University of Pavia

Faculty of Engineering

Department of Electrical, Computer and Biomedical Engineering

Model Predictive Control Strategies for Advanced Battery Management Systems

Author:
Marcello Torchio

Supervisor:
Prof. Lalo Magni

Co-supervisor:
Prof. Davide M. Raimondo

Contents

Abstract

Consumer electronics, wearable and personal health devices, power networks, microgrids, and hybrid electric vehicles (HEVs) are some of the many applications where Lithium-ion (Li-ion) batteries are employed. From a manufacturer point of view, the optimal design and management of such electrochemical accumulators are important aspects for ensuring safe and profitable operations. The adoption of mathematical models can support the achievement of the best performance, while saving time and money. In the literature, all the models used to describe the behavior of a Li-ion battery belong to one of the two following families: (*i*) Equivalent Circuit Models (ECMs), and (*ii*) Electrochemical Models (EMs). While the former family represents the battery dynamics by means of electrical circuits, the latter resorts to first principles laws of modeling. As a first contribution, this Thesis provides a thorough investigation of the pseudo-two-dimensional (P2D) Li-ion battery EM. In particular, the objectives are to provide: (*i*) a detailed description of the model formulation, (*ii*) the Li-ION SIMulation BAttery (LIONSIMBA) toolbox as a finite volume Matlab implementation of the P2D model, for design, simulation, and control of Li-ion cells or battery packs, (*iii*) a validation of the proposed tool with respect to the COMSOL MultiPhysics commercial software and the Newman's DUALFOIL code, and (*iv*) some demonstrative simulations involving thermal dynamics, a hybrid charge-discharge cycle emulating the throttle of an HEV, and a battery pack of series connected cells. The second contribution is related to the development of several charging strategies for Advanced

Battery Management Systems (ABMSs), where predictive approaches are employed to attain optimal control. Model Predictive Control (MPC) refers to a particular family of control algorithms that, according to a mathematical model, predicts the future behavior of a plant, while considering inputs and outputs constraints. According to this paradigm, in this Thesis different ABMSs strategies have been developed, and their effectiveness shown through simulations. Due to the complexity of the P2D model, its inclusion within an MPC context could prevent the online application of the control algorithm. For this reason, different approximations of the P2D dynamics are proposed and their MPC formulations carefully explained. In particular, finite step response, autoregressive exogenous, piecewise affine, and linear time varying approximations are presented. For all the aforementioned reformulations, the closed-loop performance are evaluated considering the P2D implementation of LIONSIMBA as the real plant. The closed-loop simulations highlight the suitability of the MPC paradigm to be employed for the development of the future ABMSs. In fact, its ability to predict the future behavior of the cell while considering operating constraints can help in preventing possible safety issues and improving the charging performance. Finally, the reliability and efficiency of the proposed Matlab toolbox in simulating the P2D dynamics, support the idea that LIONSIMBA can significantly contribute in the advance of the battery field.

Chapter 1

Introduction

The ever-increasing usage of consumer electronics, together with the escalating demand of electric cars and storage devices, has driven industry towards the development of Electrochemical Accumulators (EAs) of increasing performance and reliability. During the many decades of research, different chemistries of batteries have been developed, such as Nickel Cadmium (NiCd), Nickel Metal Hydride (NiMH), Lead Acid, Lithium ion (Li-ion), Lithium ion Polymer (Li-Poly), and Lithium ion metal (Li-metal) (see e.g., [Dhar et al., 1997, Linden, 1984, Ruetschi, 1977, Zhang, 2011]). Among EAs, Li-ion batteries provide one of the best tradeoffs in terms of power density, low weight, cell voltage, low self-discharge, and wide temperature operations [Van den Bossche et al., 2006, Besenhard, 2008]. Although the electrochemical characteristics of Li-ion cells are remarkable, Battery Management Systems (BMSs) are employed for monitoring and management purposes with the objective of providing good operating performance, and prolong the lifetime. Among its tasks, a BMS:

- ensures safety conditions during battery operations;

- provides proper charging protocols;
- determines the remaining charge of the battery;
- estimates the remaining lifetime;
- balances the charge among several cells;
- monitors and logs the battery history (i.e., stores of the number of full charges, full discharges, maximum and minimum voltage values, etc.);
- provides a communication interface for other devices (e.g., provide to an operating system the information about the battery of a personal computer).

When charging Li-ion cells, different protocols can be employed by the BMSs [Shen et al., 2012, Keil and Jossen, 2016]. The most common charging algorithm used in industrial BMSs rely on the so-called Constant Current - Constant Voltage (CC-CV) protocol [Sauer, 2009]. As shown in Fig. 1.1, the approach first applies a galvanostatic charge (CC stage). Once the cell voltage reaches a threshold value (V_{thr}), a potentiostatic charge (CV stage) follows. During this second phase, the current flowing through the cell decreases exponentially as a function of the governing physics. Usually, the charging phase stops when the applied current reaches a lower bound value (e.g., 0.01C), or when a maximum time index ($t_{\max}$) is reached.

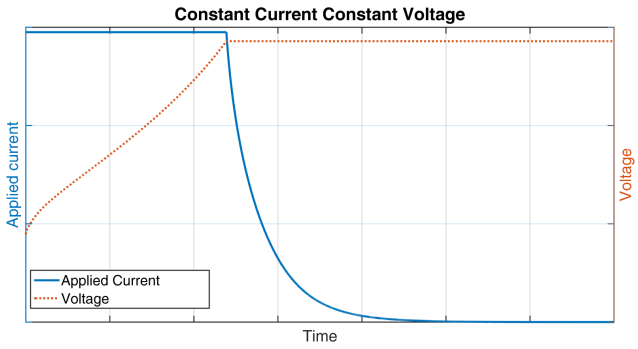


Figure 1.1: Example of CC-CV charging protocol

Another common algorithm used in industrial BMSs is the so-called Multistage Constant Current (MCC) protocol [Luo et al., 2009]. According to this strategy, the following charging algorithm is carried out Therefore,

Algorithm 1 MCC charging algorithm

1: $I = I_{\text{init}}$ ▷ Initialize the value of the applied current.
2: **while** $I > I_{\text{lower}}$ **do**
3: **if** $V \geq V_{\text{thr}}$ **then** $I \leftarrow \alpha\, I$
4: **end if**
5: Constant Current (CC) charge by applying I
6: **end while**

according to the MCC approach, starting from an initial value of current (I_{init}), the cell is charged with a CC strategy. As soon as the voltage of the cell reaches a threshold value (V_{thr}), the applied current is reduced according to the scheme

$$I \leftarrow \alpha\, I, \text{ where } 0 < \alpha < 1.$$

and the charging process continues according to the CC strategy. As soon as the value of I drops below a given value (I_{lower}), the charge stops. The

value of I_{lower}, when reached, usually corresponds to a fully charged state of the battery. In Fig. 1.2, an example of the application of the MCC algorithm in 1 is depicted. Other chargers make use of the so called Pulshe

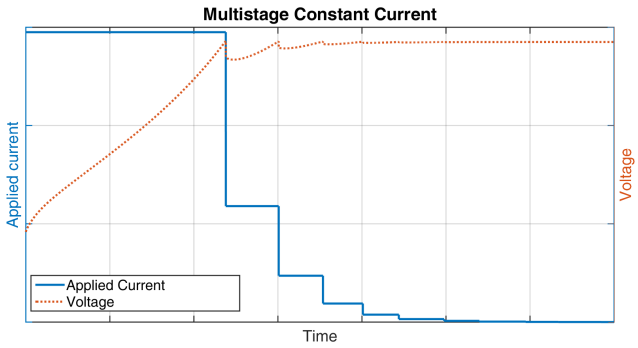


Figure 1.2: Example of MCC charging protocol

Charging (PC) algorithm [Chen et al., 2010]. As shown in Fig. 1.3, current pulses of fixed duration and amplitude are applied to the cell. Rest periods alternate between one pulse and the next. As soon as the cell reaches a threshold voltage, the charge phase stops. A thorough description of the

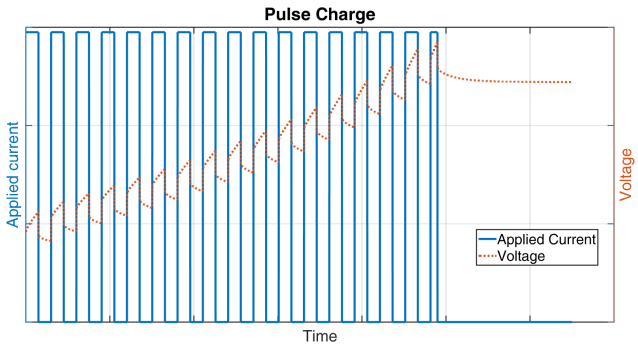


Figure 1.3: Example of PC charging protocol

various charging protocols employed in industrial BMSs can be found in

[Shen et al., 2012, Keil and Jossen, 2016] and the references therein.

While the aforementioned approaches provide reasonable performance, better results can be achieved by exploiting a mathematical model of the Li-ion cell [Ramadesigan et al., 2012]. Advanced BMSs (ABMSs) rely on accurate mathematical descriptions to attain effective control, monitoring, and diagnostics [Chaturvedi et al., 2010]. Different models have been proposed in the literature over the years. In particular two main families of models are commonly used in this field: (*i*) Equivalent Circuit Models (ECMs), and (*ii*) Electrochemical Models (EMs). While ECMs are less computationally expensive, the EMs provide more accurate description of the electrochemical phenomena occurring inside the cell. One important objective of ABMSs is to provide fast charging strategies while taking into account safety constraints (such as voltage and temperature) as well as aging constraints. For these reasons, the Model Predictive Control (MPC) paradigm is particularly suitable for the development of charging strategies for ABMSs. According to a given cost function, MPC provides a control action by optimizing the future behavior of the controlled plant, while considering inputs and outputs constraints. Different predictive control algorithms have been proposed in the literature over the years for the development of ABMSs. In particular, based on ECMs, in [Yan et al., 2011] the authors suggest a linear MPC scheme for minimum charging time control which takes into account the thermal excursion in battery packs, while the approach in [Xavier and Trimboli, 2015] considers fast charging strategies in the presence of input, output, and state constraints. Other authors addressed the development of Nonlinear MPC (NMPC) strategies for ABMSs: in [Samadi and Saif, 2014], based on an ECM, an optimal charging protocol was proposed to minimize

the charge unbalancing among series connected cells. Besides the use of the MPC paradigm for ABMSs, other control approaches have been proposed in the literature. For instance, in [Suthar et al., 2013, Perez et al., 2016], Optimal Control Problems (OCPs) were proposed to deal with reformulated versions of an EM to derive control algorithms able to maximize the transferred charge and minimize the charging time. The authors in [Tsang and Chan, 2009] and [Tsang and Chan, 2011], proposed the design of traditional PID control algorithms, while fuzzy logic techniques were considered in [Huang et al., 2009] and [Hsieh et al., 2001]. Finally, [Moura et al., 2013], proposed a modified reference governor control algorithm.

Despite the considerable number of control techniques proposed in the literature, only a few have placed particular emphasis on the aspects related to the aging of the Li-ion cells. For instance, in [Moura et al., 2009] an optimal control based approach aims to minimize the Solid Electrolyte Interface (SEI) film growth for parallel connected cells. An ECM together with a lookup table were used to account for the SEI dynamics as a function of the input current and State Of Charge (SOC). In [Klein et al., 2011] an NMPC strategy based on a EM was proposed, in which constraints over the anode overpotential were enforced in order to avoid possible lithium deposition phenomena.

1.1 Thesis contribution

This Thesis provides two main contribution to the field of the simulation and control of Li-ion batteries.

The first contribution refers to the detailed description of the numerical implementation of the well known theory-based EM named Pseudo

Two-Dimensional (P2D) model. A thorough dissertation about the numerical issues and the relative loopholes is addressed, and the complete set of equations, parameters, and the discretization approach used to implement such model is provided. As a result of this dissertation, the Lithium-ION SIMulation BAttery (LIONSIMBA) toolbox is presented. Such toolbox, distributed as a freely available Matlab® software, is able to simulate the electrochemical behavior of a Li-ion cell or battery packs composed of series-connected cells. Moreover, it provides a ready-to-use environment for the development of control algorithms, assessment of different cell parameterizations, and optimization of the design parameters.

The second contribution of this Thesis is related to the development of several MPC strategies for ABMSs. Fast charging, health-aware, linear time varying, and piecewise control strategies are proposed. Differently from what already presented in the literature, the algorithms developed in this Thesis exploit reformulated versions of the P2D dynamics in order to carry out the control action. In fact, due to the complexity of the P2D equations, linearization and order reduction techniques are employed to suitably embed the P2D model within an online MPC scheme. The effectiveness of the control algorithms is then assessed considering the P2D model as the real plant, simulated using the LIONSIMBA toolbox.

1.2 Structure of the Thesis

The Thesis is structured as follows:

- **Chapter 2**: This chapter provides an introduction to the batteries systems where the main acronyms and the common nomenclature are

presented. Following, a detailed description of a finite volume method for a pseudo-two-dimensional Li-ion battery model suitable for the development of model-based advanced battery management systems is given. In particular the following issues are addressed: (*i*) a detailed description of the model formulation, (*ii*) a compiled and parameterizable Matlab framework for battery design, simulation, and control of Li-ion cells or battery packs, (*iii*) a validation of the proposed numerical implementation with respect to the COMSOL MultiPhysics commercial software and the Newman's DUALFOIL code, and (*iv*) some demonstrative simulations involving thermal dynamics, a hybrid charge-discharge cycle emulating the throttle of an Hybrid Electric Vehicle (HEV), and a battery pack simulation.

The material presented in this chapter has appeared in:

- [Torchio et al., 2016c] - Torchio, M., Magni, L., Gopaluni, R. B., Braatz, R. D., and Raimondo, D. M. (2016c). LIONSIMBA: A Matlab framework based on a finite volume model suitable for Li-ion battery design, simulation, and control. *Journal of The Electrochemical Society*, 163(7):A1192–A1205

- **Chapter 3**: This chapter presents the Model Predictive Control paradigm together with the linearization and order reduction techniques used in this Thesis for developing the different ABMSs. Finally, the identification approaches adopted in this Thesis are discussed. In particular, two novel identification techniques are proposed for the development of piecewise affine models.

- **Chapter 4**: This chapter presents the main results obtained by us-

ing the MPC paradigm for ABMSs. In particular, different prediction models and control objectives are evaluated. The methodology introduced in chapter 3 is used for developing the proposed control strategies. A quadratic dynamic matrix control approach is used to minimize the charge time, while taking temperature and voltage constraints into account. A reduced input-output model, constructed from the P2D dynamics, is used for control purposes. To further emphasize the power of predictive algorithms, a health-aware charging protocol is presented for reducing the damages induced to the cell during charging operations. With the aim of improving the control performance, piecewise affine approximations of the cell dynamics are presented to better capture the plant nonlinearities during prediction phases. Finally, linear time varying models are used for the development of MPC-based ABMSs with the aim of reducing the computational burden while still guaranteeing good control performance.
Some preliminary results shown in this chapter have been presented in:

- [Torchio et al., 2016a] - Torchio, M., Magni, L., Braatz, R. D., and Raimondo, D. M. (2016a). Optimal charging of a Li-ion cell: A hybrid model predictive control approach. In *Proceedings of the IEEE Conference on Decision and Control*, pages 4053–4058

- [Torchio et al., 2016b] - Torchio, M., Magni, L., Braatz, R. D., and Raimondo, D. M. (2016b). Optimal health-aware charging protocol for Lithium-ion batteries: A fast model predictive

control approach. In *Proceedings of the 11th International Symposium on Dynamics and Control of Process Systems, including Biosystems*, pages 827–832

– [Torchio et al., 2015] - Torchio, M., Wolff, N. A., Raimondo, D. M., Magni, L., Krewer, U., Gopaluni, R. B., Paulson, J. A., and Braatz, R. D. (2015). Real-time model predictive control for the optimal charging of a lithium-ion battery. In *Proceedings of the American Control Conference*, pages 4536–4541

Chapter 2

Batteries

Contents

2.1 Introduction

Electrochemical accumulators, better known as batteries, are particular devices that can be used both for storing energy and for delivering it when necessary. Even though nowadays such devices are employed to provide

suitable power supplies to a wide range of applications, their discovery dates back to ancient times. Indeed, the history of batteries goes back to 250 BC when terracotta containers, used together with an iron bar and a copper cylinder immersed within an organic acid solution, were used as a galvanic cell (see Fig. 2.1) [Scrosati, 2011, Rolison and Nazar, 2011]. In modern

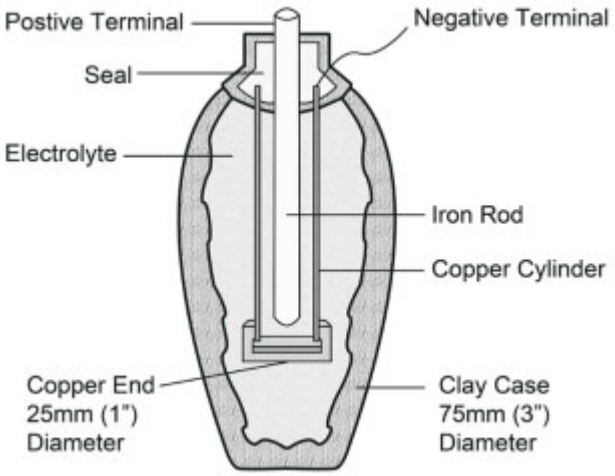


Figure 2.1: Example of the so-called Baghdad battery.

times, the first scientist who built an actual battery was Alessandro Volta. Starting from Luigi Galvani experiments, in 1800 Volta demonstrated that the electrical current can flow between two metals having an electrolyte in between. These experiments gave birth to the first pile (see Fig. 2.2) [Newman and Thomas-Alyea, 2012]. In the following years, based on Volta's works, other experiments have been performed in order to investigate the influence of different materials compositions over the pile performance. In fact, the main limitations related to the Volta's pile were related to its inability to provide current for long periods of time. In 1836 the British researcher John Frederich Daniell, through a particular arrangement of ma-

Figure 2.2: Volta's pile. Courtesy of Wikimedia

terials used to build the Daniell Cell, was able to overcome such limitations. The French Gaston Planté, in 1859, produced the first lead-acid rechargeable battery by using thin lead layers and diluted sulfuric acid [Kurzweil, 2010]. Later on, between 1895 and 1905, the Swedish Waldemar Jungner and the American Thomas Edison proposed new chemistries for rechargeable batteries based on alkaline electrolyte solutions [Nakahara, 2006]. Because of their alkaline composition, a wide range of materials were suitable to build nickel-cadmium (NiCd) and nickel-iron batteries as opposed to the lead-acid ones. In the recent years, with the advances in the development of portable electronics and their employment in a wide range of industrial applications, new batteries chemistries have been developed. In particular, in 1990 nickel-metalhydride (NiMh) and in 1991 Lithium-ion (Li-ion) batteries were commercialized with the aim of supporting the emerging market of portable devices. Through the years, such chemistries have been found suitable also for providing power in the automotive field. Indeed, even though the idea of equipping cars with electric powered engines goes back

to the early 1900, the lack of reliable and efficient EAs has de facto postponed their large scale commercialization [Kirsch, 2000]. With the arrival of Li-ion and NiMh batteries, the market shares of Electric Vehicle (EV) and HEV has significantly increased, with growth prospects for the next years [Randall, 2016]. Nowadays, the research in the development of new chemistries is moving towards the direction of organic batteries whose objective is to minimize the environmental impact, while still providing efficient and reliable devices [Chen et al., 2008, Lee et al., 2016, Yao, 2016].

2.2 Primary and secondary batteries

Among the different specifications of a battery, a key feature is related to its capability to be recharged or not. This property makes that battery belonging to one of these two categories: primary or secondary batteries. In particular, primary batteries are non-rechargeable devices, as opposed to the secondary batteries that can be discharged and charged multiple times. Examples of primary batteries are the zinc-carbon, zinc-alkaline and some Li-ion batteries, while examples of secondary batteries are the lead-acid, some Li-ion, Li-ion-polymer, and NiMh EAs [Linden, 1984].

2.2.1 Batteries keywords

In the following paragraph, a brief explanation of the most common terms used when working with batteries is given. The same terms will be used throughout the Thesis.

- **Cell:** a cell is the elementary unit of a battery, which is composed of two electrodes, an electrolyte solution and the current collectors (see

Fig. 2.3a). From a cell it is possible to convert chemical energy into electrical one and viceversa (only if secondary). Sometimes the term *battery* is also used to refer to a single cell.

- **Battery pack:** a battery pack refers to a set of connected cells (see Fig. 2.3b). The series or parallel interconnection of cells provides a battery pack with an higher voltage or capacity. A battery pack is usually entirely composed of primary or secondary cells, and not a mixture of them.

- **Anode:** the anode is the electrode which is subject to oxidation reaction (electrons flows out from the electrode to an external circuit) during the cell operating conditions. Commonly the anode is used to identify the negative electrode.

- **Cathode:** the cathode is the electrode which is subject to reduction reaction (electrons flows into the electrode taken from an external circuit) during the cell operating conditions. Commonly the cathode is used to identify the positive electrode.

- **Electrolyte:** the electrolyte is a transport medium which allows ions to flow between the two electrodes. It can be both liquid or a solid.

- **Separator:** the separator is an insulating porous layer which avoids possible short-circuits between the electrodes, while allowing ions to flow through it.

- **C-rate:** the C-rate is a relative value which represents the amount of current needed to fully discharge a cell (starting from a fully charged condition) within an hour. It equalizes the rated capacity (usually

expressed in Ah) in terms of Ampere. For example, a 600 mAh cell has a C-rate (1C) of 600 mA.

- **Cell voltage:** the cell voltage measures the difference between the electrodes potential under generic load conditions.

- **Open Circuit Voltage:** the Open Circuit Voltage (OCV) represents the voltage drop between the electrodes of a cell that is not under load conditions.

- **Cutoff Voltage:** when the OCV equals the cutoff voltage, the cell is fully discharged. This value varies according to the battery composition.

- **State Of Health:** the State Of Health (SOH) is an index used to represent the ability of a battery to deliver specified performance, compared to the ones coming from a fresh battery of the same type. Given that several phenomena can concur to degrade the battery performance, an index able to account for such losses can be useful in monitoring operating conditions and diagnosing possible faulty situations.

- **Usage cycles:** the usage cycles count the number of times that a battery can be charged and discharged, before the SOH index drops below or above a threshold value. Usually the exceeding of such a value determines the inability of the battery to provide suitable operating performance.

- **State of Charge:** the SOC index quantifies the amount of charge available inside an electrode. It is usually represented by means of

the ratio between the actual concentration of ions and the maximum allowed capacity inside an electrode. This index is commonly used to account for the remaining charge of a cell.

(a) A cylindrical 18650 cell.

(b) A battery pack

Figure 2.3: Li-ion batteries configurations

2.2.2 Li-ion batteries

Li-ion based batteries were commercialized in 1991 by the Japanese company Sony [Nishi, 2001]. With respect to the other available chemistries, Li-ion based cells exhibit an higher operating voltage (around 3.6 V), a higher specific energy (expressed in Wh/kg), a lower self-discharge, and a higher number of operating cycles (see Fig. 2.4) [Linden, 1984]. The electrodes used to build a Li-ion cell have a lattice structure. Such structure supports multiple insertion and extraction of ions, without being significantly degraded or compromised. The basic working principle of a Li-ion cell is described according to the so-called *"rocking chair"* rule [Guyomard and Tarascon, 1994]. In charging conditions, the ions will deintercalate from the cathode, and flowing within the electrolyte solution, will intercalate inside the anode. The opposite process occurs during discharging conditions. The materials used for the positive electrode are usually lithium cobalt ox-

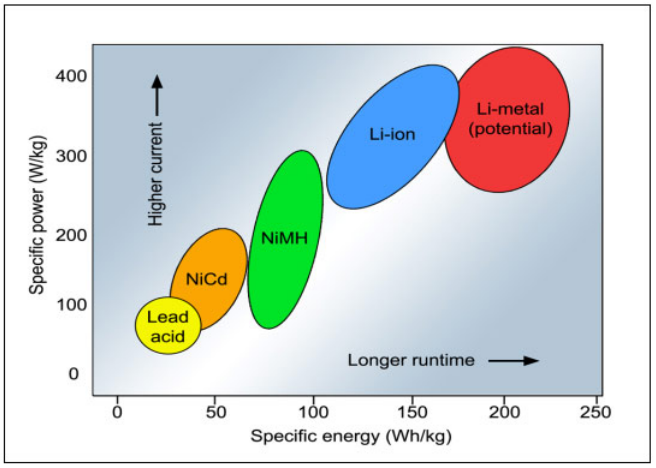


Figure 2.4: Comparison of different EAs chemistries

ide ($LiCoO_2$), lithium nickel oxide ($LiNiO_2$) or lithium manganese oxide ($LiMn_2O_4$). For the anode side, a carbon electrode made from graphite or petroleum coke is usually adopted. The combination of two different materials composing the cathode and anode gives birth to chemistries having different capacities, discharge curves shapes, cutoff voltages etc. The electrolyte is usually composed of non-aqueous solutions, where salts are dissolved within organic or inorganic solvents. The choice of the particular solvent is made according to the anodic material (coke or graphite), while a common choice for the salt falls on the lithium hexafluorophosphate ($LiPF_6$).

Li-ion-Polymer (Li-Poly – LiPo) batteries have also been proposed in 1997. The main difference with respect to the Li-ion batteries lies in the adoption of a solid-state electrolyte instead of a liquid [Meyer, 1998]. In this case, the polymer electrolyte replaces also the separator layer which is usually present in liquid-based Li-ion batteries. The main benefit of

adopting a solid-state electrolyte is related to a higher safety of the battery during operating conditions. Nevertheless, such increased safety is gained at the cost of a lower ionic conductivity which prevents the Li-Poly cells to perform as well as the Li-ion batteries [Linden, 1984].

Another development in Li-ion batteries is represented by the Li-ion metal (Li-metal) cells. In this case the anode is replaced by a metallic lithium electrode. The main advantages by making use of a metallic lithium anode is that both the energy density and specific energy are significantly increased. However, Li-metal cells are more susceptible to the loss of cyclable lithium due to the formation of a passivation layer after each charge/discharge cycle. This phenomenon is mainly due to the high reactivity of the Li-metal anode with any liquid electrolyte [Tarascon and Armand, 2001]. In order to reduce such loss, polymer electrolytes have been employed. However, even though the adoption of polymer electrolytes reduces the loss of cyclable lithium, the formation of surface irregularities (called dendrites) has been demonstrated to lead to unwanted short-circuits, thus compromising safety.

2.2.3 Batteries working principle

Secondary batteries are able to convert chemical energy into electrical energy during discharging operations, and viceversa during charging conditions. The conversion takes place with the occurrence of two main electrochemical phenomena at the electrodes-electrolyte interfaces: the reduction-oxidation (*redox*) reactions. The oxidation reaction extracts electrons from the electrode and yields them to an external circuit, while the reduction reaction takes electrons from the external circuit and feed them into the

electrode. It is worth noticing that these two reactions can involve both the positive and negative electrodes of a cell. Indeed, when discharging a cell, the negative electrode will be subject to oxidation, while the positive one to reduction. The opposite situation takes place when charging the cell. Therefore, according to the definition of anode and cathode given in Section 2.2.1, during charging conditions the anode would be the positive electrode (oxidation) of the cell whereas the cathode would be the negative one (reduction). Viceversa, in discharging conditions, the anode would be the negative electrode (oxidation), while the cathode would be the positive one (reduction). Conventionally, however, the anode refers to the negative pole, while the cathode to the positive one according to the discharging operating condition of the cell.

2.3 Mathematical models for Li-ion batteries

Mathematical models can be exploited in order to improve the manufacturing and management processes of Li-ion batteries. The optimal design of a battery plays a fundamental role in providing ever more reliable and efficient Li-ion accumulators for industrial applications. This process generally requires a large amount of time and involves the execution of numerous experimental tests. The adoption of a mathematical model can significantly reduce time and substantial savings of money. Such models can be also exploited for the development of ABMSs with the aim of improving the operating performance and to prolong the lifetime of Li-ion cells [Santhanagopalan et al., 2006, Ramadesigan et al., 2012, Torchio et al., 2015]. In the literature, models have been developed for describing the behavior of a Li-ion battery. All of these models belong to one of these two families:

(i) the equivalent circuit models, and (ii) the electrochemical models. In the following a brief overview of the models available in literature is given.

2.3.1 Equivalent circuit models

ECMs make use only of electrical components and circuits to model the dynamic behavior of a battery. Usually, resistances, capacitors, and inductances are used for these purposes. The R_{int} model (shown in Fig. 2.5) implements an ideal voltage source U_{OCP} in series with a resistance (called *internal resistance*) R_{int}. The current provided by the battery is i_{batt}, while V_{batt} represents the cell voltage. An improvement of the R_{int} was proposed

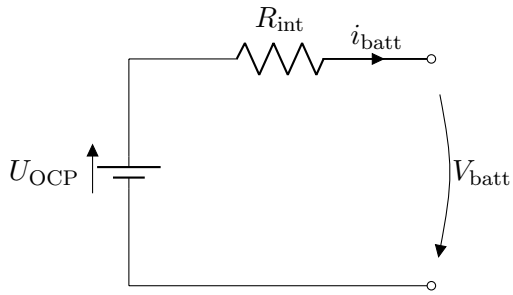


Figure 2.5: The R_{int} model

by the company SAFT [Johnson et al., 2001]. The RC model (as it was named), besides considering the presence of resistors, makes use also of capacitors. As shown in Fig. 2.6, two capacitors (C_b and C_c) with three resistors (R_{e}, R_{t}, and R_{C}) are used to define the overall model. In particular the capacitor C_c has a small capacitance and it is mainly used to model the surface effects of the battery (it takes the name of surface capacitor). On the other hand, the capacitor C_b is used to account for the battery capacity (it takes the name of bulk capacitor). The indices U_b and

U_c are used to represent the potential of the bulk and surface capacitors respectively. A further extension of the R_{int} model is given by the Thevenin

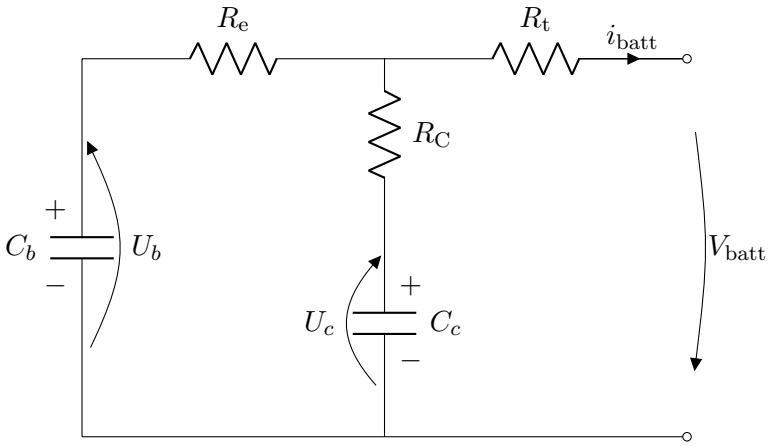


Figure 2.6: The RC model

model, where a parallel RC network is connected in series to the internal resistance as shown in Fig. 2.7. The presence of the capacitance C_{Th} is

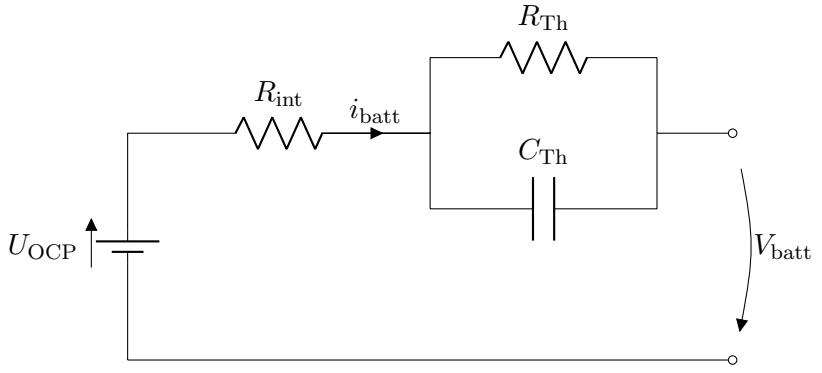


Figure 2.7: The Thevenin model

used to describe the transient behavior of the cell during operating condi-

tions. Other extensions of the R_{int} model are the Partnership for a New Generation of Vehicles (PNGV) and Dual Polarization (DP) models (see [He et al., 2011, Hu et al., 2012] and the references therein for a thorough description). Other ECMs try to reproduce in more details the behavior of a Li-ion cell by modeling separately the dynamics of the electrodes and the electrolyte by means of resistors, capacitors and inductances as shown in [Bergveld et al., 2002].

2.3.2 Electrochemical models

In contrast to ECMs, EMs explicitly represent in detail the electrochemical phenomena occurring inside a cell [Rahimian et al., 2011]. The most widely used EM in the literature is the porous electrodes theory-based pseudo two-dimensional model [Doyle, 1995]. The P2D model is represented by means of Partial Differential and Algebraic Equations (PDAEs), derived from first-principles laws of modeling. Such equations are used to describe the conservation of mass and charge in the different sections of the battery. Despite the high accuracy provided by the P2D model in predicting the electrochemical dynamics, the numerical resolution of the PDAEs requires a significant computational effort. This aspect represents the main drawback of this EM. For this reason, several approximations have been proposed in literature. Some of them aim to reduce the computational burden by approximating the electrodes kinetics with polynomials or Ordinary Differential Equations (ODEs) [Ramadesigan et al., 2010], while other approaches make simplifying assumptions related to the electrodes kinetics and the liquid phase dynamics. The Single Particle Model (SPM), for instance, is a simplified version of the P2D model, which has been demonstrated to be

accurate for simulating low to medium current densities [Santhanagopalan et al., 2006].

In the remainder of this chapter, the P2D model is considered and its numerical implementation carefully treated. In order to exploit the model for simulation and design purposes, the set of PDAEs are reformulated as a set of ordinary Differential-Algebraic Equations (DAEs). The model reformulation is very challenging to carry out in a way that is simultaneously computationally efficient and numerically stable for a wide range of battery parameters and operating conditions. A detailed description of a computationally efficient and numerically stable finite volume DAE formulation is presented, while also addressing potential pitfalls and relative loopholes. In particular boundary conditions used to enforce physical meaningfulness of the system are thoroughly discussed and their numerical implementation is explained. Particular attention is directed to the handling of interface boundary conditions in the three primary sections of the battery - positive and negative electrodes and the separator. Due to possible discontinuities between adjacent sections, a mishandling of such conditions may lead to physically inconsistent solutions. Due to its intrinsic properties, the Finite Volume Method (FVM) has been chosen to easily deal with these particular interface conditions. Finally, based on the proposed FVM discretization, the LIONSIMBA toolbox is provided, which is a fully customizable Matlab® framework suitable for simulating the dynamic behavior of Li-ion batteries. These functions are freely downloadable from the website

`http://sisdin.unipv.it/labsisdin/lionsimba.php`

The reader can implement his/her own custom-defined control algorithm to test different ABMS strategies, simulate cell behavior, optimize

manufacturing parameters or test battery packs composed of series-connected cells. The package also allows the ready implementation of algorithms to estimate indices such as the SOC and the SOH. The package comes with the experimental parameters of the battery reported in [Northrop et al., 2011]. An initialization file allows changes in battery and simulator parameters. The simulator works under Matlab® using IDA [Hindmarsh et al., 2005] to solve the set of resulting DAEs with a good tradeoff between accuracy and computational time.

2.4 Pseudo two-dimensional model

The P2D model consists of coupled nonlinear PDAEs for the conservation of mass and charge in the three sections of the battery – cathode, separator, and anode – denoted respectively by the indices p, s, and n. The positive and negative current collectors are denoted by a and z. A cross sectional representation of a Li-ion cell is depicted in Fig. 2.8. The electrodes-separator structure is immersed within an electrolyte solution which is used as a transport medium from the lithium ions. In the following, the index $i \in \mathcal{S}$ is used to refer to a particular section of the battery, where $\mathcal{S} := \{a, p, s, n, z\}$. Each section of the battery has a thickness represented by l_i, while the overall thickness is obtained by $L = \sum_i l_i$, and the notation $\hat{x}_0 = l_a$, $\hat{x}_p = l_a + l_p$, $\hat{x}_s = l_a + l_p + l_s$, and $\hat{x}_n = l_a + l_p + l_s + l_n$ is used. In order to take into account the properties of different materials used in the battery, effective diffusion and conductivity coefficients are evaluated according to the Bruggeman's theory, with "eff" suffixes representing effective values of such coefficients. Moreover, with the aim of providing a more detailed description of the conductivity and diffusion phenomena inside the

electrolyte, all the related coefficients are determined as a function of c_e and T, as discussed in [Valøen and Reimers, 2005]. For a clearer comprehension, **bold** is used in tables for coefficients whose dependence on other variables is made explicit in other equations. The complete set of equations are summarized in the Tables 2.2 and 2.3, while the nomenclature of the variables is reported in Table 2.1.

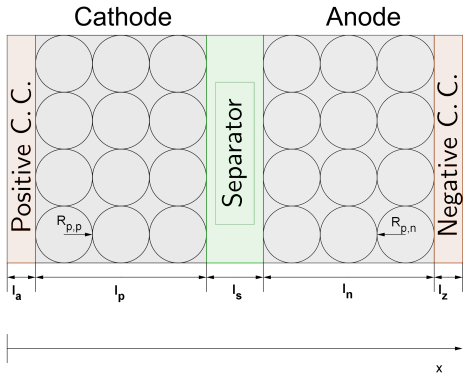


Figure 2.8: Schematic cross sectional view of a Li-ion cell

The flow of ions inside the electrolyte solution is modeled by the diffusion equation

$$\epsilon_i \frac{\partial}{\partial t} c_e(x,t) = \frac{\partial}{\partial x}\left[\boldsymbol{D}_{\mathbf{eff},i} \frac{\partial c_e(x,t)}{\partial x}\right] + a_i(1-t_+)j(x,t), \quad i \in \{p,s,n\} \tag{2.1}$$

where $x \in \mathbb{R}$ is the one-dimensional spatial direction along which the ions are transported, $t \in \mathbb{R}^+$ represents the time, $c_e(x,t)$ accounts for the electrolyte concentration of ions, the function $j(x,t)$ is the ionic flux, t_+ defines the transference number, while in the ith section a_i is the particle surface area to volume ratio, $\boldsymbol{D}_{\mathbf{eff},i}$ accounts for the effective diffusion coefficients of the electrolyte, and ϵ_i represents the material porosity. Given that ions do

not flow outside the electrodes, zero-flux boundary conditions are enforced

$$\left.\frac{\partial c_e(x,t)}{\partial x}\right|_{x=\hat{x}_0} = 0, \qquad \left.\frac{\partial c_e(x,t)}{\partial x}\right|_{x=\hat{x}_n} = 0,$$

while continuity conditions are considered across the different sections of the battery

$$\begin{aligned} -\boldsymbol{D}_{\mathbf{eff},p}\left.\frac{\partial c_e(x,t)}{\partial x}\right|_{x=\hat{x}_p^-} &= -\boldsymbol{D}_{\mathbf{eff},s}\left.\frac{\partial c_e(x,t)}{\partial x}\right|_{x=\hat{x}_p^+}, \\ -\boldsymbol{D}_{\mathbf{eff},s}\left.\frac{\partial c_e(x,t)}{\partial x}\right|_{x=\hat{x}_s^-} &= -\boldsymbol{D}_{\mathbf{eff},n}\left.\frac{\partial c_e(x,t)}{\partial x}\right|_{x=\hat{x}_s^+}. \end{aligned}$$

To model the *intercalation* and *deintercalation* processes inside the solid spherical particles with radius $R_{p,i}$, the Fick's law of diffusion is used

$$\frac{\partial c_s(r,t)}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\left[r^2\, \boldsymbol{D}^{\mathrm{s}}_{\mathbf{eff},i}\, \frac{\partial c_s(r,t)}{\partial r}\right], \quad i \in \{p,n\} \tag{2.2}$$

with boundary conditions

$$\left.\frac{\partial c_s(r,t)}{\partial r}\right|_{r=0} = 0, \qquad \left.\frac{\partial c_s(r,t)}{\partial r}\right|_{r=R_{p,i}} = -\frac{j(x,t)}{\boldsymbol{D}^{\mathrm{s}}_{\mathbf{eff},i}},$$

where $r \in \mathbb{R}$ is the radial direction along which ions diffuse within the solid particles, $c_s(r,t)$ is the solid phase ions concentration, while $\boldsymbol{D}^{s}_{\mathbf{eff},i}$ accounts for the effective diffusion coefficients of the solid phases. This model introduces a pseudo-second dimension (r). To reduce complexity and computational burden, [Subramanian et al., 2005] and [Ramadesigan et al., 2010] proposed different efficient reformulations for the solid-phase diffusion equation. As discussed in [Zhang and White, 2007], according to the particular application, different model reformulations can be employed while maintaining good accuracy. For low to medium C rates, the diffusion length method [Wang et al., 1998] or the two-term polynomial approximation method are accurate. At high C rates, higher-order polynomial

approximations or the Pseudo Steady State (PSS) [Liu, 2006] approximation can be employed. For more details, refer to [Zhang and White, 2007] and the references therein.

In the two-term polynomial approximation, concentration profiles inside the particle are assumed to be quadratic in r and (2.2) is approximated by means of average $c_s^{\text{avg}}(x,t)$ and surface $c_s^*(x,t)$ concentration of the solid particles,

$$\frac{\partial c_s^{\text{avg}}(x,t)}{\partial t} = -3\frac{j(x,t)}{R_{p,i}}, \tag{2.3}$$

$$c_s^*(x,t) - c_s^{\text{avg}}(x,t) = -\frac{R_{p,i}}{\boldsymbol{D_{\text{eff},i}^s}}\frac{j(x,t)}{5}, \quad i \in \{p,n\}. \tag{2.4}$$

This reformulation leads to a one-dimensional problem in x by removing the pseudo-second dimension r. Despite the reduced computational burden, such approximation could lead to a decrease of the prediction accuracy for high rates, short time responses or pulse currents [Zhang and White, 2007]. For these applications, higher-order polynomials or Fick's law are recommended, as discussed in Section 2.6.2.

According to Ohm's law, the conservation of charge in the electrodes can be defined as

$$\frac{\partial}{\partial x}\left[\boldsymbol{\sigma}_{\text{eff},i}\frac{\partial}{\partial x}\Phi_s(x,t)\right] = a_i F j(x,t), \quad i \in \{p,n\} \tag{2.5}$$

where $\Phi_s(x,t)$ is the solid potential, $\boldsymbol{\sigma}_{\text{eff},i}$ is the electrodes effective conductivity, while F is the Faraday's constant. In order to relate the solid phase potential with the applied current density $I_{\text{app}}(t)$, the following boundary

conditions are enforced

$$\boldsymbol{\sigma}_{\mathbf{eff},i}\frac{\partial\Phi_s(x,t)}{\partial x}\bigg|_{x=\hat{x}_0,\hat{x}_n} = -I_{\text{app}}(t),$$
$$\boldsymbol{\sigma}_{\mathbf{eff},i}\frac{\partial\Phi_s(x,t)}{\partial x}\bigg|_{x=\hat{x}_p,\hat{x}_s} = 0.$$

The potential of the Li-ion cell is obtained as

$$V_{\text{out}}(t) := \Phi_s(\hat{x}_p,t) - \Phi_s(\hat{x}_n,t).$$

Similarly, a modified Ohm's law is used to represent the charge conservation within the electrolyte:

$$\begin{aligned} a_i F j(x,t) = &-\frac{\partial}{\partial x}\left[\boldsymbol{\kappa}_{\mathbf{eff},i}\frac{\partial}{\partial x}\Phi_e(x,t)\right] \\ &+\frac{\partial}{\partial x}\left[\frac{2\boldsymbol{\kappa}_{\mathbf{eff},i}\mathrm{R}T(x,t)}{F}(1-t_+)\frac{\partial}{\partial x}\ln c_e(x,t)\right], \quad i\in\{p,s,n\} \end{aligned} \tag{2.6}$$

where $\Phi_e(x,t)$ is the electrolyte potential, $T(x,t)$ represents the temperature, R defines the universal gas constant, and $\boldsymbol{\kappa}_{\mathbf{eff},i}$ is the effective conductivity coefficient of the liquid phase in the ith section. Without loss of generality, given that only potential differences can be measured, the following boundary condition is enforced at the anode side

$$\Phi_e(x,t)\Big|_{x=\hat{x}_n} = 0,$$

while null flux conditions are enforced at the cathode side

$$\frac{\partial\Phi_e(x,t)}{\partial x}\bigg|_{x=\hat{x}_0} = 0.$$

Excessive heat generation may lead to performance degradation and, in extreme cases, thermal runaway of the cell [Bernardi et al., 1985, Bandhauer et al., 2011]. In order to address these possible safety issues, thermal dynamics are included with the set of conservation equations describing the

system in each section of the battery

$$\rho_i C_{p,i} \frac{\partial}{\partial t} T(x,t) = \frac{\partial}{\partial x}\left[\lambda_i \frac{\partial}{\partial x} T(x,t)\right] + Q_{\text{ohm},i}(x,t) + Q_{\text{rxn},i}(x,t) + Q_{\text{rev},i}(x,t), \tag{2.7}$$

where ρ_i is the material density, $C_{p,i}$ is the specific heat, λ_i is the heat diffusion coefficient. The ohmic generation rate takes into account heat generated as a consequence of the motion of Li-ions in the solid/liquid phase, and it varies according to the particular section of the battery. Inside the electrodes it is defined as

$$\begin{aligned} Q_{\text{ohm},i}(x,t) =& \boldsymbol{\sigma}_{\mathbf{eff},i}\left(\frac{\partial \Phi_s(x,t)}{\partial x}\right)^2 + \boldsymbol{\kappa}_{\mathbf{eff},i}\left(\frac{\partial \Phi_e(x,t)}{\partial x}\right)^2 \\ &+ \frac{2\boldsymbol{\kappa}_{\mathbf{eff},i}\mathrm{R}T(x,t)}{F}(1-t_+)\frac{\partial \ln c_e(x,t)}{\partial x}\frac{\partial \Phi_e(x,t)}{\partial x}, \ i \in \{p,n\}. \end{aligned}$$

In the liquid phase, due to the absence of the solid potential, its formulation is stated as follows

$$\begin{aligned} Q_{\text{ohm},i}(x,t) =& \boldsymbol{\kappa}_{\mathbf{eff},i}\left(\frac{\partial \Phi_e(x,t)}{\partial x}\right)^2 \\ &+ \frac{2\boldsymbol{\kappa}_{\mathbf{eff},i}\mathrm{R}T(x,t)}{F}(1-t_+)\frac{\partial \ln c_e(x,t)}{\partial x}\frac{\partial \Phi_e(x,t)}{\partial x}, \ i \in \{s\}, \end{aligned}$$

while in the current collectors, only Joule's effect is considered

$$Q_{\text{ohm},i}(x,t) = \frac{I^2_{\text{app}}(t)}{\boldsymbol{\sigma}_{\mathbf{eff},i}} \quad i \in \{a,z\}.$$

The reaction generation rate, present only inside the electrodes, is defined as

$$Q_{\text{rxn},i}(x,t) = F\, a_i\, j(x,t)\, \eta_i(x,t), \quad i \in \{p,n\}$$

and accounts for heat generated due to ionic flux and electrodes overpotential

$$\eta_i(x,t) = \Phi_s(x,t) - \Phi_e(x,t) - \boldsymbol{U}_i(x,t), \quad i \in \{p,n\}$$

where $\boldsymbol{U}_i(x,t)$ represents the OCV. Finally, the reversible generation rate

$$Q_{\mathrm{rev},i}(x,t) = F\, a_i\, j(x,t)\, T(x,t)\, \frac{\boldsymbol{\partial U}_i}{\boldsymbol{\partial} T}\bigg|_{T_{\mathrm{ref}}}, \quad i \in \{p,n\}$$

takes into account the heat generated by the entropy change in the electrodes structure [Kumaresan et al., 2008], where $\frac{\boldsymbol{\partial U}_i}{\boldsymbol{\partial} T}\big|_{T_{\mathrm{ref}}}$ accounts for the entropic variation of the OCV evaluated with respect to a reference temperature T_{ref}. Boundary conditions for the thermal dynamics are needed in order to: (i) guarantee continuity across the different sections of the battery

$$\begin{aligned}
-\lambda_a \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_0^-} &= -\lambda_p \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_0^+},\\
-\lambda_p \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_p^-} &= -\lambda_s \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_p^+},\\
-\lambda_s \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_s^-} &= -\lambda_n \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_s^+},\\
-\lambda_n \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_n^-} &= -\lambda_z \frac{\partial T(x,t)}{\partial x}\bigg|_{x=\hat{x}_n^+},
\end{aligned}$$

and (ii) enforce the Newton's law of cooling for heat dissipation with the surroundings

$$\begin{aligned}
-\lambda_a \frac{\partial T(x,t)}{\partial x}\bigg|_{x=0} &= h(T_{\mathrm{env}} - T(x,t)),\\
-\lambda_z \frac{\partial T(x,t)}{\partial x}\bigg|_{x=L} &= h(T(x,t) - T_{\mathrm{env}}).
\end{aligned}$$

The term h represents the heat exchange coefficient, while T_{env} represents the environmental temperature.

The above equations are coupled by means of the ionic flux, which is defined as

$$j(x,t) = 2\frac{i_0}{F}\sinh\left[\frac{0.5F}{\mathrm{R}T(x,t)}\eta_i(x,t)\right], \quad i \in \{p,n\} \tag{2.8}$$

where $c_{s,i}^{\max}$ is the maximum concentration of the solid phase, while the exchange current density is given by

$$i_0 = F\boldsymbol{k}_{\mathbf{eff},i}\sqrt{c_e(x,t)(c_{s,i}^{\max} - c_s(R_{p,i},t))c_s(R_{p,i},t)}.$$

The ionic flux $j(x,t)$ is zero inside the separator. The model equations are from [Doyle, 1995], where for convenience the electrolyte potential is related to the ionic flux $j(x,t)$ rather than to the applied current density [Smith and Wang, 2006, Bernardi and Go, 2011]. The thermal model is taken from [Kumaresan et al., 2008] while all of the parameters describing the particular chemistry are taken from [Northrop et al., 2011].

$I_{\text{app}}(t)$	Applied current density [A/m^2]
$c_e(x,t)$	Electrolyte salt concentration [mol/m^3]
$c_s(r,t)$	Solid-phase concentration [mol/m^3]
$c_s^{\text{avg}}(x,t)$	Solid-phase average concentration [mol/m^3]
$c_s^*(x,t)$	Solid-phase surface concentration [mol/m^3]
$j(x,t)$	Ionic flux [mol/(m^{2}s)]
$\Phi_e(x,t)$	Electrolyte potential [V]
$\Phi_s(x,t)$	Solid potential [V]
$\eta(x,t)$	Electrode overpotential [V]
$U(x,t)$	Open circuit voltage [V]
$T(x,t)$	Temperature [K]
$\boldsymbol{D_{\text{eff}}^s}$	Effective solid-phase diffusion coefficient [m^2/s]
$\boldsymbol{D_{\text{eff}}}$	Effective electrolyte diffusion coefficient [m^2/s]
$\boldsymbol{\sigma_{\text{eff}}}$	Effective solid-phase conductivity [S/m]
$\boldsymbol{\kappa_{\text{eff}}}$	Effective electrolyte conductivity [S/m]
$\boldsymbol{k_{\text{eff}}}$	Effective reaction rate
Q_{ohm}	Ohmic heat source term [W/m^3]
Q_{rev}	Reversible heat source term [W/m^3]
Q_{rxn}	Reaction heat source term [W/m^3]
$\boldsymbol{U_{\text{ref}}}$	Open Circuit Voltage [V]
$\left.\dfrac{\partial \boldsymbol{U}}{\partial \boldsymbol{T}}\right\vert_{\boldsymbol{T}_{\text{ref}}}$	Open Circuit Voltage Entropic Variation [V/K]
l	Thickness [m]
R_p	Particles radius [m]
ρ	Density [kg/m^3]
C_p	Specific Heat [J/(kg K)]

h	Heat diffusion coefficient [W/(m^2 K)]
λ	Thermal conductivity [W/(m K)]
ϵ	Porosity
t_+	Transference number
$E_a^{D^s}$	Solid-phase diffusion activation energy [J/mol]
E_a^k	Reaction constant activation energy [J/mol]
a	Particle surface area to volume [m^2/m^3]
F	Faraday's constant [C/mol]
R	Gas universal constant [J/(mol K)]
ϵ_f	Filler fraction

Table 2.1: Nomenclature

Current Collectors, $i \in \{a, z\}$	**Boundary Conditions**
$\rho_i C_{p,i} \dfrac{\partial T(x,t)}{\partial t} = \dfrac{\partial}{\partial x}\left[\lambda_i \dfrac{\partial T(x,t)}{\partial x}\right] + \dfrac{I_{\text{app}}^2(t)}{\boldsymbol{\sigma_{\text{eff},i}}}$	$-\lambda_a \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=0} = h(T_{\text{env}} - T(x,t))$ $-\lambda_z \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=L} = h(T(x,t) - T_{\text{env}})$
Positive and Negative Electrodes, $i \in \{p, n\}$	
$\epsilon_i \dfrac{\partial c_e(x,t)}{\partial t} = \dfrac{\partial}{\partial x}\left[\boldsymbol{D_{\text{eff},i}} \dfrac{\partial c_e(x,t)}{\partial x}\right] + a_i(1-t_+)j(x,t)$	$\dfrac{\partial c_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_0,\hat{x}_n} = 0$
$\dfrac{\partial c_s^{\text{avg}}(x,t)}{\partial t} = -3\dfrac{j(x,t)}{R_{p,i}}$	
$c_s^*(x,t) - c_s^{\text{avg}}(x,t) = -\dfrac{R_{p,i}}{\boldsymbol{D_{\text{eff},i}^{\text{s}}}} \dfrac{j(x,t)}{5}$	
$\dfrac{\partial}{\partial x}\left[\boldsymbol{\sigma_{\text{eff},i}} \dfrac{\partial \Phi_s(x,t)}{\partial x}\right] = a_i F\, j(x,t)$	$\boldsymbol{\sigma_{\text{eff},i}} \dfrac{\partial \Phi_s(x,t)}{\partial x}\Big\|_{x=\hat{x}_0,\hat{x}_n} = -I_{\text{app}}(t)$ $\boldsymbol{\sigma_{\text{eff},i}} \dfrac{\partial \Phi_s(x,t)}{\partial x}\Big\|_{x=\hat{x}_p,\hat{x}_s} = 0$
$a_i F\, j(x,t) = -\dfrac{\partial}{\partial x}\left[\boldsymbol{\kappa_{\text{eff},i}} \dfrac{\partial \Phi_e(x,t)}{\partial x}\right] + \dfrac{\partial}{\partial x}\left[\boldsymbol{\kappa_{\text{eff},i}} \Upsilon T(x,t) \dfrac{\partial \ln c_e(x,t)}{\partial x}\right]$	$\dfrac{\partial \Phi_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_0} = 0$ $\Phi_e(x,t)\Big\|_{x=\hat{x}_n} = 0$
$\rho_i C_{p,i} \dfrac{\partial T(x,t)}{\partial t} = \dfrac{\partial}{\partial x}\left[\lambda_i \dfrac{\partial T(x,t)}{\partial x}\right] + Q_{\text{ohm},i}(x,t) + Q_{\text{rxn},i}(x,t) + Q_{\text{rev},i}(x,t)$	$-\lambda_z \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_0^-} = -\lambda_p \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_0^+}$ $-\lambda_n \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_n^-} = -\lambda_z \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_n^+}$
$j(x,t) = 2\boldsymbol{k_{\text{eff},i}} \sqrt{c_e(x,t)(c_{s,i}^{\max} - c_s^*(x,t))c_s^*(x,t)} \sinh\left[\frac{0.5R}{FT(x,t)}\eta_i(x,t)\right]$	
$\eta_i(x,t) = \Phi_s(x,t) - \Phi_e(x,t) - \mathbf{U}_i(x,t)$	
Separator, $i = s$	
$\epsilon_i \dfrac{\partial c_e(x,t)}{\partial t} = \dfrac{\partial}{\partial x}\left[\boldsymbol{D_{\text{eff},i}} \dfrac{\partial c_e(x,t)}{\partial x}\right]$	$-\boldsymbol{D_{\text{eff},p}} \dfrac{\partial c_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_p^-} = -\boldsymbol{D_{\text{eff},s}} \dfrac{\partial c_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_p^+}$ $-\boldsymbol{D_{\text{eff},s}} \dfrac{\partial c_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_s^-} = -\boldsymbol{D_{\text{eff},n}} \dfrac{\partial c_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_s^+}$
$0 = -\dfrac{\partial}{\partial x}\left[\boldsymbol{\kappa_{\text{eff},i}} \dfrac{\partial \Phi_e(x,t)}{\partial x}\right] + \dfrac{\partial}{\partial x}\left[\boldsymbol{\kappa_{\text{eff},i}} T(x,t) \Upsilon \dfrac{\partial \ln c_e(x,t)}{\partial x}\right]$	$-\boldsymbol{\kappa_{\text{eff},p}} \dfrac{\partial \Phi_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_p^-} = -\boldsymbol{\kappa_{\text{eff},s}} \dfrac{\partial \Phi_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_p^+}$ $-\boldsymbol{\kappa_{\text{eff},s}} \dfrac{\partial \Phi_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_s^-} = -\boldsymbol{\kappa_{\text{eff},n}} \dfrac{\partial \Phi_e(x,t)}{\partial x}\Big\|_{x=\hat{x}_s^+}$
$\rho_i C_{p,i} \dfrac{\partial T(x,t)}{\partial t} = \dfrac{\partial}{\partial x}\left[\lambda_i \dfrac{\partial T(x,t)}{\partial x}\right] + Q_{\text{ohm},i}(x,t)$	$-\lambda_p \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_p^-} = -\lambda_s \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_p^+}$ $-\lambda_s \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_s^-} = -\lambda_n \dfrac{\partial T(x,t)}{\partial x}\Big\|_{x=\hat{x}_s^+}$

Table 2.2: Li-ion P2D model governing equations

Open Circuit Voltage (Thermal dependence)

$$\boldsymbol{U}_p(x,t) = \boldsymbol{U}_{p,\mathbf{ref}} + (T(x,t) - T_{\text{ref}})\frac{\boldsymbol{\partial U}_p}{\boldsymbol{\partial T}}\bigg|_{\boldsymbol{T}_{\text{ref}}} \qquad \boldsymbol{U}_n(x,t) = \boldsymbol{U}_{n,\mathbf{ref}} + (T(x,t) - T_{\text{ref}})\frac{\boldsymbol{\partial U}_n}{\boldsymbol{\partial T}}\bigg|_{\boldsymbol{T}_{\text{ref}}}$$

Entropy change

$$\frac{\boldsymbol{\partial U}_p}{\boldsymbol{\partial T}}\bigg|_{\boldsymbol{T}_{\text{ref}}} = -0.001\left(\frac{0.199521039 - 0.928373822\boldsymbol{\theta}_p + 1.3645506890000003\boldsymbol{\theta}_p^2 - 0.61154489399999998\boldsymbol{\theta}_p^3}{1 - 5.6614798869999997\boldsymbol{\theta}_p + 11.476361910\boldsymbol{\theta}_p^2 - 9.8243121359999998\boldsymbol{\theta}_p^3 + 3.0487550636\boldsymbol{\theta}_p^4}\right)$$

$$\frac{\boldsymbol{\partial U}_n}{\boldsymbol{\partial T}}\bigg|_{\boldsymbol{T}_{\text{ref}}} = \frac{0.001\begin{pmatrix}0.005269056 + 3.299265709\boldsymbol{\theta}_n - 91.793257980\boldsymbol{\theta}_n^2 + 1004.9110080\boldsymbol{\theta}_n^3 - 5812.2781270\boldsymbol{\theta}_n^4 + \\ 19329.75490\boldsymbol{\theta}_n^5 - 37147.894760\boldsymbol{\theta}_n^6 + 38379.181270\boldsymbol{\theta}_n^7 - 16515.053080\boldsymbol{\theta}_n^8\end{pmatrix}}{\begin{pmatrix}1 - 48.092872270\boldsymbol{\theta}_n + 1017.2348040\boldsymbol{\theta}_n^2 - 10481.804190\boldsymbol{\theta}_n^3 + 59431.30\boldsymbol{\theta}_n^4 - \\ 195881.64880\boldsymbol{\theta}_n^5 + 374577.31520\boldsymbol{\theta}_n^6 - 385821.16070\boldsymbol{\theta}_n^7 + 165705.85970\boldsymbol{\theta}_n^8\end{pmatrix}}$$

Open Circuit Voltage (Reference value)

$$\boldsymbol{U}_{p,\mathbf{ref}} = \frac{-4.656 + 88.669\boldsymbol{\theta}_p^2 - 401.119\boldsymbol{\theta}_p^4 + 342.909\boldsymbol{\theta}_p^6 - 462.471\boldsymbol{\theta}_p^8 + 433.434\boldsymbol{\theta}_p^{10}}{-1 + 18.933\boldsymbol{\theta}_p^2 - 79.532\boldsymbol{\theta}_p^4 + 37.311\boldsymbol{\theta}_p^6 - 73.083\boldsymbol{\theta}_p^8 + 95.96\boldsymbol{\theta}_p^{10}}$$

$$\boldsymbol{U}_{n,\mathbf{ref}} = 0.7222 + 0.1387\boldsymbol{\theta_n} + 0.029\boldsymbol{\theta}_n^{0.5} - \tfrac{0.0172}{\boldsymbol{\theta_n}} + \tfrac{0.0019}{\boldsymbol{\theta}_n^{1.5}} + 0.2808e^{0.9-15\boldsymbol{\theta_n}} - 0.7984e^{0.4465\boldsymbol{\theta_n}-0.4108}$$

$$\boldsymbol{\theta}_p = \frac{c_{s,p}^*(x,t)}{c_{s,p}^{\max}}$$

$$\boldsymbol{\theta}_n = \frac{c_{s,n}^*(x,t)}{c_{s,n}^{\max}}$$

Heat source terms (Anode and Cathode)

$$Q_{\text{ohm},i}(x,t) = \boldsymbol{\sigma}_{\mathbf{eff},i}\left(\frac{\partial\Phi_s(x,t)}{\partial x}\right)^2 + \boldsymbol{\kappa}_{\mathbf{eff},i}\left(\frac{\partial\Phi_e(x,t)}{\partial x}\right)^2 + \frac{2\boldsymbol{\kappa}_{\mathbf{eff},i}\mathrm{R}T(x,t)}{F}(1-t_+)\frac{\partial\ln c_e(x,t)}{\partial x}\frac{\partial\Phi_e(x,t)}{\partial x},\ i\in\{p,n\}$$

$$Q_{\text{rxn},i}(x,t) = F\,a_i\,j(x,t)\,\eta_i(x,t),\ i\in\{p,n\}$$

$$Q_{\text{rev},i}(x,t) = F\,a_i\,j(x,t)\,T(x,t)\,\frac{\boldsymbol{\partial U}_i}{\boldsymbol{\partial T}}\bigg|_{\boldsymbol{T}_{\text{ref}}},\ i\in\{p,n\}$$

Heat Source terms (Separator)

$$Q_{\text{ohm},i}(x,t) = \boldsymbol{\kappa}_{\mathbf{eff},i}\left(\frac{\partial\Phi_e(x,t)}{\partial x}\right)^2 + \frac{2\boldsymbol{\kappa}_{\mathbf{eff},i}\mathrm{R}T(x,t)}{F}(1-t_+)\frac{\partial\ln c_e(x,t)}{\partial x}\frac{\partial\Phi_e(x,t)}{\partial x},\ i=s$$

Various Coefficients

$$\boldsymbol{D}_{\mathbf{eff},i} = \epsilon_i^{\text{brugg}_i}\times 10^{-4}\times 10^{-4.43-\frac{54}{T(x,t)-229-5\times10^{-3}c_e(x,t)}-0.22\times10^{-3}c_e(x,t)}$$

$$\boldsymbol{\kappa}_{\mathbf{eff},i} = \epsilon_i^{\text{brugg}_i}\times 10^{-4}\times c_e(x,t)\begin{pmatrix}-10.5 + 0.668\cdot10^{-3}\cdot c_e(x,t) + 0.494\cdot10^{-6}c_e^2(x,t) + \\ (0.074 - 1.78\times10^{-5}c_e(x,t) - 8.86\times10^{-10}c_e^2(x,t))T(x,t) + \\ (-6.96\times10^{-5} + 2.8\times10^{-8}c_e(x,t))T^2(x,t)\end{pmatrix}^2$$

$$\boldsymbol{k}_{\mathbf{eff},i} = k_i\,e^{-\frac{E_{\mathrm{R}}^{k_i}}{\mathrm{R}}\left(\frac{1}{T(x,t)}-\frac{1}{T_{\text{env}}}\right)} \qquad \Upsilon := \frac{2(1-t_+)\mathrm{R}}{F}$$

$$\boldsymbol{D}_{\mathbf{eff},i}^{\boldsymbol{s}} = D_i^s\,e^{-\frac{E_{\mathrm{R}}^{D_i^s}}{\mathrm{R}}\left(\frac{1}{T(x,t)}-\frac{1}{T_{\text{env}}}\right)} \qquad \boldsymbol{\sigma}_{\mathbf{eff},i} = \sigma_i(1-\epsilon_i-\epsilon_{f,i})$$

Table 2.3: Additional equations

2.5 Numerical implementation

Most numerical methods for model-based estimation and control algorithms require the model to be formulated in terms of Algebraic Equations (AEs) or DAEs rather than PDAEs. Different numerical methods can be used to achieve this objective. The reformulation process from PDAEs to AEs or DAEs is carried out by discretizing the domains of the independent variables (e.g., the time domain t and the N-dimensional spatial domain $x \in \mathbb{R}^N$). The discretization can involve both time and space, to produce AEs, or only space, to produce DAEs. An example of discretization in time and space is given by the Forward-Time Central-Space (FTCS) approach [Stoer and Bulirsch, 2013]. Other techniques, like the Method Of Lines (MOLs) [Schiesser, 1991], discretize only the space domain and leave the time as a continuous variable. When this latter approach is used, finite volume, finite difference, or finite element methods can be employed to obtain the set of DAEs. Alternative approaches can be used. For example, orthogonal collocation can be used with an efficient coordinate transformation to solve the set of resulting DAEs [Northrop et al., 2011]. In this section, in order to exploit the properties of variable-step solvers, MOLs is used to reformulate the original set of PDAEs. In particular, the finite volume method is employed. Due to its ability to conserve properties with high accuracy (within numerical roundoff errors), the FVM has been used in literature to discretize models in a wide range of applications, such as heat transfer problems [Chai and Patankar, 2000], flow and transport in porous media [Jenny et al., 2005], or more general applications for hyperbolic problems as discussed in [LeVeque, 2002]. In particular, the FVM together with the Harmonic Mean (HM) have been used to deal with possible discontinu-

ities across different sections of the cell. In the following, all the numerical details are addressed.

2.5.1 Finite volume formulation

Consider a general diffusion-convection equation defined on a domain in $\mathbb{R}^N$ of the form

$$\frac{\partial \phi}{\partial t} + \nabla(\eta\phi) = \nabla(\Gamma\nabla\phi) + s \tag{2.9}$$

where ϕ is the unknown variable, η is the velocity, Γ is a diffusion coefficient and s a source term. Both the unknown ϕ and the source term s depend on time t and space $x \in \mathbb{R}^N$. For convenience define $f(\phi) := \eta\phi - \Gamma\nabla\phi$. Integrating (2.9) over a spatial domain $\Omega \subset \mathbb{R}^N$ and applying the divergence theorem produces the integral form of the conservation law:

$$\int_\Omega \frac{\partial \phi}{\partial t} dV + \oint_{d\Omega} (f(\phi) \cdot n)\, dS = \int_\Omega s dV \tag{2.10}$$

where $d\Omega$ is the boundary of the domain Ω, n is the outward pointing unit normal on the boundary of the domain, and dV and dS represent the infinitesimal volume of Ω and the infinitesimal surface of the boundary $d\Omega$ respectively. Alternatively, this integral equation could be written directly as an exact conservation equation over any prescribed spatial domain.

According to the FVM, the spatial domain Ω is divided into a set of disjoint Control Volumes (CVs) Ω_k centered in $x_k \in \mathbb{R}^N$, such that $\Omega = \cup_k \Omega_k$ and $\Omega_i \cap \Omega_j = \emptyset\,, \forall i \neq j$. The average value of the unknown variables for each CV is

$$\bar{\phi}_k(t) = \frac{1}{G_k} \int_{\Omega_k} \phi(x,t)\ dV$$

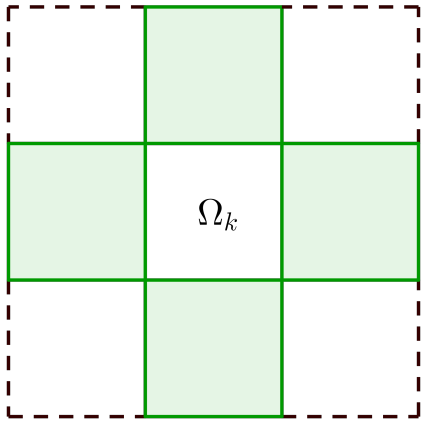


Figure 2.9: Example of a 2D FVM mesh where the set of neighbor cells $\mathcal{C}_k$ is represented by the green cells.

where G_k represents the volume of Ω_k. Using this equation, the integrals in (2.10) can be reformulated as

$$\dot{\bar{\phi}}_k(t) + \sum_{j \in \mathcal{C}_k} (F(\bar{\phi}) \cdot n)_{k,j} \approx \bar{s}_k(t) \tag{2.11}$$

where $\mathcal{C}_k$ is the set of the neighbor cells to the kth CV and $(F(\bar{\phi}) \cdot n)_{k,j}$ is the normal component of the numerical approximation of $f(\phi) \cdot n$, directed toward x_j starting from x_k. An illustrative example of the set $\mathcal{C}_k$ is given in Fig. 2.9. Suitable numerical approximations need to be employed for the term $F(\bar{\phi})$; given that the average values of the unknown variables $\bar{\phi}$ are computed in the FVM, interpolation techniques are employed to recover the value of such unknowns at the edges of the CVs [ten Thije Boonkkamp and Anthonissen, 2011]. The approximation of $F(\bar{\phi})$ is discussed in the next section.

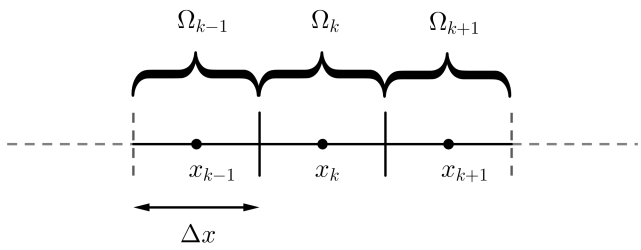


Figure 2.10: One-dimensional finite volume mesh

2.5.2 Discretization of the governing equations

The discretization method introduced in Section 2.5.1 is exploited to reformulate the set of governing equations summarized in Table 2.2. Given that all the unknowns of the Li-ion cell model are functions of the variables $t \in \mathbb{R}^+$ and $x \in \mathbb{R}$, the development of a 1D FVM model is addressed. In order to correctly carry out the discretization process, a mesh structure of the spatial domain is defined by subdividing it into $N_a + N_p + N_s + N_n + N_z$ non-overlapping volumes with geometrically centered nodes (as depicted in Fig. 2.10). Every CV is associated with a center x_k and spans the interval $[x_{k-\frac{1}{2}}; x_{k+\frac{1}{2}}]$. To facilitate the treatment of boundary and interface conditions, the edges of each CV are aligned with the domain boundaries and internal interfaces. The width of every CV is defined as $\Delta x_i = l_i/N_i$, where i represents a particular section of the battery.

Once the discretization mesh is structured, the governing equations are discretized as summarized in Table 2.4. All the interface conditions used to enforce continuity between adjacent materials are discussed in Section 2.5.3.

Particular attention is required for the thermal dynamics. The re-

versible and reactive heat sources can be discretized as

$$\bar{Q}_{\text{rev},k} = Fa_i\,\bar{j}_k(t)\,\bar{T}_k(t)\frac{\boldsymbol{\partial U_{i,k}}}{\boldsymbol{\partial T}}$$
$$\bar{Q}_{\text{rxn},k} = Fa_i\,\bar{j}_k(t)\,\bar{\eta}_{i,k}(t)$$

whereas the derivatives present in the ohmic source are numerically approximated as

$$\left.\frac{\partial \Phi_s(x,t)}{\partial x}\right|_{x_k} \approx \frac{\bar{\Phi}_{s,k+1}(t) - \bar{\Phi}_{s,k-1}(t)}{2\Delta x_i}$$
$$\left.\frac{\partial \Phi_e(x,t)}{\partial x}\right|_{x_k} \approx \frac{\bar{\Phi}_{e,k+1}(t) - \bar{\Phi}_{e,k-1}(t)}{2\Delta x_i}$$
$$\left.\frac{\partial \ln c_e(x,t)}{\partial x}\right|_{x_k} \approx \frac{\bar{c}_{e,k+1}(t) - \bar{c}_{e,k-1}(t)}{2\Delta x_i\,\bar{c}_{e,k}(t)}$$

using a central differencing scheme. Finally the term $\bar{Q}_{\text{source},k} := \bar{Q}_{\text{ohm},k} + \bar{Q}_{\text{rev},k} + \bar{Q}_{\text{rxn},k}$.

As discussed in Section 2.5.1, a suitable numerical approximation for $F(\bar{\phi})$ is needed. Given that no convective terms are present in the set of governing equations, numerical approximation is only required for the diffusive terms (e.g., $-\Gamma\nabla\phi$). In this work, all the diffusive terms are numerically approximated with a first-order scheme:

$$\left.\frac{\partial \phi(x,t)}{\partial x}\right|_{x_{k+\frac{1}{2}}} \approx \frac{\bar{\phi}_{k+1}(t) - \bar{\phi}_k(t)}{\Delta x}$$
$$\left.\frac{\partial \phi(x,t)}{\partial x}\right|_{x_{k-\frac{1}{2}}} \approx \frac{\bar{\phi}_k(t) - \bar{\phi}_{k-1}(t)}{\Delta x}$$

All the values coming from the additional equations in Table 2.3 are obtained as a function of the average values of the unknowns. Equation (T) is used to obtain the values of T, while the equations (M1), (M2), and (M3) are used to obtain the values of c_e, c_s^{avg}, and c_s^* respectively. The values of

Φ_s are obtained from (C1) while the values of Φ_e are calculated through (C2).

2.5.3 Implementation of boundary and interface conditions

Boundary conditions must be enforced to have a physically meaningful solution. As shown in Table 2.2, null-flux boundary conditions on the electrolyte diffusion equation c_e can be straightforwardly enforced by imposing $\frac{\partial c_e}{\partial x} = 0$ at $x = \hat{x}_0$ and $x = \hat{x}_n$. The same procedure can be used to enforce $\frac{\partial \Phi_e}{\partial x} = 0$ at $x = \hat{x}_0$, while $\Phi_e = 0$ at $x = \hat{x}_n$ is enforced by setting to zero the value of Φ_e at the last CV of the anode. Solid-phase potential boundaries are enforced by substituting $\frac{\partial \Phi_s}{\partial x}$ at $x = \hat{x}_0$ and $x = \hat{x}_n$ the value of $-I_{\text{app}}(t)/\boldsymbol{\sigma}_{\mathbf{eff},i}$. Similarly, at $x = \hat{x}_p$ and $x = \hat{x}_s$, $\frac{\partial \Phi_s}{\partial x}$ is replaced by the value 0. To enforce heat exchange with the surrounding environment, the terms $\frac{\partial T}{\partial x}$ evaluated at $x = 0$ and $x = L$ are substituted with the terms $h(T_{\text{env}} - \bar{T}_1)$ and $-h(\bar{T}_{\text{end}} - T_{\text{env}})$ respectively. The suffixes $_1$ and $_{\text{end}}$ refer to the first and last CV of the entire mesh. All these conditions have been formulated also for the FVM discretization as shown in Table 2.4.

Due to changes in material properties along the length of the battery, interface conditions are required to enforce continuity of the solution. For this reason, the values of different coefficients (e.g., $\boldsymbol{D}_{\mathbf{eff},i}$, $\boldsymbol{\kappa}_{\mathbf{eff},i}$, λ_i) need to be evaluated at the interface between two different materials. The easiest way would be to use an arithmetic mean (or a linear interpolation of adjacent values); however, in some cases, this approach cannot accurately handle the abrupt changes of coefficients that may occur. Instead, the HM is employed to evaluate the value at the edges of the CVs of such coefficients. The HM of two generic coefficients (k_1 and k_2) can be expressed

as

$$\frac{k_1 k_2}{\beta k_2 + (1-\beta) k_1}$$

where β represents a weight to account for the difference between the different CV widths. A common value for β is $\beta = \frac{\Delta x_1}{\Delta x_2 + \Delta x_1}$, where Δx_1 and Δx_2 represent the CV widths. This formulation produces results that are more robust in presence of the abrupt changes of the coefficients, without requiring an excessively fine grid in the vicinity of the interface [Patankar, 1980].

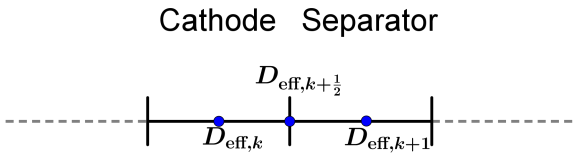


Figure 2.11: Electrolyte diffusion process: interface across the cathode and separator.

Consider Fig. 2.11 where the interface across the last volume of the cathode and the first volume of the separator is depicted. Remember that, as discussed in Section 2.5.2, the mesh structure has been chosen in order to align the CV edges with the interfaces or physical boundaries of the battery. The value of $\boldsymbol{D_{\text{eff},k+\frac{1}{2}}}$ can be obtained using the HM as

$$\boldsymbol{D_{\text{eff},k+\frac{1}{2}}} = \frac{\boldsymbol{D_{\text{eff},k} D_{\text{eff},k+1}}}{\beta \boldsymbol{D_{\text{eff},k+1}} + (1-\beta) \boldsymbol{D_{\text{eff},k}}}$$

where $\beta = \frac{\Delta x_p}{\Delta x_p + \Delta x_s}$. The electrolyte diffusion in the last volume of the

cathode is

$$\begin{aligned}\epsilon_p \frac{\partial \bar{c}_{e,k}(t)}{\partial t} &= \boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}+\frac{\mathbf{1}}{\mathbf{2}}} \frac{(\bar{c}_{e,k+1}(t) - \bar{c}_{e,k}(t))}{\Delta x_p(\Delta \tilde{x})} \\ &- \boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}-\frac{\mathbf{1}}{\mathbf{2}}} \frac{(\bar{c}_{e,k}(t) - \bar{c}_{e,k-1}(t))}{\Delta x_p^2} \\ &+ a_p \, (1 - t_+) \, \bar{j}_k(t)\end{aligned}$$

whereas

$$\begin{aligned}\epsilon_s \frac{\partial \bar{c}_{e,k+1}(t)}{\partial t} &= \boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}+\frac{\mathbf{3}}{\mathbf{2}}} \frac{(\bar{c}_{e,k+2}(t) - \bar{c}_{e,k+1}(t))}{\Delta x_s^2} \\ &- \boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}+\frac{\mathbf{1}}{\mathbf{2}}} \frac{(\bar{c}_{e,k+1}(t) - \bar{c}_{e,k}(t))}{\Delta x_s(\Delta \tilde{x})}\end{aligned}$$

in the first volume of the separator, with $\Delta\tilde{x} = \frac{\Delta x_s + \Delta x_p}{2}$. Note that when evaluating for instance the value of $\boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}}$, the polynomial function stated in Table 2.3 is considered, in which the averaged values $\bar{c}_{e,k}(t)$ and $\bar{T}_k(t)$ are used. The same procedure applies for evaluating all the other **bolds** values.

Besides being used for equation (M1), the HM has also been used in equation (T) and (C2) (Table 2.4) to enforce continuity across the different sections of the cell for the temperature $T(x,t)$ and the electrolyte potential $\Phi_e(x,t)$. Moreover, according to its FVM formulation, the second term of the left hand side of (C2) requires the evaluation of $T(x,t)$, $c_e(x,t)$, and $\boldsymbol{\kappa}_{\text{eff}}$ at the edges of the CVs. Since in this case the evaluation of such quantities does not involve the enforcement of continuity across the different sections of the cell, their edges values can be recovered using linear interpolation techniques as illustrated in Fig. 2.12.

When dealing with battery packs, in particular with series-connected cells, all the aforementioned numerical schemes have to be replicated for each

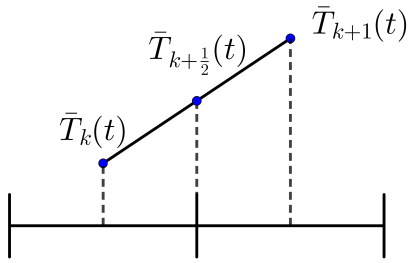


Figure 2.12: Interpolation technique to recover edge values of the unknowns.

cell. Moreover, when temperature dynamics are considered, the numerical scheme has to be adapted in order to account for continuity fluxes across the cells. Indeed, if two cells are connected in series,

$$-\lambda_{z,1}\frac{\partial T_{\text{cell 1}}(x,t)}{\partial x}\bigg|_{x=x^*} = -\lambda_{a,2}\frac{\partial T_{\text{cell 2}}(x,t)}{\partial x}\bigg|_{x=x^*}$$

must hold at the interface of the current collectors across the two cells (e.g., at $x = x^*$). Finally, Newton's law of cooling has to be enforced respectively at the positive current collector of the first cell and at negative current collector of the second cell.

2.6 LIONSIMBA: the Li-ION SIMulation BAttery toolbox

Different implementations of Li-ion cell simulation have been reported in the literature written in such languages as Maple and Fortran (DUALFOIL [Doyle, 1998]), and in commercial software such as COMSOL Multiphysics [Cai and White, 2011] and Modelica [Tiller, 2012] which provide a variety of models to simulate the behavior of a Li-ion cell. Matlab® , however, is the software language most commonly used by researchers for the devel-

opment and evaluation of different identification, estimation, and control algorithms, as Matlab® has by far the largest number of toolboxes that implement the widest variety of such algorithms. Combined with its Instrument Control Toolbox that has a very extensive suite of protocols for directly communicating and controlling laboratory equipment, Matlab® has the maximum flexibility for evaluation of control algorithms through simulations and experiments.

This work provides a freely available Matlab-based software for the simulation of Li-ion cells, the LIONSIMBA toolbox, to serve as a reference simulation environment for: (*i*) the facile development of different ABMSs strategies, (*ii*) the evaluation of identification algorithms for the estimation of indices like the SOC and SOH, and (*iii*) the optimization of the manufacturing parameters. LIONSIMBA is freely downloadable at:

`http://sisdin.unipv.it/labsisdin/lionsimba.php`

The package comes with Matlab® editable *.m* scripts:[1]

- **electrolyteDiffusionCoefficients.m**: computes the electrolyte diffusion coefficients.
- **electrolyteConductivity.m**: computes the electrolyte conductivity coefficients.
- **openCircuitPotential.m**: used to compute the open circuit voltage.
- **reactionRates.m**: computes the reaction rate coefficients for the ionic flux.

[1] This set of scripts refer to version 1.022 of the software; modifications or other scripts can be added in future releases of the software.

- **solidPhaseDiffusionCoefficients.m**: computes the solid phase diffusion coefficients.

All the parameters related to the simulator and to the battery are managed through the script **Parameters_init.m**. The customization of this script allows the user to disable features such as the thermal dynamics, change the number of CVs of the mesh, enable real-time display of results, and change the battery section lengths, thermal conductivities, porosities, and so on. The user can define the operating mode of the charge/discharge cycle by selecting between galvanostatic, potentiostatic, or variable current profile operations.

The script **getInputCurrent.m** contains an example for the definition of the variable current profile, and can be used to apply a customized current profile during the simulation of the Li-ion battery. A generic nonlinear function can be used for this purpose; extra parameters can be used inside this function: current time instant t, initial integration time t_0, final integration time t_f, and a structure-containing extra user data. For example, a possible implementation is

$$I(t) = \alpha \frac{t - t_0}{t_f - t_0} + \xi, \ [\alpha, \xi] \in \mathbb{R}$$

An additional degree of freedom is set by the possibility of defining a custom algorithm for the estimation of SOC and SOH. Within the **Parameters_init.m** script, the user can set custom functions to be externally called after each integration step; these functions will receive all the integration data of the battery and an extra structure-containing user-defined data.

A simulation can be initiated by calling from the Matlab® command line:

```
out = startSimulation(t0,tf,initialStates,I,param)
```

where

- t0: represents the initial integration time.
- tf: represents the final integration time.
- initialStates: represents the structure of initial states.
- I: represents the value of the applied input current.
- param: represents the cell array of parameters structures to be used in simulation.

The structure initialStates can be used as initial state from which to start a simulation. If left empty, LIONSIMBA will automatically compute a set of Consistent Initial Conditions (CICs) starting from which the simulation will run. If initialStates is used as a parameter, it has to be a set of CICs for the battery model in Table 2.4. In case it is not a set of CICs, the numerical integrator will fail to converge and no results will be provided. The param array, if passed, is used as the set of parameters for the simulation. If empty, the software will use a set of parameters according to the settings defined by the user in the script **Parameters_init.m**. When designing ABMSs for battery packs with series-connected cells, a cell-wise balancing must be ensured during charging [Moore and Schneider, 2001, Bentley, 1997]. LIONSIMBA can support the user in this task by providing a full independent parametrization of each cell of the series. If the param array contains a multiple parameters structure, the software will perform a simulation of a battery pack composed of several cells connected in series as shown in Section 2.7.4. Each element of the pack can be parameterized

individually, leading to independent simulations of each cell. Finally, the out structure will contain the values of all the dependent variables and parameters used in the simulations. The package requires the SUNDIALS [Hindmarsh et al., 2005] suite to be installed and correctly configured with Matlab® ; in particular, the solver IDA is used.

To obtain further help on any single script, the user can type

help <scriptname>

from the Matlab® command line or refer to the software manual.

The numerical implementation of the LIONSIMBA has been carried out according to the rules outlined in Section 2.5 and the cell considered is a $LiCoO_2$ and LiC_6 system. All the parameter values have been taken from the real battery data in [Northrop et al., 2011], and are summarized in Table 2.5.

Current Collectors, $i \in \{a, z\}$	**Boundary Conditions**
(T) $\rho_i C_{p,i} \dfrac{\partial \bar{T}_k(t)}{\partial t} = \dfrac{1}{\Delta x_i} \left[\lambda_i \dfrac{\partial T(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} + \dfrac{I_{\text{app}}^2(t)}{\boldsymbol{\sigma}_{\mathbf{eff},i}}$	$\left[\lambda_i \dfrac{\partial T(x,t)}{\partial x} \right] \Big\vert_{0} = h(T_{\text{env}} - \bar{T}_1(t))$ $\left[\lambda_i \dfrac{\partial T(x,t)}{\partial x} \right] \Big\vert_{\text{L}} = h(\bar{T}_{\text{end}}(t) - T_{\text{env}})$
Positive and Negative Electrodes, $i \in \{p, n\}$	
(M1) $\epsilon_i \dfrac{\partial \bar{c}_{e,k}(t)}{\partial t} = \dfrac{1}{\Delta x_i} \left[\boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}} \dfrac{\partial c_e(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} + a_i(1 - t_+)\bar{j}_k(t)$	$\dfrac{\partial c_e(x,t)}{\partial x}\Big\vert_{\hat{x}_0} = 0$ $\dfrac{\partial c_e(x,t)}{\partial x}\Big\vert_{\hat{x}_n} = 0$
(M2) $\dfrac{\partial \bar{c}_s^{\text{avg}}(t)}{\partial t} = -3\dfrac{\bar{j}_k(t)}{R_{p,i}}$	
(M3) $\bar{c}_s^*(t) - \bar{c}_s^{\text{avg}}(t) = -\dfrac{R_{p,i}}{\boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}}^{\text{s}}} \dfrac{\bar{j}_k(t)}{5}$	
(C1) $\left[\boldsymbol{\sigma}_{\mathbf{eff},\boldsymbol{k}} \dfrac{\partial \Phi_s(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} = a_i\, F\, \bar{j}_k(t)\, \Delta x_i$	$\left[\boldsymbol{\sigma}_{\mathbf{eff},\boldsymbol{k}} \dfrac{\partial \Phi_s}{\partial x} \right] \Big\vert_{\hat{x}_0, \hat{x}_n} = -I_{\text{app}}(t)$ $\dfrac{\partial \Phi_s(x,t)}{\partial x}\Big\vert_{\hat{x}_p, \hat{x}_s} = 0$
(C2) $-\left[\boldsymbol{\kappa}_{\mathbf{eff},\boldsymbol{k}} \dfrac{\partial \Phi_e(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} + \left[\boldsymbol{\kappa}_{\mathbf{eff},\boldsymbol{k}} T(x,t) \Upsilon \dfrac{\partial \ln c_e(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} = \Delta x_i\, a_i\, F\, \bar{j}_k(t)$	$\dfrac{\partial \Phi_e(x,t)}{\partial x}\Big\vert_{\hat{x}_0} = 0$ $\bar{\Phi}_{e,\text{end}} = 0$
(T) $\rho_i C_{p,i} \dfrac{\partial \bar{T}_k(t)}{\partial t} = \dfrac{1}{\Delta x_i} \left[\lambda_i \dfrac{\partial T(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} + \bar{Q}_{\text{source},k}$	
$\bar{j}_{i,k}(t) = 2\boldsymbol{k}_{\mathbf{eff},\boldsymbol{k}} \sqrt{\bar{c}_{e,k}(t)(c_{s,i}^{\max} - \bar{c}_{s,k}^*(t))\bar{c}_{s,k}^*(t)} \sinh\left[\frac{0{,}5R}{F\bar{T}_k(t)} \bar{\eta}_{i,k}(t) \right]$	
$\bar{\eta}_{i,k}(t) = \bar{\Phi}_{s,k}(t) - \bar{\Phi}_{e,k}(t) - \boldsymbol{U}_{\boldsymbol{i},\boldsymbol{k}}(t)$	
Separator, $i = s$	
(M1) $\epsilon_i \dfrac{\partial \bar{c}_{e,k}(t)}{\partial t} = \dfrac{1}{\Delta x_i} \left[\boldsymbol{D}_{\mathbf{eff},\boldsymbol{k}} \dfrac{\partial c_e(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}}$	
(C2) $-\left[\boldsymbol{\kappa}_{\mathbf{eff},\boldsymbol{k}} \dfrac{\partial \Phi_e(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} + \left[\boldsymbol{\kappa}_{\mathbf{eff},\boldsymbol{k}} T(x,t) \Upsilon \dfrac{\partial \ln c_e(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} = 0$	
(T) $\rho_i C_{p,i} \dfrac{\partial \bar{T}_k(t)}{\partial t} = \dfrac{1}{\Delta x_i} \left[\lambda_i \dfrac{\partial T(x,t)}{\partial x} \right] \Big\vert_{x_{k-\frac{1}{2}}}^{x_{k+\frac{1}{2}}} + \bar{Q}_{\text{ohm},k}$	

Table 2.4: P2D model FVM discretized formulation

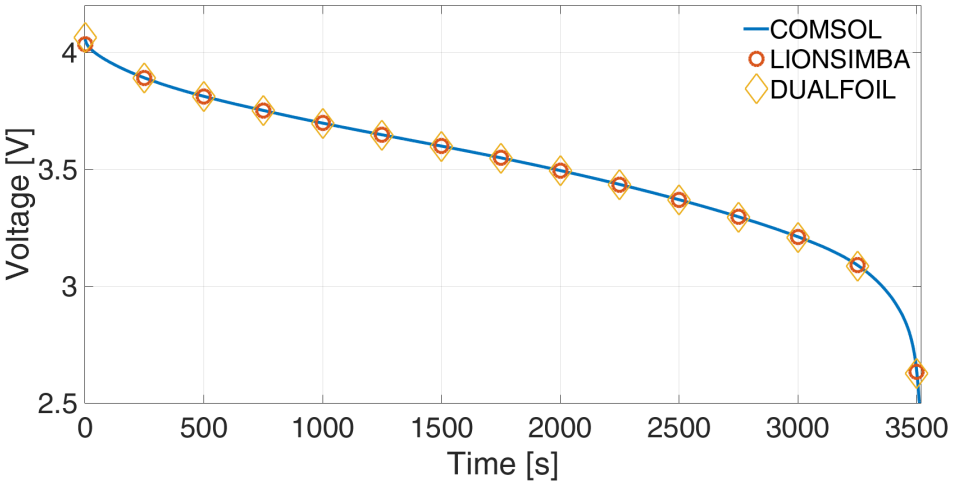
Parameter	Positive CC	Cathode	Separator	Anode	Negative CC
c_e^{init}	–	1000	1000	1000	–
$c_s^{\text{avg,init}}$	–	25751	–	26128	–
$c_s^{\max}$	–	51554	–	30555	–
D_i	–	7.5×10^{-10}	7.5×10^{-10}	7.5×10^{-10}	–
D_i^s	–	10^{-14}	–	3.9×10^{-14}	–
k_i	–	2.334×10^{-11}	–	5.031×10^{-11}	–
l_i	10^{-5}	8×10^{-5}	2.5×10^{-5}	8.8×10^{-5}	10^{-5}
$R_{p,i}$	–	2×10^{-6}	–	2×10^{-6}	–
ρ_i	2700	2500	1100	2500	8940
$C_{p,i}$	897	700	700	700	385
h	1	–	–	–	1
λ_i	237	2.1	0.16	1.7	401
σ_i	3.55×10^7	100		100	5.96×10^7
ϵ_i	–	0.385	0.724	0.485	–
a_i	–	885,000	–	723,600	–
$E_a^{D_i^s}$	–	5000	–	5000	–
$E_a^{k^i}$	–	5000	–	5000	–
brugg	–	4	4	4	–
F	96485	–	–	–	–
R	8.314472	–	–	–	–
t_+	0.364	–	–	–	–
$\epsilon_{f,i}$	–	0.025	–	0.0326	–

Table 2.5: Parameters present in LIONSIMBA

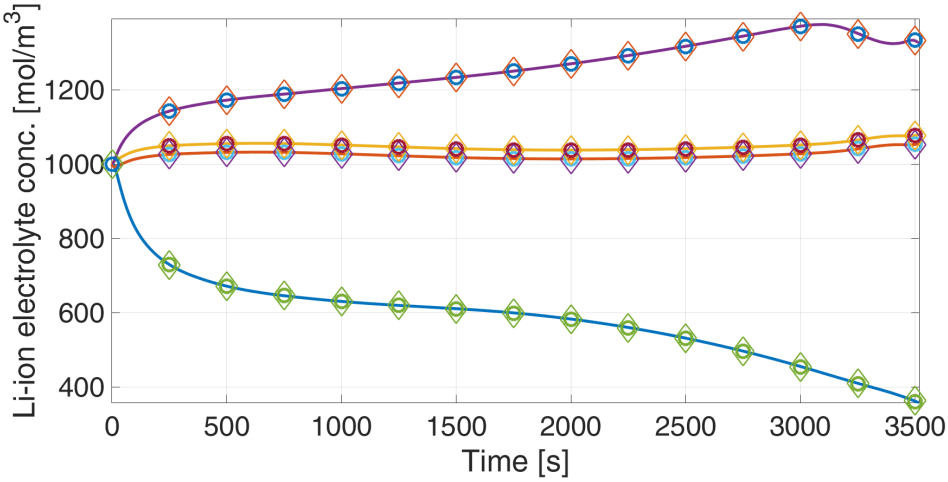
2.6.1 LIONSIMBA validation

The P2D model has been experimentally validated numerous times since [Doyle, 1995], this section addresses the numerical validation of LIONSIMBA by comparing the results coming from the proposed framework with COMSOL MultiPhysics and DUALFOIL. While COMSOL has been supplied with the same model used in our framework, where a heat diffusion Partial Differential Equation (PDE) is used to describe the thermal dynamics, DUALFOIL neglects the spatial distribution of the temperature and averages the heat generation rates over the cell [Rao and Newman, 1997]. For this reason, the comparison among the three different codes is carried out considering isothermal conditions. The thermal model is included in the comparison with COMSOL. For isothermal and thermal enabled scenarios, a 1C discharge cycle is performed, while the same set of parameters are maintained across the different codes.

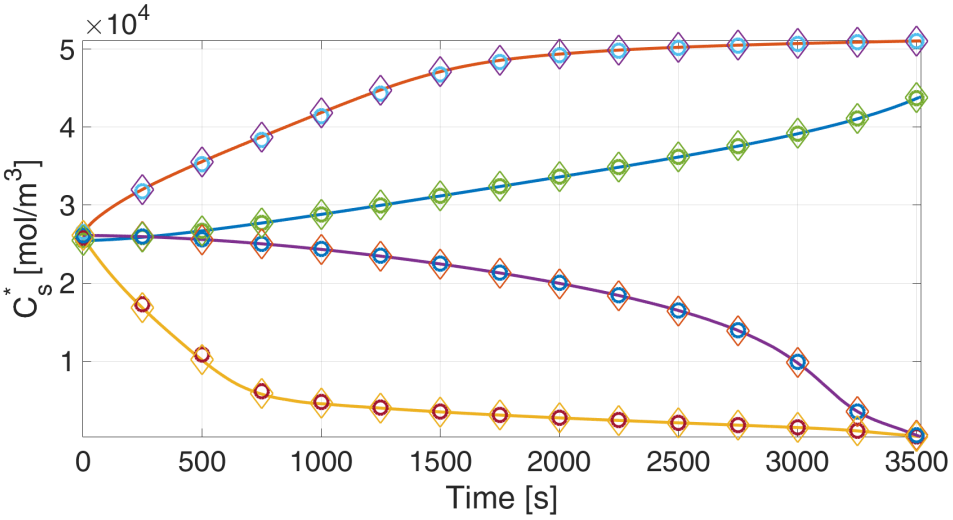
The cell potentials $V_{\text{out}}(t)$, electrolyte concentrations $c_e(x,t)$, potentials $\phi_e(x,t)$, and surface solid-phase concentrations $c_s^*(x,t)$ for the isothermal battery are nearly identical for LIONSIMBA, COMSOL, and DUALFOIL (see Fig. 2.13). For the thermal enabled scenario, the cell potentials, temperature, and other internal states for LIONSIMBA are nearly identical to COMSOL (see Fig. 2.14).



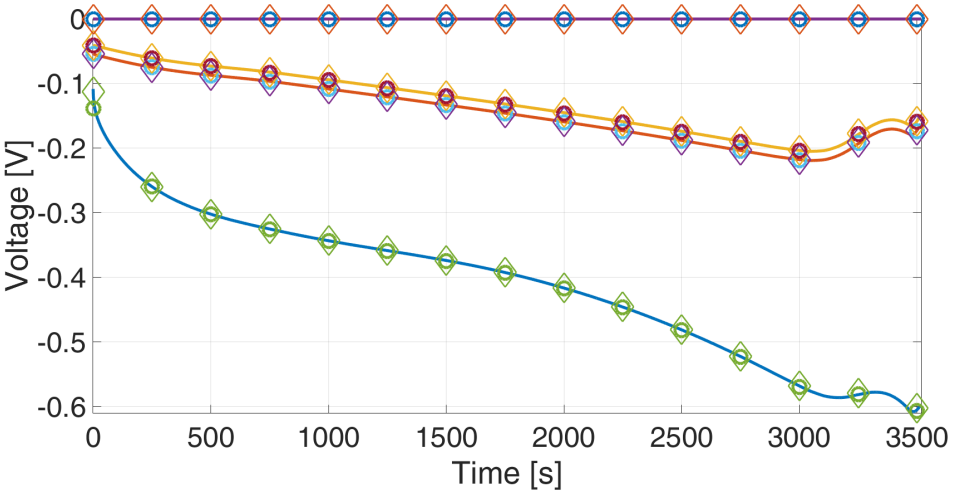
(a) *Cell potential*



(b) *Electrolyte Li-ion concentration*

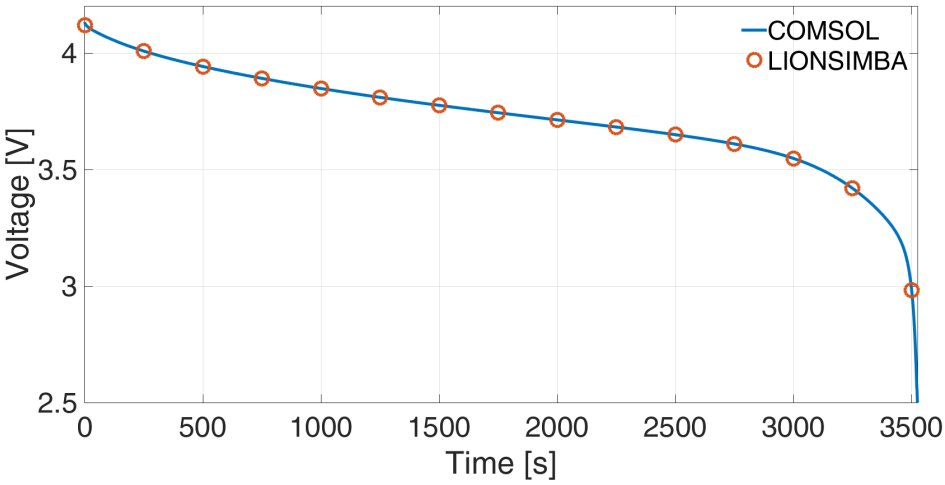


(c) *Solid phase Li-ion concentration*

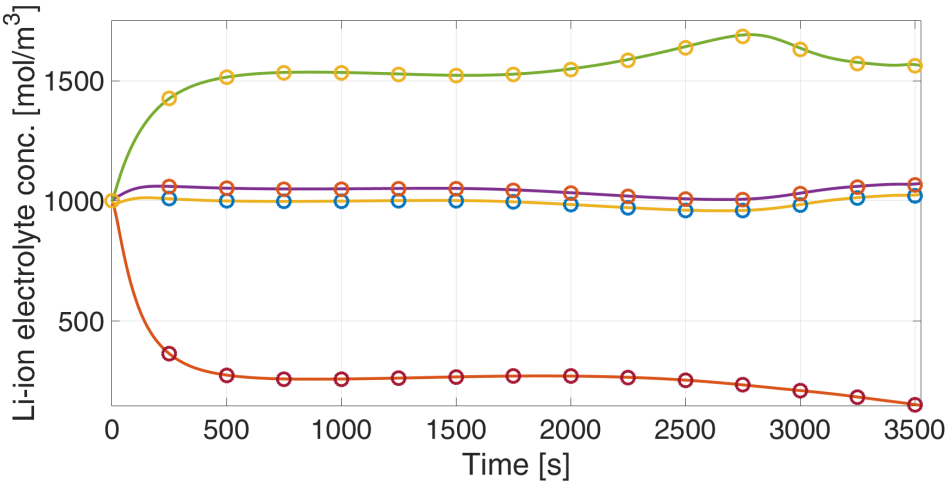


(d) *Electrolyte potential*

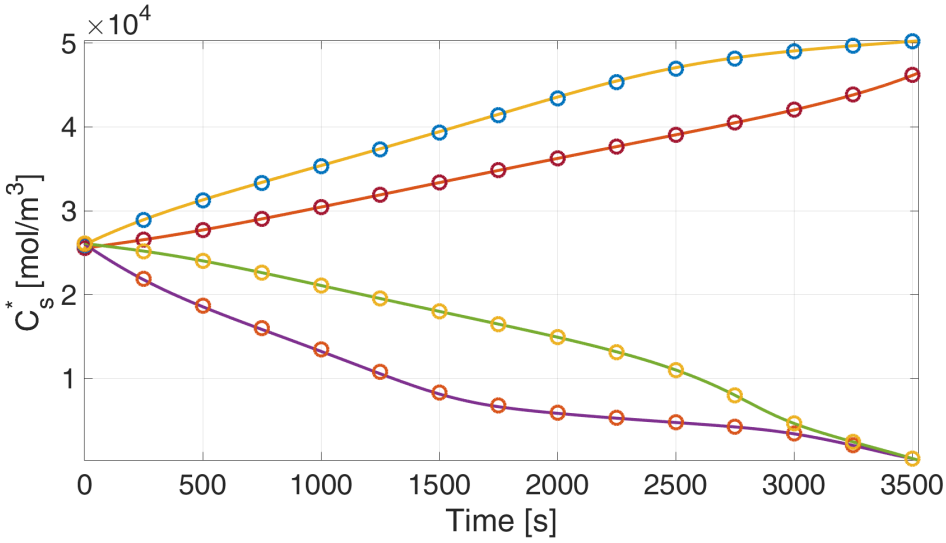
Figure 2.13: Validation of the LIONSIMBA numerical implementation in isothermal conditions, with the legend given in part a



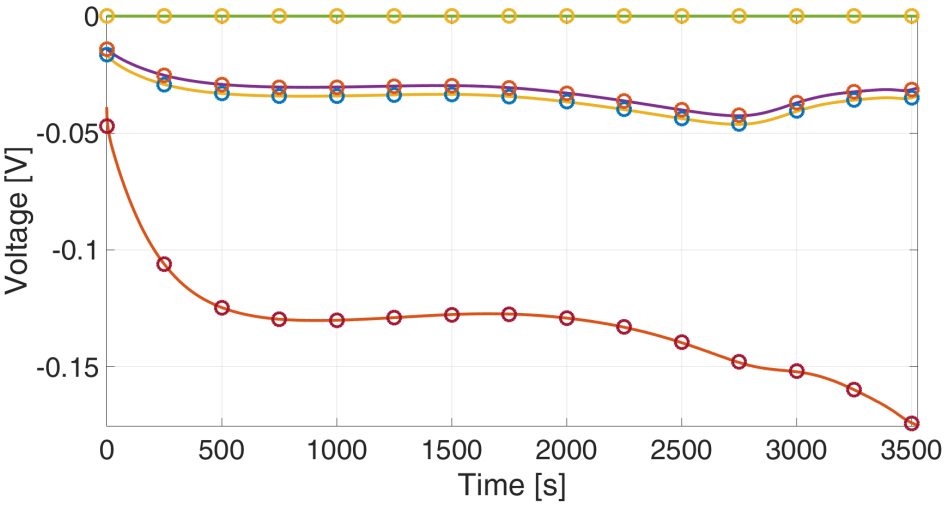
(a) *Cell potential*



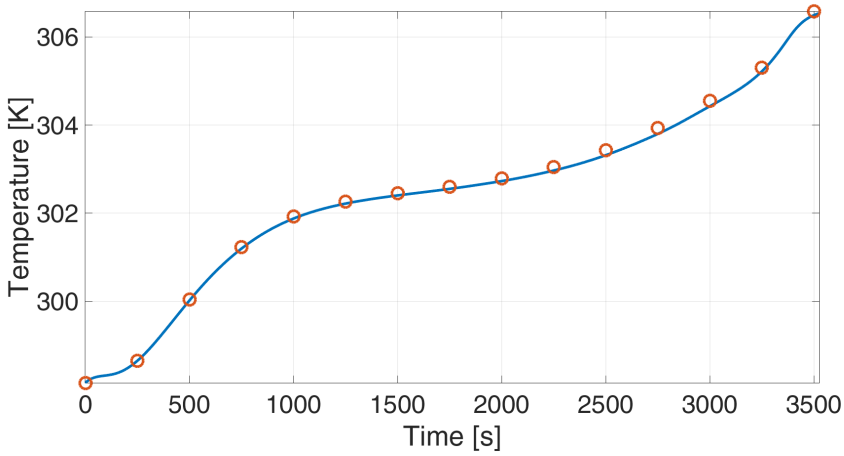
(b) *Electrolyte Li-ions concentration*



(c) *Solid phase Li-ions concentration*



(d) *Electrolyte potential*



(e) *Temperature*

Figure 2.14: Validation of the LIONSIMBA numerical implementation with thermal dynamics, with the legend given in part a

2.6.2 Solid-phase diffusion models

As introduced in Section 2.4, according to the P2D model developed in [Doyle, 1995], diffusion inside the solid particles is described using Fick's law, where the presence of a second-pseudo dimension (r) can significantly increase the computational burden. According to the particular application under study, different approximations of (2.2) can be employed without significant loss of accuracy. The choice of the solid-phase diffusion model should be cautious, as approximate models can have poor accuracy in scenarios comprising high rate of charge/discharge, short time simulations, or pulse currents [Zhang and White, 2007].

For this reason, LIONSIMBA allows the user to chose among three different models for solid-phase diffusion:

- Fick's law (including the pseudo-second dimension r):

$$\frac{\partial c_s(r,t)}{\partial t} = \frac{1}{r^2}\frac{\partial}{\partial r}\left[r^2\, \boldsymbol{D}^{\mathrm{s}}_{\mathbf{eff}}\, \frac{\partial c_s(r,t)}{\partial r}\right]$$

  with boundary conditions

$$\left.\frac{\partial c_s(r,t)}{\partial r}\right|_{r=0} = 0 \qquad \left.\frac{\partial c_s(r,t)}{\partial r}\right|_{r=R_p} = -\frac{j(x,t)}{\boldsymbol{D}^{\mathrm{s}}_{\mathbf{eff},i}}$$

- two-parameter polynomial approximation [Ramadesigan et al., 2010]:

$$\frac{\partial c_s^{\mathrm{avg}}(x,t)}{\partial t} = -3\frac{j(x,t)}{R_p}$$
$$c_s^{*}(x,t) - c_s^{\mathrm{avg}}(x,t) = -\frac{R_p}{\boldsymbol{D}^{\mathrm{s}}_{\mathbf{eff},i}}\frac{j(x,t)}{5}$$

- higher-order polynomial approximation [Ramadesigan et al., 2010]:

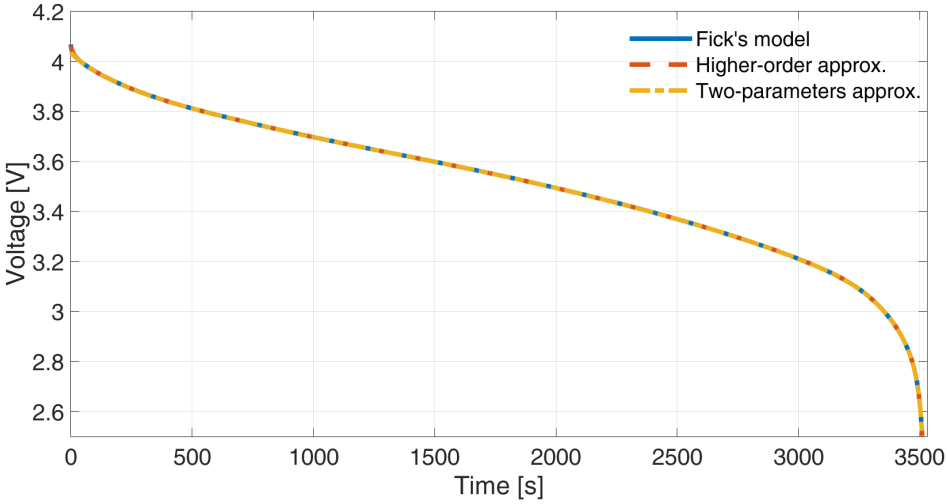
$$\frac{\partial c_s^{\text{avg}}(x,t)}{\partial t} = -3\frac{j(x,t)}{R_p}$$
$$\frac{\partial q(x,t)}{\partial t} = -30\frac{\boldsymbol{D}_{\mathbf{eff},i}^{\text{s}}}{R_p^2}q(x,t) - \frac{45}{2}\frac{j(x,t)}{R_p^2}$$
$$c_s^*(x,t) - c_s^{\text{avg}}(x,t) = -\frac{j(x,t)R_p}{35\boldsymbol{D}_{\mathbf{eff},i}^{\text{s}}} + 8R_p q(x,t)$$

The prediction accuracy of each approximate model is assessed by comparison of the cell potential vs. time profiles for different C rates for Fick's law with the two approximate models (see Fig. 7). The two-parameter approximation accurately describes the cell potential for low to medium C rates (1C to 2C, Figs. 7ab) and the higher order polynomial approximation is accurate up to the 5C rate (Fig. 7c), with increased error at the 10C rate typical of HEV applications (Fig. 7d).

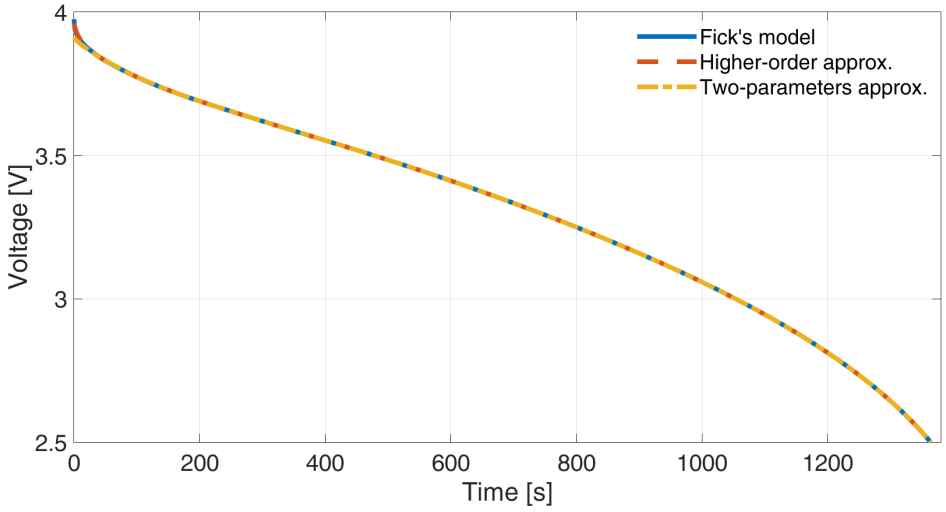
The performance of each approximate model are quantified by Root-Mean-Square Error (RMSE) and Normalized Time Index (NTI) in Table 2.6, where the RMSE is evaluated by comparing an approximate model solution with respect to the full model, while the normalized time index is the ratio between the computational time required by an approximate model and the time required by the full model with Fick's law to simulate different scenarios. The two-term polynomial approximation has much less than 1% error for the 1C and 2C rates, with more than 1% error for higher rates. In all of the scenarios, this approximate model takes $\approx 80\%$ less time than the full model to simulate the cell.

The higher-order polynomial approximation has a factor of 4 to 5 lower RMSE for each scenario than for the two-term model, but an increase in computational time by a factor of two or more due to the inclusion of

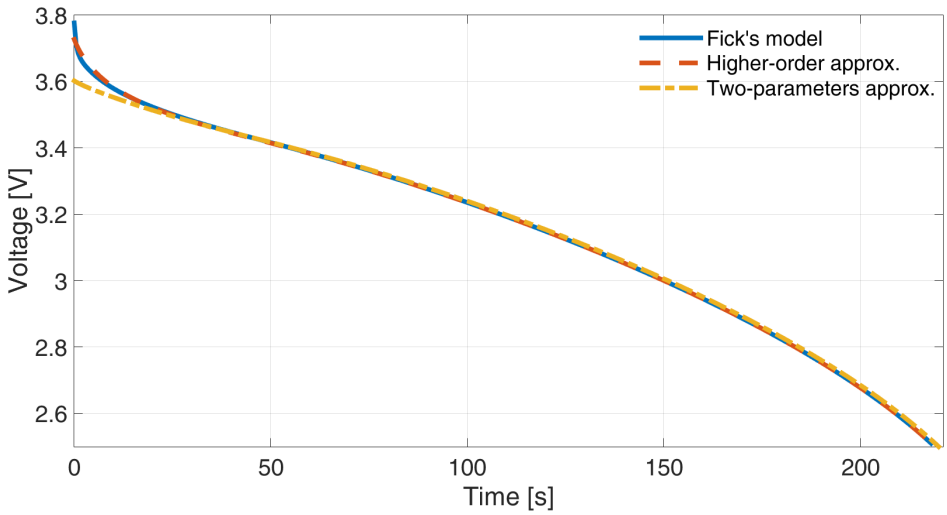
another set of PDEs. Although the RMSE of 1.87% at 10C could be small enough for some applications, the reduction in computational time is only about 37% compared to solving the full Fick's law model.



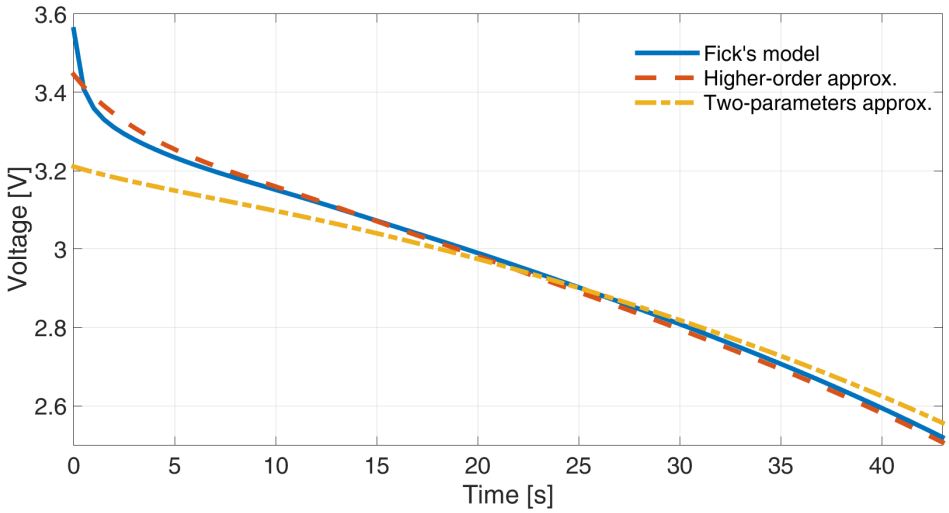
(a) 1C rate comparison



(b) 2C rate comparison



(c) 5C rate comparison



(d) 10C rate comparison

Figure 2.15: Comparison of the three different solid-phase diffusion equations implemented in LIONSIMBA.

	1C		2C		5C		10C	
	RMSE	NTI	RMSE	NTI	RMSE	NTI	RMSE	NTI
two-parameter	0.082%	20%	0.25%	18%	1.6%	22%	6.6%	24%
higher-order	0.017%	37%	0.053%	44%	0.36%	60%	1.9%	63%

Table 2.6: Comparison of different approximation methods for the diffusion in the solid particles. Root Mean Square Error (RMSE) and the Normalized Time Index (NTI) are shown.

2.7 LIONSIMBA capabilities

In order to demonstrate the capabilities of LIONSIMBA, several simulations have been performed. In particular, simulation results were obtained using Matlab® R2014b on a Windows 7@3.2GHz PC with 8 GB of RAM for the experimental battery parameters in [Northrop et al., 2011] with a cutoff voltage of 2.5 V and environmental temperature of 298.15 K. For the proposed chemistry, the 1C value is ≈ 30 A/m^2. The effectiveness and ease of use of the proposed framework are shown in a series of simulations.

2.7.1 Thermal dynamics simulations

In the first scenario (Fig. 2.16), 1C discharge simulations are compared for a very wide range of heat exchange coefficient h, with high h being the most challenging for retaining numerical stability in dynamic simulations. As expected, decreasing the value of the parameter h leads to a faster increase of the cell temperature. Moreover, due to the coupling of all of the governing equations, it is possible to note the influence of different temperatures on the cell voltage.

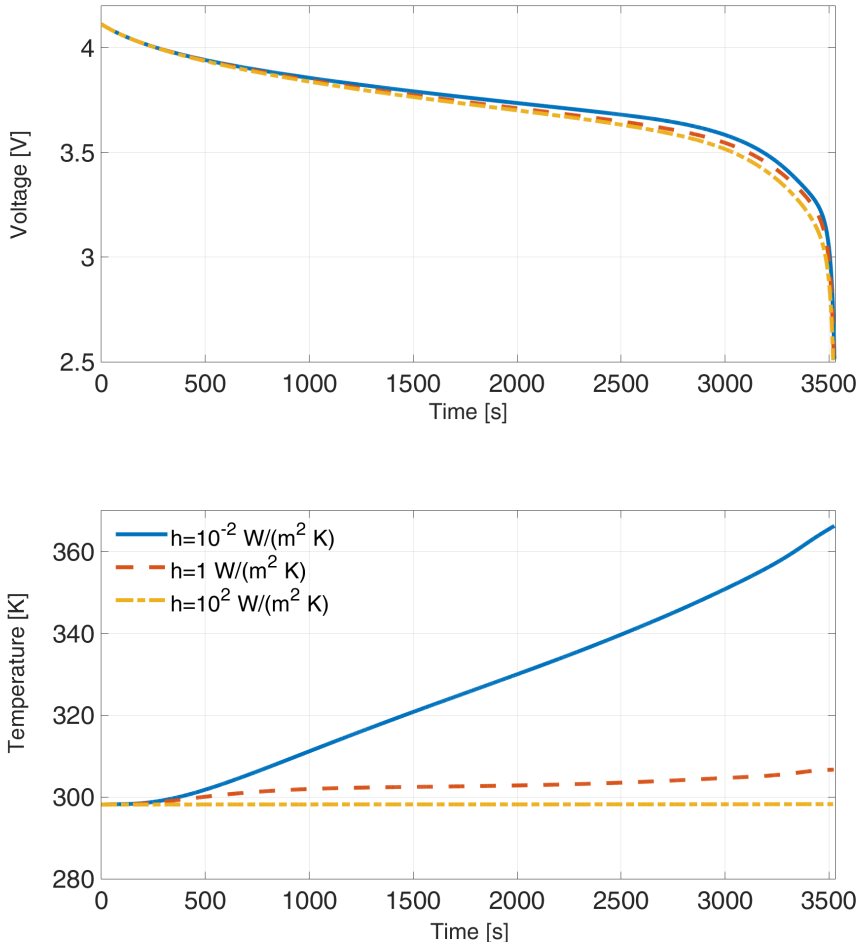


Figure 2.16: 1C discharge cycle run under different heat exchange parameters: blue line $h = 0.01\text{W}/(\text{m}^2\ \text{K})$, dashed orange line $h = 1\ \text{W}/(\text{m}^2\ \text{K})$ and dot-dashed yellow line $h = 100\ \text{W}/(\text{m}^2\ \text{K})$.

In the second scenario (Fig. 2.17), for a fixed value of $h = 1$ W/(m^{2}K), different discharge cycles are compared at 0.5C, 1C, and 2C. According to the different applied currents, the temperature rises in different ways; it is interesting to note the high slope of the temperature during a 2C discharge, mainly due to the electrolyte concentration c_e being driven to zero in the positive electrode by the high discharge rate.

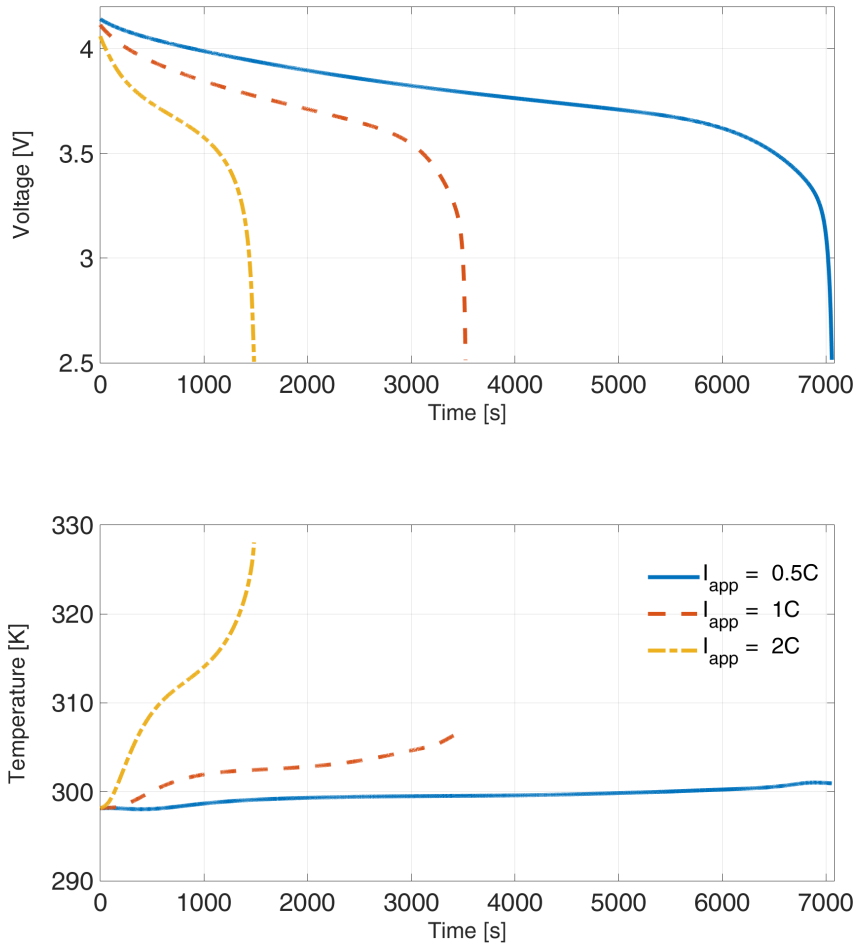


Figure 2.17: Full discharge cycle run under different C rates: 2C (dot-dashed yellow), 1C (dashed orange line), and 0.5C (blue line).

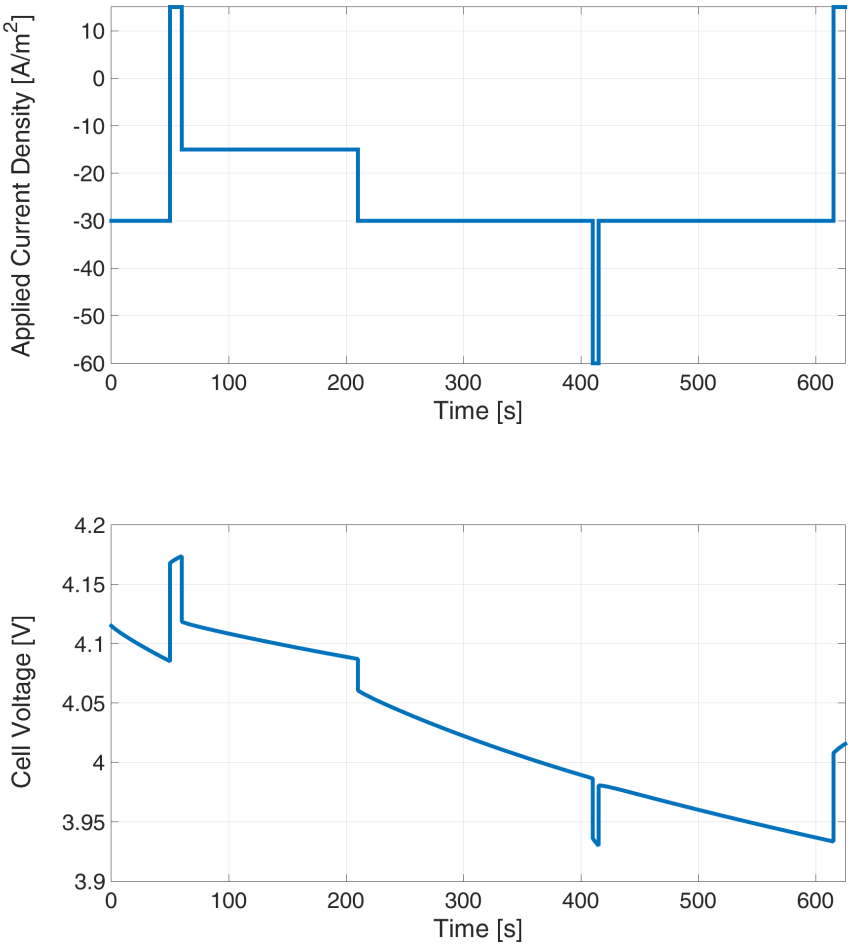
Time (s)	C rate	Description
0–50	−1 C	Moderate speed
50–60	0.5 C	Charge
60–210	−0.5C	Normal speed
210–410	−1C	Moderate speed
410–415	−2C	Overtaking
415–615	−1C	Moderate speed
615–620	0.5C	Charge

Table 2.7: Throttle configuration for hybrid charging-discharging simulation

2.7.2 HEV throttle simulation

In the third scenario, the framework is used to simulate a hybrid charge-discharge cycle, emulating the throttle of a HEV. During braking, the battery is charged. Table 2.7 resumes the configuration of the car throttle during simulations. In Fig. 2.18 it is possible to analyze the response of a single cell inside an HEV pack under a hybrid charge-discharge cycle. In this case, the effects of temperature among the different cells have been neglected. The solid potential behavior is primarily due to the different applied C rates, with discontinuous changes producing voltage drops. Different slopes of the voltage curve are related to the different C rates applied. Temperature rise is recorded in the first 50 seconds of simulations, which are followed by a slight decrease of the temperature mainly due to the exchange of heat with the surrounding environment ($h = 1$ W/(m^{2}K)) and due to the lower current density applied. At around 250 s, the temperature starts to increase due to the 1C rate applied during moderate speed; the high slope of increase at around 410 s is due to the higher value of the discharge current which during an overtake reaches the value of 2C. Returning to

moderate speed makes the temperature slope more gentle. During the last 10 seconds, temperature decreases due to the significant change in applied current and due to dissipation of heat with surrounding ambient. A sketch of the code used for this simulation is presented in Section 2.7.5, algorithm 2.



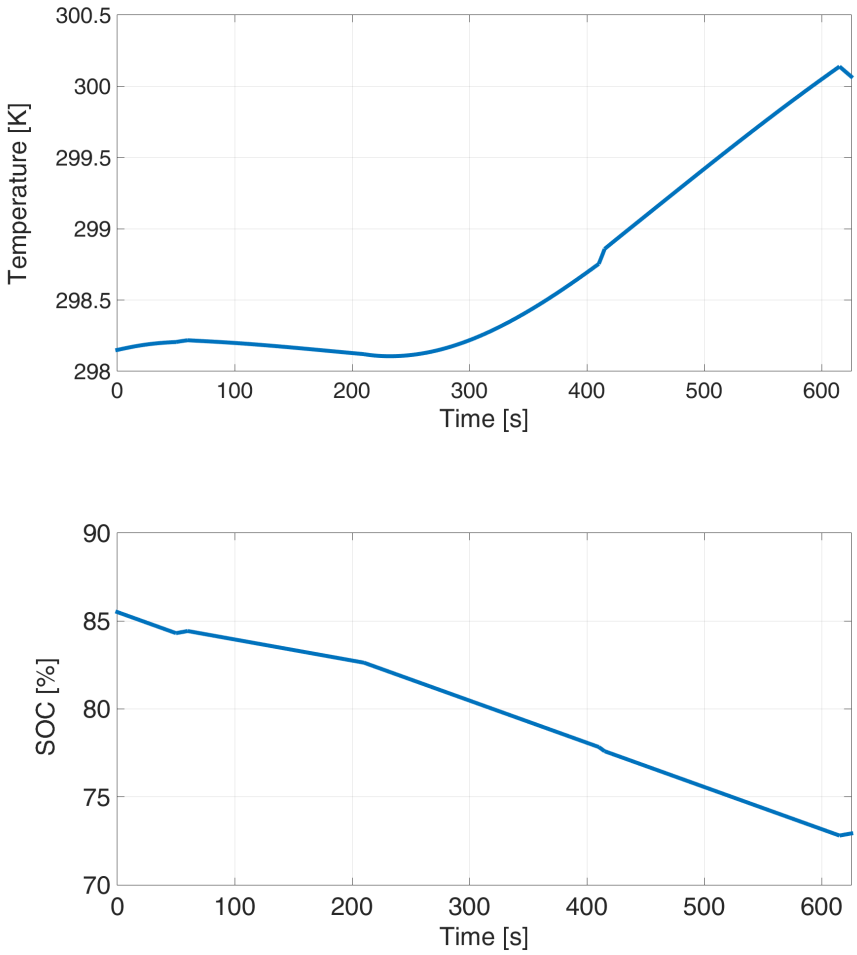


Figure 2.18: Hybrid charging-discharging cycle.

2.7.3 Isothermal simulations

LIONSIMBA allows to evaluate the dynamics of a Li-ion cell in isothermal conditions. According to this simulation setup, the thermal dynamics are neglected and the temperature index is kept fixed to the environmental value. This particular scenario can be exploited in order to assess the influence of different constant environmental temperatures at which the battery can operate. A set of isothermal simulations is depicted in Fig. 2.19, where several discharges at different C rates are presented. When thermal dynamics are disabled, the user can define custom versions of the functions used to obtain the values of diffusion end conductivity coefficients.

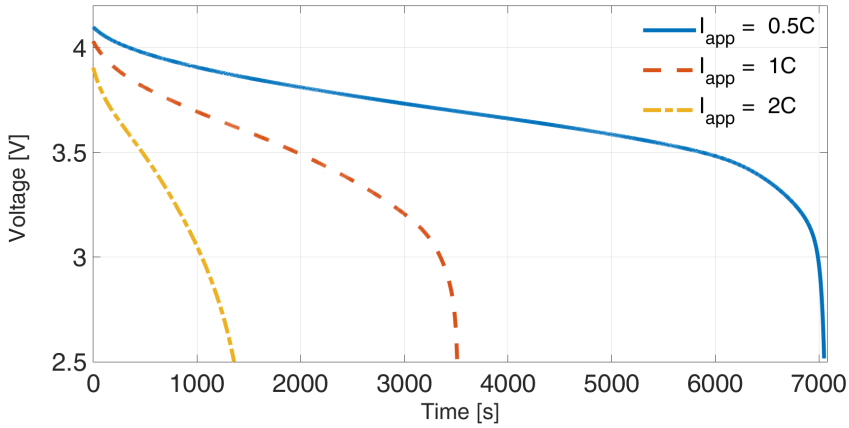


Figure 2.19: Full discharge cycle in an isothermal environment: blue line 0.5C, dashed orange line 1C, and dot-dashed yellow line 2C.

All the results of the proposed simulations can be reproduced by running the example scripts available with LIONSIMBA. To further emphasize the power of the proposed simulator, in Table 2.8 the times required by the software to simulate the different scenarios are presented. As it is possible to see, even in the presence of a wide variety of simulation conditions, the times required to solve the entire simulation a re under the 100s[2]

C rate	h value	Simulation Duration	Effective Simulation Time
1C	0.01	3523 s	72 s
1C	1	3523 s	81 s
1C	100	3523 s	77 s
0.5C	1	7050 s	56 s
2C	1	1522 s	85 s

Table 2.8: Timing comparisons of different simulation scenarios

2.7.4 Battery pack of series-connected cells

The results of a battery pack simulation are shown in Figs. 2.20 and 2.21. To emphasize the ability to independently parameterize each cell, in this scenario the SOC of cell #1 is set to the 95% of its initial value while the thickness of the cathode of cell #2 is doubled with respect to its initial value. All the other parameters are the same for the three cells. The time responses of the output voltage of the overall pack and the volume of each cell is plotted in Fig. 2.20. The starting voltage of the pack is around 12.1 V and decreases subjected to a 1C discharge current. In 3346 s, the pack is completely discharged due to cell #1 first reaching the cutoff voltage (set to

[2]These results were obtained considering 20 CVs in each battery section, with tolerances of 10^{-7} and neglecting the usage of the Jacobian matrix for the integration process. In Section 2.7.6 the adoption of an analytical Jacobian of the P2D model is considered and its contribution to the simulation performance highlighted.

2.5 V). The lowered starting SOC determined this behavior. The electrolyte and solid-phase surface concentrations as well as the electrolyte potentials are compared for the three cells in Fig. 2.21. Cell #2 has a significantly different behavior mainly due to the presence of a cathode with a thickness two times that of the other two cells. This variation has effects over the output voltages, as shown in Fig. 2.20. Besides cell # 1 which is starting from a different SOC value, the different behaviors of $V_{\text{out}}(t)$ between cell #2 and cell #3 are driven by the thickness variation.

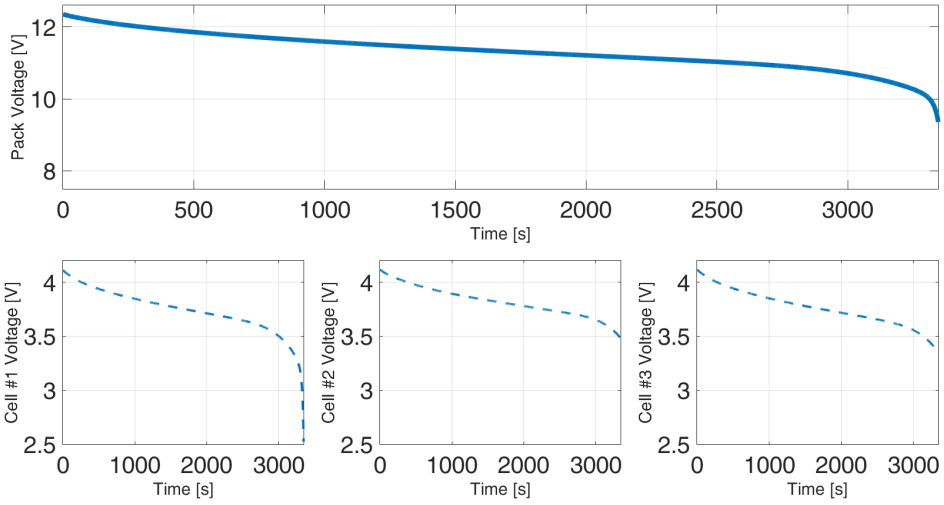


Figure 2.20: Simulation of a 3-cell pack. The upper curve represents the overall voltage of the 3 series connected Li-ion cells, while the lower plots depict the voltage of each cell in the pack. The different parametrization of each cell determines different behaviors.

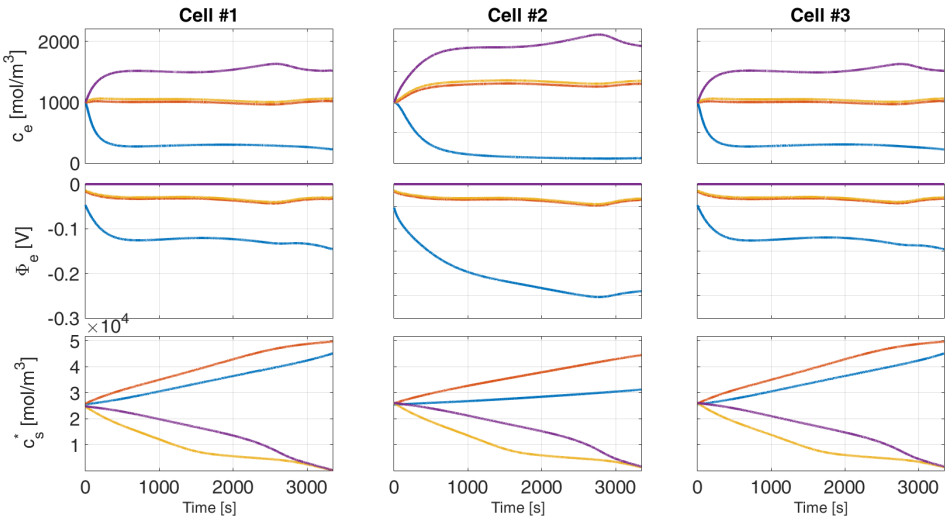


Figure 2.21: Simulation of a 3-cell pack. The profiles of different internal states inside the three cells. Individual parametrizations lead to different behaviors.

2.7.5 Example algorithms

Algorithm 2 Car cycling example code

Input setup:

I = $\{-29.5, 14.75, -14.75, -29.5, -58, -29.5, 14.75\}$ ▷ Simulation current densities

time = $\{50, 10, 150, 200, 5, 200, 10\}$ ▷ Duration of each element of $I_{applied}$ (in seconds).

t0 = 0; ▷ Init all the useful variables

tf = 0;

initialStates.Y = [ ];

initialStates.YP = [ ];

Phis_tot = [ ];

t_tot = [ ];

T_tot = [ ];

Core script:

for i = 1:length(I) **do**

    tf = tf + time(i);

    results = startSimulation(t0,tf,initialStates,I(i),[ ]);

    Phis_tot = [Phis_tot;results.original.Phis]; ▷ Concatenate results

    T_tot=[T_tot;results.original.Temperature];

    t0 = time(i);

    initialStates = results.initialStates; ▷ Update initial states for the next simulation

end for

Algorithm 3 High level control code

Init script:

```
t0 = 0;
tf = dt;                          ▷ Simulations are run over a sampling time periods
initialStates.Y = [ ];
initialStates.YP = [ ];
Condition = 1;
```

Core script:

```
while Condition do
    I = ComputeControlLaw(initialStates);
    results = startSimulation(t0,tf,initialStates,I,[ ]);
    [...]  ▷ Elaborate and concatenate the results and update the time indices
    initialStates = results.initialStates;     ▷ Update initial states for the next simulation
    if SOC reached reference value then
        Condition = 0
    end if
end while
```

2.7.6 Automatic differentiation support

LIONSIMBA can significantly speed-up its performance by means of automatic differentiation tools. In particular, CasADi [Andersson et al., 2012] is integrated with LIONSIMBA toolbox and provides functionality to obtain the analytical Jacobian matrix of the implemented P2D model. Even though the model would be modified by the user, LIONSIMBA will automatically provide the updated Jacobian for the new model or new parametrization. Once the Jacobian matrix has been evaluated, the user can feed it directly to LIONSIMBA avoiding to recompute it every simulation run. To highlight the potential of exploiting a Jacobian matrix, several 1C full discharge scenarios have been carried out, and their timings are reported in Table 2.9. As it is possible to notice, the inclusion of the knowledge coming from the Jacobian matrix significantly boosts the performance of LIONSIMBA, and in some case (i.e., the 100 cells scenario) allows to complete the simulation (something that it would not be possible without providing such information to the numerical integrator)[3].

	1 cell	**3 cells pack**	**100 cells pack**
With Jacobian	2.46 s	6.23 s	1100 s
Without Jacobian	14.8 s	144 s	N.A.

Table 2.9: Simulation scenarios run with and without the analytical Jacobian.

[3]Differently from the comparison carried out in Table 2.8, the presented results were obtained considering 10 CVs in each section of the cell, with integration tolerances of 10^{-6}

2.8 Conclusions

This chapter describes a detailed procedure for the numerical implementation of the P2D model [Doyle, 1995]. Two published approximate models for the solid-phase diffusion are also implemented, in which the pseudo-second dimension is removed to reduce the computational complexity. The treatment of boundary conditions is addressed with particular attention to the interface conditions across the different sections of the battery. Following the procedures and rules outlined in Section 2.5, the reader can implement his/her own version of the model in different programming languages. Moreover, a freely available Matlab® framework LIONSIMBA is provided that is suitable for battery design, simulation, and control. The framework is extended to account for different solid-phase diffusion models to meet required accuracy. The simulations demonstrate high numerical stability for different operating scenarios. The effectiveness of LIONSIMBA is verified considering a heterogeneous sequence of applied current coming from an HEV.

	# of discrete nodes				
	10	**20**	**30**	**40**	**50**
COMSOL	96 s	114 s	143 s	189 s	244 s
DUALFOIL	28 s	57 s	97 s	137 s	185 s
LIONSIMBA	28 s	69 s	105 s	134 s	223 s

Table 2.10: Timing comparisons among different P2D model implementations. The number of discretized nodes has been set equal for each section of the cell.

A battery pack composed of series-connected cells can be simulated by considering several independent cells with their own parameters. Due to its integration with the Matlab® environment, the framework facilitates the

development and test of different algorithms such as control algorithms, identification procedures, or optimization of manufacturing parameters. The computational times of LIONSIMBA (run without making explicit use of the Jacobian matrix) are compared to DUALFOIL and COMSOL in Table 2.10. For each code, average simulation times are reported for repeated simulations of a 1C discharging cycle in isothermal conditions. LIONSIMBA and DUALFOIL have very similar computation times for all discretizations, with COMSOL being significantly slower at lower discretizations. For the same numerical algorithm, an implementation in a compiled language such as Fortran is inherently much faster than an interpreted language such as Matlab® , indicating that the underlying numerical algorithm used by LIONSIMBA is more efficient but the higher efficiency is offset by Matlab® being an interpreted language: the two effects approximately cancel so that the overall simulation times for LIONSIMBA and DUALFOIL were very similar. Moreover, DUALFOIL runs with a self-implemented DAE solver which does not exploits all the advantages of the modern variable-step solvers (as, for instance, the IDA of the SUNDIALS suite). Finally, the numerical implementation of LIONSIMBA has been carried out with the objective of optimizing the simulation performance. This aspect has supported LIONSIMBA in performing as well as (or even better of) the commercial software COMSOL. Since COMSOL is a suite for the resolution of a wider variety of numerical problems, it is reasonable to think that it was not particularly optimized for the resolution of the P2D dynamics.

The results in this chapter demonstrate the promise of the proposed framework as a reliable, efficient, and freely available Matlab-based soft-

ware for the P2D model simulation. Further developments such as code optimization and distribution of compiled versions can only improve the current performance. The only inclusion of the Jacobian matrix knowledge resulted in a significant increase in performance. Moreover, as the proposed simulations were written in standard serial mode, the computation time could be reduced by at least a factor of ten by using a multicore CPU using parallel DAE solvers. Modern versions of Matlab® have easy-to-implement built-in options for distributing calculations among multiple cores on a single CPU, and among multiple CPUs.

Chapter 3

Model Predictive Control

Contents

3.1 Introduction

Model predictive control refers to a particular class of algorithms which make explicit use of a mathematical model that approximates the dynamics of a process, whose behavior is optimized over a future horizon while operational constraints on inputs and outputs are taken into account [Maciejowski, 2002]. In particular, as will be explained in more detail in this section, the process inputs are optimized over a prediction window and only the first element of such sequence is applied to the controlled plant. Due to the mismatch between the process model and the process itself, after the

application of the control action, the states of the plant are measured and at the next time instant the problem is solved again with the above procedure iterated. In Fig. 3.1 an high level control scheme involving a MPC algorithm is depicted. The first idea of MPC algorithms was given by [Propoi,

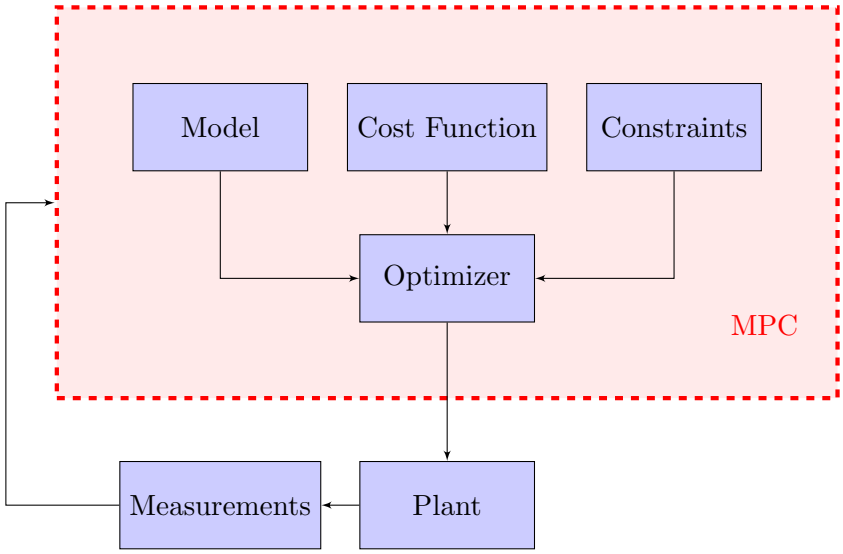


Figure 3.1: High-level schematic of a MPC algorithm

1963], while a first independent version of this control paradigm was developed in the early 70s by Shell Oil engineers. Due to the requirement of solving online optimization problems, together with the poor computational power at that time, this control paradigm was initially employed in petrochemical processes which had slow dynamics. Through the years, however, the ever increasing computational power has opened new perspectives for the employment of MPC algorithms into a wider variety of industrial applications [Qin and Badgwell, 2003, Camacho and Bordons, 2012]. The key points which made the MPC paradigm a successful control algorithm have been the following:

- its ability to handle with generic nonlinear and Multi-Input Multi-Output (MIMO) processes;

- its capability of taking into account operating constraints, whose advantage is twofold: (i) it helps to prevent possible safety issues and damages to the plant, and (ii) it brings the plant to operate in optimum conditions, which have been demonstrated to be obtained at the intersection of operational constraints [Prett and Gillette, 1980];

- it provides a control law that is obtained by optimizing a given cost function, which can reflect some economical index.

In order to formulate an MPC control algorithm, consider the following discrete-time nonlinear model used to describe a plant dynamics

$$\begin{aligned} \boldsymbol{\xi}(k+1) &= \boldsymbol{f}(\boldsymbol{\xi}(k), \boldsymbol{u}(k), k), \\ \boldsymbol{y}(k) &= \boldsymbol{g}(\boldsymbol{\xi}(k), \boldsymbol{u}(k), k), \end{aligned} \tag{3.1}$$

where $\boldsymbol{f}(\cdot)$ and $\boldsymbol{g}(\cdot)$ are nonlinear algebraic functions used to represent the evolution of the states $\boldsymbol{\xi} \in \mathbb{R}^{n_\xi}$ and the outputs $\boldsymbol{y} \in \mathbb{R}^{n_y}$ respectively as a function of the inputs $\boldsymbol{u} \in \mathbb{R}^{n_u}$, while k represents the discrete time instant. Introduce the sequences

$$\begin{aligned} \tilde{\boldsymbol{\xi}}_{k+1:k+H_p|k} =& [\boldsymbol{\xi}(k+1|k)^\top \cdots \boldsymbol{\xi}(k+H_p|k)^\top]^\top, \\ \tilde{\boldsymbol{u}}_{k:k+H_u|k} =& [\boldsymbol{u}(k|k)^\top \cdots \boldsymbol{u}(k+H_u|k)^\top]^\top, \\ \tilde{\boldsymbol{y}}_{k:k+H_p|k} =& [\boldsymbol{y}(k|k)^\top \cdots \boldsymbol{y}(k+H_p|k)^\top]^\top, \end{aligned} \tag{3.2}$$

where H_p and H_u are the prediction and control horizons, respectively, and $\boldsymbol{x}(i|j)$ denotes the value of $\boldsymbol{x}$ at time instant i starting from time instant j. At every discrete time instant k, MPC provides an optimal input sequence

by solving an online optimization problem,

$$\min_{\tilde{\boldsymbol{u}}_{k:k+H_u|k}} \; J(\tilde{\boldsymbol{u}}_{k:k+H_u|k}, \boldsymbol{\xi}_{\mathbf{init}}) \tag{3.3}$$

subject to

$$\begin{aligned}
\boldsymbol{\xi}(j+1|k) &= \boldsymbol{f}(\boldsymbol{\xi}(j|k), \boldsymbol{u}(j|k), j) \\
\boldsymbol{y}(j|k) &= \boldsymbol{g}(\boldsymbol{\xi}(j|k), \boldsymbol{u}(j|k), j) \\
\boldsymbol{\xi}(k|k) &= \boldsymbol{\xi}_{\mathbf{init}} \\
\boldsymbol{u}_{\min} &\leq \boldsymbol{u}(j|k) \leq \boldsymbol{u}_{\max} \\
\boldsymbol{\xi}_{\min} &\leq \boldsymbol{\xi}(j|k) \leq \boldsymbol{\xi}_{\max} \\
\boldsymbol{y}_{\min} &\leq \boldsymbol{y}(j|k) \leq \boldsymbol{y}_{\max}
\end{aligned}$$

where $j \in [k; k+H_p]$,

$$J(\tilde{\boldsymbol{u}}_{k:k+H_u|k}, \boldsymbol{\xi}_{\mathbf{init}}) = ||\tilde{\boldsymbol{u}}_{k:k+H_u|k} - \tilde{\boldsymbol{u}}_{\text{ref}}||^2_{\boldsymbol{R}} + ||\tilde{\boldsymbol{y}}_{k:k+H_p|k} - \tilde{\boldsymbol{y}}_{\text{ref}}||^2_{\boldsymbol{Q}}, \tag{3.4}$$

while $\tilde{\boldsymbol{u}}_{\text{ref}}$, and $\tilde{\boldsymbol{y}}_{\text{ref}}$ are reference setpoints (or trajectories) for the inputs and outputs respectively. The notation $||\boldsymbol{x}||_{\boldsymbol{Q}} = ||\boldsymbol{Q}^{1/2}\boldsymbol{x}||_2$ denotes the weighted 2-norm, $\boldsymbol{Q} \in \mathbb{R}^{H_p n_y \times H_p n_y}$ is a positive semidefinite matrix that weights the deviation of the outputs from their references, and $\boldsymbol{R} \in \mathbb{R}^{H_u n_u \times H_u n_u}$ is a positive definite matrix that accounts for the deviation of the inputs with respect to their reference value. The cost function in (3.4) can be modified in order to consider states rather than outputs, and account for input differences as will be explained in the next sections. The solution of the optimization provides an optimal control sequence $\tilde{\boldsymbol{u}}^*_{k:k+H_u|k}$. Given that $\tilde{\boldsymbol{u}}^*_{k:k+H_u|k}$ is obtained based only on the predictions of a mathematical model, its entire application to the plant would not reject unknown disturbances to the system. For this reason, the so-called Receding Horizon (RH) approach is adopted where, at each time step k, only the first

element of the optimal control sequence is applied to the plant [Kwon and Han, 2006, Mattingley et al., 2011],

$$\boldsymbol{u}_{\mathrm{RH}}(k) = \boldsymbol{u}^*(k|k). \tag{3.5}$$

At the next time instant, the array of the initial states $\boldsymbol{\xi}_{\mathbf{init}}$ is updated with the new measurements and a new optimization is solved. For computational reasons, it is common to define $H_u \leq H_p$ in order to reduce the number of optimization variables. When $H_u < H_p$, the control moves can be kept fixed after the $k+H_u$ (i.e., $\boldsymbol{u}(k+H_u|k) = \boldsymbol{u}(k+H_u+1|k) = \cdots = \boldsymbol{u}(k+H_p|k)$).

3.2 Models for control

As discussed in the previous section, MPC algorithms make use of mathematical models of the controlled plant in order to carry out the control action. In particular, the problem (3.3) has been introduced by considering the model (3.1) as described by means of generic nonlinear algebraic equations. However, even in the presence of a high computational power, the online application of MPC can become intractable when high order systems or strong nonlinear dynamics are considered [Mayne, 2000, Magni et al., 2009]. Such intractability derives mainly from the need to solve nonlinear optimization problems at each time step. In order to overcome these limitations, several strategies can be employed. Among the possible solutions, linearization and order reduction techniques have been widely adopted in literature [Kokotovic et al., 1976, Mesbah et al., 2015, Geuss et al., 2015, Saraswat and Parmar, 2015, Kuhne et al., 2004, Simon et al., 2013, Deng et al., 2009]. In the following, several approximations of (3.1) are presented. In particular, such approximations help to reduce the com-

putational complexity in solving (3.3) while providing good control performance.

3.2.1 State-space models

Linear time invariant models

Suppose that the states and outputs of the nonlinear system in (3.1) are in steady conditions $\boldsymbol{\xi}_{\mathbf{ss}}$, $\boldsymbol{y}_{\mathbf{ss}}$, as a result of the application of the steady inputs $\boldsymbol{u}_{\mathbf{ss}}$. By deriving the first order Taylor approximation of (3.1) around ($\boldsymbol{u}_{\mathbf{ss}}$,$\boldsymbol{\xi}_{\mathbf{ss}}$), one obtains

$$\boldsymbol{\xi}(k+1) \approx \boldsymbol{f}(\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}) + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\xi}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} (\boldsymbol{\xi}(k) - \boldsymbol{\xi}_{\mathbf{ss}}) + \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} (\boldsymbol{u}(k) - \boldsymbol{u}_{\mathbf{ss}}),$$

$$\boldsymbol{y}(k) \approx \boldsymbol{g}(\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}) + \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} (\boldsymbol{\xi}(k) - \boldsymbol{\xi}_{\mathbf{ss}}) + \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} (\boldsymbol{u}(k) - \boldsymbol{u}_{\mathbf{ss}}),$$

which can be rewritten as a Linear Time Invariant (LTI) system of the form

$$\begin{aligned} \boldsymbol{\delta\xi}^{\mathbf{lin}}(k+1) &= \boldsymbol{A}\boldsymbol{\delta\xi}^{\mathbf{lin}}(k) + \boldsymbol{B}\boldsymbol{\delta u}^{\mathbf{lin}}(k), \\ \boldsymbol{\delta y}^{\mathbf{lin}}(k) &= \boldsymbol{C}\boldsymbol{\delta\xi}^{\mathbf{lin}}(k) + \boldsymbol{D}\boldsymbol{\delta u}^{\mathbf{lin}}(k). \end{aligned} \tag{3.6}$$

The dynamical matrices are defined as

$$\begin{aligned} \boldsymbol{A} &= \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\xi}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} \in \mathbb{R}^{n_\xi \times n_\xi}, & \boldsymbol{B} &= \left.\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} \in \mathbb{R}^{n_\xi \times n_u}, \\ \boldsymbol{C} &= \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{\xi}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} \in \mathbb{R}^{n_y \times n_\xi}, & \boldsymbol{D} &= \left.\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{u}}\right|_{\boldsymbol{\xi}_{\mathbf{ss}}, \boldsymbol{u}_{\mathbf{ss}}} \in \mathbb{R}^{n_y \times n_u}, \end{aligned}$$

while the quantities $\boldsymbol{\delta\xi}^{\mathbf{lin}}(k)$, $\boldsymbol{\delta u}^{\mathbf{lin}}(k)$, and $\boldsymbol{\delta y}^{\mathbf{lin}}(k)$ account, respectively, for the deviation of the states, inputs and outputs with respect to their equilibrium values. The above formulation approximates the dynamics of (3.1) in a neighborhood of ($\boldsymbol{u}_{\mathbf{ss}}$,$\boldsymbol{\xi}_{\mathbf{ss}}$)

Linear time varying models

As previously discussed, the linearized dynamics (3.6) can be used to describe the dynamical behavior of the nonlinear system (3.1) in a neighborhood of the linearization points ($\boldsymbol{u}_{\mathbf{ss}}$,$\boldsymbol{\xi}_{\mathbf{ss}}$). The accuracy of such approximation strictly depends on two main factors: (i) the type of the nonlinearities in (3.1), and (ii) the norm of the deviation with respect to the equilibrium points. In order to improve the prediction accuracy, another family of linear models can be adopted. In particular, instead of linearizing (3.1) around an equilibrium point, the nonlinear dynamics are linearized around a set of nominal trajectories of the inputs and the states. This approximation leads to a Linear Time Varying (LTV) formulation.

Consider a nominal input sequence $\tilde{\boldsymbol{u}}^{\mathbf{n}} \in \mathbb{R}^{n_u\, H_{\bar{u}}}$, where $H_{\bar{u}}$ is the length of such sequence, which, if fed into the system, results in a set of nominal states $\tilde{\boldsymbol{\xi}}^{\mathbf{n}} \in \mathbb{R}^{n_\xi\, H_{\bar{u}}}$ and outputs $\tilde{\boldsymbol{y}}^{\mathbf{n}} \in \mathbb{R}^{n_y\, H_{\bar{u}}}$. By linearizing (3.1) around the nominal trajectories ($\tilde{\boldsymbol{u}}^{\mathbf{n}}, \tilde{\boldsymbol{\xi}}^{\mathbf{n}}$), the LTV dynamics

$$\begin{aligned} \boldsymbol{\delta\xi}^{\mathbf{n}}(k+1) &= \boldsymbol{A}(k)\boldsymbol{\delta\xi}^{\mathbf{n}}(k) + \boldsymbol{B}(k)\boldsymbol{\delta u}^{\mathbf{n}}(k), \\ \boldsymbol{\delta y}^{\mathbf{n}}(k) &= \boldsymbol{C}(k)\boldsymbol{\delta\xi}^{\mathbf{n}}(k) + \boldsymbol{D}(k)\boldsymbol{\delta u}^{\mathbf{n}}(k), \end{aligned} \tag{3.7}$$

are obtained, where

$$\begin{aligned} \boldsymbol{A}(k) &= \left.\frac{d\boldsymbol{f}}{d\boldsymbol{\xi}}\right|_{\tilde{\boldsymbol{\xi}}^{\mathbf{n}}(k),\tilde{\boldsymbol{u}}^{\mathbf{n}}(k)} \in \mathbb{R}^{n_\xi \times n_\xi}, \qquad \boldsymbol{B}(k) = \left.\frac{d\boldsymbol{f}}{d\boldsymbol{u}}\right|_{\tilde{\boldsymbol{\xi}}^{\mathbf{n}}(k),\tilde{\boldsymbol{u}}^{\mathbf{n}}(k)} \in \mathbb{R}^{n_\xi \times n_u}, \\ \boldsymbol{C}(k) &= \left.\frac{d\boldsymbol{g}}{d\boldsymbol{\xi}}\right|_{\tilde{\boldsymbol{\xi}}^{\mathbf{n}}(k),\tilde{\boldsymbol{u}}^{\mathbf{n}}(k)} \in \mathbb{R}^{n_y \times n_\xi}, \qquad \boldsymbol{D}(k) = \left.\frac{d\boldsymbol{g}}{d\boldsymbol{u}}\right|_{\tilde{\boldsymbol{\xi}}^{\mathbf{n}}(k),\tilde{\boldsymbol{u}}^{\mathbf{n}}(k)} \in \mathbb{R}^{n_y \times n_u}, \end{aligned}$$

and the terms $\boldsymbol{\delta\xi}^{\mathbf{n}}(k) = \boldsymbol{\xi}(k) - \tilde{\boldsymbol{\xi}}^{\mathbf{n}}(k)$, $\boldsymbol{\delta u}^{\mathbf{n}}(k) = \boldsymbol{u}(k) - \tilde{\boldsymbol{u}}^{\mathbf{n}}(k)$, and $\boldsymbol{\delta y}^{\mathbf{n}}(k) = \boldsymbol{y}(k) - \tilde{\boldsymbol{y}}^{\mathbf{n}}(k)$ account for the deviations of the linearized dynamics with respect to the nominal trajectories.

3.2.2 Input-output models

The LTI and LTV reformulations are suitable approximations that can be used to reduce the complexity of a nonlinear system by describing linearly its behavior in a neighborhood of an equilibrium point or in a neighborhood of a set of nominal trajectories. Despite this, the resulting approximations can become computationally expensive when high order systems are considered. For instance, when dynamical systems described by means of PDAEs are considered, their DAEs reformulation generally leads to high-dimensional models (see e.g. Section 2.5.1) [Foguth et al., 2015]. To overcome these limitations, linear input-output models can be adopted to cope with both: (*i*) the system nonlinearities, and (*ii*) the possible high-order structure. In the following the prediction schemes of the input-output models used in this Thesis are presented, and their formulation discussed.

Finite step response models

Finite Step Response (FSR) models are a particular family of linear input-output models that have been widely adopted in industrial applications [Yazuzturk and Spitler, 1999, Moon and Lee, 2009]. According to the FSR formulation, when a Single-Input Single-Output (SISO) system is considered, the output prediction scheme at time step k is given by

$$y^1(k) = y^1_{\text{ss}} + \sum_{g=1}^{N-1} S^g_{1,1}\Delta u(k-g) + S^N_{1,1}(u(k-N) - u^1_{\text{ss}}), \tag{3.8}$$

where u^1_{ss} and y^1_{ss} are respectively the steady-state input and output values. The FSR coefficients relating the bth output with the ith input are denoted by $S^g_{i,b}$ while $\Delta u(k) := u(k) - u(k-1)$. Notice that, for asymptotically stable systems, N represents the settling time (i.e. $y(k+N) \approx y(k+N+1) \approx$

$\cdots \approx y(\infty)$), while for integrating dynamics it represents the time instant at which the outputs become pure ramps. When considering a MIMO system, the following FSR formulation for the outputs prediction is obtained

$$
\begin{aligned}
y^1(k) &= y^1_{\text{ss}} + \sum_{i=1}^{n_u} \left(\sum_{g=1}^{N-1} S^g_{i,1} \Delta u^i(k-g) + S^N_{i,1}(u^i(k-N) - u^i_{\text{ss}}) \right), \\
y^2(k) &= y^2_{\text{ss}} + \sum_{i=1}^{n_u} \left(\sum_{g=1}^{N-1} S^g_{i,2} \Delta u^i(k-g) + S^N_{i,2}(u^i(k-N) - u^i_{\text{ss}}) \right), \\
&\vdots \\
y^{n_y}(k) &= y^{n_y}_{\text{ss}} + \sum_{i=1}^{n_u} \left(\sum_{g=1}^{N-1} S^g_{i,n_y} \Delta u^i(k-g) + S^N_{i,n_y}(u^i(k-N) - u^i_{\text{ss}}) \right).
\end{aligned}
$$

This formulation provides a prediction of the outputs by means of a linear combination of input moves and FSR coefficients.

Remark: FSR models cannot be derived for unstable systems, they can be only employed to represent the dynamics of asymptotically stable or stable systems as discussed in [Kwon and Han, 2006].

Autoregressive exogenous models

AutoRegressive eXogenous (ARX) models are another particular family of input-output affine models [Ljung, 1998]. The one-step ahead output prediction scheme of SISO system according to the ARX formulation is given by

$$
y(k+1) = \boldsymbol{\theta} \begin{bmatrix} \boldsymbol{s}(k) \\ 1 \end{bmatrix}, \tag{3.9}
$$

where $\boldsymbol{s}(k) \in \mathbb{R}^n$ is the array of *regressors*, $\boldsymbol{\theta} \in \mathbb{R}^{n+1}$ is the set of parameters. For a given time instant, $\boldsymbol{s}(k)$ is defined as

$$\begin{aligned}\boldsymbol{s}(k) = [&y(k)\ y(k-1)\ \cdots\ y(k-n_a+1)\\ &u(k+1-n_k)\ u(k-n_k)\ \cdots\ u(k-n_k-n_b+2)],\end{aligned}$$

which is composed of the past n_a output values and the past n_b input values, where $n_k \geq 0$ accounts for the *relative degree* of the system, and $n = n_a + n_b$. The model (3.9) can be straightforwardly extended to the MIMO case. Consider the matrices $\boldsymbol{N_a} \in \mathbb{R}^{n_y \times n_y}$, $\boldsymbol{N_b} \in \mathbb{R}^{n_y \times n_u}$, and $\boldsymbol{N_k} \in \mathbb{R}^{n_y \times n_u}$, be analogous to the SISO indices n_a, n_b, and n_k respectively. The element of row i and column j of $\boldsymbol{N_a}$ accounts for the number of past values of the jth output contributing to the dynamics of the ith output of the system. Similarly the effects of the mth input over the ith output are given by the element of $\boldsymbol{N_b}$ at row i and column m. Finally, the element of row i and column j of $\boldsymbol{N_k}$ accounts for the *relative degree* between input j and output i.

Piecewise affine autoregressive exogenous models

An extension of the ARX models is represented by the PieceWise affine ARX (PWARX) models. The main difference lies in the use of several ARX models defined over polyhedral regions. At each time instant, a switch among these regions determines what is the affine model used to predict the plant behavior. Define the *regressors set* $\mathcal{X} \subset \mathbb{R}^n$ as a bounded polyhedron that satisfies $\boldsymbol{s}(k) \in \mathcal{X}$ for all $\boldsymbol{s}(k)$ and is partitioned in L polyhedral subregions $\{\mathcal{X}_l\}_{l=1}^{L}$ (an example is given in Fig. 3.2). In the case of a SISO system, the following PWARX one-step ahead output prediction scheme is

obtained [Ferrari-Trecate et al., 2003]

$$y(k+1) = \begin{cases} \boldsymbol{\theta_1} \begin{bmatrix} \boldsymbol{s}(k) \\ 1 \end{bmatrix}, \text{ if } \boldsymbol{s}(k) \in \mathcal{X}_1 \\ \\ \boldsymbol{\theta_2} \begin{bmatrix} \boldsymbol{s}(k) \\ 1 \end{bmatrix}, \text{ if } \boldsymbol{s}(k) \in \mathcal{X}_2 \\ \\ \vdots \\ \\ \boldsymbol{\theta_L} \begin{bmatrix} \boldsymbol{s}(k) \\ 1 \end{bmatrix}, \text{ if } \boldsymbol{s}(k) \in \mathcal{X}_L \end{cases} \tag{3.10}$$

where $\boldsymbol{\theta}_l$ is the set of parameters related to the lth subregion, and the active dynamics (i.e., the set of parameters $\boldsymbol{\theta}_l$ used to predict the next value of the output) are chosen according to the membership of $\boldsymbol{s}(k)$ to a polyhedral partition element $\mathcal{X}_l$. The SISO formulation (3.10) can be straightforwardly extended to a general MIMO plant by considering the matrices $\boldsymbol{N_a} \in \mathbb{R}^{n_y \times n_y}$, $\boldsymbol{N_b} \in \mathbb{R}^{n_y \times n_u}$, and $\boldsymbol{N_k} \in \mathbb{R}^{n_y \times n_u}$ as in (3.9).

3.3 Model predictive control formulations

The MPC formulation in (3.3) represents a generic optimization problem, with a quadratic cost function and subject to nonlinear dynamics. The approximations of (3.1) introduced in the previous section, under suitable assumptions, allow to represent the MPC problem in a canonical form. In particular, it is possible to reformulate (3.3) in terms of a Quadratic Programming (QP) problem. The QP problems are of particular interest because their formulation guarantees the obtaining of a convex optimization

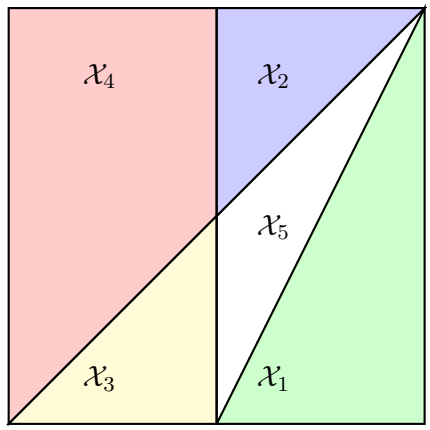


Figure 3.2: Example of a regressors set $\mathcal{X} \in \mathbb{R}^2$ partitioned in 5 different subregions.

problem that, as a result of its resolution, provides a global optimal solution. Due to the aforementioned properties, such optimization problems have been widely studied in literature and several tools have been developed to solve them in an efficient and reliable way [Boyd and Vandenberghe, 2004]. In the following, the derivation of the QP forms of (3.3) is discussed when LTI, LTV, ARX, PWARX, and FSR models are considered.

3.3.1 MPC formulation for LTI models

Consider the LTI approximation (3.6) of the nonlinear dynamics (3.1). Such a formulation allows to represent recursively the evolution of the states and outputs as function of the initial states and the control actions. This property can be exploited to derive a QP formulation of (3.3) subject to LTI dynamics. Starting from a given time instant k, the evolution of the

states can be written as

$$
\begin{aligned}
\boldsymbol{\delta\xi}^{\mathbf{lin}}(k+1|k) =& \boldsymbol{A\delta\xi}^{\mathbf{init}} + \boldsymbol{B\delta u}^{\mathbf{lin}}(k|k) \\
\boldsymbol{\delta\xi}^{\mathbf{lin}}(k+2|k) =& \boldsymbol{A\delta\xi}^{\mathbf{lin}}(k+1|k) + \boldsymbol{B\delta u}^{\mathbf{lin}}(k+1|k) = \\
=& \boldsymbol{A}\left[\boldsymbol{A\delta\xi}^{\mathbf{init}} + \boldsymbol{B\delta u}^{\mathbf{lin}}(k|k)\right] + \boldsymbol{B\delta u}^{\mathbf{lin}}(k+1|k) = \\
=& \boldsymbol{A}^2\boldsymbol{\delta\xi}^{\mathbf{init}} + \boldsymbol{AB\delta u}^{\mathbf{lin}}(k|k) + \boldsymbol{B\delta u}^{\mathbf{lin}}(k+1|k) \\
\boldsymbol{\delta\xi}^{\mathbf{lin}}(k+3|k) =& \boldsymbol{A}^3\boldsymbol{\delta\xi}^{\mathbf{init}} + \boldsymbol{A}^2\boldsymbol{B\delta u}^{\mathbf{lin}}(k|k) + \boldsymbol{AB\delta u}^{\mathbf{lin}}(k+1|k) + \\
&+ \boldsymbol{B\delta u}^{\mathbf{lin}}(k+2|k) \\
&\vdots \\
\boldsymbol{\delta\xi}^{\mathbf{lin}}(k+H_p|k) =& \boldsymbol{A}^{H_p}\boldsymbol{\delta\xi}^{\mathbf{init}} + \sum_{i=H_p}^{1} \boldsymbol{A}^{H_p-i}\boldsymbol{B\delta u}^{\mathbf{lin}}(k+i-1|k),
\end{aligned}
$$

where $\boldsymbol{\delta\xi}^{\mathbf{init}} = \boldsymbol{\xi}_{\mathbf{init}} - \boldsymbol{\xi}_{\mathbf{ss}}$ accounts for the initial states deviations. The same approach can be adopted to derive the prediction of the outputs over the future H_p steps as a function of the initial states $\boldsymbol{\delta\xi}^{\mathbf{init}}$ and control deviations $\boldsymbol{\delta u}^{\mathbf{lin}}(k+j|k)$, for $j \in [0; H_p]$. The states and outputs prediction schemes can be generalized in a compact form as:

$$
\begin{aligned}
\boldsymbol{\delta}\tilde{\boldsymbol{\xi}}_{k+1:k+H_p|k} =& \tilde{\boldsymbol{A}}\boldsymbol{\delta\xi}^{\mathbf{init}} + \tilde{\boldsymbol{B}}\boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}, \\
\boldsymbol{\delta}\tilde{\boldsymbol{y}}_{k:k+H_p|k} =& \tilde{\boldsymbol{C}}\boldsymbol{\delta\xi}^{\mathbf{init}} + \tilde{\boldsymbol{D}}\boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k},
\end{aligned} \tag{3.11}
$$

where $\tilde{\boldsymbol{A}} \in \mathbb{R}^{H_p n_\xi}$, $\tilde{\boldsymbol{B}} \in \mathbb{R}^{H_p n_\xi \times H_u n_u}$, $\tilde{\boldsymbol{C}} \in \mathbb{R}^{(H_p+1) n_y}$, and $\tilde{\boldsymbol{D}} \in \mathbb{R}^{(H_p+1) n_y \times H_u n_u}$. Assuming that $H_p = H_u$ one has

$$\tilde{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{A}^2 \\ \vdots \\ \boldsymbol{A}^{H_p} \end{bmatrix}, \quad \tilde{\boldsymbol{B}} = \begin{bmatrix} \boldsymbol{B} & 0 & \cdots & 0 \\ \boldsymbol{AB} & \boldsymbol{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}^{H_p-1}\boldsymbol{B} & \boldsymbol{A}^{H_p-2}\boldsymbol{B} & \cdots & \boldsymbol{B} \end{bmatrix},$$

$$\tilde{\boldsymbol{C}} = \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{CA} \\ \boldsymbol{CA}^2 \\ \vdots \\ \boldsymbol{CA}^{H_p} \end{bmatrix}, \quad \tilde{\boldsymbol{D}} = \begin{bmatrix} \boldsymbol{D} & 0 & \cdots & 0 \\ \boldsymbol{CB} & \boldsymbol{D} & \cdots & 0 \\ \boldsymbol{CAB} & \boldsymbol{CB} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{CA}^{H_p-1}\boldsymbol{B} & \boldsymbol{CA}^{H_p-2}\boldsymbol{B} & \cdots & \boldsymbol{D} \end{bmatrix}.$$

By making explicit the formulation of (3.11) into (3.4), neglecting all the terms which do not depend directly from $\boldsymbol{\delta u}^{\mathbf{lin}}(k)$, and bearing in mind that $\boldsymbol{y}(k) = \boldsymbol{\delta y}^{\mathbf{lin}}(k) + \boldsymbol{y}_{\mathbf{ss}}$ and $\boldsymbol{u}(k) = \boldsymbol{\delta u}^{\mathbf{lin}}(k) + \boldsymbol{u}_{\mathbf{ss}}$, the following QP formulation is derived

$$\min_{\boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}} \frac{1}{2} \boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}{}^{\top} \boldsymbol{H} \boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \boldsymbol{f}^{\top} \boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} \tag{3.12}$$

subject to

$$\underline{\boldsymbol{A}} \boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} \leq \underline{\boldsymbol{b}}$$

where

$$\boldsymbol{H} = (\boldsymbol{R} + \tilde{\boldsymbol{D}}^{\top} \boldsymbol{Q} \tilde{\boldsymbol{D}}),$$
$$\boldsymbol{f}^{\top} = (\boldsymbol{\delta\xi}^{\mathbf{init}\top} \tilde{\boldsymbol{C}}^{\top} \boldsymbol{Q} \tilde{\boldsymbol{D}} + \boldsymbol{y}_{\mathbf{ss}}{}^{\top} \boldsymbol{Q} \tilde{\boldsymbol{D}} - \tilde{\boldsymbol{y}}_{\mathrm{ref}}{}^{\top} \boldsymbol{Q} \tilde{\boldsymbol{D}} + \boldsymbol{u}_{\mathbf{ss}}{}^{\top} \boldsymbol{R} - \tilde{\boldsymbol{u}}_{\mathrm{ref}}{}^{\top} \boldsymbol{R}),$$

while the suitable matrix $\underline{\boldsymbol{A}} \in \mathbb{R}^{n_c \times n_u H_u}$, and vector $\underline{\boldsymbol{b}} \in \mathbb{R}^{n_c}$ are used to enforce inequality and equality constraints over the inputs, states and outputs. Note that the solution of (3.12) provides an optimal input deviation

sequence $\boldsymbol{\delta\tilde{u}}^*_{k:k+H_u|k}$. According to the RH approach only the first element of such sequence will be applied to the plant. Because of the meaning of $\boldsymbol{\delta\tilde{u}}^*_{k:k+H_u|k}$, before applying its first element to the plant, the steady values of the inputs have to be added, i.e.

$$\boldsymbol{u}_{\mathrm{RH}}(k) = \boldsymbol{\delta u}^*_{\mathbf{lin}}(k|k) + \boldsymbol{u}_{\mathbf{ss}}.$$

At the next time instant, the initial states $\boldsymbol{\delta\xi}^{\mathbf{init}}$ are updated according to the available measurements, and the procedure iterated.

3.3.2 MPC formulation for LTV systems

When a LTV system of the form (3.7) is considered with the cost function (3.4), the optimization (3.3) is still a QP. Starting from time instant k, the following LTV prediction scheme can be derived

$$\begin{aligned}
\boldsymbol{\delta\xi}^{\mathbf{n}}(k+1|k) =& \boldsymbol{A}(k)\boldsymbol{\delta\xi}^{\mathbf{init,n}} + \boldsymbol{B}(k)\boldsymbol{\delta u}^{\mathbf{n}}(k|k),\\
\boldsymbol{\delta\xi}^{\mathbf{n}}(k+2|k) =& \boldsymbol{A}(k+1)\boldsymbol{\delta\xi}^{\mathbf{n}}(k+1|k) + \boldsymbol{B}(k+1)\boldsymbol{\delta u}^{\mathbf{n}}(k+1|k) =\\
=& \boldsymbol{A}(k+1)\left[\boldsymbol{A}(k)\boldsymbol{\delta\xi}^{\mathbf{init,n}} + \boldsymbol{B}(k)\boldsymbol{\delta u}^{\mathbf{n}}(k|k)\right] +\\
&+ \boldsymbol{B}(k+1)\boldsymbol{\delta u}^{\mathbf{n}}(k+1|k) =\\
=& \boldsymbol{A}(k+1)\boldsymbol{A}(k)\boldsymbol{\delta\xi}^{\mathbf{init,n}} + \boldsymbol{A}(k+1)\boldsymbol{B}(k)\boldsymbol{\delta u}^{\mathbf{n}}(k|k)+\\
&+\boldsymbol{B}(k+1)\boldsymbol{\delta u}^{\mathbf{n}}(k+1|k),\\
&\vdots\\
\boldsymbol{\delta\xi}^{\mathbf{n}}(k+H_p|k) =& \prod_{p=H_p-1}^{0} \boldsymbol{A}(k+p)\boldsymbol{\delta\xi}^{\mathbf{init,n}}+\\
&+ \sum_{i=0}^{H_p-1} \underbrace{\left[\prod_{j=H_p-1}^{i+1} \boldsymbol{A}(k+j)\right]}_{\boldsymbol{I}_{n_\xi}\text{,if } j<i+1} \boldsymbol{B}(k+i)\boldsymbol{\delta u}^{\mathbf{n}}(k+i|k),
\end{aligned}$$

where $H_p \leq H_{\bar{u}}$, $\boldsymbol{I}_{n_\xi} \in \mathbb{R}^{n_\xi \times n_\xi}$ is the identity matrix, and $\boldsymbol{\delta\xi}^{\mathbf{init,n}} = \boldsymbol{\xi}(k) - \tilde{\boldsymbol{\xi}}^{\mathbf{n}}(k)$. The above prediction scheme can be written in a compact way as

$$\begin{aligned} \boldsymbol{\delta\tilde{\xi}}_{k+1:k+H_p|k} &= \tilde{\boldsymbol{A}}(k)\boldsymbol{\delta\xi}^{\mathbf{init,n}} + \tilde{\boldsymbol{B}}(k)\boldsymbol{\delta\tilde{u}}_{k:k+H_u|k}, \\ \boldsymbol{\delta\tilde{y}}_{k:k+H_p|k} &= \tilde{\boldsymbol{C}}(k)\boldsymbol{\delta\xi}^{\mathbf{init,n}} + \tilde{\boldsymbol{D}}(k)\boldsymbol{\delta\tilde{u}}_{k:k+H_u|k}, \end{aligned} \tag{3.13}$$

where the matrices $\tilde{\boldsymbol{A}}(k)$, $\tilde{\boldsymbol{B}}(k)$, $\tilde{\boldsymbol{C}}(k)$, and $\tilde{\boldsymbol{D}}(k)$ are the analogous time variant versions of $\tilde{\boldsymbol{A}}$, $\tilde{\boldsymbol{B}}$, $\tilde{\boldsymbol{C}}$, and $\tilde{\boldsymbol{D}}$ respectively. By making explicit the formulation of (3.13) into (3.4), neglecting the terms not depending on $\boldsymbol{\delta\tilde{u}}^{\mathbf{n}}_{k:k+H_p|k}$, and remembering that $\boldsymbol{y}(k) = \boldsymbol{\delta y}^{\mathbf{n}}(k) + \tilde{\boldsymbol{y}}^{\mathbf{n}}(k)$ and $\boldsymbol{u}(k) = \boldsymbol{\delta u}^{\mathbf{n}}(k) + \tilde{\boldsymbol{u}}^{\mathbf{n}}(k)$, the following QP form is derived

$$\min_{\boldsymbol{\delta\tilde{u}}^{\mathbf{n}}_{k:k+H_p|k}} \frac{1}{2}\boldsymbol{\delta\tilde{u}}^{\mathbf{n}}_{k:k+H_p|k}{}^{\top}\boldsymbol{H}(k)\boldsymbol{\delta\tilde{u}}^{\mathbf{n}}_{k:k+H_p|k} + \boldsymbol{f}^{\top}(k)\boldsymbol{\delta\tilde{u}}^{\mathbf{n}}_{k:k+H_p|k} \tag{3.14}$$

subject to

$$\underline{\boldsymbol{A}}(k)\boldsymbol{\delta\tilde{u}}^{\mathbf{n}}_{k:k+H_p|k} \leq \underline{\boldsymbol{b}}(k),$$

where

$$\begin{aligned} \boldsymbol{H}(k) =& (\boldsymbol{R} + \tilde{\boldsymbol{D}}(k)^{\top}\boldsymbol{Q}\tilde{\boldsymbol{D}}(k)), \\ \boldsymbol{f}(k)^{\top} =& (\boldsymbol{\delta\xi}^{\mathbf{init}\top}\tilde{\boldsymbol{C}}(k)^{\top}\boldsymbol{Q}\tilde{\boldsymbol{D}}(k) + \tilde{\boldsymbol{y}}^{\mathbf{n}\top}\boldsymbol{Q}\tilde{\boldsymbol{D}}(k) - \tilde{\boldsymbol{y}}_{\mathrm{ref}}{}^{\top}\boldsymbol{Q}\tilde{\boldsymbol{D}}(k) + \\ & + \tilde{\boldsymbol{u}}^{\mathbf{n}\top}\boldsymbol{R} - \tilde{\boldsymbol{u}}_{\mathrm{ref}}{}^{\top}\boldsymbol{R}), \end{aligned}$$

while the time varying matrix $\underline{\boldsymbol{A}}(k)$ and vector $\underline{\boldsymbol{b}}(k)$ are used to enforce inputs, states, and outputs constraints. Note that, differently to the LTI case, the matrix and the array used in the QP formulation (i.e., $\boldsymbol{H}(k)$ and $\boldsymbol{f}(k)$) are time dependent as well. This implies that such quantities need to be evaluated at each time step, unlike the equivalent LTI formulation that requires to compute such quantities only once.

Nominal trajectories optimization for LTV systems

As discussed in Section 3.2.1, LTV representations are used to approximate nonlinear dynamics around nominal trajectories of inputs and states. Such trajectories can be obtained in several ways, based on: (*i*) prior physical or heuristic knowledge of the plant [Schubert et al., 1994], (*ii*) some empirical rules, or (*iii*) optimization approaches that exploit a nonlinear mathematical model of the plant [Srinivasan et al., 2003, Breakwell, 1959]. When optimization-based approaches are applied over nonlinear dynamics, computationally expensive procedures are employed to obtain sub-optimal (or, if possible, optimal) solutions of the given problem. In the following, this latter approach is discussed. Starting from an initial time instant k_0, consider an optimization of the form

$$\min_{\tilde{\boldsymbol{u}}^{\mathbf{n}}_{k_0:k_0+H_{\bar{u}}}} J(\tilde{\boldsymbol{u}}^{\mathbf{n}}_{k_0:k_0+H_{\bar{u}}}, \boldsymbol{\xi}_{\mathbf{init}}) \tag{3.15}$$

subject to

$$\begin{aligned}
\boldsymbol{\xi}^{\mathbf{n}}(j+1) &= \boldsymbol{f}(\boldsymbol{\xi}^{\mathbf{n}}(j), \boldsymbol{u}^{\mathbf{n}}(j), j) \\
\boldsymbol{y}^{\mathbf{n}}(j) &= \boldsymbol{g}(\boldsymbol{\xi}^{\mathbf{n}}(j), \boldsymbol{u}^{\mathbf{n}}(j), j) \\
\boldsymbol{\xi}^{\mathbf{n}}(k_0) &= \boldsymbol{\xi}_{\mathbf{init}} \\
\boldsymbol{u}_{\min} &\leq \boldsymbol{u}^{\mathbf{n}}(j) \leq \boldsymbol{u}_{\max} \\
\boldsymbol{\xi}_{\min} &\leq \boldsymbol{\xi}^{\mathbf{n}}(j) \leq \boldsymbol{\xi}_{\max} \\
\boldsymbol{y}_{\min} &\leq \boldsymbol{y}^{\mathbf{n}}(j) \leq \boldsymbol{y}_{\max}
\end{aligned}$$

where $j \in [k_0; k_0 + H_{\bar{u}}]$ and $\tilde{\boldsymbol{u}}^{\mathbf{n}}_{k_0:k_0+H_{\bar{u}}} \in \mathbb{R}^{n_u H_{\bar{u}}}$, $\tilde{\boldsymbol{\xi}}^{\mathbf{n}}_{k_0:k_0+H_{\bar{u}}} \in \mathbb{R}^{n_\xi H_{\bar{u}}}$, and $\tilde{\boldsymbol{y}}^{\mathbf{n}}_{k_0:k_0+H_{\bar{u}}} \in \mathbb{R}^{n_y H_{\bar{u}}}$ are defined similarly to (3.2). Due to the nonlinear/complex dynamics used to formulate (3.15), this optimization is usually solved offline, and the results are used for developing the LTV approximation. To obtain a nominal trajectory useful for control purposes, the optimiza-

tion is usually solved over a horizon able to guarantee the attainment of a neighborhood of the desired setpoint.

LTV closed-loop MPC scheme

In order to carry out the closed-loop control action, the LTV-based MPC strategy needs a suitable algorithm.
Starting from $k = k_0$, the optimal solutions $\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k_0:k_0+H_{\bar{u}}}$ and $\tilde{\boldsymbol{\xi}}^{\mathbf{n},*}_{k_0:k_0+H_{\bar{u}}}$ provided by (3.15) are applied within a MPC context as follows:

1. Obtain the LTV approximation of the nonlinear dynamics using the sub-sequences $\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k:k+H_u}$ and $\tilde{\boldsymbol{\xi}}^{\mathbf{n},*}_{k:k+H_p}$ as discussed in Section 3.2.1

2. At time k, solve the optimization (3.14) to obtain $\boldsymbol{\delta}\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k:k+H_u|k}$ and update the subsequence $\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k:k+H_u} \leftarrow \tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k:k+H_u} + \boldsymbol{\delta}\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k:k+H_u|k}$

3. According to the RH approach (3.5), apply $\boldsymbol{u}_{\mathrm{RH}}(k) = \boldsymbol{u}^{\mathbf{n},*}(k|k)$ to the real plant

4. If $k+1+H_u > k_0+H_{\bar{u}}$ then extend the nominal input trajectory by repeating its last element, i.e.,

$$\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k+1:k+1+H_u} = \left[\boldsymbol{u}^{\mathbf{n},*}(k+1), \boldsymbol{u}^{\mathbf{n},*}(k+2), \cdots, \boldsymbol{u}^{\mathbf{n},*}(k+H_{\bar{u}}),\ \boldsymbol{u}^{\mathbf{n},*}(k+H_{\bar{u}})\right].$$

5. Update the initial states $\boldsymbol{\delta\xi}^{\mathbf{init,n}}$ using the plant measurements

6. Compute $\tilde{\boldsymbol{\xi}}^{\mathbf{n},*}_{k+1:k+1+H_p}$ by feeding $\tilde{\boldsymbol{u}}^{\mathbf{n},*}_{k+1:k+1+H_u}$ into the nonlinear dynamics used in (3.15)

7. Set $k \leftarrow k+1$ and go back to Step 1

3.3.3 MPC formulation for ARX and PWARX systems

Predictive control strategies can be easily developed also when the plant dynamics are represented by means of ARX or PWARX models. However, when dealing with MPC algorithms, it is more convenient to represent the dynamics of the prediction model in terms of a state-space formulation. According to the realization theory [Matsuo and Hasegawa, 2003], several canonical forms (e.g., observability or controllability) can be adopted to reformulate an input-output model in terms of a state-space one. All of these considerations also hold for the PWARX models, where the presence of switching dynamics have to be carefully treated. PWARX models can be reformulated as state-space ones using two main alternative representations [Bemporad and Morari, 1999]: (*i*) the Mixed Logical Dynamical (MLD) scheme, and (*ii*) the piecewise linear time invariant scheme. In this Thesis the MLD formulation is adopted, where a set of logic rules, physical laws, and constraints are used to represent the dynamical behavior of a switching system. The one-step ahead prediction scheme of a MLD system is

$$\begin{cases} \boldsymbol{\xi}(k+1) = \boldsymbol{A}\boldsymbol{\xi}(k) + \boldsymbol{B_1}\boldsymbol{u}(k) + \boldsymbol{B_2}\boldsymbol{\delta}(k) + \boldsymbol{B_3}\boldsymbol{z}(k) \\ \quad \boldsymbol{y}(k) = \boldsymbol{C}\boldsymbol{\xi}(k) + \boldsymbol{D_1}\boldsymbol{u}(k) + \boldsymbol{D_2}\boldsymbol{\delta}(k) + \boldsymbol{D_3}\boldsymbol{z}(k) \\ \boldsymbol{E_2}\boldsymbol{\delta}(k) + \boldsymbol{E_3}\boldsymbol{z}(k) \leq \boldsymbol{E_1}\boldsymbol{u}(k) + \boldsymbol{E_4}\boldsymbol{\xi}(k) + \boldsymbol{E_5} \end{cases} \tag{3.16}$$

where the array $\boldsymbol{\delta}(k) \in \{0,1\}^{n_\delta}$ represents the set of binary variables (logic rules) used to switch from one submodel to another, $\boldsymbol{z}(k) \in \mathbb{R}^{n_z}$ are *auxiliary variables* [Williams, 2013], and $\boldsymbol{A}$, $\boldsymbol{B_1}$, $\boldsymbol{B_2}$, $\boldsymbol{B_3}$, $\boldsymbol{C}$, $\boldsymbol{D_1}$, $\boldsymbol{D_2}$, $\boldsymbol{D_3}$, $\boldsymbol{E_1}$, $\boldsymbol{E_2}$, $\boldsymbol{E_3}$, $\boldsymbol{E_4}$, and $\boldsymbol{E_5}$ are matrices of suitable dimensions. The model (3.16) should be constructed to be well posed. The structure of (3.16) allows to reformulate (3.3) into a QP form. Let us introduce the MLD prediction

scheme over the future H_p steps as

$$\begin{cases} \tilde{\boldsymbol{\xi}}_{k+1:k+H_p|k} = \tilde{\boldsymbol{A}}\boldsymbol{\xi}_{\mathbf{init}} + \tilde{\boldsymbol{B}}_{\mathbf{1}}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \tilde{\boldsymbol{B}}_{\mathbf{2}}\tilde{\boldsymbol{\delta}}_{k:k+H_p|k} + \tilde{\boldsymbol{B}}_{\mathbf{3}}\tilde{\boldsymbol{z}}_{k:k+H_p|k} \\ \tilde{\boldsymbol{y}}_{k:k+H_p|k} = \tilde{\boldsymbol{C}}\boldsymbol{\xi}_{\mathbf{init}} + \tilde{\boldsymbol{D}}_{\mathbf{1}}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \tilde{\boldsymbol{D}}_{\mathbf{2}}\tilde{\boldsymbol{\delta}}_{k:k+H_p|k} + \tilde{\boldsymbol{D}}_{\mathbf{3}}\tilde{\boldsymbol{z}}_{k:k+H_p|k} \\ \tilde{\boldsymbol{E}}_{\mathbf{2}}\tilde{\boldsymbol{\delta}}_{k:k+H_p|k} + \tilde{\boldsymbol{E}}_{\mathbf{3}}\tilde{\boldsymbol{z}}_{k:k+H_p|k} \leq \tilde{\boldsymbol{E}}_{\mathbf{1}}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \tilde{\boldsymbol{E}}_{\mathbf{4}}\boldsymbol{\xi}_{\mathbf{init}} + \tilde{\boldsymbol{E}}_{\mathbf{5}} \end{cases} \tag{3.17}$$

where $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{C}}$ are structured as for the LTI case, while the matrices $\tilde{\boldsymbol{B}}_{\mathbf{1}} \in \mathbb{R}^{n_\xi H_p \times n_u H_u}$, $\tilde{\boldsymbol{B}}_{\mathbf{2}} \in \mathbb{R}^{n_\xi H_p \times n_u H_u}$ and $\tilde{\boldsymbol{B}}_{\mathbf{3}} \in \mathbb{R}^{n_\xi H_p \times n_z H_u}$ are defined as follows

$$\tilde{\boldsymbol{B}}_{\boldsymbol{i}} = \begin{bmatrix} \boldsymbol{B_i} & 0 & \cdots & 0 \\ \boldsymbol{A}\boldsymbol{B_i} & \boldsymbol{B_i} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{A}^{H_p-1}\boldsymbol{B_i} & \boldsymbol{A}^{H_p-2}\boldsymbol{B_i} & \cdots & \boldsymbol{B_i} \end{bmatrix}, i \in \{1,2,3\}.$$

Similarly, the matrices $\tilde{\boldsymbol{D}}_{\mathbf{1}} \in \mathbb{R}^{n_y H_p \times n_u H_u}$, $\tilde{\boldsymbol{D}}_{\mathbf{2}} \in \mathbb{R}^{n_y H_p \times n_u H_u}$ and $\tilde{\boldsymbol{D}}_{\mathbf{3}} \in \mathbb{R}^{n_y H_p \times n_z H_u}$ are defined as

$$\tilde{\boldsymbol{D}}_{\boldsymbol{i}} = \begin{bmatrix} \boldsymbol{D_i} & 0 & \cdots & 0 \\ \boldsymbol{C}\boldsymbol{B_i} & \boldsymbol{D_i} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{C}\boldsymbol{A}^{H_p-1}\boldsymbol{B_i} & \boldsymbol{C}\boldsymbol{A}^{H_p-2}\boldsymbol{B_i} & \cdots & \boldsymbol{D_i} \end{bmatrix}, i \in \{1,2,3\}.$$

Finally, $\tilde{\boldsymbol{E}}_{\mathbf{1}}$, $\tilde{\boldsymbol{E}}_{\mathbf{2}}$, and $\tilde{\boldsymbol{E}}_{\mathbf{3}}$ are matrices of suitable dimensions structured as

$$\tilde{\boldsymbol{E}}_{\boldsymbol{i}} = \begin{bmatrix} \boldsymbol{E_i} & 0 & 0 & \cdots & 0 \\ \boldsymbol{E_4}\boldsymbol{B_i} & \boldsymbol{E_i} & 0 & \cdots & 0 \\ \boldsymbol{E_4}\boldsymbol{A}\boldsymbol{B_i} & \boldsymbol{E_4}\boldsymbol{B_i} & \boldsymbol{E_i} & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{E_4}\boldsymbol{A}^{H_p-1}\boldsymbol{B_i} & \boldsymbol{E_4}\boldsymbol{A}^{H_p-2}\boldsymbol{B_i} & \cdots & \cdots & \boldsymbol{E_i} \end{bmatrix}, i \in \{1,2,3\}$$

whereas $\tilde{\boldsymbol{E}}_{\boldsymbol{4}}$, and $\tilde{\boldsymbol{E}}_{\boldsymbol{5}}$ are

$$\tilde{\boldsymbol{E}}_{\boldsymbol{4}} = \begin{bmatrix} \boldsymbol{E_4} \\ \boldsymbol{E_4 A} \\ \boldsymbol{E_4 A}^2 \\ \vdots \\ \boldsymbol{E_4 A}^{H_p} \end{bmatrix}, \qquad \tilde{\boldsymbol{E}}_{\boldsymbol{5}} = \begin{bmatrix} \boldsymbol{E_5} \\ \boldsymbol{E_5} \\ \boldsymbol{E_5} \\ \vdots \\ \boldsymbol{E_5} \end{bmatrix}.$$

As introduced for the LTI and LTV cases, it is now possible to explicitly express (3.17) into (3.4) to obtain a QP formulation. Despite this, in the MLD case, the optimization variables are composed not only of the input moves $\tilde{\boldsymbol{u}}_{k:k+H_u|k}$, but also by the binary variables $\tilde{\boldsymbol{\delta}}_{k:k+H_p|k}$, and the auxiliary variables $\tilde{\boldsymbol{z}}_{k:k+H_p|k}$. For this reason, in order to avoid bilinearities which would determine a non-convexity of the resulting optimization problem, it is convenient to introduce

$$\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} = [\tilde{\boldsymbol{u}}_{k:k+H_u|k}{}^{\top}, \tilde{\boldsymbol{\delta}}_{k:k+H_p|k}{}^{\top}, \tilde{\boldsymbol{z}}_{k:k+H_p|k}{}^{\top}]^{\top} \in \mathbb{R}^{n_u H_u + H_p(n_\delta + n_z)}.$$

According to this change of variables, the prediction scheme in (3.17) can be rewritten as

$$\begin{cases} \tilde{\boldsymbol{\xi}}_{k+1:k+H_p|k} = \tilde{\boldsymbol{A}}\boldsymbol{\xi}_{\mathbf{init}} + \tilde{\boldsymbol{B}}_{\mathbf{tot}}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} \\ \tilde{\boldsymbol{y}}_{k:k+H_p|k} = \tilde{\boldsymbol{C}}\boldsymbol{\xi}_{\mathbf{init}} + \tilde{\boldsymbol{D}}_{\mathbf{tot}}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} \\ \tilde{\boldsymbol{E}}_{\mathbf{tot}}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} - \tilde{\boldsymbol{E}}_{\boldsymbol{4}}\boldsymbol{\xi}_{\mathbf{init}} - \tilde{\boldsymbol{E}}_{\boldsymbol{5}} \leq 0 \end{cases} \tag{3.18}$$

With this formulation it is now possible to explicit (3.18) into (3.4) to obtain a QP form. By neglecting the terms that do not depend on $\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k}$, the

following Mixed Integer QP (MIQP) problem is derived

$$\min_{\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k}} \frac{1}{2}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k}{}^{\top}\boldsymbol{H}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} + \boldsymbol{f}^{\top}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} \tag{3.19}$$

subject to

$$\underline{\boldsymbol{A}}\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k} \leq \underline{\boldsymbol{b}}$$
$$\boldsymbol{\delta}(j|k) \in \{0,1\},$$

where $j \in [k; k+H_p]$, $\tilde{\boldsymbol{B}}_{\mathbf{tot}} = \begin{bmatrix}\tilde{\boldsymbol{B}}_{\mathbf{1}} & \tilde{\boldsymbol{B}}_{\mathbf{2}} & \tilde{\boldsymbol{B}}_{\mathbf{3}}\end{bmatrix}$, $\tilde{\boldsymbol{D}}_{\mathbf{tot}} = \begin{bmatrix}\tilde{\boldsymbol{D}}_{\mathbf{1}} & \tilde{\boldsymbol{D}}_{\mathbf{2}} & \tilde{\boldsymbol{D}}_{\mathbf{3}}\end{bmatrix}$, $\tilde{\boldsymbol{E}}_{\mathbf{tot}} = \begin{bmatrix}-\tilde{\boldsymbol{E}}_{\mathbf{1}} & \tilde{\boldsymbol{E}}_{\mathbf{2}} & \tilde{\boldsymbol{E}}_{\mathbf{3}}\end{bmatrix}$, and $\tilde{\boldsymbol{\Upsilon}}_{\text{ref}}$ represents a reference sequence for $\tilde{\boldsymbol{\Upsilon}}_{k:k+H_p|k}$. The mixed-integer formulation, due to the presence of the binary variables $\tilde{\boldsymbol{\delta}}_{k:k+H_p|k}$, generally makes the resolution of (3.19) more complex than in the LTI/LTV cases. From the optimal solution of the above MIQP ($\tilde{\boldsymbol{\Upsilon}}^*_{k:k+H_p|k}$), it is possible to recover the optimal input sequence $\tilde{\boldsymbol{u}}^*_{k:k+H_u|k}$ used to provide the control action to the plant, i.e.

$$\boldsymbol{u}_{\text{RH}}(k) = \mathbf{u}^*(k|k).$$

In order to avoid further repetitions, the formulation of a MPC algorithm based on an ARX model is not addressed below. In fact, the same procedure illustrated for the LTI case can be followed, provided that the ARX model (3.9) is first converted to a state-space form using some realization technique [Matsuo and Hasegawa, 2003].

Remark: The QP formulations presented for the different approximations of (3.1) are suitable only for stable or asymptotically stable systems. For example, in case of LTI unstable dynamics, the matrices $\tilde{\boldsymbol{A}}$, $\tilde{\boldsymbol{B}}$, $\tilde{\boldsymbol{C}}$, and $\tilde{\boldsymbol{D}}$ (or the equivalent ones for the other representations) could diverge or give rise to numerical problems. In such cases, it is convenient to include the model dynamics as equality constraints of the MPC problem.

3.3.4 MPC formulation for FSR models

When FSR models are considered, the formulation introduced in Section 3.2.2 can be used recursively in order to obtain a prediction scheme suitable for the definition of a QP problem. Recall from (3.8) that the prediction of the output of a SISO plant at time step k is obtained as a linear combination of consecutive input differences and step response coefficients. This scheme can be generalized in order to provide a model able to predict the plant behavior over the future H_p steps. For simplicity, in the following the SISO case will be treated in detail, while a summarized version of the MIMO one will be given.

Define the *free response* $r(k+j|k)$ at a time step k as the system output $y(k+j)$ under the assumption that changes in the control variable are zero from k into the future ($\Delta u(k+j) = 0,\ \forall j \geq 0$):

$$r(k+j|k) := y(k+j) \text{ in case of } \Delta u(k+j) = 0,\ \forall j \geq 0.$$

The predicted output values over the future H_p steps can be written as

$$\begin{aligned}
y(k+1|k) &= S^1_{1,1}\Delta u(k|k) + r(k+1|k),\\
y(k+2|k) &= S^2_{1,1}\Delta u(k|k) + S^1_{1,1}\Delta u(k+1|k) + r(k+2|k),\\
y(k+3|k) &= S^3_{1,1}\Delta u(k|k) + S^2_{1,1}\Delta u(k+1|k) + S^1_{1,1}\Delta u(k+2|k) + r(k+3|k),\\
&\vdots\\
y(k+H_p|k) &= \sum_{j=1}^{H_p} S^j_{1,1}\Delta u(k+H_p-j|k) + r(k+H_p|k),
\end{aligned}$$

that can be reformulated in a matrix form as follows

$$\tilde{\boldsymbol{y}}_{k:k+H_p|k} = \boldsymbol{\Psi}\tilde{\boldsymbol{r}}_{k:k+N|k} + \boldsymbol{S}^1_1\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}. \tag{3.20}$$

The binary matrix $\boldsymbol{\Psi}$ is used as a shifting operator for the free response sequence, and its defined as

$$\boldsymbol{\Psi} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{bmatrix} \in \mathbb{R}^{H_p \times N},$$

while the *dynamic matrix* $\boldsymbol{S}_1^1$ is composed of the step response coefficients S^g

$$\boldsymbol{S}_1^1 = \begin{bmatrix} S_{1,1}^1 & 0 & \cdots & 0 \\ S_{1,1}^2 & S_{1,1}^1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & 0 \\ S_{1,1}^{H_u} & \vdots & \vdots & S_{1,1}^1 \\ \vdots & \vdots & \vdots & \vdots \\ S_{1,1}^{H_p} & S_{1,1}^{H_p-1} & \cdots & S_{1,1}^{H_p-H_u+1} \end{bmatrix} \in \mathbb{R}^{H_p \times H_u}. \tag{3.21}$$

To take into account the discrepancy between the step response model and the original plant, a correction term based on the actual measurement of the output is required. Define $\tilde{\boldsymbol{w}}_{k:k+H_p|k} \in \mathbb{R}^{H_p}$ as the sequence of the future uncertainties acting on the system. Given that the terms in $\tilde{\boldsymbol{w}}_{k:k+H_p|k}$ cannot be measured, a good estimate is

$$w(k|k) = y^{\text{meas}}(k) - r(k|k)$$

where $y^{\text{meas}}(k)$ is the measured output at time instant k, and it is assumed that $w(k|k) = w(k+1|k) = \cdots = w(k+H_p|k)$. Therefore, the prediction equation (3.20) becomes

$$\tilde{\boldsymbol{y}}_{k:k+H_p|k} = \boldsymbol{\Psi}\tilde{\boldsymbol{r}}_{k:k+N|k} + \boldsymbol{S}_1^1\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \tilde{\boldsymbol{w}}_{k:k+H_p|k}, \tag{3.22}$$

which contains the contribution of past inputs ($\boldsymbol{\Psi}\tilde{\boldsymbol{r}}_{k:k+N|k}$), the future control actions ($\boldsymbol{S}_1^1\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}$), and a correction term ($\tilde{\boldsymbol{w}}_{k:k+H_p|k}$). The prediction scheme (3.22) can be used to obtain a QP form of the problem (3.3). Because of the FSR formulation, the control variable is expressed by means of consecutive input differences $\Delta u(k)$, rather than input moves $u(k)$. For this reason, the cost function (3.4) is rewritten in the following form

$$J(\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}, \tilde{\boldsymbol{r}}_{k:k+N|k}) = ||\tilde{\boldsymbol{y}}_{k:k+H_p|k} - \tilde{\boldsymbol{y}}_{\text{ref}}||^2_{\boldsymbol{Q}} + ||\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}||^2_{\boldsymbol{R}}. \tag{3.23}$$

By making explicit the definition of (3.22) into (3.23), the following optimization problem is obtained

$$\min_{\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}} \tilde{\boldsymbol{\varsigma}}_{k:k+H_p|k}{}^{\top}\boldsymbol{Q}\tilde{\boldsymbol{\varsigma}}_{k:k+H_p|k} + \boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k}{}^{\top}\boldsymbol{R}\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} \tag{3.24}$$

subject to

$$\begin{aligned} u_{min} \leq\ & u(k+j|k) \leq u_{max} \\ y_{min} \leq\ & y(k+j|k) \leq y_{max} \\ \Delta u_{min} \leq\ & \Delta u(k+j|k) \leq \Delta u_{max} \\ \Delta y_{min} \leq\ & \Delta y(k+j|k) \leq \Delta y_{max} \end{aligned}$$

where $j \in [1; H_p]$, while the sequence

$$\tilde{\boldsymbol{\varsigma}}_{k:k+H_p|k} = \boldsymbol{\Psi}\tilde{\boldsymbol{r}}_{k:k+N|k} + \boldsymbol{S}_1^1\boldsymbol{\Delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \tilde{\boldsymbol{w}}_{k:k+H_p|k} - \tilde{\boldsymbol{y}}_{\text{ref}}$$

represents the differences between the predicted outputs $\tilde{\boldsymbol{y}}_{k:k+H_p|k}$ and the reference trajectory $\tilde{\boldsymbol{y}}_{\text{ref}}$. From the formulation of (3.24) it is possible to

define the following QP formulation

$$\min_{\boldsymbol{\Delta\tilde{u}}_{k:k+H_u|k}} \frac{1}{2}\boldsymbol{\Delta\tilde{u}}_{k:k+H_u|k}{}^{\top}\boldsymbol{H}\boldsymbol{\Delta\tilde{u}}_{k:k+H_u|k} + \boldsymbol{f}^{\top}\boldsymbol{\Delta\tilde{u}}_{k:k+H_u|k} \tag{3.25}$$

subject to

$$\underline{\boldsymbol{A}}\boldsymbol{\Delta\tilde{u}}_{k:k+H_u|k} \leq \underline{\boldsymbol{b}},$$

where

$$\begin{aligned}\boldsymbol{H} &= (\boldsymbol{R} + \boldsymbol{S}_1^1{}^{\top}\boldsymbol{Q}\boldsymbol{S}_1^1),\\ \boldsymbol{f}^{\top} &= (\tilde{\boldsymbol{r}}_{k:k+N|k}{}^{\top}\boldsymbol{\Psi}^{\top}\boldsymbol{Q}\boldsymbol{S}_1^1 + \tilde{\boldsymbol{w}}_{k:k+H_p|k}{}^{\top}\boldsymbol{Q}\boldsymbol{S}_1^1 - \tilde{\boldsymbol{y}}_{\mathrm{ref}}{}^{\top}\boldsymbol{Q}\boldsymbol{S}_1^1),\end{aligned}$$

The solution of (3.25) provides the optimal sequence $\boldsymbol{\Delta\tilde{u}}^*_{k:k+H_u|k}$. According to the RH approach (3.5), the plant has to be subjected only to the first element of such sequence. In particular, given that $\boldsymbol{\Delta\tilde{u}}^*_{k:k+H_u|k}$ contains a sequence of input variations, the control action applied to the plant will be

$$\boldsymbol{u}_{\mathrm{RH}}(k) = u(k-1) + \Delta u^*(k|k).$$

After the resolution of (3.25), the free response array of the next time step (i.e., $k+1$) is updated by

$$\tilde{\boldsymbol{r}}_{k+1:k+1+N|k} := \boldsymbol{\Pi}\tilde{\boldsymbol{r}}_{k:k+N|k} + \boldsymbol{S}_{1,1}\Delta u(k|k) \tag{3.26}$$

where $\boldsymbol{S}_{1,1} \in \mathbb{R}^N$ is the array containing all the step response coefficients, and $\boldsymbol{\Pi} \in \mathbb{R}^{N\times N}$ is the binary matrix

$$\boldsymbol{\Pi} := \begin{bmatrix} 0 & 1 & 0 & \cdots & 0\\ 0 & 0 & 1 & \vdots & \vdots\\ \vdots & \vdots & \vdots & \ddots & 0\\ 0 & 0 & \cdots & 0 & 1\\ 0 & 0 & \cdots & 0 & 1 \end{bmatrix} \in \mathbb{R}^{N\times N}. \tag{3.27}$$

The above formulation of $\mathbf{\Pi}$ is suitable for asymptotically stable systems. When the output of the model shows an integrating behavior, the matrix $\mathbf{\Pi}$ is redefined as [Morari et al., 2002]

$$\mathbf{\Pi} := \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ 0 & 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & -1 & 2 \end{bmatrix} \in \mathbb{R}^{N \times N}. \tag{3.28}$$

In this case, the last element of $\tilde{\boldsymbol{r}}_{k+1:k+1+N|k}$ is

$$r(k+N-1|k+1) := r(k+N-1|k)+(r(k+N-1|k)-r(k+N-2|k))+S^N \Delta u(k|k)$$

which considers the integrating trend of the output. Due to its recursive formulation, the free response sequence needs to be initialized. To this aim, the quantity $\tilde{\boldsymbol{r}}_{k_0:k_0+N|k}$ is used to represent the initial value of the such sequence. Usually the plant is kept to steady state, and the output values are used to initialize the free response terms.

The prediction scheme in (3.22), suitable for SISO plants, can be straightforwardly extended to MIMO systems as follows:

$$\begin{bmatrix} \tilde{\boldsymbol{y}}^1_{k:k+H_p|k} \\ \tilde{\boldsymbol{y}}^2_{k:k+H_p|k} \\ \vdots \\ \tilde{\boldsymbol{y}}^{n_y}_{k:k+H_p|k} \end{bmatrix} = \begin{bmatrix} \mathbf{\Psi}\tilde{\boldsymbol{r}}^1_{k:k+N|k} + \tilde{\boldsymbol{w}}^1_{k:k+H_p|k} \\ \mathbf{\Psi}\tilde{\boldsymbol{r}}^2_{k:k+N|k} + \tilde{\boldsymbol{w}}^2_{k:k+H_p|k} \\ \vdots \\ \mathbf{\Psi}\tilde{\boldsymbol{r}}^{n_y}_{k:k+N|k} + \tilde{\boldsymbol{w}}^{n_y}_{k:k+H_p|k} \end{bmatrix} + \begin{bmatrix} \boldsymbol{S}^1_1 & \cdots & \boldsymbol{S}^1_{n_u} \\ \boldsymbol{S}^2_1 & \cdots & \boldsymbol{S}^2_{n_u} \\ \vdots & \ddots & \vdots \\ \boldsymbol{S}^{n_y}_1 & \cdots & \boldsymbol{S}^{n_y}_{n_u} \end{bmatrix} \begin{bmatrix} \mathbf{\Delta}\tilde{\boldsymbol{u}}^1_{k:k+H_u|k} \\ \mathbf{\Delta}\tilde{\boldsymbol{u}}^2_{k:k+H_u|k} \\ \vdots \\ \mathbf{\Delta}\tilde{\boldsymbol{u}}^{n_u}_{k:k+H_u|k} \end{bmatrix}. \tag{3.29}$$

Notice that $\tilde{\boldsymbol{r}}^b_{k:k+N|k}$ and $\tilde{\boldsymbol{w}}^b_{k:k+H_p|k}$ are the free response and disturbance estimation sequences of the bth output, respectively. The elements $\boldsymbol{S}^b_i$ are the

FSR coefficients matrices relating the ith input to the bth output structured as shown in (3.21), whereas $\boldsymbol{\Delta\tilde{u}}^{i}_{k:k+H_u|k}$ represents the input variations sequence related to the ith input.

The formulation of a predictive control algorithm as in (3.24), refers to the family of Quadratic Dynamic Matrix Control (QDMC) paradigms [Garcia and Morshedi, 1986].

3.3.5 MPC formulation with soft output constraints

When dealing with nonlinear plants, the use of approximated models for control purposes brings inevitable mismatch which may lead to constraint violations. A common approach adopted for guaranteeing recursive feasibility, is to *soften* the state and output constraints by suitably modifying the control problem. The softening of the outputs constraints can be carried out by adding a set of optimization variables to the general formulation (3.3), which leads to the optimization

$$\min_{\tilde{\boldsymbol{u}}_{k:k+H_u|k},\tilde{\boldsymbol{v}}_{k:k+H_p|k}} \hat{J}(\tilde{\boldsymbol{u}}_{k:k+H_u|k},\tilde{\boldsymbol{v}}_{k:k+H_p|k},\boldsymbol{\xi}_{\mathbf{init}}) \tag{3.30}$$

subject to

$$\begin{aligned}
\boldsymbol{\xi}(j+1|k) &= \boldsymbol{f}(\boldsymbol{\xi}(j|k),\boldsymbol{u}(j|k),j)\\
\boldsymbol{y}(j|k) &= \boldsymbol{g}(\boldsymbol{\xi}(j|k),\boldsymbol{u}(j|k),j)\\
\boldsymbol{\xi}(k|k) &= \boldsymbol{\xi}_{\mathbf{init}}\\
\boldsymbol{u}_{\min} &\le \boldsymbol{u}(j|k) \le \boldsymbol{u}_{\max}\\
\boldsymbol{\xi}_{\min} &\le \boldsymbol{\xi}(j|k) \le \boldsymbol{\xi}_{\max}\\
\boldsymbol{y}_{\min} - \boldsymbol{v}(j|k) &\le \boldsymbol{y}(j|k) \le \boldsymbol{y}_{\max} + \boldsymbol{v}(j|k)\\
\boldsymbol{v}(j|k) &\ge 0
\end{aligned}$$

where $\tilde{\boldsymbol{v}}_{k:k+H_p|k} \in \mathbb{R}^{n_y H_p}$ is defined similarly to (3.2), the cost function $\hat{J}$ is defined as

$$\hat{J}(\tilde{\boldsymbol{u}}_{k:k+H_u|k}, \tilde{\boldsymbol{v}}_{k:k+H_p|k}, \boldsymbol{\xi}_{\mathbf{init}}) = J(\tilde{\boldsymbol{u}}_{k:k+H_u|k}, \boldsymbol{\xi}_{\mathbf{init}}) + ||\tilde{\boldsymbol{v}}_{k:k+H_p|k}||^2_{\boldsymbol{P}},$$

and $\boldsymbol{P} \in \mathbb{R}^{H_p n_y \times H_p n_y}$ is a positive semidefinite matrix used to weight the constraint violations. The above formulation can be easily adapted when LTI, LTV, ARX, PWARX or FSR models are considered. Moreover, although additional optimization variables are added to account for constraints softening, the above optimization problem can be still formulated as a QP program.

3.4 Control models identification

The identification process aims to find a suitable structure and parametrization of a mathematical model, able to represent (as best as possible) a sequence of input-output measured data $(\tilde{\boldsymbol{u}}^{\text{meas}}, \tilde{\boldsymbol{y}}^{\text{meas}})$ collected from a plant. Different identification approaches can be adopted:

- **White-box identification**: the model structure is derived by means of first-principles physical laws that describe in detail the plant dynamics and all the parameters characterizing the system behavior are known. This approach implies a deep knowledge of the physics governing the overall system.

- **Grey-box identification**: the physics that governs the dynamics is not well known or understood, hence the model structure is derived on the basis of the insight and the available knowledge about the system. Nevertheless, the parameters used to characterize the model behavior need to be identified from the available plant data.

- **Black-box identification**: it is not possible to assume a priori a particular structure, but the model has to be developed uniquely on the basis of the measured data.

This brief introduction highlights how the structure of the model and its parameterization vary according to the particular identification approach. In this section, the **black box** paradigm is adopted. Starting from input-output measured data, various mathematical models are developed to represent the dynamics of a plant. To this end, the models structures already presented in the Section 3.2 are considered. In the following, the procedures used in this Thesis for the identification of the parameters of such models are presented.

3.4.1 Finite step response models

The parameters that characterize a FSR prediction model are the step response coefficients S^g. These terms are usually obtained as samples of a plant output whose dynamics are a consequence of the application of a unitary step input. The sampling is performed with a give time step T_s, and N terms are collected in order to fully characterize the input-output relationships. It is reminded that the index N represents the discrete time instant starting from which the output of an asymptotically stable system settles, or for stable systems it represents the time instant starting from which the output becomes a pure ramp. A visual example is given in Fig. 3.3, where the output of a plant is sampled every T_s seconds, and $N = 6$ coefficients are collected to fully represent the input-output dynamics according to the FSR notation.

If the system to be identified is linear, then a single experiment will be

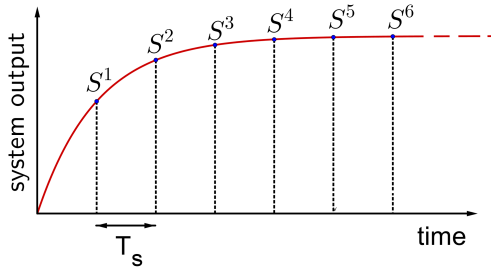


Figure 3.3: Example of FSR coefficients sampled with a given T_s. In this example, $N = 6$.

enough to describe all the possible responses that arise from step inputs of generic magnitudes (within round off errors due to possible measurements noise). On the other hand, when nonlinear systems are considered, a single experiment would not be sufficiently informative. For this reason, multiple experiments can be run and a mean model can be obtained from the collected data. Note that the concept of "informative dataset" is strictly related to the particular application tackled. In order to excite all the plant dynamics, a random input sequence should be injected to the system. An example of this approach is given by the PseudoRandom Bynary Sequences (PRBSs) (see for instance [Hunt et al., 1998, Miao et al., 2005]). However, the application of random sequences to the system often cannot be carried out for safety reasons. Therefore, a common approach involves the application of input profiles that will result to be close to the ones used during regular plant operations.

Data collection

Consider a plant having a set of indices of manipulated variables $\mathcal{C} := \{1, 2, \cdots, n_u\}$ and a set of indices of measured variables $\mathcal{B} := \{1, 2, \cdots, n_y\}$.

For a given input channel $i \in \mathcal{C}$, the FSR coefficients relating such input to every output $b \in \mathcal{B}$ are sampled according to this algorithm:

1. Define a set of different magnitudes for input channel i as $\mathcal{J}_i := \{j_{1,i}, \cdots, j_{d_i,i}\}$ and set $q = 1$
2. With the plant starting from a steady condition $(\boldsymbol{u_{ss}}, \boldsymbol{y_{ss}})$, apply a step of magnitude $j_{q,i} \in \mathcal{J}_i$ to the ith input channel, while keeping all the other inputs (i.e., $\mathcal{C} - \{i\}$) at steady values.
3. The data sampled from each bth output channel are collected in the array $\boldsymbol{\Lambda^j_{i,b}} \in \mathbb{R}^N$ and normalized according to
$$\underline{\boldsymbol{\Lambda}}^{\boldsymbol{j}}_{\boldsymbol{i,b}} = \frac{(\boldsymbol{\Lambda}^{\boldsymbol{j}}_{\boldsymbol{i,b}} - y^b_{\text{ss}})}{j_{q,i}}.$$
4. If $q = d_i$ the algorithm stops, otherwise $q = q + 1$ and go back to step 2

Note that $|\mathcal{J}_i| = d_i$, and the nomenclature $|\cdot|$ represents the cardinality of a set. Once the above algorithm has been repeated for every input magnitude, the normalized data is collected into the array

$$\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{i,b}} := [(\underline{\boldsymbol{\Lambda}}^{\boldsymbol{j_1}}_{\boldsymbol{i,b}})^\top, \cdots, (\underline{\boldsymbol{\Lambda}}^{\boldsymbol{j_d}}_{\boldsymbol{i,b}})^\top]^\top. \tag{3.31}$$

which accounts for the different experiments containing the normalized FSR coefficients relating the ith input to the bth output. The above procedure is then repeated for each input channel, and the normalized arrays (3.31) are collected for each input-output pair.

Step response coefficients identification

Once the arrays $\tilde{\boldsymbol{\Lambda}}_{\boldsymbol{i,b}}$ have been collected $\forall i \in \mathcal{C}$, and $\forall b \in \mathcal{B}$, the step response coefficients can be identified. To this aim, the best FSR coefficients

relating the ith input to the bth output, i.e.

$$\boldsymbol{S}^*_{i,b} := [S^{*,1}_{i,b},\ \cdots\ , S^{*,N}_{i,b}]^\top,$$

are obtained by minimizing

$$J_{LS}(\boldsymbol{S}_{i,b}) := \left\| \boldsymbol{\Phi}_{\mathrm{LS}} \boldsymbol{S}_{i,b} - \tilde{\boldsymbol{\Lambda}}_{\boldsymbol{i,b}} \right\|_2^2,$$

where $\boldsymbol{\Phi}_{\mathrm{LS}} := [\boldsymbol{I}_N, \boldsymbol{I}_N, \cdots, \boldsymbol{I}_N]^\top \in \mathbb{R}^{N|\mathcal{J}_i| \times N}$. The same operation is repeated for all the input-to-output relationships. Due to the absence of constraints, the solution of the above Least Squares (LS) problem can be derived analytically. The optimal solution $\boldsymbol{S}^*_{i,b}$ is obtained by evaluating

$$\boldsymbol{S}^*_{i,b} = (\boldsymbol{\Phi}_{\mathrm{LS}}{}^\top \boldsymbol{\Phi}_{\mathrm{LS}})^{-1} \boldsymbol{\Phi}_{\mathrm{LS}}{}^\top \tilde{\boldsymbol{\Lambda}}_{\boldsymbol{i,b}}, \tag{3.32}$$

where the quantity $(\boldsymbol{\Phi}_{\mathrm{LS}}{}^\top \boldsymbol{\Phi}_{\mathrm{LS}})^{-1} \boldsymbol{\Phi}_{\mathrm{LS}}{}^\top$ is usually referred to be the pseudo-inverse matrix of $\boldsymbol{\Phi}_{\mathrm{LS}}$, and shorthanded with the notation $\boldsymbol{\Phi}_{\mathrm{LS}}{}^\dagger$.

3.4.2 Autoregressive exogenous models

ARX models (3.9) predict the output of a system by means of a linear combination of past inputs and outputs values. According to their formulation, the parameters that characterize the prediction scheme are collected in the vector $\boldsymbol{\theta}$. Such vector is an unknown quantity and needs to be identified. Contrary to the FSR models, the data used for the identification of ARX ones can be obtained as a result of the application of random input profiles (not only step changes). In particular, the collected data have to be suitably informative so as to emphasize the dynamics of the plant that one wants to capture. In the following, the identification of a SISO ARX model is presented.

Data Collection

Suppose that, starting from a given initial instant $k = k_0$, the plant to be identified is operating in an equilibrium condition represented by the pair $(\boldsymbol{u}_{\mathbf{ss}}, \boldsymbol{y}_{\mathbf{ss}})$. The application of an input deviation sequence $\boldsymbol{\delta}\tilde{\boldsymbol{u}}^{\text{meas}}_{k_0:k_0+N_{u^{\text{meas}}}} \in \mathbb{R}^{N_{u^{\text{meas}}}}$ (on top of $\boldsymbol{u}_{\mathbf{ss}}$), produces an output deviation sequence $\boldsymbol{\delta}\tilde{\boldsymbol{y}}^{\text{meas}}_{k_0:k_0+N_{y^{\text{meas}}}} \in \mathbb{R}^{N_{y^{\text{meas}}}}$ (on top of $\boldsymbol{y}_{\mathbf{ss}}$) whose elements are obtained by sampling the plant output every T_s seconds. The collected data need to be structured according to the ARX formulation introduced in Section 3.2.2. Note that, due to the procedure explained above, the ARX model presented in the following will consider δu and δy quantities instead of u and y, respectively. In order to proceed to the identification of $\boldsymbol{\theta}$, the indices n_a, n_b, and n_k need to be defined. Such choice can be conducted by analyzing quantitative indices (e.g. Akaike's information criterion, the final prediction error, or the minimum description length criterion), or exploiting some prior knowledge about the plant dynamics. According to the chosen order incides, the measured data deviations are structured by assuming that they have been obtained from an ARX model, i.e.

$$\delta y^{\text{meas}}(n_m + 1) = \boldsymbol{\theta}\, \underline{\boldsymbol{s}}(n_m) + e(n_m), \tag{3.33}$$

$$\delta y^{\text{meas}}(n_m + 2) = \boldsymbol{\theta}\, \underline{\boldsymbol{s}}(n_m + 1) + e(n_m + 1),$$

$$\delta y^{\text{meas}}(n_m + 3) = \boldsymbol{\theta}\, \underline{\boldsymbol{s}}(n_m + 2) + e(n_m + 2),$$

$$\vdots$$

$$\delta y^{\text{meas}}(N_{y^{\text{meas}}}) = \boldsymbol{\theta}\, \underline{\boldsymbol{s}}(N_{y^{\text{meas}}} - 1) + e(N_{y^{\text{meas}}} - 1),$$

where $e(k)$ accounts for the measurement noise and (if present) for plant-model mismatches, $\underline{\boldsymbol{s}}(k) = [\boldsymbol{s}(k)^\top, 1]^\top$, and $n_m = \max(n_a, n_b)$. Note that the above ARX formulation differs from (3.9), because the one presented in Section 3.2.2 represents the ARX prediction scheme and not the complete ARX model. Let generalize the above prediction scheme in a matrix form as

$$\boldsymbol{\delta\tilde{y}}_{n_m+1:N_{y^{\text{meas}}}} = \boldsymbol{\Phi}_{\text{arx}}\,\boldsymbol{\theta}^\top + \boldsymbol{e}, \tag{3.34}$$

where $\boldsymbol{\Phi}_{\text{arx}} \in \mathbb{R}^{(N_{y^{\text{meas}}}-n_m)\times(n+1)}$ is

$$\boldsymbol{\Phi}_{\text{arx}} = \begin{bmatrix} \underline{\boldsymbol{s}}(n_m) & \underline{\boldsymbol{s}}(n_m+1) & \cdots & \underline{\boldsymbol{s}}(N_{y^{\text{meas}}}-1) \end{bmatrix}^\top.$$

Remark: When dealing with MIMO systems, there are not quantitative indices that can support the choice of the order matrices. For this reason, such choice it is largely based on the insight, analysis of the input-output data, and some prior knowledge of the plant.

Parameters vector identification

The structured data 3.34 are then used to identify the parameters vector $\boldsymbol{\theta}$. Similarly to the FSR case, the process of identifying the array $\boldsymbol{\theta}^*$ finds the best set of parameters able to represent the collected input-output data, which minimizes the following cost function

$$J_{\text{arx}}(\boldsymbol{\theta}) := \left\| \boldsymbol{\Phi}_{\text{arx}}\boldsymbol{\theta}^\top - \boldsymbol{\delta\tilde{y}}_{n_m:N_{y^{\text{meas}}}} \right\|_2^2.$$

Analogously to (3.32), because of the absence of constraints, the quantity $\boldsymbol{\theta}^*$ can be obtained analytically as

$$\boldsymbol{\theta}^* = {\boldsymbol{\Phi}_{\text{arx}}}^\dagger\, \boldsymbol{\delta\tilde{y}}_{n_m:N_{y^{\text{meas}}}}.$$

Remark: The aforementioned identification approach accounts for input and output deviations with respect to the equilibrium operating conditions, rather than input and output sequences. Such formulation, generalized for nonlinear plants, reduces to the form presented in Section 3.2.2 when linear plants are considered.

3.4.3 Piecewise affine autoregressive exogenous models

The identification of PWARX models has been addressed by many authors. In [Ferrari-Trecate et al., 2003] a modified *K-means* algorithm is proposed to cluster the measurement data, followed by a weighted least squares technique for parameter identification. A Kohonen neural network-based method was proposed in [Lassoued and Abderrahim, 2013] for both clustering and identification of PWARX parameters, while [Vidal et al., 2003] proposed a geometric-algebraic approach for the identification of hybrid systems. A thorough comparison of identification techniques for PWARX models is given in [Juloski et al., 2005] and references therein. Although identification toolboxes for PWARX exist (e.g., HIT [Ferrari-Trecate, 2005], PWAOAFID [Stevek and Kozak, 2011]), the support for MIMO systems is limited, the scalability with respect to the number of identification data points is poor, and there is not the possibility to account for the direct feedthrough between a given input-output pair. For these reasons, in this Thesis two tailored algorithms have been developed able to provide suitable identification processes for PWARX systems. The two approaches are presented for the SISO case, and their extension to the MIMO one is straightforward.

Data collection

The data collection process, the order indices choice, and the data structuring can be carried out similarly to the ARX case presented in Section 3.4.2. Starting from the structured data, it is possible to define the regressors arrays $\boldsymbol{s}(n_m + j)$, with $j \in \left[0; N_{y^{\text{meas}}} - 1\right]$. All of these vectors are used to form the bounded polyhedron $\mathcal{X} \in \mathbb{R}^n$, which is named as *regressors set*. The data belonging to $\mathcal{X}$ must be appropriately processed in order to partition such set in different subregions where, in each of which, different ARX models will be identified. The identification of a PWARX model usually involves three main steps:

1. define a polyhedral partition $\{\mathcal{X}_l\}_l$ of the bounded polyhedron $\mathcal{X}$

2. cluster the input-output data $(\boldsymbol{\delta\tilde{u}}^{\text{meas}}_{k_0:k_0+N_{u^{\text{meas}}}}, \boldsymbol{\delta\tilde{y}}^{\text{meas}}_{k_0:k_0+N_{y^{\text{meas}}}})$ in each partition element $\mathcal{X}_l$

3. estimate the parameters vectors $\boldsymbol{\theta}_l$ related to the ARX model in each partition element $\mathcal{X}_l$

The above three steps are carried out by means of two different approaches that are presented in the following.

Algorithm 1: Iterative grid approach

In order to partition $\mathcal{X}$, introduce a set of *cut direction indices* $\mathcal{F} = \{1, \cdots, n_c\}$. For a given value of $\bar{f} \in \mathcal{F}$, introduce also a set of *cut point indices* $\mathcal{G}_{\bar{f}} = \{1, \cdots, n_{\phi_{\bar{f}}}\}$. Fixing $\bar{j} \in \mathcal{G}_{\bar{f}}$, according to the H-representation [Grötschel and Henk, 2003], the lth element of the polyhedral partition is

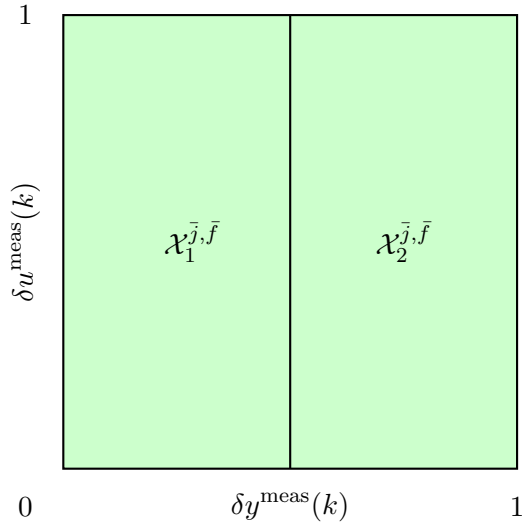


Figure 3.4: Example of a regressors set $\mathcal{X} \in \mathbb{R}^2$ partitioned in 2 different subregions.

defined as

$$\mathcal{X}_l^{\bar{j},\bar{f}} = \left\{ \boldsymbol{s} \in \mathcal{X} \,\middle|\, \boldsymbol{\Gamma}_l^{\bar{f}} \boldsymbol{s} \leq \boldsymbol{\phi}_{l,\bar{f}}^{\bar{j}} \right\}, \qquad (3.35)$$
$$\forall l = 1, \cdots, 2^q,$$

where q is the iteration number of the algorithm, $\boldsymbol{\Gamma}_l^{\bar{f}} \in \mathbb{R}^{q \times n}$ is a suitable matrix, and $\boldsymbol{\phi}_{l,\bar{f}}^{\bar{j}} \in \mathbb{R}^q$ is a suitable array. An example of this representation is given in Fig. 3.4. In this case, for a fixed cut direction and point ($\bar{f}$, and $\bar{j}$ respectively), at iteration $q = 1$ the partition element $\mathcal{X}_1^{\bar{j},\bar{f}}$ is described with $\boldsymbol{\Gamma}_1^{\bar{f}} = [1,\ 0]$ and $\boldsymbol{\phi}_{1,\bar{f}}^{\bar{j}} = 0.5$, while $\mathcal{X}_2^{\bar{j},\bar{f}}$ is represented using $\boldsymbol{\Gamma}_2^{\bar{f}} = [-1,\ 0]$ and $\boldsymbol{\phi}_{2,\bar{f}}^{\bar{j}} = -0.5$. The polyhedral partition $\left\{ \mathcal{X}_l^{\bar{j},\bar{f}} \right\}_{l=1}^{2^q}$ is *accepted* if

$$\left| \mathcal{X}_l^{\bar{j},\bar{f}} \right| \geq \upsilon,\ \forall l,$$

where υ is a tunable threshold. For each *accepted* polyhedral partition, the corresponding set of parameters vectors $\{\boldsymbol{\theta}_{l,\bar{f}}^{\bar{j}}\}_l$ is estimated according to

the LS method,

$$\boldsymbol{\theta}_{l,\bar{f}}^{*,\bar{j}} = \arg\min \left\| \mathcal{X}_l^{\bar{j},\bar{f}} \boldsymbol{\theta}_{l,\bar{f}}^{\bar{j}} - \mathcal{Y}_l \right\|_2^2, \ \forall l = 1, \cdots, 2^q,$$

where $\mathcal{Y}_l$ is the subset of $\boldsymbol{\delta\tilde{y}}_{k_0:k_0+N_{y^{\text{meas}}}}^{\text{meas}}$ that is supposed to be generated from the regressors of the lth partition element $\mathcal{X}_l^{\bar{j},\bar{f}}$, according to the PWARX model in (3.10). Finally, the model accuracy is evaluated using the fitness function

$$J_{\bar{f}}^{q,\bar{j}} = 100 \left(1 - \frac{\left\| \boldsymbol{\delta\tilde{y}}^{\text{meas}} - \boldsymbol{\delta\hat{y}}^{\bar{j},\bar{f}} \right\|_2}{\| \boldsymbol{\delta\tilde{y}}^{\text{meas}} - \delta\bar{y}^{\text{meas}} \|_2} \right), \tag{3.36}$$

where $\boldsymbol{\delta\hat{y}}^{\bar{j},\bar{f}}$ defines the prediction of $\boldsymbol{\delta\tilde{y}}^{\text{meas}}$ according to (3.10) with the polyhedral partition defined by $\boldsymbol{\Gamma}_l^{\bar{f}}$ and $\boldsymbol{\phi}_{l,\bar{f}}^{\bar{j}}$, $l = 1, \cdots, 2^q$, and $\delta\bar{y}^{\text{meas}}$ corresponds to the mean value of $\boldsymbol{\delta\tilde{y}}^{\text{meas}}$.

The fitness function in (3.36) is evaluated $\forall \bar{f} \in \mathcal{F}$ and $\forall \bar{j} \in \mathcal{G}_{\bar{f}}$. The set of parameters $(\boldsymbol{\Gamma}^{*,q}, \boldsymbol{\phi}^{*,q})$ which produces $J^{*,q} = \max_{\boldsymbol{\Gamma},\boldsymbol{\phi}} \mathcal{P}^q$ is chosen, where $\mathcal{P}^q$ is the set which contains all the fitness function results of the qth iteration.

The $(q+1)$th iteration starts with $\mathcal{X}$ already partitioned according to the $\{(\boldsymbol{\Gamma}^{*,l}, \boldsymbol{\phi}^{*,l})\}_{l=1}^{q}$ pairs, and the above procedure will be repeated.

Each iteration is considered to be *successful* if $J^{*,q+1} > J^{*,q}$. If $J^{*,q+1} \leq J^{*,q}$, the algorithm will stop. By denoting with s the number of *successful* iterations, the final PWARX model will have 2^s submodels.

Algorithm 2: K-means based approach

The objective of the *K-means* algorithm is to divide $N_r = N_{y^{\text{meas}}} - n_m$ points (which represent the data defined over the regressors set) into L clusters $\mathcal{X}_l$, $l \in \{1, \cdots, L\}$ [Jain, 2010]. This objective is accomplished in

an iterative way, by defining a suitable *distance function* $d(\boldsymbol{x}, \boldsymbol{y})$. Such function is used to represent a cluster $\mathcal{X}_l^q$ at each iteration q as

$$\mathcal{X}_l^q = \{\boldsymbol{s}|\ d(\boldsymbol{s}, \boldsymbol{C}_l^q) \leq d(\boldsymbol{s}, \boldsymbol{C}_c^q),\ \forall c \neq l\},$$

where $\boldsymbol{C}_l^q \in \mathbb{R}^n$ indicates the *centroid* of the lth cluster. Before starting iteration $q+1$, the centroid of each partition is updated according to this formula

$$\boldsymbol{C}_l^{q+1} = \frac{1}{|\mathcal{X}_l^q|} \sum_{\boldsymbol{s} \in \mathcal{X}_l^q} \boldsymbol{s}, \forall l \in \{1, \cdots, L\}.$$

At iteration $q+1$ the clusters are recomputed considering the new centroids. The algorithm finds the optimal cluster configuration $\mathcal{X}^* := \{\mathcal{X}_1^*, \mathcal{X}_2^*, \cdots, \mathcal{X}_L^*\}$ by minimizing the following cost function

$$J^q(\mathcal{X}) := \sum_{l=1}^{L} \sum_{\boldsymbol{s} \in \mathcal{X}_l} d(\boldsymbol{s}, \boldsymbol{C}_l^q).$$

The choice of the distance function $d(\boldsymbol{x}, \boldsymbol{y})$ can significantly influence the shape and structure of the optimal cluster configuration provided by the algorithm. This aspect, although generally does not pose an issue, has to be taken into strong consideration when working with optimization-based control algorithms. In fact, the particular choice of $d(\boldsymbol{x}, \boldsymbol{y})$ can determine the convexity or not of the resulting optimal partition $\mathcal{X}^*$. Therefore, in order to ensure the obtaining of convex optimization problems, in the following it is assumed that the distance function is represented by means of the Euclidean distance, i.e.

$$d(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||_2 = \sqrt{\sum (\boldsymbol{x} - \boldsymbol{y})^2}. \tag{3.37}$$

Under this assumption, the optimal cluster configuration results in a set of polyhedra $\{\mathcal{X}_l^*\}_{l=1}^L$, such that $\mathcal{X}^* = \bigcup_l \mathcal{X}_l^*$ and $\mathcal{X}_p^* \cap \mathcal{X}_j^* = \emptyset$, $\forall p \neq j$.

An example of this algorithm is given in Fig. 3.5, where some collected input-output data has been used to define $\mathcal{X}$, with $n_a = 1$, $n_b = 1$, and $n_k = 1$.

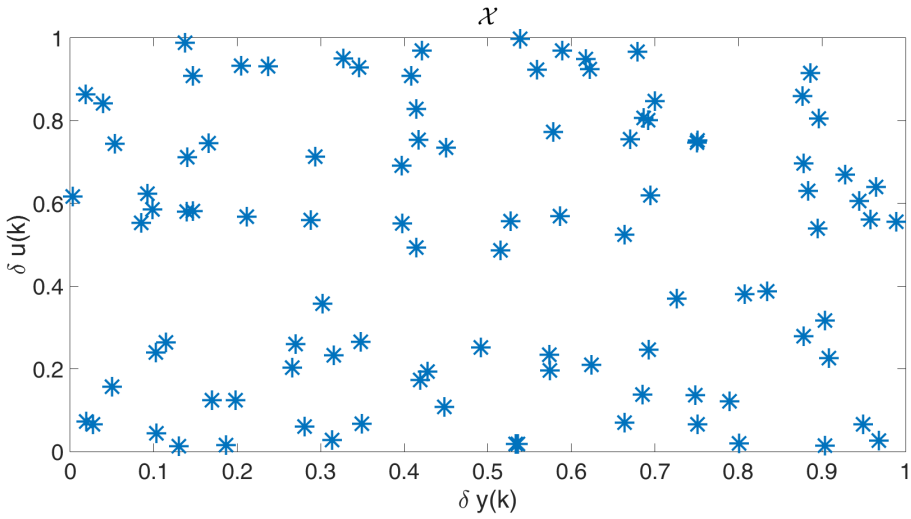


Figure 3.5: Example of input-output data composing the regressors set $\mathcal{X}$

By applying the K-means algorithm with the distance function (3.37), the partition $\{\mathcal{X}_l^*\}_{l=1}^{L}$ with $L = 30$ elements is obtained as depicted in Fig. 3.6.

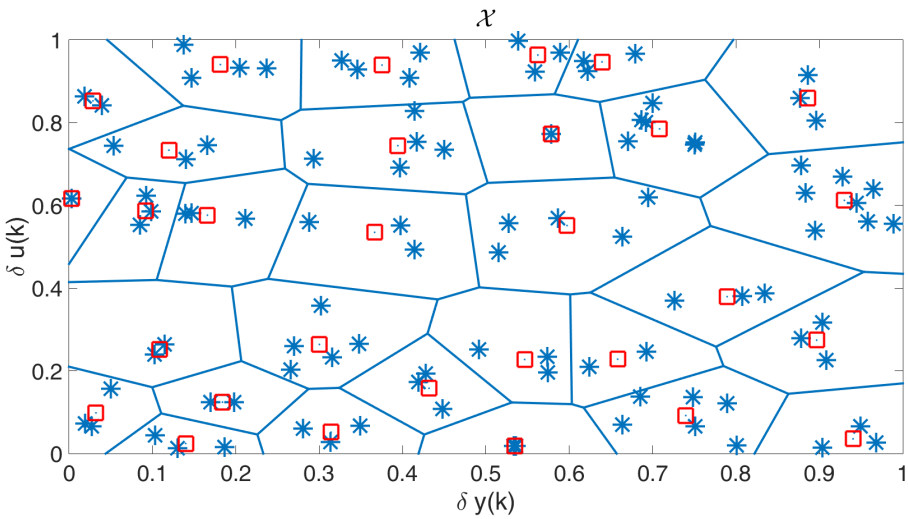


Figure 3.6: Polyhedral partition $\{\mathcal{X}_l^*\}_{l=1}^{30}$ of the regressors set $\mathcal{X}$ using the K-means algorithm. The red squares represent the optimal centroids, whereas the lines are used to represent the edges of each cluster.

After the obtaining of an optimal cluster configuration $\mathcal{X}^*$, the set of parameters $\{\boldsymbol{\theta}_l\}_{l=1}^{L}$ is identified over each partition element $\mathcal{X}_l^*$. To this end, similarly to the previous cases, a LS approach is used to find the best set of parameters $\boldsymbol{\theta}_l^*$ able to minimize

$$J(\boldsymbol{\theta}_l) := \|\mathcal{X}_l^*\boldsymbol{\theta}_l - \mathcal{Y}_l\|_2^2,$$

where $\mathcal{Y}_l$ has the same meaning as explained in the previous algorithm.

State-space partitioning

As introduced in Section 3.3.3, when dealing with MPC algorithms, it is common to reformulate ARX or PWARX dynamics in terms of state-space models. In general, for both linear or piecewise formulations, this objective can be achieved by means of realization techniques [Matsuo and Hasegawa, 2003], whose goal is to provide an equivalent state-space representation of given input-output dynamics. Let consider an ARX system of the form (3.9) having transfer function $G(z)$ (where $G(z)$ represents z transform of the impulse response of the system [Ogata, 1995]). The objective of the realization theory is to find a suitable set of state-space matrices A, B, C, and D such that

$$G(z) = C(zI - A)^{-1}B + D.$$

Note that there are infinite sets of state-space matrices that can enforce such equality. Observability or controllability canonical forms can be adopted to find state-space representations of a given input-output system which will be guaranteed to be observable or controllable, respectively.

When PWARX dynamics are considered, a number of state-space realizations has to be carried out according to the number of regions of the

regressors set (see Section 3.2.2). Moreover, the partitioning of the regressors set needs to be suitably translated into a partitioning of the state-space in which the new dynamics evolve. Given that the regressors set is composed of past values of inputs and outputs, it is preferable to perform a realization of the input-output dynamics such that the meaning of each state composing the new formulation will reflect the physical meaning of the quantities in the regressors set. For instance, let consider a PWARX model reformulated in terms of a PieceWise affine system (PWASystem) of the following form

$$\boldsymbol{\xi}(k+1) = \begin{cases} A_1\boldsymbol{\xi}(k) + B_1\boldsymbol{u}(k) + f_1 \text{ if } (\boldsymbol{\xi}(k), \boldsymbol{u}(k)) \in \mathcal{R}_1 \\ A_2\boldsymbol{\xi}(k) + B_2\boldsymbol{u}(k) + f_2 \text{ if } (\boldsymbol{\xi}(k), \boldsymbol{u}(k)) \in \mathcal{R}_2 \\ \vdots \\ A_L\boldsymbol{\xi}(k) + B_L\boldsymbol{u}(k) + f_L \text{ if } (\boldsymbol{\xi}(k), \boldsymbol{u}(k)) \in \mathcal{R}_L \end{cases}$$

$$\boldsymbol{y}(k) = \begin{cases} C_1\boldsymbol{\xi}(k) + D_1\boldsymbol{u}(k) + g_1 \text{ if } (\boldsymbol{\xi}(k), \boldsymbol{u}(k)) \in \mathcal{R}_1 \\ C_2\boldsymbol{\xi}(k) + D_2\boldsymbol{u}(k) + g_2 \text{ if } (\boldsymbol{\xi}(k), \boldsymbol{u}(k)) \in \mathcal{R}_2 \\ \vdots \\ C_L\boldsymbol{\xi}(k) + D_L\boldsymbol{u}(k) + g_L \text{ if } (\boldsymbol{\xi}(k), \boldsymbol{u}(k)) \in \mathcal{R}_L \end{cases}$$

in which $\mathcal{R}_i$, $\forall i \in \{1, \cdots, L\}$ represent the state-space partitions, and the active dynamics are chosen according to the membership of the current states and inputs $(\boldsymbol{\xi}(k), \boldsymbol{u}(k))$ to a given partition element $\mathcal{R}_i$. Despite the above formulation will not be used in the remainder of this Thesis, it has been introduced to highlight how state-space models can deal with piecewise dynamics. In fact, for computational purposes, the MLD formulation results to be more efficient than the PWASystem. For a complete overview of the different formulations of hybrid dynamical systems and their equiv-

alences, please refer to [Heemels et al., 2001] and the references therein.

Chapter 4

Advanced Battery Management Systems

Contents

4.1 Introduction

In this chapter, the main results related to the development of ABMSs based on MPC strategies are presented. All the methodologies introduced in the

previous chapters is used to develop the different control algorithms. In particular, the modeling and identification approaches presented in Section 3.2 are employed for the development of the proposed ABMSs. Finally, in order to assess the effectiveness of the control strategies, the different ABMSs are evaluated considering the real Li-ion cell represented by means of the P2D model implemented in the LIONSIMBA toolbox. The control scheme adopted in the remainder of this chapter is shown in Fig. 4.1.

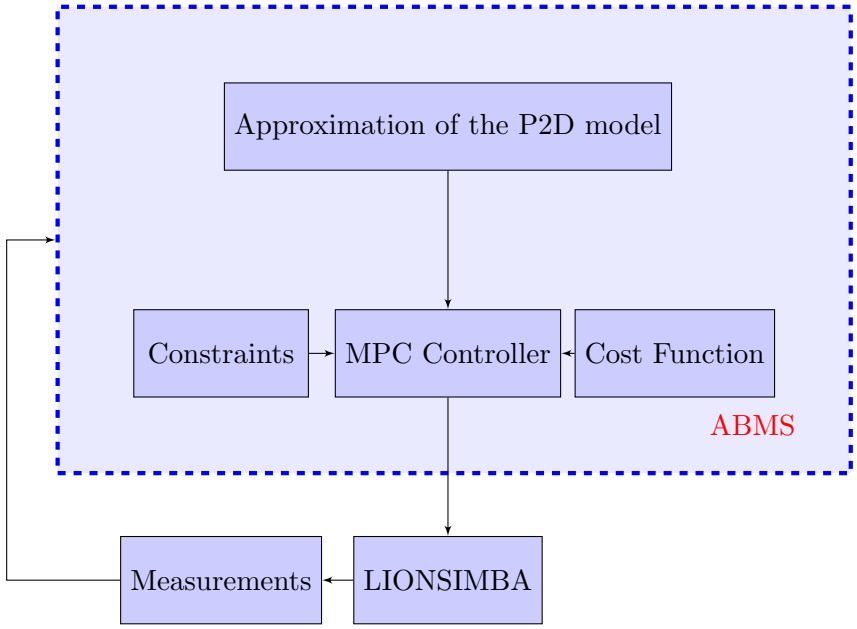


Figure 4.1: Control scheme of the proposed ABMSs. The real plant is represented with the P2D implementation of LIONSIMBA.

In all the proposed control strategies, the Li-ion cell is considered as a Single-Input Multi-Output (SIMO) plant where the applied current density $I_{\text{app}}(t)$ is the manipulated variable, while the temperature $T(t)$, voltage $V_{\text{out}}(t)$, and SOC(t) are considered measurable quantities. Only the SOC(t)

is the controlled variable. In the following, the SOC has been defined as

$$\mathrm{SOC}(t) = \frac{1}{l_n\, c_{s,n}^{\max}} \int_0^{l_n} c_s^{\mathrm{avg}}(x,t)\, dx \tag{4.1}$$

As already introduced in Section 2.7, according to the parameters used to simulate the P2D model, the 1C rate of the considered cell (i.e., I_{1_C}) is approximately equal to 30 A/m^2.

In the following, the notation $\boldsymbol{Q}_b \in \mathbb{R}^{H_p \times H_p}$ is used to refer to the diagonal matrix, portion of $\boldsymbol{Q}$, used to weight the bth output deviations. Moreover, the notation $\boldsymbol{Q}_b = \alpha$, where $\alpha \in \mathbb{R}$, is equal to $\boldsymbol{Q}_b = diag\{\alpha, \alpha, \alpha, \cdots, \alpha\}$. The same meaning is given to $\boldsymbol{R}_i \in \mathbb{R}^{H_u \times H_u}$ and $\boldsymbol{R}_i = \alpha$. Finally, the constraints violation multipliers belonging to the diagonal of $\boldsymbol{P}$, and related to the bth output are denoted with γ_b. For convenience the outputs of the Li-ion battery are represented in a compact way as $\boldsymbol{y} = [\mathrm{SOC}(t)\,, V_{\mathrm{out}}(t)\,, T(t)]$, while the input $\boldsymbol{u} = I_{\mathrm{app}}(t)$.

4.2 Fast charging algorithms for Li-ion batteries

4.2.1 Introduction

The objective of this work is to provide an optimal control strategy able to reduce the charging time of a Li-ion battery while satisfying physical constraints on output voltage, input current, and battery temperature. The outcomes of this control algorithm are obtained by approximating the P2D dynamics with a linear input-output FSR model. The approximated model is then used in a QDMC [Garcia and Morshedi, 1986] formulation to design an optimal charging strategy for the Li-ion battery. The QDMC approach is widely adopted in industrial applications and can handle constraints on the process input and output variables. Soft constraints on the output

variables and hard constraints on the input are included in the QDMC formulation. The effectiveness of the approach has been tested on the P2D model in several scenarios. The results show the suitability of the proposed approach for online implementation.

4.2.2 Simulation setup

The step response model used for output predictions was identified using the LS approach outlined in Section 3.4.1. Define $\mathcal{J} = \{ 5, 8, 11, 14, 17, 20, 23, 28, 31, 34, 37 \}$, and for each output $N = 185$ step response coefficients have been collected with a sampling time of $T_s = 10$ s. Figs. 4.2, 4.3, and 4.4 show the identified step response models for the three measured variables (Voltage, SOC, and Temperature). These outputs show an integrating behavior, which leads to a QDMC implementation using the $\mathbf{\Pi}$ matrix defined in (3.28).

The controlled variable SOC is tracked to a setpoint. The starting value of SOC is 49% (in order to simulate a half discharged battery) and the setpoint is set equal to 85%. The battery is initially kept at steady state by dropping the current to zero and waiting for the temperature to stabilize. $\boldsymbol{Q}_{SOC} = 10$ for each scenario, while the value of $\boldsymbol{R}$ varies. The control horizon H_u and prediction horizon H_p were chosen equal to 80 steps, with sampling time $T_s = 10$ s which is consistent with the bandwidth of the system. The simulations are run using Matlab® and MOSEK [Andersen and Andersen, 2016] for the optimal control algorithm on a i5 @ 3.2-GHz 64-bit CPU system with 8 GB of RAM and O.S. Ubuntu 14.04. MOSEK required CPU time to solve the optimization problem is around 0.1 s. Simulations were run under different scenarios, with the controller

parameters summarized in Table 4.1. Finally, for each of the proposed

Parameter	**Value**
$\boldsymbol{y}_{\min}$	$[45\%,\ 2.6\,\mathrm{V},\ 220\,\mathrm{K}]^\top$
$\boldsymbol{y}_{\max}$	$[95\%,\ 4.2\,\mathrm{V},\ (*)\,\mathrm{K}]^\top$
$\boldsymbol{u}_{\min}$	$0\ \mathrm{A/m^2}$
$\boldsymbol{u}_{\max}$	$I_{2_C}\ \mathrm{A/m^2}$
$\boldsymbol{\Delta u}_{\min}$	$-1.5\ \mathrm{A/(m^2\ s)}$
$\boldsymbol{\Delta u}_{\max}$	$1.5\ \mathrm{A/(m^2\ s)}$
$\boldsymbol{Q}_{SOC}$	10
$\boldsymbol{Q}_V$	0
$\boldsymbol{Q}_T$	0
$\boldsymbol{Q}_{Q_s}$	0
$\boldsymbol{R}$	(*)
$\boldsymbol{y}_{\mathrm{ref}}$	$[85,\ 0,\ 0]^\top$
$\boldsymbol{u}_{\mathrm{ref}}$	0

Table 4.1: Controller parameters. The upper bound values of the temperature $T(t)$ and the values of $\boldsymbol{R}$ are reported in Table 4.2

scenario, the configurations in Table 4.2 are considered (where not specified,

	$T(t)$	T_{env}	$\gamma_{V_{\text{out}}(t)}$	$\boldsymbol{R}$
Scenario 1:	$\leq 313.5\,\text{K}$	$298.15\,\text{K}$	10^3	$0.1, 1, 10$
Scenario 2:	$\leq +\infty\,\text{K}$	$298.15\,\text{K}$	10^2	1

Table 4.2: Simulation scenarios parameters

$\gamma_{\Delta V_{\text{out}}(t)} = \gamma_{\text{SOC}(t)} = \gamma_{V_{\text{out}}(t)} = \gamma_{T(t)} = 10^3$):

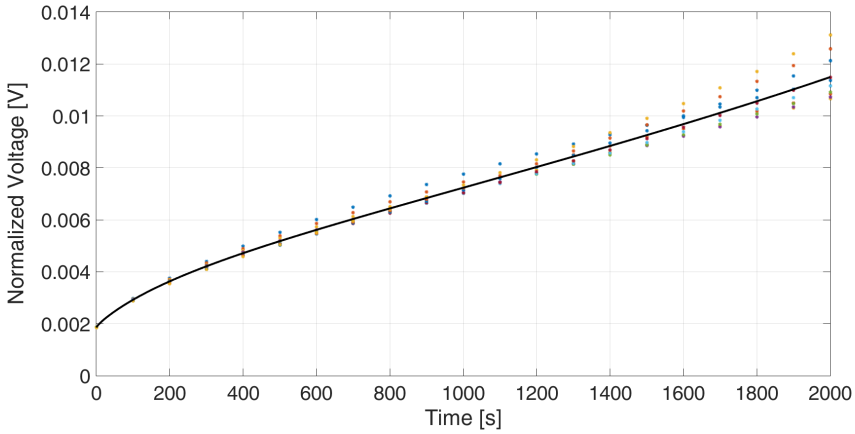


Figure 4.2: *Voltage Step Response Model*: estimated LS input-output model (solid black line) and output values obtained with different step inputs (stars).

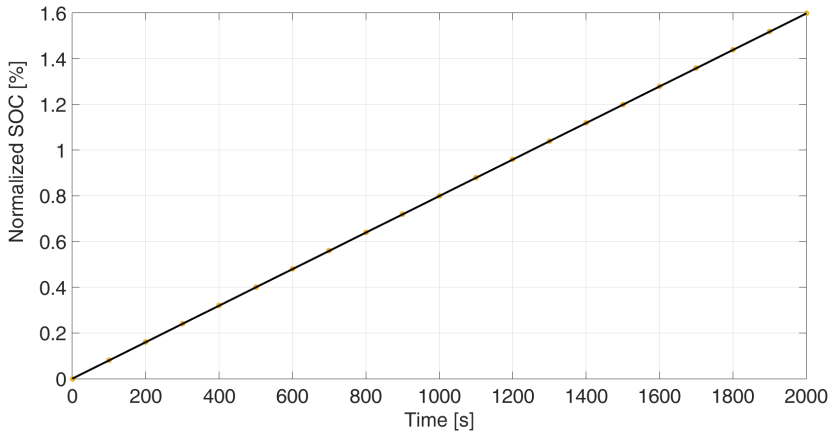


Figure 4.3: *State-of-Charge Step Response Model*: estimated LS input-output model (solid black line) and output values obtained with different step inputs (stars).

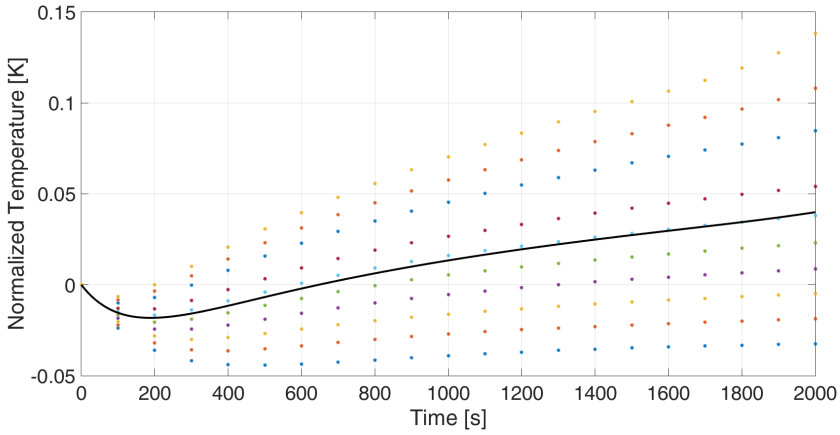


Figure 4.4: *Temperature Step Response Model*: estimated LS input-output model (solid black line) and output values obtained with different step inputs (stars).

4.2.3 Scenario 1 results

Scenario 1 evaluates the influence of $\boldsymbol{R}$ on the control action (see Fig. 4.5). The values of $\boldsymbol{R} = 0.1$ and 1 have a similar trend, while the less aggressive $\boldsymbol{R} = 10$ shows a delayed control action. The SOC profiles in Fig. 4.6 show that the specified value of 85% is reached in about 1700 s for all values of $\boldsymbol{R}$, with $\boldsymbol{R} = 10$ giving a larger value of the SOC around 500 s, followed by a lower value around 1000 s. Both voltage constraints are satisfied for all time for $\boldsymbol{R} = 1$ and 10 (Fig. 4.7), with a slight violation of the upper bound at 1500 s for $\boldsymbol{R} = 0.1$, with the voltage approaching 4.1 V at longer times for all values of $\boldsymbol{R}$. The temperature constraint is slightly violated around 500 s for each value of $\boldsymbol{R}$ (Fig. 4.8), which is permitted by the constraint softening. The violation can be reduced or removed by more strongly weighing the soft constraint or by shifting the upper bound on the temperature in the algorithm down by a small amount.

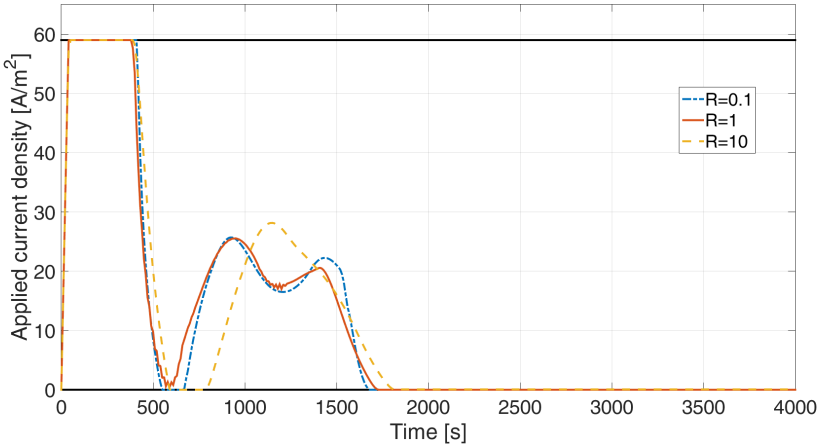


Figure 4.5: *Input current*: solid black lines are the hard constraints on the input.

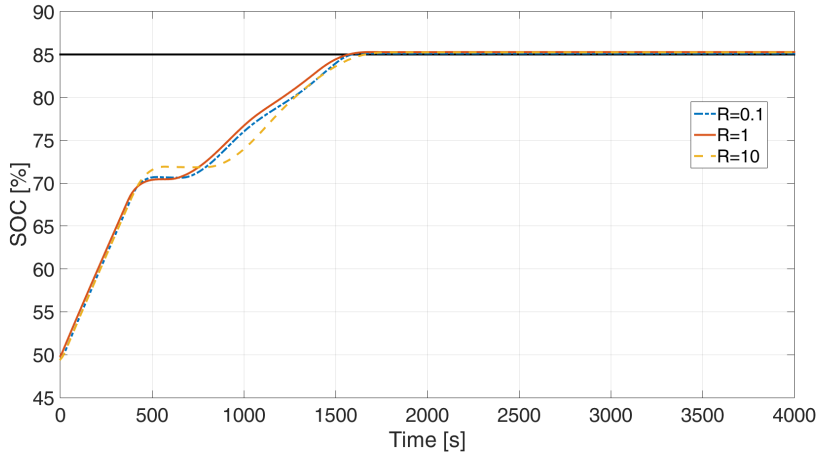


Figure 4.6: *SOC*: black horizontal line represents the reference value.

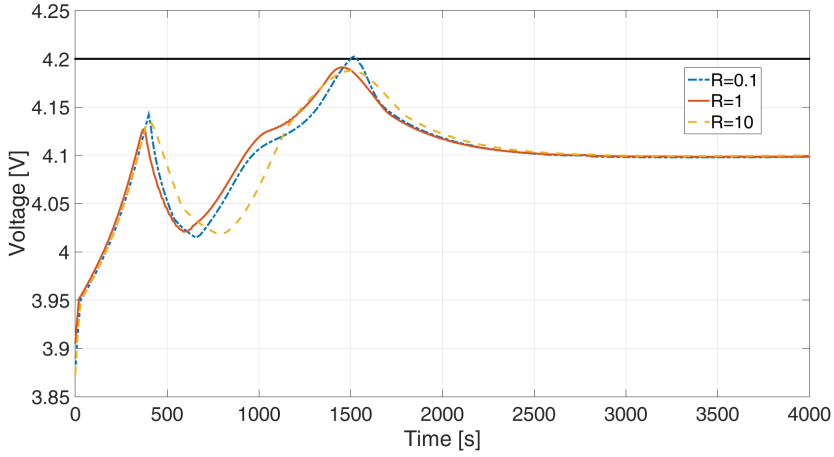


Figure 4.7: *Voltage*: black horizontal line is the voltage upper soft constraint.

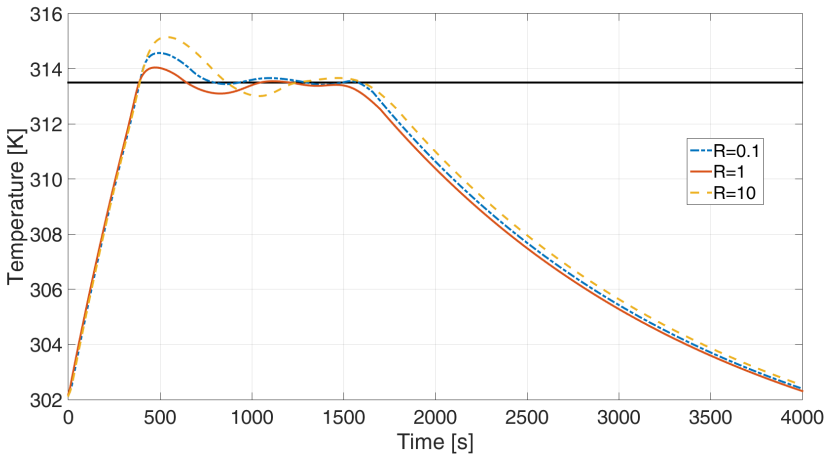


Figure 4.8: *Temperature*: black horizontal line is the upper bound soft constraint.

4.2.4 Scenario 2 results

To reduce the battery charging time, this scenario relaxes the temperature upper bound constraint and voltage violation multiplier $\gamma_{V_{\text{out}}(t)}$. As expected, the temperature and voltage go over their hypothetical upper bounds (see Figs. 4.12 and 4.11), by about 10 K and 0.08 V, respectively. In contrast with Scenario 1 (Fig. 4.5), the injected current for Scenario 2 in Fig. 4.9 does not drop nearly to zero at 600 s to remodulate itself for enforcement of the temperature constraints. Dropping the constraints results in the input current almost immediately reaching its maximum allowed value (I_{2_C}) for the entire charging process and falling only when the SOC is near its final setpoint. These plots show the high importance of including temperature constraints when optimally charging batteries.

The SOC in Scenario 2 reaches the reference value of 85% in about 1200 s (Fig. 4.10), which is nearly ten minutes less than in Scenario 1. The slope of the SOC is nearly constant for the entire charging process, to reach the setpoint earlier.

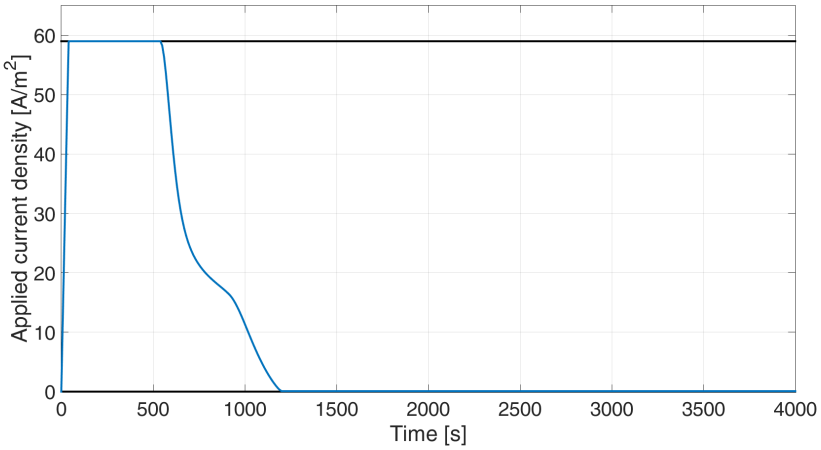


Figure 4.9: *Input current*: actual (blue solid) and hard constraints (black line).

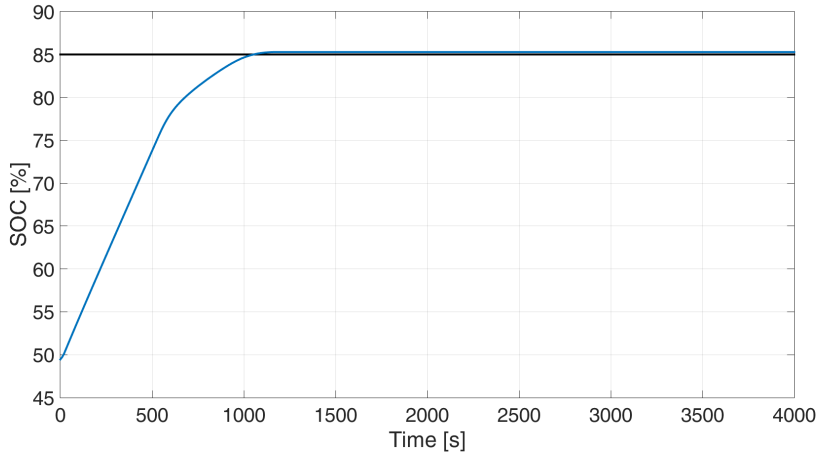


Figure 4.10: *SOC*: actual (blue solid), reference (black horizontal).

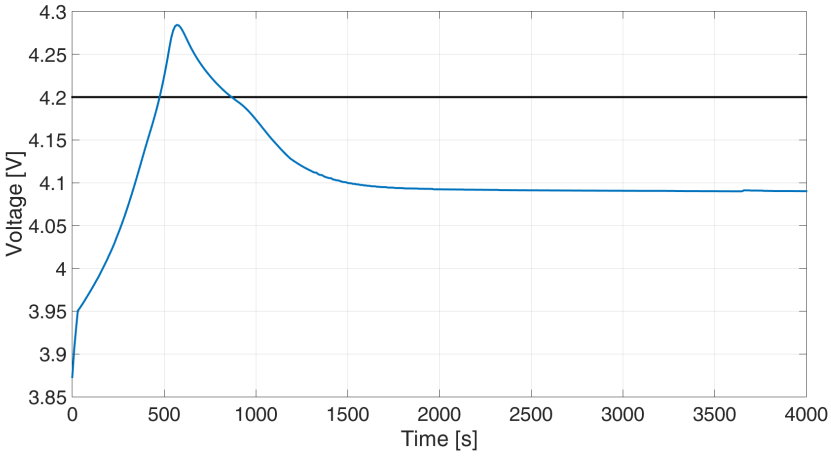


Figure 4.11: *Voltage*: actual (blue solid) and upper bound soft constraint (black line).

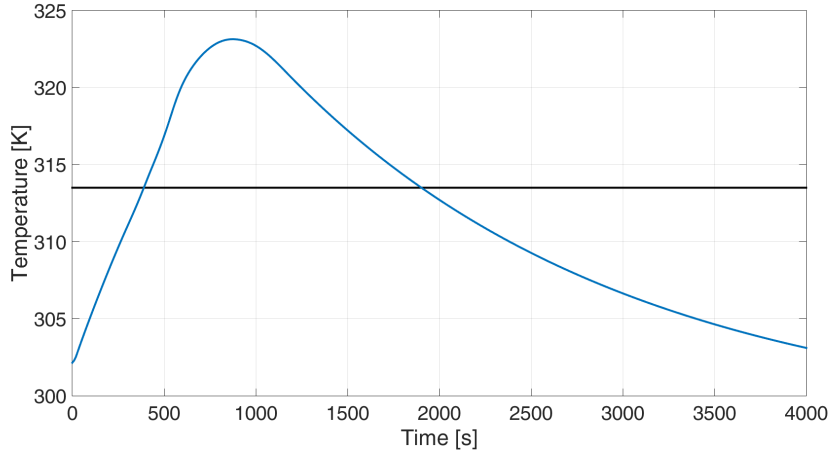


Figure 4.12: *Temperature*: actual (blue solid) and the removed upper bound soft constraint (black line).

4.2.5 Summary

In the simulations, an approximation of the P2D model is developed and incorporated into a quadratic dynamic matrix controller to obtain a fast charging protocol. The control algorithm accounts for constraints on current, SOC, battery temperature, and voltage, and it is designed in such a way that the constraints are allowed to be softened in order to improve the performance. Simulation results show that temperature constraints play a fundamental role in the optimal charging process of a battery. In fact, by removing temperature upper bounds and performing a minimal violation of the voltage constraints (+0.08V), the charging time was reduced by roughly 600s. Nevertheless, an appropriate handling of temperature constraints need to be enforced in order to avoid possible safety issues. The adoption of a FSR input-output model has provided good control results, while being a linear approximation of the P2D dynamics. Simulations indicate the suitability of this approach to real-time applications.

4.3 Health-aware charging protocols for Li-ion batteries

4.3.1 Introduction

Long-term degradation effects occurring during battery operations are important when designing ABMSs. The modeling of these effects has been addressed by many authors in the recent years (for example, see [Sankarasubramanian and Krishnamurthy, 2012], [Zhang and White, 2008]). Among them, a first-principles model able to accurately describe the capacity fade mechanisms has been proposed in [Ramadass et al., 2003]. This model is able to reproduce: (i) the formation of the solid-electrolyte interface layer, which results in an additional and variable resistance between the electrolyte and active material (anode), and (ii) the capacity fade effects which lead to a continuous loss of capacity during the battery cycling.

The objective of this investigation is to design an ABMS for Li-ion batteries using the P2D model in combination with a description of the capacity fade mechanisms. The inclusion of the latter mechanisms allows the design of health-aware charging strategies. The ABMS proposed in the following is based on a QDMC algorithm ([Garcia and Morshedi, 1986]), where FSR models are used to approximate the input-output dynamics of the P2D model. Differently from the results proposed in the previous section ([Torchio et al., 2015]), the inclusion of the aging dynamics allows the design of a health-aware charging protocol by adding soft constraints on the battery capacity fade on top of the already existing constraints on voltage and temperature. In simulations, the proposed approach shows that different tradeoffs between battery aging and minimum time charging can

be obtained by properly tuning the control parameters.

4.3.2 Aging model

To consider battery aging, the model introduced in Section 2.4 is extended to account for the dynamics describing the capacity fade effects and the formation of the SEI layer at the electrolyte-anode interface. To this end, the formulation of the ionic flux in (2.8) needs to be modified as follows

$$j(x,t) = j_{\text{int}}(x,t) + j_{\text{side}}(x,t),$$

where $j_{\text{int}}(x,t)$ represents the regular intercalation/deintercalation flux (exactly as defined in (2.8)), while $j_{\text{side}}(x,t)$ accounts for side reactions that occur during the charging of the battery. Given that the side reactions are considered to occur only at the electrolyte-anode interface, the contribution of $j_{\text{side}}(x,t)$ at the cathode side is null and no SEI resistance between cathode and electrolyte is considered in the following. As in [Ramadass et al., 2004], the side reaction flux is modeled using a Tafel relation:

$$j_{\text{side}}(x,t) = -\frac{i_{0,\text{side}}(t)}{F} \exp\left(\frac{0.5F}{\mathrm{R}T(x,t)}\eta_{\text{side}}\right), \tag{4.2}$$

where $i_{0,\text{side}}(t)$ is the side reaction exchange current and η_{side} is the side reaction anodic overpotential defined as

$$\eta_{\text{side}} := \Phi_s(x,t) - \Phi_e(x,t) - U_{\text{SEI}} - Fj(x,t)R_f(t),$$

where the term U_{SEI} represents the side reaction OCV and $Fj(x,t)R_f(t)$ accounts for an extra voltage drop due to the presence of the SEI resistance $R_f(t)$. The growth of the SEI layer is modeled as

$$\frac{\partial}{\partial t}\delta(x,t) = -\frac{M_w}{\rho} j_{\text{side}}(x,t), \tag{4.3}$$

where M_w is the molar weight of the electrode and $\delta(x,t)$ represents the film thickness. The overall film resistance is given by

$$R_f(t) = R_{\text{SEI}} + \frac{\bar{\delta}(t)}{\nu},$$

where R_{SEI} is the initial SEI layer resistance, ν is the admittance of the film, and

$$\bar{\delta}(t) = \frac{1}{l_n}\int_0^{l_n} \delta(x,t)\,dx.$$

The side reaction exchange current $i_{0,\text{side}}(t)$ depends on the battery applied current density $I_{\text{app}}(t)$ ([Rashid and Gupta, 2014]). No experimental data are available for the identification of such relation, and the empirical equation

$$i_{0,\text{side}}(t) = i_{0,\text{base}}\left(\frac{I_{\text{app}}(t)}{I_{1C}}\right)^w \tag{4.4}$$

is adopted. At the anode side, due to the presence of the SEI layer, the diffusion process of Li-ions within the electrode (2.3), (2.4) is driven from $j(x,t) = j_{\text{int}}(x,t)$. The dynamics describing the capacity fade effects can be represented by the ODE

$$\frac{\partial Q_s}{\partial t} = -a_n F\int_0^{l_n} j_{\text{side}}(x,t)dx,$$

where $Q_s(t)$ accounts for the capacity fading as a function of the side reaction flux. At the end of each charge cycle N_c, an estimate of the overall lost capacity is computed as

$$Q_s^{N_c} = Q_s(t_f^{N_c})$$

where $t_f^{N_c}$ represents the duration of the N_cth charging cycle. Before starting the next discharging cycle, the initial concentration of the anode is

updated by

$$c_s^{\text{avg,init}} \leftarrow c_s^{\text{avg,init}} - \frac{Q_s^{N_c} - Q_s^{N_c-1}}{F\epsilon_n l_n}.$$

According to this scheme, the long-term degradation effects considered in this work produce a reduction of the cell capacity after each charging cycle. After the first charging cycle (i.e. $N_c = 1$), the quantity Q_s^0 is initialized at zero.

4.3.3 Simulation setup

The effectiveness of the proposed health-aware control algorithm is demonstrated in this section. Besides voltage, temperature, SOC, and current density, also the lost capacity $Q_s(t)$ is assumed to be a measurable variable. The controller presented in Section 3.3.4 has been implemented to track a reference value of the SOC while fulfilling input and output constraints. The FSR coefficients used for online optimizations, were estimated according to the approach outlined in Section 3.4.1. Voltage, temperature, and SOC coefficients are the same as the ones presented in Section 4.2. Additionally, FSR coefficients related to the aging dynamics have been identified, as depicted in Fig. 4.13. Due to the presence of an additional measured output (i.e., $Q_s(t)$), in the following of this section a new output set is defined

$$\underline{\boldsymbol{y}} = [\boldsymbol{y},\, Q_s(t)]\,.$$

All the proposed scenarios share the constraints reported in Table 4.3. To assess the effectiveness of the proposed control algorithm, the simulations have been run considering different values of $\gamma_{\Delta Q_s}$. The parameters used for the side reaction dynamics are the base-side reaction current $i_{0,\text{base}} =$

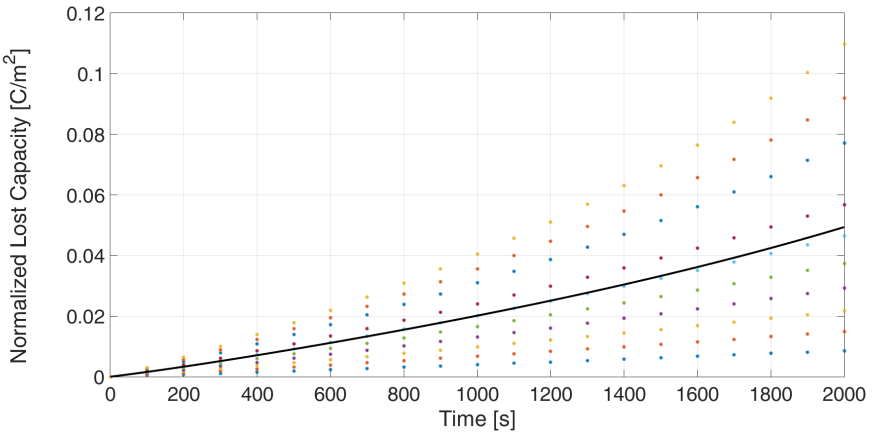


Figure 4.13: *Capacity fade FSR model*: identified coefficients (solid black line) and data (stars) obtained by application of the set of input variations $\mathcal{J} = \{5, 8, 11, 14, 17, 20, 23, 28, 31, 34, 37\}$ A/m^2, starting from a rest condition, to the battery.

8×10^{-11} A/m^2, the film conductivity $\nu = 3.79 \times 10^{-7}$ S/m, and $w = 2$. The molar weight is obtained from the cell parameters as $M_w = 73 \times 10^{-3}$ kg/mol and the side reaction OCV is set to $U_{\text{side}} = 0.4$ V. All the scenarios run with the environmental temperature of $T_{\text{env}} = 298.15$ K. The sampling time has been chosen $T_s = 10$ s. Prediction and control horizon have been set to $H_p = H_u = 200$ steps. In all the scenarios the QDMC controller has been set up with $\boldsymbol{Q}_{SOC} = 10$ and $\boldsymbol{R} = 1$. The resulting QP problem is solved using the Matlab® quadprog solver. The simulations are performed on a Windows 10 machine with 8 Gbytes of RAM and a i5 vPro processor @2.5 GHz.

Parameter	Value
$\underline{\boldsymbol{y}}_{\min}$	$[0.8\%,\ 2.5\,\mathrm{V},\ 270\,\mathrm{K},\ 0\ \mathrm{C/m^2}\ ]^\top$
$\underline{\boldsymbol{y}}_{\max}$	$[95\%,\ 4.2\,\mathrm{V},\ 320\,\mathrm{K},\ \infty\ \mathrm{C/m^2}\ ]^\top$
$\boldsymbol{\Delta}\underline{\boldsymbol{y}}_{\max}$	$\left[\infty\%/\mathrm{s},\ \infty\,\mathrm{V/s},\ \infty\,\mathrm{K},\ 10^{-9}\ \mathrm{C/(m^2\ s)}\right]^\top$
$\boldsymbol{u}_{\min}$	$0\ \mathrm{A/m^2}$
$\boldsymbol{u}_{\max}$	$I_{1_C}\ \mathrm{A/m^2}$
$\boldsymbol{\Delta u}_{\min}$	$-1.5\ \mathrm{A/(m^2\ s)}$
$\boldsymbol{\Delta u}_{\max}$	$1.5\ \mathrm{A/(m^2\ s)}$
$\boldsymbol{Q}_{SOC}$	10
$\boldsymbol{Q}_V$	0
$\boldsymbol{Q}_T$	0
$\boldsymbol{R}$	1
$\underline{\boldsymbol{y}}_{\mathrm{ref}}$	$[50,\ 0,\ 0,\ 0]^\top$
$\boldsymbol{u}_{\mathrm{ref}}$	0

Table 4.3: Controller parameters.

4.3.4 Results

In Figures 4.14 to 4.18 different charging protocols are computed by means of different weights of $\gamma_{\Delta_{Q_s}}$. As it is possible to see with $\gamma_{\Delta_{Q_s}} = 0$ (dashed purple line), the algorithm provides a charging current density which is for most of the time set to the maximum value of I_{1_C} and starts to drop when approaching the final charging stage at around 1300 s. The corresponding behavior of the output voltage shows a fast increase of $V_{\text{out}}(t)$ followed by a rest transient which settles to 3.87 V. Finally, the temperature shows a steeper increase which reaches 300 K at 1200 s and the capacity fade effect which results in a total decrease of the anode concentration of 0.35 mol/m^3. By weighting the aging dynamics in the optimization, QDMC provides more conservative control actions. The applied current profiles have different shapes according to the different weights: $\gamma_{\Delta_{Q_s}} = 1$ (yellow line), $\gamma_{\Delta_{Q_s}} = 5$ (dotted orange line), and $\gamma_{\Delta_{Q_s}} = 10$ (dot-dashed blue line). By increasing $\gamma_{\Delta_{Q_s}}$, the control action provided by QDMC is less aggressive and lasts longer. The voltage profiles show a more gentle rise during the charging process and all of them settle at 3.87 V, while temperature never goes above 298.7 K. In all the different cases, the reference value of the SOC is reached at different time instant; according to the most conservative control action, the reference is reached in 6000 s. Finally, after one charging cycle, the overall lost capacity of the most conservative profile results to be less than 1/3 of most aggressive control action. The reduction in anode capacity over multiple charging cycles are compared in Fig. 4.19, whereas Table 4.4 summarizes the lost anode capacity as a function of the cycle number. In order to compare the obtained results over multiple cycles, a common operation protocol has been defined:

1. Starting from a rest condition, charge the battery from the 20% to the 50% of SOC with the proposed health-aware algorithm
2. Stop when the applied current drops below 10^{-3} A/m^2
3. Apply 0 [A/m^2] for a rest period of 1300 s
4. Discharge the battery down to 20% of SOC with a galvanostatic current of -20 A/m^2
5. Apply 0 A/m^2 for a rest period of 1300 s
6. Go back to step 1

After 10 charging cycles, the total lost capacity is significantly reduced when aging dynamics are weighted in the optimization, which magnifies the results obtained over a single charging cycle.

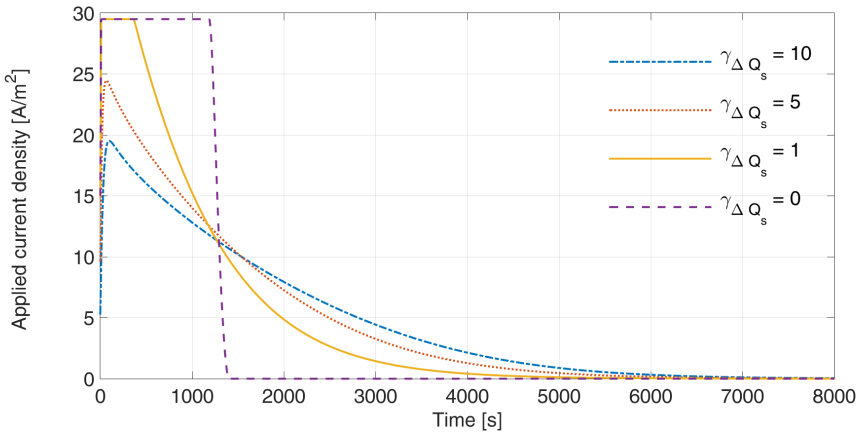


Figure 4.14: $I_{\text{app}}(t)$ comparison for different violation weights.

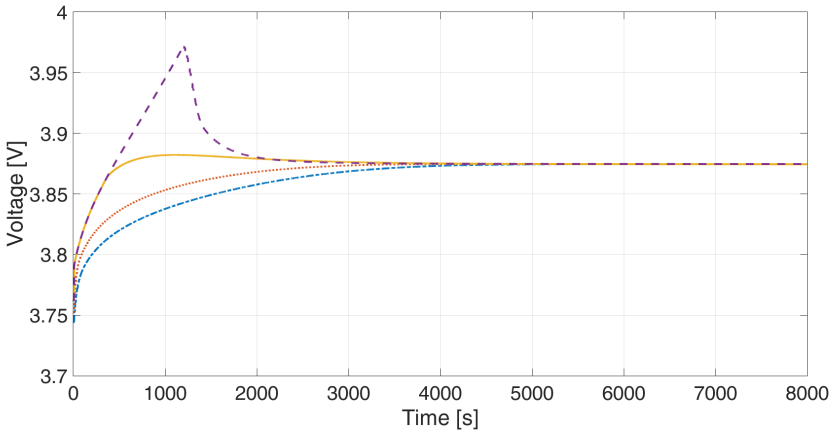


Figure 4.15: $V_{\text{out}}(t)$ comparison for different violation weights.

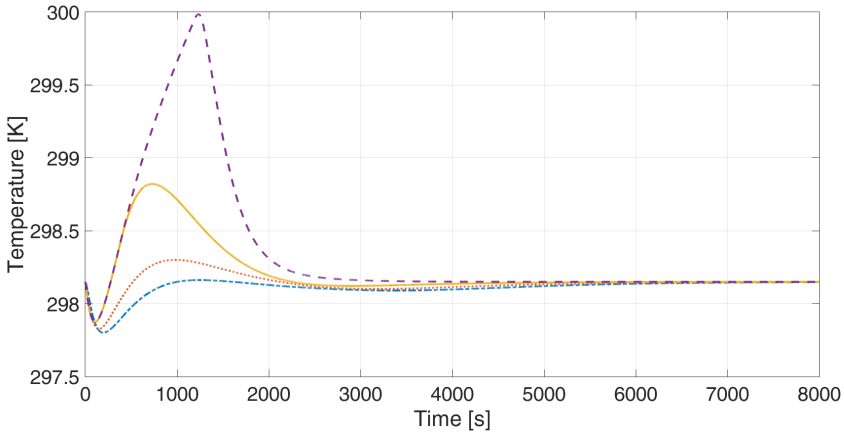


Figure 4.16: $T(t)$ comparison for different violation weights.

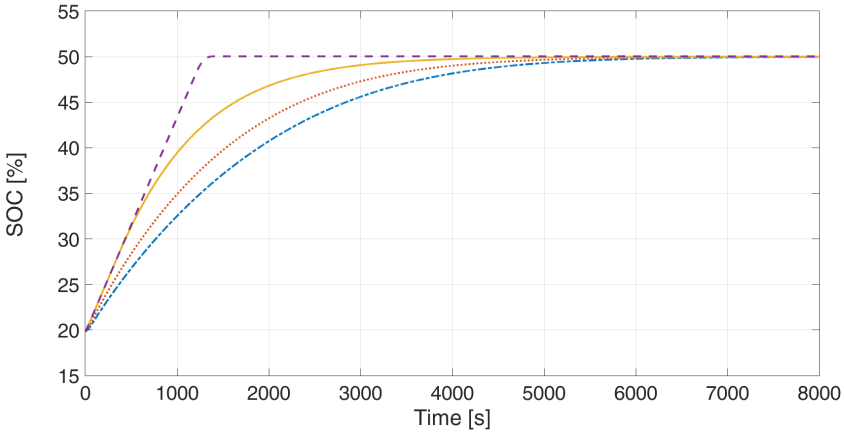


Figure 4.17: SOC(t) comparison for different violation weights.

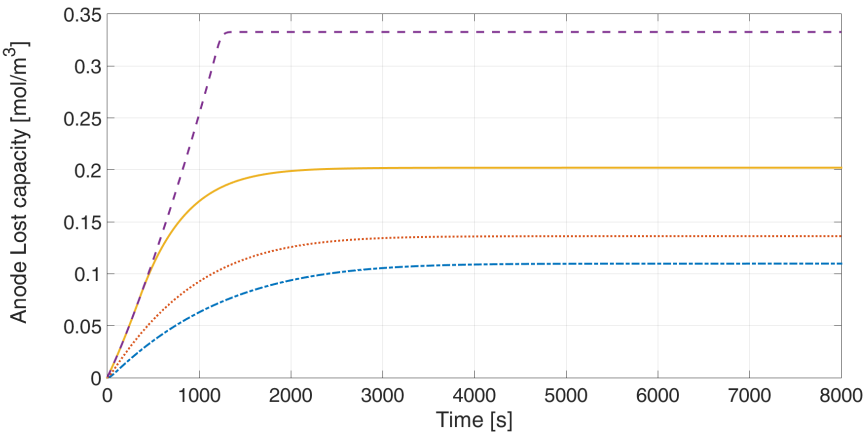


Figure 4.18: Lost anode capacity for different violation weights.

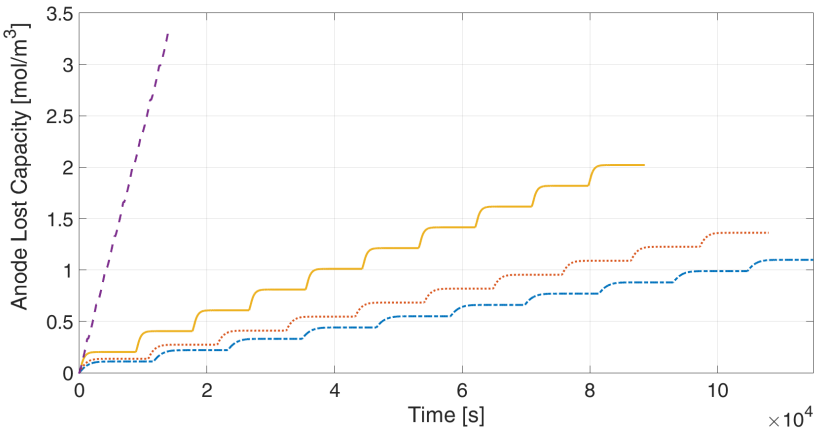


Figure 4.19: Lost anode capacity over multiple cycles with different violation weights.

Cycle #	$\gamma_{\Delta Q_s} = 0$	$\gamma_{\Delta Q_s} = 1$	$\gamma_{\Delta Q_s} = 5$	$\gamma_{\Delta Q_s} = 10$
1	0.3327	0.202	0.1362	0.1098
5	1.661	1.011	0.6829	0.5498
10	3.322	2.021	1.362	1.099

Table 4.4: Lost anode capacity over multiple cycles.

4.3.5 Summary

An ABMS is investigated that accounts for long-term degradation effects. Starting from the P2D model, additional equations are included in order to consider the aging dynamics. A QDMC predictive approach is adopted to perform an optimal charge of a Li-ion cell while taking into account both input and outputs constraints. A linear representation of the Li-ion battery is employed to reduce the computational burden while still guaranteeing good control performance. By varying the multiplier $\gamma_{\Delta_{Q_s}}$, different protocols can be obtained. The results show that the aggressiveness of the charge profiles provided by the controller have a direct impact on the loss of capacity. In fact, despite longer charging times, a significant life-cycle improvement can be gained by defining more conservative charging protocols. Moreover, the results obtained after a single charging cycle have been magnified by evaluating the influence of different values of $\gamma_{\Delta_{Q_s}}$ over multiple charging cycles, where a common charging and discharging protocol is provided. The proof of principle provided by this work highlights the capabilities of predictive algorithms for use in a health-aware ABMS application.

4.4 Optimal charging of a Li-ion cell: A hybrid model predictive control approach

4.4.1 Introduction

When designing an ABMS, an accurate model of the thermal behaviour is quite important, since it allows to reduce safety risks, possible damages, and, in extreme cases, avoid thermal runaways [Bernardi et al., 1985, Bandhauer et al., 2011]. In this section, we introduce a PWARX approximation of the P2D model that well describes the temperature dynamics in a Li-ion cell. Starting from a set of input-output data collected from the P2D EM, the PWARX model is identified using a tailored algorithm (explained in detail in Section 3.4.3, algorithm 1) and converted into a state-space model using the MLD formulation [Bemporad and Morari, 1999]. According to Section 3.3.3, it is possible to use the obtained model (which involves both continuous and binary variables) to formulate a Hybrid MPC (HMPC) [Camacho and Alba, 2013] problem which requires, at each time step, the solution of a MIQP program. While the hybrid nature of the approximation increases the complexity of the MPC problem when compared to the linear case, the time required to solve the optimization is still compatible with the sampling time. Moreover, simulations show that the HMPC provides better performance in terms of constraint satisfaction.

4.4.2 Simulation setup

As a first step, data are collected and used to identify the PWARX model included into the HMPC algorithm according to the procedure outlined in Section 3.4.3. To this aim, consider the battery working at the equilibrium

point defined by $V_{ss} = 3.733$ V, $SOC_{ss} = 19.69\%$, $T_{ss} = 298.15$ K, and $I_{ss} = 0$ A/m^2. Starting from this condition, a set of random variations of $I_{\text{app}}(t)$ have been applied to the P2D model, and the resulting set of input-output identification data has been collected as depicted in Fig. 4.20. Note that, as outlined in Section 3.4, the input sequence applied to the P2D model has been chosen to be as close as possible to the input profiles used during regular operating conditions of the cell. The order matrices used to identify the PWARX model are

$$\boldsymbol{N_a} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \end{bmatrix}, \ \boldsymbol{N_b} = \begin{bmatrix} 1 \\ 2 \\ 5 \end{bmatrix}, \ \boldsymbol{N_k} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}. \tag{4.5}$$

This particular choice of the order matrices was made upon the analysis of the cell dynamics (even though it does not represent the only suitable configuration). As shown in Fig. 4.20, the SOC(t) exhibits an integral behavior with respect to the applied current density, whereas $V_{\text{out}}(t)$ also exhibits higher order dynamics during relaxation periods. Moreover, the dynamics of the voltage shows a direct feedthrough with respect to $I_{\text{app}}(t)$, which motivates the structure of $\boldsymbol{N_k}$. The temperature is mainly affected by the applied current density and the SOC. Whereas SOC(t) and $V_{\text{out}}(t)$ are well described by linear models, a linear approximation of $T(t)$ provides poor performance (see last row of Table 4.5, where the results refer to the validation dataset). Indeed, inspection of Fig. 4.21 indicates that (*i*) undershoots in the temperature profile can be observed for certain values of the SOC, (*ii*) a change of rising slope can be found according to the duration of the charging or discharging current, and (*iii*) the temperature rises in

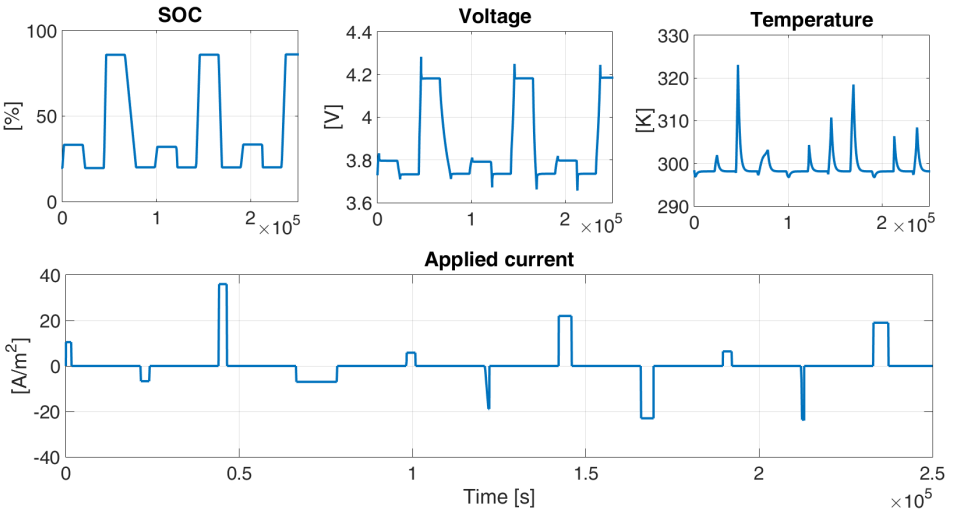


Figure 4.20: Input-output data from the P2D model. Before starting to identify the PWARX model, the data have been normalized with respect to the equilibrium point $V_{ss} = 3.733$ V, $SOC_{ss} = 19.69\%$, $T_{ss} = 298.15$ K, and $I_{ss} = 0$ A/m^2.

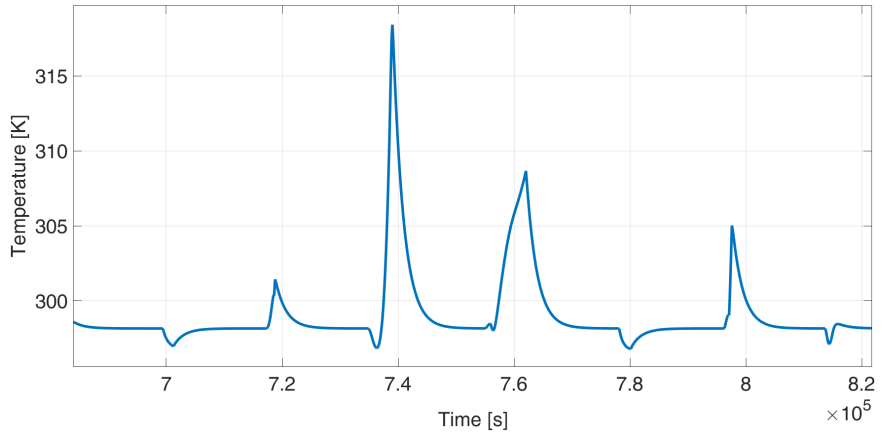


Figure 4.21: A portion of the temperature profile of the identification dataset $\mathcal{I}$.

the presence of either positive or negative values of $I_{\text{app}}(t)$. Therefore, in order to better capture these nonlinear phenomena, a PWARX model was employed instead, where the partitioning of the regressors set was carried out according to the algorithm 1 outlined in Section 3.4.3. On the other side, the SOC and the voltage behaviors were described with ARX models obtained from *arx* command of the Matlab® identification toolbox. The choice made in (4.5) implies that the regressors set $\mathcal{X}_T \subset \mathbb{R}^8$. A total of 58 different *cut directions* were tested and, for a given direction, a gridding approach has been used to define the set $\mathcal{G}_{\bar{f}}$. The iterative algorithm stopped after 2 iterations and found 4 different submodels. For comparison purposes, a linear ARX model representing the overall input-output cell dynamics has been identified according to the order matrices in (4.5). According to Section 3.4.3, all the data used for identification purposes have been expressed in terms of deviations with respect to the equilibrium conditions.

The fitness function in (3.36) has been evaluated for the ARX and PWARX cases and their results are compared in Table 4.5, with some of the validation results presented in Fig. 4.22 (the validation was performed with respect to a completely different dataset from the identification one, still obtained from the P2D model).

	ARX	PWARX
SOC(t)	96%	96%
$V_{\text{out}}(t)$	83%	83%
$T(t)$	−3%	83%

Table 4.5: Fitness function values for the ARX and PWARX models evaluated using (3.36). with respect to validation data.

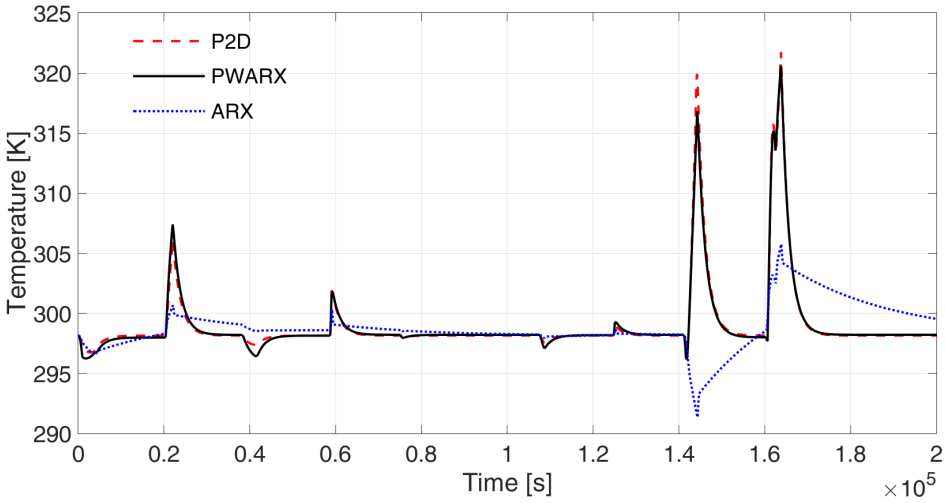


Figure 4.22: Comparison of the temperature profiles for the P2D (true), ARX, and PWARX models.

Besides the quantitative index presented in Table 4.5, the graphical comparison in Fig. 4.22 highlights that the adoption of a set of affine models (instead of a single linear model) allows to obtain much better prediction performance. The constraints and the controller parameters for all of the simulations are summarized in Table 4.6. The control and prediction horizons were set as $H_p = H_u = 20$, with the objective of the ABMS to track the SOC to a reference value of 75%, starting from 20%. The weight matrices have been set to $\boldsymbol{Q}_{SOC} = 10$, $\boldsymbol{R} = 1$, while $\boldsymbol{P} = 3 \cdot 10^3$. The simulations are performed using Matlab® . The commercial solver CPLEX [IBM, 2010] was used to solve the resulting MIQP problem on a i7@ 2.7-Ghz 64-bit CPU system with 8 Gbytes of RAM and Ubuntu 14.04 machine. The sampling time chosen for this application is $T_s = 80$ s, which is suitable with the estimated bandwidth of the system. The control algorithm presented in Section 3.3.3 is adopted. Given that the P2D approximation embed-

ded into the ABMS has been identified using data expressed as deviations with respect to the equilibrium conditions, the algorithm in (3.19) considers the optimization variables $\tilde{\boldsymbol{u}}_{k:k+H_u|k} = \boldsymbol{\delta}\tilde{\boldsymbol{u}}_{k:k+H_u|k} + \boldsymbol{u}_{\mathbf{ss}}$ and the outputs $\tilde{\boldsymbol{y}}_{k:k+H_p|k} = \boldsymbol{\delta}\tilde{\boldsymbol{y}}_{k:k+H_p|k} + \boldsymbol{y}_{\mathbf{ss}}$. The PWARX-based HMPC algorithm took a mean of 0.0659 s to solve the online MIQP at each time step, while the ARX-based MPC algorithm took a mean of 0.0210 s to solve the resulting online QP problem.

Parameter	**Value**
$\boldsymbol{y}_{\min}$	$[0\%,\ 2.5\,\mathrm{V},\ 290\,\mathrm{K}]^\top$
$\boldsymbol{y}_{\max}$	$[95\%,\ 4.2\,\mathrm{V},\ 306.5\,\mathrm{K}]^\top$
$\boldsymbol{u}_{\min}$	$0\ \mathrm{A/m^2}$
$\boldsymbol{u}_{\max}$	$I_{1.35_C}\ \mathrm{A/m^2}$
$\boldsymbol{Q}_{SOC}$	10
$\boldsymbol{Q}_V$	0
$\boldsymbol{Q}_T$	0
$\boldsymbol{R}$	1
$\boldsymbol{y}_{\mathrm{ref}}$	$[75,\ 0,\ 0]^\top$
$\boldsymbol{u}_{\mathrm{ref}}$	0

Table 4.6: Controller parameters.

4.4.3 Results

The MPC-based and HMPC-based ABMSs are compared in Figures 4.23 to 4.26. Both control actions drive the Li-ion cell to the reference SOC of 75%. The HMPC algorithm is able to better accommodate the temperature constraints by re-modulating the input current earlier with respect to MPC. Indeed, HMPC only slightly violates temperature constraint (by 1.9 K), whereas MPC significantly violated the constraint (by 10.1 K). The re-modulation of $I_{\text{app}}(t)$ has a consequence also on the behavior of both SOC and $V_{\text{out}}(t)$. In particular, the SOC driven by HMPC decreases its slope (at around 58%) and reaches the target SOC with 1700 s of delay with respect to MPC. On the other side, due to the higher current applied, the MPC voltage profile shows a steeper increase (followed by an overshoot), whereas HMPC reaches the final voltage of 4.08 V in a more gentle way. This aspect makes the HMPC charging protocol to be less stressful for the Li-ion cell by inducing a lower potential drop between the two electrodes [Torchio et al., 2016b], [Ramadass et al., 2004].

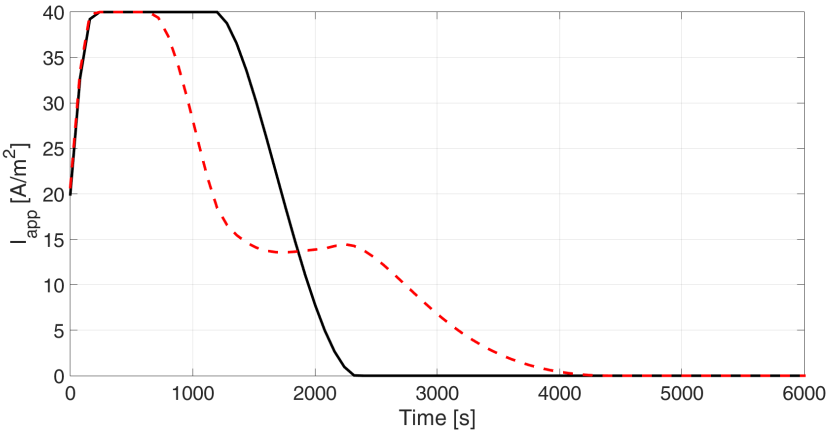


Figure 4.23: *Applied current densities comparison*: MPC based on the ARX model (black line) is compared to HMPC equipped with the PWARX (dashed red line).

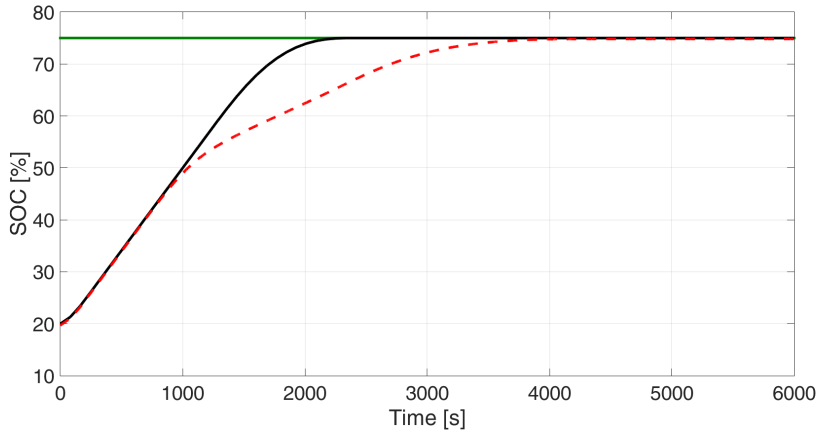


Figure 4.24: *SOC comparison*: MPC based on the ARX model (black line) is compared to HMPC equipped with the PWARX (dashed red line).

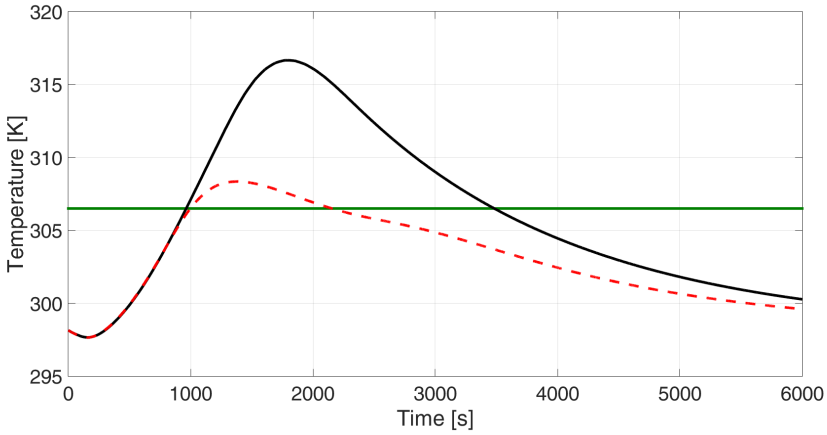


Figure 4.25: *Temperature profiles comparison*: MPC based on the ARX model (black line) is compared to HMPC equipped with the PWARX (dashed red line).

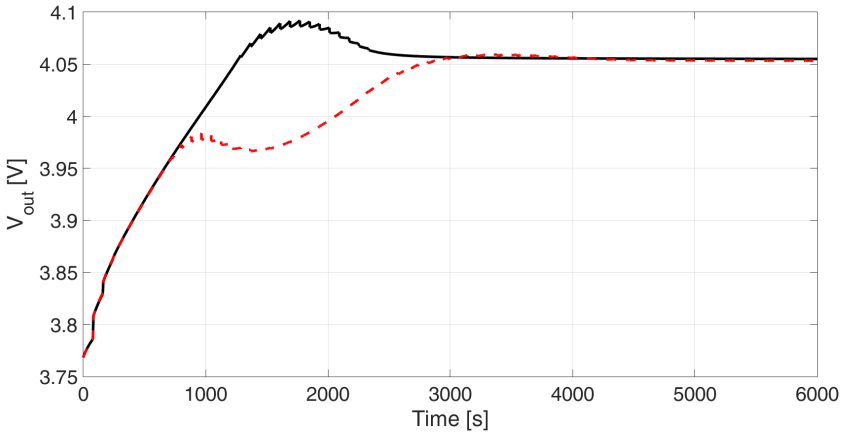


Figure 4.26: *Voltage profiles comparison*: MPC based on the ARX model (black line) is compared to HMPC equipped with the PWARX (dashed red line).

4.4.4 Summary

This section proposes a HMPC algorithm for the development of a Li-ion cell ABMS. In order to overcome possible limitations related to the on-line application of the HMPC algorithms, a linearized version of the Li-ion cell plant is identified by means of a PWARX model. The identification of such models was carried out according to the tailored algorithm presented in Section 3.4.3 (algorithm 1). Through an analysis of the identification dataset, only the thermal dynamics was modeled using a PWARX representation. Voltage and SOC were found to be suitably represented by means of linear ARX models. For comparison purposes, the overall set of dynamics were also represented by means of an ARX model. The validation of the PWARX and ARX models highlighted the capabilities of piecewise affine dynamics to better approximate the thermal nonlinearities. Closed-loop performance results show the effectiveness of the proposed algorithm, with particular emphasis over the enforcement of thermal constraints. Even though the clustering and identification algorithm are based on empirical rules, the proof of concept provided by this work highlights the capabilities of PWARX models to be embedded into ABMS in order to ensure good closed-loop performance while guaranteeing reduced online computational cost.

4.5 Linear time varying strategies for the optimal charging of Li-ion batteries

4.5.1 Introduction

As a further step towards the development of ever more reliable and efficient ABMSs, this section proposes the comparison of MPC strategies based on linear, piecewise affine, and linear time varying approximations of the P2D model. As previously addressed, it has been shown that linearized models can provide interesting results. Nevertheless, due to the strong nonlinearities driving the thermal dynamics, the discrepancies between the linearized models and such dynamics makes it extremely difficult to enforce temperature constraints. For this reason, piecewise affine approximations of the P2D model have been proposed in the previous section. PWARX models are formulated as a set of affine dynamics, where binary variables are used to represent switches among the different submodels. While this choice represents a compromise between accuracy and complexity, its online application becomes prohibitive as the number of submodels increases. With the aim of further reducing the computational burden, while still providing satisfactory results, LTV approximations of the P2D model are also considered. As outlined in Section 3.2.1, these models are obtained by linearizing the dynamics around a nominal trajectory [Borhan et al., 2012, Barbarisi et al., 2009, Falcone et al., 2007, Falcone et al., 2006]. Such approximation is accurate in the neighborhood of the nominal trajectory, and has the benefit of not relying on binary variables, therefore dramatically reducing the online computational cost.

In all the proposed control scenarios, the objective is to charge the Li-ion

cell to a reference value of the SOC while enforcing temperature, voltage, SOC, and applied input current constraints. The results show a significant improvement in terms of constraint satisfaction when both PWARX or LTV ABMSs are considered.

4.5.2 Simulation setup

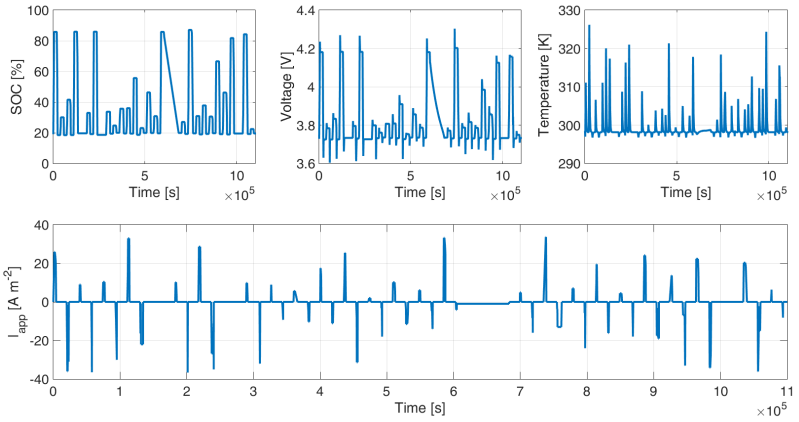


Figure 4.27: Li-ion cell data used for the identification of the PWARX models.

To identify the ARX and PWARX approximations of the Li-ion cell dynamics, a set of charging and discharging input profiles were applied to the P2D model similarly to what shown in Section 4.4. The cell starts from a steady-state condition characterized by $V_{ss} = 3.79$ V, $SOC_{ss} = 20\%$ and $T_{ss} = 298.15$ K, which corresponds to the application of $I_{ss} = 0$ A/m^2. The identification dataset $\mathcal{I}$ (i.e., the values of $I_{\text{app}}(t)$, $V_{\text{out}}(t)$, $T(t)$, and SOC(t)) are reported in Fig. 4.27. A similar dataset, $\mathcal{V}$, was used for validation purposes. ARX and PWARX models were obtained according to the procedures summarized in sections 3.4.2 and 3.4.3 respectively, with $\boldsymbol{N_a}$,

$\boldsymbol{N_b}$, and $\boldsymbol{N_k}$ as in (4.5). Two different PWARX models were identified, with 4 and 8 clusters respectively using the identification algorithm 2 outlined in Section 3.4.3. The performance of the obtained approximations are reported in Table 4.7, where the fitness function (3.36) is computed. Since the ARX models used in the three experiments are the same for SOC(t) and $V_{\text{out}}(t)$, their fitness values remain unchanged. On the other hand, the use of PWARX models improves significantly the approximation of the temperature behavior as observed in Fig. 4.28, which compares the temperature profiles for a subset of the validation profile. In particular, the ARX model sometimes even fails to identify the correct sign of the temperature change. The PWARX models produce much more accurate temperature predictions, even when only a few clusters are considered. Fig. 4.29 reports the switches among the different submodels, for PWARX models with 4 and 8 clusters. The identification of the PWARX models presented above, was carried out considering the collected data to be deviation sequences on top of the equilibrium condition $(\boldsymbol{u_{ss}}, \boldsymbol{y_{ss}})$.

# of clusters	**SOC**	**Voltage**	**Temperature**
0	96%	83%	−3%
4	96%	83%	72%
8	96%	83%	89%

Table 4.7: Comparison of the fitness function (3.36) among different models. Where the number of clusters is 0, a linear ARX model is used.

4.5.3 Results

According to Section 4.5.2, the PWARX model with 8 clusters was the most accurate approximation of the P2D model. On the other hand, due to its hybrid nature, its model form has the highest online computational

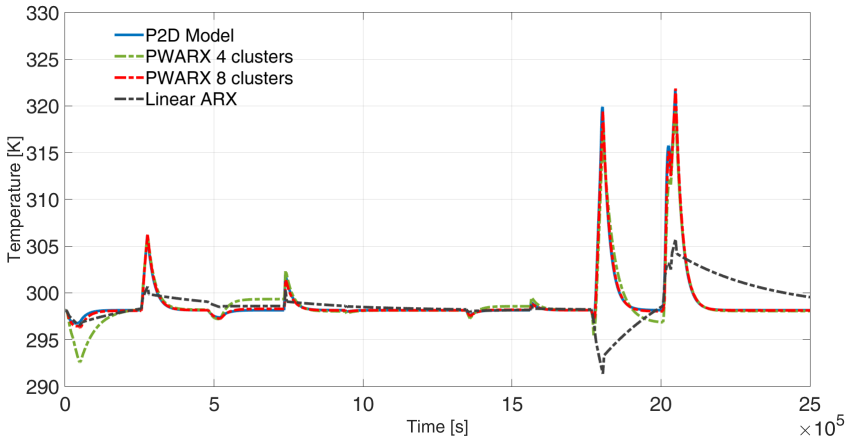


Figure 4.28: The temperature profiles for three different identified models and the P2D dynamics using the $\mathcal{V}$ dataset.

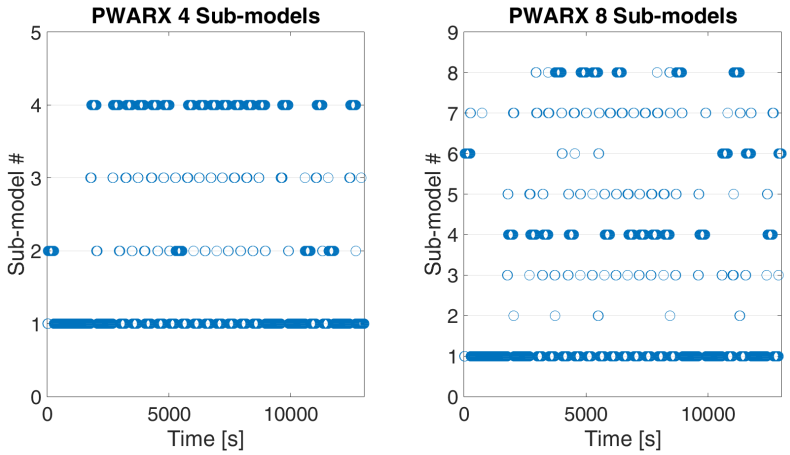


Figure 4.29: Switches among the different submodels for the 4- and 8-cluster PWARX models. These data were obtained during the validation of the hybrid models.

cost when used in an MPC algorithm. Indeed, its MPC formulation would require the solution of an MIQP at each time instant, with an online com-

putational cost of ~ 3600 s[1], which is much higher than the sampling time $T_s = 80$ s). To reduce the online computational cost while still providing satisfactory results, an LTV approximation of the 8-cluster PWARX model was computed [2]. The nominal trajectory used to compute the LTV model was obtained according to problem (3.15) with initial condition $V_{ss} = 3.79$ V, $SOC_{ss} = 20\%$ and $T_{ss} = 298.15$ K (steady-state condition corresponding to $I_{ss} = 0$ A/m^2), $H_{\bar{u}} = 30$ steps, and optimization parameters as in Table 4.8. In this case, the optimization (3.15) was solved without any softening of the output constraints. To optimally charge the Li-ion cell, MPC strategies based on ARX, PWARX, and LTV models were designed using the same parameters as in Table 4.8 and $H_p = H_u = 30$ steps. Due to the mismatch between the P2D dynamics and the models used for control, soft constraints were considered as in (3.30) with $\boldsymbol{P} = 3 \cdot 10^3$. The MPT Toolbox [Herceg et al., 2013] was used to compute the MLD dynamics, and the commercial solver CPLEX [IBM, 2010] to solve the resulting MIQP and QP problems on a i5@ 2.7-GHz 64-bit CPU system with 16 Gbytes of RAM running Windows 10 Pro.

The closed-loop responses are compared in Figures 4.30 to 4.33 for MPC based on ARX (solid blue), PWARX (dashed red), and LTV (dot-dashed yellow) models. MPC with the ARX model attained the desired SOC in the shortest time (1280 s) but had the highest temperature constraint viola-

[1]The commercial solver CPLEX [IBM, 2010] was used to solve the resulting MIQP and QP problems on a i5@ 2.7-GHz 64-bit CPU system with 16 Gbytes of RAM running Windows 10 Pro

[2]Even though a LTV representation of the entire P2D model would be more accurate, the computational burden related to the computation of the time-varying dynamical matrices at each time step, and the need of adopting a state observer would make its online usage very challenging.

Parameter	Value
$\boldsymbol{y}_{\min}$	$[0\%,\ 2.5\,\mathrm{V},\ 290\,\mathrm{K}]^\top$
$\boldsymbol{y}_{\max}$	$[95\%,\ 4.2\,\mathrm{V},\ 303.65\,\mathrm{K}]^\top$
$\boldsymbol{u}_{\min}$	$0\ \mathrm{A/m^2}$
$\boldsymbol{u}_{\max}$	$I_{1.35_C}\ \mathrm{A/m^2}$
$\boldsymbol{\Delta u}_{\min}$	$-10\ \mathrm{A/(m^2\ s)}$
$\boldsymbol{\Delta u}_{\max}$	$10\ \mathrm{A/(m^2\ s)}$
$\boldsymbol{Q}_{SOC}$	10
$\boldsymbol{Q}_V$	0
$\boldsymbol{Q}_T$	0
$\boldsymbol{R}$	1
$\boldsymbol{y}_{\mathrm{ref}}$	$[50,\ 0,\ 0]^\top$
$\boldsymbol{u}_{\mathrm{ref}}$	0
T_s	80 s

Table 4.8: Parameters used for the nominal trajectory optimization and the synthesis of the controllers

tion (305.7 K). Moreover, due to the significant mismatch between the P2D model and its ARX approximation, the closed-loop current profile exhibited undesired fluctuations, mainly between 800 and 1100 seconds. The use of a PWARX model with 4 clusters provided significant improvement both in terms of constraint satisfaction and current profile. This improved performance comes with increased complexity (2.5 s to solve each optimization compared to 0.02 s for the ARX model) but is still fast enough for online application (the sampling time is $T_s = 80$ s). Finally, the use of an LTV model leads to even better performance, with a smoother input profile when

compared to the PWARX model. In fact, the better model approximation leads to an anticipated current drop, which allows the MPC algorithm to (*i*) avoid undesired fluctuations in the input profile, (*ii*) attain the reference SOC value in a shorter time (1840 s vs. 2100 s for the PWARX model), and (*iii*) almost not exceed the temperature constraints (303.9 K vs. 304.3 K for the PWARX model). Since the LTV-based MPC requires the solution of a QP (rather than an MIQP), its online cost (0.022 s) is comparable to the ARX case. A potential limitation of the LTV-based approach is that it relies on the linearization of the 8-cluster PWARX around a nominal trajectory. Given that such trajectory was obtained offline starting from a particular initial condition, the closed-loop performance may be poor when different starting conditions are considered. For this reason, the developed LTV-based MPC was also tested for steady states with $SOC_{ss} = 15\%$ and $SOC_{ss} = 25\%$. According to Figures 4.34 to 4.36, which compare the LTV (red dashed line) with the 4-cluster PWARX (blue solid line), even in the presence of initial condition uncertainties, the LTV-based approach still provides better closed-loop performance.

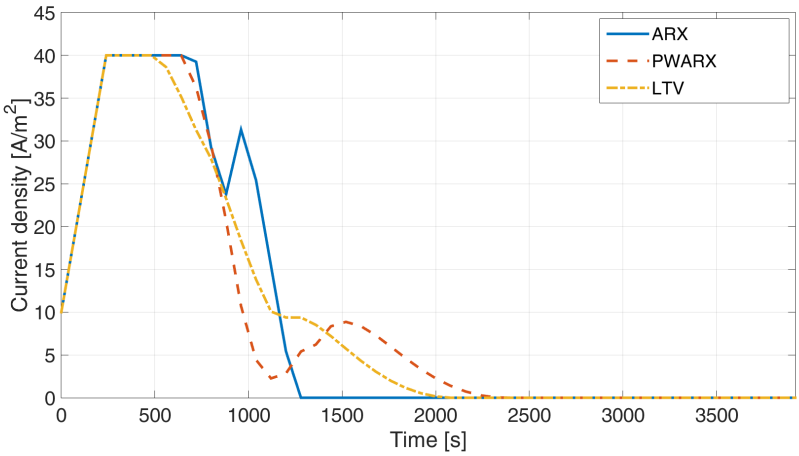


Figure 4.30: $I_{\text{app}}(t)$ profiles for three ABMSs: ARX-based (solid blue line), PWARX-based (dashed-orange line), and LTV-based (dot-dashed yellow line).

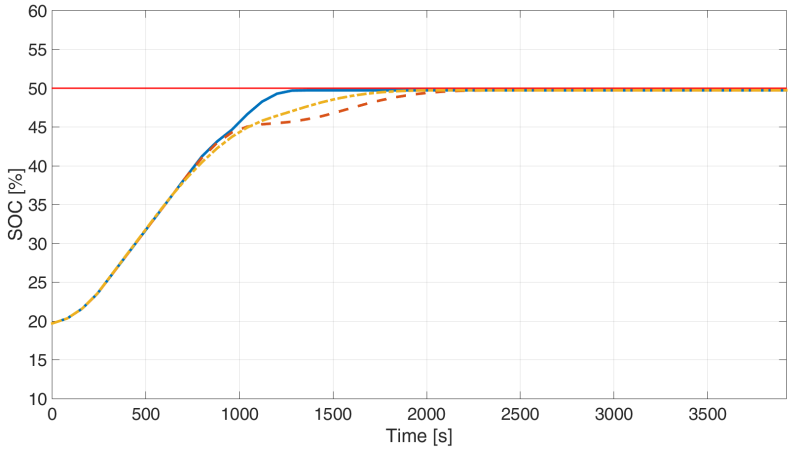


Figure 4.31: SOC(t) profiles for three ABMSs: ARX-based (solid blue line), PWARX-based (dashed-orange line), and LTV-based (dot-dashed yellow line).

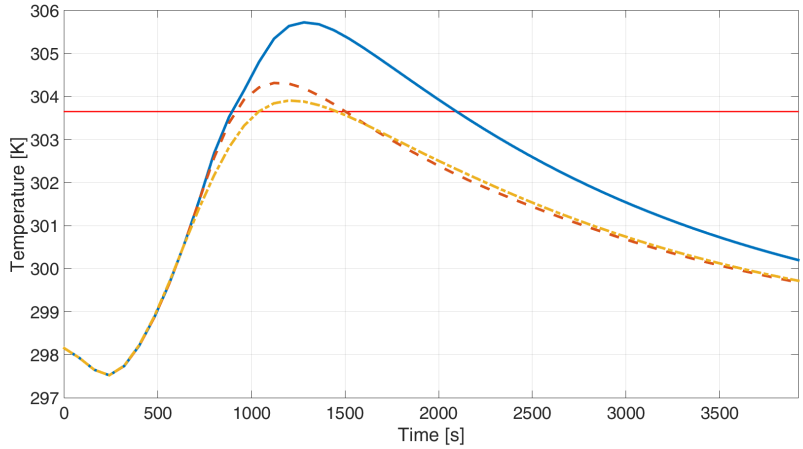


Figure 4.32: $T(t)$ profiles for three ABMSs: ARX-based (solid blue line), PWARX-based (dashed-orange line), and LTV-based (dot-dashed yellow line). The solid black line represents the temperature upper bound.

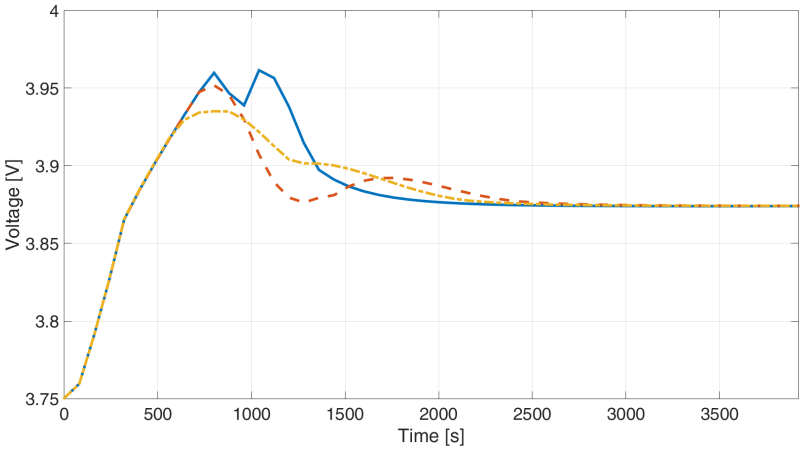
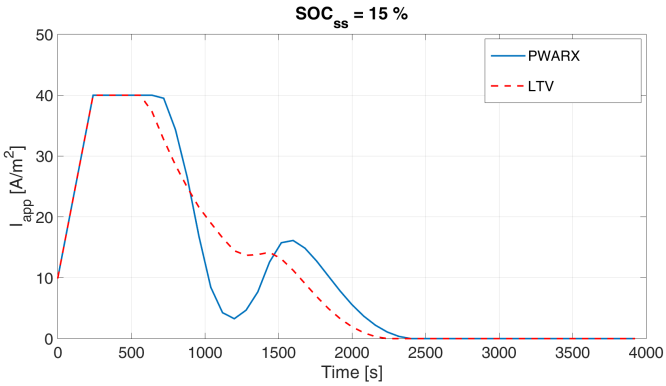
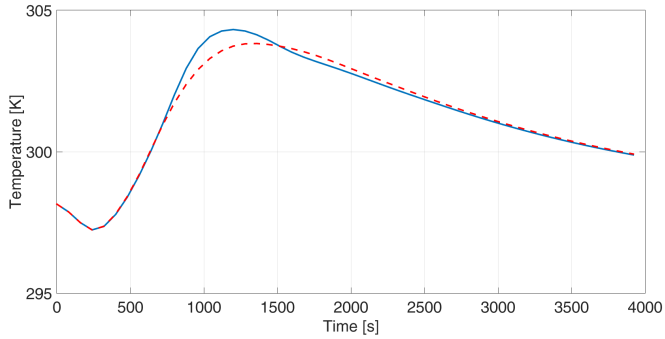


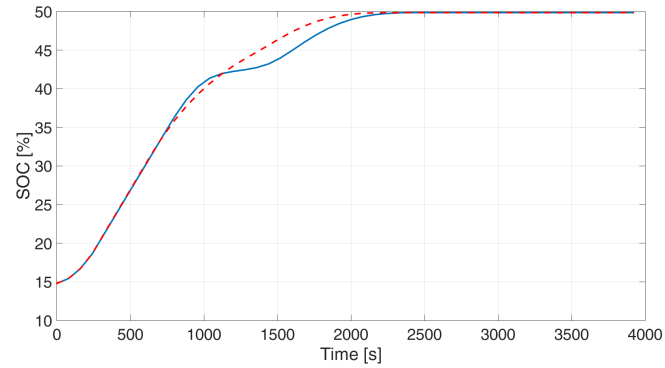
Figure 4.33: $V_{\text{out}}(t)$ profiles for three ABMSs: ARX-based (solid blue line), PWARX-based (dashed-orange line), and LTV-based (dot-dashed yellow line).



(a) Applied current density

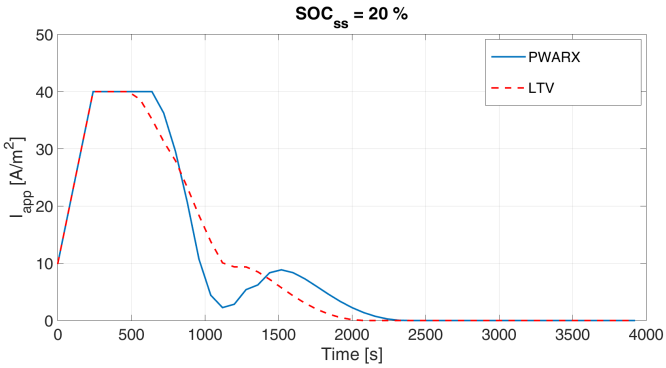


(b) Temperature

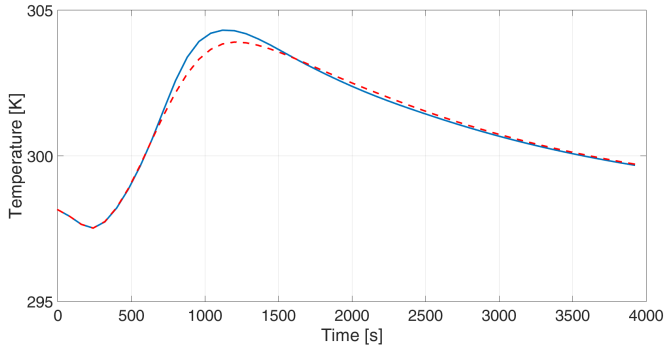


(c) State of Charge

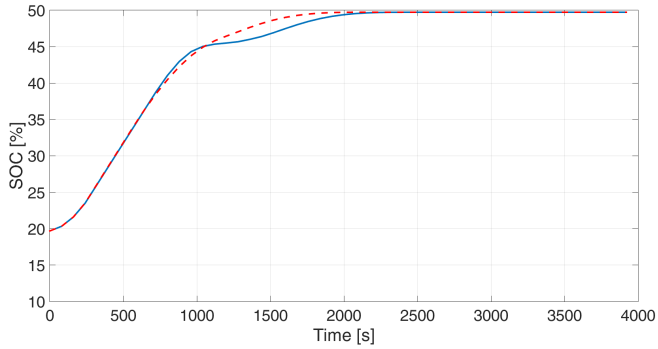
Figure 4.34: Simulations with an initial value of $SOC_{ss} = 15\%$. HMPC enabled ABMS (blue solid line) compared to the LTV enabled ABMS (dashed red line)



(a) Applied current density

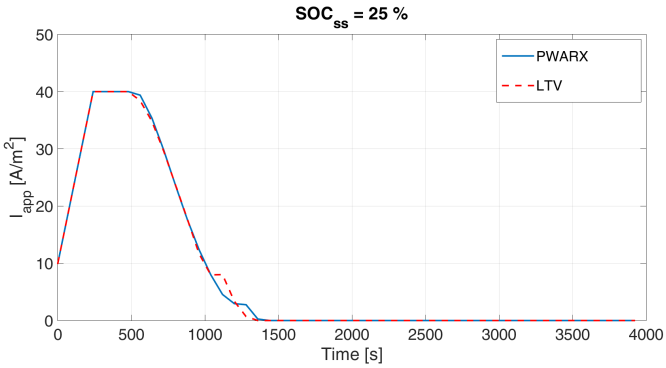


(b) Temperature

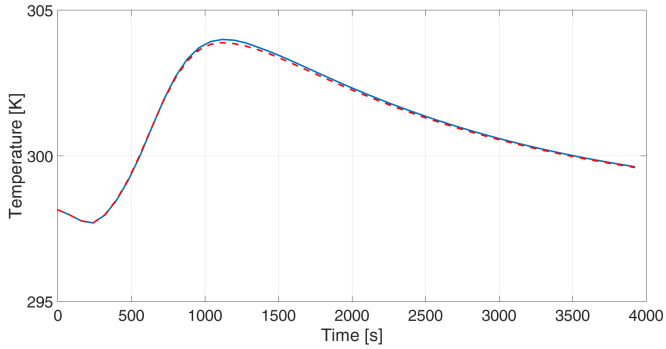


(c) State of Charge

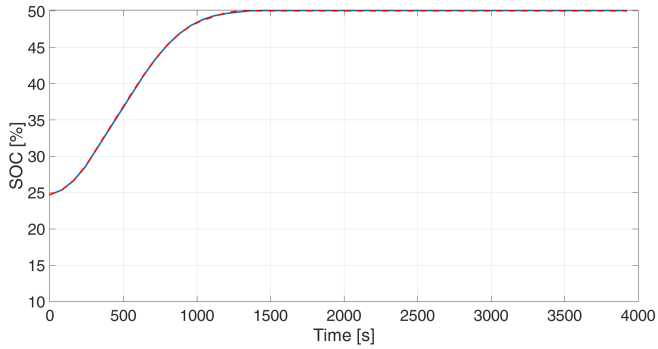
Figure 4.35: Simulations with an initial value of $SOC_{ss} = 20\%$. HMPC enabled ABMS (blue solid line) compared to the LTV enabled ABMS (dashed red line)



(a) Applied current density



(b) Temperature



(c) State of Charge

Figure 4.36: Simulations with an initial value of $SOC_{ss} = 25\%$. HMPC enabled ABMS (blue solid line) compared to the LTV enabled ABMS (dashed red line)

4.5.4 Summary

This investigation considers the optimal charging of a Li-ion cell using a variety of MPC strategies that were validated using the well-known P2D model based on porous electrode theory. The complexity of the P2D model, which is a set of highly nonlinear and tightly coupled PDAEs, makes its direct usage within a MPC framework impractical. For this reason, the P2D dynamics were represented by means of a linear ARX model. Although the ARX model is able to provide interesting results, the presence of strong nonlinearities for the thermal behavior called for the adoption of more sophisticated approximations. As a step towards the improvement of the closed-loop performance, PWARX models were proposed based on the identification algorithm 2 outlined in Section 3.4.3. A comparison among the prediction accuracy of the ARX, 4-cluster, and 8-cluster PWARX models highlighted the capabilities of the piecewise affine dynamics to better approximate the P2D nonlinearities and lead to improved closed-loop performance. A drawback of PWARX models is their use of binary variables, which lead to the formulation of MIQPs that need to be solved online by the MPC algorithm. Since the complexity of such problems grows exponentially with the number of clusters, their real-time application can become expensive. To reduce online computational cost, the 8-cluster PWARX model was approximated with an LTV representation obtained around a set of nominal trajectories. With respect to the ARX model, the MPC strategies based on the 4-cluster PWARX and the LTV approximations provided better closed-loop performance. To the absence of binary variables, the LTV-based MPC algorithm is formulated as a QP instead of an MIQP and so is the least expensive for online application. Even in the

presence of perturbations of the initial states, the LTV strategy provided good closed-loop performance.

Acronyms

ABMS Advanced BMS

AE Algebraic Equation

ARX AutoRegressive eXogenous

BMS Battery Management System

CC-CV Constant Current - Constant Voltage

CIC Consistent Initial Condition

CV Control Volume

CC Constant Current

DAE Differential-Algebraic Equation

DP Dual Polarization

EA Electrochemical Accumulator

ECM Equivalent Circuit Model

EM Electrochemical Model

EV Electric Vehicle

FSR Finite Step Response

FTCS Forward-Time Central-Space

FVM Finite Volume Method

HEV Hybrid Electric Vehicle

HM Harmonic Mean

HMPC Hybrid MPC

LIONSIMBA Lithium-ION SIMulation BAttery

LS Least Squares

LTI Linear Time Invariant

LTV Linear Time Varying

MCC Multistage Constant Current

MIMO Multi-Input Multi-Output

MIQP Mixed Integer QP

MLD Mixed Logical Dynamical

MOL Method Of Line

MPC Model Predictive Control

NMPC Nonlinear MPC

NTI Normalized Time Index

OCP Optimal Control Problem

OCV Open Circuit Voltage

ODE Ordinary Differential Equation

P2D Pseudo Two-Dimensional

PC Pulshe Charging

PDAE Partial Differential and Algebraic Equation

PDE Partial Differential Equation

PNGV Partnership for a New Generation of Vehicles

PWARX PieceWise affine ARX

PWASystem PieceWise affine system

PRBS PseudoRandom Bynary Sequence

QDMC Quadratic Dynamic Matrix Control

QP Quadratic Programming

RH Receding Horizon

RMSE Root-Mean-Square Error

SEI Solid Electrolyte Interface

SIMO Single-Input Multi-Output

SISO Single-Input Single-Output

SOC State Of Charge

SOH State Of Health

SPM Single Particle Model

List of Figures

List of Tables

Bibliography

[Andersen and Andersen, 2016] Andersen, E. D. and Andersen, K. D. (2016). The MOSEK optimization software.

[Andersson et al., 2012] Andersson, J., Åkesson, J., and Diehl, M. (2012). *CasADi: A Symbolic Package for Automatic Differentiation andOptimal Control*, pages 297–307. Springer Berlin Heidelberg, Berlin, Heidelberg.

[Bandhauer et al., 2011] Bandhauer, T. M., Garimella, S., and Fuller, T. F. (2011). A critical review of thermal issues in lithium-ion batteries. *Journal of The Electrochemical Society*, 158(3):R1–R25.

[Barbarisi et al., 2009] Barbarisi, O., Palmieri, G., Scala, S., and Glielmo, L. (2009). LTV-MPC for yaw rate control and side slip control with dynamically constrained differential braking. In *Proceedings of the European Control Conference*, pages 4810–4815.

[Bemporad and Morari, 1999] Bemporad, A. and Morari, M. (1999). Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–428.

[Bentley, 1997] Bentley, W. F. (1997). Cell balancing considerations for lithium-ion battery systems. In *Proceedings of the Twelfth Annual Battery Conference on Applications and Advances*, pages 223–226.

[Bergveld et al., 2002] Bergveld, H., Kruijt, W., and Notten, P. (2002). *Battery Management Systems, Design by Modeling*. Springer.

[Bernardi et al., 1985] Bernardi, D., Pawlikowski, E., and Newman, J. (1985). A general energy balance for battery systems. *Journal of the Electrochemical Society*, 132(1):5–12.

[Bernardi and Go, 2011] Bernardi, D. M. and Go, J.-Y. (2011). Analysis of pulse and relaxation behavior in lithium-ion batteries. *Journal of Power Sources*, 196(1):412–427.

[Besenhard, 2008] Besenhard, J. O. (2008). *Handbook of Battery Materials*. Wiley.

[Borhan et al., 2012] Borhan, H., Vahidi, A., Phillips, A. M., Kuang, M. L., Kolmanovsky, I. V., and Cairano, S. D. (2012). MPC-based energy management of a power-split hybrid electric vehicle. *IEEE Transactions on Control Systems Technology*, 20(3):593–603.

[Boyd and Vandenberghe, 2004] Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press, Cambridge.

[Breakwell, 1959] Breakwell, J. V. (1959). The optimization of trajectories. *Journal of the Society for Industrial and Applied Mathematics*, 7(2):215–247.

[Cai and White, 2011] Cai, L. and White, R. E. (2011). Mathematical modeling of a lithium ion battery with thermal effects in COMSOL Inc.

Multiphysics (MP) software. *Journal of Power Sources*, 196(14):5985–5989.

[Camacho and Alba, 2013] Camacho, E. F. and Alba, C. B. (2013). *Model Predictive Control*. Springer Science & Business Media.

[Camacho and Bordons, 2012] Camacho, E. F. and Bordons, C. (2012). *Model Predictive Control in the Process Industry*. Springer Verlag, London.

[Chai and Patankar, 2000] Chai, J. C. and Patankar, S. V. (2000). Finite-volume method for radiation heat transfer. *Advances in Numerical Heat Transfer*, 2:109–141.

[Chaturvedi et al., 2010] Chaturvedi, N. A., Klein, R., Christensen, J., Ahmed, J., and Kojic, A. (2010). Algorithms for advanced battery-management systems. *IEEE Control Systems*, 30(3):49–68.

[Chen et al., 2008] Chen, H., Armand, M., Demailly, G., Dolhem, F., Poizot, P., and Tarascon, J.-M. (2008). From biomass to a renewable lixc6o6 organic electrode for sustainable li-ion batteries. *ChemSusChem*, 1(4):348–355.

[Chen et al., 2010] Chen, Y.-S., Chang, K.-H., Hu, C.-C., and Cheng, T.-T. (2010). Performance comparisons and resistance modeling for multi-segment electrode designs of power-oriented lithium-ion batteries. *Electrochimica Acta*, 55(22):6433 – 6439.

[Deng et al., 2009] Deng, J., Becerra, V., and Stobart, R. (2009). Input constraints handling in an MPC/feedback linearization scheme. *Interna-*

tional Journal of Applied Mathematics and Computer Science, 19(2):219–232.

[Dhar et al., 1997] Dhar, S. K., Ovshinsky, S. R., Gifford, P. R., Corrigan, D. A., Fetcenko, M. A., and Venkatesan, S. (1997). Nickel/metal hydride technology for consumer and electric vehicle batteries – A review and update. *Journal of Power Sources*, 65(1):1–7.

[Doyle, 1995] Doyle, C. M. (1995). *Design and Simulation of Lithium Rechargeable Batteries*. PhD thesis, University of California, Berkeley.

[Doyle, 1998] Doyle, M. C. (1998). FORTRAN Programs for the Simulation of Electrochemical Systems. http://www.cchem.berkeley.edu/jsngrp/fortran.html.

[Falcone et al., 2006] Falcone, P., Borrelli, F., Asgari, J., Tseng, H. E., and Hrovat, D. (2006). A real-time model predictive control approach for autonomous active steering. In *Proceedings of the IFAC Workshop on Nonlinear Model Predictive Control for Fast Systems*, Grenoble, France.

[Falcone et al., 2007] Falcone, P., Tufo, M., Borrelli, F., Asgari, J., and Tseng, H. E. (2007). A linear time varying model predictive control approach to the integrated vehicle dynamics control problem in autonomous systems. In *Proceedings of the IEEE Conference on Decision and Control*, pages 2980–2985.

[Ferrari-Trecate, 2005] Ferrari-Trecate, G. (2005). Hybrid Identification Toolbox (HIT).

[Ferrari-Trecate et al., 2003] Ferrari-Trecate, G., Muselli, M., Liberati, D., and Morari, M. (2003). A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217.

[Foguth et al., 2015] Foguth, L. C., Paulson, J. A., Braatz, R. D., and Raimondo, D. M. (2015). Fast robust model predictive control of high-dimensional systems. In *Proceedings of the European Control Conference*, pages 2009–2014. IEEE.

[Garcia and Morshedi, 1986] Garcia, C. E. and Morshedi, A. M. (1986). Quadratic programming solution of dynamic matrix control (QDMC). *Chemical Engineering Communications*, 46(1–3):73–87.

[Geuss et al., 2015] Geuss, M., Lohmann, B., Peherstorfer, B., and Willcox, K. (2015). A black-box method for parametric model order reduction. *IFAC-PapersOnLine*, 48(1):168–169.

[Grötschel and Henk, 2003] Grötschel, M. and Henk, M. (2003). The representation of polyhedra by polynomial inequalities. *Discrete & Computetional Geometry*, 29(4):485–504.

[Guyomard and Tarascon, 1994] Guyomard, D. and Tarascon, J.-M. (1994). Rocking-chair or Lithium-ion Rechargeable Lithium Batteries. *Advanced Materials*, 6(5):408–412.

[He et al., 2011] He, H., Xiong, R., and Fan, J. (2011). Evaluation of lithium-ion battery equivalent circuit models for state of charge estimation by an experimental approach. *Energies*, 4(4):582–598.

[Heemels et al., 2001] Heemels, W. P., De Schutter, B., and Bemporad, A. (2001). Equivalence of hybrid dynamical models. *Automatica*, 37(7):1085–1091.

[Herceg et al., 2013] Herceg, M., Kvasnica, M., Jones, C., and Morari, M. (2013). Multi-Parametric Toolbox 3.0. In *Proceedings of the European Control Conference*, pages 502–510.

[Hindmarsh et al., 2005] Hindmarsh, A. C., Brown, P. N., Grant, K. E., Lee, S. L., Serban, R., Shumaker, D. E., and Woodward, C. S. (2005). SUNDIALS: Suite of nonlinear and differential/algebraic equation solvers. *ACM Transactions on Mathematical Software (TOMS)*, 31(3):363–396.

[Hsieh et al., 2001] Hsieh, G. C., Chen, L. R., and Huang, K. S. (2001). Fuzzy-controlled Li-ion battery charge system with active state-of-charge controller. *IEEE Transactions on Industrial Electronics*, 48(3):585–594.

[Hu et al., 2012] Hu, X., Li, S., and Peng, H. (2012). A comparative study of equivalent circuit models for Li-ion batteries. *Journal of Power Sources*, 198:359–367.

[Huang et al., 2009] Huang, J. W., Liu, Y. H., Wang, S. C., and Yang, Z. Z. (2009). Fuzzy-control-based five-step Li-ion battery charger. In *Proceedings of the International Conference on Power Electronics and Drive Systems*, pages 1547–1551.

[Hunt et al., 1998] Hunt, K. J., Munih, M., Donaldson, N. d. N., and Barr, F. M. (1998). Investigation of the hammerstein hypothesis in the model-

ing of electrically stimulated muscle. *IEEE Transactions on Biomedical Engineering*, 45(8):998–1009.

[IBM, 2010] IBM (2010). ILOG CPLEX Optimizer.

[Jain, 2010] Jain, A. K. (2010). Data clustering: 50 years beyond k-means. 31(8):651–666.

[Jenny et al., 2005] Jenny, P., Lee, S. H., and Tchelepi, H. A. (2005). Adaptive multiscale finite-volume method for multiphase flow and transport in porous media. *Multiscale Modeling & Simulation*, 3(1):50–64.

[Johnson et al., 2001] Johnson, V. H., Pesaran, A. A., Sack, T., and America, S. (2001). *Temperature-dependent Battery Models for High-power Lithium-ion Batteries*. National Renewable Energy Laboratory, Golden, Colorado.

[Juloski et al., 2005] Juloski, A. L., Heemels, W., Ferrari-Trecate, G., Vidal, R., Paoletti, S., and Niessen, J. (2005). Comparison of four procedures for the identification of hybrid systems. In Morari, M. and Thiele, L., editors, *Hybrid Systems: Computation and Control*, pages 354–369. Springer Verlag, Berlin Heidelberg.

[Keil and Jossen, 2016] Keil, P. and Jossen, A. (2016). 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*, 6:125–141.

[Kirsch, 2000] Kirsch, D. A. (2000). *The Electric Vehicle and the Burden of History*. Rutgers University Press.

[Klein et al., 2011] Klein, R., Chaturvedi, N. a., Christensen, J., Ahmed, J., Findeisen, R., and Kojic, A. (2011). Optimal charging strategies in lithium-ion battery. In *Proceedings of the American Control Conference*, pages 382–387. IEEE.

[Kokotovic et al., 1976] Kokotovic, P. V., O'Malley, R., and Sannuti, P. (1976). Singular perturbations and order reduction in control theory – An overview. *Automatica*, 12(2):123–132.

[Kuhne et al., 2004] Kuhne, F., Lages, W. F., and da Silva Jr., J. M. G. (2004). Model predictive control of a mobile robot using linearization. In *Proceedings of the International Conference on Mechatronics and Robotics*, pages 525–530, Aachen, Germany.

[Kumaresan et al., 2008] Kumaresan, K., Sikha, G., and White, R. E. (2008). Thermal model for a Li-ion cell. *Journal of the Electrochemical Society*, 155(2):A164–A171.

[Kurzweil, 2010] Kurzweil, P. (2010). Gaston planté and his invention of the lead–acid battery—the genesis of the first practical rechargeable battery. *Journal of Power Sources*, 195(14):4424–4434.

[Kwon and Han, 2006] Kwon, W. H. and Han, S. H. (2006). *Receding Horizon Control: Model Predictive Control for State Models*. Springer-Verlag, London.

[Lassoued and Abderrahim, 2013] Lassoued, Z. and Abderrahim, K. (2013). A Kohonen neural network based method for PWARX identification. In *Adaptation and Learning in Control and Signal Processing*, volume 11, pages 742–747.

[Lee et al., 2016] Lee, J., Kim, H., and Park, M. J. (2016). Long-life, high-rate lithium-organic batteries based on naphthoquinone derivatives. *Chemistry of Materials*, 28(7):2408–2416.

[LeVeque, 2002] LeVeque, R. J. (2002). *Finite Volume Methods for Hyperbolic Problems*, volume 31. Cambridge University Press, Cambridge, UK.

[Linden, 1984] Linden, D. (1984). *Handbook of Batteries and Fuel Cells*. McGraw-Hill Book Company, New York.

[Liu, 2006] Liu, S. (2006). An analytical solution to Li/Li^+ insertion into a porous electrode. *Solid State Ionics*, 177(1):53–58.

[Ljung, 1998] Ljung, L. (1998). System identification. In Prochazka, A., Kingsbury, N., Payner, P. J. W., and Uhlir, J., editors, *Signal Analysis and Prediction*, pages 163–173. Birkhäuser, Basel, Switzerland.

[Luo et al., 2009] Luo, Y. F., Liu, Y. H., and Wang, S. C. (2009). Search for an optimal multistage charging pattern for lithium-ion batteries using the taguchi approach. In *TENCON Conference*, pages 1–5.

[Maciejowski, 2002] Maciejowski, J. M. (2002). *Predictive Control: With Constraints*. Harlow Pearson Education, Piscataway, New Jersey.

[Magni et al., 2009] Magni, L., Raimondo, D. M., and Allgöwer, F., editors (2009). *Nonlinear Model Predictive Control*. Springer, Berlin Heidelberg.

[Matsuo and Hasegawa, 2003] Matsuo, T. and Hasegawa, Y. (2003). *Realization Theory of Discrete-time Dynamical Systems*. Springer Verlag, Berlin Heidelberg.

[Mattingley et al., 2011] Mattingley, J., Wang, Y., and Boyd, S. (2011). Receding horizon control. *IEEE Control Systems*, 31(3):52–65.

[Mayne, 2000] Mayne, D. (2000). Nonlinear model predictive control: Challenges and opportunities. In Magni, L., Raimondo, D. M., and Allgöwer, F., editors, *Nonlinear Model Predictive Control*, pages 23–44. Springer, Berlin Heidelberg.

[Mesbah et al., 2015] Mesbah, A., Paulson, J. A., Lakerveld, R., and Braatz, R. D. (2015). Plant-wide model predictive control for a continuous pharmaceutical process. In *Proceedings of the American Control Conference*, pages 4301–4307.

[Meyer, 1998] Meyer, W. H. (1998). Polymer electrolytes for lithium-ion batteries. *Advanced Materials*, 10(6):439–448.

[Miao et al., 2005] Miao, B., Zane, R., and Maksimovic, D. (2005). Practical on-line identification of power converter dynamic responses. In *Proceedings of the IEEE Applied Power Electronics Conference*, pages 57–62.

[Moon and Lee, 2009] Moon, U.-C. and Lee, K. Y. (2009). Step-response model development for dynamic matrix control of a drum-type boiler–turbine system. *IEEE Transactions on Energy Conversion*, 24(2):423–430.

[Moore and Schneider, 2001] Moore, S. W. and Schneider, P. J. (2001). A review of cell equalization methods for lithium ion and lithium polymer battery systems. Technical report, Society of Automative Engineers.

[Morari et al., 2002] Morari, M., Lee, J. H., and García, C. E. (2002). Model Predictive Control.

[Moura et al., 2013] Moura, S. J., Chaturvedi, N. A., and Krstic, M. (2013). Constraint management in Li-ion batteries: A modified reference governor approach. In *Proceedings of the American Control Conference*, pages 5332–5337.

[Moura et al., 2009] Moura, S. J., Forman, J. C., Stein, J. L., and Fathy, H. K. (2009). Control of film growth in lithium ion battery packs via switches. In *Proceedings of the ASME Dynamic Systems and Control Conference*, volume 1, pages 139–147.

[Nakahara, 2006] Nakahara, H. (2006). *Study of Passive Film Formation on Graphite Surface Lithiated in the Polysiloxane Based Electrolyte for the Application to Lithium Secondary Battery*. ProQuest.

[Newman and Thomas-Alyea, 2012] Newman, J. and Thomas-Alyea, K. E. (2012). *Electrochemical Systems*. John Wiley & Sons.

[Nishi, 2001] Nishi, Y. (2001). Lithium ion secondary batteries; past 10 years and the future. *Journal of Power Sources*, 100(1):101–106.

[Northrop et al., 2011] Northrop, P. W. C., Ramadesigan, V., De, S., and Subramanian, V. R. (2011). Coordinate transformation, orthogonal collocation, model reformulation and simulation of electrochemical-thermal behavior of lithium-ion battery stacks. *Journal of The Electrochemical Society*, 158(12):A1461–A1477.

[Ogata, 1995] Ogata, K. (1995). *Discrete-time Control Systems*, volume 2. Prentice Hall Englewood Cliffs, New Jersey.

[Patankar, 1980] Patankar, S. (1980). *Numerical Heat Transfer and Fluid Flow*. CRC Press, Boca Raton, Florida.

[Perez et al., 2016] Perez, H., Hu, X., and Moura, S. (2016). Optimal charging of batteries via a single particle model with electrolyte and thermal dynamics. In *Proceedings of the American Control Conference*, pages 4000–4005. American Automatic Control Council (AACC).

[Prett and Gillette, 1980] Prett, D. M. and Gillette, R. (1980). Optimization and constrained multivariable control of a catalytic cracking unit. In *Joint Automatic Control Conference*, number 17, page 73.

[Propoi, 1963] Propoi, A. (1963). Application of linear programming methods for the synthesis of automatic sampled-data systems. *Avtomat. i Telemekh*, 24(7):912–920.

[Qin and Badgwell, 2003] Qin, S. J. and Badgwell, T. A. (2003). A survey of industrial model predictive control technology. *Control Engineering Practice*, 11(7):733–764.

[Rahimian et al., 2011] Rahimian, S. K., Rayman, S., and White, R. E. (2011). Comparison of single particle and equivalent circuit analog models for a lithium-ion cell. *Journal of Power Sources*, 196(20):8450–8462.

[Ramadass et al., 2004] Ramadass, P., Haran, B., Gomadam, P. M., White, R., and Popov, B. N. (2004). Development of first principles capacity fade model forLi-ion cells. *Journal of the Electrochemical Society*, 151(2):A196–A203.

[Ramadass et al., 2003] Ramadass, P., Haran, B., White, R., and Popov, B. N. (2003). Mathematical modeling of the capacity fade of Li-ion cells. *Journal of Power Sources*, 123(2):230–240.

[Ramadesigan et al., 2010] Ramadesigan, V., Boovaragavan, V., Pirkle, J. C., and Subramanian, V. R. (2010). Efficient reformulation of solid-phase diffusion in physics-based lithium-ion battery models. *Journal of The Electrochemical Society*, 157(7):A854–A860.

[Ramadesigan et al., 2012] Ramadesigan, V., Northrop, P. W. C., De, S., Santhanagopalan, S., Braatz, R. D., and Subramanian, V. R. (2012). Modeling and simulation of lithium-ion batteries from a systems engineering perspective. *Journal of The Electrochemical Society*, 159(3):R31–R45.

[Randall, 2016] Randall, T. (2016). Here's how electric cars will cause the next oil crisis. http://www.bloomberg.com/features/2016-ev-oil-crisis/.

[Rao and Newman, 1997] Rao, L. and Newman, J. (1997). Heat-generation rate and general energy balance for insertion battery systems. *Journal of the Electrochemical Society*, 144(8):2697–2704.

[Rashid and Gupta, 2014] Rashid, M. and Gupta, A. (2014). Mathematical model for combined effect of SEI formation and gas evolution in Li-ion batteries. *ECS Electrochemistry Letters*, 3(10):A95–A98.

[Rolison and Nazar, 2011] Rolison, D. R. and Nazar, L. F. (2011). Electrochemical energy storage to power the 21st century. *MRS Bulletin*, 36(07):486–493.

[Ruetschi, 1977] Ruetschi, P. (1977). Review on the lead-acid battery science and technology. *Journal of Power Sources*, 2(1):3–120.

[Samadi and Saif, 2014] Samadi, M. F. and Saif, M. (2014). Nonlinear model predictive control for cell balancing in li-ion battery packs. In *Proceedings of the American Control Conference*, pages 2924–2929. IEEE.

[Sankarasubramanian and Krishnamurthy, 2012] Sankarasubramanian, S. and Krishnamurthy, B. (2012). A capacity fade model for lithium-ion batteries including diffusion and kinetics. *Electrochimica Acta*, 70:248–254.

[Santhanagopalan et al., 2006] Santhanagopalan, S., Guo, Q., Ramadass, P., and White, R. E. (2006). Review of models for predicting the cycling performance of lithium ion batteries. *Journal of Power Sources*, 156(2):620–628.

[Saraswat and Parmar, 2015] Saraswat, P. and Parmar, G. (2015). A comparative study of differential evolution and simulated annealing for order reduction of large scale systems. In *Proceedings of the International Conference on Communication Control and Intelligent Systems*, pages 277–281.

[Sauer, 2009] Sauer, D. (2009). {BATTERIES} — charge–discharge curves. In Garche, J., editor, *Encyclopedia of Electrochemical Power Sources*, pages 443 – 451. Elsevier, Amsterdam.

[Schiesser, 1991] Schiesser, W. E. (1991). *The Numerical Method of Lines*. Academic Press, San Diego.

[Schubert et al., 1994] Schubert, J., Simutis, R., Dors, M., Havlik, I., and Lubbert, A. (1994). Hybrid modeling of yeast production processes –

Combination of a-priori knowledge on different levels of sophistication. *Chemical Engineering Technology*, 17(1):10–20.

[Scrosati, 2011] Scrosati, B. (2011). History of lithium batteries. *Journal of Solid State Electrochemistry*, 15(7):1623–1630.

[Shen et al., 2012] Shen, W., Vo, T. T., and Kapoor, A. (2012). Charging algorithms of lithium-ion batteries: An overview. In *Proceedings of the IEEE Conference on Industrial Electronics and Applications*, pages 1567–1572.

[Simon et al., 2013] Simon, D., Löfberg, J., and Glad, T. (2013). Nonlinear model predictive control using feedback linearization and local inner convex constraint approximations. In *Proceedings of the European Control Conference*, pages 2056–2061.

[Smith and Wang, 2006] Smith, K. and Wang, C.-Y. (2006). Solid-state diffusion limitations on pulse operation of a lithium ion cell for hybrid electric vehicles. *Journal of Power Sources*, 161(1):628–639.

[Srinivasan et al., 2003] Srinivasan, B., Palanki, S., and Bonvin, D. (2003). Dynamic optimization of batch processes: I. Characterization of the nominal solution. *Computers & Chemical Engineering*, 27(1):1–26.

[Stevek and Kozak, 2011] Stevek, J. and Kozak, S. (2011). Matlab toolbox for PWA identification of nonlinear systems. In *Proceedings of the International Conference on Process Control*, pages 111–118.

[Stoer and Bulirsch, 2013] Stoer, J. and Bulirsch, R. (2013). *Introduction to Numerical Analysis*, volume 12. Springer-Verlag, New York.

[Subramanian et al., 2005] Subramanian, V. R., Diwakar, V. D., and Tapriyal, D. (2005). Efficient macro-micro scale coupled modeling of batteries. *Journal of The Electrochemical Society*, 152(10):A2002–A2008.

[Suthar et al., 2013] Suthar, B., Ramadesigan, V., Northrop, P. W. C., Gopaluni, B., Santhanagopalan, S., Braatz, R. D., and Subramanian, V. R. (2013). Optimal control and state estimation of lithium-ion batteries using reformulated models. In *Proceedings of the American Control Conference*, pages 5350–5355.

[Tarascon and Armand, 2001] Tarascon, J.-M. and Armand, M. (2001). Issues and challenges facing rechargeable lithium batteries. *Nature*, 414(6861):359–367.

[ten Thije Boonkkamp and Anthonissen, 2011] ten Thije Boonkkamp, J. H. M. and Anthonissen, M. J. H. (2011). The finite volume-complete flux scheme for advection-diffusion-reaction equations. *Journal of Scientific Computing*, 46(1):47–70.

[Tiller, 2012] Tiller, M. (2012). *Introduction to Physical Modeling with Modelica*. Kluwer Academic Publishers, Boston.

[Torchio et al., 2016a] Torchio, M., Magni, L., Braatz, R. D., and Raimondo, D. M. (2016a). Optimal charging of a Li-ion cell: A hybrid model predictive control approach. In *Proceedings of the IEEE Conference on Decision and Control*, pages 4053–4058.

[Torchio et al., 2016b] Torchio, M., Magni, L., Braatz, R. D., and Raimondo, D. M. (2016b). Optimal health-aware charging protocol for Lithium-ion batteries: A fast model predictive control approach. In *Pro-*

ceedings of the 11th International Symposium on Dynamics and Control of Process Systems, including Biosystems, pages 827–832.

[Torchio et al., 2016c] Torchio, M., Magni, L., Gopaluni, R. B., Braatz, R. D., and Raimondo, D. M. (2016c). LIONSIMBA: A Matlab framework based on a finite volume model suitable for Li-ion battery design, simulation, and control. *Journal of The Electrochemical Society*, 163(7):A1192–A1205.

[Torchio et al., 2015] Torchio, M., Wolff, N. A., Raimondo, D. M., Magni, L., Krewer, U., Gopaluni, R. B., Paulson, J. A., and Braatz, R. D. (2015). Real-time model predictive control for the optimal charging of a lithium-ion battery. In *Proceedings of the American Control Conference*, pages 4536–4541.

[Tsang and Chan, 2009] Tsang, K. and Chan, W. (2009). A simple and low-cost charger for lithium-ion batteries. *Journal of Power Sources*, 191(2):633–635.

[Tsang and Chan, 2011] Tsang, K. and Chan, W. (2011). Current sensorless quick charger for lithium-ion batteries. *Energy Conversion and Management*, 52(3):1593–1595.

[Valøen and Reimers, 2005] Valøen, L. O. and Reimers, J. N. (2005). Transport properties of $LiPF_6$-based Li-ion battery electrolytes. *Journal of The Electrochemical Society*, 152(5):A882–A891.

[Van den Bossche et al., 2006] Van den Bossche, P., Vergels, F., Van Mierlo, J., Matheys, J., and Van Autenboer, W. (2006). SUBAT: An

assessment of sustainable battery technology. *Journal of Power Sources*, 162(2):913–919.

[Vidal et al., 2003] Vidal, R., Soatto, S., Ma, Y., and Sastry, S. (2003). An algebraic geometric approach to the identification of a class of linear hybrid systems. In *Proceedings of the IEEE Conference on Decision and Control*, volume 1, pages 167–172.

[Wang et al., 1998] Wang, C. Y., Gu, W. B., and Liaw, B. Y. (1998). Micro-macroscopic coupled modeling of batteries and fuel cells: I. Model development. *Journal of the Electrochemical Society*, 145(10):3407–3417.

[Williams, 2013] Williams, H. P. (2013). *Model Building in Mathematical Programming*. John Wiley & Sons, New York.

[Xavier and Trimboli, 2015] Xavier, M. A. and Trimboli, M. S. (2015). Lithium-ion battery cell-level control using constrained model predictive control and equivalent circuit models. *Journal of Power Sources*, 285:374–384.

[Yan et al., 2011] Yan, J., Xu, G., Qian, H., Xu, Y., and Song, Z. (2011). Model predictive control-based fast charging for vehicular batteries. *Energies*, 4(8):1178–1196.

[Yao, 2016] Yao, M. (2016). Rechargaeble organic batteries using naphthazarin derivatives: The effect of chloro-substituents on the battery performance. In *PRiME 2016/230th ECS Meeting (October 2-7, 2016)*. Ecs.

[Yazuzturk and Spitler, 1999] Yazuzturk, C. and Spitler, J. D. (1999). A short time step response factor model for vertical ground loop heat exchangers. *ASHRAE Transactions*, 105:475–485.

[Zhang and White, 2007] Zhang, Q. and White, R. E. (2007). Comparison of approximate solution methods for the solid phase diffusion equation in a porous electrode model. *Journal of Power Sources*, 165(2):880–886.

[Zhang and White, 2008] Zhang, Q. and White, R. E. (2008). Capacity fade analysis of a lithium ion cell. *Journal of Power Sources*, 179(2):793–798.

[Zhang, 2011] Zhang, W. (2011). A review of the electrochemical performance of alloy anodes for lithium-ion batteries. *Journal of Power Sources*, 196(1):13–24.