favourable to lithium ions intercalation.

The cathode is a lithiated metal oxide that mostly affects the cell's performance. Common cathode materials include lithium cobalt oxide ( $\text{LiCoO}_2$ ), lithium iron phosphate ( $\text{LiFePO}_4$ , or LFP), and lithium nickel-cobalt-manganese (NCM) compounds [4]. The manufacturers choose the cathode material according to the quality they want to improve, e.g., cell capacity, high current, or long life cycle, but cobalt is rare and toxic, hence emerging alternatives are preferred.

The electrolyte, typically lithium salts dissolved in non-aqueous organic solvents, does not participate in the chemical reactions but only facilitates the movement of lithium ions between the electrodes. At the same time, it works together with the separator to prevent electrons from passing, forcing them to flow through external circuitry wires. The separator membrane not only physically isolates the positive and





**Figure 1.1:** Cell structure and operating principle

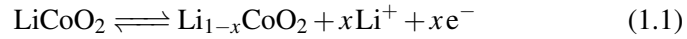
negative electrodes but is also an ionic conductor and an electronic insulator. This prevents internal short-circuiting between the two electrodes [5]. The simplified representation of a lithium-ion cell structure is shown in Fig. 1.1.

## 1.2 Cell Operations

The electrochemical potential energy at the electrodes causes a potential difference. Therefore, the cell nominal voltage is related to how the employed structure materials interact. Oxidation and reduction reactions occur inside the cell during charge and discharge operations. When the cell is being charged, oxidation takes place in the cathode. In (1.1), a  $\text{LiCoO}_2$  cathode is considered, but the chemical balance applies to different metal oxides as well. The resulting lithium ion moves through the electrolyte and embeds into the anode. Consequently, a free electron flows through the wires from the cathode to the anode, resulting in a current flow in the opposite direction. Meanwhile, on the anode, the combination of the lithium ion and the electron results

in a reduction reaction (1.2). The  $x$  represents the degree of lithiation, close to 1 when the cell is charged or 0 when discharged and the negative electrode is largely depleted of lithium ions.

The process is completely reversible and reverses when the cell is discharging. As shown in Fig. 1.1, lithium ions and electrons move toward the cathode along different paths, causing a current flow from the positive electrode toward the negative electrode.



Many side effects happen during these operations, which affect the cell capacity and the voltage measured at the external terminals. First, lithium ions move inside the electrolyte with a certain resistance, which results in a voltage drop affecting the measured voltage. Moreover, a chemical polarization causes the effect known as *voltage relaxation*, whose duration is related to the temperature and the electrode materials [6].

When the electrolyte solvents react with graphite, they form an irreversible Solid Electrolyte Interface (SEI) at the anode surface. The SEI is a layer that contains products of electrolyte decomposition and acts as a passivation layer. A stable SEI layer helps to improve the performance of the reaction by effectively insulating the electrons while allowing ions to move freely. It is also vital for stabilizing the electrode as it prevents the solvents from reacting further with the underlying graphite. However, if the SEI layer is unstable, it can continuously consume electrolytes during consecutive cycles, increasing its thickness and reducing the ions' ability to intercalate into the anode. This results in capacity fading and increased cell resistance, consequently limiting the cell charging/discharging efficiency and accelerating its degradation, ultimately shortening its operational lifespan, in a process commonly known as *ageing*. Using the current, voltage or temperature data to directly evaluate the SoC or SoH of a cell is unfeasible due to these complex electrochemical reactions and these internal side effects, but an estimation can be performed by correctly modelling or predicting the cell behaviour. In a battery pack, each cell must be in-

dividually monitored thus a properly designed Battery Management System must be implemented. In the following chapters, we will investigate the state of the art for the SoC estimation in BMSs and evaluate the implementation of new techniques.

## 1.3 State of Charge

The State of Charge (SoC) is a vital parameter for proper cell operations and is the main indicator employed in management algorithms. The SoC is defined as a unitless quantity, being 100% (or 1) when the cell is fully charged and 0% (or 0) when fully discharged. The SoC formulation is given by equation (1.3), which must be multiplied by a factor of 100 to obtain a percentage.

$$SOC = \frac{Q}{Q_{tot}} \quad (1.3)$$

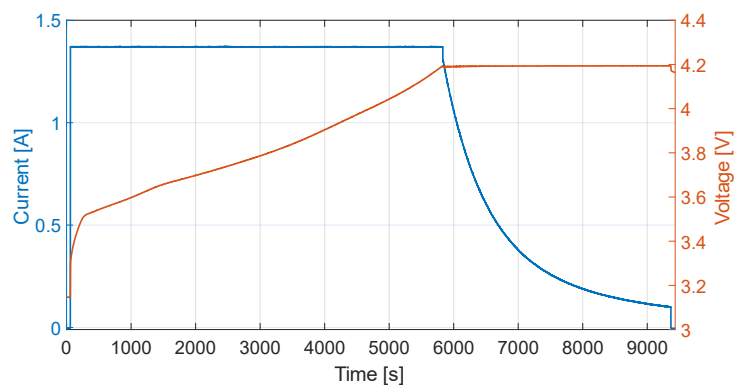
The remaining charge,  $Q$ , is related to the cell total charge capacity  $Q_{tot}$  (or total capacity), which is defined in Ampere-hour (Ah). The value of  $Q_{tot}$  may vary from cell to cell due to the materials employed in the electrodes and the format factor. It also differs between two identical brand-new cells due to tolerances in the manufacturing process, although the process has become more and more precise over the years. When the manufacturer provides the total capacity, it is referred to as *nominal capacity*  $Q_{nom}$ . The total capacity tends to decrease gradually as the cell ages due to parasitic side reactions (such as SEI formation) and degradation of electrode material. Therefore, the total capacity should be periodically characterised, when possible, to preserve the accuracy of the BMS algorithms.

## 1.4 Key Concepts

### 1.4.1 Standard Charging Procedure

There are three commonly used charging methods:

- Constant Current. This is the simplest method employing constant current until a defined voltage is reached. A higher current allows for faster charging, but



**Figure 1.2:** Standard Constant Current Constant Voltage charging procedure with 0.5C current rate.

the voltage threshold is reached earlier due to the increased internal voltage drop. Therefore, the charging current is a trade-off between the charging speed and the quantity of stored charge.

- Constant Voltage. The voltage is kept constant, and the current decreases as the battery charges. Over-voltage is prevented, but the higher current at the start of the process may damage the cell.
- Constant Current Constant Voltage (CC-CV). This is the standard charging procedure used as a reference in literature. The cell is charged at a constant current until its voltage reaches the upper threshold, usually defined in the data-sheet. After reaching the upper voltage threshold, the voltage is kept constant at this level, and the charging current gradually decreases. The cell is considered fully charged when the current drops below the specified minimum current threshold. In this fully charged state, the SoC can be defined as 100% with sufficient accuracy. The entire procedure represented in Fig. 1.2 may take 2 to 4 hours, depending on the constant current rate, which is usually 0.5C for safe charging operations.

Other charging profiles were investigated in the literature to obtain fast charging without compromising the accumulated charge [7], [8], but increasing procedure complexity.

### 1.4.2 Error metrics

In the literature, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) metrics are commonly adopted to evaluate the performance of the methods under investigation. Some authors also evaluate the maximum Absolute Error (AE), the Mean Squared Error (MSE) or the  $R^2$ . They are defined as:

$$\text{RMSE} = \sqrt{\frac{1}{m} \sum_{k=1}^m (S\hat{o}C_k - y_k)^2} \quad (1.4)$$

$$\text{MAE} = \frac{1}{m} \sum_{k=1}^m |S\hat{o}C_k - y_k| \quad (1.5)$$

$$\text{AE} = \max(|S\hat{o}C_k - y_k|) \quad (1.6)$$

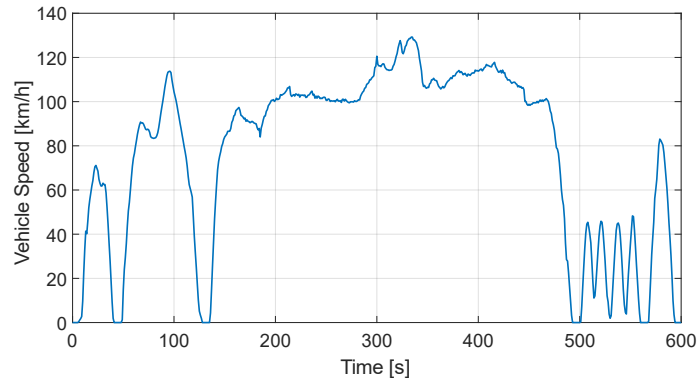
$$\text{MSE} = \frac{1}{m} \sum_{k=1}^m (S\hat{o}C_k - y_k)^2 \quad (1.7)$$

$$R^2 = 1 - \frac{\sum (y_k - S\hat{o}C_k)^2}{\sum (y_k - \bar{y})^2} \quad (1.8)$$

where,  $S\hat{o}C_k$  represents the predicted value of SoC at the observation  $k$ ,  $y_k$  is the observed value, i.e., the target expected value, and  $m$  is the number of observations. The  $\bar{y}$  is the mean of the observed values  $y_k$ .

### 1.4.3 Drive Cycles

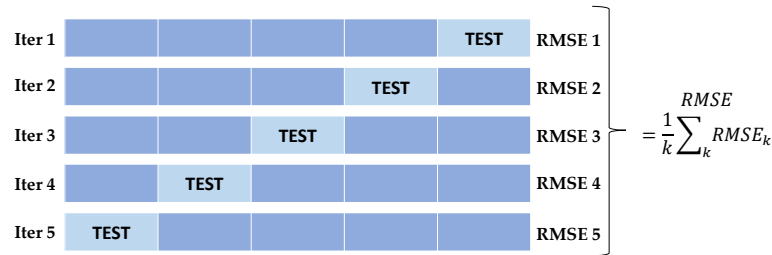
A drive cycle is a series of data points representing the speed of a vehicle versus time. Different countries and organizations standardized the drive cycle profiles to assess the vehicle fuel consumption, emissions and autonomy in the laboratory environment. This allows the classification of vehicle performance. Vehicle manufacturers in



**Figure 1.3:** Vehicle speed as a function of time in the FTP-75 US06 drive cycle.

Europe must follow strict regulations for their vehicle emissions. The New European Driving Cycle (NEDC) was the European standard drive cycle, but it was considered unrealistic for actual driver's behaviour. In 2017, the Worldwide harmonized Light vehicles Test Cycles (WLTC, or WLTP) became the new standard adopted in different countries. In the United States of America, the Environmental Protection Agency (EPA) introduced a collection of four Federal Test Procedures (FTP-75) to measure vehicle emissions and fuel consumption: the Urban Dynamometer Driving Schedule (UDDS) simulating an urban route with stops, the Highway Fuel Economy Driving Schedule (HWFET) simulating highway trips, the SC03 simulating the use of the air conditioning, and the US06 simulating aggressive driving behaviour, high speed and speed fluctuations, as shown in Fig. 1.3. In laboratory research, the most common drive cycles for simulations are the WLTC, NEDC, and the US06.

For the purposes of this dissertation, the vehicle current consumption must be obtained from the drive cycle vehicle speed. This was performed with a dedicated vehicle simulator that considered vehicle and battery dynamics, as detailed in Appendix A. The US06 was widely employed due to the large speed fluctuations, i.e., the worst scenario for an Electric Vehicle causing large current consumption.



**Figure 1.4:** Example of a 5-fold cross-validation procedure.

#### 1.4.4 Cross-Validation

The  $k$ -fold cross-validation is proficiently used in the literature to evaluate the effectiveness of a trained ML model, [9], [10]. With the  $k$ -fold cross-validation, the dataset is randomly split into nearly equal  $k$  folds, usually composed of  $N/k$  data points, with  $N$  total data points as in Fig 1.4. For each fold, a model is trained on the  $N/k$  data (only one fold) and then tested over the remaining folds ( $N - N/k$  data), computing the RMSE or MSE. The process is then iterated for each fold, therefore the choice of  $k$  determines the computational time, i.e., higher  $k$  means higher iterations and higher time. Finally, the accuracy obtained in each iteration is then averaged to get the model accuracy. Once repeated for different sets of parameters, the averaged error can be used then to compare and select the optimum model.

The extreme case in which  $k = N$  is called *Leave-One-Out cross-validation*. The computational cost is maximum because only one sample is used as a test, and  $N$  iterations are performed. This is unsuitable for large datasets. On the other hand, the *Hold Out cross-validation* splits the dataset in a specific percentage for the training and the remaining for the test, as Fig. 1.5. The main difference is that, in this case, only one training iteration is performed, and the model error is not averaged. Therefore, the time required to accomplish the validation is usually lower than with the



**Figure 1.5:** Example of a Hold Out cross-validation.

k-fold. However, in [9], it was found that the k-fold cross-validation is more accurate when employed for large datasets, even after the time trade-off, but the  $k$  value must be limited. For the purposes of this dissertation, the 10-fold cross-validation will be employed to validate the models during the optimization phases.



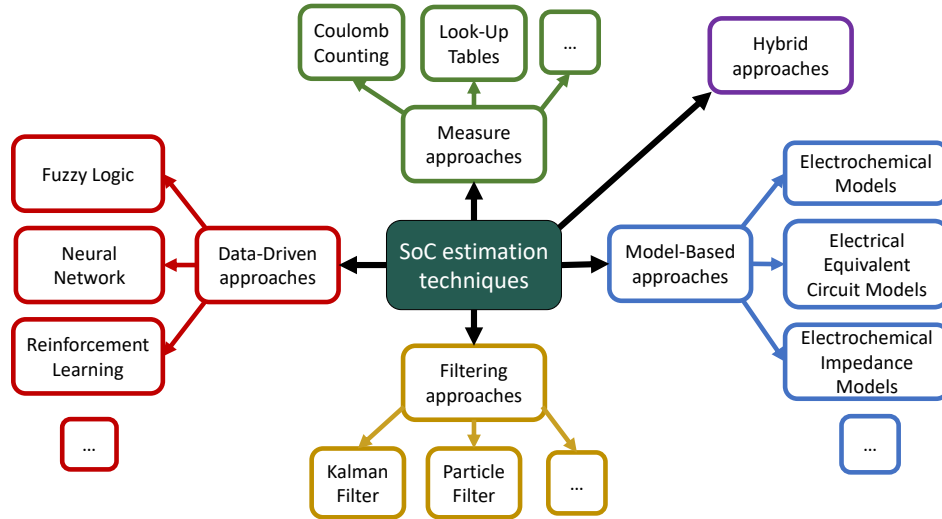
## Chapter 2

# State of the Art

Determining the SoC of a battery using (1.3) is not possible because measuring the charge capacity  $Q$  is not feasible. However, various methods have been developed to estimate SoC using measurable quantities such as current and voltage. Several recent paper reviews, including [11], [12], [13], [14], [15], have summarized these techniques, and an overview is presented in Fig. 2.1.

The choice of the approach for the specific application depends on the requirements and the trade-offs between accuracy, complexity, and real-time capabilities. When real-time estimation is needed with minimal computational cost, Coulomb Counting (CC) is preferred, albeit accepting lower accuracy. Model-based approaches, on the other hand, are useful when high accuracy is required, but they come at the cost of a more complicated implementation and time-consuming cell characterisation. However, they do not properly account for varying cell conditions. In such cases, a hybrid approach that updates model parameters is suggested, albeit with increased complexity. Finally, Data-Driven approaches can be employed when cell characterisation is not feasible, such as in real-world scenarios outside of laboratories, and a model that can be applied to different cells is preferred. These approaches can provide good accuracy but require large datasets to infer a Data-Driven Model (DDM).

Moreover, the choice of the estimation approach should rely on the final application field. Indeed, an optimum BMS must monitor each cell of the battery pack



**Figure 2.1:** State of Charge estimation techniques

simultaneously, but this requires a lot of computational power. Depending on the battery pack characteristics and the requirement for an actual real-time estimation, different approaches could be chosen. Several implementations have been presented in literature based on microcontroller systems [16], [17], [18]. The microcontroller implementation could be low cost but has some limitations when managing real-time tasks. The main drawbacks are the operating frequency limiting the number of operations and the fixed data type. On the other hand, a Field Programmable Gate Array (FPGA) allows the implementation of multiple instances of the same algorithm, exploiting the intrinsic parallelism of these architectures. Recently, different solutions have been proposed to estimate the SoC on FPGAs [19], [20]. Traditionally, the Hardware Description Language (HDL) coding for programming FPGA devices is time-consuming, involving typing, testing and programming. Nevertheless, recently the model-based design tools allowed for rapid system development. The MATLAB/Simulink environment, for example, offers the HDL Coder tool to automatically generate certified HDL code from a Simulink schematic. The tool allows to select the target FPGA or generate a general HDL code. This allows greater

portability on FPGA from different manufacturers, unlike some approaches employing vendor-specific tools as in [21].

## 2.1 Measure Approaches

The most practical method for estimating the SoC is to rely on physical parameters such as voltage, current, temperature or impedance. The SoC is estimated through the gathered data thus it relies on the accuracy of the measurements.

### 2.1.1 Coulomb Counting

The CC method is one of the most widely used due to its computational simplicity at the cost of reduced accuracy. The accumulated charge can be obtained by integrating the current over time. Indeed, the CC formulation is (2.1)

$$SOC(t_0 + t) = SOC(t_0) - \frac{1}{Q_{tot}} \int_{t_0}^{t_0+t} \eta(\tau) I_L(\tau) d\tau \quad (2.1)$$

where  $SOC(t_0)$  is the SoC at the initial time,  $\eta(\tau)$  is the charge efficiency, and  $I_L(\tau)$  is the load current as a function of time  $\tau$  flowing into (negative sign) or out (positive sign) of the cell. It has to be noted that the current sign convention only affects the + or - sign in (2.1). The charge efficiency indicates the ratio between the charge flowing in the cell during charge and the charge flowing out during discharge. For lithium-ion cells, it is higher than 99%, so it is usually reasonably assumed to be unity. In embedded systems, data are usually discrete-time sampled, so the following formulation should be used.

$$SOC(k + 1) = SOC(k) - \frac{\Delta t}{Q_{tot}} \eta(k) I_L(k) \quad (2.2)$$

In (2.2), the sampling rate is assumed to be regular with period  $\Delta t$ , and the current is assumed to be constant over the sampling interval  $\Delta t$ . These assumptions are usually reasonable in embedded systems with a short sampling interval. Finally, the  $k$  represents the sample number. Again, the charge efficiency  $\eta$  can be assumed as constant and unity.

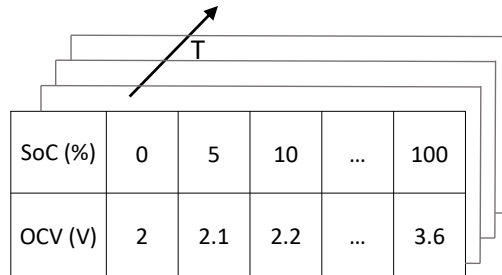
Implementing the formula is straightforward, and computations are not complex, making it an attractive choice for real-time SoC estimation, but this method also shows clear disadvantages. First of all, it is only suitable for ideal scenarios, such as in laboratory workbenches. In reality, the current measurement is affected by noise and sensor error, both of which add to the actual data in the integral in an open loop. Even small errors can be significant due to the cumulative effect, resulting in a cumulated error that must be reset at a certain point. In laboratory workbenches, high-resolution instruments can be used, and the noise is usually reduced compared to real-life settings. Moreover, the initial state  $SOC(t_0)$  is generally unknown unless the cell is completely charged or discharged, in which case  $SOC(t_0)$  can be defined as 100% or 0%, respectively. If the initial state is unknown, it must be retrieved from a Look-Up Table (LUT), reducing the accuracy of this approach. Therefore, the CC technique can be used with high confidence only for SoC estimation in laboratory measurements. Due to these shortcomings, the CC method in real-world applications should be combined with other techniques to enhance the estimation reliability.

### 2.1.2 Look-Up Tables

Most of the methods relying on measured data require obtaining the relationship between the SoC and the measured quantity. Intensive experiments must be performed in the laboratory to characterise the behaviour of each cell and tabulate this relationship. Each cell characterisation must be stored then in a specific LUT to be accessed for computations during battery operations.

The Open Circuit Voltage (OCV) method is simple and accurate but takes huge time to characterise the cell and measure the OCV values to store in the LUT. Indeed, firstly, the cell must be fully charged and allowed to rest until the voltage relaxation disappears. This relaxation period is necessary to remove any polarization effects. The voltage exponentially decays during this period, and after 3 to 4 times the voltage time constant  $\tau$  it can be considered finished. The specific rest time necessary for a cell can vary and must be experimentally determined, but usually, 30 to 60 minutes is considered sufficient.

Then, the cell is fully discharged by applying multiple current pulses. Each pulse

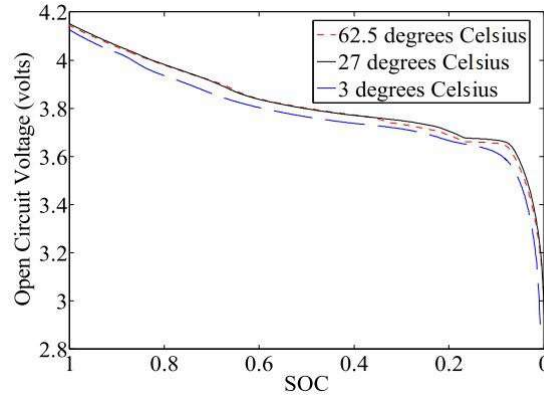


SoC (%)	0	5	10	...	100
OCV (V)	2	2.1	2.2	...	3.6

**Figure 2.2:** Example of Look-Up Table storing OCV-SOC data pairs as a function of the temperature  $T$

removes a specific percentage of SoC from the cell, based on its duration and amplitude. After each pulse, the cell is rested for the relaxation period and the OCV is recorded. Finally, a list of OCV-SoC pairs is obtained, and the same procedure must be repeated for the charging phase. The mapping can be stored in a LUT, e.g., Fig. 2.2, or a polynomial function can be fitted. The SoC resolution, i.e., the number of data points, can be changed by modulating the amplitude and duration of the pulses. Higher resolution allows for more accurate SoC estimation but also requires more characterisation time and increased available space to store larger LUTs. Then, at each time step, the corresponding SoC level can be determined by measuring the OCV during open-circuit conditions, or computing the OCV through other approaches.

This method is accurate but has significant drawbacks. The static state OCV can not be measured because cells are continuously operated in real-world applications. Moreover, it is also influenced by other factors, such as ambient temperature and ageing. Therefore, the stored LUT may not be suitable for the specific conditions, adversely affecting the SoC estimation accuracy. In Fig. 2.3 from [22], a OCV-SoC relationship and the temperature dependence are shown. Even if different cell parameters, such as AC impedance or terminal voltage, could be characterised and stored in the LUT, these drawbacks still cannot be overcome.



**Figure 2.3:** Open Circuit Voltage - State of Charge characteristic at different temperatures

## 2.2 Model-Based Approaches

The Model-Based approach relies on creating a model to describe the cell's behaviour. The deeper the understanding of the phenomenon to be represented, the higher the level of accuracy at which the system can be modelled, and hence the better the results obtained.

A Li-ion cell can be represented by different kinds of models. The most accurate technique is to design an Electrochemical Model describing the ions and electron kinetics caused by chemical reactions [14], also considering the lithium-ion concentration at each electrode. Multiple partial differential equations and huge amounts of variables make this model computationally complex and hard to implement [11], [12], [15], [23]. Despite the high accuracy, they are not suitable for real-time estimation due to the high complexity.

The cell behaviour can be modelled also with an Electrochemical Impedance model reflecting the mass transport and diffusion dynamics between the electrolyte, the membrane and the electrodes. Different resistance in the charge dynamics can be observed with dedicated characterisation procedures. In particular, the Electrochemical Impedance Spectroscopy (EIS) is mainly employed to obtain the impedance frequency spectrum to be included in the model. This method is highly accurate when

the impedance model is correctly characterised but it is not suitable for online SoC estimation due to the long and complex characterisation procedures.

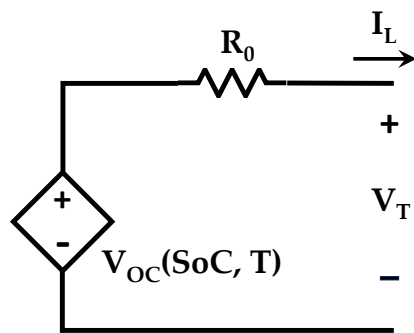
On the other hand, the cell dynamics can be investigated and replicated by lumped electrical components and their equations to obtain an electrical ECM. The purpose of these models is to represent the cell's internal electrochemical processes, so deep knowledge of the operating principles is required to design model parameters accurately. The ECM is the most used model in BMSs due to its simplicity and robustness [24]. As investigated in [15], patents from different companies show that their BMSs mainly employ the ECMs. ECMs provide good accuracy with a reasonable computational cost for many applications despite not requiring a high level of detail as in electrochemical models. Different variants have been proposed in the literature, but the least complex and most widely used models resulted in the Rint and Thevenin models.

The Rint model shown in Fig. 2.4 is the simplest cell model. An ideal voltage source  $V_{OC}$  represents the cell's open-circuit voltage OCV. A temperature and SoC dependency can be added to better emulate the cell behaviour, but the relationship still needs to be characterised and stored in a LUT. A series resistor  $R_0$  is connected to the voltage source to describe the voltage drop response to a load current. This resistor represents the Equivalent Series Resistance (ESR) of the cell. Indeed, the terminal voltage  $V_T$  drops below  $V_{OC}$  when the cell discharges and rises above when the current flows into the cell. The voltage behaves as depicted in (2.3), where  $I_L$  flowing out of the cell has a positive sign. The voltage drop on  $R_0$  will be named  $V_0$  in the following.

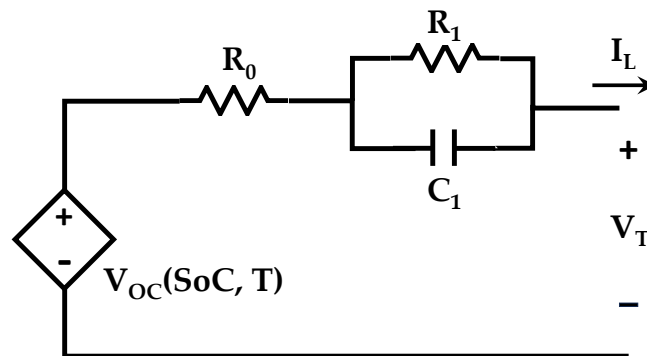
$$V_T(t) = V_{OC}(SoC(t), T(t)) - I_L(t)R_0 \quad (2.3)$$

It has to be noted that also the ESR  $R_0$  is a function of the temperature  $T$  and the SoC. Improving the model accuracy by considering these relationships requires the implementation of a LUT.

However, the Thevenin model shown in Fig. 2.5 can better represent the cell dynamics. It is typically used due to its increased accuracy and is designed using one RC group, a resistance for the ESR and a voltage source for the OCV. The RC group replicates the voltage relaxation phenomenon related to the chemical polarization

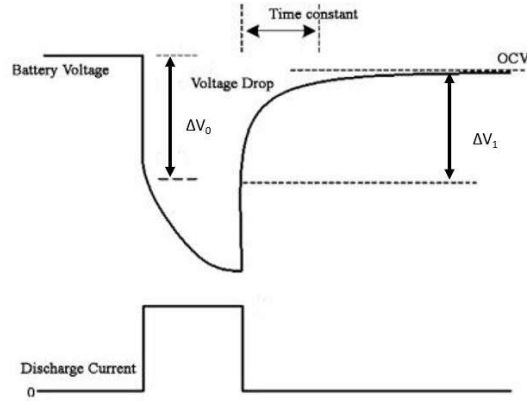


**Figure 2.4:** Electrical Equivalent Circuit Model - Rint model



**Figure 2.5:** Electrical Equivalent Circuit Model - Thevenin model





**Figure 2.6:** Dynamic behaviour of cell voltage in response to an input current step

caused by the current flowing through the cell. The voltage relaxation response to current steps is an exponential decay as in Fig. 2.6, equivalent to a parallel RC in the electrical domain. By adding multiple RC branches in series, the model accuracy can be increased [25]. However, this comes at the cost of increased computational complexity. Since the increase in computational complexity is usually greater than the advantages in terms of accuracy, the one RC branch variant is typically preferred [25]. The voltage drop on the RC branch is named  $V_{RC}$ , or either  $V_1 \dots V_n$  if multiple RC branches are used. The model can be described by the following equations (2.4) derived from Kirchoff's Laws.

$$\begin{aligned}
 V_T(t) &= V_{OC}(SoC(t), T(t)) - I_L(t)R_0 - V_1(t) & (2.4) \\
 \frac{dV_1(t)}{dt} &= -\frac{V_1(t)}{R_1 C_1} + \frac{I_L(t)}{C_1} \\
 I_L(t) &= I_{R_1}(t) + C_1 \frac{dV_1(t)}{dt}
 \end{aligned}$$

Each parameter, i.e.,  $R_0$ ,  $R_1$ ,  $C_1$ ,  $V_{OC}$ , is also a function of the temperature and the SoC. When a proper characterisation has been performed, high accuracy in SoC estimation can be obtained. In the case of a laboratory workbench, there is a long time available to deeply characterise the behaviour of each cell. Therefore, some off-line methods have been developed, such as the Hybrid Pulse Power Characterization

(HPPC) and the EIS. Some curve-fitting algorithms and Least Squares (LS) are also available, depending on the chosen model and type of characterisation. However, cells are usually placed in on-site applications and internal equivalent parameters change over time and are affected by environmental conditions. If the conditions differ from the characterisation setup, the model is not reliable anymore and the accuracy decreases. Consequently, it is crucial to update in real-time the parameters to adapt to varying conditions to preserve accuracy, and this is usually accomplished by combining the designed model with other approaches or performing online parameters identification.

### 2.2.1 Offline parameter identification

#### Hybrid Pulse Power Characterization

The HPPC testing procedure is based on pulsed signals and relaxation periods, similar to the OCV LUTs approach described in Section 2.1.2, and is commonly employed to derive the parameters needed for the ECMs and evaluate the cell power capabilities, [24]. As detailed in [26], a series of current pulses is applied and the corresponding cell voltage response is recorded during the charge/discharge and the resting phases. By analysing the dynamic behaviour, that will be as in Figure 2.6, the time constant of the voltage response can be correlated to equivalent electrical components, i.e., the required ECM parameters. Considering the 1-RC Thevenin model as in Figure 2.5, it is straightforward to extrapolate the  $R_0$ ,  $R_1$ ,  $C_1$  and  $V_{OC}$  parameters, by managing the voltage HPPC data. For example,  $R_0$  is computed by looking at the instantaneous voltage change during current drops as (2.5), and  $R_1$  and  $V_{OC}$  from the steady state voltage after relaxation. The  $C_1$  parameter is then obtained by evaluating the final relaxation time (4 time constants usually).

$$R_0 = |\Delta V_0 / I| \quad (2.5)$$

$$R_1 = V_{OC} / I - R_0 \quad (2.6)$$

$$C_1 = t_{end} / (4R_1)$$

It is worth noting that the obtained parameters are a function of the SoC, but the entire procedure must be repeated for each temperature of interest.

### Electrochemical Impedance Spectroscopy

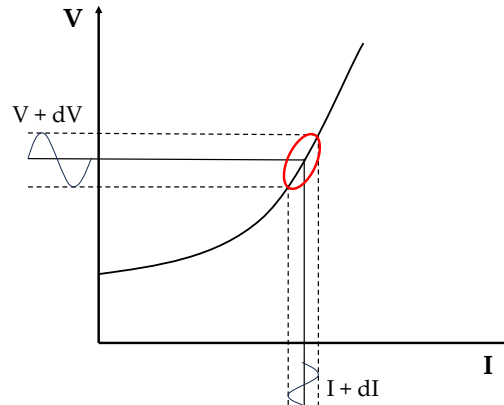
Mainly employed for Electrochemical Impedance models, the EIS technique allows to obtain the impedance spectrum of a battery cell by applying an AC perturbation and computing the Ohm's law applied in the frequency domain (2.7):

$$Z(\omega) = \frac{V(\omega)}{I(\omega)} \quad (2.7)$$

where the impedance  $Z$  evaluated at a specific frequency  $\omega$  is the ratio of the Fast Fourier Transform (FFT) of voltage  $V$  and current  $I$  in the time domain. It is worth noting that when dealing with hardware implementation on embedded systems, the FFT algorithm can be computationally intensive. Nevertheless, there are alternative algorithms that can be employed instead, such as the Goertzel algorithm [27]. Due to complex chemical reactions inside the cell, various phenomena happen at different frequencies, e.g., ohmic phenomena in the kHz range or mass transport at the mHz range. Each phenomenon contributes to the overall impedance [28], [29]. It is crucial to span most of the frequency range to capture as much cell information as possible.

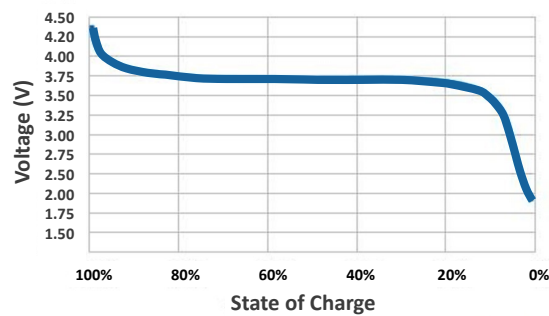
Before applying this technique, some prerequisites must be met [30], [31]. The foremost requirement is that the system must be linear so that, considering a cell, a specific input  $I$  is linearly related to the specific output  $V$ , and the impedance remains consistent during operations. However, the voltage-current relationship in a battery cell is generally non-linear. This can be overcome by applying a sufficiently small AC input to cause a limited deviation from the operational point, as in Fig. 2.7, thus preserving a pseudo-linear relationship between the perturbation signal and the cell response [31].

Additionally, the system must be in a stationary condition, i.e., time-invariant. This is mostly the case if the DC component of the applied input  $I$  is zero. In contrast, during battery cell operations in which the current contains a non-zero DC component, this condition is not verified because the voltage varies with the varying SoC



**Figure 2.7:** Example of pseudo-linear relationship in a reduced interval.

level. It is worth noting that the EIS can be applied as long as the voltage drift is removed before computing the impedance. The longer the measurement, i.e., the lower the frequency, the more noticeable the voltage drift becomes, as shown in Fig. 2.8. Proper voltage drift correction is mandatory to obtain a meaningful FFT spectrum, as depicted in [31]. A corrupted voltage spectrum compromises the computation of the impedance, affecting (2.7).



**Figure 2.8:** Typical voltage curve versus cell State of Charge

Moreover, the SoC deviation during the measurements must be limited to obtain a reliable spectrum. The SoC variation  $\Delta SoC$  can be computed with (2.8), derived from (2.2). In the literature, a deviation of 1% to 5% during measurements is commonly considered acceptable [32].

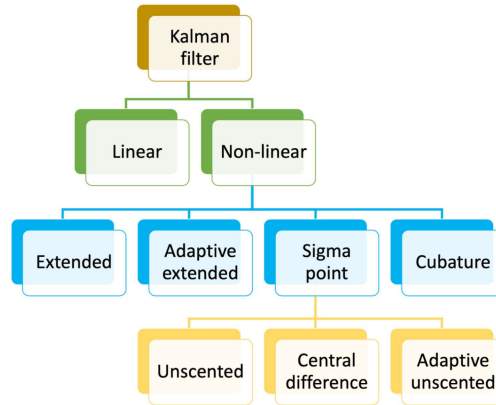
$$\Delta SoC(\%) = \frac{I_{DC} \times \Delta t}{Q_{tot}} \times 100 \quad (2.8)$$

where  $Q_{tot}$  is the battery capacity and  $\Delta t$  is the measurement time.

### 2.2.2 Online parameter identification

Despite the wide variety of offline characterisation tests available, the internal dynamics of Li-ion cells are temperature and SoC dependent, leading to inaccurate results if the model parameters are not updated during the cell's operations. Various techniques can be applied to overcome this problematic drawback. Among the online parameter identification algorithms, the Recursive Least Squares (RLS) is one of the most employed [33]. However, it is highly susceptible to noisy measurements, hence different variants are usually employed, such as the Recursive Total Least Squares (RTLTS) [33], the Weighted Least Squares (WLS)[15] and the Moving Window Least Squares (MWLS)[34].

The LS algorithms aim to minimise the sum of the squared of the residuals, i.e., to minimise the error between the observed value and the fitted model value. This is usually accomplished by evaluating the model equations in a matrix form. Considering the discrete-time domain, it is possible to evaluate the state change from one time step to the next. The state matrix  $B$  is the result of multiplying the model variables matrix  $A$  (such as current and voltage) by the desired parameters matrix  $x$  (i.e.,  $Ax = B$ ). Therefore, computing the inverse of a matrix is necessary to obtain the model's parameters using the LS method (i.e.,  $x = A^{-1}B$ ). However, this process may become computationally complex if the size of the required matrices is very large excessive.



**Figure 2.9:** Kalman Filter variants

## 2.3 Filtering Approaches

Filtering algorithms are usually jointly implemented with model-based approaches to improve predictions by real-time state estimation [11], [14]. Typically, methods based on filtering provide maximum errors in SoC estimation even below 1% [12], but implementation complications arise due to the high computational complexity.

### 2.3.1 Kalman filters

The most widely used algorithm for this method is the Kalman Filter (KF) and its variants shown in Fig. 2.9 from [12], [15]. Specifically, the Extended KF (EKF) was introduced to deal with the inefficient performance of standard KF applied to non-linear systems. The KF is a recursive mathematical algorithm used for estimating the state of a dynamic system from a series of noisy measurements. A set of state variables, which may include the SoC or other relevant parameters, is recursively updated after a prediction step. The KF predicts the state variables based on the equivalent model, estimating how the state is expected to evolve over time. Real-time measurements, such as voltage and current, are used to update the estimated state variables, providing a weighted combination of prediction and measurement. The KF also estimates the covariance of the state variables, representing the uncertainty in the state

estimates. The process iterates as new measurements become available, continuously refining the state estimates. Applying KF for SoC estimation yields acceptable accuracy and effectively handles system noise. However, the algorithm is complex and requires significant computational memory and proper initialization [34]. Indeed, the initial state uncertainty in SoC estimation can significantly affect the accuracy of the KF since any error in the initial estimation can propagate throughout the filtering process. Nevertheless, when combined with other approaches, feedback can be obtained to correct errors.

### 2.3.2 Particle filter

Besides KF, other adaptive filters have been studied in the literature, such as the Particle Filter (PF) [35], [23]. This algorithm is based on stochastic modelling and Monte Carlo simulation from a probability density function. However, the high computational complexity makes this approach unsuitable in most of the cases.

### 2.3.3 Recursive Least Squares filter

The working principle of the RLS is still based on recursive state estimation, updating the system states based on feedback errors to minimise the squared of the residuals, e.g., the error between the modelled terminal voltage  $V_T$  and the actual measured voltage, but this approach still does not overcome the drawback of the KFs. However, KF and RLS are usually employed together to assess the battery state.

## 2.4 Data-Driven Approaches

A wide variety of algorithms based on large dataset evaluation (i.e., data-driven) have been employed in the literature for the SoC estimation [11], [14], [15]. Algorithms based on statistical tools, such as Fuzzy Logic (FL), or Reinforcement Learning (RL), have been successfully employed in the SoC estimation field [23]. FL is adaptive to changes in the system conditions being based on rules rather than system description [23]. At the same time, a similar rule-based approach is also provided by RL,

where an agent learns multiple policies by trial and error interactions with a dynamic environment [36]. However, most of the Data-Driven approaches in the SoC estimation field rely on the use of Artificial Intelligence (AI), i.e., ML algorithms. In [13], [37], some review of AI-based approaches are presented. Support Vector Machine (SVM), Gradient-Boosting, and Artificial Neural Networks are reported and compared. In [38], the SVM in the regression form is employed, and various kernel functions are evaluated. In [39], the SoC estimation is performed by an RBF-Neural Network, and in [40], a DCNN was trained to be applied to other cells with the transfer learning technique, similarly to [41]. In [42], [43], also Deep Learning (DL) and Convolutional Neural Network (NN) are discussed.

In this dissertation, the several ML algorithms and their variations that can be found in the literature will not be detailed to improve conciseness. However, in the following chapters, some will be discussed when needed.

One of the advantages of the ML algorithms is that DDMs can be inferred without prior knowledge of the cell and they do not rely on electrochemical behaviour emulation. Indeed, the inferred models are based on self-learning parameters, preventing the need for cell characterisation or filtering techniques. During the *training phase*, a large dataset is fed into the ML algorithm, which learns and adapts to the cell behaviour based on historical data. The algorithm searches for patterns and meaningful relationships within the training data, evaluating several data features. As a result, the inferred models are often black boxes for the user, who has to pre-process train data to optimize the estimation accuracy properly. The choice of the dataset is crucial for the accuracy of the algorithm. The dataset needs to represent real-world conditions and should include a range of scenarios the algorithm is likely to encounter. Furthermore, data-driven approaches allow high flexibility. The inferred model can be trained with measurements from different kinds of cells or different scenarios, resulting in a single model adaptable to various applications. This is a clear advantage compared to model-based approaches, in which each cell must be individually characterised, and each scenario must be considered separately. These algorithms can achieve high levels of accuracy, in particular when trained on large datasets. Unfortunately, the need for large datasets results in a drawback if data are unavailable



and must be acquired. However, some battery-related datasets have been provided in relevant studies [44], [45], [46] to help overcome this drawback. Finally, it has to be considered that the implementation of these algorithms in embedded systems requires high computing resources, which makes them usually unsuitable for low-resource systems.

## 2.5 Hybrid Approaches

Estimating the SoC accurately and robustly is essential for BMSs. Hybrid approaches, or *mixed algorithms*, are a combination of multiple techniques and often incorporate model-based approaches, filtering techniques, and real-time measurements to improve the SoC estimation [47]. However, the models may struggle in dynamic situations due to conditions differing from the learned or characterised parameters. The KF, in particular, is a popular choice in mixed algorithms [48], [49], due to its ability to integrate dynamic models with noisy measurements. Additionally, Least Squares (LS) algorithms may be used to update model parameters. Furthermore, the CC can serve as a reference for providing feedback on the model output. In [50], a physics-constrained NN has been trained and reinforced with the CC technique. By combining different approaches, feedback information can be introduced in the estimation loop, thereby enhancing the accuracy of the stand-alone techniques. Ultimately, this results in a system that benefits from both the employed methods and improves SoC estimation accuracy. However, the combined techniques must be tuned to properly act together, and the system complexity increases. Consequently, the final implementation in embedded systems may be more difficult, as demonstrated in [B1].

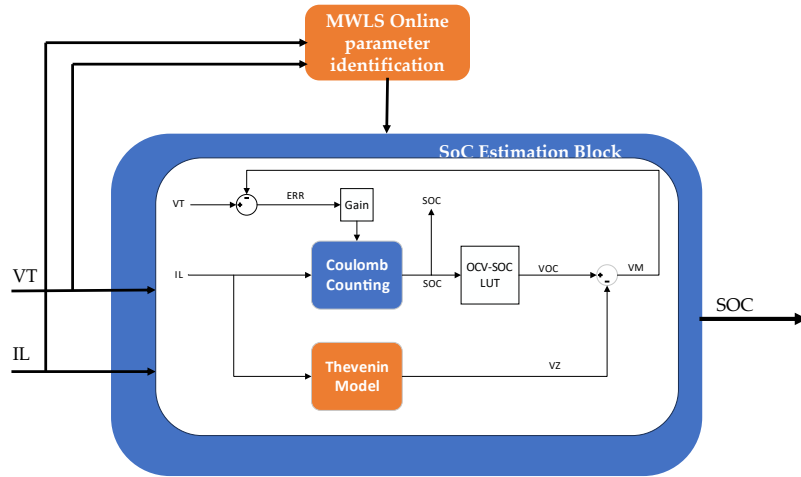


## Chapter 3

# Implementation of Model-Based and Data-Driven approaches

### 3.1 Equivalent Circuit Model and Coulomb Counting

A variety of hybrid approaches combining different techniques have been investigated in the literature, but one of the most straightforward involves improving the ECM with a feedback loop from the CC as in [51], [21], [B1]. The Thevenin model in Figure 2.5 has been used, as being one of the most popular cell models. The CC, on the other side, is the most reliable SoC estimation when performed in a controlled environment. Nevertheless, in the proposed approach, the feedback loop including the ECM allows for overcoming the accumulating measurement noise error typical of the CC computations. As aforementioned, the main drawback of model-based approaches is the need for extensive characterisation to obtain equivalent cell parameters. This process is detailed in [52], where a Panasonic NCR18650 2.75 Ah cell was thoroughly characterised with the HPPC procedure. However, in the proposed approach, once the system is implemented in the on-site application it must be able to retain SoC estimation accuracy over time and in different ageing and temperature conditions. Therefore, the equivalent parameters of the Thevenin model must be identified online during cell operations to improve SoC accuracy. As detailed in Sec-



**Figure 3.1:** System overview for a hybrid approach based on Equivalent Circuit Model, Coulomb Counting and online parameter identification.

tion 2.2, the online parameter identification algorithms are usually based on Least Squares or KF. For a first evaluation, in this particular implementation, the KF was discarded due to the high algorithmic complexity shown in the literature [34], which could lead to high resource utilization. The MWLS technique has been then chosen, as successfully employed in [34], [53], [54]. The architecture of the complete system is shown in Fig. 3.1.

A high-level design has been chosen for the proposed system, to increase flexibility and portability to different hardware platforms. Indeed, the Thevenin model has been realized in the Simulink environment, along with the complete MWLS algorithmic procedure as proposed in [34]. A *Discrete Time Integrator* block performs the accumulation function required by the CC. The entire system has been designed by employing only HDL Coder compatible blocks to improve code portability, instead of developing the system with vendor-specific design tools such as the Intel DSP Builder Tool. The MWLS subsystem requires  $L$  samples of current and voltage to estimate the cell parameters, where  $L$  is the window length and depends on the sampling time. Longer window length allows for more accurate parameter identific-

ation, but increasing the size of the matrices leads to higher computational resources as depicted in 2.2. As previously detailed, the MWLS requires matrix inversion to compute the parameters matrix. The Simulink standard block for this operation is not supported by the HDL Coder tool, therefore an equivalent algorithm based on row swaps has been designed.

The operations have been performed in fixed-point precision with a signed 32-bit word length data and 21 fractional bits to obtain a simpler hardware architecture. This data type can represent a range from -1024 to 1023 and a precision of  $4 \times 10^{-7}$ , which covers the data range for typical battery packs. Finally, the system has been converted into VHDL code through the Simulink embedded HDL Coder tool without specifying the target FPGA, to highlight the high portability of the code. The behavioural simulation has been performed in the Xilinx Vivado Design Suite, obtaining a maximum percentage error in the magnitude of  $10^{-4}$  in parameters value when compared with offline performed HPPC characterisation values. Additionally, the synthesis elaboration for a Digilent Nexys 4 DDR Development Board, which is based on a Xilinx Artix-7 FPGA, resulted in 22.76% of the occupied area, as detailed in Table 3.1.

TABLE 3.1  
AREA OCCUPATION FROM SYNTHESIS ELABORATION

Slice LUTs utilization (%)	Slice Registers (%)	DSPs utilization (%)
14427/63400 (22.76%)	196/126800 (0.15%)	8/240 (3.33%)

The resulting resource utilization allows for a maximum of four algorithm instances to be implemented on the same Xilinx Artix-7 FPGA. The ECM itself resulted in low computational complexity, but adapting the Model-Based approach to the online parameters update led to excessively high computational complexity, mainly due to the matrix operations. However, according to the literature, a similar problem arises when employing KF. Nevertheless, the proposed system can still achieve good parallelism for multiple cells with proper pipelining.

## 3.2 Support Vector Machines

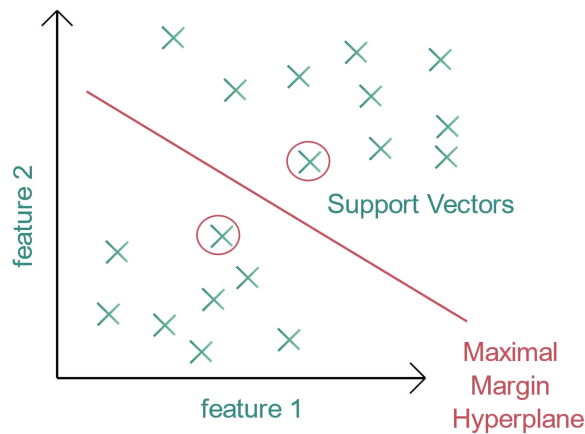
Given the results obtained by the hybrid approach in terms of ease of implementation and occupied resources, the implementation of a Data-Driven approach has been thoroughly investigated to be fairly compared on the same hardware platform. In the field of SoC estimation, different ML algorithms have been employed, including NN and DL. SVMs have also been explored, specifically in the Support Vector Regression (SVR) version. These algorithms have commonly been investigated on PC-based platforms [10], with only a few works presenting their implementation on microcontrollers [17]. The microcontroller constrains the implementation and limits the model's flexibility. However, a FPGA platform offers some advantages, including performing real-time estimations and providing hardware flexibility and reconfigurability. ML algorithms require high computational power, but the goal of estimating the SoC in real-time can be obtained in an FPGA-based system. In the case of a battery pack, each cell (or module) is usually monitored by a slave board belonging to a master BMS board. By exploiting FPGAs, slave boards could be embedded in the same FPGA platform equipped with multiple instances of the same monitoring algorithm, acting in parallel on each cell, thereby reducing the hardware requirements.

At the time of this dissertation, FPGA implementations of AI approaches for the SoC estimation were limited to [55] in which a NN model has been considered. However, it resulted in high area occupancy, making it difficult to export to low-cost FPGAs and implement multiple instances. On the other hand, the SVR algorithm was implemented on FPGAs in different applications but not in the SoC estimation field. Therefore, the SVR algorithm was chosen to implement a data-driven approach for the SoC estimation on FPGA platforms. The main goal was to achieve good accuracy in SoC estimation, comparable to that of PC-based implementations available in the literature, while still resulting in low area occupation.

### 3.2.1 Support Vector Regression

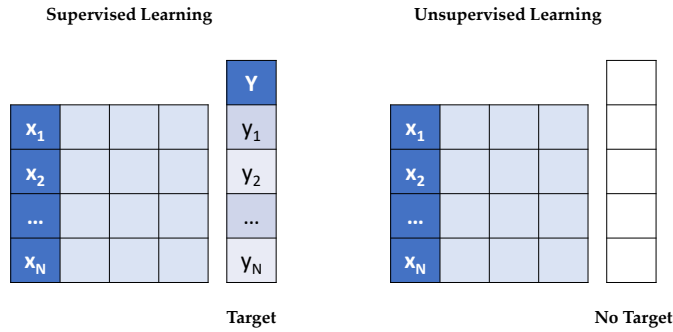
The SVMs were initially developed by V. Vapnik in 1995 for binary classification [56]. Given a set of training points and two classes, the SVM algorithm aims to define and

orientate a hyperplane in such a way as to be as far as possible from the closest members of both classes, namely Support Vectors (SV), as in Fig. 3.2. The solution to



**Figure 3.2:** Support Vector Machines for binary classification

this constrained minimization problem is detailed in [57]. However, the algorithm has been adapted for managing regression problems, resulting in the Support Vector Regression [58]. It has to be noted that this dissertation focuses on the regression problem of SoC estimation, and therefore, the acronyms SVM and SVR will be used interchangeably throughout the text, indicating the same regression task. The SVR is a supervised learning algorithm. In other words, the inferred model must be trained with labelled data, i.e., each input data vector is associated with a known corresponding output target, as in Fig. 3.3. This type of algorithm is usually employed for classification and regression tasks, while unsupervised learning algorithms are better suited for clustering tasks. In SVR, a set of  $N$  data vectors  $x_i \in R^n$  and their corresponding known outputs  $y_i \in R$  are used to construct a regression function by solving the SVR quadratic programming problem [57]. The SVR algorithm penalizes predicted values more than a distance  $\epsilon$  from the actual value, as shown in Fig. 3.4. The region of non-penalization is called  $\epsilon$ -tube, and defines the approximation accuracy.



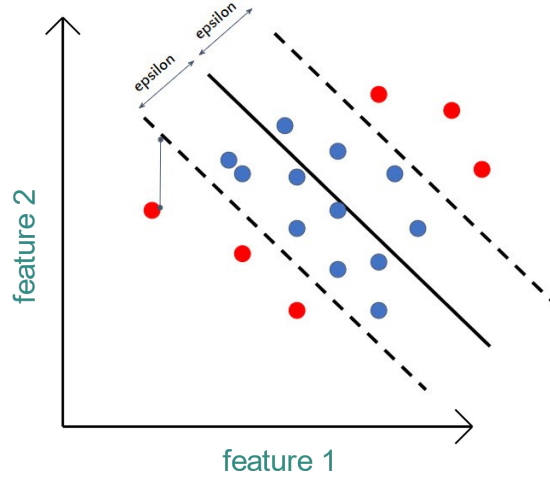
**Figure 3.3:** Supervised and Unsupervised Learning

The penalty factor  $C$ , also named *box constraint*, determines how much a data point outside the  $\varepsilon$ -tube must be penalized in the final solution. The performance of the SVR model varies depending on the proper optimization of these variables, but a detailed mathematical explanation of  $\varepsilon$  and  $C$  is beyond the scope of this dissertation, and it can be found in [57].

Each element of the  $x_i$  vector comprises multiple input data features, such as the raw data or some statistics. The primary idea of SoC estimation is to map some cell-related input features to the corresponding cell State of Charge. Usually, current, voltage and temperature are the most commonly employed input features, but in the literature, different feature extraction techniques have been studied, such as Principal Component Analysis (PCA) [59]. Most of the additional input features are related to mathematical operations or statistics, such as mean and standard deviation in the data. In Chapter 5, an advanced set of input features including the cell impedance is also investigated. A Kernel function performs the mapping, whose choice depends on the specific selected input features. The SVR estimation function for a new input vector  $x$  is (3.1).

$$f(x) = \sum_{i=1}^{N_{SV}} (\alpha_i - \alpha_i^*) K(x, x_i) + b \quad (3.1)$$





**Figure 3.4:** Support Vector Machines for regression tasks

where  $\alpha_i$  and  $\alpha_i^*$  are the Lagrange multipliers. The  $N_{SV}$  sample vectors associated with nonzero Lagrange multipliers are called SVs.  $K(x, x_i)$  is a kernel function that maps the input space  $R^n$  to a high dimensional feature space  $R^{n_k}$ , where regression is performed, and  $b$  is a bias term. Some possible choices [60], [61] for kernel functions in SoC estimation are:

- Linear:  $K(x, x_i) = x \cdot x_i$
- Polynomial:  $K(x, x_i) = ((x \cdot x_i) + p)^d$
- Gaussian Radial Basis Function (RBF):  $K(x, x_i) = \exp(-\|x - x_i\|^2 / 2\sigma^2)$

For each ML algorithm, large datasets are required for the training phase. For the purposes of this first approach, a large collection of data acquired on a Panasonic 18650 Li-Ion battery cell, publicly available [44], was used. The cell characteristics are detailed in Table 3.2.

In [44], the current profiles and the associated measurements were collected from real batteries applying some of the most popular drive cycles. The "Neural Network

TABLE 3.2  
PANASONIC NCR18650 ELECTRICAL CHARACTERISTICS

Chemistry	LiNiCoMnO <sub>2</sub>
Nominal capacity at 25 °C after standard CC-CV charge [Ah]	2.75
Nominal Voltage [V]	3.6
Voltage cut-off [V] (lower-upper)	2.5 – 4.2
Max. discharge current [A]	10
Standard CC-CV cut-off current [A]	0.10

drive cycle" was chosen for this study because it was specifically designed to be used within ML training processes [44].

Since developing new ML algorithms was beyond the scope of this dissertation, the MATLAB design suite and the embedded *fitrsvm* function [62] were used to train the SVR model with the aforementioned data. During the training process, various input features and kernels were experimented with to determine the best compromise between high accuracy (i.e., low error metrics) and low implementation complexity. Furthermore, to test the generalization capabilities of the models obtained with different kernel solutions, each was validated on a distinctive dataset from the same collection, namely the US06, one of the most aggressive drive cycles [63], [64]. It is worth noting that this dataset was not included in the training phase.

The details regarding the kernel validation results can be found in Section 3.2.4. Based on these results, a linear kernel was selected. The choice was also led by its low computational complexity [65], [66], which results in low resource utilization for the FPGA implementation while still providing good results in terms of RMSE. To perform successive inferences for a new input vector  $x$ , the trained linear kernel SVR model can be utilized by employing the following estimation function:

$$f(x) = \alpha^T \cdot \left( \frac{x}{KS} \cdot \frac{SV^T}{KS} \right)^T + b \quad (3.2)$$

where  $KS$  stands for the Kernel Scale and is a positive scalar value used to divide all input data before the kernel computation is performed. The stored vector  $\alpha$  represents the differences between the two Lagrange multipliers for each Support Vector  $SV$ . The scalar value  $b$  is the bias parameter in (3.1). Finally, the transpose operator is denoted by  $T$ .

### 3.2.2 Ant Colony Optimization

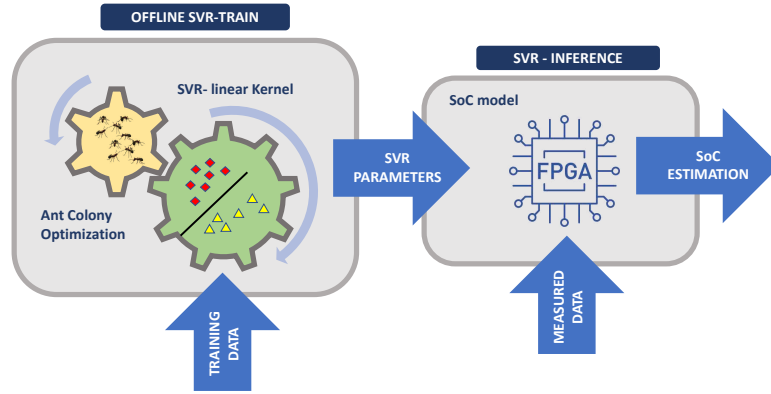
After selecting the input features and kernel, the SVR model must be properly trained. Typically, an optimization stage is used to obtain the optimal set of SVR parameters that result in the best SVR model. The optimal parameter set is usually evaluated using the cross-validation procedure, as described in Chapter 1. The MATLAB environment provides different built-in optimization algorithms, such as Bayesian and Grid Search optimizations [62]. Nevertheless, the literature shows that the Ant Colony Optimization (ACO) and the Particle Swarm Optimization (PSO) obtain the best results when fine-tuning the SVR model parameters [67], [68], [69]. The ACO and the PSO are part of a family of algorithms for continuous optimization, therefore their working principles are similar. Despite that, no evidence of ACO's previous application in the field of SoC assessment can be found at the time of this dissertation. The ACO algorithm is inspired by the foraging behaviour of ants [70]. When an ant discovers a food source, i.e., a set of SVM parameters, it leaves a trail of pheromones on the path leading to that source. The amount of pheromones depends on the quality of the food source, i.e., the quality of the model associated with those parameters (e.g., lower RMSE, higher quality). Other ants in the colony then move towards the path with the most quantity of pheromones, searching for a set of parameters close to the best one found so far. This helps to concentrate the search in regions of the search space containing high-quality solutions. The process continues until a certain number of iterations is completed or the changes to the parameters are smaller than a defined threshold.

The kernel scale  $KS$ , the approximation accuracy  $\epsilon$ , and the box constraint  $C$  were included in the optimization process of the SVR model since they are the same parameters used by the built-in MATLAB optimization algorithms [62]. This allowed

for a benchmark comparison to evaluate the effectiveness of ACO.

### 3.2.3 FPGA Implementation

The presented overall system architecture can be seen in Fig. 3.5. In the literature, it is common practice to perform the training phase offline since it involves storing large datasets and can take a long time to complete. Therefore, only the optimized SVR inference model was implemented.



**Figure 3.5:** Proposed Ant-Colony-Optimized SVR system architecture

The SVR inference equation to be implemented is (3.2). Manipulating the equation for an easier implementation with less resource usage, it can be greatly simplified if considering a parameter  $\beta$  defined as (3.3):

$$\beta = \alpha^T \cdot \left( \frac{SV}{KS} \right) \quad (3.3)$$

Furthermore, it is possible to define a scaled parameter  $\beta_{scaled}$  from  $\beta$  as (3.4)

$$\beta_{scaled} = \frac{\beta}{KS} \quad (3.4)$$

Finally, it is possible to transform (3.2) into (3.5) by substituting  $\beta$  and  $\beta_{scaled}$ .

$$f(x) = \langle \beta_{scaled}, x \rangle + b \quad (3.5)$$

where  $\langle \cdot, \cdot \rangle$  represents a scalar product.

In the end, the hardware architecture required to process new input samples and estimate a new SoC on an FPGA device involves computing a scalar product and the sum of the  $b$  parameter. Additionally, this approach eliminates the need for division operation, which can be complicated to implement on an FPGA.

The Hardware Description Language (HDL) code associated with (3.5) was authored and assessed within the Xilinx Vivado Design Suite framework. The VHDL code was a DSP-free architecture, allowing for implementation on a variety of FPGA boards, even on those that are not equipped with DSP slices, allowing the user also to choose cheaper boards, if the resource usage is suitable. A simpler hardware architecture was obtained by performing all the operations in fixed-point precision. To represent data ranging from -256 to 255.999 a signed 32-bit word length data with 23 fractional bits were chosen. This data type is compliant with the most common battery cells and modules [71] and allows a precision of  $1.2 \times 10^{-7}$ . It is worth noting that this data type is suitable for modules with up to about 60 cells in series. The VHDL code was implemented on a Xilinx Artix 7 [72] XC7A100T-1CSG324C FPGA. Resources utilization and timing simulations are reported in Section 3.2.4. Along with the SVR estimation model, a UART interface was implemented on the FPGA board for testing purposes. At each time step, the test input vector  $x$  is fed through this communication port, which finally sends back to the PC the SoC value to assess the estimation error. The hardware-implemented, ACO-optimized linear SVR model was finally evaluated against the US06-based dataset.

### 3.2.4 Results and Discussion

For a preliminary evaluation, different input features and kernel choices were tested. Each model was trained with the aforementioned NN drive cycle and tested on a different dataset, i.e., the US06 drive cycle. Finally, the optimum input vector comprised:

- Current ( $I$ )
- Voltage ( $V$ )

- Temperature ( $T$ )
- State of Charge at the previous time step ( $prevSoC$ )

with which the linear kernel resulted in the most accurate estimation when tested on the US06 drive cycle, as shown in Table 3.3. Indeed, including the State of Charge computed at the previous time step can significantly linearize the behaviour of the SoC observed in that specific time step, even if the global trend during cell discharge is known to be highly non-linear, improving the effectiveness of the linear kernel [73].

TABLE 3.3  
RMSE RESULTS FOR FOUR SVR KERNELS TRAINED ON THE NN DRIVE CYCLE  
DATASET AND TESTED ON THE US06 DRIVE CYCLE DATA.

Kernel	RMSE (%)
Linear	8.7
Quadratic	16.7
Cubic	35.9
Sigmoid	30.2
RBF	35.5

Consequently, the SVR model parameters were optimized with the ACO. The algorithm was initialized with a total ant colony population  $M$  of 30 ants, each performing 30 moves (i.e., number of iterations) during the search process. Each ant started searching for the best parameter set from random positions. This initialization choice resulted in the best compromise between model accuracy and processing time [74]. Each ant performed the *fitrsvm* function for the specified number of iterations, leading to a model with a specific combination of the aforementioned  $KS$ ,  $\epsilon$ , and  $C$  parameters. Finally, the model selected for the successive implementation was the one with the lowest RMSE. Moreover, the optimization process was compared with the MATLAB built-in optimization options, namely the *Bayesian*, the *Grid Search* and the *Random Search*. It has to be noted that, for each algorithm, the optimization results can change over different trials due to the stochastic processes [62]. Therefore, each

was run fifteen times and the mean RMSE and MAE were evaluated on the US06 dataset. Results are summarized in Table 3.4.

TABLE 3.4  
RMSE AND MAE RESULTS OVER 15 RUNS OF DIFFERENT OPTIMIZATION  
ALGORITHMS.

Algorithm	RMSE (%)			MAE (%)		
	Min	Mean	Max	Min	Mean	Max
Bayesian	1.9	9.6	25.3	1.6	8.4	21.8
Grid Search	1.6	7.2	18.8	1.3	6.1	15.0
Random Search	1.4	6.8	53.6	1.2	5.9	46.5
ACO	1.4	3.9	7.2	1.2	3.2	5.8

The *Random Search* and ACO obtained the lowest estimation error in terms of both RMSE and MAE. Nevertheless, the remaining optimization algorithms were found to be worse than with the ACO.

Once the ACO SVR architecture was established, a 10-fold cross-validation was performed to assess the accuracy in terms of the maximum error between the estimated and expected SoC. as detailed in Section 1.4.4. Therefore, in this case, the properly split NN drive cycle dataset was used for both the training and the test phases. The effectiveness of the ACO approach in SoC estimation was assessed by comparing the validation results with a recent PC-based implementation of an SVR algorithm employing the PSO [10] in which a maximum validation error of 2.5% was obtained. On the other hand, the proposed approach resulted in a maximum validation error of 1.2%, offering good performance.

Finally, the obtained optimized model was coded in VHDL as described in Section 3.2.3 and implemented on the FPGA after simulations in the Vivado Design Suite. The embedded system performance in SoC estimation was then tested with the US06 test dataset. Using a dataset different from that used in the training phase allows to evaluate the generalization capability of the approach. For clarity purposes, the datasets exploited at each training and test step are summarized in Table 3.5.

A first behavioural simulation was performed in the Xilinx Vivado Design Suite

TABLE 3.5  
SUMMARY OF DATASETS USED IN EACH EVALUATION STEP, FOR TRAINING AND TEST.

Phase	Train Set	Test Set
Evaluation of different Kernel functions	NN drive cycle	US06 drive cycle
Comparison between MATLAB optimization and ACO	NN drive cycle	US06 drive cycle
ACO-optimized Linear SVR cross-fold validation evaluation	NN drive cycle	NN drive cycle
FPGA-implemented ACO-optimized Linear SVR evaluation	Not performed	US06 drive cycle

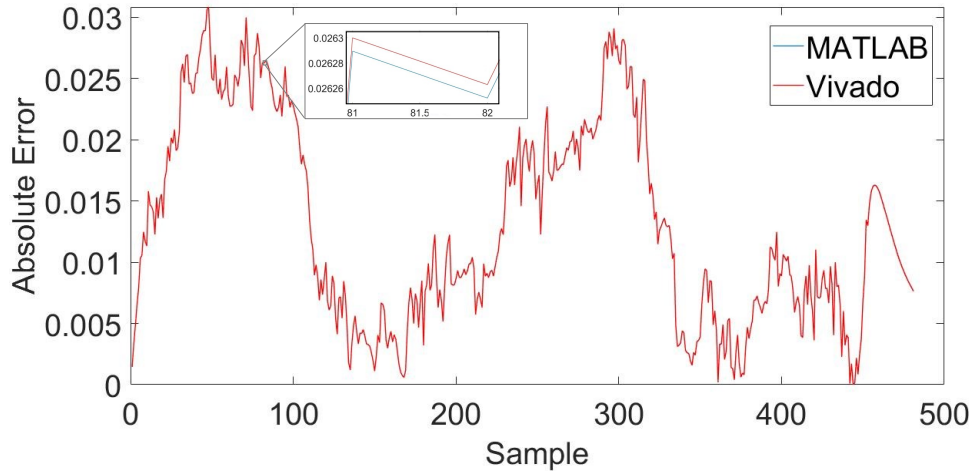
to evaluate the fixed-point quantization error. SoC estimation data are sent to MATLAB and compared with the US06 target SoC data, computing the Absolute Error as (1.6) where  $\hat{SoC}_k$  is the predicted SoC and  $y_k$  is the US06 target SoC.

The AE of the HDL-coded SVR was compared with that achieved with a MATLAB SVR simulation performed in double precision. As shown in Fig. 3.6, a maximum error of 3.1% was achieved. The figure demonstrates that it is difficult to distinguish between the MATLAB and Vivado (behavioural simulation) SoC estimations. Therefore, an enlarged image is included, revealing an error of approximately  $10^{-5}$  between the two approaches.

Next, post-implementation simulations were performed to evaluate the area occupation and timing performance. In Table 3.6, the occupied area is reported.

When compared to other FPGA-implemented techniques, e.g., the Neural Network architecture in [55], the presented approach requires very limited resources, employing only 1.39% of the available LUTs and 0.24% of the Slice Registers. This means that multiple instances of the proposed algorithm can be placed on the same FPGA. This feature can be of primary importance when several cells need to be monitored in parallel, such as in a battery pack. For example, the proposed approach can easily manage the case of a pack composed of 60 cells in series. By comparison, the





**Figure 3.6:** Absolute Error compared with the target US06 SoC for the Vivado fixed-point behavioural simulation and the MATLAB floating-point double precision.

ECM model presented in [B1] occupied 23% of the entire FPGA, which prevented more than four instances from being programmed on the same platform. Furthermore, in contrast with [55], the proposed approach does not require DSP slices. This means that smaller and low-cost devices can be used instead. Moreover, the post-implementation timing performance resulted in Worst Negative Slack = 0.188 ns, Worst Hold Slack = 0.054 ns, and Worst PulseWidth Slack = 4.5 ns, considering a clock frequency of 100 MHz.

Finally, the RMSE of the proposed solution was evaluated on the implemented circuit, again using measured data not included in the offline training phase. The US06 test data were sent to the FPGA through the designed VHDL UART module. The processed SoC estimation values were sent back to the MATLAB environment and compared to the expected SoC to assess the RMSE. In Fig. 3.7, the estimated SoC is compared with the expected US06 SoC values. The FPGA computations (red line) predict the cell SoC (blue line) with an RMSE of 1.4% and an MAE of 1.2%. The behavioural simulation completely overlaps the final implementation results hence it

TABLE 3.6  
AREA OCCUPATION COMPARISON WITH OTHER STUDIES.

	FPGA	Slice LUTs Utilization (%)	Slice Registers (%)	DSP Slices (%)
Proposed	Artix 7	880/63.400 (1.39%)	300/126.800 (0.24%)	0/240 (0%)
ECM [B1]	Artix 7	14.427/63.400 (22.76%)	196/126.800 (0.15%)	8/240 (3.33%)
NN [55]	Virtex 7	1123/303.600 (1%)	751/607.200 (1%)	1125/2800 (40%)

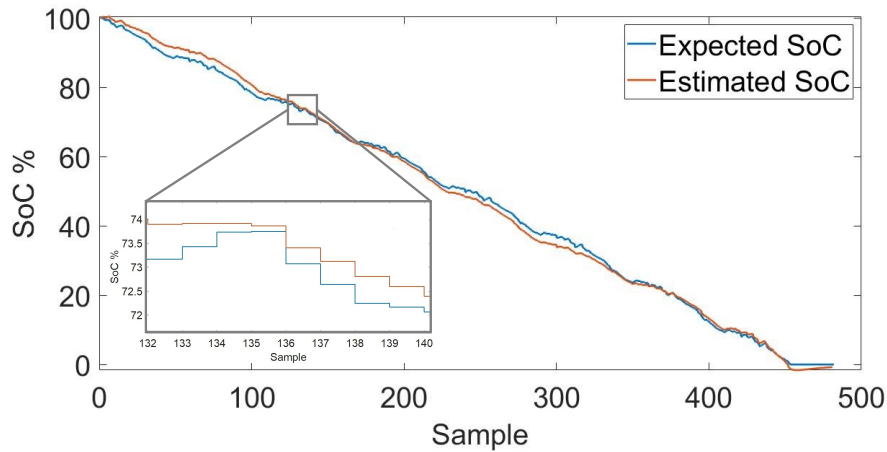
is not shown in Fig. 3.7.

Further results can be investigated by comparing this approach with a model-based approach and with other literature employing SVR models. In Table 3.7, a Thevenin ECM was considered either with constant model parameters or with variable parameters indexed in LUTs.

TABLE 3.7  
COMPARISON OF PERFORMANCE IN SoC ESTIMATION BETWEEN SVR APPROACH, ECM APPROACH EITHER WITH CONSTANT AND SoC-VARYING PARAMETERS, AND THE LITERATURE.

Approach	Kernel	RMSE (%)	Max Error (%)	MAE (%)
ACO-SVR	Linear	1.4	3.1	1.2
ECM const. par.	-	1.5	4.3	1.1
ECM var. par.	-	2.5	5.4	2.2
[17]	Quadratic	2.5	13	-
[10]	RBF	-	1.5	1.2

The ACO-SVR model outperforms both cell model-based approaches in terms of estimation accuracy. This may be due to the loss of accuracy in the LUT stored parameters when the battery cell is almost fully charged or discharged. The same



**Figure 3.7:** Comparison of expected SoC with SoC processed data on the FPGA board.

MAE was obtained as in [10], but in the proposed approach, the feasibility in real environments has been proved by implementing the algorithm on an embedded system and tested on different test sets. Moreover, estimation accuracy overcomes [17], also reducing the kernel complexity.

Concluding, this proposed solution allows the estimation of SoC without the need to model the particular cell with an ECM. After the training with a large dataset and the ACO, the obtained model was implemented on a FPGA. The successful SoC estimation resulted in comparable or better accuracy than the literature while maintaining a low resource usage.



## Chapter 4

# Application-Independent Training for Support Vector Machines

In the literature, methods for estimating the SoC of batteries are usually evaluated through current profiles obtained from standardized drive cycles, which are commonly used for computing fuel consumption in traditional thermal engine vehicles. The reason behind this is the growing market for electric vehicles and the associated increased quantity of battery packs mounted on EVs. Therefore, research efforts have been focused on the EV field. In particular, drive cycle based datasets are used to train and test ML models. This approach yields high accuracy in predicting SoC; however, it also means that the developed system relies on the specific application and discharge profiles used in its training. If the application field changes, the training procedure must be executed again, which is a long process and requires recording a new set of measurements.

After the validation of the ML approach proposed in Chapter 3, the goal of this dissertation was then to further generalize the model training phase employing a novel approach. A DDM based on SVR can be trained using data acquired while applying a constant current to the battery cell. The constant current profile is very generic as it is not related to a specific application, making the inferred model likely to be applied to any field which requires a lithium battery, not just the automotive

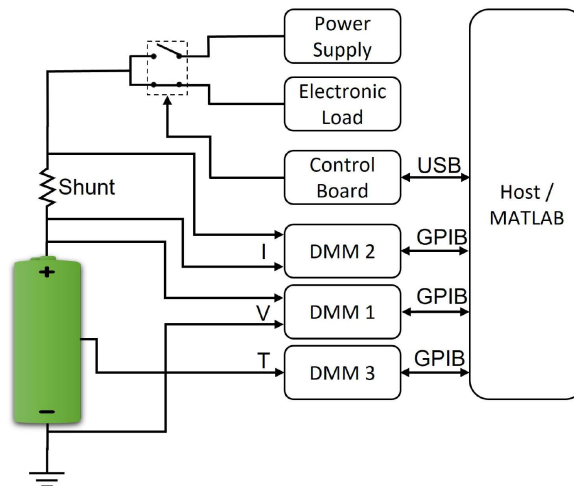
sector. This means that cell manufacturers or research laboratories can easily train a model independently from the final specific application, reducing the modelling effort required by BMS designers.

## 4.1 Preliminary Approach Validation

A preliminary validation was performed on a reduced-size dataset to evaluate the effectiveness of the proposed approach. A dataset including constant current cell data had to be acquired for the training phase thus the data acquisition setup has been designed to gather data from a widely used 2.75 Ah Panasonic NCR18650PF lithium-ion cell [71], [75]. Cell characteristics have been detailed in Table 3.2.

### 4.1.1 Data Acquisition Workbench

The setup overview is shown in Fig. 4.1.



**Figure 4.1:** Data acquisition schematic overview.

An Agilent E3646A dual-output DC power supply was employed for charging the cell. The standard CC-CV charging process was applied to avoid cell damages,

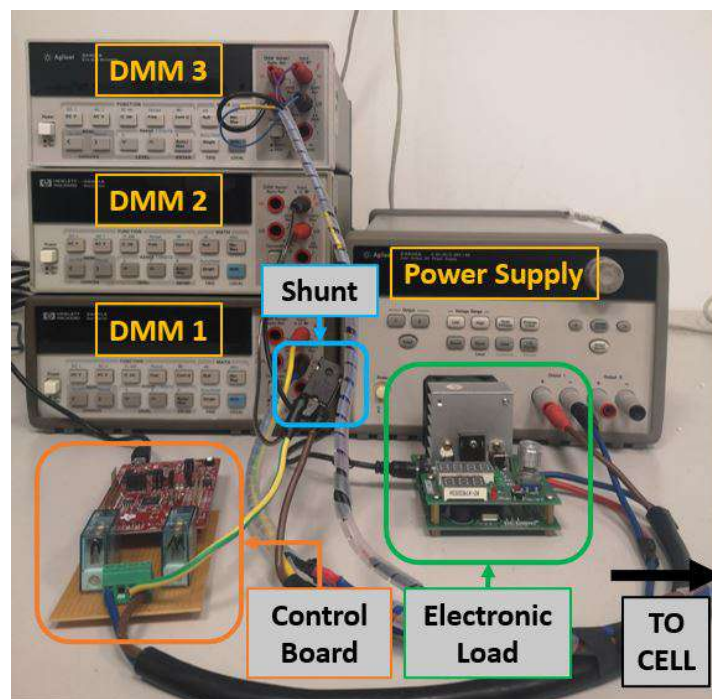
according to the specific datasheet [71]. On the other hand, a ZHIYU Multifunction 110 W, 30 V, 9.99 A electronically controlled load discharges the cell at a constant current, with a maximum error in current sinking of  $0.7\% \pm 10$  mA, equivalent to a maximum current error of 80 mA. The battery cell is connected in parallel to both devices output via an 18650 battery holder. The CC3200 MicroController Unit (MCU) equipped on a Texas Instrument's LaunchXL board controls charges and discharges operations through two digital outputs. During operations, two HP 34401A 6  $1/2$  digits DMMs acquire the cell voltage and current in a 4-wire configuration. The flowing current is converted in a differential voltage and measured by Digital MultiMeter (DMM) DMM2 across a PBH  $10\text{ m}\Omega \pm 1\%$  shunt resistor. The cell voltage is measured at the cell terminals by DMM1, removing most parasitics voltage drops on wires. Finally, a third DMM (i.e., DMM3) acquires the temperature data, sensing the resistance of a  $10\text{ k}\Omega \pm 1\%$  NTC thermistor longitudinally applied on the surface of the cell. The NTC thermistor has  $10\text{ k}\Omega$  resistance at  $25\text{ }^\circ\text{C}$ , and a characteristic material temperature of  $3435\text{ K}$ . Sampling data with a reading speed of 10 readings/s is a trade-off between the monitoring accuracy and the computational demand due to the quantity of sample points [76]. In Table 4.1, detailed considerations on instrument configuration, resolution and accuracy are presented. It must be

TABLE 4.1  
DIGITAL MULTIMETER SETTINGS AND SPECIFICATIONS

	Instrument		
	DMM 1	DMM 2	DMM 3
Measured Cell Feature	Voltage	Current	Temperature
Configuration	Voltage	Voltage	4-wire resistance
Range	10 V	0.1 V	100 k $\Omega$
Resolution	100 $\mu\text{V}$	1 $\mu\text{V}$	1 $\Omega$
Accuracy	0.0035 +	0.0035 +	0.01 +
$\pm$ (% reading + % range)	0.0005	0.0005	0.001

## 54 Chapter 4. Application-Independent Training for Support Vector Machines

noted that the DMM2 voltage resolution of  $1 \mu\text{V}$  in the  $0.1 \text{ V}$  range corresponds to a  $100 \mu\text{A}$  current resolution. Moreover, the DMM3 measured the NTC resistance in the  $100 \text{ k}\Omega$  range, with a  $1 \Omega$  resolution, not considering temperatures below  $-20 \text{ }^\circ\text{C}$ . All DMMs have the auto-zero function enabled to prevent drift, while the auto-range function is disabled for faster readings. This setup allows for a maximum of  $5 \frac{1}{2}$  digits. The entire workbench is remotely controllable through a specifically designed MATLAB script. An Agilent 82357B USB/GPIB interface allows communication with the daisy-chained DMMs by sending IEEE 488 standard SCPI commands. The CC3200 MCU communicates via USB serial connection with the MATLAB script, controlling the cell operations. The actual complete workbench is shown in Fig. 4.2



**Figure 4.2:** Data acquisition workbench for constant current measurements.

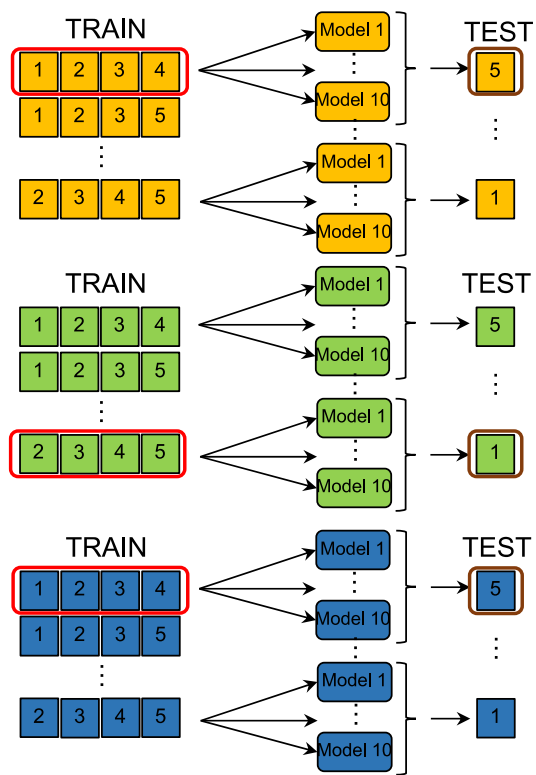


### 4.1.2 Model Training

A total of 15 discharge cycles was collected, i.e., a set of five cycles (numbered #1 to #5) for three different current rates, i.e., 1C, 2C and 0.5C, at ambient temperature ( $23\text{ }^{\circ}\text{C} \pm 2\text{ }^{\circ}\text{C}$ ). Each cycle was followed by 30 minutes of rest to allow polarization effects to disappear. The cell current, voltage and temperature were used as input features. Before the training phase, the expected SoC was computed using the CC method, as all the supervised ML algorithms require a target vector to be trained.

For each set, 80% of the data was assigned to the training set, and the remaining 20% was used for validation. The RBF kernel function was used in the SVR model, chosen for its good performance in various applications [13], [77], [38]. The MATLAB built-in *Regression Learner* tool was used to optimize the SVR model with the *Bayesian* algorithm, which ran for a default total of 30 iterations. For each iteration, the tool performed the 10-fold cross-validation and the optimum SVR parameter set was finally obtained. This was repeated for each of the available cycle combinations, to avoid biasing the result against one specific cycle combination. The considered error metrics were the Root Mean Squared Error, the Mean Absolute Error and the maximum Absolute Error, which are formulated in (1.4), (1.5), and (1.6). The optimization algorithm had stochastic initialization, hence, the optimization process was repeated ten times, obtaining ten models for each possible cycle combination. The model effectiveness is assessed by computing the average of error metrics. They were then tested with the validation set, as shown in Fig. 4.3. The different current rates are represented by different colours, and the numbers denote the cycles on which the models were trained. The best cycle combinations for each current rate are marked with a red circle and were included in a new training phase for a global SVR.

Three different test sets, emulating dynamic current profiles, were built by randomly mixing the 20% of the cycles allocated for the test. Therefore, the samples were randomly chosen but still preserving a monotonic decreasing SoC profile, thus emulating a discharge cycle at different current rates. The resulting error metrics are listed in Table 4.2. As can be seen, the optimized SVR model shows good accuracy in SoC estimation, with an RMSE lower than 1% in each test set. It is worth noting that the error metrics obtained from the tests seem to be lower than what is reported

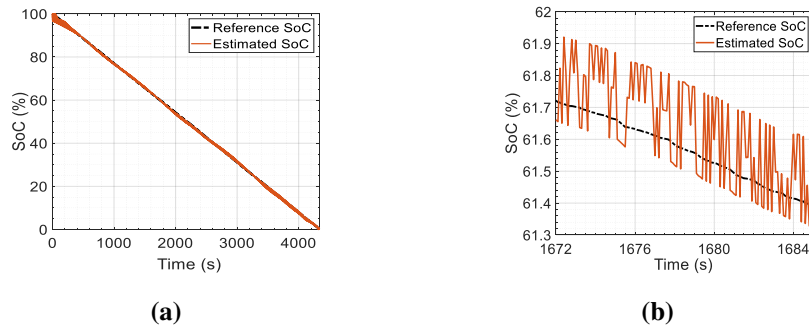


**Figure 4.3:** SVR training and testing workflow.

TABLE 4.2  
SVR MODEL ERROR METRICS ON DIFFERENT DISCHARGE CYCLES

Test Set	RMSE (%)	MAE (%)	Max AE (%)
Dynamic Cycle 1	0.576	0.409	3.2
Dynamic Cycle 2	0.578	0.412	3.2
Dynamic Cycle 3	0.564	0.405	3.2

in the literature. However, the reason is that the tests were conducted on a subset (20%) of the total dataset. As a result, some of the information from the test data is also already partially included in the training set. This is a common practice during *validation* and differs from testing the model with completely new data. The resulting SoC estimation for the test dynamic profile 1 is shown in Fig. 4.4, compared with the reference SoC. An enlarged view is also shown for clarity purposes.



**Figure 4.4:** (a) Comparison between the reference CC computed SoC and the SVR estimated SoC. (b) Detail of the SoC estimation profile

Nevertheless, the results when training with constant current data and validating with emulated dynamic profiles showed promising error metrics so that the approach could be extended to a more detailed dataset.

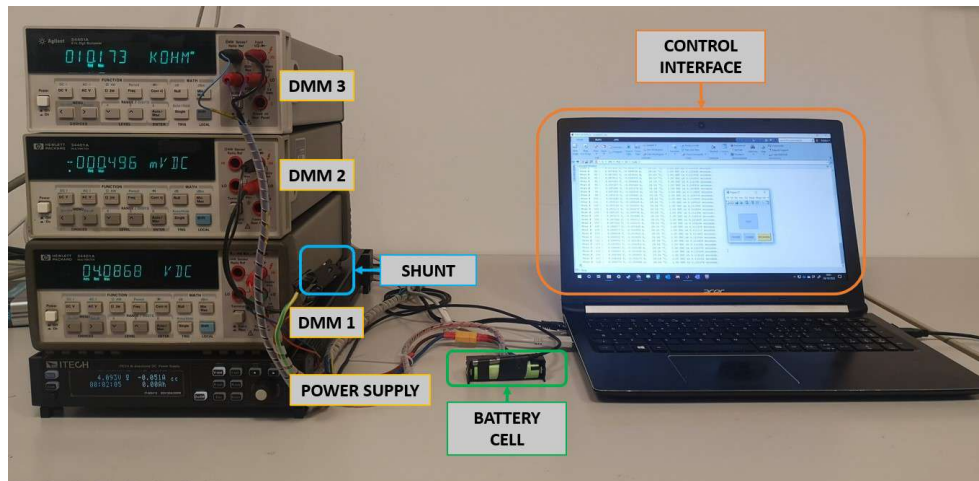
## 4.2 Application Independent Training Data

Next, the aim was to train the SVR with a multitude of constant current profiles in a wider current range, and then test it with real-world dynamic profiles. For this reason, the current range 2 A - 4 A was chosen. This resulted from a trade-off between the cell discharge rate (which affected the time duration of the data acquisition campaign) and the actual discharge current a device usually experiences, e.g., quadcopters, smartphones, cordless power tools or medical devices. Moreover, a wide test set composed of different realistic dynamic profiles had to be acquired. This led to the need for an update to the workbench for data acquisition.

### 4.2.1 Improved Workbench

The workbench in Fig. 4.2 was capable of discharging the cell through an electronic load. This device can not be programmed in real-time, therefore discharging the cell with a dynamic profile was not possible.

The workbench was modified as in Fig. 4.5, replacing both the power supply and the electronic load with a programmable 200 W ITECH IT-M3412 bidirectional power supply. It is rated for 60 V and  $\pm 30$  A, hence it is perfectly suitable for cell-level operations. According to the datasheet [78], the accuracy is better than 0.1% of the maximum voltage when working in constant voltage mode, whilst the programming accuracy in constant current mode is better than 0.1% plus 0.1% of the full scale. The power supply works in a 4-wire configuration, thus sensing the cell voltage directly on the terminals, removing the wires parasitics voltage drops. Introducing a bidirectional power supply also allowed the removal of the control board for switching between the charge and discharge operation. The cell under test was again a 2.75 Ah Panasonic NCR18650, and the DMMs were configured as described in Table 4.1. To minimize parasitic voltage drops, the system was wired using AWG 14 copper wires for power delivery, compliant with the wire section range recommended in the IT-M3412 datasheet. The workbench is capable of remote and autonomous operations and a MATLAB script designed specifically for this purpose communicates with each DMM using IEEE 488 standard SCPI commands for initial configuration



**Figure 4.5:** Improved workbench for data acquisition.

and memory data fetching. The same script interfaces through an ethernet connection with the IT-M3412 power supply by sending SCPI commands to its fixed IP address. The script autonomously controls the CC-CV charging procedure and the cell technical discharge limits outlined in the datasheet. It can also start and stop the acquisitions or repeat them for a specific number of iterations while managing proper rest periods for dealing with voltage relaxation. The autonomous script aided in building a large dataset in a relatively short amount of time.

#### 4.2.2 SVR Training Phase

The training dataset consisted of current, voltage and temperature measurements acquired during discharge cycles at the specified constant current rates. All the cycles have been started at the ambient temperature.

## 60 Chapter 4. Application-Independent Training for Support Vector Machines

---

For the first investigation, data from constant current discharges at rates between 3 A and 4 A in steps of 100 mA were acquired. For each current rate, a total of five discharges were performed, totalling 55 cycles for the training set. With a reading every 100 ms, the dataset was composed of more than a million data samples.

To build a large test set, several test cycles were also acquired. Five different current profiles were generated by applying random current values between 3 A and 4 A to be used as test cycles. A minimum current variation of 0.1 mA was set, which is different from the 0.1 A steps used for the acquired current rates. This ensured that the test set also included discharge current values that were not present in the training set. Additionally, for further testing, some realistic battery-powered drill current profiles [79] were also acquired and appropriately scaled to fit the considered current range. Finally, a US06 drive cycle current profile was acquired to validate the proposed approach and compare it against the literature in which drive cycle profiles are commonly used.

Once the dataset was acquired, the SVR model was trained following the same procedure as before. The reference State of Charge was computed by employing the Coulomb Counting algorithm, as (2.1). The high resolution and accuracy of the employed DMMs and the laboratory environment make the CC suitable for the specific dataset. Finally, the input vector comprised the current, voltage and temperature raw data as features, and the target vector was the CC computed reference SoC. The *Bayesian* optimization for the RBF kernel [77], [38] was performed in the *Regression Learner* tool. The optimum SVR parameters were obtained after 100 optimization iterations, in which the 10-fold cross-validation was applied to identify the lowest RMSE.

### 4.2.3 Dataset Downsampling

As demonstrated in [80], a BMS can efficiently manage battery cells by sampling data at 2 Hz. Initially, a downsampling factor of five was applied to reduce the dataset size and speed up computations, reducing from a reading each 100 ms to a reading each 500 ms.

The training dataset composed of 55 cycles in the 3 A to 4 A range downsampled

to 2 Hz required about two weeks to obtain a model with a validation RMSE of 1.4% on a personal computer with an 11th generation Intel i9-11900 core at 2.5 GHz and 32 GB RAM. The training time was quite long, considering that multiple SVR models may be trained for different ranges. It has to be noted that in this validation, the same kind of data was used for train and test, i.e., constant current. The effects of a more severe downsample factor were investigated. The aim was to minimize the training time while maintaining the same accuracy in the SoC estimation. The resulting datasets included data with a sample every 1 s (downsample factor 10), 10 s (downsample factor 100), and 100 s (downsample factor 1000). The required training timing and the related validation accuracy are reported in Table 4.3.

TABLE 4.3  
EFFECTS OF DATA DOWNSAMPLING ON TRAINING TIME AND ACCURACY

Training Data Readings/s	Training Time	RMSE (%)
2	15 days	0.9
1	6 days	0.6
0.1	7 hours	1.1
0.01	5 minutes	1.3

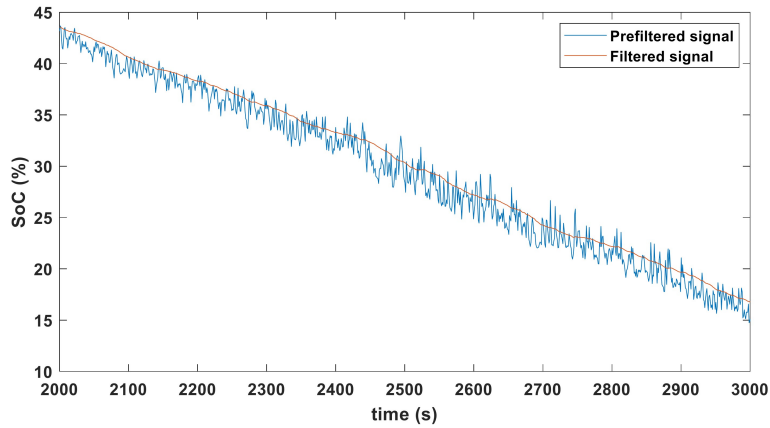
As can be seen in Table 4.3, the training time drop is not linear with a higher down-sampling because it depends not only on the dataset size but also on other parameters such as the processor computations. Moreover, considering the validation accuracy, it can be seen that a dependency with the downsampling factor is not straightforward. The significant improvement in the training time and a limited worsening in the validation accuracy made the 100 s dataset the best option for further investigation.

#### 4.2.4 Post-Processing Stage

To overcome the abrupt changes in the predicted value of SoC that AI approaches are prone to produce [50], data post-processing stages were implemented to further

## 62 Chapter 4. Application-Independent Training for Support Vector Machines

improve the estimation accuracy. The post-processing stages were intended to be applied on-board to the specific application data, thus they had to be easy to implement. Firstly, the SoC was limited to discard unfeasible values, i.e., values beyond the physical threshold of 100% - 0%. The spikes in the SoC estimation were then smoothed by a 64-tap low-pass FIR filter, considering a normalized cut-off frequency of 0.01 Hz. An example of the resulting filtering is shown in Fig. 4.6.



**Figure 4.6:** Prefiltered SoC (blue line) compared with the filtered SoC signal (red line).

Finally, an error calibration stage was implemented. The error  $\varepsilon$  can be expressed as:

$$\varepsilon = \hat{SoC} - y \quad (4.1)$$

A linearly fitted calibration line can be used to correct the estimated SoC, i.e.,  $\hat{SoC}$ . Given  $m$  and  $q$  as the slope and intercept of the calibration line, the corrected SoC can be computed as:

$$SoC_c = \frac{\hat{SoC} - q}{1 + m} \quad (4.2)$$

To account for non-linearities in the full 100% - 0% range, it was split into smaller 10% width ranges, obtaining a calibration line for each of them. To compute the linear



fit, the Simple Linear Regression (SLR) algorithm was considered [81], [82]. This is an efficient formulation for the MATLAB *polyfit* function, which, on the other hand, implements the more complex general least squares algorithm. The SLR minimizes the residual errors between each data point and the best first-order polynomial. The cost function to be minimized is:

$$J(m, q) = \sum_i^n (y_i - mx_i - q)^2 \quad (4.3)$$

where  $(x_i, y_i)$  are the coordinates of each data point to be fitted and  $n$  the total number of points. To minimize the cost function, the partial derivatives must be equal to zero, obtaining:

$$\sum_{i=1}^n y_i = nq + m \sum_{i=1}^n x_i \quad (4.4)$$

$$\sum_{i=1}^n x_i y_i = m \sum_{i=1}^n x_i^2 + q \sum_{i=1}^n x_i \quad (4.5)$$

The complete mathematical solution is depicted in [81], [82]. When introducing the sum of squares, defined as (4.6) - (4.9), the solution can be significantly simplified.

$$S_x = \sum_i x_i \quad (4.6)$$

$$S_y = \sum_i y_i \quad (4.7)$$

$$S_{xx} = \sum_i x_i^2 \quad (4.8)$$

$$S_{xy} = \sum_i x_i y_i \quad (4.9)$$

Then, the fit line can be computed through the pseudo-code in Alg. 1.

When considering a 32-bit microcontroller, the averaged error in the  $m$  and  $q$  parameters compared to the MATLAB *polyfit* function are of  $3.5 \times 10^{-5}$  and  $2.5 \times 10^{-5}$ , respectively. As observed in the pseudo-code, the computational complexity is  $O(n)$ , much lower than the training phase whose complexity is  $O(n^3)$  [83]. Therefore, the training phase is unlikely to be performed online. However, this calibration stage

## 64 Chapter 4. Application-Independent Training for Support Vector Machines

---

---

**Algorithm 1** Simple Linear Regression algorithm

---

```
1:  $S_x \leftarrow 0$ 
2:  $S_y \leftarrow 0$ 
3:  $S_{xx} \leftarrow 0$ 
4:  $S_{xy} \leftarrow 0$ 
5:  $n \leftarrow$  number of data points
6: for  $i = 1:n$  do
7:    $S_x \leftarrow S_x + x(i)$ 
8:    $S_y \leftarrow S_y + y(i)$ 
9:    $S_{xx} \leftarrow S_{xx} + x(i) \cdot x(i)$ 
10:   $S_{xy} \leftarrow S_{xy} + x(i) \cdot y(i)$ 
11: end for
12:  $m \leftarrow (n \cdot S_{xy} - S_x \cdot S_y) / (n \cdot S_{xx} - S_x \cdot S_x)$ 
13:  $q \leftarrow (S_{xx} \cdot S_y - S_x \cdot S_{xy}) / (n \cdot S_{xx} - S_x \cdot S_x)$ 
14: return  $m$  and  $q$ 
```

---

can be implemented on devices that have limited computing resources, calibrating the system for the user's application-specific field. Therefore, in the final application, the algorithm will calibrate the error in the SoC estimation during the first operating cycle, store the  $m$  and  $q$  parameters, and apply them to the following cycles.

### 4.2.5 Testing Phase

For the 3 A - 4 A range, the five dynamic profile test cycles were tested on the obtained SVR model. The RMSE and MAE error metrics were considered for the accuracy assessment. To apply the post-processing stage, one of the test cycles was used to calibrate the linear fit parameters, which were applied to the remaining four tests. To avoid the choice of the calibration test influencing the results, this approach was repeated on each of the five tests. Then, it has to be noted that the specific SVR model accuracy was defined as the averaged RMSEs and MAEs obtained during the process. The effectiveness of the post-processing stages is shown in Table 4.4, where

the error metrics are shown after each stage. As can be seen, the filtering reduces the

TABLE 4.4  
TEST RESULTS WITH RANDOM DYNAMIC PROFILES IN THE RANGE 3A-4A

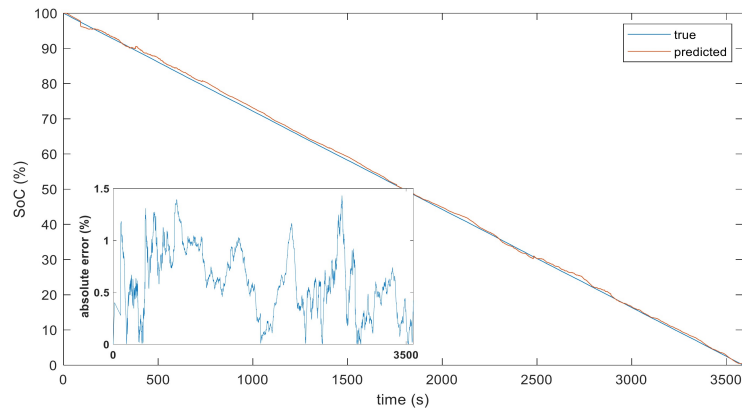
Stage Output	RMSE (%)	MAE (%)
SVR model	3.98	3.57
FIR Filter	2.87	2.49
Error Correction	1.2	1

RMSE of 28% and then the error correction stage of a further 58%, down to a RMSE of 1.2% and MAE of 1%. In Fig. 4.7, the estimated and the expected SoC are shown, along with the absolute error.

The training and testing procedure was repeated for the 2 A - 3 A range and the totality of the dataset (2 A - 4 A range) to validate the effectiveness and flexibility over different situations. The 2 A - 3 A range dataset was composed of 55 cycles acquired as the 3 A - 4 A range dataset, and the final dataset was composed of a total of 105 cycles. The five dynamic profiles in the range 2 A - 3 A range yielded similar results in terms of RMSE and MAE as that of the 3 A - 4 A range. For the 2 A - 4 A range, two approaches were proposed. In the first approach, a single SVR was trained with the entire 2 A - 4 A range dataset, while in the other approach, the two models, 2 A - 3 A range and 3 A - 4 A range, were used together, applying an input decision logic to choose the most suitable model according to the specific test input vector current feature. In Table 4.5, the results of the tests in each current range are summarized, showing consistency over each current range.

To further validate the proposed technique and make a fair comparison with the literature, the RMSE and the MAE were evaluated in the case of two different and more realistic application fields: some discharge cycles following the typical behaviour of a battery-powered drill and other drive cycles acquired according to the US06 standard speed profile [63].

The battery-powered drill current consumption was obtained in [79] by measuring the current during a screwing process. The current profile shows a spike at the



**Figure 4.7:** SoC value at the output of the system (red line) compared to the reference (blue line). In the box, the absolute error.

startup, and then the current falls and rises again as the friction of the screw increases. Finally, a higher current value is due to the conclusive tightening.

On the other hand, the US06 is defined as vehicle speed over time. The speed set points were converted to current set points by a specifically designed vehicle simulator as detailed in Appendix A. The reference vehicle was a 2012 Tesla Model S 75.

For testing purposes, both the vehicle and drill current profiles have been scaled to the required current range, still preserving the shape. In the case of the power tool an RMSE of 0.90% and an MAE of 0.60% were obtained, while in the case of US06

TABLE 4.5  
TEST RESULTS WITH RANDOM DYNAMIC PROFILES IN DIFFERENT CURRENT RANGES

Current Range	RMSE (%)	MAE (%)
2A - 3A	1	0.89
3A - 4A	1.2	1
2A - 4A (single SVR model)	0.8	0.65
2A - 4A (multiple SVR models)	0.77	0.61

drive cycle an RMSE of 0.96% and an MAE of 0.76% were achieved. In Table 4.6, the average of the tests is reported and compared with the literature.

TABLE 4.6  
COMPARISON WITH THE STATE OF THE ART

Work	Application field	Same profiles for training and test	RMSE (%)	MAE (%)
[38]	Automotive	YES	1.18	0.94
[39]	Automotive	Partially	2	–
ACO Chapter 3	Automotive	NO	1.4	1.2
[40]	Automotive	NO	2.47	–
[49]	Automotive	NO	1.51	1.32
[41]	Automotive	NO	1.37	1.12
<b>Proposed</b>	<b>Miscellaneous</b>	<b>NO</b>	<b>0.94</b>	<b>0.75</b>

It is important to note that many of the studies that have been reported use the same profiles for both the training and testing phases, as in [38]. In [39], some constant current profiles are used in the training set together with drive cycles profiles for training an RBF-Neural Network. In [40], a DCNN was trained with drive cycle

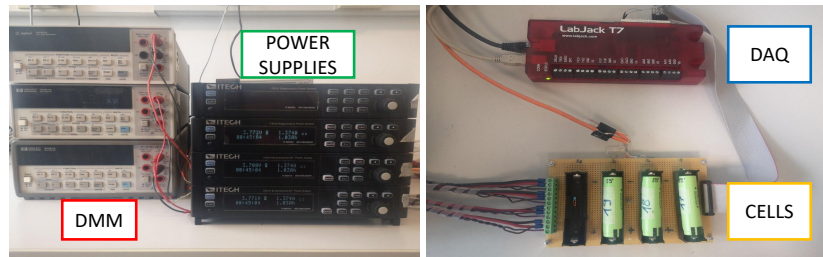
## 68 Chapter 4. Application-Independent Training for Support Vector Machines

---

profiles with the aim of applying the model to other cells with transfer learning, similarly to [41]. This justifies higher error metrics. In [49], a hybrid approach is trained and validated with drive cycle profiles. Nevertheless, the proposed method allows for a more general approach while still achieving similar performance. Some works [47], [50] achieved even lower errors, but by using more complicated methods and a higher number of input features. On the other hand, the proposed method achieved lower RMSE and MAE than papers exploiting different training and test sets. These results confirm the feasibility of the method based on SVR training with constant currents, which allows a more general approach to the problem of SoC estimation.

Finally, the computational cost was evaluated for the SoC assessment. With the SVR model trained with the 2 A - 3 A dataset, the SoC value was inferred and post-processed in 21.2 ms, while 18.2 ms were necessary with the 3 A - 4 A range SVR model. This is also very important when considering the actual implementation of the algorithm in a BMS, which must accomplish different monitoring tasks in a short amount of time for different cells. It is worth considering that if the implementation relies on a microcontroller device, the operating frequency will limit the maximum number of cells that can be evaluated. This is because the SoC must be estimated sequentially for each cell. However, using an algorithm that has a low computational cost, like the one proposed, allows to monitor multiple cells at the same time. Alternatively, if the BMS includes a FPGA, the algorithm can be replicated multiple times in hardware. This would enable multiple cell SoCs to be evaluated simultaneously on the same board. This approach could be extended to more comprehensive current ranges or to different cells, e.g., different chemistries or format factors. Moreover, acquiring data from different battery cells allows to train ML models with datasets composed of various data that help in the model generalization if applied to different kinds of cells.

However, a larger dataset must be built. An improved workbench was developed to fasten data acquisition on multiple battery cells at the same time. A board with a set of four 18650 battery holders was wired with four IT-M3412 power supplies, as shown in Fig. 4.8. Each battery holder and power supply was completely inde-



**Figure 4.8:** Workbench for parallel data acquisition on multiple cells.

pendent. A flat cable connects to a LabJack T7 data acquisition (DAQ) board, which gathers voltage data from the cell terminal and the temperature data from a  $10\text{ k}\Omega$  NTC placed on the cell surface. Finally, the current is derived from the voltage sensed across a PBH  $10\text{ m}\Omega$  shunt resistor and measured by three DMMs in voltage measurement mode. The fourth cell was not connected due to the lack of a fourth DMM, but the system is ready to be used. The workbench has been successfully validated by performing some measurements on three battery cells at the same time. This paragraph was only meant to provide useful information on a setup for parallel data acquisition, and will not be used in this dissertation because data acquisition is still in progress.





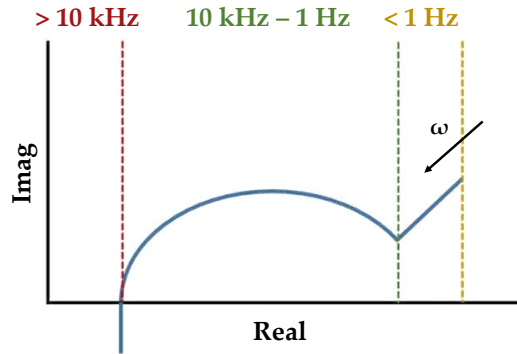
## Chapter 5

# Advanced Input Features Investigation

As detailed in Section 2.2 and Chapter 1, the internal impedance of a battery cell is one of the most significant indicators of its SoC. Impedance variations can be observed during cell operation, and a dependency on current and temperature has also been studied [29], [27]. The Electrochemical Impedance Spectroscopy is a well-known technique for examining the internal properties of electrochemical systems [84]. During the PhD, in the context of an International Mobility Program at the Department of Electrical Engineering and Electronics of the University of Liverpool, the battery cell impedance was evaluated as an additional input vector feature to be included in the training set of a SVR model. Constant current cycles data were used rather than application-specific data, as this approach has been proven successful in Chapter 4. The main goal was to evaluate the effects on SoC estimation when including the impedance spectrum and possibly identify a specific frequency, or a set of frequencies, at which the SoC estimation performs better [B4].

## 5.1 Online Electrochemical Impedance Spectroscopy

As detailed in Section 2.2.1, the EIS analysis results in the impedance frequency spectrum by applying (2.7) in the frequency domain. A single frequency perturbation can be iteratively applied to obtain the full (but still discrete) impedance spectrum [85], as in Fig. 5.1. However, this is a time-consuming process, as the cell must be com-

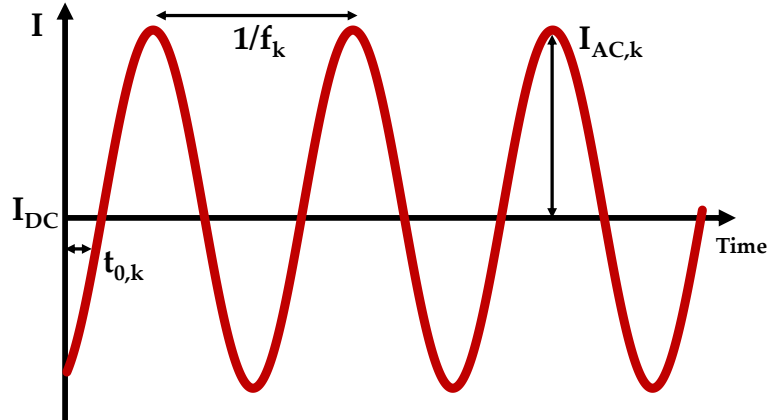


**Figure 5.1:** Typical cell impedance spectrum

pletely charged and discharged for each frequency that has to be applied. This is not appropriate for online operations where gathering as much information as possible in a limited time is crucial. Several papers proposed to measure the impedance at multiple frequencies simultaneously [86], [87]. To achieve parallel acquisition, the cell must be subjected to a multi-sine current signal, defined as (5.1):

$$I(t) = I_{DC} + \sum_k I_{AC_k} \sin(2\pi f_k(t - t_{0_k})) \quad (5.1)$$

where  $I_{DC}$  refers to the DC component, and the terms inside the summation are the AC perturbations. The  $I_{AC}$  and  $\varphi_k = 2\pi f_k t_{0_k}$  are the amplitude and phase shift, respectively, of the  $k$ -th sine signal at the  $f_k$  frequency, as Fig. 5.2. Then, for each  $f_k$  frequency, the corresponding impedance value can be computed. This approach allows the acquisition of impedance data at multiple frequencies in a shorter timeframe,



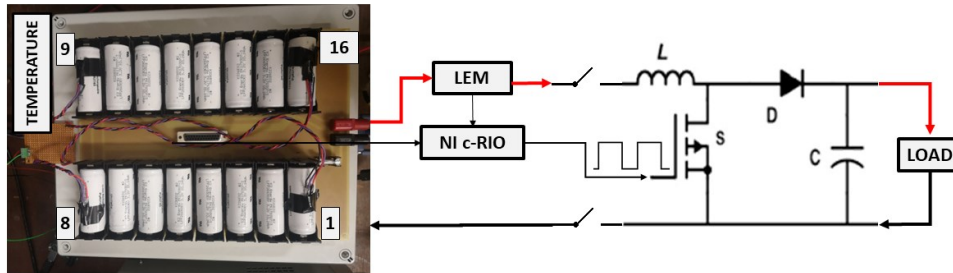
**Figure 5.2:** Sinewave component for the multi-sine signal.

as they can be obtained within the same discharge cycle. On the other hand, it worsens the time resolution of the high-frequency measurements and increases the overall peak-to-peak amplitude of the perturbation.

It has to be noted that dedicated instruments are required to perform the EIS since high-frequency measurements must be acquired with sufficient accuracy, and complex computations are required to obtain the impedance value. As a result, EIS was previously limited to offline characterisation in research laboratories until recent studies explored the feasibility of an in-situ EIS evaluation of the cell impedance, exploiting different perturbation sources or perturbation types [88], [89], [90]. Indeed, it has been successfully proven that the AC input can be injected into the battery through a DC/DC converter [91], [92], [93]. This prevents any interruption of normal operations and allows for online impedance evaluation. Employing this setup allows the AC perturbations  $I_{AC,k}$  to be superimposed on the DC operations of the battery by appropriately controlling the switching devices.

## 5.2 EIS Dedicated Workbench

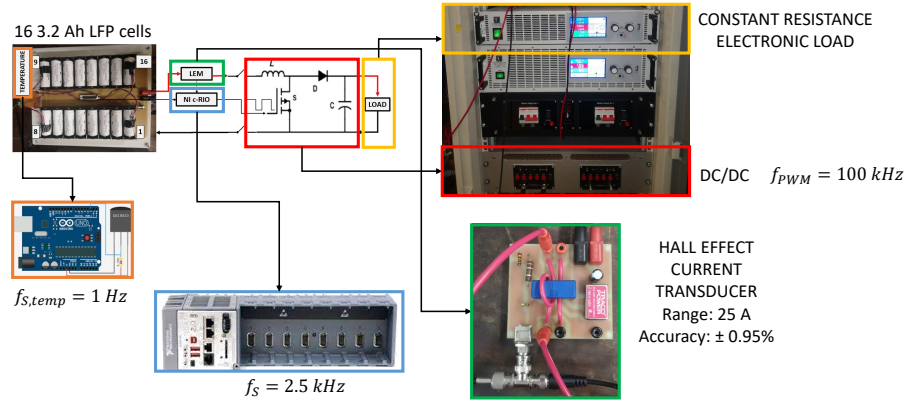
In this dissertation, the DC/DC converter strategy was chosen to perform an in-situ EIS. The cell was subjected to several discharge cycles and different DC current rates, superimposing a multi-sine signal to obtain the impedance values at different frequencies. Working in the frequency domain with the FFT requires sampling data at a much higher frequency than the workbench previously designed in Section 4.2.1. Therefore, a dedicated workbench was designed. The schematic overview of the workbench is shown in Fig. 5.3 and detailed in Fig. 5.4



**Figure 5.3:** Schematic overview of the proposed workbench for EIS dataset.

The unit under test was a module composed of sixteen K226650E02 3.2 Ah 26650 Lithium iron phosphate (LFP) cells, from the K2 Energy manufacturer, connected in series. This allows the acquisition of data from each cell at the same time, with proper precautions. The available cells were not brand new but were subjected to a limited and equal number of discharge cycles in the controlled laboratory environment.

The module is connected in series to a DC/DC boost converter. A Labview code im-



**Figure 5.4:** Detailed EIS workbench.

plements a closed-loop PI controller for the 100 kHz PWM signal, which is forwarded to the converter by a NI 9401 digital module interfaced with a National Instruments (NI) compactRIO 9035.

The multi-sine signal defined in (5.1) contains the AC excitation input. Five sinusoidal perturbations, i.e.,  $f_k$  at each decade from 10 mHz ( $k=1$ ) to 100 Hz ( $k=5$ ), with tones amplitude  $I_{AC_k}$  equal to 5 mA, composed the signal  $I(t)$  to be applied to the module. An Elektro-Automatik EL9000B electronic load is connected on the load side, operating in constant resistance mode.

The NI 9205 voltage acquisition module receives the voltage data. The voltage is sensed by the board on which the cells are mounted, which is also equipped with isolation amplifiers. On the other hand, the current is sensed by a 25 A LEM LA 25-P Hall-effect current transducer, with  $\pm 0.95\%$  accuracy at nominal current and  $25^\circ\text{C}$ , and then acquired by a NI 9215 acquisition module. The voltage and current signals are sampled at a frequency  $f_s$  of 2.5 kHz, an integer multiple of all frequency components. The signal acquisition synchronized with the waveform generation ensures no spectral leakage occurs when acquiring a full signal period. At the same time, the

chosen sampling frequency is a sub-multiple of the PWM 100 kHz frequency so that the aliasing from the switching frequency affects only the DC component and not the measurements at the AC perturbation frequencies [31]. Finally, a C-coded Arduino Uno acquires at  $f_{s,temp} = 1$  Hz the sensed temperature from four DS18B20 temperature sensors placed on cells 1, 8, 9, 16 in Fig. 5.3, with a resolution of  $0.0625$  °C. Sampling temperature data at a faster rate is not necessary due to the long time constant in temperature changes.

## 5.3 Impedance Measurements

### 5.3.1 Dataset

The current range from 2 A to 4 A was considered, as in Section 4.2, with 0.5 A steps. The module was discharged for a total of five current cycles, and repeated to obtain three rounds of acquisition, named  $R_1, R_2, R_3$ . For EIS technique, the maximum discharge current and the sampling frequency are crucial for limiting the  $\Delta SoC$  in a time step, as depicted in (2.8). In the worst-case scenario, the highest current rate, i.e., 4 A, and the longest measurement time, i.e., 100 s for the 10 mHz frequency, the change in SoC during measurements is limited to 3.47%, from (2.8).

Proper CC-CV charge and 30 minutes of rest period were applied, followed by the discharge cycle until the lower cut-off voltage of 2.5 V. A total of three sets composed of five discharges were then obtained.

### 5.3.2 Voltage Drift Correction

As depicted in Section 2.2.1, the voltage drift must be corrected before processing the data with the FFT to preserve the waveform periodicity. In Fig. 5.5a, a 100 s window (a full 10 mHz period) shows that the voltage drift is noticeable due to the long acquisition time and highly affects the voltage spectrum in the frequency domain, as shown in Fig. 5.6a. It has to be noted that the drift in each window is almost linear, and different techniques can be employed with different complexities to compensate for it [31], [94], [30].

In this case, a computationally simple trend linear fitting was used. Firstly, the signal was averaged and downsampled over 10 s windows. This allowed to retain only the fundamental sine wave, filtering out perturbations at higher frequencies. Then, the slope of the straight line joining the window period's first point with the next period's first point was computed, i.e., obtaining the voltage drift slope. The obtained line was then subtracted from the original signal to compensate for the drift. The effects of the compensation technique are shown in Fig. 5.5b, where the waveform periodicity was recovered. This was repeated for each period of each full discharge cycle.

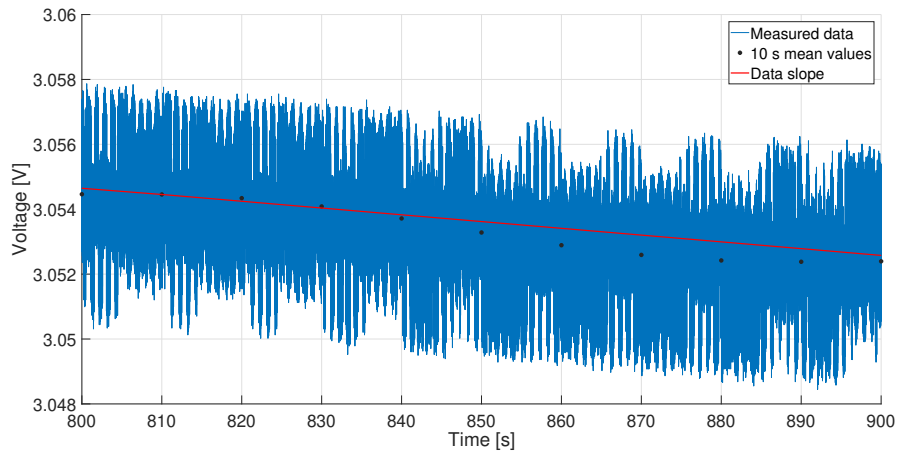
The slope of the voltage drift was assumed to be linear during the period, therefore this technique can not be applied in the highly non-linear regions of the voltage characteristic, visible in Fig. 2.8 corresponding to SoC values near 100% and 0%. As a consequence, the first and the last 100 s periods were discarded from the later computations. Therefore, it has to be noted that the first impedance value was obtained at time  $t = 200$  s. If the slowest  $f_k$  is higher than the 10 mHz employed here, i.e., shorter windows, multiple periods should be discarded to ensure being out of the non-linear region. Moreover, the last incomplete period was discarded, e.g., if the cell reached the cut-off voltage before completing a 100 s period.

Finally, the FFT was performed in the MATLAB environment on each acquired period of each discharge cycle, obtaining the voltage spectra. In Fig. 5.6b, the FFT spectrum after the drift compensation is reported for the period in Fig. 5.5. Observing Fig. 5.6, it is clear the quality improvement: in Fig. 5.6a, the compromised waveform periodicity cause other frequencies components to appear, which are then removed after the drift compensation, obtaining much more distinguishable lower frequencies components. Indeed, after the compensation, the perturbation frequencies appear more than one decade larger than the neighbouring frequencies.

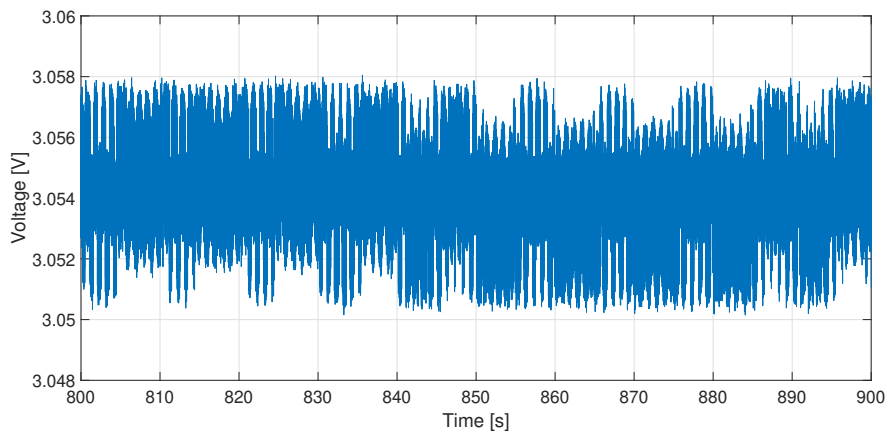
### 5.3.3 SVR Training and Test

During each measurement period, also the current data withstand FFT analysis without any pre-processing, as there was no drift. Finally, the impedance values were obtained by applying (2.7).

Given  $T$  the time duration in seconds of a discharge cycle for a total of  $T \times f_s$



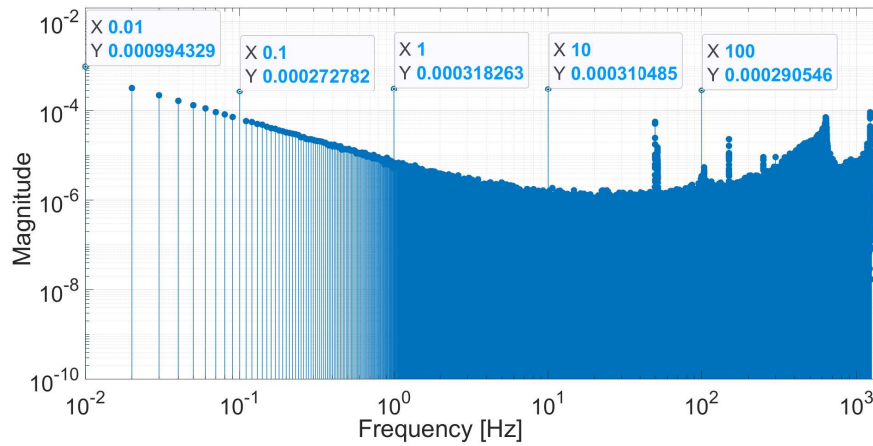
(a)



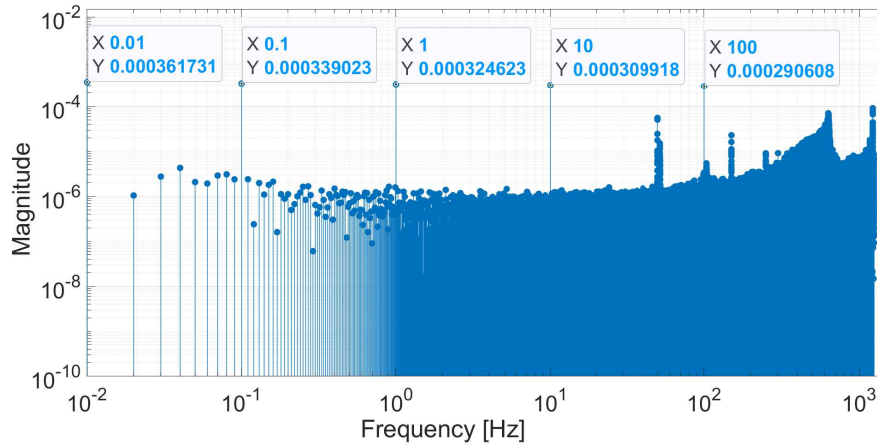
(b)

**Figure 5.5:** Voltage data comparison before (a) and after (c) the drift correction a single acquisition period.





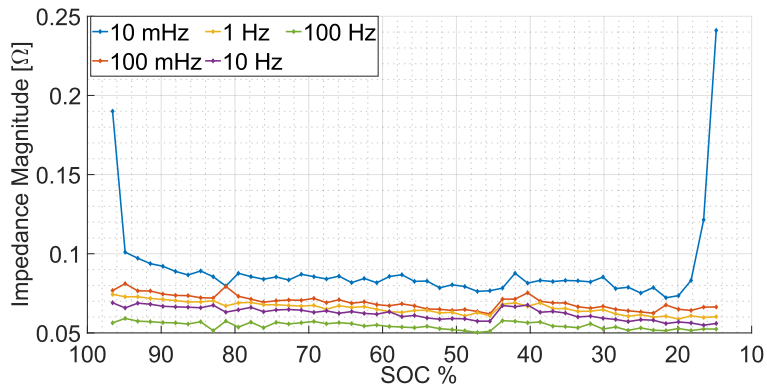
(a)



(b)

**Figure 5.6:** Voltage FFT spectrum corresponding to a single acquisition period before (a) and after (b) the drift compensation.

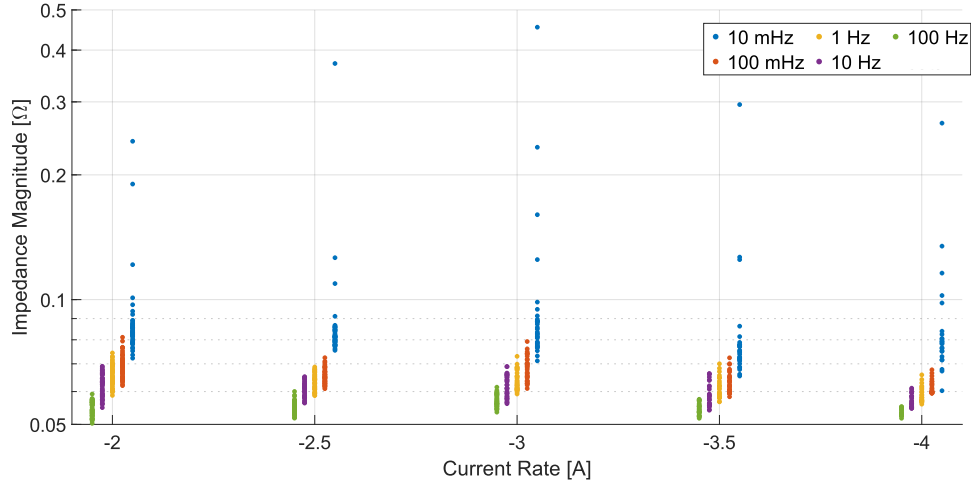
data points, the obtained dataset was composed of  $T \times f_1 - d$  measurement windows  $n_w$ , where  $f_1$  is the slowest frequency and  $d$  is the number of discarded periods, in this case  $d=3$  (the first, the last incomplete, and the last periods, as aforementioned). For the fifteen discharge cycles of the three acquisition rounds,  $n_w$  impedance values were obtained for the five frequencies. In Fig. 5.7, the magnitude of the impedance at the five frequencies is plotted against the SoC while discharging at 2 A to clarify the impedance behaviour during the discharge cycle. The trend is similar for each current rate.



**Figure 5.7:** Magnitude of the impedance for cell number 1 discharged at 2 A as a function of the SoC.

In Fig. 5.8, the impedance magnitudes are plotted against the current rate but a linear impedance-current dependency can not be found.

Finally, the mean current, voltage and temperature in the 100 s period were computed to eliminate oscillations in the data due to the injected perturbations, possibly affecting the training phase. At the end of pre-processing, each dataset  $R_i$  comprised the input vector in Table 5.1 for each cell and current rate, where the impedance magnitude and phase were computed as (5.2) and (5.3), respectively, and  $Re\{Z\}$  and



**Figure 5.8:** Magnitude of the impedance for cell number 1 at different current rates.

$Re\{Z\}$  and  $Im\{Z\}$  are the real and imaginary parts of the complex impedance.

$$|Z| = \sqrt{(Re\{Z\})^2 + (Im\{Z\})^2} \quad (5.2)$$

$$\angle Z = \arctan \frac{Im\{Z\}}{Re\{Z\}} \quad (5.3)$$

TABLE 5.1  
INPUT VECTORS FOR EACH CELL AND EACH CURRENT RATE.

$x_1$	$I_{mean}$	$V_{mean}$	$T_{mean}$	$ Z_{10mHz} $	$\angle Z_{10mHz}$	...	$ Z_{100Hz} $	$\angle Z_{100Hz}$
...	...	...	...	...	...	...	...	...
$x_{n_w}$	$I_{mean}$	$V_{mean}$	$T_{mean}$	$ Z_{10mHz} $	$\angle Z_{10mHz}$	...	$ Z_{100Hz} $	$\angle Z_{100Hz}$

The aim of this work was to evaluate the input features that resulted in the optimum SoC estimation, identifying the most important frequencies. As can be seen, training an SVR model for each of the available combinations ( $2^{13}$ ) was not feasible. Therefore, choices were made. The current, voltage and temperature information

were included in each training set, as these were the input features typically employed in the literature, and the magnitude and phase at the same frequency were treated as a single feature. This led to only six combinations of features, i.e., the combinations proposed in Table 5.2.

The SVR reference target SoC was computed by (2.1). It is worth noting that in a series connection, the CC technique must face some problems. The sixteen cells in the module could be different within the manufacturer's tolerances and despite having the same working history, they may have aged slightly differently. Moreover, cell temperatures may differ during the discharge phase, causing a non-synchronous cut-off. These differences cannot be accounted for by the CC method, therefore the cells may not reach the effective full charge and full discharge state at the same time [95]. In this work, the maximum deviation in the SoC of different cells has been estimated to be 1.8%. For a preliminary investigation, in this work, only cell number 1 of sixteen was considered for training and testing the SVR, granting consistency in the data. Out of the three rounds of acquisitions, two were assigned to the training set and the last one was used for the testing phase.

The SVR was trained with the RBF kernel and 10-fold cross-validation, which was appropriate with the constant current approach in Chapter 4. Moreover, a linear kernel works poorly since, from Fig. 5.7, linear fitting of the magnitude as a function of the SoC leads to inaccurate results. For a benchmark, an SVR was trained with only current, voltage and temperature and then compared with the SVR models trained with different impedance information. A post-processing stage was applied to the estimated SoC, limiting physically unfeasible values [50] and constraining the SVR output by the maximum and minimum removable charge  $\Delta Q$  in a time step  $\Delta t$ .

$$\Delta Q = I\Delta t \quad (5.4)$$

A 3% tolerance on the current measurement  $I$  in (5.4) was used, to avoid constraining the SoC estimate to follow the CC result. Indeed, ideally setting 0% tolerance and the initial SoC to the actual starting SoC, it should be possible to obtain an unrealistic 0% RMSE. This was obviously avoided to ensure realistic results to be applied to real systems.

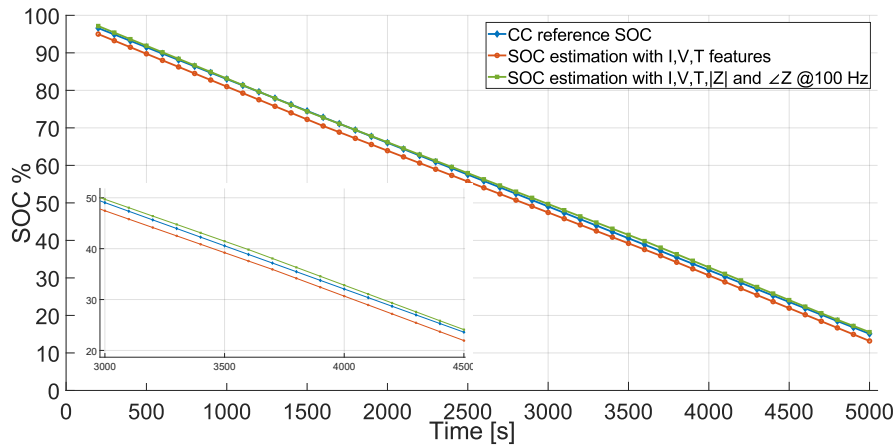
TABLE 5.2  
DIFFERENT INPUT FEATURES FOR SOC ESTIMATION ON THE SAME CELL

Input Features	RMSE
I,V,T	1.34%
I,V,T, Z ,∠Z @ 10 mHz	1.25%
I,V,T, Z ,∠Z @ 100 mHz	1.06%
I,V,T, Z ,∠Z @ 1 Hz	1.19%
I,V,T, Z ,∠Z @ 10 Hz	1.25%
I,V,T, Z ,∠Z @ 100 Hz	1.09%

#### 5.3.4 Results and Discussion

The resulting SoC estimation values were evaluated with the RMSE metric in (1.4). The comparison between the SVR models trained with the magnitude and phase of the impedance and the reference SVR model is listed in Table 5.2. The tests were performed with the five discharges of the left-apart acquisition round. Therefore, the reported RMSE values are the averaged metrics.

As can be seen in Table 5.2, the reference SVR model without impedance information resulted in a mean RMSE after the post-processing stage of 1.34%. All the five SVMs trained with the impedance information in the feature set performed better than the reference model. Among these feature combinations, the lowest RMSEs equal to 1.06% and 1.09% were obtained with the 100 mHz and the 100 Hz impedance information, respectively. The estimated SoC is shown in Fig. 5.9 for the SVR model with the 100 Hz impedance features. From Fig. 5.9, it can be seen that the discharge ended above 10% SoC since (2.1) does not consider battery ageing which affects  $Q_{tot}$ , and the last periods were discarded as aforementioned. Moreover, it is important to mention that only a limited number of possible combinations of input features were considered during the analysis. Therefore, it is necessary to investigate more feature combinations, including the feasibility of using impedance at more frequencies in the training set at the same time. Among the tested combinations, the 100 Hz impedance allows for shortening the measurement time compared to the 100 mHz impedance.



**Figure 5.9:** State of Charge estimated by SVR trained with (green line) and without (red line) the impedance information at 100 Hz compared with the reference CC SoC.

Therefore, for later analyses, the 100 Hz model was used for reference.

Indeed, some more testing was performed using the remaining module cells (i.e., cells 2 to 16) not used in the training phase. The model with the 100 Hz information resulted in a mean RMSE of 1.23% over the cells number 8, 9, and 16, equipped with the temperature sensor. The error slightly increased, compared to the same-cell testing, but it is justified by the different cell's behaviour and impedance.

Compared with the literature, the extended set of input features performs similarly when compared to the 1.8% RMSE of an SVR trained and tested on road drive cycles in [38], i.e., a different approach than constant current discharges data. When considering the testing on different cells, in [40] a DCNN was trained on different cell data with the transfer learning technique. Once tested on different cells than those used in the training, it resulted in a 2.42%, slightly worse than the proposed method with extended input features, in which the cells belonged to the same manufacturer.

The hardware implementation of the proposed SVR trained with enhanced input features will be investigated in the next developments. However, the FPGA imple-

mentation of a trained SVR for SoC estimation has been already detailed in previous Chapter 4 and Chapter 3. The SVR resulted in low resource usage, so changing the input vector may have limited effects on the hardware implementation but this will still be investigated. In contrast, when implementing the impedance feature extraction, the high computational complexity of the FFT algorithm must be considered. Nevertheless, alternative methods such as the Goertzel algorithm are usually employed to reduce the computational complexity of these procedures.





## Chapter 6

# Conclusions

The Lithium-ion batteries are complex electrochemical systems that must be monitored for secure user operation. Indeed, BMSs equipped with proper management algorithms are crucial to avoid damages and extend battery lifespan. The keystone of each algorithm is the SoC indicator, which cannot be directly computed with straightforward formulas but must be accurately estimated. Most approaches discussed in the literature only evaluate their algorithms in PC-based systems within laboratory environments. While this is crucial for a preliminary evaluation, real-environment BMSs must be able to monitor several cells inside a battery pack at the same time. Therefore, the designer must guarantee that the algorithms fit into the BMS platform (a microcontroller or FPGA-based system) and can execute quickly enough to achieve a real-time SoC estimation properly. Unfortunately, these considerations are typically not investigated in PC-based approaches.

Some approaches to the SoC estimation have been investigated in this dissertation. Each discussed approach has been either implemented or designed with future implementation in mind, making choices to reduce the computational complexity. Firstly, the state of the art has been investigated, showing that the most accurate approaches rely on extensive cell characterisation to obtain ECMs. These model-based approaches can be further improved by using concurrent algorithms that introduce a feedback loop. However, in most cases, the user cannot go through this

time-consuming process. Despite the high estimation accuracy, it has been proven in [B1] and in Chapter 3 that implementing such architectures requires large computational resources. Indeed, when implemented on a Xilinx Artix-7 FPGA, the hybrid approach that combines an ECM and the CC formula resulted in occupying about 23% of the available area. This can be acceptable if a small number of cells are being monitored or a good pipelined design is implemented. Nevertheless, it is still not suitable for low-resource (and often low-cost) platforms.

The DDM approaches have been then investigated to obtain a more suitable technique for embedded systems [B2]. Among several ML algorithms, the SVR has been selected as it requires less training data but still achieves comparable results to the most popular NNs. Moreover, in the literature, the SVR applied to the SoC estimation field has never been investigated for the implementation on embedded systems. In this dissertation, the ACO was used instead of the traditional optimization algorithms embedded in the MATLAB environment, with the ACO resulting in better SVR modelling. The SVR model's inferring function was computationally simplified leading to an easier implementation. This was obtained by employing a Linear Kernel to map the input training features to the expected target output (i.e., the reference SoC computed through the CC, which is suitable for laboratory environments). If properly manipulated, the Linear Kernel inferring function can be implemented efficiently with only a scalar product and a sum, thus significantly limiting the required resources. This was made possible by adding an input feature - the SoC at the previous time step - that linearized the cell behaviour. The dataset for training and testing was available online to speed up the process. Finally, the VHDL code for the SVR model was implemented on a Xilinx Artix-7 FPGA, along with a communication protocol for testing purposes. The SoC estimation resulted in a RMSE of 1.4% and MAE of 1.2% when tested on a US06 drive cycle test set. When compared with the literature, it resulted in better error metrics, and FPGA resource utilisation was only limited to 1.4% of the available LUTs. The occupied area has been improved by a factor of 16 compared with the previously investigated hybrid approach in which 23% was used.

Investigating the state of the art, it is clear that most of the work focuses on EVs

due to the large market share. Therefore, most approaches are trained and tested on application-specific data, i.e. the simulated EV data called drive cycles, but it is not clear whether they can be applied to cells used in different scenarios or not. In this dissertation, a more general approach has been studied, in which only constant current profiles were used for the training phase, and the obtained model was tested on application-specific data [B3], [B5]. A dedicated workbench has been designed to gather data from a Panasonic NCR18650 battery cell during operations. Constant current discharges can be easily performed with any instrument or with simple load resistors, but acquiring dynamic drive cycle profiles for the testing phase was not straightforward. Indeed, the drive cycles are defined as speed setpoints that must be converted into current setpoints to be applied to the battery cell. This conversion is vehicle and battery dependent, therefore a vehicle simulator was coded in MATLAB to account for EV technical specifications. Different SVR models have been trained with constant current cell data (i.e., current, voltage, temperature) and the RBF Kernel to deal with the batteries' non-linear behaviour. The acquired full current range was 2 A to 4 A, but it has been also split into two subsets to validate the approach on a total of three different scenarios (2A-4A range, 2A-3A range and 3A-4A range). The obtained SVR models have been tested with five test cycles representing random dynamic cycles, one US06 drive cycle profile and one dynamic profile from a battery-powered drill. For each of these scenarios, a post-processing stage was implemented to further improve the SoC estimation, and the resulting averaged RMSE was about 1% on these application-specific tests, with a minimum RMSE of 0.77%, despite the SVR models being trained on application-independent data. The resulting error metrics were consistent over different current rates, proving the generalization capabilities of the proposed approach, and it could definitely be extended to datasets including a wider current range.

Finally, with the same purposes, a more advanced set of input features was investigated to improve this general approach. Along with the constant current data, the AC battery impedance was studied at different frequencies since the cell's impedance is one of the most important cell state indicators. The impedance spectrum can be obtained by employing the EIS. In this dissertation, a multi-sine signal composed

of sine waves at different frequencies (i.e., five frequencies logarithmically spaced) is superimposed to the DC current. Injecting the perturbations during the cell discharge through a DC/DC converter allowed the online evaluation of the impedance without stopping cell operations. The workbench for applying the perturbations and acquiring data has been designed following some crucial rules of thumb to allow effective online EIS. The voltage and current in the time domain must be Fast-Fourier transformed in the frequency domain to obtain the impedance value. However, before processing data, the voltage drift resulting from the cell discharge must be corrected by introducing a drift compensation pre-processing stage. Finally, the FFT has been applied and the obtained five impedance values have been introduced in the training input vector as magnitude and phase. Three datasets were acquired comprising five different current rates, using the first two for training and the latter for testing. A reference SVR model has been trained with only current, voltage and temperature but each SVR model that included the impedance in the input vector resulted in lower RMSE when tested [B4]. Finally, the 100 Hz impedance is a trade-off between SoC estimation accuracy and the implementation complexity. Indeed, faster frequencies allow for shorter measurement time, lower voltage drift, and lower SoC change during measurements, improving the whole architecture. Therefore, this approach could be further investigated and improved, and finally implemented on BMSs to operate the EIS during actual battery operations.

## Appendix A

# Vehicle Simulator

Given that drive cycle profiles are the most commonly used reference tests for battery simulations, it is crucial to use comparable test sets to achieve fair comparisons with the literature. However, drive cycle profiles are not available in terms of current consumption but they are described in terms of speed as a function of time, as detailed in Section 1.4.3.

Therefore, a simulator for vehicle consumption computation has been designed. The simulator can derive the required current profile for a moving Electric Vehicle, according to the selected drive cycle and vehicle characteristics. For this purpose, a 2012 Tesla Model S 85 has been chosen due to the online largely available technical specifications [96]. Moreover, Tesla's battery pack is composed of 18650 size factor cells (Fig. A.1), in particular 7104 Panasonic NCR18650 (16 modules, each one 6s74p). Therefore, the simulated battery cells should reflect the same behaviour as the 18650 battery cells used in this dissertation.

### A.1 Required Force

Drive cycles are standardized and they are expressed as time-speed setpoints sequences, as shown in Fig. 1.3. Firstly, obtaining the required acceleration  $a$  [ $m/s^2$ ]



**Figure A.1:** Tesla Model S 85 kWh battery pack.

for each time step is straightforward by employing (A.1).

$$a = \frac{\Delta v}{\Delta t} \quad (\text{A.1})$$

where  $\Delta v = v_t - v_{t-1}$  is the speed change in each time step,  $t$  is the time index, and  $\Delta t = t - (t - 1)$  is the time step. To accelerate the vehicle of the desired  $a$ , a specific tractive force  $F_t$  is required. This force can be derived from the equation of motion when considering resistance forces acting on a vehicle of total mass  $M$  [kg] [97], [98], [99], [100]. The resulting equation is (A.2) and a schematic of the forces acting on the vehicle is shown in Fig. A.2.

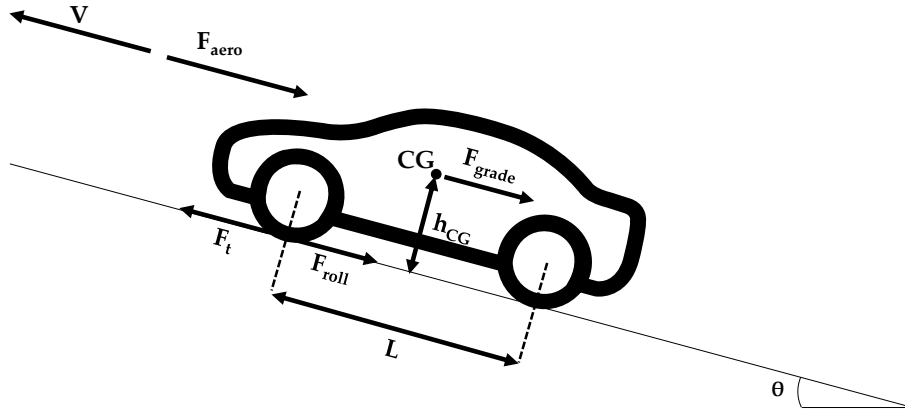
$$\delta \cdot M \cdot a = F_t - F_{aero} - F_{roll} - F_{grade} \quad (\text{A.2})$$

$$F_{aero} = \frac{1}{2} \cdot \rho \cdot A \cdot C_d \cdot (V + V_{wind})^2 \quad (\text{A.3})$$

$$F_{grade} = M \cdot g \cdot \sin(\theta) \quad (\text{A.4})$$

$$F_{roll} = P^\alpha \cdot W^\beta \cdot (a + bV + cV^2) \quad (\text{A.5})$$

The inertial equivalent mass must be considered, especially when dealing with heavy electric vehicles, hence, a rotational inertia coefficient  $\delta$  is introduced. The



**Figure A.2:** Schematic representation of the forces acting on a moving vehicle.

$\delta$  factor can be estimated using an empirical relationship (A.6) to deal with unknown angular moments of all the rotating components, reported here but detailed in [98], [101].

$$\delta = 1 + 0.04 + 0.0025i_g^2i_0^2 \quad (\text{A.6})$$

where  $i_g$  is the gear ratio and  $i_0$  is the gear ratio of the final drive.

The aerodynamic drag force  $F_{aero}$  (A.3) is a function of vehicle frontal area  $A$  [ $m^2$ ], its drag coefficient  $C_d$  [unitless], and its speed  $V$  [ $m/s$ ] and wind speed  $V_{wind}$  [ $m/s$ ]. This vehicle simulator also considers the air density  $\rho$  [ $kg/m^3$ ] modelled as a function of the temperature and altitude [102] to deal with different driving environmental conditions.

The rolling resistance force  $F_{roll}$  (A.5) acting on the wheels is affected by tyres pressure  $P$  [ $kPa$ ], normal force  $W$  [ $N$ ] and vehicle speed  $V$  [ $km/h$ ], and the relation coefficients  $\alpha, \beta, a, b, c$  has been modelled in SAE J2452 test procedure [103], [104]. Since its equation coefficients are tyre dependent, some typical values can be found in the literature [105]. The road grade  $\theta$  causes a force  $F_{grade}$  acting on the vehicle (A.4)

and affecting the normal forces  $W_f$ ,  $W_r$  applied on front and rear axles respectively [98], [106], formulated as (A.7).

$$\begin{aligned} W_f &= \frac{-F_{aero}h_{aero} - Mah_{CG} - Mgh_{CG} \sin(\theta) + Mgl_r \cos(\theta)}{l_f + l_r} \\ W_r &= \frac{F_{aero}h_{aero} + Mah_{CG} + Mgh_{CG} \sin(\theta) + Mgl_f \cos(\theta)}{l_f + l_r} \end{aligned} \quad (\text{A.7})$$

where  $h_{aero}$  is the height of aerodynamic force application point, and it is assumed the same as the centre of gravity height  $h_{CG}$  [98]. In this case  $l_f$  and  $l_r$  are the distances from the front and rear axle to the centre of gravity, respectively, as they sum to the vehicle wheelbase  $L$ .

## A.2 Adherence Coefficient

For stable vehicle control, the computed tractive force  $F_t$  should not exceed the maximum tractive force  $F_{t_{max}}$  permitted by the tyre-road adherence coefficient  $\mu$  and the normal force acting on the driven axle (A.8) [98].

$$F_{t_{max}} = \mu(s)W_{f/r} \quad (\text{A.8})$$

The adherence coefficient  $\mu$  highly depends on the slipping of the driven wheels (namely slip ratio  $s$ ), which is defined as (A.9) [106].

$$\begin{cases} s = 1 - \frac{V}{r_{eff}\omega} & , \text{ during driving } (F_t \geq 0) \\ s = \frac{r_{eff}\omega}{V} - 1 & , \text{ during braking } (F_t < 0) \end{cases} \quad (\text{A.9})$$

where  $V$  is the translational speed of the tyre centre,  $\omega$  is the tyre rotational speed and  $r_{eff}$  is the effective radius of the rolling tyre, computed as (A.10) [106].

$$r_{eff} = r \times \frac{\sin\left(\cos\left(\frac{r - W_{f/r}/k}{r}\right)^{-1}\right)}{\cos\left(\frac{r - W_{f/r}/k}{r}\right)^{-1}} \quad (\text{A.10})$$

where  $k[kg/m]$  is the tyre stiffness for which a simplified formulation is given in [104]. Since  $\mu$  is a function of the slip ratio, a simplified formulation is provided in [107].



In [108], a simplified friction model has been employed in which the adherence coefficient is described as a function of the slip ratio: it grows linearly up to the peak value  $\mu_{peak}$ , reached in correspondence of a critical slip ratio  $s_{crit}$ , then above the critical value,  $\mu$  is constantly considered equal to a value  $\mu_{dyn}$ . This simplified formulation is also similar to the “tanh model” [109]. In literature, some typical values for  $\mu_{peak}$ , and corresponding critical slip ratio  $s_{crit}$ , are presented [98], [109], [110]. The maximum allowed tractive force can be then computed as (A.8) substituting  $\mu(s)$  with  $\mu_{peak}$ . For normal driving, such as standard drive cycles considered in this dissertation, the slip ratio has been assumed to stay within  $s_{crit}$ . Therefore, the slip ratio can be computed by the simple linear relationship in (A.11), and the actual wheel speed  $\omega$  can be derived from (A.9).

$$s = \frac{F_t}{W_{f/r}} \times \frac{s_{crit}}{\mu_{peak}} \quad (\text{A.11})$$

In addition, wheel speed affects motor shaft speed  $\omega_m$ , as in (A.12) [111], which may be limited by motor specifications, hence limiting actual vehicle speed. Actual vehicle speed can be derived from (A.12) and (A.9), employing the limited  $\omega_m$  [rpm].

$$\omega_m = \omega_{wheel} \times G \times \frac{60}{2\pi} \quad (\text{A.12})$$

where  $G$  represents the product of gear ratio of the final drive by the gear ratio of the transmission, called *gear ratio* from now on.

### A.3 Required Torque

Two distinct scenarios must be considered to obtain battery power from the tractive force value.

#### Accelerating

The limited tractive force can be directly converted into motor torque  $T_m$  through the wheel radius  $r$  and gear ratio  $G$  [111] with (A.13)

$$T_m = \frac{F_t \times r}{\eta_G \times G} \quad (\text{A.13})$$

In this equation, the transmission efficiency  $\eta_G$  cannot be neglected. In the case of a Tesla Model S 85, the vehicle is equipped with a 9.73:1 single-gear transaxle, similar to Nissan Leaf transmission, for which in similar papers a 97% efficiency has been assumed [112], [113], [114]. The resulting motor torque  $T_m$  can be limited by motor specifications, depending on the motor operating point defined by the shaft speed  $\omega_m$  obtained in (A.12) [115], [116], [117].

### Braking

Further considerations must be examined when the vehicle brakes due to EV's distinctive regenerative braking. EV motors can recover part of the available kinetic energy to partially recharge the battery pack, hence improving the vehicle range. The vehicle manufacturers program the motor controller to obtain the desired regenerative braking under different driving conditions (e.g., high/low speed, high/low battery State of Charge, etc.). As a reference for a preliminary simulation, Tesla's battery can accomplish an absolute maximum regenerated power  $P_{\text{bat,IN}}$  of 60 kW. Therefore, a maximum braking force can be computed by arranging the equations (A.14) depicted in [111].

$$\begin{aligned}
 F_t &= \frac{T_w}{r} & (A.14) \\
 T_w &= \frac{T_{m,\text{brk}}}{\eta_G} \times G \\
 T_{m,\text{brk}} &= \frac{P_{m,\text{brk}}}{\omega_m \times \frac{2\pi}{60}} \\
 P_{m,\text{brk}} &= \frac{P_{\text{bat\_in}}}{\eta_E} \\
 F_{\text{brk,max}_{\text{wheels}}} &= \frac{P_{\text{bat\_in,max}}}{\eta_E} \times \frac{1}{\omega_m \times \frac{2\pi}{60}} \times \frac{G}{\eta_G} \times \frac{1}{r}
 \end{aligned}$$

In (A.14), motor brake power  $P_{m,\text{brk}}$  stored into the battery is affected by electric efficiency  $\eta_E$ , which comprises motor, inverter, and battery efficiencies. This maximum battery-limited braking force is defined in optimum battery conditions: the manufacturer motor controller firmware adapts this value according to the SoC level

and ambient temperature. In [118], a characterisation of coast-down decelerations for different Tesla vehicles has been performed, reporting three deceleration phases for the Tesla Model S in Standard and Low Mode regeneration. Here, Standard Mode is considered. The first phase of increasing deceleration lasts 0.68 s at  $-0.23$  g/s. Since the standard drive cycle time step is 1 second, this phase has been neglected during simulation. The second phase is characterised by steady-state deceleration  $a_{\max}$  at an average of  $-0.19$  g. This has been reported for different speeds, and a reduction in deceleration intensity was found to start under a threshold  $V_{\text{th1}} = 10.34$  mph (i.e., 16.64 km/h). Therefore, in this simulator, deceleration under this threshold speed has been computed as a linearly decreasing function since this third phase has not been analyzed in detail in [118]. In a first approximation, these values can be acceptable since obtaining precise values from each manufacturer is unfeasible. Coast-down deceleration limit leads to the maximum motor braking force definition in (A.15).

$$F_{\text{brk,coastdown}} = \begin{cases} a_{\max} \times g \times \delta M & \text{for } V \geq V_{\text{th1}} \\ a_{\max} \times \frac{V - V_{\text{th2}}}{V_{\text{th1}} - V_{\text{th2}}} \times g \times \delta M & \text{for } V_{\text{th2}} < V < V_{\text{th1}} \\ 0 & \text{for } V \leq V_{\text{th2}} \end{cases} \quad (\text{A.15})$$

A low-speed threshold  $V_{\text{th2}}$  for regenerative braking activation is usually adopted due to low available torque conditions [114], [112], here it is assumed to be 5 km/h. It has to be noted the coast-down test is performed by accelerating to the desired speed and then releasing the throttle pedal, thus  $a_{\max}$  comprises road forces, which are then subtracted when computing brake force at the motor. It amounts to this, if  $F_t$  (previously limited by road adherence) exceeds  $F_{\text{brk,coastdown}}$  then motor brake force at the wheels  $F_{\text{brk,mot@wh}}$  is limited by the lowest of  $F_{\text{brk,coastdown}}$  and  $F_{\text{brk,max_wheels}}$ , else it is limited only by  $F_{\text{brk,max_wheels}}$ . Limits on maximum braking force are used to compute electrical and mechanical brake force partitioning (A.16).

$$F_{\text{brk,mech}} = F_t - F_{\text{brk,mot@wh}} \quad (\text{A.16})$$

Brake pads effort  $F_{\text{brk,mech}}$  can be reduced to zero allowing the electric motor to sustain the desired deceleration as far as it does not exceeds limits. Electric motor

braking capability has been limited to comply with battery limits, therefore when the battery is almost completely charged, or the temperature is too low, regeneration is disabled. Electric motor braking force can then be converted into motor torque through wheel radius and gear ratio (A.17).

$$T_{m,\text{brk}} = -\eta_G \times \frac{F_{\text{brk,mot@wh}} \times r}{G} \quad (\text{A.17})$$

For simulation evaluation purposes, also the total recoverable energy  $E_{\text{avail}}$  has been computed with (A.18).

$$\begin{aligned} E_{\text{avail}} &= \Delta E_{\text{kin}} - E_{\text{loss}} \\ E_{\text{loss}} &= (F_{\text{aero}} + F_{\text{roll}} + F_{\text{grade}}) \times V \times \Delta t \end{aligned} \quad (\text{A.18})$$

where  $\Delta E_{\text{kin}}$  is the difference in kinetic energy while decelerating during a single time step, and  $E_{\text{loss}}$  represents energy losses due to road forces.

#### A.4 Battery Power

Actual acceleration for the considered time step can be derived from (A.2), where  $F_t$  is the actual limited tractive force during acceleration or braking. As a consequence, in case one of the described limits has been exceeded, the vehicle will not be able to fulfil the required speed for the specific time step. Finally, the total torque computed for acceleration and braking phases can be converted into motor power  $P_m$ , then into the required battery power (A.19) [111].

$$P_m = \begin{cases} T_m \times \frac{2\pi}{60} \times \frac{(\omega_{m,\text{prev}} + \omega_m)}{2} & \text{during driving} \\ P_{m,\text{brk}} = T_{m,\text{brk}} \times \frac{2\pi}{60} \times \frac{(\omega_{m,\text{prev}} + \omega_m)}{2} & \text{during braking} \end{cases} \quad (\text{A.19})$$

In (A.19), the motor is assumed to be linearly accelerating; hence, a mean value for the motor speed over the time step is assumed. The required battery power is affected by electric efficiency  $\eta_E$ . Motor efficiency for a Tesla Model S 85 induction motor has been assumed to be 93.4% at the maximum output power operating

point [119], even though a look-up table for speed-power efficiency heatmap could be introduced to enhance simulation performance or exploiting piece-wise approximation [112].

Tesla's inverter and battery efficiencies can roughly be estimated by literature mean efficiencies, as reported in [120], where inverter efficiency 96% and battery efficiency 95% are assumed, leading to 85.2% overall electrical efficiency  $\eta_E$ .

The total battery power must also account for auxiliary onboard loads, which can affect the total vehicle consumption. They are usually powered by a 12 V battery charged from the high-voltage battery through a DC/DC converter, whose efficiency  $\eta_{DC}$  has been assumed to be 95% [114]. In [114], the main EV auxiliaries and their corresponding approximated power consumption are listed. For the NEDC drive cycle, lights and auxiliary devices must be switched off, except for those required for testing and daytime operation of the vehicle [114]. Therefore, when the simulated drive cycle is the NEDC, only driving control, power steering, and energy management systems will be activated, resulting in a total of 700 W consumption.

Finally, the total battery power can be computed with (A.20) [111].

$$P_{\text{bat}} = \begin{cases} \frac{P_{\text{aux}}}{\eta_{DC}} + \frac{P_m}{\eta_E} & \text{during driving} \\ \frac{P_{\text{aux}}}{\eta_{DC}} + P_{m,\text{brk}} \times \eta_E & \text{during braking} \end{cases} \quad (\text{A.20})$$

## A.5 Current Consumption Profile

Once the battery pack specifications and the required power are known, computing the battery pack current is a straightforward calculation with (A.21).

$$I_{\text{pack}} = -\frac{P_{\text{bat}} [\text{W}]}{V_{\text{nom,pack}}} \quad (\text{A.21})$$

where  $V_{\text{nom,pack}}$  is the nominal battery pack cell voltage, usually available from manufacturer vehicle data or derived from the nominal cell voltage and series connected cells. The sign convention here is that negative current flows out of the battery.

The current flowing on each cell can be then derived by knowing the battery pack format factor (A.22).

$$I_{\text{cell}} = \frac{I_{\text{pack}}}{N_p} \quad (\text{A.22})$$

where  $N_p$  represents the total number of parallel cells in the battery pack.

Moreover, since battery power is highly dependent on battery internal resistance, which depends on SoC and battery environmental conditions, it is also possible to accurately model  $I_{\text{pack}}$  as a function of SoC, as depicted in [106], [111]. Current could be then expressed as (A.23)

$$I_{\text{pack}} = \frac{OCV - \sqrt{OCV^2 - 4R_{\text{int}}(-P_{\text{bat}})}}{2R_{\text{int}}} \quad (\text{A.23})$$

where  $OCV$  is the battery open-circuit voltage and  $R_{\text{int}}$  is the internal battery resistance, but they have to be previously characterised [121].

## A.6 Range Estimation

Once the battery and cell current have been obtained, the cycle profile can be used as a test for the SoC estimation algorithms. For the purposes of simulations, the designed vehicle simulator also included a SoC estimation formula based on the CC (2.2) to determine a theoretical autonomy range if the vehicle is driven according to the proposed drive cycle. In this simulator, all the battery pack cells are assumed to be equalized by the vehicle BMS, thus, each cell SoC should match the total battery pack SoC.

The total distance the vehicle has run during the drive cycle can be computed through the integration of the vehicle speed, averaged over the time step to take into account the linear acceleration. Therefore, by knowing the total distance and total charge consumption, it is possible to compute the theoretical maximum vehicle range undergoing the repetition of a specific drive cycle as (A.24)

$$\text{range} = \frac{100\%}{\Delta\text{SOC}} \times \text{distance} \quad (\text{A.24})$$

where  $\Delta\text{SoC}$  is the quantity of charge used during the cycle. In case the current profile has been evaluated through (A.23), equation (A.24) is no longer accurate since  $I_{\text{pack}}$  is referred only to a specific SoC. Hence,  $\Delta\text{SoC}$  derived from CC cannot be assumed

to be the same for each drive cycle repetition. In this case, the simulation must be looped until the resulting SoC equals zero.

At the end of the drive cycle, it is also possible to compute the vehicle efficiency in terms of percentage recovered energy (A.25).

$$E_{\text{recov}} = \frac{\sum \int_0^t P_{\text{bat,brk}}(t) dt}{\sum E_{\text{avail}}(t)} \quad (\text{A.25})$$

where  $P_{\text{bat,brk}}$  are the negative values of  $P_{\text{bat}}$ . The available energy during braking is computed with (A.18) for each time step, and the integral of the battery power during braking represents the total energy flowed into the battery in the same time step.

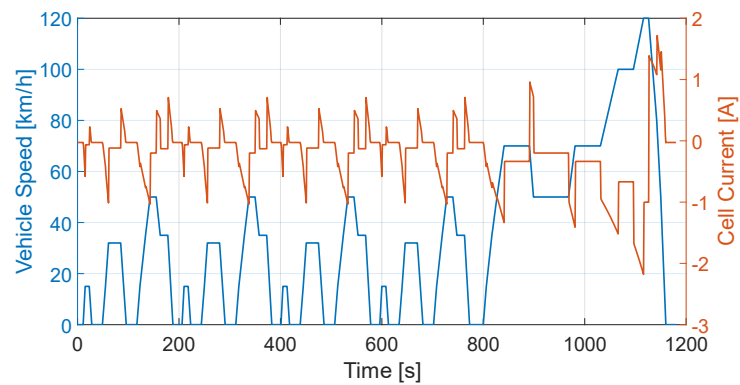
## A.7 Results

The Tesla Model S 85kWh NEDC ratings are available online [122] and reported in Table A.1. Therefore, the NEDC drive cycle was used as a reference to test the accuracy of the designed simulator. The vehicle was assumed to drive at ambient temperature at sea level and on a flat road surface.

TABLE A.1  
COMPARISON BETWEEN NEDC RATINGS AND SIMULATION RESULTS FOR A TESLA  
MODEL S 85KWH.

	<b>Energy Consumption</b> (kWh/100km)	<b>Range</b> (km)
NEDC	16.1	502
Simulator	17.1	488
Error	+6.2%	-2.79%

Results reported in Table A.1 show that an error lower than 3% is obtained in the estimated vehicle range, with 6.2% more battery consumption. It is worth noting that, even if several data are available reporting the Tesla Model S 85kWh technical specifications [96], the simulator must introduce different assumptions on the vehicle efficiencies and environmental conditions. Therefore, the results are optimum for the



**Figure A.3:** NEDC drive cycle speed setpoints (blue line) and the corresponding cell current (red line).

purposes of this simulator, which only aims to convert the desired speed setpoints into current setpoints.

The NEDC vehicle speed setpoints and the resulting current profile are shown in Fig. A.3.



## Appendix B

### List of Publications

The same author already published part of this work in the following papers:

- B1. M. Stighezza, V. Bianchi, and I. De Munari, "HDL Code Generation from SIMULINK Environment for Li-Ion Cells State of Charge and Parameter Estimation", *Lecture Notes in Elect. Eng.*, 2021, vol. 738, pp. 136 – 143, doi: 10.1007/978-3-030-66729-0\_16.
- B2. M. Stighezza, V. Bianchi, and I. De Munari, "FPGA Implementation of an Ant Colony Optimization Based SVM Algorithm for State of Charge Estimation in Li-Ion Batteries," *Energies*, vol. 14, no. 21, p. 7064, 2021, doi:10.3390/en14217064.
- B3. M. Stighezza, V. Bianchi, A. Toscani, and I. De Munari, "A flexible machine learning based framework for state of charge evaluation," in *2022 IEEE Int. Workshop on Metrology for Automotive (MetroAutomotive)*, 2022, pp. 111–115, doi: 10.1109/metroautomotive54295.2022.9855050.
- B4. M. Stighezza, R. Ferrero, V. Bianchi and I. De Munari, "Machine learning and impedance spectroscopy for battery state of charge evaluation," *2023 IEEE International Workshop on Metrology for Automotive (MetroAutomotive)*, Modena, Italy, 2023, pp. 24-29, doi: 10.1109/MetroAutomotive57488.2023.10219121.

- B5. V. Bianchi, M. Stighezza, A. Toscani, G. Chiorboli and I. De Munari, "An Improved Method Based on Support Vector Regression With Application Independent Training for State of Charge Estimation," in IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1-11, 2023, Art no. 2524811, doi: 10.1109/TIM.2023.3306816.

# Bibliography

- [1] IEA, “Global EV Outlook 2023,” 2023, <https://www.iea.org/reports/global-ev-outlook-2023>, Last accessed on 2023-10-03.
- [2] IMARC Group, “Lithium-ion Battery Market: Global Industry Trends, Share, Size, Growth, Opportunity and Forecast 2023-2028,” IMARC Group, Tech. Rep., Mar. 2023.
- [3] P. Nzereogu, A. Omah, F. Ezema, E. Iwuoha, and A. Nwanya, “Anode materials for lithium-ion batteries: A review,” *Applied Surface Science Advances*, vol. 9, p. 100233, 2022, doi: <https://doi.org/10.1016/j.apsadv.2022.100233>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2666523922000253>
- [4] Battery University, “Understanding Lithium-ion,” 2016, <https://batteryuniversity.com/article/understanding-lithium-ion>, Last accessed on 2023-10-16.
- [5] J. B. Goodenough, “How we made the Li-ion rechargeable battery,” *Nature Electronics*, vol. 1, p. 204, 2018, doi: <https://doi.org/10.1038/s41928-018-0048-6>. [Online]. Available: <https://www.nature.com/articles/s41928-018-0048-6>
- [6] H. Rouhi, E. Karola, R. Serna-Guerrero, and A. Santasalo-Aarnio, “Voltage behavior in lithium-ion batteries after electrochemical discharge and its implications on the safety of recycling processes,” *Journal of Energy*

- Storage*, vol. 35, p. 102323, 2021, doi: <https://doi.org/10.1016/j.est.2021.102323>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352152X21000864>
- [7] K. Liu, X. Hu, Z. Yang, Y. Xie, and S. Feng, "Lithium-ion battery charging management considering economic costs of electrical energy loss and battery degradation," *Energy Conversion and Management*, vol. 195, pp. 167–179, 2019, doi: <https://doi.org/10.1016/j.enconman.2019.04.065>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0196890419304996>
- [8] P. Keil and A. Jossen, "Charging protocols for lithium-ion batteries and their impact on cycle life—An experimental study with different 18650 high-power cells," *Journal of Energy Storage*, vol. 6, pp. 125–141, 2016, doi: <https://doi.org/10.1016/j.est.2016.02.005>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352152X16300147>
- [9] S. Yadav and S. Shukla, "Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification," in *2016 IEEE 6th International Conference on Advanced Computing (IACC)*, 2016, pp. 78–83, doi: 10.1109/IACC.2016.25.
- [10] R. Li, S. Xu, S. Li, Y. Zhou, K. Zhou, X. Liu, and J. Yao, "State of Charge Prediction Algorithm of Lithium-Ion Battery Based on PSO-SVR Cross Validation," *IEEE Access*, vol. 8, pp. 10 234–10 242, 2020, doi: 10.1109/ACCESS.2020.2964852.
- [11] D. N. T. How, M. A. Hannan, M. S. Hossain Lipu, and P. J. Ker, "State of Charge Estimation for Lithium-Ion Batteries Using Model-Based and Data-Driven Methods: A Review," *IEEE Access*, vol. 7, pp. 136 116–136 136, 2019, doi: 10.1109/ACCESS.2019.2942213.
- [12] I. B. Espedal, A. Jinasena, O. S. Burheim, and J. J. Lamb, "Current Trends for State-of-Charge (SoC) Estimation in Lithium-Ion Battery Electric

- Vehicles,” *Energies*, vol. 14, no. 11, 2021, doi: 10.3390/en14113284. [Online]. Available: <https://www.mdpi.com/1996-1073/14/11/3284>
- [13] C. Vidal, P. Malysz, P. Kollmeyer, and A. Emadi, “Machine Learning Applied to Electrified Vehicle Battery State of Charge and State of Health Estimation: State-of-the-Art,” *IEEE Access*, vol. 8, pp. 52 796–52 814, 2020, doi: 10.1109/ACCESS.2020.2980961.
- [14] M. O. Qays, Y. Buswig, M. L. Hossain, and A. Abu-Siada, “Recent progress and future trends on the state of charge estimation methods to improve battery-storage efficiency: A review,” *CSEE Journal of Power and Energy Systems*, vol. 8, no. 1, pp. 105–114, 2022, doi: 10.17775/CSEEJPES.2019.03060.
- [15] P. Shrivastava, T. K. Soon, M. Y. I. B. Idris, and S. Mekhilef, “Overview of model-based online state-of-charge estimation using Kalman filter family for lithium-ion batteries,” *Renewable and Sustainable Energy Reviews*, vol. 113, p. 109233, 2019, doi: <https://doi.org/10.1016/j.rser.2019.06.040>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1364032119304332>
- [16] Y. Wu and D. Zhou, “Simulation Research and Embedded Implementation of Model Based SOC Estimation for Lithium Battery,” in *2021 6th International Conference on Smart Grid and Electrical Automation (ICSGEA)*, 2021, pp. 125–129, doi: 10.1109/ICSGEA53208.2021.00033.
- [17] T. Hansen and C.-J. Wang, “Support vector based battery state of charge estimator,” *Journal of Power Sources*, vol. 141, no. 2, pp. 351–358, 2005, doi: <https://doi.org/10.1016/j.jpowsour.2004.09.020>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378775304010626>
- [18] F. Zhang, M. M. Ur Rehman, H. Wang, Y. Levron, G. Plett, R. Zane, and D. Maksimović, “State-of-charge estimation based on microcontroller-implemented sigma-point kalman filter in a modular cell balancing system for lithium-ion battery packs,” in *2015 IEEE 16th Workshop on Con-*

- trol and Modeling for Power Electronics (COMPEL)*, 2015, pp. 1–7, doi: 10.1109/COMPEL.2015.7236525.
- [19] M. Kim and J. So, “Design of State of Charge and Health Estimation for Li-ion Battery Management System,” in *2022 19th International SoC Design Conference (ISOCC)*, 2022, pp. 322–323, doi: 10.1109/ISOCC56007.2022.10031465.
- [20] X. Tian, B. Jeppesen, T. Ikushima, F. Baronti, and R. Morello, “Accelerating state-of-charge estimation in FPGA-based Battery Management Systems,” in *6th Hybrid and Electric Vehicles Conference (HEVC 2016)*, 2016, pp. 1–6, doi: 10.1049/cp.2016.0964.
- [21] R. Morello, R. Di Rienzo, F. Baronti, R. Roncella, and R. Saletti, “System on chip battery state estimator: E-bike case study,” in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, 2016, pp. 2129–2134, doi: 10.1109/IECON.2016.7794140.
- [22] R. C. Kroeze and P. T. Krein, “Electrical battery model for use in dynamic electric vehicle simulations,” in *2008 IEEE Power Electronics Specialists Conference*, 2008, pp. 1336–1342, doi: 10.1109/PESC.2008.4592119.
- [23] V. Selvaraj and I. Vairavasundaram, “A comprehensive review of state of charge estimation in lithium-ion batteries used in electric vehicles,” *Journal of Energy Storage*, vol. 72, p. 108777, 2023, doi: <https://doi.org/10.1016/j.est.2023.108777>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352152X23021746>
- [24] G. L. Plett, *Battery Management Systems, Volume I: Battery Modeling*, ser. Artech House power engineering series. Artech House, 2015. [Online]. Available: <https://books.google.it/books?id=suLRCgAAQBAJ>
- [25] H. W. You, J. I. Bae, S. J. Cho, J. M. Lee, and S.-H. Kim, “Analysis of equivalent circuit models in lithium-ion batteries,” *AIP Advances*,

- vol. 8, p. 125101, 2018, doi: 10.1063/1.5054384. [Online]. Available: <https://doi.org/10.1063/1.5054384>
- [26] N. Ding, K. Prasad, T. T. Lie, and J. Cui, “State of Charge Estimation of a Composite Lithium-Based Battery Model Based on an Improved Extended Kalman Filter Algorithm,” *Inventions*, vol. 4, no. 4, 2019, doi: 10.3390/inventions4040066. [Online]. Available: <https://www.mdpi.com/2411-5134/4/4/66>
- [27] A. Loechte, I. Rojas Ruiz, and P. Gloesekoetter, “Battery State Estimation with ANN and SVR Evaluating Electrochemical Impedance Spectra Generalizing DC Currents,” *Applied Sciences*, vol. 12, no. 1, 2022, doi: 10.3390/app12010274. [Online]. Available: <https://www.mdpi.com/2076-3417/12/1/274>
- [28] A. Jossen, “Fundamentals of battery dynamics,” *Journal of Power Sources*, vol. 154, no. 2, pp. 530–538, 2006, doi: <https://doi.org/10.1016/j.jpowsour.2005.10.041>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0378775305014321>
- [29] W. Waag, S. Käbitz, and D. U. Sauer, “Experimental investigation of the lithium-ion battery impedance characteristic at various conditions and aging states and its influence on the application,” *Applied Energy*, vol. 102, pp. 885–897, 2013, doi: <https://doi.org/10.1016/j.apenergy.2012.09.030>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S030626191200671X>
- [30] N. Hallemans, D. Howey, A. Battistel, N. F. Saniee, F. Scarpioni, B. Wouters, F. La Mantia, A. Hubin, W. D. Widanage, and J. Lataire, “Electrochemical impedance spectroscopy beyond linearity and stationarity—A critical review,” *Electrochimica Acta*, vol. 466, p. 142939, 2023, doi: <https://doi.org/10.1016/j.electacta.2023.142939>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0013468623011143>

- [31] A. Sandschulte, R. Ferrero, L. Hardwick, and E. Patelli, "Approach to Wide-Frequency Battery Impedance Measurements in Commercial Applications," in *2019 IEEE 10th International Workshop on Applied Measurements for Power Systems (AMPS)*, 2019, pp. 1–6, doi: 10.1109/AMPS.2019.8897753.
- [32] L. Teo, V. R. Subramanian, and D. T. Schwartz, "Dynamic Electrochemical Impedance Spectroscopy of Lithium-ion Batteries: Revealing Underlying Physics through Efficient Joint Time-Frequency Modeling," *Journal of The Electrochemical Society*, vol. 168, no. 1, 2021, doi: 10.1149/1945-7111/abda04. [Online]. Available: <https://iopscience.iop.org/article/10.1149/1945-7111/abda04>
- [33] X. Du, J. Meng, K. Liu, Y. Zhang, S. Wang, J. Peng, and T. Liu, "Online Identification of Lithium-ion Battery Model Parameters with Initial Value Uncertainty and Measurement Noise," *Chinese Journal of Mechanical Engineering*, vol. 36, no. 1, p. 7, Jan 2023, doi: 10.1186/s10033-023-00846-0. [Online]. Available: <https://doi.org/10.1186/s10033-023-00846-0>
- [34] F. Poloei, A. Akbari, and Y.-F. Liu, "A Moving Window Least Mean Square Approach to State of Charge Estimation for Lithium Ion Batteries," in *2019 1st Global Power, Energy and Communication Conference (GPECOM)*, 2019, pp. 398–402, doi: 10.1109/GPECOM.2019.8778563.
- [35] S. W. Arsri, O. Wahyunggoro, and A. I. Cahyadi, "A Review of Adaptive Filter Algorithm-Based Battery State of Charge Estimation," in *2021 International Seminar on Machine Learning, Optimization, and Data Science (ISMODE)*, 2022, pp. 125–130, doi: 10.1109/ISMODE53584.2022.9743104.
- [36] S. Irum, T. Muhammad, U. Mukhtar, and P. H. Vincent, "Deep reinforcement learning based state of charge estimation and management of electric vehicle batteries," *IET Smart Grid*, vol. 6, no. 4, pp. 422 – 431, MAr 2023, doi: 10.1049/stg2.12110. [Online]. Available: <https://doi.org/10.1049/stg2.12110>



- [37] A. Manoharan, K. Begam, V. R. Aparow, and D. Sooriamoorthy, "Artificial Neural Networks, Gradient Boosting and Support Vector Machines for electric vehicle battery state estimation: A review," *Journal of Energy Storage*, vol. 55, p. 105384, 2022, doi: <https://doi.org/10.1016/j.est.2022.105384>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352152X22013780>
- [38] S. Jumah, A. Elezab, O. Zayed, R. Ahmed, M. Narimani, and A. Emadi, "State of Charge Estimation for EV Batteries Using Support Vector Regression," in *2022 IEEE Transportation Electrification Conference & Expo (ITEC)*, 2022, pp. 964–969, doi: [10.1109/ITEC53557.2022.9813811](https://doi.org/10.1109/ITEC53557.2022.9813811).
- [39] B. Fu, W. Wang, Y. Li, and Q. Peng, "An improved neural network model for battery smarter state-of-charge estimation of energy-transportation system," *Green Energy and Intelligent Transportation*, vol. 2, no. 2, p. 100067, 2023, doi: <https://doi.org/10.1016/j.geits.2023.100067>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2773153723000038>
- [40] Q. Wang, M. Ye, M. Wei, G. Lian, and Y. Li, "Deep convolutional neural network based closed-loop SOC estimation for lithium-ion batteries in hierarchical scenarios," *Energy*, vol. 263, p. 125718, 2023, doi: <https://doi.org/10.1016/j.energy.2022.125718>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0360544222026044>
- [41] Y. Li, K. Li, X. Liu, Y. Wang, and L. Zhang, "Lithium-ion battery capacity estimation — A pruned convolutional neural network approach assisted with transfer learning," *Applied Energy*, vol. 285, p. 116410, 2021, doi: <https://doi.org/10.1016/j.apenergy.2020.116410>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0306261920317773>
- [42] S. Wang, P. Ren, P. Takyi-Aninakwa, S. Jin, and C. Fernandez, "A Critical Review of Improved Deep Convolutional Neural Network for Multi-Timescale State Prediction of Lithium-Ion Batteries," *Energies*,

- vol. 15, no. 14, 2022, doi: 10.3390/en15145053. [Online]. Available: <https://www.mdpi.com/1996-1073/15/14/5053>
- [43] Y. Liu, Y. He, H. Bian, W. Guo, and X. Zhang, "A review of lithium-ion battery state of charge estimation based on deep learning: Directions for improvement and future trends," *Journal of Energy Storage*, vol. 52, p. 104664, 2022, doi: <https://doi.org/10.1016/j.est.2022.104664>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352152X22006776>
- [44] P. Kollmeyer, "Panasonic 18650PF Li-ion Battery Data," Mendeley Data V1, 2018, doi: 10.17632/wykht8y7tg.1, Last accessed on 2023-10-19.
- [45] B. Bole, C. Kulkarni, and M. Daigle, "Randomized Battery Usage Data Set," NASA Prognostics Data Repository, NASA Ames Research Center, <https://www.nasa.gov/intelligent-systems-division/discovery-and-systems-health/pcoe/pcoe-data-set-repository/>, Last accessed on 2023-10-19.
- [46] P. Kollmeyer, C. Vidal, M. Naguib, and M. Skells, "LG 18650HG2 Li-ion Battery Data and Example Deep Neural Network xEV SOC Estimator Script," Mendeley Data V3, 2020, doi: 10.17632/cp3473x7xv.3, Last accessed on 2023-10-19.
- [47] Q. Shi, Z. Jiang, Z. Wang, X. Shao, and L. He, "State of charge estimation by joint approach with model-based and data-driven algorithm for lithium-ion battery," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–10, 2022, doi: 10.1109/TIM.2022.3199253.
- [48] R. Morello, F. Baronti, X. Tian, T. Chau, R. Di Rienzo, R. Roncella, B. Jeppesen, W. H. Lin, T. Ikushima, and R. Saletti, "Hardware-in-the-loop simulation of FPGA-based state estimators for electric vehicle batteries," in *2016 IEEE 25th International Symposium on Industrial Electronics (ISIE)*, 2016, pp. 280–285, doi: 10.1109/ISIE.2016.7744903.

- [49] Z. Cui, L. Kang, L. Li, L. Wang, and K. Wang, "A combined state-of-charge estimation method for lithium-ion battery using an improved BGRU network and UKF," *Energy*, vol. 259, p. 124933, 2022, doi: <https://doi.org/10.1016/j.energy.2022.124933>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0360544222018345>
- [50] Z. Ni and Y. Yang, "A Combined Data-Model Method for State-of-Charge Estimation of Lithium-Ion Batteries," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–11, 2022, doi: 10.1109/TIM.2021.3137550.
- [51] F. Baronti, R. Roncella, R. Saletti, and W. Zamboni, "FPGA implementation of the mix algorithm for state-of-charge estimation of Lithium-ion batteries," in *IECON 2014 - 40th Annual Conference of the IEEE Industrial Electronics Society*, 2014, pp. 5641–5646, doi: 10.1109/IECON.2014.7049364.
- [52] M. Stighezza, "FPGA Automatic Code Generation of an Algorithm for State-of-Charge Estimation of Lithium-ion Battery Cells," Master's thesis, Università degli Studi di Parma - Facoltà di Ingegneria, 2020.
- [53] R. Morello, R. Di Rienzo, R. Roncella, R. Saletti, and F. Baronti, "Tuning of Moving Window Least Squares-based algorithm for online battery parameter estimation," in *2017 14th International Conference on Synthesis, Modeling, Analysis and Simulation Methods and Applications to Circuit Design (SMACD)*, 2017, pp. 1–4, doi: 10.1109/SMACD.2017.7981558.
- [54] B. Xia, Z. Lao, R. Zhang, Y. Tian, G. Chen, Z. Sun, W. Wang, W. Sun, Y. Lai, M. Wang, and H. Wang, "Online Parameter Identification and State of Charge Estimation of Lithium-Ion Batteries Based on Forgetting Factor Recursive Least Squares and Nonlinear Kalman Filter," *Energies*, vol. 11, no. 1, 2018, doi: 10.3390/en11010003. [Online]. Available: <https://www.mdpi.com/1996-1073/11/1/3>
- [55] B. Kumar, N. Khare, and P. Chaturvedi, "FPGA-based design of advanced BMS implementing SoC/SoH estimators," *Microelectronics Reliability*,

- vol. 84, pp. 66–74, 2018, doi: <https://doi.org/10.1016/j.microrel.2018.03.015>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S002627141830129X>
- [56] C. Cortes and V. Vapnik, “Support-vector networks,” *Machine Learning*, vol. 20, p. 273–297, 1995, doi: <https://doi.org/10.1007/BF00994018>.
- [57] T. Fletcher, “Support Vector Machines Explained,” 2008, Available online: [https://www.csd.uwo.ca/~xling/cs860/papers/SVM\\_Explained.pdf](https://www.csd.uwo.ca/~xling/cs860/papers/SVM_Explained.pdf), Last accessed on 2023-10-22.
- [58] G. W. Flake and S. Lawrence, “Efficient SVM Regression Training with SMO,” *Machine Learning*, vol. 46, p. 271–290, 2002, doi: <https://doi.org/10.1023/A:1012474916001>.
- [59] M. Greenacre, P. J. F. Groenen, T. Hastie, A. I. D’Enza, A. Markos, and E. Tuzhilina, “Principal component analysis,” *Nature Reviews Methods Primers*, vol. 2, no. 1, p. 100, Dec 2022, doi: [10.1038/s43586-022-00184-w](https://doi.org/10.1038/s43586-022-00184-w). [Online]. Available: <https://doi.org/10.1038/s43586-022-00184-w>
- [60] H. Wang and D. Hu, “Comparison of SVM and LS-SVM for Regression,” in *2005 International Conference on Neural Networks and Brain*, vol. 1, 2005, pp. 279–283, doi: [10.1109/ICNNB.2005.1614615](https://doi.org/10.1109/ICNNB.2005.1614615).
- [61] M. Aftowicz, K. Lehniger, and P. Langendoerfer, “Scalable FPGA Hardware Accelerator for SVM Inference,” in *2022 11th Mediterranean Conference on Embedded Computing (MECO)*, 2022, pp. 1–4, doi: [10.1109/MECO55406.2022.9797110](https://doi.org/10.1109/MECO55406.2022.9797110).
- [62] Mathworks, “Fitrsvm MATLAB Function,” Available online: <https://it.mathworks.com/help/stats/fitrsvm.html>, Last accessed on 2023-10-22.
- [63] DieselNet, “US06 Drive Cycle,” Available online: [https://dieselnet.com/standards/cycles/ftp\\_us06.php](https://dieselnet.com/standards/cycles/ftp_us06.php), Last accessed on 2023-10-22.

- [64] A. H. U. Bhatti, S. A. A. Kazmi, A. Tariq, and G. Ali, "Development and analysis of electric vehicle driving cycle for hilly urban areas," *Transportation Research Part D: Transport and Environment*, vol. 99, p. 103025, 2021, doi: <https://doi.org/10.1016/j.trd.2021.103025>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1361920921003230>
- [65] G. C. Batista, D. L. Oliveira, O. Saotome, and W. L. Silva, "A low-power asynchronous hardware implementation of a novel svm classifier, with an application in a speech recognition system," *Microelectronics Journal*, vol. 105, p. 104907, 2020, doi: <https://doi.org/10.1016/j.mejo.2020.104907>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0026269220305061>
- [66] D. Mahmoodi, A. Soleimani, H. Khosravi, and M. Taghizadeh, "FPGA Simulation of Linear and Nonlinear Support Vector Machine," *Journal of Software Engineering and Applications*, vol. 4, no. 5, pp. 320–328, 2011, doi: [10.4236/jsea.2011.45036](https://doi.org/10.4236/jsea.2011.45036). [Online]. Available: <https://www.scirp.org/journal/paperinformation.aspx?paperid=5072>
- [67] H. Wang and S. Yang, "Electricity Consumption Prediction Based on SVR with Ant Colony Optimization," *Journal of Electrical Engineering*, vol. 11, p. 6928–6934, 2013. [Online]. Available: <http://www.iaesjournal.com/online/index.php/TELKOMNIKA/article/view/3557>
- [68] W.-C. Hong, Y. Dong, F. Zheng, and C.-Y. Lai, "Forecasting urban traffic flow by SVR with continuous ACO," *Applied Mathematical Modelling*, vol. 35, no. 3, pp. 1282–1291, 2011, doi: <https://doi.org/10.1016/j.apm.2010.09.005>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0307904X10003409>
- [69] L. Zheng, M. Yu, and S. Yu, "Support vector regression and ant colony optimization for combustion performance of boilers," in *2008 Fourth International Conference on Natural Computation*, vol. 2, 2008, pp. 178–182, doi:10.1109/ICNC.2008.479.

- [70] M. Dorigo and C. Blum, "Ant colony optimization theory: A survey," *Theoretical Computer Science*, vol. 344, no. 2, pp. 243–278, 2005, doi: <https://doi.org/10.1016/j.tcs.2005.05.020>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0304397505003798>
- [71] "Panasonic Lithium Ion NCR18650PF Datasheet," Available online: <https://na.industrial.panasonic.com/products/batteries/rechargeable-batteries/lineup/lithium-ion/series/90729/model/90730>, Last accessed on 2023-10-22.
- [72] "Xilinx Inc. 7 Series FPGAs Data Sheet: Overview (DS180)," Available online: [https://docs.xilinx.com/v/u/en-US/ds180\\_7Series\\_Overview](https://docs.xilinx.com/v/u/en-US/ds180_7Series_Overview), Last accessed on 2023-10-22.
- [73] J. Li, M. Ye, W. Meng, X. Xu, and S. Jiao, "A novel state of charge approach of lithium ion battery using least squares support vector machine," *IEEE Access*, vol. 8, pp. 195 398–195 410, 2020, doi:10.1109/ACCESS.2020.3033451.
- [74] M. M. Alobaedy, A. A. Khalaf, and I. D. Muraina, "Analysis of the number of ants in ant colony system algorithm," in *2017 5th International Conference on Information and Communication Technology (ICoICT)*, 2017, pp. 1–5, doi: 10.1109/ICoICT.2017.8074653.
- [75] S. Baksa and W. Yourey, "Consumer-Based Evaluation of Commercially Available Protected 18650 Cells," *Batteries*, vol. 4, no. 3, 2018, doi: 10.3390/batteries4030045. [Online]. Available: <https://www.mdpi.com/2313-0105/4/3/45>
- [76] P. Gu, Z. Zhou, S. Qu, C. Zhang, and B. Duan, "Influence Analysis and Optimization of Sampling Frequency on the Accuracy of Model and State-of-Charge Estimation for LiNCM Battery," *Energies*, vol. 12, no. 7, 2019, doi: 10.3390/en12071205. [Online]. Available: <https://www.mdpi.com/1996-1073/12/7/1205>
- [77] Y. Zhang, J. Yu, C. Xia, K. Yang, H. Cao, and Q. Wu, "Research on GA-SVM Based Head-Motion Classification via Mechanomyography Feature

- Analysis,” *Sensors*, vol. 19, no. 9, 2019, doi: 10.3390/s19091986. [Online]. Available: <https://www.mdpi.com/1424-8220/19/9/1986>
- [78] “ITECH IT-M3400 Datasheet,” Available online: <https://www.itechate.com/en/product/dc-power-supply/IT-M3400.html>, Last accessed on 2023-10-22.
- [79] “Duplicating Real Life Load Profiles in the Laboratory,” Available online: <http://basytec.de/applications/reallife.pdf>, Last accessed on 2023-10-22.
- [80] L. Zhou, L. He, Y. Zheng, X. Lai, M. Ouyang, and L. Lu, “Massive battery pack data compression and reconstruction using a frequency division model in battery management systems,” *Journal of Energy Storage*, vol. 28, p. 101252, 2020, doi: <https://doi.org/10.1016/j.est.2020.101252>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S2352152X19312587>
- [81] S. M. Ross, *Introduction to Probability and Statistics for Engineers and Scientists*, 5th ed. Elsevier, 2014, doi: <https://doi.org/10.1016/C2013-0-19397-X>.
- [82] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes: The Art of Scientific Computing*, 3rd ed. Cambridge University Press, 2007.
- [83] H. Huang, X. Wei, and Y. Zhou, “An overview on twin support vector regression,” *Neurocomputing*, vol. 490, pp. 80–92, 2022, doi: <https://doi.org/10.1016/j.neucom.2021.10.125>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0925231222003101>
- [84] M. V. Haeverbeke, M. Stock, and B. De Baets, “Equivalent Electrical Circuits and Their Use Across Electrochemical Impedance Spectroscopy Application Domains,” *IEEE Access*, vol. 10, pp. 51 363–51 379, 2022, doi: 10.1109/ACCESS.2022.3174067.
- [85] W. Huang and J. A. Abu Qahouq, “An Online Battery Impedance Measurement Method Using DC–DC Power Converter Control,” *IEEE Transactions on Industrial Electronics*, vol. 61, no. 11, pp. 5987–5995, 2014, doi: 10.1109/TIE.2014.2311389.

- [86] R. Ferrero, C. Wu, A. Carboni, S. Toscani, M. De Angelis, H. George-Williams, E. Patelli, and P. A. Pegoraro, "Low-Cost Battery Monitoring by Converter-Based Electrochemical Impedance Spectroscopy," in *2017 IEEE International Workshop on Applied Measurements for Power Systems (AMPS)*, 2017, pp. 1–6, doi: 10.1109/AMPS.2017.8078334.
- [87] L. Regnacq, Y. Wu, N. Neshatvar, D. Jiang, and A. Demosthenous, "A Goertzel Filter-Based System for Fast Simultaneous Multi-Frequency EIS," *IEEE Transactions on Circuits and Systems II: Express Briefs*, vol. 68, no. 9, pp. 3133–3137, 2021, doi: 10.1109/TCSII.2021.3092069.
- [88] E. Din, C. Schaef, K. Moffat, and J. T. Stauth, "A Scalable Active Battery Management System With Embedded Real-Time Electrochemical Impedance Spectroscopy," *IEEE Transactions on Power Electronics*, vol. 32, no. 7, pp. 5688–5698, 2017, doi: 10.1109/TPEL.2016.2607519.
- [89] Z. Xia and J. A. Abu Qahouq, "High Frequency Online Battery Impedance Measurement Method Using Voltage and Current Ripples Generated by DC-DC Converter," in *2020 IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2020, pp. 1333–1338, doi: 10.1109/APEC39645.2020.9124465.
- [90] Z. Xia and J. A. A. Qahouq, "An online battery impedance spectrum measurement method with increased frequency resolution," in *2018 IEEE Applied Power Electronics Conference and Exposition (APEC)*, 2018, pp. 1930–1933, doi: 10.1109/APEC.2018.8341281.
- [91] H. Beiranvand, J. M. Placzek, M. Liserre, G. Zampardi, D. C. Brogioli, and F. La Mantia, "Review of Power Converter Topologies for Electrochemical Impedance Spectroscopy of Lithium-Ion Batteries," in *2022 24th European Conference on Power Electronics and Applications (EPE'22 ECCE Europe)*, 2022, pp. 1–10.



- [92] G. Dotelli, R. Ferrero, P. G. Stampino, S. Latorrata, and S. Toscani, "PEM Fuel Cell Drying and Flooding Diagnosis With Signals Injected by a Power Converter," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 8, pp. 2064–2071, 2015, doi: 10.1109/TIM.2015.2406051.
- [93] —, "Low-Cost PEM Fuel Cell Diagnosis Based on Power Converter Ripple With Hysteresis Control," *IEEE Transactions on Instrumentation and Measurement*, vol. 64, no. 11, pp. 2900–2907, 2015, doi: 10.1109/TIM.2015.2434093.
- [94] N. Hallemans, W. D. Widanage, X. Zhu, S. Moharana, M. Rashid, A. Hubin, and J. Lataire, "Operando electrochemical impedance spectroscopy and its application to commercial Li-ion batteries," *Journal of Power Sources*, vol. 547, p. 232005, 2022, doi: <https://doi.org/10.1016/j.jpowsour.2022.232005>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S037877532200982X>
- [95] M. Naguib, P. Kollmeyer, and A. Emadi, "Lithium-Ion Battery Pack Robust State of Charge Estimation, Cell Inconsistency, and Balancing: Review," *IEEE Access*, vol. 9, pp. 50 570–50 582, 2021, doi: 10.1109/ACCESS.2021.3068776.
- [96] Tesla Motors, "2012 TESLA MODEL S: SPECIFICATIONS AND FEATURES," 2012, [https://www.tesla.com/sites/default/files/2012\\_model\\_s\\_specifications\\_and\\_features.pdf](https://www.tesla.com/sites/default/files/2012_model_s_specifications_and_features.pdf) Last accessed on 2023-10-30.
- [97] F. Adegbohun, A. von Jouanne, B. Phillips, E. Agamloh, and A. Yokochi, "High Performance Electric Vehicle Powertrain Modeling, Simulation and Validation," *Energies*, vol. 14, no. 5, 2021, doi: 10.3390/en14051493. [Online]. Available: <https://www.mdpi.com/1996-1073/14/5/1493>
- [98] M. Ehsani, Y. Gao, S. Longo, and K. M. Ebrahimi, *Modern Electric Hybrid Electric and Fuel Cell Vehicles*, 3rd ed. CRC Press, 2018, doi: <https://doi.org/10.1201/9780429504884>.

- [99] N. Hinov, P. Punov, B. Gilev, and G. Vacheva, "Model-Based Estimation of Transmission Gear Ratio for Driving Energy Consumption of an EV," *Electronics*, vol. 10, no. 13, 2021, doi: 10.3390/electronics10131530. [Online]. Available: <https://www.mdpi.com/2079-9292/10/13/1530>
- [100] X. Sun, C. Shao, G. Wang, L. Yang, X. Li, and Y. Yue, "Research on electrical brake of a series-parallel hybrid electric vehicle," in *2016 World Congress on Sustainable Technologies (WCST)*, 2016, pp. 70–75, doi: 10.1109/WCST.2016.7886594.
- [101] T. Gillespie, *Fundamentals of Vehicle Dynamics*. Society of Automotive Engineers Inc., 1992, available online: <https://www.sae.org/publications/books/content/r-114/>.
- [102] National Oceanic and Atmospheric Administration, "U.S. Standard Atmosphere," U.S. Government Printing Office, Washington D.C., Tech. Rep., 1962, available online: [https://www.ngdc.noaa.gov/stp/space-weather/online-publications/miscellaneous/us-standard-atmosphere-1976/us-standard-atmosphere\\_st76-1562\\_noaa.pdf](https://www.ngdc.noaa.gov/stp/space-weather/online-publications/miscellaneous/us-standard-atmosphere-1976/us-standard-atmosphere_st76-1562_noaa.pdf).
- [103] P. S. Grover, "Modeling of Rolling Resistance Test Data," *SAE Transactions*, vol. 107, pp. 497–506, 1998. [Online]. Available: <http://www.jstor.org/stable/44740977>
- [104] B. P. Wiegand, "Estimation of the rolling resistance of tires," in *SAE 2016 World Congress and Exhibition*. SAE International, apr 2016, doi: <https://doi.org/10.4271/2016-01-0445>. [Online]. Available: <https://doi.org/10.4271/2016-01-0445>
- [105] D. E. Hall and J. C. Moreland, "Fundamentals of Rolling Resistance," *Rubber Chemistry and Technology*, vol. 74, no. 3, pp. 525–539, July 2001, doi: 10.5254/1.3547650. [Online]. Available: <https://meridian.allenpress.com/rct/article-abstract/74/3/525/92181/Fundamentals-of-Rolling-Resistance?redirectedFrom=fulltext>

- [106] R. Rajamani, *Vehicle Dynamics and Control*, 2nd ed., ser. Mechanical Engineering Series. Springer New York, December 2011, doi: <https://doi.org/10.1007/978-1-4614-1433-9>.
- [107] H. Pacejka, *Tire and Vehicle Dynamics*, 3rd ed. Butterworth-Heinemann, April 2012, doi: <https://doi.org/10.1016/C2010-0-68548-8>.
- [108] M. Dalboni, D. Mangoni, A. Soldati, F. Corradini, A. Tasora, F. Savi, and D. Lusignani, “A fast and lightweight dynamics model oriented to electric vehicle design,” in *IECON 2017 - 43rd Annual Conference of the IEEE Industrial Electronics Society*, 2017, pp. 4603–4608, doi: 10.1109/IECON.2017.8216793.
- [109] R. N. Jazar, *Vehicle Dynamics: Theory and Application*, 3rd ed. Springer Cham, May 2017, doi: <https://doi.org/10.1007/978-3-319-53441-1>.
- [110] J. Y. Wong, *Theory of Ground Vehicles*, 3rd ed. Wiley Interscience, March 2001, doi: 10.1002/9781119719984.
- [111] S. Hong, H. Hwang, D. Kim, S. Cui, and I. Joe, “Real driving cycle-based state of charge prediction for ev batteries using deep learning methods,” *Applied Sciences*, vol. 11, no. 23, 2021, doi: 10.3390/app112311285. [Online]. Available: <https://www.mdpi.com/2076-3417/11/23/11285>
- [112] K. N. Genikomsakis and G. Mitrentsis, “A computationally efficient simulation model for estimating energy consumption of electric vehicles in the context of route planning applications,” *Transportation Research Part D: Transport and Environment*, vol. 50, pp. 98–118, 2017, doi: <https://doi.org/10.1016/j.trd.2016.10.014>. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1361920915302881>
- [113] J. G. Hayes and K. Davis, “Simplified electric vehicle powertrain model for range and energy consumption based on EPA coast-down parameters and test validation by Argonne National Lab data on the Nissan Leaf,” in *2014 IEEE*

- Transportation Electrification Conference and Expo (ITEC)*, 2014, pp. 1–6, doi: 10.1109/ITEC.2014.6861831.
- [114] I. Miri, A. Fotouhi, and N. Ewin, “Electric vehicle energy consumption modelling and estimation—A case study,” *International Journal of Energy Research*, vol. 45, no. 1, pp. 501–520, 2021, doi: <https://doi.org/10.1002/er.5700>. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/er.5700>
- [115] G. Sieklucki, “An Investigation into the Induction Motor of Tesla Model S Vehicle,” in *2018 International Symposium on Electrical Machines (SME)*, 2018, pp. 1–6, doi: 10.1109/ISEM.2018.8442648.
- [116] K. Nguyen-Thac, T. Orłowska-Kowalska, and G. Tarchala, “Influence of the stator winding resistance on the field-weakening operation of the DRFOC induction motor drive,” *Bulletin of the Polish Academy of Sciences Technical Sciences*, vol. 60, no. No 4, pp. 815–823, 2012, doi: 10.2478/v10175-012-0095-5. [Online]. Available: [http://journals.pan.pl/Content/83662/PDF/paper\\_18.pdf](http://journals.pan.pl/Content/83662/PDF/paper_18.pdf)
- [117] D. Swierczynski, P. Wojcik, M. P. Kazmierkowski, and M. Janaszek, “Direct torque controlled PWM inverter fed PMSM drive for public transport,” in *2008 10th IEEE International Workshop on Advanced Motion Control*, 2008, pp. 716–720, doi: 10.1109/AMC.2008.4516155.
- [118] O. Siddiqui, D. Simacek, R. Hoang, N. Famiglietti, B. Nguyen, and J. Landerville, “Characterizing Regenerative Coast-Down Deceleration in Tesla Model 3, S, and X,” in *WCX SAE World Congress Experience*. SAE International, apr 2020, doi: <https://doi.org/10.4271/2020-01-0883>. [Online]. Available: <https://doi.org/10.4271/2020-01-0883>
- [119] R. Thomas, L. Garbuio, L. Gerbaud, and H. Chazal, “Modeling and design analysis of the Tesla Model S induction motor,” in *2020 International Con-*

- ference on Electrical Machines (ICEM)*, vol. 1, 2020, pp. 495–501, doi: 10.1109/ICEM49940.2020.9270646.
- [120] J. Drobnik and P. Jain, “Electric and Hybrid Vehicle Power Electronics Efficiency, Testing and Reliability,” *World Electric Vehicle Journal*, vol. 6, no. 3, pp. 719–730, 2013, doi: 10.3390/wevj6030719. [Online]. Available: <https://www.mdpi.com/2032-6653/6/3/719>
- [121] P. Iora and L. Tribioli, “Effect of Ambient Temperature on Electric Vehicles’ Energy Consumption and Range: Model Definition and Sensitivity Analysis Based on Nissan Leaf Data,” *World Electric Vehicle Journal*, vol. 10, no. 1, 2019, doi: 10.3390/wevj10010002. [Online]. Available: <https://www.mdpi.com/2032-6653/10/1/2>
- [122] AutoData, “Tesla Model S 85 85 kWh (362 Hp) 2012, 2013, 2014, 2015 Specs,” <https://www.auto-data.net/it/tesla-model-s-85-85-kwh-362hp-18611> Last accessed on 2023-10-30.



# Acknowledgments

This dissertation is the result of my hard work and dedication. However, it would not have been possible without all the people who have crossed the path of my life and supported me in their own way, whether it was professional support or a smile between a chat and another. In particular, I would like to express my profound gratitude:

To my mentors, Ilaria De Munari and Valentina Bianchi, for pointing me in the right direction since the Master's Degree and improving my work with valuable advice and suggestions.

To Professor Roberto Ferrero, who welcomed me into his laboratory at the University of Liverpool, supervised my work with his expertise and, above all, made me feel at home in a foreign country.

To all the Professors at the University of Parma, who shared some help, advice and ideas for my work.

To old Friends, who have always been there for light-hearted conversations and shared moments, easing my mind from work and daily troubles.

To the colleagues at Palazzina 4, who became new Friends sharing my path, with whom I enjoyed engaging in nice and interesting conversations while sharing a coffee break.

*Infine, ma non per importanza, un ringraziamento speciale deve andare alla mia Famiglia, che ha supportato il mio percorso e dato una parola di conforto quando ne avevo più bisogno, e alla mia amata Sabrina, che nonostante tutto mi è stata accanto anche quando la lontananza o le difficoltà erano tante.*