State and Parametric Estimation of Li-Ion Batteries in Electrified Vehicles

ANAND NARAYAN
State and Parametric Estimation of Li-Ion Batteries in Electrified Vehicles

ANAND NARAYAN

Master of Science thesis in Electrical Machines and Drives at the School of Electrical Engineering KTH Royal Institute of Technology Stockholm, Sweden, September 2017.

Supervisor at NEVS: Dr. Faisal Altaf
Examiner at KTH: Assoc. Prof. Oskar Wallmark

TRITA-EE 2017:156
State and Parametric Estimation of Li-Ion Batteries in Electrified Vehicles
ANAND NARAYAN

© ANAND NARAYAN, 2017.

Department of Electric Power and Energy Systems
Kungliga Tekniska högskolan
SE–100 44 Stockholm
Sweden
Abstract

The increasing demand for electric vehicles (EVs) has led to technological advancements in the field of battery technology. State of charge (SOC) estimation is a vital function of the battery management system - the heart of EVs, and Kalman filtering is a common method for SOC estimation. Due to the non uniformities in tuning and testing scenarios, quantifying performance of SOC estimation algorithms is difficult. Gathering data for different operational scenarios is also cumbersome. In this thesis, SOC estimation algorithms are developed and tested for a variety of scenarios like varying sensor noise and bias properties, varying state and parameter initializations as well as different initial cell temperatures. A validated and open-source simulation plant model is used to enable easy gathering of data for different operational scenarios.

The simulation results show that unscented Kalman filter performs better than extended Kalman filter in presence of hard nonlinearities and high initial uncertainties. However, both filters gave similar performance under nominal conditions implying that the choice of estimation algorithms must depend on operational scenarios. Observability analysis also gave valuable information to aid in selection of algorithms. The simulation plant model facilitated easy data collection for initial development of algorithms, which were then tested successfully using a real dataset. Further testing using real datasets is required to enhance validation.

Keywords: Battery management system, Kalman filter, li-ion cells, observability analysis, state and parametric estimation, sensor bias, state of charge.
Sammanfattning

Den ökande efterfrågan på elfordon har lett till teknologiska framsteg inom området batteriteknik. Estimering av batteriets laddningstillstånd är en essentiell funktion i batteristyrsystemet, hjärtat i ett elfordon, och görs ofta genom att tillämpa metoden Kalmanfiltring. På grund av varierande implementations och testmetodik i litteraturen är det svårt att kvantifiera estimeringsalgoritmer. I denna avhandling utvecklas algoritmer för att estimera ett batteris laddningstillstånd. Algoritmerna testas för olika former av sensorfel och initialtillstånd, samt för en rad olika temperaturer. En validerad datormodell av batteri, sensorer och omgivning nyttjas för att generera representativa data för de olika förhållanden.

Simuleringsresultat visar att den så kallade doftlösa varianten av Kalmanfiltret (UKF) presterade bättre än det utvidgade Kalmanfiltret (EKF) i fall där systembeteendet är mycket olinjärt och då initialtillståndet är osäkert. Under normala förhållanden presterar dock de båda algoritmerna likvärdigt, vilket antyder att valet av algoritm bör göras med avseende användningsscenario. En observerbarhetsanalys av de olika filtervarianterna gav ytterligare värdefull information för valet av algoritm. Efter utveckling av filteringsalgoritmerna i simuleringsmiljö utfördes tester på faktiska mätdata med goda resultat. För att fullständig validering av algoritmerna krävs emellertid mer uttömmande tester.

Nyckelord: Batteristyrsystemet, Kalmanfiltret, li-jonceller, laddningstillstånd, stat och parametrisk estimering, sensorfel, observerbarhetsanalys.
Acknowledgements

First of all, I would like to thank my supervisor Dr. Faisal Altaf; for giving me the opportunity to do this thesis, for all the patient teachings, guidance and support, and for introducing me to the fascinating world of battery technology. I would like to thank Dr. Hessam Mahdianfar for all the interesting discussions and feedbacks during my thesis. I would also like to thank the entire Advanced Battery Technology team at National Electric Vehicles Sweden (NEVS) for proving an amazing work atmosphere to carry out my thesis successfully.

I would like to thank my examiner at KTH, Assoc. Prof. Oskar Wallmark; for all the valuable suggestions, willingness to help promptly, for all the insightful feedback on the thesis and for enabling smooth collaboration between KTH and NEVS.

I would like to thank my family; for enabling my education, for believing in me, and for the endless love and affection. Finally, I would like to thank my friends for the constant support and encouragement during the thesis and throughout my masters.

Anand Narayan
Stockholm, Sweden
September 2017.
Contents

Abstract iii

Sammanfattning v

Acknowledgements vii

Contents ix

1 Introduction 1
   1.1 Motivation .................................................. 1
   1.2 Related Work .............................................. 2
      1.2.1 Battery Modeling .................................... 2
      1.2.2 Estimation Algorithms .............................. 3
   1.3 Objectives ................................................ 4
   1.4 Organization of Thesis ................................. 5

2 System Modeling and Analysis 7
   2.1 Overview of Li-ion battery ............................ 7
      2.1.1 State of Charge ...................................... 8
      2.1.2 Open-circuit Voltage ............................... 9
      2.1.3 Cell Polarization ................................... 9
   2.2 Electrical Equivalent Circuit Model .................. 10
      2.2.1 Continuous Time Model ............................. 11
      2.2.2 Discrete Time Model ............................... 12
   2.3 Sensor Bias Modeling ................................... 13
      2.3.1 Current Sensor Bias ............................... 14
      2.3.2 Voltage Sensor Bias ............................... 14
   2.4 Observability Analysis ................................ 15

3 Algorithms and Implementation 19
   3.1 Bayesian Inference ..................................... 19
   3.2 Linear Kalman Filter ................................. 22
## Contents

3.3 Extended Kalman Filter ................................................. 25
3.4 Unscented Kalman Filter ............................................. 27
3.5 State and Parametric Estimation ..................................... 29
  3.5.1 Joint Estimation .................................................. 30
  3.5.2 Dual Estimation .................................................. 30
3.6 Initialization and Tuning .............................................. 32

4 Results and Discussions .................................................. 37
  4.1 The Setup ............................................................ 37
    4.1.1 Simulation Setup ............................................... 37
    4.1.2 Performance Indices .......................................... 38
  4.2 Simulation Results .................................................. 39
    4.2.1 State EKF vs Dual EKF ....................................... 39
    4.2.2 Sensor Bias .................................................... 43
    4.2.3 Dual EKF vs Dual UKF ....................................... 46
  4.3 Experimental Results ............................................... 49

5 Conclusion ......................................................................... 53

A Simulation Plant Model of LFP cell ..................................... 57
  A.1 Model Description ..................................................... 57
    A.1.1 Electrical Subsystem ........................................... 58
    A.1.2 Thermal Subsystem ............................................. 59
  A.2 Modifications to the Simulink Plant Model ......................... 60
    A.2.1 Addition of Sensors ............................................. 62
    A.2.2 Current Input .................................................... 64

B Kalman Filtering Algorithms ............................................. 67

References ........................................................................... 73
Chapter 1

Introduction

1.1 Motivation

The focus on electric vehicle (EV) technology is increasing rapidly owing to its reduced emissions when compared to internal combustion (IC) engine vehicles. EV technology is considered to be a prominent solution to reduce global air pollution. In addition to decreased emissions, they also have the advantage of being energy efficient and less noisy. An EV is any road vehicle with an electric propulsion. It also includes intermediate solutions like plug-in hybrids (PHEV) and hybrid electric vehicles (HEV) which use a combination of both fuel and electricity [1]. Similar to the fuel-tank in IC engine vehicles, electricity needs to be stored in electrified vehicles. Li-ion batteries, featured by high power density, high energy density, long life and eco-friendliness are used widely in the automotive industry [2].

EV batteries consist of numerous single cells connected in either series or parallel so as to cater to the large energy and power demands. Even though these cells have the same specifications, there exist certain differences in their characteristics due to the manufacturing conditions. This deviation in their characteristics tends to increase with usage causing the cells to behave more differently. The cells also have strict operational restrictions based on voltage, temperature and current [2]. In addition to this, the harsh operating conditions in EVs necessitates for a system to protect, monitor and control the batteries. Such a system is called the Battery Management System (BMS). Among several key functions of the BMS, one particular function, namely State of Charge (SOC) estimation is investigated in this thesis.

SOC estimation is equivalent to the fuel gauge of an IC engine vehicle but unlike the fuel level, SOC cannot be measured directly as it depends on the concentration of lithium ions at the electrodes [3]. Moreover, due to the differences among the cells, finding SOC for a whole battery pack can be challenging. This motivates the use of algorithms capable of
estimating SOC accurately and reliably using other measurable quantities. Challenges for reliable SOC estimation include noisy sensor measurements, variations in temperature and battery parameters, battery aging as well as the overall complex and nonlinear behavior of batteries. Model-based estimation techniques, specifically Kalman filtering is used in this thesis to estimate SOC. The objective of this thesis is to develop an algorithm to estimate SOC accurately and precisely, and to test its performance under a wide-range of operating conditions.

1.2 Related Work

State estimation in batteries is a relatively new topic, but has been well explored in the past decade owing to the emergence of PHEVs and HEVs [4]. In addition to determining the driving range of the vehicle, SOC estimation aids in energy and power calculations. It is also used to determine efficient control strategies to ensure safe utilization and prolonged life of the battery. Hence, automotive applications demand accurate and precise SOC estimation in real-time as the vehicle is running [5].

There are three primary methods of SOC estimation [6]. Current-based methods use the ‘Coulomb counting’ equation relating the current drawn from (or supplied to) the battery and its capacity to estimate SOC. It is an accurate method of estimating SOC. However, the initial SOC must be known so as to find the SOC at any future time instant and this may not always be possible. Furthermore, when Coulomb counting is done over a long duration, the accumulated current measurement errors may cause inaccuracies in estimation. Voltage-based methods use the relationship between open-circuit voltage (OCV) and SOC. The drawback here is that the battery has to be sufficiently rested for the terminal voltage to reach OCV. Moreover, the relationship tends to change with aging. Model-based estimation uses mathematical models to relate measured signals like terminal voltage to SOC and is known to give accurate and precise estimates. It uses components of both current and voltage based estimation methods [7, 8]. Model-based estimation is used in this thesis and has two distinct sub-problems, namely the mathematical model of the battery and the estimation algorithm.

1.2.1 Battery Modeling

For SOC estimation using model-based algorithms, an accurate battery model is of utmost importance. A model that accurately captures the important battery dynamics but is also easy to implement with high computational efficiency is required. Battery modeling can be generally classified into three types. Electrochemical models use partial differential equations to describe the electrochemical processes that occur within the battery. They represent the battery phenomenon accurately but are complex with significant
1.2. Related Work

memory and computation requirements. *Black-box models* use learning algorithms to find a model of the battery solely based on experimental training data without the underlying process knowledge. These models are normally identified and validated in a certain desired frequency range of operation. Therefore, the training dataset must be logged carefully through proper excitation of the system in the desired bandwidth [9, 10]. *Equivalent circuit models* (ECM) approximate the electrochemical process of the cell using electrical components like resistors and capacitors. ECMS are widely used in model-based estimation due to their balance between accuracy and simplicity in implementation [11].

Gregory Plett in [12, 13] proposes a combined ECM consisting of Shepherd, Unnewher and Nernst models. Other models like zero-state hysteresis, one-state hysteresis and enhanced self-correcting (ESC) models are also presented. The accuracy of these models can be significantly improved by adding resistor-capacitor (RC) branches. However, addition of such branches may lead to increase in complexity and parameterization efforts [11]. In [14], a comparative study of twelve state-of-the-art lumped ECMSs is presented along with an analysis of the model structure and parameter identification. Among the different models considered, second-order RC model is shown to be the preferred model for SOC estimation especially in case of high dynamic loads. This model is used for SOC estimation in [15–17] as well. Considering hysteresis effects may result in a slight increase in the model accuracy but at the cost of increased computational complexity [18]. It was also found that parameterizing the ECMSs to fit the model-based estimation framework require elaborate laboratory experiments along with system identification [19]. Also, since these parameters vary based on several factors like temperature, SOC, current and aging; frequent re-parameterization maybe required.

1.2.2 Estimation Algorithms

A comprehensive review of different SOC estimation methods is provided in [9]. Since SOC cannot be directly measured, the estimation algorithm has to use available measurements to reconstruct the internal states of the system. This can be done using state observers. Extended Kalman filter (EKF) is a popular choice for estimation in nonlinear systems [20]. Gregory Plett, in a series of three highly influential papers [4, 12, 21], extensively investigates the EKF algorithm coupled with an ESC battery model. In [13, 22], Plett revisits the same SOC estimation problem using Sigma-point Kalman filters (SPKF) are found to perform better than EKF. The limitations of EKF are also explained. Plett’s experiments considers only a benign Urban Dynamometer Driving Schedule (UDDS) drive cycle with constant temperatures whereas EV batteries are generally subjected to much harsher conditions. In both the paper sets, state estimation considering time-varying parameters using dual or joint estimation gave better results as they can track both state and parameter variations. This is further explored in [23].
In [7] four state observers, namely; Luenberger observer (LO), Sliding-mode observer (SMO), EKF and SPKF are compared in terms of estimation performance and computational complexity. While SPKF performed best, it was computationally demanding. EKF and SMO both showed the best balance of code properties and performance. However, in the scenarios with disturbance, EKF and SPKF performed relatively better. [8] compares Unscented Kalman filter (UKF) a type of SPKF, EKF and $H_{\infty}$ filter for LFP and NMC cell chemistries. EKF performed well for LFP whereas UKF for NMC, showing that the choice of observer algorithms depends on the cell chemistries. The tests also included a few disturbance scenarios but voltage sensor bias and cell temperature dependency were not analyzed. In [15], a set of new performance indices were proposed to compare SOC estimation algorithms while [11] compares 18 different Kalman filtering implementations using the indices. The tests were performed using a synthetic load cycle developed by analyzing different drive cycles. This required elaborate experiments under controlled environments. Temperature range from $-10\,^\circ C$ to $40\,^\circ C$ was also considered with fixed set of tuning parameters. From the results, dual EKF was found to perform the best. Disturbances, especially in sensors were not considered in these tests.

SOC estimation using Particle filters (PF) is investigated in [5, 24]. Results show that PF has similar performance to that of EKF and UKF but with a much higher computational cost. Both these papers consider the LFP cell chemistry which can make voltage based SOC estimation challenging due to the flat OCV-SOC curve. Due to its longer life, higher power density and inherent safety, LFP cells are widely used in EVs [8] and hence, is interesting to investigate.

### 1.3 Objectives

From the above discussion, it can be argued that Kalman filtering with second-order RC equivalent circuit model is the preferred choice for accurate and reliable SOC estimation. However, there is a difficulty in quantifying the performance of different Kalman filters due to the non uniformities in filter tuning and testing scenarios. It can also be seen that acquiring the data required to develop and test algorithms for a variety of operating scenarios is a cumbersome process. Additionally, the LFP cell chemistry, due to its flat OCV-SOC curve, offers some challenges in SOC estimation. This thesis aims to address these issues.

The objective of this thesis is to develop a SOC estimation algorithm for the LFP cell and to test its performance under different operational scenarios, such as

- Varying sensor noise properties
1.4 Organization of Thesis

• Varying sensor bias properties
• Different states and parameter initializations
• Different initial cell temperatures ($-10^\circ C$ to $40^\circ C$)

These scenarios are chosen based on what a typical BMS would experience in an EV. The primary challenge is to acquire the system data corresponding to the operational scenarios which would require numerous reparations of elaborate and time-consuming lab experiments.

1.4 Organization of Thesis

This thesis is organized as follows. Chapter 2 presents different aspects involved in electrical equivalent circuit battery modeling along with the various definitions and assumptions used. The continuous as well as discrete state-space models of the battery are also presented. This is followed by the modeling of sensor bias and observability analysis of the developed models. In Chapter 3, after a brief review of Bayesian inference framework, the various Kalman filtering algorithms used in this thesis are presented. The implementation aspects of these algorithms specific to this thesis are also discussed. Chapter 4 presents the simulation results for the different scenarios and algorithms considered along with the respective discussion and inferences. Here, test results using a real dataset is also presented. Finally, Chapter 5 contains the conclusion and some suggestions for future work.
Chapter 1. Introduction
Chapter 2

System Modeling and Analysis

In this chapter, the modeling aspects of the battery as well as sensor bias are discussed. First, the electrical equivalent circuit model of the battery required for Kalman filtering is presented. Various definitions and assumptions used for the battery modeling are also described along with the state-space model of the battery. This is followed by a discussion on sensor bias modeling. Finally, the system’s observability is analyzed so as to get an idea on choice of estimation algorithm.

2.1 Overview of Li-ion battery

Battery is an electro-chemical device that stores chemical energy and transforms it into electrical energy when delivering it to an external load. A battery usually consists of a group of individual cells packed together in a certain configuration. In this thesis, single-cell battery is considered and hence, the terms ‘battery’ and ‘cell’ are used interchangeably. Li-ion battery cells generally consists of four parts: a positive electrode (or cathode) of a lithium metal oxide material, e.g. LiFePO$_4$, a negative electrode (or anode) of carbon doped with lithium, e.g. Li$_x$C, an electrolytic material in which the electrodes are dipped into and a permeable separator between the electrodes to prevent electric short circuit. The potential difference between the electrodes, i.e. cell’s terminal voltage, drives an electric load when connected across the cell terminals [10].

During discharge, the negative electrode undergoes oxidation releasing Lithium ions and electrons. The ions travel though the electrolyte and diffuses into the positive electrode. The electrons produced at the negative electrode cannot pass though the electrolyte and hence, flow through the external circuit causing a current flow in the reverse direction. During charging, this process is reserved i.e. oxidation occurs at the positive electrode and reduction at the negative electrode. The ampere-hour capacity of the cell depends on the total number of Lithium particles that can reside in the interstitial spaces of the electrodes. The electrical equivalent circuit battery modeling considered in this thesis
requires the following definitions. In the following section, various aspects required for battery modeling are explained.

### 2.1.1 State of Charge

One of the most important aspects of a cell is its state of charge (SOC), denoted by \( z \). Battery is a device used for electrical energy storage and SOC quantifies the amount of energy stored in a cell. The formal definitions of SOC requires the following definitions [12] which are illustrated in Figure 2.1.

**Total capacity** \( (Q) \) is defined as the maximum number of ampere-hours (or Coulombs) that can be drawn from the fully charged state of a cell to its fully discharged state, at room temperature. The cell capacity may vary based on short-term effects (like c-rate or temperature effects) as well as long-term effects (like aging). However, these effects are not considered in this study and therefore, capacity is assumed to be constant.

**Residual capacity** \( (Q_{res}) \) is the number of ampere-hours (or Coulombs) that can be drawn from a cell in its current charge state to its fully discharged state, at room temperature. It can also be defined as the remaining dischargeable capacity of a cell at room temperature. For a fully charged cell, \( Q = Q_{res} \).

**State of charge** \( (z) \) is defined as the ratio of a cell’s residual capacity to it’s total capacity. SOC can either be expressed as \( z \in [0, 100\%] \) or \( z \in [0, 1] \). 100\% or 1 corresponds to a fully charged cell whereas SOC of 0\% or 0 corresponds to an empty cell.

*Fig. 2.1* Figure to illustrate various definitions regarding a battery.
2.1. Overview of Li-ion Battery

2.1.2 Open-circuit Voltage

Open-circuit voltage or OCV is the no-load voltage across battery terminals when battery has attained internal equilibrium. OCV normally increases monotonically as a nonlinear function of SOC. OCV may also depend on cell temperature, but this effect is negligible compared to the effect of SOC [25, 26] and, hence, is neglected in this thesis.

For a given cell, the OCV curve/map as a function of SOC can be found in the form of a look-up table, using laboratory experiments as shown in [25]. Note that cells may follow different OCV-SOC curves during charging and discharging i.e., hysteresis, but the model here considers only the averaged OCV curve. Additionally, the long-term aging of OCV is also neglected.

Different cell chemistries exhibit different OCV-SOC relationships, which may have a significant impact on the SOC estimation accuracy. For the LiFePO$_4$ (or LFP) cell, which is considered in this thesis, the curve is quite flat from 20% to 90% SOC as shown in Figure 2.2. Due to this flatness, a small variation in OCV will correspond to a large variation in SOC which implies that any small noise or disturbance in the voltage measurement will have a large impact on SOC estimation. Hence, for an LFP cell, an accurate and precise voltage measurement is critical.

![OCV-SOC relationship for LiFePO$_4$ cell.](image)

2.1.3 Cell Polarization

Polarization is the deviation of cell terminal voltage from its OCV value. It mainly consists of static and dynamic parts.

Static Polarization

Static Polarization occurs due to ohmic losses inside electrode, conductors, and terminals of cells. This loss is modeled using a series resistance and is also referred to as *internal*...
Dynamic Polarization

Dynamic polarization is due to the kinetic limitation of various cell processes including mass transfer effects through diffusion (of charge transport) and electrode reaction (or charge transfer reaction). It mainly consists of two parts, namely activation polarization and concentration polarization. When the load of the battery is removed, the terminal voltage slowly increases towards the OCV value. This is due to the relaxation of these polarization losses.

Activation Polarization - It refers to the reaction at the interface between electrode and electrolyte where charge particles are exchanged. It is further composed of two phenomena. The charge transfer reaction is the process of ion transfer between electrode and electrolyte. It is represented by the charge transfer resistance. When ions and electrons get accumulated at the electrode-electrolyte interface, it brings in an effect known as double-layer capacitance [27].

Concentration Polarization - The diffusion of ions between the electrodes, though the electrolyte, is caused by the concentration gradient that exists in the electrolyte. The limitation of this charge-transport kinetics leads to concentration polarization. It is also referred to as diffusion polarization.

2.2 Electrical Equivalent Circuit Model

Electrical equivalent circuit models (EECM) use lumped electrical components to represent the static and dynamic behaviors of the cell and is found to be a good candidate for the Kalman estimation [9]. This model captures the important battery dynamics well but is also easy to implement with high computational efficiency. Furthermore, among the different EECMs, second-order RC (2 RC) or dual-polarization model is considered as it offers a good balance between complexity and accuracy [28, 29]. This model is shown in Figure 2.3. OCV is represented by \( V_{OC}(z) \) i.e. a SOC dependent voltage source.

From the definition of SOC, since \( Q \) remains constant, it is evident that SOC changes only with \( Q_{res} \) which, in turn changes with the amount of current passed through the cell i.e. \( \dot{Q}_{res} = I(t) \). Using this, the SOC dynamics is given by

\[
\dot{z}(t) = -\frac{\eta}{Q} I(t) \quad \Rightarrow \quad z(t) = z(0) - \frac{\eta}{Q} \int_0^t I(t) dt, \tag{2.1}
\]
where, $z(0)$ is the initial SOC, $\eta$ is the Coulombic efficiency representing energy loss during charging. Normally, $\eta$ is unity during discharge and close to unity during charge [12]. In this study, it is considered to be unity in both cases. The instantaneous current $I(t)$ is assumed to be positive during discharge and negative during charge. This equation is called ampere-hour counting or Coulomb-counting and is a very accurate method for SOC calculation. However, this method is prone to errors due to the highly uncertain initial SOC value and current measurement errors like noise and bias [30]. For the cell considered in this thesis, the capacity is $2.3 \text{Ah}$ (Refer Appendix A.1.1).

In Figure 2.3, $R_s$ is the internal resistance and is shown in series with the voltage source. Internal resistance is highly dependent on SOC and tends to increase at extreme values of SOC i.e. $z > 90\%$ and $z < 10\%$. It remains fairly constant in the middle range of SOC. The two RC branches model the dynamics of dynamic polarization effect. $R_1$ represents the charge transfer resistance and $C_1$ represents the double-layer capacitance. The concentration polarization effects are modeling by $R_2$ and $C_2$. When compared to activation polarization, concentration polarization has slower dynamics. Therefore, the time constant of $R_1C_1$ branch is lower than that of $R_2C_2$ i.e. $\tau_1 < \tau_2$ [29, 30]. The resistances are in ohms ($\Omega$) and capacitances in Farads ($F$).

### 2.2.1 Continuous Time Model

Applying Kirchoff’s current law to the electrical circuit in Figure 2.3 gives the dynamics of the voltage across RC branches.

$$
\dot{V}_1(t) = -\frac{1}{R_1C_1} V_1(t) + \frac{1}{C_1} I(t) 
$$

(2.2)

$$
\dot{V}_2(t) = -\frac{1}{R_2C_2} V_2(t) + \frac{1}{C_2} I(t).
$$

(2.3)
Now, by applying Kirchoff’s voltage law to the same circuit, the terminal voltage $V_T$ can be calculated as

$$V_T(t) = V_{OC}(z(t)) - V_1(t) - V_2(t) - R_s I(t). \quad (2.4)$$

The voltages are measured in volts ($V$). The term $R_s I(t)$ represents the voltage drop across the internal resistance. It is pertinent to mention that the EECM is only a lumped electrical approximation of the complex electrochemical processes inside a battery, but it still has sufficient accuracy for Kalman filtering applications [29, 30]. The continuous time state-space model of the 2 RC battery model is given by:

$$
\dot{V}_1(t) = -\frac{1}{R_1C_1}V_1(t) + \frac{1}{C_1}I(t) \quad (2.5)
$$

$$
\dot{V}_2(t) = -\frac{1}{R_2C_2}V_2(t) + \frac{1}{C_2}I(t) \quad (2.6)
$$

$$
\dot{z}(t) = -\frac{\eta}{Q}I(t) \quad (2.7)
$$

$$
V_T(t) = V_{OC}(z(t)) - V_1(t) - V_2(t) - R_s I(t). \quad (2.8)
$$

Equations (2.5) to (2.7) are the state equations and Equations (2.8) is the output equation. This system has linear state equations and a nonlinear output equation due to the $V_{OC}(z(t))$ term. Linearizing this system using first order Taylor’s series expansion around SOC operating point $z_0$ gives the state-space model of the form:

$$
\dot{x}(t) = Ax(t) + Bu(t) \quad (2.9)
$$

$$
y(t) = Cx(t) + Du(t).
$$

where

\begin{align*}
\text{States} &= x(t) = [V_1(t) \ V_2(t) \ z(t)]^T \\
\text{Parameters} &= \theta = [R_s \ R_1 \ C_1 \ R_2 \ C_2]^T \\
\text{Input} &= u(t) = I(t) \\
\text{Output} &= y(t) = V_T(t) \\
A &= \text{diag}\left[\begin{array}{ccc}
-\frac{1}{R_1C_1} & -\frac{1}{R_2C_2} & 0 \\
\frac{1}{C_1} & \frac{1}{C_2} - \frac{\eta}{Q}
\end{array}\right]^T \\
B &= \begin{bmatrix}
\frac{1}{C_1} & 1 & -\frac{\eta}{Q}
\end{bmatrix}^T \\
C &= \begin{bmatrix}
-1 & -1 & \frac{dV_{OC}(z_0)}{dz}
\end{bmatrix} \\
D &= [-R_s]
\end{align*}

Note that the parameters ($\theta$) are dependent on SOC as well as temperature and can be adapted online through simultaneous state and parameter estimation scheme (refer Chapter 3). The time constant of the RC branches can be calculated as $\tau_1 = R_1C_1$ and $\tau_2 = R_2C_2$.

### 2.2.2 Discrete Time Model

Kalman filtering algorithms work only with discrete time systems [31]. Therefore, the continuous time battery model has to be discretized. The discrete model can be obtained
by substituting $\dot{x}(t) = (x_{k+1} - x_k) / \Delta t$ in Equations (2.5) to (2.8). This is known as Euler’s discretization. The sampling time is constant at $\Delta t = 1s$ and 3600 is the conversion factor seconds to hours. The discrete time state-space of 2RC battery model is:

\[
V_{1,k+1} = \left(1 - \frac{\Delta t}{R_1C_1}\right) V_{1,k} + \frac{\Delta t}{C_1} I_k \tag{2.10}
\]

\[
V_{2,k+1} = \left(1 - \frac{\Delta t}{R_2C_2}\right) V_{1,k} + \frac{\Delta t}{C_1} I_k \tag{2.11}
\]

\[
z_{k+1} = z_k - \frac{\eta \Delta t}{3600 Q} I_k \tag{2.12}
\]

\[
V_{T,k} = V_{OC}(z_k) - V_{1,k} - V_{2,k} - R_s I_k. \tag{2.13}
\]

### 2.3 Sensor Bias Modeling

The Kalman filtering algorithms use the system model as well as measurements from the actual system to give an optimal estimate of the required states. It predicts the state using the model and then updates the prediction using the measurement. Sensors are used to feed the measurements as well as the input signals to the algorithm. In addition to noise, any real sensor would also have bias in its measurement which adversely affects the estimation algorithm [16]. All sensors have a certain amount of bias in them and the more precise the sensor, the lesser is the bias. Known biases are generally handled by calibration. However, due to aging, calibration errors or external factors, the sensors may also have unknown biases referred to as bias fault. One way of detecting this is by using redundant sensors to measure the same signal. If there is a notable difference in the measurements, then a bias fault can be detected. This maybe an expensive solution since it requires more sensors [16]. Refer Section A.2.1 in appendix for more information of sensor noise and bias.

In this thesis, the bias is estimated along with other states using an augmented state-space model involving the bias. Once estimated, the bias can be eliminated from the corresponding measured signal. Bias is modeled as random-walk where, it is assumed to be essentially constant but is capable of varying slowly over time, driven by some process modeled using a small fictitious noise\(^1\). Both voltage and current sensor biases are considered independently in this thesis. These biases may also vary based on some factors like electromagnetic interference and physical stress. Figure A.6 shows the data flow between the system and the estimation algorithm.

Let $I_b, V_b$ represent the bias in current and voltage sensors respectively, and $I_m, V_m$ represent

\(^1\)This noise shall be considered as a part of the process noise ($Q$) matrix of Kalman filter. It is explained further in Chapter 3]
resent the corresponding measured (inaccurate) signals. Now, the actual measurements $I$ and $V_T$ can be recovered using:

$$I = I_m - I_b \quad \text{where} \quad \dot{I}_b \approx 0, \quad (2.14)$$

$$V_T = V_m - V_b \quad \text{where} \quad \dot{V}_b \approx 0. \quad (2.15)$$

2.3.1 Current Sensor Bias

The continuous time state-space model of the system with current sensor bias can be obtained by substituting Equation (2.14) in Equations (2.5) to (2.8). This augmented model has 4 states. The output is the same terminal voltage $V_T$ but the input is now the measured current $I_m$ i.e. current signal with bias (if any). The model can be discretized using Euler’s discretization as shown in the previous section.

$$\dot{V}_1(t) = -\frac{1}{R_1C_1}V_1(t) + \frac{1}{C_1}(I_m(t) - I_b(t))$$

$$\dot{V}_2(t) = -\frac{1}{R_2C_2}V_2(t) + \frac{1}{C_2}(I_m(t) - I_b(t))$$

$$\dot{z}(t) = -\frac{\eta}{Q}(I_m(t) - I_b(t)) \quad (2.16)$$

$$\dot{I}_b(t) = 0$$

$$V_T(t) = V_{OC}(z(t)) - V_1(t) - V_2(t) + R_s(I_m(t) - I_b(t)).$$

2.3.2 Voltage Sensor Bias

Similar to the case with current sensor bias, the continuous time state-space model of the system with voltage sensor bias can be obtained by substituting Equation (2.15) in Equations (2.5) to (2.8). The resulting augmented model also has 4 states. The input is the current $I$ but the output is now the measured terminal voltage $V_m$.

$$\dot{V}_1(t) = -\frac{1}{R_1C_1}V_1(t) + \frac{1}{C_1}I(t)$$

$$\dot{V}_2(t) = -\frac{1}{R_2C_2}V_2(t) + \frac{1}{C_2}I(t)$$

$$\dot{z}(t) = -\frac{\eta}{Q}I(t) \quad (2.17)$$

$$\dot{V}_b(t) = 0$$

$$V_m(t) = V_{OC}(z(t)) - V_1(t) - V_2(t) + V_b(t) - R_sI(t).$$

The sensors used in a typical BMS are prone to effects like electromagnetic interference, physical stress, ambient temperatures, etc; which may cause a variation in their bias-level. Therefore, it is essential to test the algorithm under varying bias conditions as well. In this
study, the biases are modeled such that they vary slowly over time and hence, any abrupt changes will cause a mismatch between the bias model and the actual bias variation, which in turn may result in poor estimation performance. One way of handling varying bias is by adjusting (tuning) the variance of the small fictions noise which drives the bias to vary. This can be done within the Kalman filtering framework and shall be explained further in Chapter 3.

2.4 Observability Analysis

As mentioned earlier, the Kalman filtering algorithm uses both the model and the measurement to give an estimation of the required state. It is vital to ensure the possibility of estimating the required states using the developed model and the available measurement. Observability analysis provides an insight into this. The state trajectory of an observable system can be uniquely identified for a particular set of inputs provided the corresponding outputs are known. For an unobservable system, it is not possible to estimate states accurately regardless of how well the model matches the system.

Definition - “Consider any two system states \( x^1 \in X \) and \( x^2 \in X \). The output \( y \) can be represented in terms of the states and input \( u \) as \( y(x^1, u, t) \) and \( y(x^1, u, t) \). The two states are distinguishable if there exists an input \( u \) such that \( y(x^1, u, t) \neq y(x^2, u, t) \). The system is locally observable at \( x_1 \) if there exists a neighborhood \( N \) around \( x_1 \) in which the only indistinguishable state from \( x_1 \) is \( x_1 \) itself. The system is globally observable if it is locally observable at every \( x \in X \)” [16]. Observability of a nonlinear system may not be the same as that of its linearized version as nonlinear system [32]. For the linear continuous-time battery model in Equation (2.9), observability can be analyzed from the rank of its observability matrix.

\[
\mathcal{O} = \begin{bmatrix}
C \\
CA \\
CA^2
\end{bmatrix} = \begin{bmatrix}
-1 & -1 & \frac{dV_{OC}(z_0)}{dz} \\
\frac{1}{\tau_1} & \frac{1}{\tau_2} & 0 \\
-\left(\frac{1}{\tau_1^2}\right) & -\left(\frac{1}{\tau_2^2}\right) & 0
\end{bmatrix}.
\] (2.18)

This matrix has full rank iff \( \frac{dV_{OC}(z_0)}{dz} \neq 0 \) which in turn depends on the nonlinearity of the OCV-SOC curve. Observability is also lost when \( \tau_1 = \tau_2 \) but, this is not possible as they represent different chemical phenomena.

Since the system is originally nonlinear, its local observability can be analyzed by the rank test of a matrix involving Lie derivatives. This is done in [16] and it is found that the system is observable iff there exists an \( n \in \mathbb{Z}^+ \) such that

\[
\frac{d^nV_{OC}(z_0)}{dz^n} \neq 0.
\] (2.19)
This means that the nonlinear battery model is locally observable if all order of derivatives of $V_{OC}$ with respect to SOC considered in the algorithm do not become zero simultaneously. For LPF cell, due to the flat OCV curve, the first-order derivative has four zero-crossings as shown in Figure 2.4. Hence, the linearized system considering only the first-order derivatives of Taylor series expansion looses observability at these discrete SOC operating points. Observability of this system can be improved by including higher order derivatives from Taylor series expansion such that, all the derivatives considered do not become zero simultaneously.

![OCV vs SOC - First derivative](image)

**Fig. 2.4** First derivative of OCV-SOC curve.

For the augmented models with sensor bias, the observability matrices with current and voltage sensor bias are shown in Equations (2.20) and (2.21) respectively.

$$O_I = \begin{bmatrix} -1 & -1 & \frac{dV_{OC}(z_0)}{dz} & R_s \\ (1/\tau_1) & (1/\tau_2) & 0 & \frac{dV_{OC}(z_0)}{dz} \cdot (1/Q) + (1/C_1) + (1/C_2) \\ - (1/\tau_1^2) & - (1/\tau_2^2) & 0 & - (1/C_1\tau_1) - (1/C_2\tau_2) \\ (1/\tau_1^3) & (1/\tau_2^3) & 0 & (1/C_1\tau_1^2) + (1/C_2\tau_2^2) \end{bmatrix},$$

$$O_V = \begin{bmatrix} -1 & -1 & \frac{dV_{OC}(z_0)}{dz} & 1 \\ (1/\tau_1) & (1/\tau_2) & 0 & 0 \\ - (1/\tau_1^2) & - (1/\tau_2^2) & 0 & 0 \\ (1/\tau_1^3) & (1/\tau_2^3) & 0 & 0 \end{bmatrix}.$$  (2.20)

For the case current sensor bias ($O_I$), the first-order linearized system is observable except at the zero-crossings in Figure 2.4 i.e. this system has the same observability criterion as that of the system in Equation (2.18). But, in the case of voltage sensor bias, the system looses observability as the last two columns of Equation (2.21) are linearly dependent and
hence, the matrix loses full rank. This implies that an estimation algorithm which considers only first-order terms of Taylor’s series expansion may not give good performance for this case. However, good performance maybe achieved for current sensor bias case as the system is unobservable only at some discrete SOC operating points which tend to pass quickly as the system operates.

Equation (2.22) shows observability matrix of the nonlinear system with voltage sensor bias [16]. The inclusion of higher order terms in the Taylor’s series expansion prevents the matrix from losing rank, provided that the condition in Equation (2.19) is satisfied.

\[
\mathcal{O}_V = \begin{bmatrix}
-1 & -1 & \frac{dV_{OC}(z_0)}{dz} & 1 \\
(1/\tau_1) & (1/\tau_2) & 0 & 0 \\
-(1/\tau_1^2) & -(1/\tau_2^2) & 0 & 0 \\
0 & 0 & 1 & \frac{d^nV_{OC}(z_0)}{dz^n}
\end{bmatrix}
\]  

(2.22)

From the observability analysis, it can be concluded that the first-order linearized system with voltage sensor bias is not observable and this can be improved by including higher order terms from the Taylor’s series expansion. This gives some insight into the choice of suitable estimation algorithm and is further investigated in Chapter 4.

In the above test scenarios, the biases are considered individually. For the system with bias in both sensors, the model will have both \(I_b\) and \(V_b\) i.e. 5 states. The input will be \(I_m\) and the output will be \(V_m\). In this case, it may not be possible to estimate SOC accurately as there is an ambiguity regarding weather an error in the estimated terminal voltage compared with measured terminal voltage is caused by voltage sensor bias or current sensor bias. Other methods are to be analyzed for this scenario and is not considered in this thesis.
Chapter 3

Algorithms and Implementation

In this chapter, the various model based estimation algorithms used in this thesis are presented. The chapter begins with an introduction to Bayesian estimation and how it is used in the model-based estimation approach. This is followed by the discussion on various Kalman filtering algorithms used along with their implementation specific to this study. Finally, an overview of simultaneous state and parametric estimation within the Kalman filtering framework is looked into.

3.1 Bayesian Inference

In general, any causal dynamic system generates outputs based on past and present inputs. The effect of all the past inputs on a system can be summarized by the system’s states. This means that the past inputs and states of a system need not be stored. The evolution of these states and outputs can be described using quasi-static numerical quantities referred to as system’s parameters. In general, the states change rapidly compared to the parameters which may also remain constant [22]. For many applications, it is desirable for the user to find the values of specific states and parameters as the system operates in real-time. This may not be straightforward for some cases where the required states and parameters are hidden i.e. cannot be measured. This depends on the structure of the system over which the user has little control of. Moreover, even the measurable variables, referred to as observations, can be affected by noise and uncertainties introduced by the measuring equipments. This prevents us from knowing the values of these measured variables accurately.

The explanations in this chapter are based on the discrete time state-space model, shown in Equations (3.1) and (3.2).

\[ x_k = f(x_{k-1}, u_{k-1}, w_{k-1}, k - 1) \]  
\[ y_k = g(x_k, u_k, v_k, k) \]
Equation (3.1) is the state equation and it gives the state dynamics. The state and input at time $k$ are given by $x_k$ and $u_k$ respectively. The input $u_k$ is generally known. Equation (3.2) is the output equation with $y_k$ as the output at time $k$. The functions $f(\cdot)$ and $g(\cdot)$ are called the state transition and measurement (or output) functions respectively. They can be linear or nonlinear depending on the system. The process noise $w_{k-1}$ is stochastic and accounts for uncertainties in the input signal as well as inaccuracies in the state equation. $v_k$ is also stochastic and is referred to as the sensor (or measurement) noise. It accounts for errors in the measurement as well as inaccuracies of the output equation. The inaccuracies in state and output equations maybe due to mismatch between actual plant and the model used by the algorithm.

Figure 3.1 shows the evolution of hidden states and measurable observations in Bayesian estimation framework. From Equations (3.1) and (3.2), it can be seen that the conditional probability $p(x_k|x_{k-1})$ for state evolution depends on the state $x_{k-1}$, known input $u_{k-1}$ and the stochastic noise $w_k$. Similarly, the conditional probability for output $p(y_k|x_k)$ depends on the state $x_k$, known input $u_k$ and stochastic noise $v_k$.

**Fig. 3.1** Sequential Bayesinan inference. Horizontal lines denote the evolution of state $x$ across time. Vertical lines denote the state to output transition within the same time interval.

Bayesian inference, also known as probabilistic inference, is a method to estimate these hidden states using the noisy observations. This method, which is very effective in dynamic analysis of sequential data, uses Bayes’ theorem to update the probability of an event as more information become available. This is also known as optimal filtering problem since it aims to compute the best possible estimate of the hidden state $x_k$ given sequence of noisy observations $\mathbb{Y}_k = \{y_1, y_2, \ldots, y_k\}$. The state estimate $\hat{x}_k$, also known as the conditional mean, given observations $\mathbb{Y}_k$ can be found using

$$\hat{x}_k = \mathbb{E}[x_k|\mathbb{Y}_k] = \int_{\mathbb{R}_{x_k}} x_k p(x_k|\mathbb{Y}_k) dx_k.$$  \hspace{1cm} (3.3)

Here, $\mathbb{R}_{x_k}$ is the set comprising of the range of possible states $x_k$. The optimal solution of this equation can be computed recursively in two steps so as to obtain the posterior
conditioned probability density \( p(x_k | \mathbb{Y}_k) \) [33].

**Step-1:** The *a priori* probability of predicting state \( x_k \) given all past observations \( \mathbb{Y}_{k-1} \) can be computed using the Chapman-Kolmogorov equation:

\[
p(x_k | \mathbb{Y}_{k-1}) = \int_{\mathbb{R}^{x_{k-1}}} p(x_k | x_{k-1}) p(x_{k-1} | \mathbb{Y}_{k-1}) \, dx_{k-1}. \tag{3.4}
\]

**Step-2:** Using Bayes’ rule, the *a posteriori* probability to update the state prediction given the present observation \( y_k \) is given by

\[
p(x_k | \mathbb{Y}_k) = \frac{p(x_k | \mathbb{Y}_{k-1}) p(y_k | x_k)}{p(y_k | \mathbb{Y}_{k-1})}. \tag{3.5}
\]

Here, the *a posteriori* probability \( p(x_{k-1} | \mathbb{Y}_{k-1}) \) is acquired from the previous step of the recursion or initialized by the user. The states are assumed to satisfy Markov’s property which implies that the present state \( x_k \) depends only on the previous state \( x_{k-1} \) and is independent of all other past states and observations. Consequentially, the present observation \( y_k \) is conditionally independent of previous observations given the present state \( x_k \). Using this, the required conditional probabilities in Equations (3.4) and (3.5) can be computed as

\[
p(y_k | \mathbb{Y}_{k-1}) = \int_{\mathbb{R}^{x_k}} p(y_k | x_k) p(x_k | \mathbb{Y}_{k-1}) \, dx_k \tag{3.6}
\]

\[
p(x_k | x_{k-1}) = \sum_{[w : x_k = f(x_{k-1}, u_{k-1}, w, k-1)]} p(w) \tag{3.7}
\]

\[
p(y_k | x_k) = \sum_{[v : y_k = g(x_k, u_k, v, k)]} p(v). \tag{3.8}
\]

If all the probability densities are assumed to be Gaussian, the Kalman filter framework offers a simplified solution to these equations. Instead of propagating the entire density function through time, only the mean and covariance of the state are to be evaluated once at each time step. Here, \( w_k \) and \( v_k \) are independent Gaussian noise processes with covariances denoted by \( P_w \) and \( P_v \) respectively where \( P \) is the covariance operator. As shown in [22], the recursions to calculate the *a posteriori* probability density now becomes:

\[
\hat{x}_k^+ = \hat{x}_k^- + L_k(y_k - \hat{y}_k) \tag{3.9}
\]

\[
P^{+}_{\hat{x},k} = P^{-}_{\hat{x},k} - L_k P^-_{\hat{y},k} L_k^*. \tag{3.10}
\]
In the above equations,

\[
\begin{align*}
A \text{ priori state estimate} : \hat{x}^-_k &= \mathbb{E}[x_k|Y_{k-1}] \\
A \text{ posteriori state estimate} : \hat{x}^+_k &= \mathbb{E}[x_k|Y_k] \\
A \text{ priori state estimation error} : \tilde{x}^-_k &= x_k - \hat{x}^-_k \\
A \text{ posteriori state estimation error} : \tilde{x}^+_k &= x_k - \hat{x}^+_k \\
A \text{ priori state estimation error covariance} : P^-_{\tilde{x},k} &= \mathbb{E}[(\tilde{x}^-_k)(\tilde{x}^-_k)^T] \\
\text{Post state estimation error covariance} : P^+_{\tilde{x},k} &= \mathbb{E}[(\tilde{x}^+_k)(\tilde{x}^+_k)^T] \\
\text{Output estimate} : \bar{y}_k &= \mathbb{E}[y_k|Y_{k-1}] \\
\text{Output estimation error (Innovation)} : \tilde{y}_k &= y_k - \bar{y}_k \\
\text{Output estimation error covariance} : P^-_{\tilde{y},k} &= \mathbb{E}[(\tilde{y}_k)(\tilde{y}_k)^T] \\
\text{Cross-covariance between state and output} : P^-_{\tilde{x}\tilde{y},k} &= \mathbb{E}[(\tilde{x}_k)(\tilde{y}_k)^T] \\
\text{Estimator gain matrix} : L_k &= \mathbb{E}[(\tilde{x}_k)(\bar{y}_k)^T]P^{-1}_{\tilde{y},k} = P^-_{\tilde{x}\tilde{y},k}P^{-1}_{\tilde{y},k}.
\end{align*}
\] 

The general algorithm of sequential probabilistic inference is shown in Appendix B Algorithm 2. In the next section, this Bayesian inference algorithm is used to derive the Kalman filtering algorithm.

### 3.2 Linear Kalman Filter

Kalman filter (KF), also known as linear quadratic estimator was first proposed by a Hungarian scientist Rudolf E. Kálmán as a solution to the optimal estimation problem. It follows the sequential probabilistic inference framework and can be used to effectively estimate hidden states using system model and noisy observations. The algorithm has two major stages, namely predict and update. First, the present state is predicted using the prior information of the system. This step is called time update. Next, it corrects (or updates) the prediction using the current measurement (or observation). This step is called the measurement update. In addition to state estimate, the Kalman filter also calculates the error covariance which gives information about uncertainty of the state estimate. Figure 3.2 illustrates the Kalman filter information flow.

It is important to mention that KF algorithm applies only to linear systems. This is because in linear systems, if the stochastic inputs are Gaussian, all the other probability densities remain Gaussian and this is crucial for the assumptions made in the previous section to be valid. The general state-space model of a linear system is of the form

\[
\begin{align*}
x_k &= A_{k-1} x_{k-1} + B_{k-1} u_{k-1} + w_{k-1} \\
y_k &= C_k x_k + D_k u_k + v_k.
\end{align*}
\]
3.2. Linear Kalman Filter

where, $A_k$, $B_k$, $C_k$ and $D_k$ define the system dynamics. Another important assumption made is that both process and measurement noises are considered to be mutually independent white Gaussian noise processes with zero-mean and a known covariances. For simplification purposes, in this thesis, these noises are assumed to be additive i.e. they only affect the state and output in a linear additive manner. The covariances of these noises can be defined as:

$$\mathbb{E}[w_n w_k^T] = \begin{cases} P_w & \text{for } n = k; \\ 0 & \text{for } n \neq k. \end{cases} \quad \mathbb{E}[v_n v_k^T] = \begin{cases} P_v & \text{for } n = k; \\ 0 & \text{for } n \neq k. \end{cases}$$

Using the probabilistic inference framework, the Kalman filtering algorithm can be derived for the linear model shown in Equations (3.22) and (3.23). It consists of 6 steps which are executed in recursion. These steps form basis for all Kalman filtering algorithms.

**Step-1: State estimate time update.** The a priori state estimate at time $k$ can be calculated by applying linearity of expectation operation and noting that $w_{k-1}$ has zero mean.

$$\hat{x}^-_k = \mathbb{E}[A_{k-1} x_{k-1} + B_{k-1} u_{k-1} + w_{k-1} | \mathcal{Y}_{k-1}] = A_{k-1} \hat{x}^+_k + B_{k-1} u_{k-1}. \quad (3.24)$$

**Step-2: State estimation error covariance time update.** The state estimation error is first computed by comparing Equation (3.22) with $\hat{x}^-_k$ from Step-1, as

$$\tilde{x}^-_k = x_k - \hat{x}^-_k = A_{k-1} \hat{x}^+_k + w_{k-1}, \quad (3.25)$$

which can be used to find the state estimation error covariance. Since $w_{k-1}$ is considered to be independent of state $x_{k-1}$, the cross product terms involving $w_{k-1}$ will become zero.

$$P^-_{\tilde{x},k} = \mathbb{E}[(\tilde{x}^-_k)(\tilde{x}^-_k)^T] = \mathbb{E}[(A_{k-1} \hat{x}^+_k + w_{k-1})(A_{k-1} \hat{x}^+_k + w_{k-1})^T] = A_{k-1} P^+_{\hat{x},k-1} A_{k-1}^T + P_w. \quad (3.26)$$

**Step-3: Estimate system output.** The output at time $k$ is estimated using the available a priori information of system and the corresponding state estimate. Linearity of expectation is once again applied along with the assumption that $v_k$ has zero mean.

$$\hat{y}_k = \mathbb{E}[C_k \hat{x}^-_k + D_k u_k + v_k | \mathcal{Y}_{k-1}] = C_k \hat{x}^-_k + D_k u_k. \quad (3.27)$$
Step-4: Calculate estimator gain. To compute the estimator or Kalman gain $L_k$, the values of $P_{\tilde{y}, k}$ and $P_{\tilde{x}, k}$ are first computed by using Equations (3.19) and (3.20) respectively. Using the result of Step-3 and Equation (3.23), the output estimate error, also known as innovation, can be calculated as

$$\tilde{y}_k = y_k - \hat{y}_k = C_k \tilde{x}_k + v_k.$$  

(3.28)

The required covariance values are given by

$$P_{\tilde{y}, k} = E[(C_k \tilde{x}_k + v_k)(C_k \tilde{x}_k + v_k)^T] = C_k P_{\tilde{x}, k} C_k^T + P_v$$  

(3.29)

$$P_{\tilde{x}, k} = E[\tilde{x}_k \tilde{y}_k^T] = E[\tilde{x}_k (C_k \tilde{x}_k + v_k)^T] = P_{\tilde{x}, k} C_k^T.$$  

(3.30)

The Kalman gain can now be computed by substituting these equations in Equation 3.21.

$$L_k = P_{\tilde{x}, k} P_{\tilde{y}, k}^{-1} = P_{\tilde{x}, k} C_k^T (C_k P_{\tilde{x}, k} C_k^T + P_v)^{-1}.$$  

(3.31)

Step-5: State estimate measurement update. The a priori state estimate is updated using values of estimator gain $L_x$, and innovation $\tilde{y}_k$. The a posteriori state estimate can be computed as

$$\hat{x}_k^+ = \hat{x}_k^- + L_k \tilde{y}_k.$$  

(3.32)

Step-6: State error covariance measurement update. The a posteriori error covariance gives a measure of uncertainty in a posteriori state estimate and is found by updating the a priori state error covariance. Substituting Equation (3.21) in Equation (3.10) gives:

$$P_{\tilde{x}, k} = P_{\tilde{x}, k} - L_k P_{\tilde{y}, k} L_k^T$$

$$= P_{\tilde{x}, k} - L_k P_{\tilde{y}, k} (P_{\tilde{y}, k})^{-1} P_{\tilde{x}, k}$$

$$= P_{\tilde{x}, k} - L_k C_k P_{\tilde{x}, k} = (I - L_k C_k) P_{\tilde{x}, k}.$$  

(3.33)

$\hat{x}_k^+$ and $P_{\tilde{x}, k}^+$ are the outputs of the Kalman filter. The estimator on receiving the next observation, proceeds to the next iteration (from Step-1) with these values as input. Appendix B Algorithm 3 summarizes the Kalman filtering algorithm.

During the first iteration, the state and error covariance estimates are to be initialized by the user. Since these values directly enter the prediction step, they are considered to be posteriori i.e. $\hat{x}_0^+$ and $P_{\tilde{x}, 0}^+$. The process and measurement noise covariances should also be defined. They constitute the tuning parameters of the Kalman filter and are generally chosen based on the heuristic knowledge of the system. The filter performance is highly dependent on these tuning parameters and hence, care has to be taken while selecting them [34]. $P_w$ and $P_v$ are also denoted by the $Q$ and $R$ respectively. They help the filter to make an optimal trade-off between trusting the model and the sensor measurement to achieve the best possible estimate. $\hat{x}_0^+$ represents the initial state guess and $P_{\tilde{x}, 0}^+$ tells the algorithm how certain the user is about this initial guess.
3.3 Extended Kalman Filter

The Kalman filtering algorithm works only for linear systems as a closed-form solution to implement general probabilistic inference. One way to handle nonlinear systems is by using a linearization process at every time step. Using this, the general KF algorithm can be extended to handle nonlinear systems as well. This gives rise to Extended Kalman filter (EKF) and is described in this section.

EKF is basically a Kalman filter preceded by a linearizing block which approximates the nonlinear system as a linear time varying system at each time step. The state-space equations of a discrete time nonlinear system is given by:

\[ x_{k+1} = f(x_k, u_k) + w_k \]  \hspace{1cm} (3.34)
\[ y_k = g(x_k, u_k) + v_k. \] \hspace{1cm} (3.35)

This model is similar to the one described by Equations (3.1) and (3.2) but with additive noise, which follows the same assumptions made in the previous section. EKF uses first-order Taylor series expansion to linearize the system at each time step. For this, it is essential to assume that the functions \( f(\cdot) \) and \( g(\cdot) \) are differentiable at all operating points \((x_k, u_k)\). Using first-order Taylor series expansion on \( f(x_k, u_k) \) and \( g(x_k, u_k) \), we get:

\[ f(x_k, u_k) \approx f(\hat{x}_k, u_k) + \left. \frac{\partial f}{\partial x_k} \right|_{x_k = \hat{x}_k} (x_k - \hat{x}_k) \] \hspace{1cm} (3.36)
\[ g(x_k, u_k) \approx g(\hat{x}_k, u_k) + \left. \frac{\partial g}{\partial x_k} \right|_{x_k = \hat{x}_k} (x_k - \hat{x}_k). \] \hspace{1cm} (3.37)

Note that the Jacobian function of \( f(\cdot) \) with respect to \( u_k \) becomes zero as linearization is done around current input \( u_k \) and hence, control deviation is zero. Comparing Equations (3.34) and (3.35) with Equations (3.36) and (3.37) respectively, gives the linearized equations of the system as a function of state \( x_k \), known input \( u_k \) and \( \hat{x}_k \), and stochastic inputs \( w_k \) and \( v_k \).

\[ x_{k+1} \approx A_k x_k + f(\hat{x}_k, u_k) - A_k \hat{x}_k + w_k \] \hspace{1cm} (3.38)
\[ y_k \approx C_k x_k + g(\hat{x}_k, u_k) - C_k \hat{x}_k + v_k. \] \hspace{1cm} (3.39)

These equations are similar to the linear state-space model described by Equations (3.22) and (3.23). \((\hat{x}_k, u_k)\) define the system operating points. The EKF algorithm is summarized in Appendix B Algorithm 4. In EKF framework, \( A_k \) and \( C_k \) are referred to as Jacobian
and they define the dynamics of the linearized system.

Even though EKF is the most popular and commonly used nonlinear Kalman filter, it suffers from a number of flaws due to certain assumptions made in its formulation. Firstly, to find the state and output estimate (Steps 10 and 13 of Algorithm 4), EKF assumes $E[f(x)] \approx f(E(x))$. This is valid only for linear systems and systems with mild non-linearities. It may not be valid for systems with high nonlinearities. Secondly, the EKF formulation considers only first-order Taylor series approximation. Higher order terms are neglected which may lead to loss of accuracy and poor estimation performance. One solution to this is to use second-order or iterative EKF which includes higher order terms in Taylor series expansion. This would require the calculation of Hessians which tend to become tedious.

For the discrete time battery model shown in Equations (2.10) to (2.13), the Jacobians can be calculated by taking partial derivatives of these equations with respect to the states.

$$A_k = \text{diag} \left[ \left( 1 - \frac{\Delta t}{R_1 C_1} \right) \left( 1 - \frac{\Delta t}{R_2 C_2} \right) 1 \right] \quad C_k = \left[ -1 \quad -1 \quad \frac{\partial V_{OC}(z_k)}{\partial z} \right] \bigg|_{z_k = \hat{z}_k^+}$$

Note that the system is linearized around the posteriori state estimate $\hat{z}_k^+$. The Jacobian $A_k$ remains constant as it is independent of states but $C_k$ has a state-dependent differential term which can be efficiently computed using Euler’s approximation as shown in Algorithm 1. Linear interpolation can be used to find intermediate values between two discrete points.

**Algorithm 1** : To find $\frac{\partial V_{OC}(z_k)}{\partial z}$

1. **Given**: The curve $V_{OC}$ vs $z$ with ‘$n$’ number of discrete points.
2. **for** $k = 1, 2, ... n$ **do**
   ▷ Loop till number of points ‘$n$’
   3. if $k = 0$, $\frac{\partial V_{OC}(z_k)}{\partial z} = \frac{V_{OC}(z_{k+1}) - V_{OC}(z_k)}{z_{k+1} - z_k}$ ▷ Only forward Euler for first point
   4. else if $k = n$, $\frac{\partial V_{OC}(z_k)}{\partial z} = \frac{V_{OC}(z_k) - V_{OC}(z_{k-1})}{z_k - z_{k-1}}$ ▷ Only backward Euler for last point
   5. else $V_{OC,fwd} = \frac{V_{OC}(z_{k+1}) - V_{OC}(z_k)}{z_{k+1} - z_k}$ ▷ Forward Euler
   6. $V_{OC,bwd} = \frac{V_{OC}(z_k) - V_{OC}(z_{k-1})}{z_k - z_{k-1}}$ ▷ Backward Euler
   7. $\frac{\partial V_{OC}(z_k)}{\partial z} = \frac{V_{OC,fwd} + V_{OC,bwd}}{2}$ ▷ Average of both for the intermediate points
8. **end for**
3.4 Unscented Kalman Filter

The Unscented Kalman Filter (UKF) is an alternate method for applying Kalman filter algorithm to nonlinear systems which numerically approximates the mean and covariance. The UKF falls under a category of sample-based filters called Sigma-point Kalman filters (SPKF) and does not suffer the same drawbacks of Taylor series approximation. The set of sigma-points at time step $k$ is chosen deterministically such that their mean and covariance match with those of the posteriori estimate at time step $k-1$. These points are passed through nonlinear system equations to get a set of transformed points whose means and covariance can be used to approximate the required mean and covariance of state estimate [8]. UKF algorithm is shown in Appendix B Algorithm 6 and is explained briefly in this section. It can be seen that UKF algorithm has the same overall structure as that of other Kalman filters.

The discrete nonlinear system in Equations (3.1) and (3.2) with a state vector of size $n$ will have $p + 1 = 2n + 1$ sigma-points which can be computed by:

$$X^+_{k-1} = \left\{ \hat{x}^+_{k-1}, \hat{x}^+_{k-1} + \gamma \sqrt{P_{\hat{x},k-1}^+}, \hat{x}^+_{k-1} - \gamma \sqrt{P_{\hat{x},k-1}^+} \right\}. \quad (3.40)$$

The parameter ‘$\gamma$’ controls the spread of sigma-points around $\hat{x}^+_{k-1}$. The sigma-points $X^+_{k-1}$, when propagated though the state-transition function $f(\cdot)$, yield a set of transformed points whose weighted mean and covariance give the required a priori state estimation mean and error covariance.

$$X^-_{k,i} = f(X^+_{k-1,i}, u_{k-1}) \quad (3.41)$$

$$\hat{x}^-_k = \sum_{i=0}^{p} \alpha^m_i X^-_{k,i} \quad (3.42)$$

$$P^-_{\hat{x},k} = \sum_{i=0}^{p} \alpha^c_i (X^-_{k,i} - \hat{x}^-_k)(X^-_{k,i} - \hat{x}^-_k)^T + P_w. \quad (3.43)$$

It is to be noted that the mean and covariance of $X^-_{k,i}$ is equal to that of the corresponding state vector only for a specific set of parameters given by $\{\gamma, \alpha^m_i \text{ and } \alpha^c_i\}$. The weights $\alpha^m_i$ and $\alpha^c_i$ are real scalars with the necessary but not sufficient conditions

$$\sum_{i=0}^{p} \alpha^m_i = 1 \text{ and } \sum_{i=0}^{p} \alpha^c_i = 1. \quad (3.44)$$

Various implementation of SPKF differ only in the choice of weights and scaling factor. For UKF, these parameters are shown in Appendix B Algorithm 3.1 where $\lambda = a^2(n + \kappa) - n$ is the scaling parameter with $10^{-2} \leq a \leq 1$, $b$ incorporates prior information ($b = 2$ for Gaussian process) and $\kappa \in [-n, n]$ is an arbitrary scalar chosen based on performance.
Table 3.1 Weighing parameters of UKF.

<table>
<thead>
<tr>
<th>UKF</th>
<th>$\gamma$</th>
<th>$\alpha_m^0$</th>
<th>$\alpha_m^1$</th>
<th>$\alpha_c^0$</th>
<th>$\alpha_c^i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\sqrt{n + \lambda}$</td>
<td>$\lambda$</td>
<td>$\frac{\lambda}{n + \lambda}$</td>
<td>$\frac{\lambda}{2(n + \lambda)}$</td>
<td>$\frac{\lambda}{n + \lambda} + (1 - a^2 + b)$</td>
<td>$\frac{\lambda}{2(n + \lambda)}$</td>
</tr>
</tbody>
</table>

$\lambda$ controls the spread of sigma-points around a posteriori state estimate $\hat{x}_{k-1}^+$. Once the a priori mean $\hat{x}_k^-$ and covariance $P_{\hat{x},k}$ are computed, the output estimate and output estimate error covariance can be computed by propagating the corresponding sigma-points through the measurement function $g(\cdot)$.

$$Y_{k,i} = g(X^-_{k,i}, u_k)$$  
$$\hat{y}_k = \sum_{i=0}^p \alpha_m^i Y_{k,i}$$  
$$P_{\hat{y},k} = \sum_{i=0}^p \alpha_c^i (Y_{k,i} - \hat{y}_k)(Y_{k,i} - \hat{y}_k)^T + P_v$$  
$$P_{\hat{x}\hat{y},k} = \sum_{i=0}^p \alpha_c^i (X^-_{k,i} - \hat{x}_k^-)(Y_{k,i} - \hat{y}_k)^T.$$  

Finally, using the Kalman gain $L_k$ and the error covariances, the a priori state estimate and estimation error covariance can be updated. The calculation involved in these steps are the same as that of linear KF as described in Equations (3.21), (3.9) and (3.10). In addition to the four tuning parameters of linear KF, UKF algorithm also has $a$, $b$ and $\kappa$ to decide the estimation performance.

UKF has several advantages over EKF. First, it does not need computation of Jacobians which means that the original function need not necessarily be differentiable. This makes the algorithm versatile i.e. by changing the model, the algorithm can be easily extended to other cases whereas, in case of EKF, the Jacobians need to be recomputed. Note that the mean given by UKF is accurate up to second-order term in Taylor series, but covariance is accurate only up to first-order term. The overall performance is quite close to second-order EKF. However, this does not imply that the UKF is always better than EKF. In particular, UKF may outperform EKF for systems with hard nonlinearities and high initial uncertainties. In the cases of mild nonlinearities, EKF and UKF may give similar performance. This is investigated in this thesis.

The implementation of UKF for the discrete time battery model in Chapter 2 is straightforward as shown in Appendix B Algorithm 6. It can be seen that UKF involves a more vector manipulations when compared to EKF due to the presence of sigma-points. Using
MATLAB and by considering the properties of matrices, UKF can be efficiently implemented. For example,

\[ \hat{x}^-_k = \sum_{i=0}^p \alpha_{i}^m X^-_{k,i} \iff \hat{x}^-_k = [\alpha^m][X^-] \]

where, the sum of products of two vectors can also be represented by their dot product.

### 3.5 State and Parametric Estimation

The Kalman filtering algorithms presented so far can estimate the system states for a given model as shown in Equations (3.34) and (3.35). These equations generally involve several parameters in their computations; some of them maybe intrinsically constant whereas others may vary depending on some system factors [23].

In KF state estimation, the parameters are assumed to remain constant. However, time-varying parameters when ignored may lead to incorrect state estimation results. This necessitates a framework capable of estimating states as well as accounting for parameter variations. The process of estimating the parameters given known states, model and noisy measurements is called *parametric estimation*. Considering these time-varying parameters \( \theta_k \), the discrete state-space model of the system with additive noise is given by:

\[
x_{k+1} = f(x_k, u_k, \theta_k) + w_k \tag{3.49}
\]
\[
y_k = g(x_k, u_k, \theta_k) + v_k \tag{3.50}
\]

The constant parameters of the model are considered as a part of \( f(\cdot) \) and \( g(\cdot) \) and are not in \( \theta_k \). Assuming that the true value of parameters \( \theta_k \) describes the model at time \( k \), the parameter estimate \( \hat{\theta}_k \) is required to converge to \( \theta_k \). The state-space model describing the dynamics of the parameters is given by

\[
\theta_{k+1} = \theta_k + r_k \tag{3.51}
\]
\[
d_k = g(x_k, u_k, \theta_k) + e_k. \tag{3.52}
\]

Similar to the case with sensor bias in Chapter 2, Equation (3.51) represents a random-walk model. The parameters tend to vary slowly over time driven by a small fictitious noise \( r_k \). Random-walk model is used as it is difficult to model the parameter dynamics exactly in many systems. The noises, \( r_k \) and \( e_k \), are assumed to be independent Gaussian white noise with zero mean and \( P_r \) and \( P_e \) as covariances respectively.

Equation (3.52) represents the output equation of true parameter dynamics. Comparing with Equation (3.35), the outputs \( y_k \) and \( d_k \) are generally the same and hence, \( e_k \) is equal
to the sensor noise $v_k$. In general, parameters change slowly when compared to states. With the Kalman filtering framework, the state and parametric estimation can be done in two ways.

### 3.5.1 Joint Estimation

In joint estimation, the state and parameter equations are combined to form an augmented state-space model which is used to estimate both states and parameters simultaneously. This model which has both state and parameter dynamics is given by:

$$
\begin{bmatrix}
    x_{k+1} \\
    \theta_{k+1}
\end{bmatrix} =
\begin{bmatrix}
    f(x_k, u_k, \theta_k) + w_k \\
    \theta_k + r_k
\end{bmatrix}
$$

(3.53)

$$
y_k = g(x_k, u_k, \theta_k) + v_k.
$$

(3.54)

The disadvantage of joint estimation is that it involves large matrix operations due to high dimensionality of the augmented model and hence is computationally expensive. The calculations may also result in ill-conditioned matrices due to the vast difference in time scale between state and parameters [23]. However, joint estimate is relatively simple and straightforward to implement, and exactly follows the steps in Appendix B Algorithm 3 and 4 with larger matrix operations.

### 3.5.2 Dual Estimation

Dual estimation consists of two carefully coupled Kalman filters with information exchange and can estimate both, state and parameters. The state filter uses Equations (3.49) and (3.50) whereas the parameter filter uses Equations (3.51) and (3.52). Dual estimation is computationally less complex compared to joint estimation as the matrices involved maybe numerically better conditioned. This also makes it easy to implement dual estimation in actual hardware. However, since the states and parameters are decoupled, any cross-relation between them is lost which may have an impact on the estimation accuracy. Nevertheless, this thesis employs dual EKF/UKF estimation schemes due to its better computational efficiency.

**Dual Extended Kalman Filter**

Dual extended Kalman Filter (or DEKF) consists of two coupled EKFs with information exchange between them as shown in Figure 3.3. The steps involved in DEKF are shown in Appendix B Algorithm 5. It can be seen that both state and parameter filters follow similar steps. The algorithm is initialized using the best guess of true initial state and parameter i.e. $\hat{x}_0^+$ and $\hat{\theta}_0^+$ respectively. The corresponding covariances are also initialized. DEKF has
3.5. State and Parametric Estimation

four noise covariances, namely $Q_x$, $R_x$, $Q_\theta$ and $R_\theta$.

Fig. 3.3 Information flow DEKF. Dashed lines represent error covariances. Black lines represent information flow within the filters whereas green lines denote information exchange between state and parametric filters.

The Jacobian $C^\theta_k$, unlike other Jacobians, is a total-derivative. This is because $\hat{x}_k$ is a function of $\theta_k$ but the converse is not true. This is evident by looking at the Equations (3.49) and (3.51). $C^\theta_k$ can be computed using the following total-derivative expansion. The three total-derivatives involved are initialized as zero during the first iteration and then computed recursively.

$$C^\theta_k = \left. \frac{dg(\hat{x}^-_k, u_k, \theta)}{d\theta} \right|_{\theta = \hat{\theta}_k}$$  \hspace{1cm} (3.55)

$$\frac{dg(\hat{x}^-_k, u_k, \theta)}{d\theta} = \frac{\partial g(\hat{x}^-_k, u_k, \theta)}{\partial \theta} + \frac{\partial g(\hat{x}^-_k, u_k, \theta) d\hat{x}^-_k}{d\theta}$$ \hspace{1cm} (3.56)

$$\frac{d\hat{x}^-_k}{d\theta} = \frac{\partial f(\hat{x}^+_k-1, u_{k-1}, \theta)}{\partial \theta} + \frac{\partial f(\hat{x}^+_k-1, u_{k-1}, \theta)}{\partial \hat{x}^+_k-1} \frac{d\hat{x}^+_k-1}{d\theta}$$ \hspace{1cm} (3.57)

$$\frac{d\hat{x}^+_k-1}{d\theta} = \frac{d\hat{x}^+_k-1}{d\theta} - L_x \frac{dg(\hat{x}^-_k-1, u_{k-1}, \theta)}{d\theta}.$$ \hspace{1cm} (3.58)

For DEKF application in this thesis, the state filter uses the model in Equations (2.10) to (2.13) while the parameter filter uses Equations (3.51) and (3.52). The current input $I$ and terminal voltage output $V_T$ are same for both filters. Therefore, the measurement equation $d_k$ is the same as that of 2.13. DEKF requires three Jacobians, namely $A_k$, $C^e_k$ and $C^\theta_k$. The computation of the first two is similar to that of state EKF but considering

---

1In the dual estimation algorithms in Appendix B, the covariances $Q_x$, $R_x$, $Q_\theta$ and $R_\theta$ are denoted by $P_w$, $P_v$, $P_r$ and $P_e$ respectively.
the time-varying parameters $\theta_k$.

$$A_k = \text{diag}\left[\left(1 - \frac{\Delta t}{R_{1,k}C_{1,k}}\right) \left(1 - \frac{\Delta t}{R_{2,k}C_{2,k}}\right) 1\right] \quad C^z_k = \left[-1 -1 \frac{\partial V_{OC}(z_k)}{\partial z}\right]_{z_k=z_0}$$

The Jacobian $C^\theta_k$ is given by the total-derivative computation. The matrices involved in the computation are shown below. By making appropriate substitutions, the derivative values can be found. Note that for conciseness, superscripts to denote a priori and posteriori are not shown.

$$\frac{\partial f(x_k, u_k, \theta)}{\partial \theta} = \begin{bmatrix} 0 & \frac{V_{1,k}\Delta t}{R_{1,k}C_{1,k}} - \frac{I_{k}\Delta t}{C_{1,k}} & 0 & \frac{V_{2,k}\Delta t}{R_{2,k}C_{2,k}} & 0 & \frac{V_{2,k}\Delta t}{R_{2,k}C_{2,k}} - \frac{I_{k}\Delta t}{C_{2,k}} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial f(x_k, u_k, \theta)}{\partial x_k} = \text{diag}\left[\left(1 - \frac{\Delta t}{R_{1,k}C_{1,k}}\right) \left(1 - \frac{\Delta t}{R_{2,k}C_{2,k}}\right) 1\right]$$

$$\frac{\partial g(x_k, u_k, \theta)}{\partial \theta} = \begin{bmatrix} -I_k & 0 & 0 & 0 \end{bmatrix}$$

$$\frac{\partial g(x_k, u_k, \theta)}{\partial x_k} = \begin{bmatrix} -1 -1 \frac{\partial V_{OC}(z_k)}{\partial z} \end{bmatrix}$$

For the case with sensor bias, the calculation of Jacobians is similar but using the augmented state-space models with the bias term and hence are not shown.

**Dual Unscented Kalman Filter**

Dual unscented Kalman Filter (or DUKF) consists of two coupled UKFs with information exchange between them. The steps involved in DUKF are shown in Appendix B Algorithm 7 which has a similar structure to that of DEKF. Figure 3.4 shows information flow in DUKF. An advantage of DUKF over DEKF is that it does not require calculation of Jacobians. Also, in case of DUKF, the parameter filter does not require the state filter’s Kalman gain $L_{k-1}^x$ i.e. fewer information exchange between filters.

### 3.6 Initialization and Tuning

As mentioned in Section 3.2, the performance of the Kalman filter is highly dependent on its tuning. In this section, the intuitions used in initializing and tuning the filters are described. These intuitions serve as a starting points for tuning the filters. Table 3.2 summarizes the initializing and tuning for the different filters used in this thesis.

**Initial States and Parameters**

These vectors represent the initial guess of the states $x_0$ and time-varying parameters $\theta_0$. The state vector has three states, namely $V_1, V_2$ and $z$. Assuming that the battery is suffi-
3.6. Initialization and Tuning

Figure 3.4  Information flow in DUKF. Dashed lines represent error covariances. Black lines represent information flow within the filters whereas green lines denote information exchange between state and parametric filters.

ciently rested between successive driving episodes, the dynamic polarization would have reached equilibrium with the capacitors fully discharged. Now, the voltage across the RC branches would have dropped to zero and this could be used as a good initial guess for \( V_1 \) and \( V_2 \). It is not always possible to know the value of initial SOC accurately and hence, the algorithms are tested for a variety of initial SOC guesses.

In case of the parameter vector, since the parameters cannot be measured directly, finding an accurate initial guess \( \theta_0 \) is not easy. System identification techniques are generally used to find the initial guess for the parameter vector [25]. In this thesis, parameters are extracted from the simulation plant model (Appendix A) corresponding to the value of initial SOC and initial cell temperature of \( 25^\circ C \). Refer Chapter 4 Section 4.2 for further details. In case of pure state estimation, parameters are considered to remain static at \( \theta_0 \). Also, since the parameters are dependent on charging and discharging, their average value is considered.

Initial State and Parameter Estimation Error Covariance

The initial state and parameter error covariances, namely \( P_{\tilde{x},0} \), \( P_{\tilde{\theta},0} \), tells the algorithm how certain the user is about this initial guesses in terms of statistical variance. If an initial state is unknown, then the corresponding value in \( P_{\tilde{x},0} \) matrix will be high. High covariance results in faster converge but with larger overshoot. In this thesis, since the cell is assumed to have sufficiently rested, \( V_1 \) and \( V_2 \) can have low covariance values if their initial state guess is zero.
The initial parameter covariance $P_{\theta,0}$ comes into play only in case of dual estimation. Since, the initial values of parameters cannot be known, $P_{\theta,0}$ is selected using trial and error by observing the filter performance. This is applicable for the covariance of SOC as well since its initial value cannot be determined accurately. If the system states are assumed to be independent of each other, $P_{x,k}$ is a diagonal matrix denoting zero cross-covariance. The size of $P_{x}$ is $(n_x \times n_x)$ and that of $P_{\theta}$ is $(n_\theta \times n_\theta)$.

**Process and Measurement Noise Covariance**

The process noise covariance $Q$ and measurement noise covariance $R$ help the filter weigh the model and sensor measurements to find the optimal estimate. In case of dual estimation, there exists two sets of noise covariance matrices, i.e. $Q$, $R$ and $Q_{\theta}$, $R_{\theta}$. In this thesis, $Q$ and $R$ are tuned using trial and error with the objective of improving SOC estimate. However, some knowledge of the system modeling can be used for tuning. As discussed in Chapter 2, Coulomb counting is an accurate method for SOC calculation provided that the initial SOC is known. Also, the voltages $V_1$ and $V_2$ are electrical approximations of chemical phenomena. Therefore, in $Q_x$ matrix, the value corresponding to SOC can be made lower than the ones corresponding to the voltages implying more confidence in SOC. Similarly, if accurate and precise sensors are used, the corresponding value in $R$ can be reduced. The size of $Q_x$ is $(n_x \times n_x)$, $Q_{\theta}$ is $(n_\theta \times n_\theta)$ and the size of $R_x$ is $(n_y \times n_y)$. Since the same terminal voltage measurement is used by both filters, $R_x = R_{\theta}$. 
### Table 3.2 Initialization values and tuning parameters for Kalman filtering algorithms.

- **'-I'** represents algorithm for current sensor bias, **'-V'** represent algorithm for voltage sensor bias, **'-T'** represent varying initial cell temperatures where two sets of tuning parameter are used. Initial SOC is dependent on operating scenario as well.
Chapter 3. Algorithms and Implementation
Chapter 4

Results and Discussions

The simulation results along with discussions are presented in this chapter. Initially, an overview of the simulation setup is presented followed by the performance indices used. Various simulation results obtained in this study are then discussed. Finally, the experimental result using a real dataset is discussed.

4.1 The Setup

4.1.1 Simulation Setup

The overall simulation setup consists of the estimation algorithms (Chapter 3) used with the battery model (Chapter 2). For this study, a MATLAB Simulink plant model of LiFePO$_4$ cell is used to simulate the measurements required for the Kalman algorithms. It is a non-linear model of LFP cell parameterized experimentally and validated by University of Michigan, Ann Arbour [25]. The Simulink implementation of this model is published with open-access [26] and is presented in Appendix A. For coherence, the salient features are presented here as well.

The electro-thermal model for A123 26650 LFP cell has an electrical equivalent circuit model coupled with a two-state thermal model. This simulation model is structurally similar to the model in Chapter 2. But, the parameters here have nonlinear dependencies on temperature, SOC, current direction as well as other factors; and hence is more accurate than the model in Chapter 2. Using the simulation plant model provides the flexibility to develop estimation algorithms and test it under a different operating scenarios. It helps in generating the required test data which is normally an elaborate process in case of batteries. To exploit this feature, certain modifications are done to this Simulink plant model.

A typical BMS is equipped with current, voltage and temperature sensors which have limited accuracy due to intrinsic measurement noise and bias. To test estimation algorithms
under different sensor properties, noise and bias are added to both current as well as terminal voltage signals using Simulink blocks. The noise added is Gaussian with zero-mean and a standard deviation of 1% of the corresponding signal’s maximum value. The noise standard-deviation is allowed to increase up to 2% to simulate effects like aging, stress and electromagnetic interference. The bias level is set as 2.5% of the corresponding signal’s maximum value. Bias level is stepped up from 0% to 2.5% and then 2.5% to 5%. Note that the Simulink plant model receives the true values of the signals (without noise) whereas the estimation algorithm receives the noisy and biased signals. Varying noise properties is considered for all simulations but, bias is considered only in the Sensor Bias section. To enable testing for both positive as well as negative initial SOC errors, the initial SOC of the simulation plant model is set to 90%. This is shown in the later sections. In addition to this, three different standard drive-cycles (or input datasets), representing benign to aggressive driving behaviors are also considered. They are chosen to cover c-rates up to 18C which means that the estimation algorithm may work for HEV and PHEV applications as well. Appendix A presents the detailed description of the thesis setup along with more information on the drive cycles used.

### 4.1.2 Performance Indices

The performance indices used to analyze the SOC estimation algorithms in this thesis are presented in this section. In [11, 35], several indices for estimation algorithms are discussed. In this study, to mitigate the effect of SOC initialization error, the first 600 samples (10 min) are ignored in the calculation of these indices except root mean square error. Lower index values indicate better performance.

**Root mean square error**: RMSE is the square root of mean of square of all errors. It is calculated using the actual and estimated values, and is computed for SOC as well as terminal voltage. It denotes the estimation accuracy. In this thesis, the algorithms are developed with the aim of $\tilde{z}_{rms} < 5\%$. The term $T$ denotes the total number of samples.

$$
\tilde{z}_{rms} = \sqrt{\frac{1}{T} \sum_{k=1}^{T} (z_k - \hat{z}_k)^2}, \quad \tilde{y}_{rms} = \sqrt{\frac{1}{T} \sum_{k=1}^{T} (y_k - \hat{y}_k)^2},
$$

**Infinity Norm of SOC Error**: It gives the worse case measure of the SOC error and is given by $|\tilde{z}|_{\infty} = \max_{k=n:N} |\tilde{z}_k|$ where $n = 600$ and $N$ is the length of the drive cycle. Since the sampling time is 1 s, this corresponding to ignoring the first 10 min of data.

**Mean SOC Error**: It is the average value of SOC error. It measures the average residual error in the estimate over the whole simulation time (first 600 samples are excluded). It is
denoted by $\mu \tilde{z}$.

**Variance of SOC Error:** It refers to the average variance of SOC error over whole simulation time (first 600 samples are excluded). Variance measures the estimate’s uncertainty and is denoted by $P \tilde{z}$. With every new measurement, the Kalman filter aims to reduce uncertainty and hence, the variance ideally decreases and remains constant at steady-state.

### 4.2 Simulation Results

In the following sections, simulation results for the various test scenarios are presented. When the actual initial SOC is unknown, the algorithm can be initialized at 50% SOC to restrict the maximum possible initial SOC error to less than 50%. Hence, the results shown are for 50% SOC initialization error and initial cell temperature of 25°C (standard ambient) unless otherwise specified. Note that since actual initial SOC is 90%, the estimation algorithms are initialized with SOC of 40% i.e. an initial SOC error of 50%.

The performance indices along with graphs are used to analyze the performance of the estimation algorithms. The SOC estimation error i.e. $\tilde{z} = z - \hat{z}$ along with three standard-deviation (3σ) uncertainty bounds are plotted for all test scenarios. Ideally, the estimation error should converge to zero. But, this does not happen due to model mismatch arising from parametric variations and sensor uncertainties. The error bounds represent uncertainty present in the estimation. The algorithms are also run for 60 drive cycles to test long-term performance.

#### 4.2.1 State EKF vs Dual EKF

Using Figures 4.1, 4.2 and Tables 4.1, 4.2; the performance of state and dual EKF can be compared. Regarding long-term performance (60 drive cycles), Dual EKF has approximately 2% improvement in SOC estimation for all three drive cycles. Both algorithms remove the large (50%) initial SOC error quickly in less than 10 min and SOC starts converging towards the true value. This is indicated by the SOC error plots approaching zero. These plots settle with a positive offset (less than 5%) except the case of US06 with state EKF where there are oscillations. At $t = 0$, the error bounds are large implying a large initial state uncertainty. However, as more measurement comes in, the uncertainty decreases and hence, the error bounds converge. Both algorithms show good long-term performance as well.

From the voltage tracking plots in Figures 4.1 and 4.2, it can be seen that even though the Kalman filter receives noisy voltage measurement, its voltage estimate is noiseless
showing that the Kalman filter is capable of removing noise. Since the noise is Gaussian with zero-mean, the estimate should lie in the middle of the noisy signal. For Artemis and US06 with state EKF, this does not happen implying poor performance. This can also be seen from the terminal voltage estimation error plots as well as from the large $\tilde{y}_{\text{rms}}$ values (Table 4.1). State EKF has a much higher voltage estimation error with US06 being the worst case. Since terminal voltage depends on the cell parameters, state EKF is unable to track terminal voltage accurately as it assumes the parameters to remain static. In US06, due to its aggressiveness, there exists large changes in temperature and current leading to large parametric variations, which results in large estimation errors. On the contrary, due to the benign nature of NEDC, there is relatively less parametric variation which results in a better terminal voltage estimation. The effect of stepping up noise variance in input and measurement is not visible in the estimates indicating the fact that Kalman filter eliminates the noise quickly. As expected, DEKF gives a much better voltage estimation as it can adapt to the variation of parameters.

Figure 4.3 shows the evolution of parameters for DEKF. It can be seen that all the parameters converge to reasonable value indicating steady-state and stability of the estimation algorithm. As expected, the internal resistance $R_s$ decreases with SOC and reaches a steady-state value. $R_s$ also tracks the actual value and this can be a helpful feature for power prediction. $R_1$ also exhibits a similar behavior. The parameters $R_2, C_2$ hardly exhibit any variation from initial to final values. This is because the input signal lacks the frequency components to excite the slow dynamics of $R_2, C_2$ branch. In such cases, better estimation may be achieved by holding $R_2, C_2$ constant and allowing only the other three parameters vary; however, this is not investigated in this thesis. Moreover, it is difficult to accurately estimate all five parameters from only one terminal voltage measurement.

During initialization, the initial SOC guess is generally based on the knowledge of previous charge/discharge cycle or by using the OCV-SOC relationship. Parameters are initialized corresponding to this initial SOC value. Since state EKF considers constant parameters, elaborate parameter maps corresponding to SOC values are to be stored so as to get an accurate initial guess for parameters. This requires large memory leading to increased costs. Preparing these maps is tedious requiring laboratory parameterization of the battery. Moreover, due to aging, these parameters tend to change thus, requiring frequent recalibration. However, in case of dual EKF, only the parameters corresponding to certain SOC values need to be stored as the algorithm is capable of tracking small variation in parameters. Due to this, recalibration is required less frequently. Hence, to match the performance of DEKF, state EKF requires large parameter maps.
4.2. Simulation Results

Figure 4.1 Results of EKF state estimation. (a) SOC estimation error along with $3\sigma$ error bounds, (b) Terminal voltage tracking and (c) Terminal voltage estimation error.

<table>
<thead>
<tr>
<th>State EKF Index</th>
<th>For 7 drive cycles</th>
<th>For 60 drive cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NEDC</td>
<td>Artemis</td>
</tr>
<tr>
<td>$\tilde{z}_{rms}$ [%]</td>
<td>2.69</td>
<td>2.49</td>
</tr>
<tr>
<td>$\tilde{y}_{rms}$ [mV]</td>
<td>8.8</td>
<td>19.94</td>
</tr>
<tr>
<td>$</td>
<td>\tilde{z}</td>
<td>_{\infty}$ [%]</td>
</tr>
<tr>
<td>$\mu_{\tilde{z}}$ [%]</td>
<td>2.50</td>
<td>2.47</td>
</tr>
<tr>
<td>$P_{\tilde{z}}$ [%]</td>
<td>0.01</td>
<td>0.014</td>
</tr>
</tbody>
</table>

Table 4.1 Performance indices for EKF state algorithm with correct SOC initialization.
Figure 4.2 Results of dual EKF algorithm. (a) SOC estimation error along with $3\sigma$ error bounds, (b) Terminal Voltage tracking, (c) Terminal voltage estimation error.

<table>
<thead>
<tr>
<th>Dual EKF</th>
<th>For 7 drive cycles</th>
<th>For 60 drive cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Index</td>
<td>NEDC</td>
<td>Artemis</td>
</tr>
<tr>
<td>$\tilde{z}_{rms}$ [%]</td>
<td>2.15</td>
<td>2.46</td>
</tr>
<tr>
<td>$\tilde{y}_{rms}$ [mV]</td>
<td>5.89</td>
<td>7.43</td>
</tr>
<tr>
<td>$</td>
<td>\tilde{z}</td>
<td>_{\infty}$ [%]</td>
</tr>
<tr>
<td>$\mu_{\tilde{z}}$ [%]</td>
<td>2.60</td>
<td>2.39</td>
</tr>
<tr>
<td>$P_{\tilde{z}}$ [%]</td>
<td>0.01</td>
<td>0.005</td>
</tr>
</tbody>
</table>

Table 4.2 Performance indices for dual EKF algorithm with correct SOC initialization.
From the next section, for brevity, results are presented only for US06 drive cycle due to its aggressive nature with large voltage and current variations. It is also evident that DEKF has an overall better performance and hence, it is considered over state EKF for following tests scenarios.

4.2.2 Sensor Bias

Current Sensor Bias

The deteriorating effects of sensor bias on the estimation algorithms are demonstrated in [16]. The results of DEKF with constant as well as varying current sensor bias are shown in Figures 4.4 and 4.5 respectively. DEKF performs well in the case of constant current sensor bias with SOC error settling at 1.28%. The indices $\bar{z}_{rms} = 2.32\%$ and $\bar{y}_{rms} = 8.46mV$ also indicate good performance. The algorithm is able to track the bias which is modeled as random-walk. The bias is initialized at zero (no bias) as the level is generally unknown unless the sensor is recently calibrated. Moreover, the algorithm should perform well for any bias-level in the range $[-x_{b,max}, x_{b,max}]$ where $x_{b,max}$ is the maximum bias-level which, in case of current sensor bias, is different for different drive cycles. For constant bias case, the bias estimation error settles within 100 min. Figure 4.5 shows that DEKF performs well under varying current sensor bias as well. The effect of increase in sensor bias-level at around 100 min and 550 min is seen by the corresponding increase in the SOC error but, it eventually decreases and settles down indicating stable performance. The algorithm takes a long time (approximately 400 min) to catch-up to
Chapter 4. Results and Discussions

the changing bias. Further tuning could improve the performance keeping in mind the trade-off between settling time and peak overshoot. Nonetheless, it can be concluded that DEKF is capable of handling constant as well as varying current sensor bias.

Figure 4.4 Results of DEKF algorithm with constant current sensor bias for US06 drive cycles. (a) SOC estimation error along with $3\sigma$ error bounds and (b) Current sensor bias tracking.

Figure 4.5 Results of DEKF algorithm with varying current sensor bias for US06 drive cycle: (a) SOC estimation error along with $3\sigma$ error bounds and (b) Current sensor bias tracking.

Voltage Sensor Bias

Figure 4.6 shows DEKF results with constant voltage sensor bias. It can be seen that the algorithm does not perform well as the SOC error settles at around 18%. Even after 120 min, the error bounds are still large implying a large estimation uncertainty. This is
4.2. Simulation Results

because the algorithm is unable to track voltage sensor bias accurately. The bias estimate plot shows large sustained oscillations with an average error of 24%. This behavior can be attributed to the flat OCV curve of LFP cell, especially in the range of [20%, 90%] where a small variation in voltage can correspond to a large difference in SOC value. This shows that SOC estimation for LFP cells requires voltage measurement with relatively high accuracy and precision than that of current measurement. This can be further understood from the observability analysis shown in Chapter 2.

Figure 4.6  Results of DEKF algorithm with constant voltage sensor bias for US06 drive cycle. (a) SOC estimation error with $3\sigma$ error bounds. (b) Voltage sensor bias tracking. The SOC error settles at around 18% implying poor performance.

The poor performance of DEKF for voltage sensor bias becomes obvious when observability analysis (discussed in Chapter 2) is considered. It was concluded that the system loses observability if only the first-order derivative term of Taylor series expansion is considered. DEKF uses first-order linearization and hence, the system with voltage sensor bias is not observable with DEKF. However, good performance is achieved with DEKF for the case of current sensor bias as the system is unobservable at only four discrete SOC operating points which tend to pass quickly as the system operates.

UKF can be used to improve the performance in case of voltage sensor bias. Also, since dual estimation framework was found to be better, dual UKF (or DUKF) is used. Note that the performance of UKF is similar to that of second-order EKF which includes second-order derivative terms in the Taylor series expansion [32] but, UKF is simpler to implement as it does not require calculation of Jacobian and Hessian matrices. Figure 4.7 shows that, in comparison to DEKF, DUKF performs better with $\tilde{z}_{rms} = 3.02\%$ and the average bias estimation error is now reduced to around 7%. DUKF reaches steady-state with very small error bounds within 50s and thus, has a much faster response. This performance improvement is achieved mainly due to the better approximation of mean and covariance as DUKF includes higher-order terms in Taylor’s series. Therefore, for DUKF with volt-
age sensor bias, the observability matrix in Equation (2.22) is now applicable instead of Equation (2.21).

![Figure 4.7](attachment:image.png)

*Figure 4.7* Results of DUKF algorithm with a constant voltage sensor bias for US06 drive cycle. (a) SOC estimation error with $3\sigma$ error bounds, (b) Voltage sensor bias tracking. The SOC error settles at around 2% implying good performance.

### 4.2.3 Dual EKF vs Dual UKF

In this section, dual EKF and dual UKF algorithms are compared in terms of robustness to state and parameter initialization as well as initial cell temperatures. Once again, Simulink plant model enables us to gather the required data to perform such tests. Without it, elaborate lab tests are needed to gather the required test data.

From Tables 4.2 and 4.3, it can be inferred that, both DEKF and DUKF give almost similar performance under nominal scenarios. DUKF is found to have slightly higher $\bar{y}_{rms}$ which maybe due to the artificial saturation in the of OCV (Refer Section 3.4).

<table>
<thead>
<tr>
<th>Dual UKF</th>
<th>For 7 drive cycles</th>
<th>For 60 drive cycles</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NEDC</td>
<td>Artemis</td>
</tr>
<tr>
<td>$z_{rms}$ [%]</td>
<td>2.16</td>
<td>1.28</td>
</tr>
<tr>
<td>$y_{rms}$ [mV]</td>
<td>11.16</td>
<td>8.98</td>
</tr>
<tr>
<td>$</td>
<td>\bar{z}</td>
<td>_\infty$ [%]</td>
</tr>
<tr>
<td>$\mu_{\bar{z}}$ [%]</td>
<td>1.78</td>
<td>0.87</td>
</tr>
<tr>
<td>$P_{\bar{z}}$ [%]</td>
<td>0.07</td>
<td>0.015</td>
</tr>
</tbody>
</table>

*Table 4.3* Performance indices for dual UKF algorithm with correct SOC initialization.
**Robustness to State and Parameter Initialization**

Robustness of the algorithms to different state and parameter initializations can be compared using Figure 4.8. As mentioned earlier, if the actual initial SOC is not known, then initializing the algorithm with a SOC guess of 50% restricts the maximum initial SOC error to 50%. Parameters are then initialized based on the respective initial SOC guess. It would also be of interest to test algorithms for initial SOC guess greater than 90% where the OCV curve becomes highly nonlinear. Hence, with the actual SOC at 90%, the simulations are run for SOC errors ranging from −5% to 50% which corresponds to initial SOC guess values from 95% to 40%.

It can be seen that both algorithms perform well for 30% and 50% SOC error cases with $\tilde{z}_{\text{rms}} < 2\%$. However, the other three cases have a negative overshoot of up to 22% with DEKF and 10% with DUKF. The cases corresponding to −5% and 10% SOC errors give the worst performance for both algorithms with the plots for these two cases being nearly indistinguishable. This is because the nonlinear effects come into play when operating in this SOC range. Parameters also exhibit large variations at extreme SOC values. However, both algorithms converge to steady-state values for all cases. DUKF has a settling mean of less than 5% for all cases but for DEKF, the settling mean is around 6.5% for −5% and 10% SOC error cases. From the zoomed-in plots, it is evident that DUKF is much faster in eliminating the effect of incorrect initial SOC guess compared to DEKF. In short, it can be said that both algorithms are robust to different state and parameter initializations with DUKF offering better performance especially when operating around 95% SOC.

**Robustness to Initial Cell Temperatures**

As explained in Chapter 2, the battery parameters depend on the cell temperatures which can vary with the environment. Therefore, it is essential to test the performance of the SOC estimation algorithm for different initial cell temperatures.

Figure 4.9 shows a significant difference in performances of DEKF and DUKF for varying initial cell temperatures. It is known that the cell parameters depend on temperature as well as SOC and, these parameters vary quickly at low temperatures in the range of $[-10^\circ, 10^\circ C]$. But, the variation of the parameters is much slower for temperatures greater than $10^\circ C$. Based on the surface temperature ($T_s$) measurement which is available in a typical BMS, two sets of tuning parameters are used; one for $T_s \leq 10^\circ C$ and other for $T_s > 10^\circ C$, to enable the algorithm to cope up with both the fast as well as slow variation of the parameters. The tuning parameters for $T_s \leq 10^\circ C$ have higher values of noise covariance. DUKF performs well for all cases with the SOC error settling at less than 5%. The lower the temperature, the longer the algorithm takes to reach steady-state as lower temperatures have high parametric modeling uncertainty caused by the fast variation of
Chapter 4. Results and Discussions

Fig. 4.8  Results of DEKF and DUKF algorithms for different state and parameter initializations. The plots are shown for different SOC errors. Parameters are initialized based on the corresponding initial SOC guess. The horizontal lines show ±5% SOC error. The plots on the right are zoomed-in versions to show initial SOC error response clearly.
parameters at lower temperatures. On the other hand, DEKF does not perform well at low temperatures even with two sets of tuning parameters. For initial cell temperatures of $-10^\circ C$ and $10^\circ C$, $\tilde{z}_{rms} > 30\%$. The best performance for both algorithms is got for initial cell temperatures of $30^\circ C$ and $40^\circ C$. This results show that DUKF can perform well for a wide range initial cell temperatures provided that sufficient tuning is done whereas DEKF fails.

From these tests, it can be concluded that UKF is able to perform well even under large uncertainties where EKF struggles. This is due to the fact that EKF performs first-order approximations of nonlinear functions which may have large errors if the nonlinearity is high. However, for nominal cases (no bias and initial cell temperatures at $25^\circ C$), both EKF and UKF exhibit similar performance.

![Figure 4.9](image.png)  
Figure 4.9  Results of DEKF and DUKF algorithms for different initial cell temperatures. The horizontal line shows $5\%$ SOC error.

### 4.3 Experimental Results

In this thesis, so far, tests are done based on data from the Simulink battery model. To validate the developed estimation algorithm, both DEKF and DUKF algorithms are tested using a real experimental dataset from Dr. Gregory L. Plett at the University of Colorado Springs. It is a widely used dataset in the battery community to validate algorithms and is available at [36].

The dataset consists of time series signals for cell’s SOC, terminal voltage and load current at different temperatures as well as the corresponding OCV-SOC curves. While gathering data, temperature is held constant using a thermal chamber [21]. The temperature
dependent values for Coulombic efficiency $\eta$ and total cell capacity $Q$ are also provided. The current input is a sequence of Urban Dynamometer Drive Schedule (UDDS) cycles separated by $5\text{min}$ relaxation periods with zero load current. The cell with initial SOC of 100% is subjected to a $40\text{A}$ discharge pulse and the UDDS-relaxation cycles are then applied alternatively in the SOC range $z \in [90\%, 10\%]$. Figure 4.10 shows input, measurement and true SOC of Plett’s dataset.

For the thesis, the dataset corresponding to cell temperature of $25^\circ C$ is used. The OCV-SOC curve considered is the average of charge and discharge curves. Assuming that the actual initial SOC is not known, the algorithms are initialized with a SOC guess of $50\%$. The parameters are also initialized corresponding to this SOC value but taken from the Simulink battery model since system identification is not done.

The simulation results, i.e. Figure 4.11 and Table 4.4, show that both DEKF and DUKF exhibit nearly similar performance. This is similar to the results obtained using the simulation plant model shown in Section 4.2.3. Both algorithms eliminate the initial SOC error quickly. A sudden increase in SOC error can be observed at around $580\text{min}$ caused by the abrupt decrease in the terminal voltage measurement data (Figure 4.10). This is because, when SOC is close to $0\%$, the battery is disconnected to prevent over-discharge causing this sudden terminal voltage drop. The results of this test show that the developed algorithms perform well under real scenarios as well.

<table>
<thead>
<tr>
<th>Index</th>
<th>DEKF</th>
<th>DUKF</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tilde{z}_{\text{rms}}$ [%]</td>
<td>0.96</td>
<td>0.74</td>
</tr>
<tr>
<td>$\tilde{y}_{\text{rms}}$ [mV]</td>
<td>12.32</td>
<td>18.48</td>
</tr>
<tr>
<td>$</td>
<td>\tilde{z}</td>
<td>_{\infty}$ [%]</td>
</tr>
<tr>
<td>$\mu_{\tilde{z}}$ [%]</td>
<td>-0.6</td>
<td>-0.23</td>
</tr>
<tr>
<td>$P_{\tilde{z}}$ [%]</td>
<td>0.005</td>
<td>0.004</td>
</tr>
</tbody>
</table>

*Table 4.4 Performance indices of DEKF and DUKF algorithms for Plett’s dataset. Initial SOC is set to $50\%$ and parameters are initialized corresponding to this SOC.*
4.3. Experimental Results

**Fig. 4.10** Plots showing Plett’s dataset. (a) Input - UDDS current drive cycle current, (b) Measurement - Terminal voltage, (c) True or Actual SOC.

**Fig. 4.11** Results of DEKF and DUKF for Plett’s dataset showing SOC estimation error with $3\sigma$ error bounds. Initial SOC error is 50%. For clarity purpose, the plots are zoomed-in along SOC error axis and hence, 50% error is not visible.
Chapter 5

Conclusion

The main contribution of this thesis lies in the development and testing of SOC estimation algorithms for a wide range of operational scenarios. There exists a difficulty in quantifying the performance of Kalman filtering algorithms due to the non-uniform tuning and testing conditions. Additionally, acquiring suitable battery system data for several different operating scenarios is cumbersome. In this thesis, an open-access and validated simulation model of a LiFePO₄ battery is used to develop the algorithms. This model facilitates fast and easy collection of the required data without the need for elaborate laboratory experiments. The operational scenarios considered are varying sensor noise properties, varying sensor bias properties, different initial states, parameters as well as cell temperatures; all of which are known to have a major impact on SOC estimation. The algorithms are also tested with three different drive cycles chosen to cover the c-rates corresponding to PHEVs and HEVs as well.

The state-space modeling of the battery and its parameters is first presented along with the augmented models considering the sensor biases. This is followed by an overview of the various Kalman filtering algorithms used. The development process started from the state EKF algorithm. A comparison between state and dual EKF revealed that, since dual EKF is able to adapt to parameter variations, it performs better. With suitable bias modeling, DEKF is also able to handle constant as well as varying current sensor bias. However, the algorithm failed in the case of voltage sensor bias as the first-order linearized system with voltage sensor bias was unobservable and this could be improved by including higher order derivatives in the linearization. UKF, an algorithm which performs similar to second-order EKF, gave good performance for the case of voltage sensor bias. DEKF and DUKF algorithms were then tested for different initial state, cell parameters and cell temperature initializations. Both algorithms gave reasonably good performance for different state and parameter initializations with DUKF being slightly better. Both algorithms could not handle the high parametric modeling uncertainties at low temperatures but, using a tuning-parameters scheduling scheme based on surface temperature
measurement enabled DUKF to give good performance for the entire range considered. Both algorithms were found to be robust to varying sensor noise properties. Finally, the algorithms were tested using a real dataset.

From the simulation results, it can be concluded that UKF performs better than EKF in case of hard nonlinearities and high initial uncertainties. However, under nominal scenarios, both algorithms exhibit similar performance. Considering the fact that EKF is simpler with less computational demands, the choice of estimation algorithms should be based on the battery’s operating conditions. From the sensor bias scenarios, it can be inferred that sensor bias modeling is vital for better estimation performance and observability analysis can give an insight into the choice and performance of estimation algorithms. Using the simulation model enabled fast and easy collection of the data required to develop and test SOC estimation algorithms for a variety of operational scenarios. From the test with different initial cell temperatures, it can be concluded that both the filter type as well as its tuning is significant for performance of estimation algorithms.

Suggestions for Future Work

A list of suggestions are presented here as possible extensions or improvements to the thesis.

**Non-Gaussian noise** - Kalman filtering framework is based on the assumption that the noise is Gaussian with zero-mean. However, real systems may be affected by non-gaussian noises and hence, the algorithms have to be tested for such noise properties as well. As a start, the sensor properties in the simulation plant model can be modified to include non-gaussian noise.

**Operation with bias in current and voltage sensors** - The estimation algorithms with bias modeling cannot handle both biases simultaneously as there will be an ambiguity whether the error in estimated output is due to current sensor bias or voltage sensor bias. Other methods are to be investigated to tackle this issue. Nonlinear observability analysis considering both biases is also not done.

**Validation with real data** - The estimation algorithms are developed and tested using the simulation plant model and hence, require further validation using real data. As a start, Plett’s dataset for LPF cell was used. With the availability of more real datasets from different scenarios, further validation can be done.

**Aging and Capacity estimation** - Aging has a significant effect on the battery parameters and also causes the capacity to decrease. It was not considered in this thesis as the
Simulink model did not have aging characteristics. By performing further experiments and collecting suitable data, the Simulink model can be upgraded to consider aging as well. The same model can then be used to develop state-of-health algorithms as well, which are also essential for the BMS. Aging and capacity estimation can be done in a separate loop operating on a slower time scale.

**SIL/HIL/BIL testing** - BMS algorithms are implemented using microcontrollers which has constraints on memory and processing power. The developed algorithms can be tested using software-in-loop (SIL), hardware-in-loop (HIL) and battery-in-loop (BIL) setups to bring it closer to hardware implementation. As a prerequisite, MATLAB to microcontroller code generation should be first investigated.

**Better tuning procedure** - The impact of tuning on the filter performance is well known and in this thesis, a considerable amount of time was spent on tuning. Moreover, due to the structure of ECM parameter models, it was hard to get a intuitive understanding of the tuning process. A more systematic method is worth exploring; one that can ease the tuning process and can give a better understanding of it.
Chapter 5. Conclusion
Appendix A

Simulation Plant Model of LFP cell

In this appendix chapter, the Simulink plant model of LiFePO$_4$ cell used in this thesis to simulate the measurement data required for Kalman filtering algorithm is presented. Using a simulation model for the plant offers the flexibility to perform tests with the estimation algorithm under different scenarios. The modifications made to this simulation plant model are also discussed.

A.1 Model Description

The nonlinear plant model of LiFePO$_4$ cell is experimentally parameterized and validated by University of Michigan, Ann Arbor [25]. The Simulink implementation of this model is published with open-access [26].

It is an electro-thermal single-cell model consisting of an electrical equivalent circuit model (EECM) along with a two-state thermal model, and is constructed for A123 26650 LiFePO$_4$ battery. Both electrical and thermal subsystems are lumped, and are coupled through heat generation ($Q_H$) and temperature dependency of the electrical parameters. This model, which has 5 states in total, captures state of charge (SOC), voltages across RC branches ($V_1, V_2$), core temperature ($T_c$) and surface temperature ($T_s$). The electrical and thermal subsystems are parameterized by pulse-relaxation and drive-cycle tests separately. Validation is done using terminal voltage and surface temperature measurements from realistic drive cycle experiments [25].

The simulation model has two inputs, namely current ($I$) and coolant temperature ($T_f$). The outputs can be configured to obtain a wide range of variables as required. In the next section, the electrical and thermal sub-models are explained in detail.

An important point to note is that plant model described here is different from the one in Chapter 2. This model is used to simulate the measurements required for the Kalman filter
Appendix A. Simulation Plant Model of LFP cell

whereas the model in Chapter 2 is used by the filter itself. Apart from the measurements, the Kalman filter does not have access to any other parameters from this model.

A.1.1 Electrical Subsystem

The electrical subsystem consists of a 2 RC EECM of the LiFePO$_4$ cell. This model 3 state states, namely SOC ($z$), voltage across the two RC pairs ($V_1, V_2$). The electrical subsystem is shown in Figures A.1 and A.2. The inputs are current ($I$) and average of core and surface temperatures of the battery ($T_m$) whereas, the output is the terminal voltage ($V_T$). These parameters of the electrical model are dependent on SOC, $T_m$ as well as the current direction i.e. charge and discharge. The battery capacity ($Q$) is constant at 2.3Ah and is independent of the c-rate.

Laboratory experiments are conducted using an actual A123 26650 LiFePO$_4$ cell to parameterize the electrical subsystem [25] and three look-up tables are obtained. These tables are stored as MATLAB and are used by the model. OCV-SOC relationship for a battery can be found by cycling the battery with suitable rest periods. Hysteresis results in two separate OCV-SOC curves, i.e. for charge and discharge. This model uses the average of these curves to find the value of OCV corresponding to a known SOC. The 2D look-up tables are used to select electrical circuit parameters based on $\text{sgn}(I)$, $T_m$ and SOC. The $\frac{\text{d}V_{OC}}{\text{d}T_m}$ curve is used in calculation of entropic heat generation.

The continuous time state-space model of the electrical subsystem is given by

\[
\dot{V}_1(t) = -\frac{1}{R_1(z, T_m, \text{sgn}(I))C_1(z, T_m, I)} V_1(t) + \frac{1}{C_1(z, T_m, \text{sgn}(I))} I(t) \quad (A.1)
\]

\[
\dot{V}_2(t) = -\frac{1}{R_2(z, T_m, \text{sgn}(I))C_2(z, T_m, \text{sgn}(I))} V_2(t) + \frac{1}{C_2(z, T_m, \text{sgn}(I))} I(t) \quad (A.2)
\]

\[
\dot{z}(t) = -\frac{\eta}{3600Q} I(t) \quad (A.3)
\]

\[
V_T(t) = V_{OC}(z(t)) - V_1(t) - V_2(t) - R_s(z, T_m, \text{sgn}(I))I(t). \quad (A.4)
\]

Equation (A.3) calculates SOC using Coulomb counting where $\eta$ (considered as unity here) is the Coulombic efficiency. The OCV corresponding to a known SOC can be found using the OCV-SOC look-up table. With the values of $I$, $T_m$ and SOC, the parameter look-up table can be used to find the electrical circuit parameters, which can then be used to find $V_1$ and $V_2$ from Equations (A.1) and (A.2). The terminal voltage ($V_T$) can be calculated by applying Kirchoff’s voltage law to the electrical circuit (Equation (A.4)).

The total heat generation ($Q_H$) inside a cell is given by Equation (A.5), where the first and second terms correspond to joule and entropic heating respectively. The unit of $T_m$ is
A.1. Model Description

Kelvin. The value of $Q_H$ is then fed to the thermal subsystem.

$$Q_H(t) = \underbrace{I(t)(V_{OC}(z(t)) - V_T(t))}_{\text{Joule heating}} - \underbrace{I(t)T_m(t)\frac{dV_{OC}}{dT_m}}_{\text{Entropic heat generation}} \quad (A.5)$$

Although the electrical model (Chapter 2) used for estimator design (Chapter 3) is structurally similar to the simulation model presented here, there is a significant model mismatch due to state-dependent parametric variations (uncertainty). The proposed estimation algorithms handles this parametric uncertainty through online adaptation scheme i.e., dual state and parameter estimation.

![Fig. A.1 LFP Simulink plant model - Electrical subsystem.](image)

**Fig. A.1** LFP Simulink plant model - Electrical subsystem.

### A.1.2 Thermal Subsystem

The continuous time state-space model of the thermal subsystem is given by

$$\dot{T}_c(t) = \frac{T_s(t) - T_c(t)}{R_cC_c} + Q_H(t) \quad (A.6)$$

$$\dot{T}_s(t) = \frac{T_c(t) - T_s(t)}{R_cC_s} + \frac{T_f(t) - T_s(t)}{R_uC_s}. \quad (A.7)$$
The thermal sub-model is shown in Figure A.3. The state variables are core temperature ($T_c$) and surface temperature ($T_s$). The total heat generated ($Q_H$) and coolant temperature ($T_f$) are the inputs. Unlike the electrical parameters, the thermal parameters, namely thermal conductive resistance ($R_c$), thermal convective resistance ($R_u$), heat capacity of battery surface ($C_s$) and heat capacity of battery core ($C_c$) are considered to be constant. These parameters are identified using Urban Assault Cycle (UAC) experimental procedure described in [25] and are shown in Table A.1. The thermal-to-electrical subsystem coupling is through the average cell temperature given by

$$T_m(t) = \frac{T_s(t) + T_c(t)}{2}$$  \hspace{1cm} (A.8)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_c$ ($K/W$)</td>
<td>1.94</td>
</tr>
<tr>
<td>$R_u$ ($K/W$)</td>
<td>15</td>
</tr>
<tr>
<td>$C_s$ ($J/K$)</td>
<td>62.7</td>
</tr>
<tr>
<td>$C_c$ ($J/K$)</td>
<td>4.5</td>
</tr>
</tbody>
</table>

Table A.1 Thermal subsystem parameters.

Using a simulation model for the plant offers several advantages. It provides us more flexibility to conduct different tests under various operating conditions. Additionally, it also facilitates the evaluation of estimator’s tracking performance by giving access to true SOC and parameter values. It is pertinent to mention here that the typical approach to evaluate tracking performance based on using the same current measurement for SOC reference generation as well as for SOC estimation can be prone at errors caused by the inaccuracies in the sensor used for SOC reference generation. This issue can be easily resolved in a simulation model but in experimental studies, it would require a very high precision current measurement equipment to get true SOC reference trajectory.

### A.2 Modifications to the Simulink Plant Model

To exploit the afore-mentioned flexibility offered by the simulation model, some modifications are done to the Simulink model as shown in Figure A.5.
A.2. Modifications to the Simulink Plant Model

Fig. A.3  LFP Simulink plant model - Thermal subsystem.

Fig. A.4  LFP Simulink plant model - Coupled electrical and thermal subsystems.
A.2.1 Addition of Sensors

A typical BMS is equipped with current, voltage, and temperature sensors. These sensors have limited accuracy due to intrinsic measurement noise and bias. In addition, sensors are also prone to electromagnetic interference, physical stress, and ambient temperature, which may cause variation in their bias-level and noise statistics. To ensure reliability, the estimation algorithms must be robust to such variations.

Figure A.6 shows the overall simulation setup consisting of the plant as well as the estimation algorithm. The Simulink plant model receives the true values of signals without noise whereas the estimation algorithm receives the noisy and biased sensor measurements.

Sensor Noise

To model sensor noise, Random Number blocks are added to the Simulink model both at the input ($I$) and output ($V_T$). It is common to assume sensor noise to be Gaussian with zero mean and a known standard deviation [14]. In this thesis, the noise standard deviation is set as 1% of the maximum value of the signal under consideration. For e.g., if maximum terminal voltage of the cell is 3.6V, the noise added to this signal by the Random Number block would have zero mean and a standard deviation of 0.036V (1% of 3.6). The estimation algorithm receives this noisy signal. In addition to this, the noise standard deviation is stepped up from 1% to 2% at 60% (arbitrarily chosen) of total simulation duration to test the robustness of estimation algorithm under varying sensor noise properties. Figure A.7
A.2. Modifications to the Simulink Plant Model

Fig. A.6  MATLAB Simulation Setup

shows an example of a terminal voltage signal with and without the added sensor noise. It can be seen that around 80 min, there is an increase in the signal noise.

Fig. A.7  Terminal Voltage signal with and without sensor noise.

Sensor Bias

Calibration errors and aging in any real sensor may give rise to unknown biases in the sensor measurement. Bias is basically a DC offset present in the sensor measurement and is known to adversely deteriorate the accuracy of estimation algorithm over time [16]. In this study, bias is considered in both current as well as voltage sensors. The bias-level is set at 2.5% of maximum value of signal under consideration. For e.g., if the maximum terminal voltage is 3.6V, the bias added to this signal is 0.09V (2.5% of 3.6). Similar to the case with sensor noise, the robustness of the estimation algorithm to varying sensor bias should also be tested. In this case, the bias is increased in two steps i.e from 0% to 2.5% and 2.5% to 5% at 10% and 50% of the total simulation times respectively. Figure A.8 shows an example of actual and measured current signals with two step changes in
Appendix A. Simulation Plant Model of LFP cell

bias (around 15 min and 65 min of driving instants). The measured signal is affected by noise as well as bias and is fed to the estimation algorithm.

Fig. A.8 Figures showing actual and measured current signals. Measured signal is affected by sensor noise and bias. (a) Current signal, (b) Magnified to show increase in bias.

A.2.2 Current Input

Based on how the electric vehicle is driven, the battery is subjected to different loading (current) profiles. In this thesis, three different standard driving cycles, representing benign to aggressive driving patterns, are considered to test the estimation algorithm. Each of these cycle are chosen to represent a certain driving behavior. Aggressive drive cycles are characterized by large variations in speeds and accelerations which cause large variations in currents and temperatures.

In order to test the long-term performance of the estimation algorithm, several drive (discharge) cycles of the same type are put-together interleaved with charging cycles. The charging cycle is designed to restore SOC back to the same initial value before the occurrence of the discharge cycle. The charging c-rate is kept constant at 4c for all drive cycles. The initial SOC can be specified in MATLAB. Table A.2 shows the characteristics of drive cycles used. The current profiles are obtained by simulating these drive cycles on a Toyota Prius II PHEV in pure EV mode. The c-rate of the drive cycles used cover the range of BEV, PHEV as well as HEV.

New European Drive Cycle (NEDC)

The NEDC is designed to assess fuel economy and emission levels in passenger cars. It was the reference drive cycle used for homologating vehicles until Euro 6. The cycle is
supposed to represent a typical European car usage but is often criticized for not representing real life driving conditions and producing unrealistic economy figures [37]. NEDC, in this thesis, represents mild driving behavior with a number of soft accelerations, constant speed cruises and idling. Figure A.9a shows the current profile for NEDC.

**Artemis Rural Drive Cycle**

Common Artemis Driving Cycles (CADC) are developed as a part of European Artemis project and are based on statistical analysis of a large database of European real world driving patterns [38]. The cycle has three variants, each representing urban road, rural road and motorway. In this thesis, the Artemis rural road drive cycle is used as it represents realistic driving conditions and hence can access the real performance of the estimation algorithm. Artemis rural drive cycle is slightly more aggressive than NEDC with a higher average speed, current and cell temperature. Figure A.9b shows the current profile of Artemis rural drive cycle.

**US06 Drive Cycle**

The US06 drive cycle was designed to test the exhaust emissions of vehicles under high speeds and accelerations. It represents an aggressive driving profile with high speeds and rapid speed fluctuations [39]. In this thesis, US06 is the most aggressive drive cycle used and is chosen so as to test the estimator’s performance under extreme conditions of current and cell temperature. Figure A.9c shows the current profile of US06 drive cycle.

<table>
<thead>
<tr>
<th></th>
<th>Distance (km)</th>
<th>Duration (s)</th>
<th>Avg Speed (km/h)</th>
<th>c-rate</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>NEDC</strong></td>
<td>11</td>
<td>1180</td>
<td>33.6</td>
<td>6.8c</td>
</tr>
<tr>
<td><strong>Artemis Rural</strong></td>
<td>17.3</td>
<td>1082</td>
<td>57.5</td>
<td>9.7c</td>
</tr>
<tr>
<td><strong>US06</strong></td>
<td>12.8</td>
<td>593</td>
<td>77.9</td>
<td>17c</td>
</tr>
</tbody>
</table>

*Table A.2* Characteristics of the drive cycles used [37] [39].
Figure A.9  Current profiles for the drive cycles used. (a) NEDC, (b) ARTEMIS Rural and (c) US06; listed from benign to aggressive.
Appendix B

Kalman Filtering Algorithms

The steps for various Kalman filtering algorithms used in thesis are shown in this appendix chapter. Algorithm 2 shows the general Bayseian inference followed by the linear Kalman filter, EKF and UKF. The last two algorithms namely, DEKF and DUKF are for simultaneous state and parametric estimation. These algorithms are further described in Chapter 3.

Algorithm 2 : General Bayseian inference algorithm for state estimation from [13]

1: General state-space model
   \[ x_k = f(x_{k-1}, u_{k-1}, w_{k-1}, k - 1) \]  \( \triangleright \) State equation
2: \[ y_k = g(x_k, u_k, v_k, k) \]  \( \triangleright \) Output equation
3: Initialization : For \( k = 0 \), define
   4: \[ \hat{x}_0^+ = \mathbb{E}[x_0] \]  \( \triangleright \) Initialize mean
   5: \[ P_{\hat{x},0}^+ = [(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T] \]  \( \triangleright \) Initialize error covariance
6: \textbf{for} \( k = 1, 2, \ldots N \) \textbf{do}
   7: \[ \hat{x}_k^- = \mathbb{E}[f(x_{k-1}, u_{k-1}, w_{k-1}, k - 1)|y_{k-1}] \]  \( \triangleright \) Loop till number of samples N
   8: \[ P_{\hat{x},k}^- = \mathbb{E}[(\hat{x}_k^-)(\hat{x}_k^-)^T] \]  \( \triangleright \) State estimate time update
   9: \[ \hat{y}_k = \mathbb{E}[h(\hat{x}_k^-, u_k, v_k)|y_{k-1}] \]  \( \triangleright \) Error covariance time update
10: \[ \hat{y}_k = y_k - \hat{y}_k \]  \( \triangleright \) Output estimate
11: \[ L_k = \mathbb{E}[(\hat{x}_k^-)(\hat{y}_k)^T]\mathbb{E}[(\hat{y}_k)(\hat{y}_k)^T] \]  \( \triangleright \) Innovation
12: \[ \hat{x}_k^+ = \hat{x}_k^- + L_k\hat{y}_k \]  \( \triangleright \) Kalman gain
13: \[ P_{\hat{x},k}^+ = P_{\hat{x},k}^- - L_kP_{\hat{y},k}^-L_k \]  \( \triangleright \) State estimate measurement update
14: \textbf{end for}  \( \triangleright \) Error covariance measurement update
### Algorithm 3: Linear Kalman filter algorithm for state estimation from [4]

Linear discrete state-space model with additive noise

1: $x_{k+1} = A_k x_k + B_k u_k + w_k$  \(\triangleright\) State equation
2: $y_k = C_k x_k + D_k u_k + v_k$  \(\triangleright\) Output equation

3: Initialization: For $k = 0$, define
4: $\hat{x}_0^+ = \mathbb{E}[x_0]$  \(\triangleright\) Initialize mean
5: $P_{\hat{x},0}^+ = [(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T]$  \(\triangleright\) Initialize error covariance

6: for $k = 1, 2, \ldots, N$ do  \(\triangleright\) Loop till number of samples $N$
7: $\hat{x}_k^- = A_k \hat{x}_{k-1}^+ + B_{k-1} u_{k-1}$  \(\triangleright\) State estimate time update
8: $P_{\hat{x},k}^- = A_k^- P_{\hat{x},k-1}^+ A_k^T + P_w$  \(\triangleright\) Error covariance time update
9: $L_k = P_{\hat{x},k}^- C_k [C_k P_{\hat{x},k}^- C_k + P_v]^{-1}$  \(\triangleright\) Kalman gain
10: $\hat{y}_k = y_k - [C_k \hat{x}_k^- + D_k u_k]$  \(\triangleright\) Innovation
11: $\hat{x}_k^+ = \hat{x}_k^- + L_k \hat{y}_k$  \(\triangleright\) State estimate measurement update
12: $P_{\hat{x},k}^+ = (I - L_k C_k) P_{\hat{x},k}^-$  \(\triangleright\) Error covariance measurement update
13: end for

### Algorithm 4: Extended Kalman filter algorithm for state estimation from [4]

Non-linear discrete state-space model with additive noise

1: $x_{k+1} = f(x_k, u_k) + w_k$  \(\triangleright\) State equation
2: $y_k = g(x_k, u_k) + v_k$  \(\triangleright\) Output equation

3: Calculate Jacobians:
4: $A_{k-1} = \left. \frac{\partial f(x_{k-1}, u_{k-1})}{\partial x_{k-1}} \right|_{x_{k-1} = \hat{x}_{k-1}^+}$, $C_k = \left. \frac{\partial g(x_k, u_k)}{\partial x_k} \right|_{x_k = \hat{x}_k^+}$

5: Initialization: For $k = 0$, define
6: $\hat{x}_0^+ = \mathbb{E}[x_0]$  \(\triangleright\) Initialize mean
7: $P_{\hat{x},0}^+ = [(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T]$  \(\triangleright\) Initialize error covariance

8: for $k = 1, 2, \ldots, N$ do  \(\triangleright\) Loop till number of samples
9: $\hat{x}_k^- = f(\hat{x}_{k-1}^+, u_{k-1})$  \(\triangleright\) State estimate time update
10: $P_{\hat{x},k}^- = A_{k-1} P_{\hat{x},k-1}^+ A_{k-1}^T + P_w$  \(\triangleright\) Error covariance time update
11: $L_k = P_{\hat{x},k}^- C_k [C_k P_{\hat{x},k}^- C_k + P_v]^{-1}$  \(\triangleright\) Kalman gain
12: $\hat{y}_k = y_k - g(\hat{x}_{k}^-, u_k)$  \(\triangleright\) Innovation
13: $\hat{x}_k^+ = \hat{x}_k^- + L_k \hat{y}_k$  \(\triangleright\) State estimate measurement update
14: $P_{\hat{x},k}^+ = (I - L_k C_k) P_{\hat{x},k}^-$  \(\triangleright\) Error covariance measurement update
15: end for
Algorithm 5: Dual extended Kalman filter algorithm for state and parameter estimation from [21]

Non-linear discrete state-space model with additive noise

1: \( x_{k+1} = f(x_k, u_k, \theta_k) + w_k, y_k = g(x_k, u_k, \theta_k) + v_k \) \hspace{1cm} \triangleright \text{For state filter}

2: \( \theta_{k+1} = \theta_{k+1} + r_k, d_k = g(x_k, u_k, \theta_k) + e_k \) \hspace{1cm} \triangleright \text{For parameter filter}

3: Calculate Jacobians:

4: \( A_{k-1} = \frac{\partial f(x_{k-1}, u_{k-1}, \hat{\theta}_k)}{\partial x_k} \bigg|_{x_k = \hat{x}_k^-} \), \( C_{\theta}^x = \frac{\partial g(x_k, u_k, \hat{\theta}_k)}{\partial \theta_k} \bigg|_{\theta = \hat{\theta}_k^-} \)

5: \( C_{\theta}^\theta = \frac{\partial g(\hat{x}_k^-, u_k, \theta)}{\partial \theta} \bigg|_{\theta = \hat{\theta}_k^-} \)

6: Initialization: For \( k = 0 \), define mean and error covariance

7: \( \hat{\theta}_0^+ = \mathbb{E}[\theta_0] \), \( P_{\theta,0}^+ = [(\theta_0 - \hat{\theta}_0^+)(\theta_0 - \hat{\theta}_0^+)^T] \) \hspace{1cm} \triangleright \text{For parameter filter}

8: \( \hat{x}_0^+ = \mathbb{E}[x_0] \), \( P_{x,0}^+ = [(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T] \) \hspace{1cm} \triangleright \text{For state filter}

9: for \( k = 1, 2, ..., N \) do \hspace{1cm} \triangleright \text{Loop till number of samples}

10: \( \hat{\theta}_k^- = \hat{\theta}_{k-1} \) \hspace{1cm} \triangleright \text{Parameter filter time update}

11: \( P_{\theta,k}^- = P_{\theta,k-1}^- + P_{\theta} \) \hspace{1cm} \triangleright \text{State filter time update}

12: \( \hat{x}_k^- = f(\hat{x}_{k-1}^+, u_{k-1}, \hat{\theta}_k^-) \)

13: \( P_{\hat{x},k}^- = A_{k-1} P_{\hat{x},k-1}^- A_{k-1}^T + P_w \)

14: \( L_k^x = P_{\hat{x},k}^- (C_{\theta}^x)^T [C_{\theta}^x P_{\hat{x},k}^- (C_{\theta}^x)^T + P_v]^{-1} \) \hspace{1cm} \triangleright \text{State filter Kalman gain}

15: \( \hat{y}_k^x = y_k - g(\hat{x}_k^-, u_k, \hat{\theta}_k^-) \) \hspace{1cm} \triangleright \text{State filter innovation}

16: \( \hat{x}_k^+ = \hat{x}_k^- + L_k^x \hat{y}_k^x \) \hspace{1cm} \triangleright \text{State estimate measurement update}

17: \( P_{\hat{x},k}^+ = (I - L_k^x C_{\theta}^x) P_{\hat{x},k}^- \) \hspace{1cm} \triangleright \text{State error covariance measurement update}

18: \( L_k^\theta = P_{\theta,k}^- (C_{\theta}^\theta)^T [C_{\theta}^\theta P_{\theta,k}^- (C_{\theta}^\theta)^T + P_v]^{-1} \) \hspace{1cm} \triangleright \text{Parameter filter Kalman gain}

19: \( \hat{y}_k^\theta = y_k - g(\hat{x}_k^-, u_k, \hat{\theta}_k^-) \) \hspace{1cm} \triangleright \text{Parameter filter innovation}

20: \( \hat{\theta}_k^+ = \hat{\theta}_k^- + L_k^\theta \hat{y}_k^\theta \) \hspace{1cm} \triangleright \text{Parameter estimate measurement update}

21: \( P_{\hat{\theta},k}^+ = (I - L_k^\theta C_{\theta}^\theta) P_{\hat{\theta},k}^- \) \hspace{1cm} \triangleright \text{Parameter error covariance measurement update}

22: end for
Algorithm 6 : Unscented Kalman filter algorithm for state estimation from [13]

Non-linear discrete state-space model with additive noise

1: $x_{k+1} = f(x_k, u_k) + w_k$ \quad \triangleright \text{State equation}

2: $y_k = g(x_k, u_k) + v_k$ \quad \triangleright \text{Output equation}

3: Define : $p = 2 \cdot \dim(x_k)$

4: Initialization : For $k = 0$, define

5: $\hat{x}_0 = \mathbb{E}[x_0]$ \quad \triangleright \text{Initialize mean}

6: $P_{\hat{x},0} = [(x_0 - \hat{x}_0)(x_0 - \hat{x}_0)^T]$ \quad \triangleright \text{Initialize error covariance}

7: for $k = 1, 2, \ldots, N$ do \quad \triangleright \text{Loop till number of samples}

8: $X^+_{k-1} = \{\hat{x}^+_{k-1}, \hat{\hat{x}}^+_{k-1} + \gamma \sqrt{P^+_{\hat{x},k-1}}, \hat{x}^-_{k-1} - \gamma \sqrt{P^+_{\hat{x},k-1}}\}$ \quad \triangleright \text{State sigma points}

9: $X^+_{k,i} = f(X^+_{k-1,i}, u_{k-1})$ \quad \triangleright \text{State estimate time update}

10: $\hat{x}^-_k = \sum_{i=0}^{p} \alpha_{m} X^+_{k,i}$

11: $P_{\hat{x},k} = \sum_{i=0}^{p} \alpha_{c} (X^+_{k,i} - \hat{x}_k)(X^+_{k,i} - \hat{x}_k)^T + P_w$ \quad \triangleright \text{Error covar time update}

12: $Y^+_{k,i} = g(X^+_{k,i}, u_k)$

13: $\hat{y}_k = \sum_{i=0}^{p} \alpha_{m} Y^+_{k,i}$ \quad \triangleright \text{Output estimate}

14: $P_{\hat{y},k} = \sum_{i=0}^{p} \alpha_{c} (Y^+_{k,i} - \hat{y}_k)(Y^+_{k,i} - \hat{y}_k)^T + P_v$ \quad \triangleright \text{Output error covariance}

15: $P_{\hat{x},\hat{y},k} = \sum_{i=0}^{p} \alpha_{c} (X^+_{k,i} - \hat{x}_k)(Y^+_{k,i} - \hat{y}_k)^T$

16: $L_k = P_{\hat{x},\hat{y},k}^{-1} P_{\hat{y},k}$ \quad \triangleright \text{Kalman gain}

17: $\hat{x}^+_k = \hat{x}^-_k + L_k(y_k - \hat{y}_k)$ \quad \triangleright \text{State estimate measurement update}

18: $P^+_{\hat{x},k} = P^-_{\hat{x},k} - L_k P_{\hat{y},k} L_k^T$ \quad \triangleright \text{Error covariance measurement update}

19: end for
Algorithm 7: Dual unscented Kalman filter algorithm for state and parameter estimation from [22]

Non-linear discrete state-space model with additive noise

1. $x_{k+1} = f(x_k, u_k, \theta_k) + u_k$, $y_k = g(x_k, u_k, \theta_k) + v_k$  \(\triangleright\) For state filter
2. $\theta_{k+1} = \theta_{k+1} + r_k$, $d_k = g(x_k, u_k, \theta_k) + e_k$  \(\triangleright\) For parameter filter

3. Define: $p = 2 \cdot dim(x_k)$
4. Initialization: For $k = 0$, define mean and error covariance
5. $\hat{\theta}_0^+ = E[\theta_0]$, $P_{\hat{\theta},0}^+ = [(\theta_0 - \hat{\theta}_0^+)(\theta_0 - \hat{\theta}_0^+)^T]$  \(\triangleright\) For parameter filter
6. $\hat{x}_0^+ = E[x_0]$, $P_{\hat{x},0}^+ = [(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T]$  \(\triangleright\) For state filter

7. for $k = 1, 2, \ldots N$ do  \(\triangleright\) Loop till number of samples
8. $\hat{\theta}_k^- = \hat{\theta}_{k-1}^-$  \(\triangleright\) Parameter filter time update
9. $P_{\hat{\theta},k}^- = P_{\hat{\theta},k-1}^- + P_r$  \(\triangleright\) Error covar time update
10. $X_{k-1}^- = \{\hat{x}_{k-1}^-, \hat{x}_{k-1}^+ + \gamma_x \sqrt{P_{\hat{x},k-1}^- \hat{x}_{k-1}^+ - \gamma_x \sqrt{P_{\hat{x},k-1}^-} \}$  \(\triangleright\) State sigma pts.
11. $X_{k,i}^- = f(X_{k-1,i}^-, u_{k-1}, \hat{\theta}_k^-)$  \(\triangleright\) State estimate time update
12. $\hat{x}_k^- = \sum_{i=0}^{p} \alpha_i^- X_{k,i}^-$
13. $P_{\hat{x},k}^- = \sum_{i=0}^{p} \alpha_i^- (X_{k,i}^- - \hat{x}_k^-)(X_{k,i}^- - \hat{x}_k^-)^T + P_w$  \(\triangleright\) Error covar time update
14. $\Theta_k = \{\hat{\theta}_k^-, \hat{\theta}_k^+ + \gamma_{\theta} \sqrt{P_{\hat{\theta},k}^-} \hat{\theta}_k^+ - \gamma_{\theta} \sqrt{P_{\hat{\theta},k}^-} \}$  \(\triangleright\) Parameter sigma points
15. $D_{k,i} = h(f(\hat{x}_{k-1,i}^-, u_{k-1}, \Theta_{k,i}), u_k, \Theta_{k,i})$  \(\triangleright\) Parameter filter output estimate
16. $d_k^- = \sum_{i=0}^{p} \alpha_i^- D_{k,i}$
17. $Y_{k,i}^- = g(X_{k,i}^-, u_k, \hat{\theta}_k^-)$  \(\triangleright\) State filter output estimate
18. $\hat{y}_k^- = \sum_{i=0}^{p} \alpha_i^- Y_{k,i}^-$
19. $P_{\hat{y},k}^- = \sum_{i=0}^{p} \alpha_i^- (Y_{k,i}^- - \hat{y}_k^-)(Y_{k,i}^- - \hat{y}_k^-)^T + P_v$  \(\triangleright\) State output error covariance
20. $P_{\hat{x}\hat{y},k}^- = \sum_{i=0}^{p} \alpha_i^- (X_{k,i}^- - \hat{x}_k^-)(Y_{k,i}^- - \hat{y}_k^-)^T$  \(\triangleright\) State filter Kalman gain
21. $L_k^+ = P_{\hat{x}\hat{y},k}^- P_{\hat{y},k}^{-1}$  \(\triangleright\) State filter Kalman gain
22. $P_{\hat{d},k}^- = \sum_{i=0}^{p} \alpha_i^- (D_{k,i}^- - \hat{d}_k^-)(D_{k,i}^- - \hat{d}_k^-)^T + P_e$  \(\triangleright\) Para. output error covar.
23. $P_{\hat{\theta}\hat{d},k}^- = \sum_{i=0}^{p} \alpha_i^- (\Theta_{k,i}^- - \hat{\theta}_k^-)(D_{k,i}^- - \hat{d}_k^-)^T$  \(\triangleright\) Parameter filter Kalman gain
24. $L_k^+ = P_{\hat{\theta}\hat{d},k}^- P_{\hat{d},k}^{-1}$  \(\triangleright\) Parameter filter Kalman gain
25. $\hat{x}_k^+ = \hat{x}_k^- + L_k^+ (y_k - \hat{y}_k^-)$  \(\triangleright\) State estimate measurement update
26. $P_{\hat{x},k}^+ = P_{\hat{x},k}^- - L_k^+ P_{\hat{y},k}^- (L_k^+)^T$  \(\triangleright\) State error covariance measurement update
27. $\hat{\theta}_k^+ = \hat{\theta}_k^- + L_k^+ (y_k - \hat{d}_k^-)$  \(\triangleright\) Parameter estimate measurement update
28. $P_{\hat{\theta},k}^+ = P_{\hat{\theta},k}^- - L_k^+ P_{\hat{d},k}^- (L_k^+)^T$  \(\triangleright\) Parameter error covar. measurement update
29. end for
References


References


References


