NTNU

Norwegian University of
Science and Technology

Exploration of Nonlinearities in a Lithium Ion Battery Cell Model and their impact on Particle Filters and Nonlinear Kalman Filters for State of Charge Estimation

Kristian Eggereide Roaldsnes

Master of Science in Cybernetics and Robotics
Submission date: June 2018
Supervisor: Marta Maria Cabrera Molinas, ITK

Norwegian University of Science and Technology
Department of Engineering Cybernetics

Problem description

The State of Charge is an important state in a lithium ion battery cell, and its determination remain an active research topic. Researchers have applied various Bayesian Kalman Filters to this problem, but the Kalman Filter is highly dependent on a set of key assumptions. The Particle Filter is a less restrictive Bayesian filter that might provide advantages under certain conditions and assumptions. This thesis will explore the nonlinearities presented by a Lithium ion battery system and the impact these nonlinearities have on Particle Filters and Nonlinear Kalman Filters for estimating the state of charge in a lithium ion battery cell.

Abstract

Lithium-ion batteries are currently being used for a wide variety of applications across different industries due to their many advantages. However, lithium batteries are also sensitive to their operating conditions, and present unique challenges with regards to safety. Accurately knowing the State of Charge of batteries can therefore significantly aid in ensuring safe operation and optimal usage of stored energy.

Over the course of the last 20 years, different solutions have been proposed for State of Charge estimation, among them a variety of Nonlinear Kalman Filters. Kalman Filters are accurate and simple to implement, but require that all likelihood densities in the system can be approximated to be Gaussian. A slightly different approach is the Particle Filter, which uses randomly sampled state vectors (particles) to represent likelihood functions. Particle Filters can theoretically work with any likelihood density provided enough particles.

In this thesis, a nonlinear Kalman Filter and three different Particle Filter variants are compared for use in the Lithium-ion battery State of Charge estimation problem. A current load curve extracted from a Formula Student electric Race car forms the basis for the test case. The model is augmented to include the parameters for estimation and the chosen estimators are the Joint Central Difference Kalman Filter, the Joint Bootstrap Particle Filter, the Joint Imprecise Bootstrap Particle Filter and the Joint Auxiliary Particle Filter.

A Monte Carlo based system exploration method is suggested and applied to the Joint estimation problem to assess the amount of nonlinearity caused by the model. A combination of Gaussian distributions are passed through the model and the distortion is evaluated by estimating the resulting skew and kurtosis. The nonlinearity of the system is evaluated at different States of Charge and at different levels of uncertainty. The nonlinear equations of the augmented 2RC cell model is seen to distort the distributions quite significantly for State of Charge values below 12% and at 85%.

The Particle Filters and the Kalman filter perform very similarly. For State of Charge estimation, all the filters produce average RMS errors of between 0.87% and 1.16%, with the Joint Bootstrap Filter producing the lowest average error at 0.87% and the Joint Imprecise Bootstrap Particle Filter producing the highest average error at 1.16%. Particle Filters and Kalman Filter also perform similarly in voltage prediction, with all the filters producing approximately 10mV of average RMS voltage error.

The thesis concludes that Particle Filters or other Monte Carlo based methods might have merit in joint parameter estimation applications, and suggests that this is further investigated in the future.

Sammendrag

Litium-ion batterier benyttes til en rekke applikasjoner på tvers av mange industrier verden over, blant annet på grunn av sine mange gode egenskaper. Desverre er Litium ion-batterier også svært sensitive for operasjonsbetingelsene de blir utsatt for, og dette medfører en rekke utfordringer med tanke på å sikre trygg bruk. Dersom man er i stand til å estimere batteriets prosentvise restkapasitet (SoC) nøyaktig, kan dette være til stor hjelp for å sikre trygg og optimal utnyttelse av lagret energi.

De siste 20 årene har en rekke løsninger blitt foreslått for å estimere SoC, blandt dem ulineære Kalmanfilter. Kalmanfiltre er nøyaktige og enkle å implementere, men krever i utgangspunktet at alle sannsynlighetsfordelinger kan antas å være gaussisk fordelt. En noen annerledes løsning er Partikkelfilteret, som benytter en ansamling av tilfeldig genererte tilstandsvektorer (partikler) for å representere sannsynlighetsfordelinger. I teorien betyr dette at partikkelfilteret kan tilnærme en hvilken som helst sannsynlighetsfordeling, gitt nok partikler.

I denne masteroppgaven sammnlignes et ulineært kalmanfilter og tre varianter av partikkelfilter til bruk i SoC-estimering. En lastkurve hentet fra en elektrisk Formula Student-bil danner grunnlaget for testingen. Cellemodellen augmenteres med sine parametre for slik å kunne brukes til parameterestimering, og de valgte filterene er eit Joint Central Difference kalmanfilter, ett Joint Bootstrap partikkelfilter, ett Joint Imprecise Bootstrap partikkelfilter og ett Joint Auxiliary partikkelfilter.

En Monte Carlo-basert fremgangsmåte benyttes for å evaluere hvor mye ulinearitet som er tilstede i cellemodellen. En kombinasjon av ulike gaussiske sannsynlighetsfordelinger projiseres gjennom modellen og den resulterende forvrengingen evalueres ved å estimere verdier for skjevhet og kurtosis. Metodikken benyttes på den augmenterte cellemodellen ved ulike SoC verdier. Den ulineære systemmodellen gir tydelig forvrengning for noen SoC-tilstander, og da spesielt for SoC under 12% og i området rundt 85% SoC.

Partikkelfiltrene og Kalmanfiltret presterer meget likt. I SoC-estimering produserer alle filtrene gjennomsnittlig effektiv feil mellom 0.87% og 1.16%. Bootstrap-filteret har den lavest gjennomsnittsfeilen, med 0.87% mens Imprecise Bootstrap-filteret har den høyeste på 1.16%. I spenningsprediksjon har partikkelfiltrene og kalmanfilteret også lignende resultater, med en gjennomsnittlig effektiv feilverdi på rundt 10mV.

Denne oppgaven konkluderer med at bruk av partikkelfiltre eller andre Monte Carlo-metoder kan være fordelaktig ved parameterestimering, og anbefaler at dette utforskes videre.

Preface

I first came into contact with the field of State of Charge estimation as a part of my three-year participation in the Revolve NTNU Formula Student project. In 2015 I was tasked with developing a Battery Management System for use in a high-power electric vehicle, and in 2016 I was in charge of the development of the full accumulator. This threw me head-first into a fascinating and challenging engineering field in which safety, cost and performance is key. In terms of State of Charge estimation, the team had to settle for simple solutions such as the well known and much used Coulomb Counting technique, a solution which left much to be desired. Not being able to compensate for aging or temperature, the chosen solutions were safe but generally too conservative.

In 2016, Revolve contacted Professor Marta Molinas at NTNU to discuss the possibilities of writing a masters thesis on the subject. Subsequently, in 2017 Ørjan Gjengedal wrote a masters thesis in collaboration with Revolve NTNU about using Kalman filters for the purposes of estimating impedance and State of Charge on-line. I then became involved in the fall of 2017 as an effort to explore the possibility of using the Particle instead of the Kalman Filter. To this end, I wrote a technical report in the Fall of 2017 on a comparison between a simple Bootstrap Particle Filter and a Central Difference Kalman Filter. The results from the report indicated that there might be an advantage to using the Particle Filter in certain situations, and I therefore chose to extend the work into this Master Thesis. In order to keep the focus towards the theoretical aspects of the problem, I decided not to collaborate directly with Revolve. Still, the thesis is written mainly with the race car application in mind. This is particularly reflected in the use of load profiles extracted from Revolve field data.

My main supporter in this work has been my supervisor Marta Molinas, with whom I have had monthly meetings throughout the semester. She has given me freedom to control the direction of my thesis in a way that I have seen fit, and her insights has helped me greatly. I have also had several discussions with Ørjan Gjengedal which has helped further my understanding of both the Particle Filter and the Kalman Filter methods. Ørjan has also graciously provided the data sets that have been used throughout this work. Senior Scientist Dr. Preben Johannes Svela Vie at the Institute of Energy Technology (IFE) in Lillestrøm has also shown interest in my work and provided some enlightening discussions.

As a part of this thesis I spent two weeks at IFE where I was given access to a work station and testing facilities. The original plan was to conduct low-temperature cell tests, but due to scheduling issues and time constraints I was not able to complete them. This led to a change of focus where the room-temperature tests conducted by Ørjan became the main test case.

The thesis work has been conducted on computer equipment provided by NTNU, and all the software has been developed in MATLAB R2017b using a MATLAB Academic Licence provided by NTNU. All the references given in this thesis were chosen by me.

Acknowledgements

I would like to thank Professor Marta Molinas for her invaluable help and enlightening discussions throughout this past semester and year. She has given me great freedom to explore, powerful insight through our discussions and words of encouragement when I have had doubts. I would also like to thank Ørjan Gjengedal for our discussions and for him allowing me to use his data sets. His deep understanding, thoughtful comments and humble demeanor has been both a motivation and a great help for me. I am also grateful for the sincere interest in my work shown by Dr. Preben Vie, and the hospitality shown by both him and IFE in allowing me to stay there for a two-week period in March.

Finally, I want to express my deep gratitude to my partner Kristine who have supported me in so many ways through my final years at NTNU.

Kristian Eggereide Roaldsnes

Trondheim, June 2018

List of abbreviations

APF Auxiliary Particle Filter.

BMS Battery Management System.
BPF Bootstrap Particle Filter.

CC Coulomb Counting.
CC/CV Constant Current / Constant Voltage.
CDF Cumulative Density Function.
CDKF Central Difference Kalman Filter.
CKF Cubature Kalman Filter.

ECM Equivalent Circuit Model.
EKF Extended Kalman Filter.
EMF Electromotive Force.

FS Formula Student.

HMM Hidden Markov Model.

JAPF Joint Auxiliary Particle Filter.
JBPF Joint Bootstrap Particle Filter.
JCDKF Joint Central Difference Kalman Filter.
JIBPF Joint Imprecise Bootstrap Particle Filter.

KF Kalman Filter.

LCO Lithium Cobalt Oxide.
LiBs Lithium Ion Batteries.
LKF Linear Kalman Filter.

MC Monte Carlo.

MCMC Markov Chain Monte Carlo.

OCV Open Circuit Voltage.

PDF Probability Density Function.
PF Particle Filter.

SMC Sequential Monte Carlo.
SoC State of Charge.
SPKF Sigma-Point Kalman Filter.

UKF Unscented Kalman Filter.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Lithium Ion Batteries (LiBs) were first introduced commercially in the early 1990s [1] and have since become a critical part of many electric power systems. Their popularity can be ascribed to the many advantages LiBs enjoy compared to previous generations of battery technology such as Lead-acid and Nickel Manganese Hydrate. Among other things, LiBs have significantly higher volumetric and gravimetric energy densities, lower internal impedance, no memory effect and longer life spans than other technologies [1], [2].

However, LiBs also presents unique challenges that must be overcome in order to fully exploit these advantages. LiBs can be irreversibly damaged and/or become unstable if subjected to high temperatures or if the voltage of the cells are not kept within safe operating boundaries [3]. To mitigate the risks, battery systems are typically outfitted with a monitoring circuit, often referred to as a Battery Management System (BMS). The BMS continuously supervises and monitors the cells, and can also be responsible for estimating various important metrics of the cells, such as the State of Charge (SoC)[1].

The SoC is the *relative remaining charge* in an individual battery cell. Since it is relative to the maximum charge the cell can hold at its present age, it provides an intuitive measure of remaining charge (i.e. a cell with a SoC of 50% always has half of its maximum charge left regardless of what its actual capacity is in terms of Ah).

1.1 Motivation

The Formula Student (FS) competition is an international student competition which tasks teams of students with building and racing a full size race car. A growing interest in electric vehicles has led to the inclusion of electric vehicles in the competition and currently hundreds of teams around the world develop electric race cars for the competition. ”Revolve NTNU” is such a

[1] BMSs can be further subdivided into categories based on the level of autonomy and specific safety functions provided by the system (such as current shutdown). This will not be discussed in any detail here.

student team based in Trondheim, Norway. Revolve NTNU has built electric vehicles since 2014, and remain active to date.

Knowing the SoC accurately is important in more or less all applications, but is particularly critical in a high-performance application where the battery can be expected to be depleted (or nearly depleted) as a part of its normal operation or in the case of an emergency. This is certainly the case for the electric vehicles made by Revolve. In this context, accurately estimating the cell behaviour is critical for maximizing performance and safety. More generally, knowing the SoC accurately also allow for predictable operation of ships, drones and other electric vessels and reduces range anxiety in commercial electric cars. Accurate knowledge of the SoC can also form the basis for an accurate Available Power estimation [4] which can allow applications to extract the maximum charge left in the cell without overdischarging the cell. Available power can also be used to calculate the maximum charging current allowed in the case of regenerative braking.

1.2 Problem description

Unfortunately, there is no direct way to measure the SoC when the cell is electrically loaded. The only measurements that are realistically available to a BMS are the current, the terminal voltage and the exterior temperature of the cell. While there is a (more or less fixed) relationship between the at-rest voltage of the battery cell and the SoC, chemical interactions give rise to parasitic voltages that obscure this relationship both during and after electrical current has passed through the cell.

Many techniques have been proposed for solving this challenge [5], ranging from the very simple Coulomb Counting (CC) to model based current interruption observation schemes [6] to complex nonlinear Kalman Filters with parameter estimation [7]–[12] and Particle Filters (PFs) [13], [14]. Simple methods such as CC, where the measured current is integration from an assumed starting SoC, are widely used in the industry. This is due to their simplicity and predictability, even with the obvious shortcomings. For instance, the CC method can be very accurate for shorter periods of time, but only under the assumption that it can be initialized correctly. CC also lacks robustness since it is unable to compensate for bad initializations, and it cannot discover and handle loss of sensors or sensor bias.

The main focus of this thesis is examining the major differences between using Kalman Filters and Particle Filters for live estimation of the SoC in a battery under heavy, dynamic load profiles. The main hypothesis is that the nonlinearities in a battery system is non-negligible for low SoC. Thus, the assumptions made by the nonlinear Kalman Filters (KFs) restrict the accuracy in these regions and PF are more suited if higher accuracy is desired in these regions.

Other authors have previously compared the two classes of filters in the context of LiBs but to the best knowledge of this author, little literature is available that examines and/or verifies the

assumptions needed. Plett [11] states briefly that the consensus of literature and experience is that the Gaussian assumption on the noise matrices work well, but provide no citations for this claim. In a later paper Plett concedes that PFs might be necessary to increase accuracy [10], but at a high cost in terms of computational complexity. Bhuvana et al [15] presents a direct comparison of an Extended Kalman Filter (EKF), an Unscented Kalman Filter (UKF), a Cubature Kalman Filter (CKF) and a simple Bootstrap Particle Filter (BPF). While the BPF is shown to be slightly more accurate than the various Kalman Filters (again at a high computational cost), the causes for this result is not explained to any degree of details.

Available computational power is continuously increasing, and at some point it might be feasible to consider PFs if the specific application show enough potential gain. For a designer to select which scheme to ultimately apply to a given problem, it is necessary to have broad knowledge of the advantages and disadvantages of the available methods. To this end, the contributions of this thesis is two-fold:

1. An exploration of the nonlinearities present in the typical SoC estimation problem for a Lithium Battery. The system is explored to more clearly expose for what real-life situations Sequential Monte Carlo (SMC) methods might be appropriate despite their presumed high computational cost.

2. An exploration of a selection of SMC methods applied to the SoC estimation problem. The Joint Bootstrap Particle Filter (JBPF), Joint Auxiliary Particle Filter (JAPF) and Joint Imprecise Bootstrap Particle Filter (JIBPF) are developed and compared to a Joint Central Difference Kalman Filter (JCDKF). The estimators are tested on a realistic load cycle from the Revolve NTNU 2016 car 'Gnist'.

The end goal of this thesis is to determine whether or using the PFs can significantly help in the estimation of SoC or cell parameters.

1.3 Organization of the thesis

The material is organized as follows. Chapter 1 (Introduction) discusses the historic backdrop, motivation for the topic and contributions of the thesis. Chapter 2, (Basics of noise description and distributions) provide a basic review of probability theory and noise description methods. Methods for discretely representing continuous probability densities and estimating continuous densities from discrete points are also discussed briefly. Chapter 3, (System modelling) give a brief background on the behaviour of battery cells and the methods commonly used to model the behaviour. Chapter 4 (Estimation) discusses the theoretical background of the employed estimation methods and presents algorithms for each of the methods used later. Chapter 5 (Methods), presents the implemented methods of the thesis. First the cell type, test setup, model

and algorithms for calculating reference SoC and starting parameters are explained. Secondly the implemented estimators are presented with tuning settings. Finally the method employed for exploring the cell model nonlinearity and evaluating the adequacy of the Gaussian assumption made by the KF is explained in detail. Chapter 6 (Results of the study) chapter presents the results achieved and Chapter 7 (Discussion) provides an extensive analysis. Finally Chapter 8 suggests future work and Chapter 9 presents the final conclusion to the thesis.

Chapter 2

Basics of noise description and distributions

As this thesis is primarily aimed at readers with only a rudimentary understanding of probability theory, the following section will aim to introduce the necessary fundamentals to appreciate the later discussions on SMC methods and likelihood distributions.

2.1 Random variables and probability density functions

A *random variable* is a variable that can potentially take on many different values when it is assigned. The relative *likelihood* for the possible values it can take on are specified with a Probability Density Function (PDF). The relationship between the random variable and its PDF is described mathematically as

$$X \sim p(\cdot) \tag{2.1}$$

where X is a random variable, $p(\cdot)$ represents an arbitrary PDF possibly dependent on a set of parameters and $\sim$ means "distributed according to". Random variables are typically distinguished using upper-case letters. When mathematically manipulating random variables the output is not a value, but a new random variable distributed according to a different distribution. Drawing a value from the distribution is called *sampling* and when a value is sampled from the distribution, the resulting variable value is a *realization* of the random variable, often referred to as a *sample*. When drawing more than one samples from a PDF, the individual samples are denoted as $[X^1, ..., X^N]$ where N is the total number of samples.

Figure 2.1 shows a plot of a simple normal distribution which will be discussed in more detail below. The intuition of the distribution representation is simply that the higher the value on the y-axis the more likely the values along the x-axis are. Thus, when drawing samples from

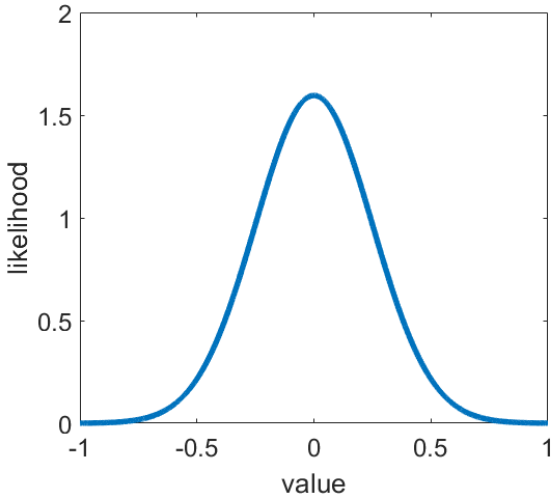


Figure 2.1: Example normal distribution with $\sigma = 0.25, \mu = 0$

this specific example distribution most of the samples returned will have values close to 0, and few samples will have values less than -0.5 or greater than 0.5.

2.2 The normal distribution

An important and often used distribution is the *Normal* distribution, often referred to as the *Gaussian* distribution[1]. The Normal distribution is important because it is often observed in various natural processes and also has mathematical properties that make it well suited for many applications (e.g. filtering). The assumption of Gaussian distributed noise is key to the derivation of the Kalman Filter discussed in Section 4.3.

The Normal distribution is symmetric and is fully specified by just two parameters, the *mean* μ and *variance* σ^2. The density function is evaluated as

$$\mathcal{N}(x; \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \tag{2.2}$$

and when specifying a distribution the shorthand version

$$X \sim \mathcal{N}(\mu, \sigma) \tag{2.3}$$

implies that X is distributed according to a Normal distribution of μ mean and σ^2. The mean determines the center of the distribution, while the variance determines the width. The square root of the variance is called the *standard deviation* σ, which is a more intuitive measure of the width of the distribution due to the so-called '68-95-99.7' rule [16]. For the Normal distribution, approximately 68% of the likelihood is collected in the interval $\mu \pm \sigma$. In other words, if samples

[1] In this thesis, Normal and Gaussian are used interchangeably.

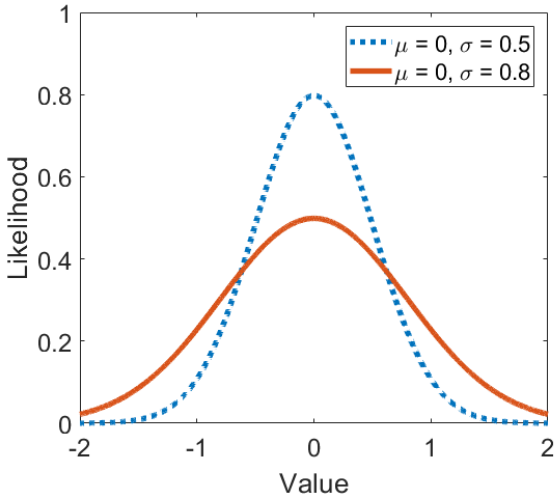


Figure 2.2: Example normal distributions

If $[X_1 ... X_n]$ are normally distributed random variables with mean μ_i and variance σ_i^2 and

$$Y = a_1 X_1 + a_2 X_2 + ... + a_n X_n \tag{2.4}$$

Then the resulting random variable Y has mean:

$$\mu_Y = a_1 \mu_1 + a_2 \mu_2 + ... + a_n \mu_n \tag{2.5}$$

and variance

$$\sigma_Y^2 = a_1^2 \sigma_1^2 + a_2^2 \sigma_2^2 + + ... a_n^2 \sigma_x^2 \tag{2.6}$$

Table 2.1: The reproductive property of the Normal distribution, Theorem 7.11 of [17]

are taken from a Normal distribution $X \sim \mathcal{N}(\mu, \sigma)$, approx. 68% of the samples will fall within the so-called 1σ band. Similarly, approx 95% of the samples will fall within the 2σ band and 99.7% within the 3σ band. Two example Normal distributions are shown in Figure 2.2 with equal μ but different standard deviations.

The Normal distribution exhibits the *reproductive property* [17, p.221], summarized in Table 2.1. This property simply states that linear combinations of Normally distributed random variables are also Normally distributed.

2.3 Conditional likelihood and Bayes Rule

The notation $p(\cdot|\cdot)$ is used to denote *conditional likelihood*, that is the distribution $p(X|Y)$ is the distribution of X, given some knowledge about Y. Conversely, $p(Y|X)$ is the distribution of Y given some knowledge about X.

Bayes rule provides a way to link the conditional likelihoods, and is essential to the formulation of Bayesian inference, which will be discussed in Chapter 4.2. Bayes rule is given in Equation (2.7)

$$p(X|Y) = \frac{p(Y|X)p(X)}{p(Y)} \tag{2.7}$$

2.4 PDF estimation

Frequently, information from a process is only available as discrete samples. When the samples are realizations of a random process, it might be necessary to form estimates of the underlying distribution. In some cases it is possible to assume that the samples originate from a specific distribution and estimating the distribution becomes a matter of determining the parameters of the distribution. In other cases it is not possible to reliably assume the shape of the distribution, and then it is necessary to employ some form of nonparametric PDF estimation algorithm. Two popular methods for this are the Histogram estimator and the Kernel estimator.

Histogram methods

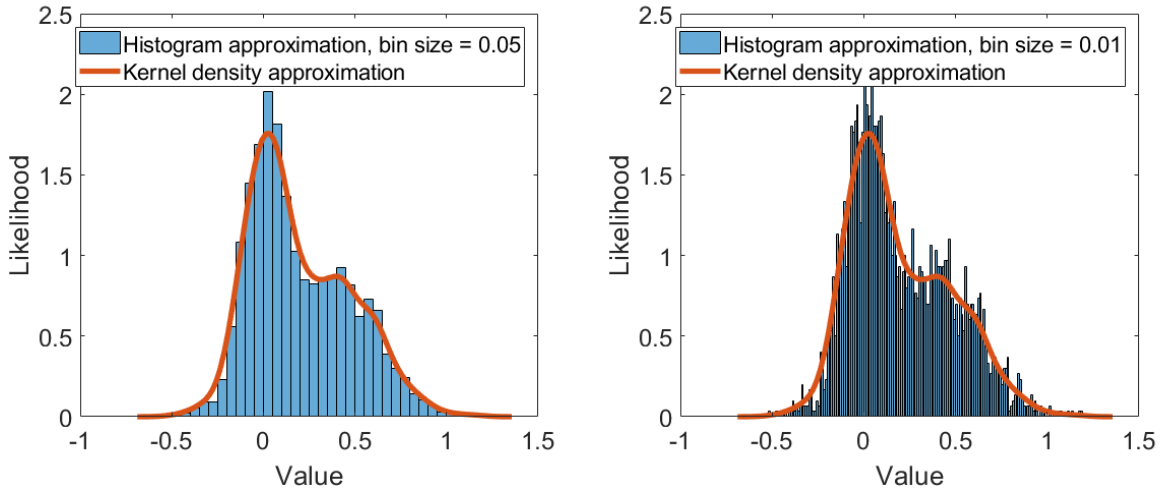
The most popular, and arguably the simplest, method for estimating a likelihood distribution is the Histogram method [18]. A series of *bins* are selected, typically of uniform width. The histogram method then simply counts the number of samples that fall in each of the bins, and the counts are normalized so that the total area of the distribution curve is 1.

While simple to implement and intuitive, the Histogram method has its flaws. A careful balance has to be kept between the number of samples used and the size of the bins to avoid noisy graphs and it can be difficult to spot the most important features (such as bimodality). It also produces a discontinuous function, which might be undesirable.

Kernel Estimator

To improve the quality of the estimated PDF, a popular method is the *Kernel Estimator* method [18], [19]. The bins are replaced by individual *Kernels*, often chosen to be a Normal distribution. The final distribution is simply the normalized sum of many such Kernel likelihoods. This provides a much smoother PDF estimate.

In Figure 2.3, a comparison between the Kernel Estimator and Histogram outputs are shown for 3000 samples taken from a multimodal Normal distribution. The Histogram bin size is set to 0.05 and 0.01. The Kernel Estimator and the Histogram reveal the same features, but note that the smooth curve of the Kernel estimate makes it a much more intuitively appealing representation.



(a) Histogram bin size = 0.05

(b) Histogram bin size = 0.01

Figure 2.3: Kernel Estimator and Histogram outputs for a set of N=3000 samples from a multimodal Normal distribution.

2.5 Skewness and Kurtosis

It might be enlightening to calculate measures of shape of the distribution at hand. Two useful parameters for this purpose are the *skewness* γ and the *kurtosis* κ.

The skewness can be defined in different ways, in this thesis the definition used by Owen [19] is adopted:

$$\gamma = \frac{E((X - E(X))^3}{E((X - E(X))^2)^{3/2}} \tag{2.8}$$

Similarly, the definition of kurtosis is also borrowed from Owen:

$$\kappa = \frac{E((X - E(X))^4)}{E((X - E(X))^2)^2} - 3 \tag{2.9}$$

where $E(X)$ is the *expected value* of X.

The skewness parameter is a measure of symmetry. If one tail is much longer than the other, the long side is said to be a *heavy* tail, and the PDF is said to be skewed. A Normal distribution will have $\gamma = 0$. The kurtosis expresses how narrow a distribution is, compared to a Normal distribution. If the tails fall faster (making a narrow distribution) than the tails of a normal distribution, $\kappa < 0$, and if the tails fall slower than a Normal distribution, $\kappa > 0$. As a result estimates of these two measures can be used to evaluate aspects of a likelihood distribution, even if these two parameters alone do not provide a full description of the similarity to a Normal distribution. In Figure 2.4 an arbitrary distribution is shown together with a Normal distribution for comparison purposes. For the distribution shown, the skewness and kurtosis are estimated to be $\hat{\gamma} = 0.699$ and $\hat{\kappa} = 1.3286$, respectively.

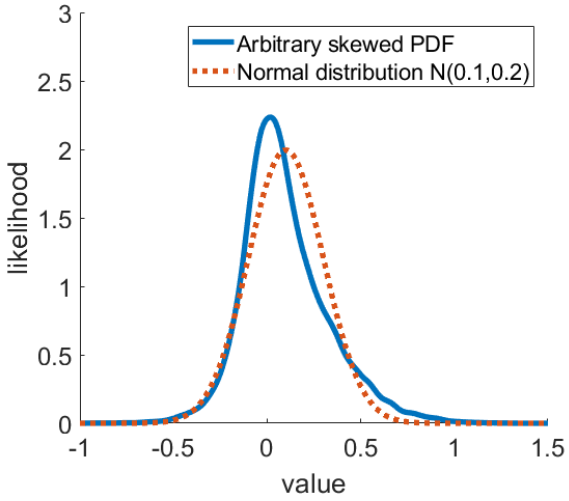


Figure 2.4: Example distribution with skew and kurtosis $\neq 0$. A normal distribution with $\mathcal{N}(0.1, 0.2)$ is plotted for comparison. $\hat{\gamma} = 0.699$ and $\hat{\kappa} = 1.3286$

2.6 Nonparametric confidence intervals for skewness and kurtosis - the Bootstrap method

When only discrete samples of an unknown distribution are available (as is the case in the example given in Section 2.5), calculation of skew and kurtosis by Equation (2.8)-(2.9) can only return estimates. In other words the sample set available will directly impact the calculated value, and even with a very high number of samples the estimated mean is unlikely to match the exact underlying (true) mean of the distribution from which the samples were drawn. Practically, this means that naively calculating the parameters directly from the observed data is not sufficient to make assertions about the true value of the parameters. In fact, it is necessary to compute confidence intervals of the parameter values in order to assert the degree of trust to put in the value calculated. While the arithmetic mean follows the Central Limit Theorem [17, p.234] and thus the error of the mean estimate is approximately Normally distributed for a high number of observed samples ($n \geq 30$), this is not necessarily the case for other parameters such as the skew and kurtosis.

If no model can reasonably be assumed for the distribution of the parameter of interest, *non-parametric* methods can be applied such as the Bootstrap method [20] or empirical likelihood [19]. For the purposes of this thesis the Bootstrap method (hereafter referred to as simply the Bootstrap, not to be confused with the Bootstrap Particle Filter) has been employed.

The Bootstrap is a numerical method for evaluating the distribution of some parameter of a distribution (e.g. the mean, median or skew.), given a limited number of samples from a population. The key idea is that the Bootstrap assumes that the samples given are the best possible representation of the population that can be made available, and simply uses the samples as a

representative of the distribution [20]. The Bootstrap then samples N times with replacement from the distribution formed by the samples to form a new set of samples. The parameter in question is then calculated for the new set of sampled values. This process is repeated N_{bs} times, resulting in a set of N_{bs} values for the parameter. Using PDF estimation on this set (e.g. the histogram method) yields an approximation of the distribution of the parameter.

MATLAB provides an implementation of the Bootstrap method, called bootstrp(·) for ordinary Bootstrap sampling. MATLAB also provides a function bootci(·) for calculating 95% confidence bounds with the Bootstrap method.

2.7 Monte Carlo approximation to complex PDFs

When handling PDFs that cannot easily or accurately be described using parametric likelihood functions (e.g. the Normal distribution), analytical computations and estimation might quickly become unfeasible. One way of handling this is introducing numerical methods for representing the likelihood distributions. Monte Carlo methods are a powerful set of tools that can be applied to learn the behaviour of complex systems [21]. Consider a distribution $p(\cdot)$ of an arbitrary shape. Assuming that the distribution can be directly sampled from[2], N samples can be taken so that $X^{(i)} \sim p(\cdot),\ \text{for } i = 1...N$. $p(\cdot)$ can then be approximated as

$$\hat{p}(\cdot) = \frac{1}{N}\sum_{i=1}^{N} \delta_{X^{(i)}}(dx) \tag{2.10}$$

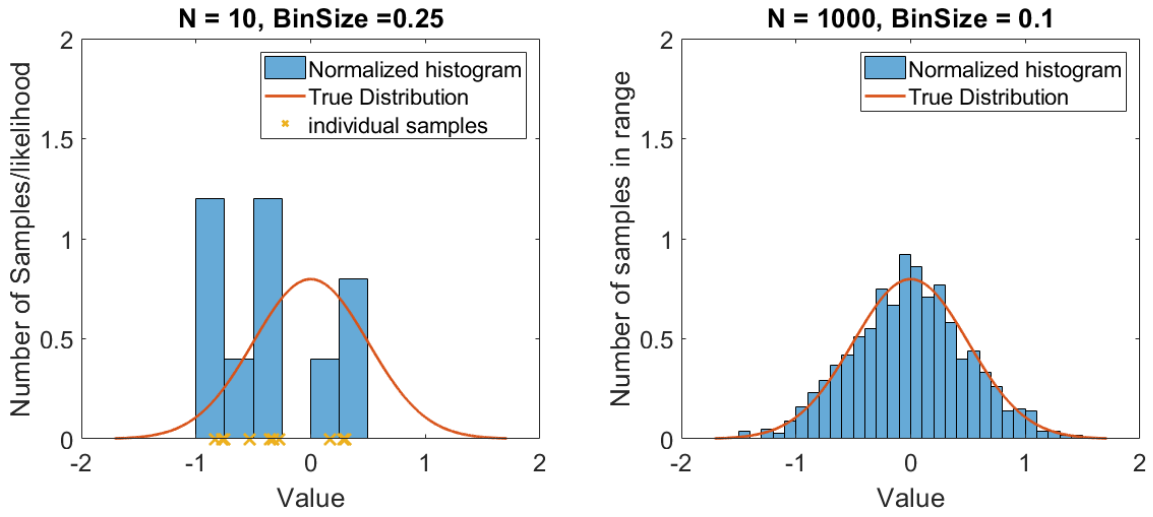
where

$$\delta_x(A) = \mathbb{1}_A(x) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{otherwise} \end{cases} \tag{2.11}$$

is a Dirac Delta as discussed by Tulsyan et al [21]. In other words, the MC method represents the likelihood in an interval by the number of equally weighted samples in the interval, divided by the total number of particles. The likelihood in an interval is determined by the number of particles present in the interval. In Figure 2.5, two histograms[3] of particles representing the distribution ($p(\cdot) = \mathcal{N}(0, 0.5^2)$), are shown for $N = 10$ and $N = 1000$. It is clear from this simple example that having more particles will lead to a more accurate result.

[2]Note that this usually is not the case when the distribution is nontrivial. Special techniques are required to circumvent this problem, as discussed later

[3]Note that the histograms shown have been normalized so that the total area of the histogram is equal to 1, for simple comparison with the generating distribution. The weighting of 1/N in the Monte Carlo (MC) method means that the area of the likelihood distribution is not necessarily equal to 1.



(a) Low number of samples

(b) Higher number of samples

Figure 2.5: Representing a likelihood using N unweighted samples. In a) the individual samples are shown as crosses on the x-axis. Both distributions are Normal distributions with $\mu = 0$ and $\sigma = 0.5$.

2.8 Cumulative Density Functions and confidence intervals

The Cumulative Density Function (CDF) is the cumulative (integrated or summed) probability for the density function, defined for the continuous case as [17, p. 85]

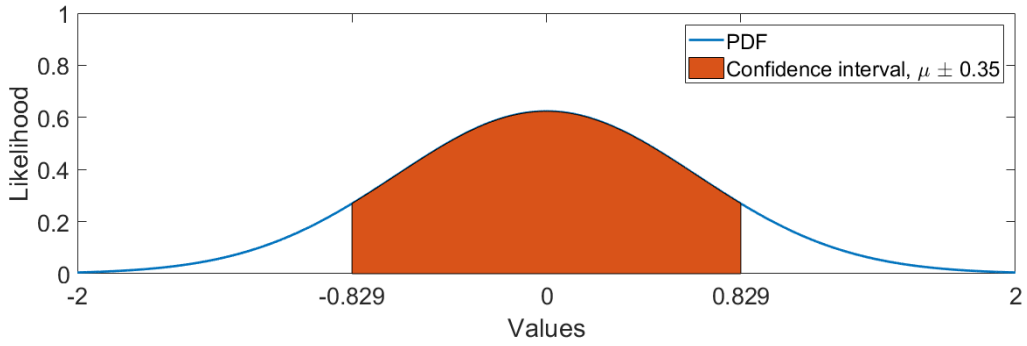
$$CDF(x) = P(X < x) = \int_{-\infty}^{x} p(t)dt \tag{2.12}$$

The CDF simply expresses the probability that a sample from the distribution is below x. By evaluating the inverse CDF, one can compute the corresponding *confidence intervals*. In the context of state estimation, a confidence interval can be used to express the interval required for a certain confidence. As an example, an estimator can provide a $\mathcal{C}\%$ confidence interval by computing

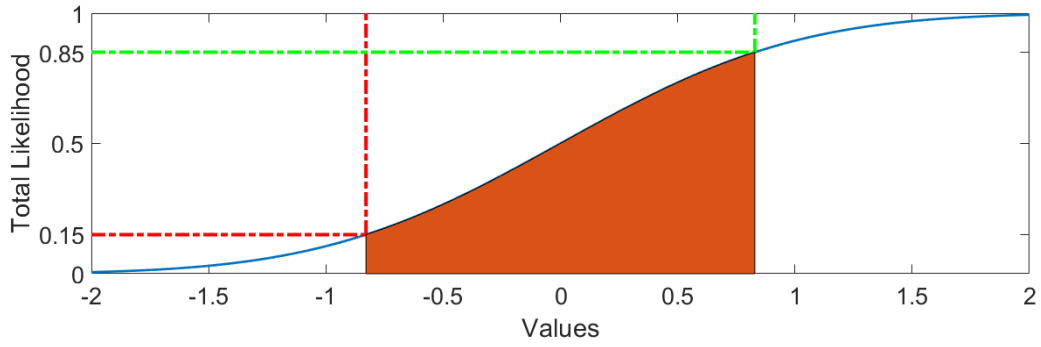
$$\hat{x}_{\mathcal{C}\%,\text{upper}} = CDF^{-1}(0.5 + \frac{1}{2}\frac{\mathcal{C}}{100}) \tag{2.13}$$

$$\hat{x}_{\mathcal{C}\%,\text{lower}} = CDF^{-1}(0.5 - \frac{1}{2}\frac{\mathcal{C}}{100}) \tag{2.14}$$

In Figure 2.6 the relationship between the PDF and the CDF is illustrated with the $\mathcal{C} = 70\%$ interval marked. If $\mu = 0$ is a state estimate produced by an estimator, the true value of the process is expected to lie within the $\mathcal{C} = 70\%$ confidence interval of between -0.829 to 0.829 with a likelihood of 0.7. Consequently, increasing the confidence level will give wider intervals but also increases the chance that the correct value lies in the specified interval. The method outlined here implies that all confidence level values are chosen to be symmetric around the



(a) PDF



(b) CDF

Figure 2.6: PDF and CDF for an example continuous Normal distribution

mean. This does not imply that the confidence **intervals** must be symmetric however. While this is the case for the Normal distribution, it is not necessarily true for other distributions.

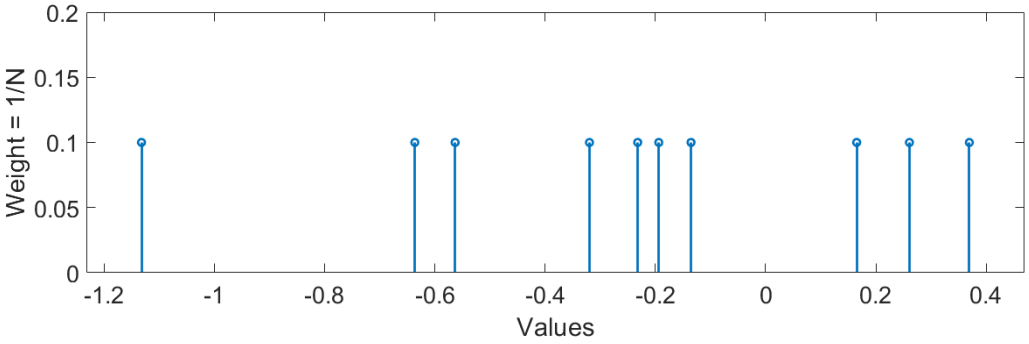
For the MC case, the CDF can be defined as [21]:

$$CDF(x) = P(X \leq x) = \frac{1}{N} \sum_{i=1}^{N} \mathbb{1}_{(-\inf, x]}(X^{(i)}) \tag{2.15}$$

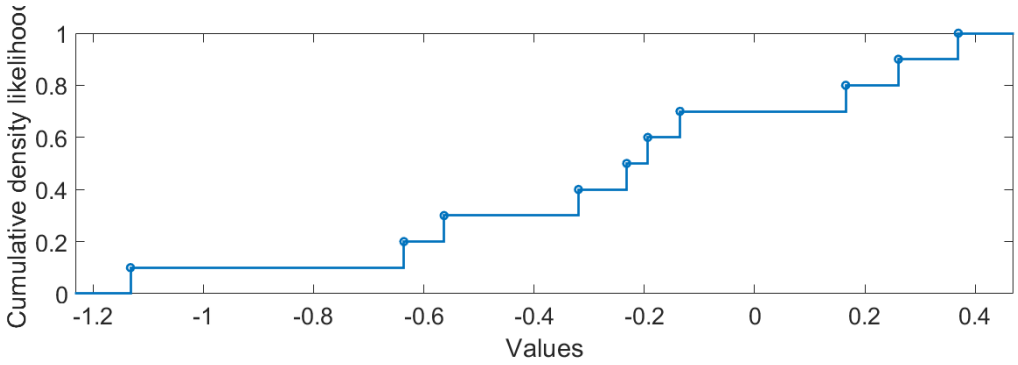
in the unweighted case and

$$CDF(x) = P(X \leq x) = \frac{1}{N} \sum_{i=1}^{N} w^{(i)} \mathbb{1}_{(-\inf, x]}(X^{(i)}) \tag{2.16}$$

in the importance sampling (weighted) case. Importance sampling will be discussed in more detail in section 4.4. The calculation of confidence intervals is similar to the continuous case presented above, but note that a strategy must be selected for handling the case when a confidence level is selected that does not directly correspond to a sample in the MC set. One possible strategy is to simply linearly interpolate between the samples. In Figure 2.7 an example CDF constructed from a set of Normally distributed values are shown.



(a) Stem plot of values with uniform weight



(b) CDF plot

Figure 2.7: Constructing the CDF from a set of N = 10 values distributed according to $\mathcal{N}(0, 0.5)$

Chapter 3

System Modelling

In this chapter the basic principles of battery cell modelling is briefly presented. First a short description of LiBss in general will be presented. Emphasis is put on the major effects observed in cell behaviour under load. Secondly, the Electrochemical and Equivalent Circuit Model (ECM) cell modelling approaches are discussed. Since the main focus of this thesis is not on cell modelling, the level of detail is deliberately kept low and emphasis is put on presenting the main features of current cell modelling difficulties and solutions.

3.1 Lithium Ion Batteries

LiBs are electrochemical energy storage units, consisting of an electrically insulating ion conductor sandwiched between a positive (cathode) and a negative (anode) active material [22]. Li^+ ions swing back and forth between the anodes through the ion conductor [23]. The insulating property of the ion conductor mean that while the lithium ions can pass more or less freely between the electrodes, the electrons are forced to pass through a closed external circuit - generating an electric current. The overall chemical reaction is shown for the case of a Lithium Cobalt Oxide (LCO) battery in equation (3.1).

$$LiCoO_2 + 6C \leftrightarrow Li_{1-x}CoO_2 + Li_xC_6 \tag{3.1}$$

3.2 State of Charge

The State of Charge (SoC) is defined as the remaining charge in the battery cell divided by the maximum charge available when the cell is fully charged. This can easily be reformulated to a definition which involves the amount of charge moved in or out of the cell:

$$SoC = \frac{Q_{\text{remaining}}}{Q_{\text{max}}} = \frac{Q_{\text{start}} - Q_{\text{discharged}}}{Q_{\text{max}}} \tag{3.2}$$

where Q_{start} is the (assumed known) charge in the cell at some starting time t_0. Provided perfect knowledge of the current, the discharged current can be expressed as an integral,

$$Q_{\text{discharged}} = \int_{t_0}^{t_1} I(t)\, dt \tag{3.3}$$

which leads to the ideal time-dependent definition of SoC given in equation (3.4)

$$SoC(t_1) = SoC(t_0) - \frac{1}{Q_{\text{max}}} \int_{t_0}^{t_1} I(t)\, dt \tag{3.4}$$

As discussed by Julien et al. [24], various unwanted side effects give rise to small inefficiencies, expressed in total as the *coloumbic* efficiency η, which can be defined as

$$\eta = \begin{cases} \frac{Q_{\text{discharge}}}{Q_{\text{charge}}} & I_{\text{cell}} < 0 \\ 1 & \text{otherwise} \end{cases} \tag{3.5}$$

Including η into the definition from Equation (3.4), we get the alternative SoC definition:

$$SoC(t_1) = SoC(t_0) - \frac{1}{Q_{\text{max}}} \int_{t_0}^{t_1} \eta I(t)\, dt \tag{3.6}$$

In this thesis, current is defined as positive going out of a cell which means η is only active for charging currents.

3.3 State of Charge - Open Circuit Voltage relationship

The chemical properties of the LiBs give rise to a nonlinear monotonic relationship between the SoC and the so called Open Circuit Voltage (OCV) that is relatively stationary[1]. In other words, there is a direct (but usually nonlinear) relationship between the terminal voltage of the cell and the SoC. If this SoC-OCV relationship can be established experimentally, it can be used to characterize the cell and used as a parameter during estimation. Barai et al [27] and Farmann and Sauer [25] provide insight into the characteristics of the OCV as well as methods for determining it.

However, crucially, this relationship is *only valid for the at-rest output voltage of the cell*, at-rest here implying that the cell is in electrochemical balance and has not had current passing through it for a long time. The necessary duration of rest will depend on temperature, SoC and

[1]This relationship is likely influenced by the temperature and age of the battery [25], but is assumed to be stationary for the purposes of this thesis. Some researchers argue that the SoC-OCV relationship is independent of temperature and age but instead depends only on the (temperature and age dependent) capacity of the battery [26]

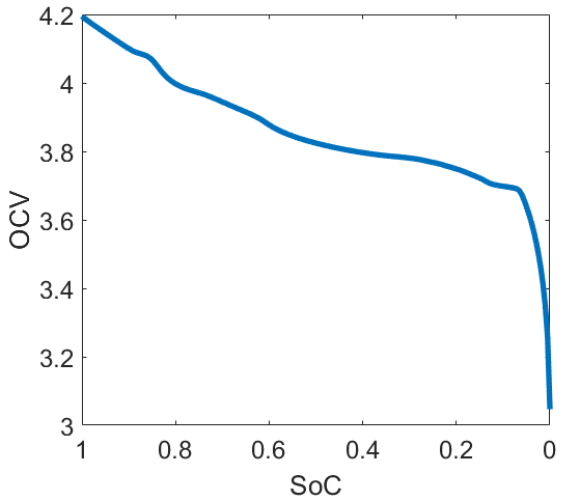


Figure 3.1: SoC-OCV example relationship

aging of the cell, but can potentially be very long (Huria et. al [28] observed a cell that had not reached its at-rest voltage after 13 hours).

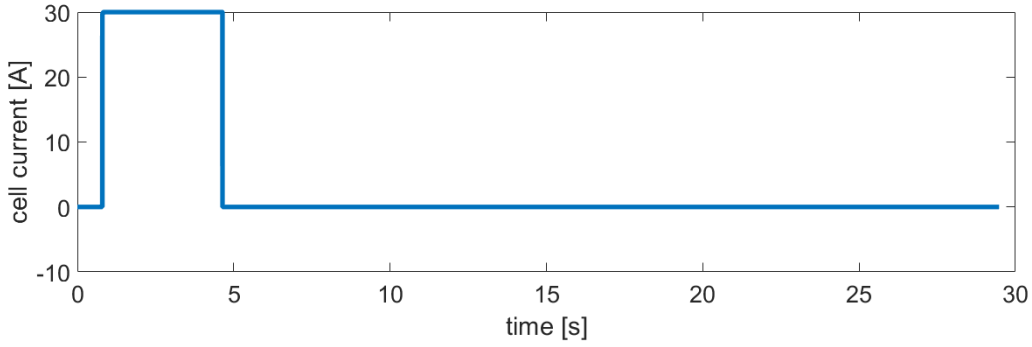
Since the duration of the open circuit condition is so crucial, the term OCV can be perceived as somewhat misleading. In some sources the (ideal) at-rest voltage is referred to as the Electromotive Force (EMF) of the cell [6] to separate it from the immediately available terminal voltage of the cell. However, since many researchers still use OCV and the term seems to be more or less exclusively used to refer to the EMF in literature, this thesis will continue this usage. OCV is thus implied to refer to the (ideal) EMF of the battery, and ”terminal voltage” or V_t is used to refer to the instantaneous voltage observed on the cell terminal outputs. An example SoC-OCV curve is shown in Figure 3.1.

3.4 Polarization voltage and hysteresis

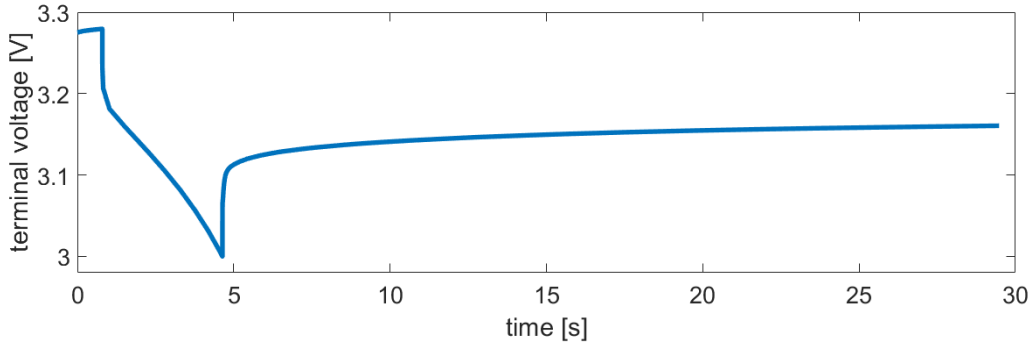
When current starts flowing in the cell, a number of kinetic *polarization effects* must be considered [3]:

- *activation* polarization - related to the charge-transfer process at the interface of the electrodes.
- *ohmic* polarization - caused by resistance of the cell materials and as contact problems
- *concentration* polarization - caused by concentration differences interior to the active material.

The ohmic polarization is instantaneous and disappears when the current of the cell is zero. However, both activation and concentration polarization can persist for a significant period of



(a) Current pulse



(b) Voltage response

Figure 3.2: Polarization effects when a current pulse is applied to a cell. The ohmic polarization is seen as the instantaneous drop and rise of the voltage when the pulse is started and stopped, while the activation and concentration polarization is seen as the slowly climbing voltage after the current is shut off.

time (as discussed in Section 3.2). An example of this is shown in Figure 3.2, where a current pulse is applied momentarily to a battery cell. The end result of these polarizations are that the observed terminal voltage of the cell is not equal to the OCV but is a sum of the OCV and the polarization voltages:

$$V_t = OCV + V_{\text{activation}} + V_{\text{ohmic}} + V_{\text{concentration}} \tag{3.7}$$

In addition to the polarization effects, the voltage output can additionally be affected by *hysteresis*. This is seen as a persistent voltage deviation from the expected OCV that is dependent on the path of the cell current. Barai et al. [27] describe methods for determining hysteresis, and found hysteresis voltages of up to 38mV for some cells.

3.5 Equivalent Circuit Modelling

When modelling battery cells, two major approaches are currently used. The modelling can start from first principles, starting with the physical and chemical laws governing the reactions

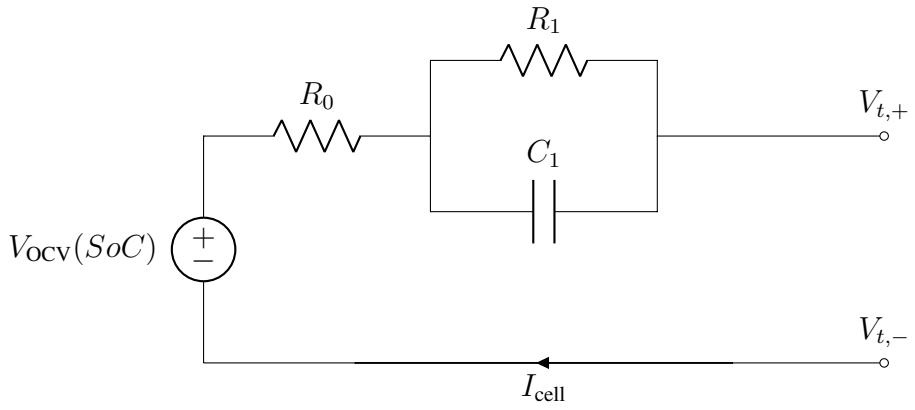


Figure 3.3: Example ECM model with a SoC-OCV relationship, one ohmic resistance (R_0) and one RC element (R_1 and C_1)

interior to the cell. This is often referred to as electrochemical modelling. The resulting models are typically very accurate and can account for most if not all operating conditions of the cells. When used in state estimation the estimated parameters also reveal important insight into the status of the cell. The downside is that processing of these models require vast computational power, making them (currently) unfit for online simulation or real time estimation. This is an active research area and currently is being invested in finding effective model reductions that can be used for real time applications and estimation - often in combination with Particle Filters (see for example [14], [29]–[31]).

A different and popular approach is the use of Equivalent Circuit Models (ECMs). This is a grey-box behavioural approach where emphasis is placed only on the *electrical behaviour* of the cell. The electrical behaviour is approximated through a combination of electrical elements, such as resistors and capacitors. A typical simple ECM is shown in figure 3.3. The result is a lightweight, intuitive and potentially very accurate representation of battery cell behaviour. This method has been and still is very popular due to its low complexity and high accuracy, but these advantages do not come without a price. Most importantly, while the electrical elements can be used to model separate parts of the polarization voltages, it is difficult to draw direct parallels between the parameters of the elements and the actual physical state of the cell.

Chapter 4

Estimation

In this chapter, statistical filtering in a Bayesian framework is discussed in some detail. First the so called Bayesian statistical paradigm is presented along with its implications for inference techniques. It is seen that for the general case, Bayesian inference requires solving intractable integrals. Thereafter the Kalman Filter and Sequential Monte Carlo methods are introduced as practical methods that circumvent this difficulty, under specific assumptions. Thereafter, a more direct comparison of the KF and the SMC methods and their respective assumptions are presented.

4.1 Hidden Markov Model

To be able to use a cell model in an estimator, it must be formulated in a mathematical framework. One very flexible framework for this is the Hidden Markov Model (HMM) model

$$X_1 \sim \mu(x_1) \tag{4.1}$$

$$X_n|(X_{n-1} = x_{n-1}) \sim f(x_n|x_{n-1}) \tag{4.2}$$

$$Y_n|(X_n = x_n) \sim g(y_n|x_n) \tag{4.3}$$

Where $\{X_n\}_{n\geq 1}$ is a discrete-time Markov process, $f(x_n|x_{n-1})$ is the *process transition likelihood* and $g(y_n|x_n)$ is the *measurement likelihood* [32]. This can be reformulated to a wide variety of different subclasses of models. An example of an important special case of the HMM model is given in Equations (4.4)-(4.5). The Kalman Filter presented later requires that the system can be formulated on this form.

$$\mathbf{x}_n = \boldsymbol{f}(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) + \boldsymbol{v}_{n-1} \tag{4.4}$$

$$\boldsymbol{y}_n = g(\boldsymbol{x}_n, \boldsymbol{u}_n) + \boldsymbol{w}_n \tag{4.5}$$

where $\boldsymbol{w}_n, \boldsymbol{w}_n,\ i = 1...n$ are independent additive white noise series.

4.2 Bayesian Filtering

Bayesian Filtering is concerned with using presently available system information and inputs to determine (or estimate) the likelihood of the system states in some future. Informally, this involves having the previous-step state distribution (the *prior* distribution) available and combining this with the system input to form an estimate of the next-step state distribution (the *posterior* distribution) through application of Bayes rule given in Equation (2.7). The posterior distribution of interest is the distribution of the states, given information about the system outputs:

$$p(x_{1:n}|y_{1:n}) \tag{4.6}$$

Applying Bayes rule yields

$$p(x_{1:n}|y_{1:n}) = \frac{p(x_{1:n}, y_{1:n})}{p(y_{1:n})} = \frac{p(x_{1:n})p(y_{1:n}|x_{1:n})}{p(y_{1:n})} \tag{4.7}$$

where

$$p(x_{1:n}) = \mu(x_0) \sum_{k=1}^{n} f(x_k|x_{k-1}) \tag{4.8}$$

$$p(y_{1:n}|x_{1:n}) = \sum_{k=0}^{n} g(y_k|x_k) \tag{4.9}$$

$$p(y_{1:n}) = \int p(x_{1:n})p(y_{1:n}|x_{1:n})\, dx_{1:n} \tag{4.10}$$

with $\mu(x_0)$, $f(x_k|x_{k-1})$ and $g(y_k|x_k)$ given by Equations (4.1), (4.2) and (4.3) respectively.

Furthermore, $p(x_{1:n}, y_{1:n})$ satisfies:

$$p(x_{1:n}, y_{1:n}) = p(x_{1:n-1}, y_{1:n-1}) f(x_n|x_{n-1}) g(y_n|x_n) \tag{4.11}$$

It can then be shown that the posterior satisfies the recursion [32]:

$$p(x_{1:n}|y_{1:n}) = p(x_{1:n-1}|y_{1:n-1}) \frac{f(x_n|x_{n-1})g(y_n|x_n)}{p(y_n|y_{1:n-1})} \tag{4.12}$$

$$p(y_n|y_{1:n-1}) = \int p(x_{n-1}|y_{n-1}) f(x_n|x_{n-1}) g(y_n|x_n)\, dx_{n-1:n} \tag{4.13}$$

It is then straightforward to integrate out $x_{1:n-1}$ in Equation 4.12. The resulting equations, given in (4.14) and (4.15) are typically presented in literature when Bayesian filtering is discussed. Equation (4.14) is often referred to as the *prediction* step, while Equation (4.15) is referred to as the *update* step.

$$p(x_n|y_{1:n-1}) = \int f(x_n|x_{n-1})p(x_{n-1}|y_{1:n-1})dx_{n-1} \tag{4.14}$$

$$p(x_n|y_{1:n}) = \frac{g(y_n|x_n)p(x_n|y_{1:n-1})}{p(y_n|y_{1:n-1})} \tag{4.15}$$

Unfortunately the prediction step integral in Equation (4.14) is intractable save for a few special cases, such as for linear models with Gaussian additive noise [33] (in which case the popular Kalman Filter can be derived). This has led to the development of a wide variety of techniques for handling the more general case of nonlinear and (possibly) non-Gaussian models. Among these are SMC methods and various nonlinear Kalman Filters. These will be presented next.

4.3 Kalman filtering

Kalman Filtering dates back to a paper published by Rudolph Kalman in 1960 [34] and remains one of the most popular filtering methods used for state estimation problems. While the original KF presented by Kalman only applies optimally to linear systems, several extensions have enabled the filter to be used in a wide variety of applications - including the SoC estimation problem. However as will be seen, the fundamental assumptions made to derive the KF also limit the possible accuracy for certain applications. In the following sections a short review of various Kalman Filters will be presented. Main emphasis will be on the nonlinear variants.

The Linear Kalman Filter

If it is assumed that all noise distributions and the prior distribution are Gaussian distributions, it can be shown that the resulting posterior distribution is also Gaussian (see Plett [10] for a proof). Provided that the next step of the algorithm uses the previous posterior as the next prior, this implies that all future posterior densities are Gaussian. Since a Gaussian distribution is fully characterized using only its mean and variance values, this removes the need to store discrete representations of the distributions. If the system additionally is linear, the system equations in discrete time can be expressed as:

$$\mathbf{x}_{n+1} = \mathbf{A_d}\mathbf{x}_n + \mathbf{B_d}\mathbf{u}_n + \mathbf{Q_n} \tag{4.16}$$

$$\mathbf{y}_{n+1} = \mathbf{C_d}\mathbf{x}_{n+1} + \mathbf{D_d}\mathbf{u}_{n+1} + \mathbf{R}_{n+1} \tag{4.17}$$

This is another special case of the HMM, and the *Linear Kalman Filter (LKF)* can be applied. For completeness, the algorithm for the LKF is given in Appendix A. It can be shown that for this particular case, it is optimal in a minimum absolute error sense.

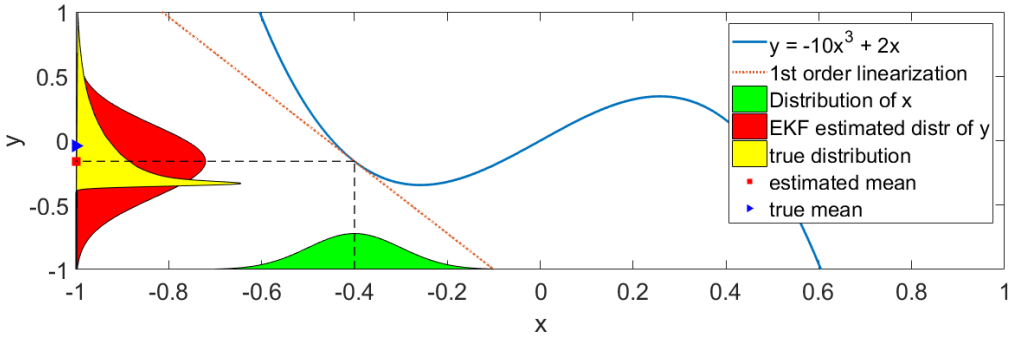
Nonlinear Kalman Filtering

Unfortunately, the algorithm of the LKF is not directly applicable if the system contains nonlinearities. To circumvent this, the system must be linearized in some way. A popular algorithm for this is the EKF, which simply performs a first-order linearization around the previously estimated state value. However, there are several problems with this. The EKF requires the system Jacobian to be computed (either analytically or numerically) in order to linearize. More importantly, if the system is very nonlinear, linearizing around the previous working point might yield an estimate that is far off, causing the next step prediction to miss as well, increasing the error with each step. This can lead to both inaccurate results and divergence of the filter. More recently, a new family of nonlinear KFs have emerged that uses a very different approach: Sigma-Point Kalman Filters (SPKFs). In an SPKF, the linearization is performed by deterministically sampling from the previous-step Gaussian posterior distribution, resulting in a set of so-called 'Sigma Points' that are propagated through the nonlinear functions of the model. The Sigma Points typically have weights attached to them. The resulting 'cloud' of points are assumed to represent a Gaussian distribution, and the updated mean and covariance are estimated from this cloud.

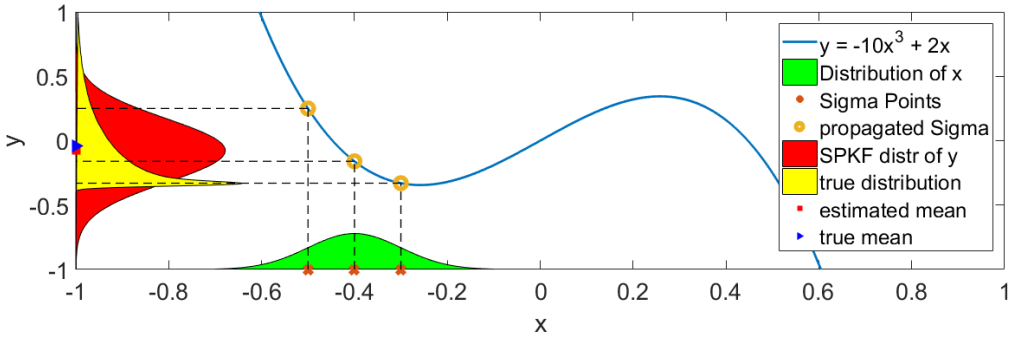
The two approaches are demonstrated for a single-dimensional case in Figure 4.1(a) and 4.1(b). A Normal Distribution (shown along the bottom of the vertical axis) is transformed through the function $y = f(x) = -10x^3 + 2x$, and the corresponding distribution for y is shown on the left of each plot. Figure 4.1(a) show the result of using a 1st order linearization, while Figure 4.1(b) shows the results of using a Sigma-Point approach. As can clearly be seen, the true distribution is distinctly nongaussian and hence the nonlinear KFs are *by design unable to correctly estimate the full distribution*. This is confirmed by comparison of the estimated distributions produced. Still, the Sigma-point approach is able to place the arithmetic mean much more accurately than the 1st order linearization approach. Furthermore, the 1st order linearization procedure creates a distribution with a much wider coverage than the Sigma-Point procedure. If the only metrics of interest is the arithmetic mean and variance of the distribution, the Sigma-Point approach is here seen to be superior to the first order linearization approach.

The Central Difference Kalman Filter (CDKF) algorithm

The most popular variants of the SPKF are the CDKF and the Unscented Kalman Filter (UKF) algorithms. As noted by Plett [10], the two variants are derived using different approaches, but vary only in the weighting scheme used for the Sigma Points. The CDKF only has one tuning parameter (h), while the UKF provides the option of tuning the spread of the Sigma Points as well as the kurtosis of the assumed distribution. While this does provide an opening for accounting somewhat for non-gaussian distributions in the UKF, the parameters must either be fixed or estimated at each step which introduces another complexity. Merwe [35] provide a



(a) First order linearization approach



(b) Sigma Point approach

Figure 4.1: Example of linearization, as performed by the a) EKF b) SPKF. Note that the amplitude of the distributions have been scaled for illustrative purposes.

more thorough discussion of the difference between the UKF and the CDKF, and show that the differences are more nuanced than what is indicated by Plett. Nonetheless, for the purposes of this thesis the approach taken by Plett is presented here. For this thesis the CDKF variant was chosen for its lower complexity (only one tuning parameter). A more thorough discussion on the CDKF algorithm and its relation to the UKF is provided by Plett [10], and a short review is also provided by Roaldsnes [36], from which the contracted algorithm in Table 4.1 is reproduced with some modification. Notably, the version given in [36] calculate sigma points twice per iteration. The one given in Table 4.1 have been modified to only calculate sigma points once, for the sake of efficiency. The CDKF algorithm is intuitively pleasing, and avoiding the need for computing the Jacobian eases the implementation of the system.

For $n = 0$

$$\boldsymbol{x}_0 = E(\boldsymbol{x}) \qquad\qquad \boldsymbol{P}_0 = E(\boldsymbol{P}_0)$$

For $n \geq 1$

Select sigma points and propagate:

$$\begin{aligned}\boldsymbol{\mathcal{X}_{n-1}} &= \left[\hat{\boldsymbol{x}}_{n-1}, \quad \hat{\boldsymbol{x}}_{n-1} + h\sqrt{\boldsymbol{P}_{n-1}}, \quad \hat{\boldsymbol{x}}_{n-1} - h\sqrt{\boldsymbol{P}_{n-1}}\right] \\ \boldsymbol{\mathcal{X}_n^-} &= \boldsymbol{f}(\boldsymbol{\mathcal{X}_{n-1}})\end{aligned}$$

Construct the a priori state estimate and error covariance:

$$\begin{aligned}\hat{\boldsymbol{x}}_{n-1}^- &= \sum_{i=0}^{2N} \omega_i^{\mu} \boldsymbol{\mathcal{X}_n^{(i)-}} \\ \boldsymbol{P}_n^- &= \left\{ \sum_{i=0}^{2N} \omega_i^C (\boldsymbol{\mathcal{X}}_n^{(i)-} - \hat{\boldsymbol{x}}_{n-1}^{(i)-})(\boldsymbol{\mathcal{X}}_n^{(i)-} - \hat{\boldsymbol{x}}_{n-1}^-)^T \right\} + \boldsymbol{Q}_n\end{aligned}$$

Propagate new sigma points through the measurement function:

$$\boldsymbol{\mathcal{Z}}_n^- = h(\boldsymbol{\mathcal{X}_n^-}) \qquad\qquad \hat{\boldsymbol{z}}_n^- = \sum_{i=0}^{2N} \omega_i^{\mu} \boldsymbol{\mathcal{Z}}_n^-$$

Calculate the measurement covariances and Kalman gain:

$$\begin{aligned}(\boldsymbol{P_{zz}^-})_n &= \left\{ \sum_{i=0}^{2N} \omega_i^C (\boldsymbol{\mathcal{Z}}_n^{(i)-} - \hat{\boldsymbol{z}}_n^-)(\boldsymbol{\mathcal{X}}_n^{(i)-} - \hat{\boldsymbol{x}}_n^-)^T \right\} + \boldsymbol{R}_n \\ (\boldsymbol{P_{xz}^-})_n &= \left\{ \sum_{i=0}^{2N} \omega_i^C (\boldsymbol{\mathcal{X}}_n^{(i)-} - \hat{\boldsymbol{x}}_n^-)(\boldsymbol{\mathcal{X}}_n^{(i)-} - \hat{\boldsymbol{z}}_n^-)^T \right\} \\ \boldsymbol{K}_n &= (\boldsymbol{P_{xz}^-})_n (\boldsymbol{P_{zz}^-})_n^{-1}\end{aligned}$$

Calculate the updated state estimate and covariance matrix estimate:

$$\hat{\boldsymbol{x}}_n = \hat{\boldsymbol{x}}_{n-1} + \boldsymbol{K}_n(\boldsymbol{z}_n - \hat{\boldsymbol{z}}_n) \qquad\qquad \boldsymbol{P}_n = \boldsymbol{P}_n^- - \boldsymbol{K}_n(\boldsymbol{P_{zz}^-})_n \boldsymbol{K}_n^T$$

Table 4.1: CDKF algorithm, adapted from Brown and Hwang [37, p.268-269], Plett [10] and Roaldsnes [36].

4.4 Sequential Monte Carlo Filtering

The Kalman Filter uses the Gaussian assumption on the shape of the posterior to simplify the calculations. However, as seen from the shape of the distributions in Figure 4.1, this assumption is not always valid. Even if advanced KFs might be able to estimate the arithmetic mean quite well despite the nonlinearities, the error bounds can still be unreliable. Sequential Monte Carlo (SMC) methods (commonly referred to as PFs) are an attractive alternative when the system is nonlinear, and is used for a variety of applications such as target tracking. The SMC algorithms are intuitively appealing, and mainly distinguish themselves from the nonlinear KFs in two ways: 1) SMC methods are not constrained by the Gaussian assumption, allowing the filter to estimate nontrivial posterior PDFs such as multimodal functions. 2) The model structure of SMC methods are much less restrictive than that of KFs, allowing SMC methods to apply to a diverse array of HMM models such as non-standard measurements [38].

Doucet and Johansen [32] and Tulsyan et al [21] each provide comprehensive overviews of basic and advanced SMC concepts, and the notation used in this thesis is heavily influenced by these researchers. In the following section the main nomenclature and ideas behind SMC methods will be reviewed.

The generic SMC algorithm

SMC methods attempt to estimate an evolving series of (ideal) *target distributions*, denoted $\pi(\cdot)$ and defined as:

$$\pi(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{Z_n} \tag{4.18}$$

where Z_n is an (unknown) normalizing constant, defined as

$$\int \gamma_n(x_{1:n})\, dx_{1:n} \tag{4.19}$$

A typical choice for the target distribution is to set $\pi(x_{1:n}) = p(x_{1:n}|y_{1:n})$ (the posterior distribution), but a variety of choices are available. A main point of emphasis of Doucet and Johansen [32] is the idea that all SMC methods can be designed, described and analyzed within a common framework - referred to as the *generic SMC algorithm*. The main elements of the algorithm is thus similar (or equal) across different variations of the SMC methods irrespective of the chosen target distribution.

As previously described in Section 2.7, the chosen target distribution can be approximated as:

$$\hat{\pi}_n(x_{1:n}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x^i_{1:n}}(x_{1:n}) \tag{4.20}$$

This approximation requires that the target distribution can be sampled from, but this is very rarely the case. In order to circumvent this problem, an *importance distribution* (sometimes referred to as a *proposal distribution*) $q(\cdot)$ with the property:

$$\pi_n(x_{1:n}) > 0 \implies q_n(x_{1:n}) > 0 \tag{4.21}$$

Under this assumption, the target distribution can be rewritten as:

$$\pi_n(x_{1:n}) = \frac{w_n(x_{1:n})q_n(x_{1:n})}{Z_n} \tag{4.22}$$

where:

$$w_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{q_n(x_{1:n})} \tag{4.23}$$

is an unnormalized weighting function. This reformulation means that the importance distribution can be *chosen* among distributions which it is easy to draw samples from, for example the Normal distribution. The weighting expresses how likely a sample from the importance distribution is to be classified as a sample from the target distribution [21].

Due to restrictions on computational power and memory it is not practical to compute the full $\pi_n(x_{1:n})$ at each time step n. Instead, a recursive form of the importance distribution can be written as:

$$q_n(x_{1:n}) = q_{n-1}(x_{1:n-1})q_n(x_n|x_{1:n-1}) \tag{4.24}$$

which leads to a recursive formulation of the unnormalized weighting function:

$$w_n(x_{1:n}) = w_{n-1}(x_{1:n-1}) \cdot \alpha(x_{1:n}) \tag{4.25}$$

where:

$$\alpha_n(x_{1:n}) = \frac{\gamma_n(x_{1:n})}{\gamma_{n-1}(x_{1:n-1})q(x_n|x_{1:n-1})} \tag{4.26}$$

is the *incremental importance weight* function.

A problem with sequential importance sampling is that it tends to center all the weight into a single or very few particles after just a few cycles. As a result, most of the particles are left useless (degenerate), this is known as the *degeneracy* phenomena [37, p. 271], [39]. The degeneracy problem is countered by introducing *resampling* to redistribute the particles so that as many particles as possible remain useful.

The idea behind resampling is to redistribute the particles so that most of the particles are moved to areas of high likelihood. In practice, during resampling the particles with high weights will be replicated many times, while the particles with low weight will either not be replicated or replicated only a few times. A number of different resampling schemes exist, among them

Step	Explanation
1.	sample $U_1 \sim U(0, 1/N)$, where $U(\cdot)$ is a Uniform distribution
2.	$U_i = U_1 + \frac{i-1}{N}$, for $i = 2, ..., N$
3.	build cumulative sum series S $S^{(0)} = 0$, $S^i = \left\{ \sum_{j=1}^{j=i} W_n^{(j)} \right\}$ for $i = 1, .., N$
4.	interpolate in S to select Offspring particles N_n^i by satisfying $N_n^i = \left\| \left\{ S^{(i-1)} \leq U_j \leq S^i \right\} \right\|$

Table 4.2: Systematic Resampling Algorithm

systematic resampling, *residual* resampling, *multinomial* resampling and *stratified* resampling [32], [40]. In this thesis the systematic resampling scheme has been applied, the algorithm is summarized in Table 4.2.

As noted by Doucet and Johansen [32], the process of resampling also introduces some additional variance into the particle set. Consequently, resampling at every time step will cause high variance in the posterior estimate. To avoid this problem somewhat, resampling is performed only when needed. There are several methods of determining when resampling is needed, a popular one is the Effective Sample Size (ESS) [32], as defined in Equation (4.27).

$$ESS = \frac{1}{\sum_{i=1}^{N} (W_n^i)^2} \tag{4.27}$$

The ESS can be interpreted as the number of perfect samples that the current sample set is approximately equivalent to in terms of estimator variance. In other words, if the number of particles is 100 and the ESS is 50, using the 100 particles for inference is approximately equal to having 50 perfect samples with which to perform inference. At each time step, as weight is gradually centered in a few particles and the variance of the weights increases, the ESS will drop. A typical choice is to resample the particles when the ESS falls below $N/2$.

Combining the MC in a recursive formulation with resampling leads to the formulation of the generic SMC algorithm, which is summarized in Table 4.3. In the formulation chosen in this thesis, the calculation of state estimates are placed before the resampling. This way, the state estimate output is less affected by the added sample variance of the resampling procedure.

For $n = 0$

Sample the initial state distribution and weights.

$$X_0^i \sim q(x_0) \qquad\qquad w_0(X_0^i) = \frac{1}{N}$$

For $n \geq 1$

Sample from the importance distribution

$$X_n^i \sim q(x_n^i | X_{1:n-1}^i)$$
$$X_{1:n}^i = \left\{ X_{1:n-1}^i, X_n^i \right\}$$

Compute and normalize weights

$$w_n^i = \alpha_n w_{n-1}$$
$$W_n \propto w_n$$

Calculate state estimate

$$\hat{\pi}(x_n) = \sum_{i=0}^{N} W_n^i \delta_{X_n^i}(x_n)$$

Resample if necessary

if $ESS < N/2$:

resample to get $\{\overline{W}_n^i, \overline{X}_n^i\}$ and set

$$X_{1:n}^i = \left\{ X_{1:n-1}^i, \overline{X}_n^i \right\}$$
$$W_n^i = \overline{W}_n^i = \frac{1}{N}$$

Table 4.3: Generic SMC algorithm. Adapted with modifications from Doucet & Johansen [32]

The Particle Filter

Starting from the generic SMC method in Table 4.3, the simplest and most obvious choice of target distribution is simply the posterior distribution:

$$\pi(x_{1:n}) = p(x_{1:n}|y_{1_n}) \tag{4.28}$$

Considering Equations (4.7) and (4.11) this implies that

$$\gamma_n(x_{1:n}) = p(x_{1:n}, y_{1:n}) = p(x_{1:n-1}, y_{1:n-1}) f(x_n|x_{n-1}) g(y_n|x_n) \tag{4.29}$$

By inserting this into the definition of α_n from Equation (4.26), the following reduction can be made:

$$\alpha_n(x_{1:n}) = \frac{p(x_{1:n-1}, y_{1:n-1})}{p(x_{1:n-1}, y_{1:n-1})} \frac{f(x_n|x_{n-1}) g(y_n|x_n)}{q(x_n|x_{n-1})} = \frac{f(x_n|x_{n-1}) g(y_n|x_n)}{q(x_n|x_{n-1})} \tag{4.30}$$

The full algorithm is given in Table 4.4.

For $n = 0$

Sample the initial state distribution and weights.

$$X_0^i \sim p(x_0) \qquad\qquad w_0(X_0^i) = \frac{1}{N}$$

For $n \geq 1$

Sample from the importance distribution

$$X_n^i \sim f(x_n^i | X_{1:n-1}^i)$$

Compute and normalize weights

$$\begin{aligned} w_n^i =& w_{n-1} \frac{f(X_n^i | X_{n-1}^i) g(y_n | X_n^i)}{q(X_n^i | X_{n-1}^i)} \\ W_n \propto& w_n \end{aligned}$$

Calculate state estimate

$$\hat{\pi}(x_n) = \sum_{i=0}^{N} W_n^i \delta_{X_n^i}(x_n)$$

Resample if necessary

if $ESS < N/2$:

resample to get $\{\overline{W}_n^i, \overline{X}_n^i\}$ and set

$$X_{1:n}^i = \left\{X_{1:n-1}^i, \overline{X}_n^i\right\}$$

$$W_n^i = \overline{W}_n^i = \frac{1}{N}$$

Table 4.4: Particle Filter algorithm. Partially adapted (with modifications) from Doucet & Johansen [32]

The Auxiliary Particle Filter (APF)

Compared to the simple PF, the Auxiliary Particle Filter (APF) is a slightly more advanced particle filter variant. The main difference is that an additional (auxiliary), *predictive* distribution is formed to aid in weighting the particles. This can be reinterpreted as running the generic SMC algorithm with $\gamma(x_{1:n})$ set to

$$\gamma(x_{1:n}) = p(x_{1:n}, y_{1:n})\tilde{p}(y_{n+1}|x_n) \tag{4.31}$$

where $\tilde{p}(y_{n+1}|x_n)$ approximates a prediction density $p(y_{n+1}|x_n)$ [32]. The incremental weight function becomes:

$$\alpha_n(x_{1:n}) = \frac{g(y_n|x_n)f(x_n|x_{n-1})\tilde{p}(y_{n+1}|x_n)}{\tilde{p}(y_n|x_{n-1})q(x_n|y_n, x_{n-1})} \tag{4.32}$$

and since the estimator targets a different distribution than the posterior distribution of interest a separate weight value must be calculated to retrieve the estimate of the state estimate [32]:

$$\tilde{w}_n = \frac{g(y_n|x_n)f(x_n|x_{n-1})}{\tilde{p}(y_n|x_{n-1})q(x_n|x_{n-1})} \tag{4.33}$$

$$\hat{p}(x_n|y_{1:n}) = \sum_{i=1}^{N} \tilde{W}_n^i \delta_{X_{1:n}^i} \tag{4.34}$$

where $\tilde{W}_n \propto \tilde{w}_n$ if resampling was performed the previous iteration and

$$\tilde{W}_n = \tilde{W}_{n-1}^i \tilde{w}_n(X_{n-1:n}^i) \tag{4.35}$$

if not. The complete APF algorithm is summarized in Table 4.5.

For $n = 0$

Sample the initial state distribution and weights.

$$X_0^i \sim p(x_0) \qquad\qquad w_0(X_0^i) = \frac{1}{N}$$

For $n \geq 1$

Sample from the importance distribution

$$X_n^i \sim f(x_n^i | X_{1:n-1,y_{1:n-1}}^i)$$

Compute and normalize weights

$$w_n^i = w_{n-1} \frac{g(y_n|X_n^i) f(X_n^i|X_{n-1}^i) \tilde{p}(y_{n+1}|X_n^i)}{\tilde{p}(y_n|X_{n-1}^i) q(X_n^i|y_n, X_{n-1}^i)}$$

$$W_n \propto w_n$$

Calculate and normalize auxiliary weight

$$\tilde{w}_n = \frac{g(y_n|X_n^i) f(X_n^i|X_{n-1}^i)}{\tilde{p}(y_n|X_{n-1}^i) q(X_n^i|X_{n-1}^i)}$$

If resampling previous iteration: $\tilde{W}_n \propto \tilde{w}_n$

else: $\tilde{W}_n \propto \tilde{W}_{n-1}^i \tilde{w}_n(X_{n-1:n}^i)$

Calculate state estimate

$$\hat{p}(x_n|y_{1:n}) = \sum_{i=1}^{N} \tilde{W}_n^i \delta_{X_{1:n}^i}$$

Resample if necessary

if $ESS < N/2$:

resample to get $\{\overline{W}_n^i, \overline{X}_n^i\}$ and set

$$X_{1:n}^i = \left\{X_{1:n-1}^i, \overline{X}_n^i\right\}$$

$$W_n^i = \overline{W}_n^i = \frac{1}{N}$$

Table 4.5: APF algorithm. Adapted with modifications from Doucet & Johansen [32]

Alternative measurement function: Imprecise modelling

In many papers on the subject of SMC methods for SoC estimation, $g(y_n|x_n)$ is simply selected to be the default Normal distribution. This is also the case for most of the estimators presented in this thesis. However, one of the advantages of using SMC methods is the possibility of employing a wide variety of likelihood functions. As an example, Schwunk et al [13] exchanged the Normal distribution for the fatter-tailed Cauchy-Laurentz distribution.

As another alternative, *imprecise measurement modelling* can be used to create an alternative measurement function $g(y_n|x_n)$ assuming uncertainty in some of the model parameters. This approach has been discussed for the general case previously by Ristic [38], [41].

Imprecise measurements are the result of lack of knowledge, resulting in imprecision in the relationship between the states and the measurements. This is in contrast to the 'precise' models used earlier in this thesis where the relationship between measurement and states is assumed to be known precisely save for some random noise[1].

Similarly to the form in Equations (4.4)-(4.5), the imprecise measurement model can be expressed mathematically as [41]:

$$\mathbf{x}_n = f(\mathbf{x}_{n-1}, \mathbf{u}_{n-1}) + \mathbf{w}_{n-1} \tag{4.36}$$

$$y_n = g(\mathbf{x}_n, \mathbf{u}_n; [\theta_i]_n) + \mathbf{v}_n \tag{4.37}$$

where $\boldsymbol{\theta}_i$ is a vector of uncertainty intervals (the subscript i indicates *imprecise* to distinguish it from the cell parameter vector $\boldsymbol{\theta}_p$). Equation (4.37) is a one-to-many mapping and as such is not a function. However, it can be modelled as a random closed set Σ_x. Ristic [41], following the approach of Mahler [42], show that this leads to the *generalized likelihood*:

$$g_n(y|x_n^i) = \Pr\{y_n \in \Sigma_x\} \tag{4.38}$$

In the case where v is assumed to be additive gaussian noise (which is what is assumed for all models presented in this thesis) $g_n(y|x)$ takes the general form used in this thesis, summarized in Table 4.6

This new form is different from the normal distribution in several ways. Firstly when using the normal distribution as the update function model uncertainty must be accounted for through addition of more measurement noise. In the imprecise approach, uncertain parameters can be accounted for directly in the model formulation and can be asymmetric. Secondly, the imprecise measurement approach allows the measurement function to be directly and dynamically scaled or skewed by the inputs to the system.

[1] Note that imprecision is distinctly separate from randomness [38]

$$g(y|\boldsymbol{x})_{\text{imprecise}} = \int_{\underline{h}_x}^{\overline{h}_x} \mathcal{N}(h; y, R) \tag{4.39}$$

$$= CDF(y; \underline{h}_x, R) - CDF(y; \overline{h}_x, R) \tag{4.40}$$

where

$$\begin{aligned} \underline{\boldsymbol{h}}_{\boldsymbol{x}} &= min\{\boldsymbol{g}(\boldsymbol{x}; \underline{\boldsymbol{\theta}}_{\boldsymbol{i}}), \boldsymbol{g}(\boldsymbol{x}; \overline{\boldsymbol{\theta}}_{\boldsymbol{i}})\} \\ \overline{\boldsymbol{h}}_{\boldsymbol{x}} &= max\{\boldsymbol{g}(\boldsymbol{x}; \underline{\boldsymbol{\theta}}_{\boldsymbol{i}}), \boldsymbol{g}(\boldsymbol{x}; \overline{\boldsymbol{\theta}}_{\boldsymbol{i}})\} \end{aligned} \tag{4.41}$$

Table 4.6: The imprecise likelihood model in the case of Gaussian noise.

Chapter 5

Methods

5.1 Description of battery cell, use case and cell tests

The battery cell used in this thesis is a SLPBB042126 6550mAh 10C Lithium Cobalt Oxide (LCO) pouch cell from Melasta. The cell is optimized for high gravimetric energy density and high power output, and this cell and others similar to it is frequently used in weight and power sensitive applications such as Formula Student race cars. This particular cell is the same type being used by the Formula Student team "Revolve NTNU" in the 2016-2018 seasons. An excerpt of the data sheet is given in Appendix B.

The test performed on the cell was designed and performed by Ørjan Gjengedal in collaboration with Senior Researcher Preben Johannes Svela Vie at the Institute of Energy Technology in Lillestrøm, Norway. The data sets used in this thesis has previously been presented by Gjengedal [12] in the context of Impedance Estimation. The test was performed using a PEC ACT0550 cell tester running LifeTest software. The cells were placed in a Termaks climate chamber set to an ambient temperature of 25 degrees celsius. Two channels of the PEC ACT0050 was connected in parallel in order to allow for a max current of $\pm 100\,\mathrm{A}$. The voltage sense wires were routed separately and connected close to the cell tabs to reduce error from wire resistance. One PT-100 F2020 temperature sensor was attached to the center of the main body of the cell. The sensor was covered with a piece of foam to insulate it from the ambient temperature. An image of the actual cell under test was not available, but an equivalent cell and test setup is shown in figure 5.1.

The current waveform applied to the cell was designed to imitate the typical use case of a Formula Student car when running the 22 km long 'Endurance' race. The race consists of 22 rounds around a 1 km long lap characterized by short straights and tight turns. The 1 km lap features short straights and tight corners, as a result the cells are frequently subjected to short bursts of powerful acceleration followed by aggressive braking. After 11 km a mandatory driver change is performed, resulting in a short break approximately halfway through.

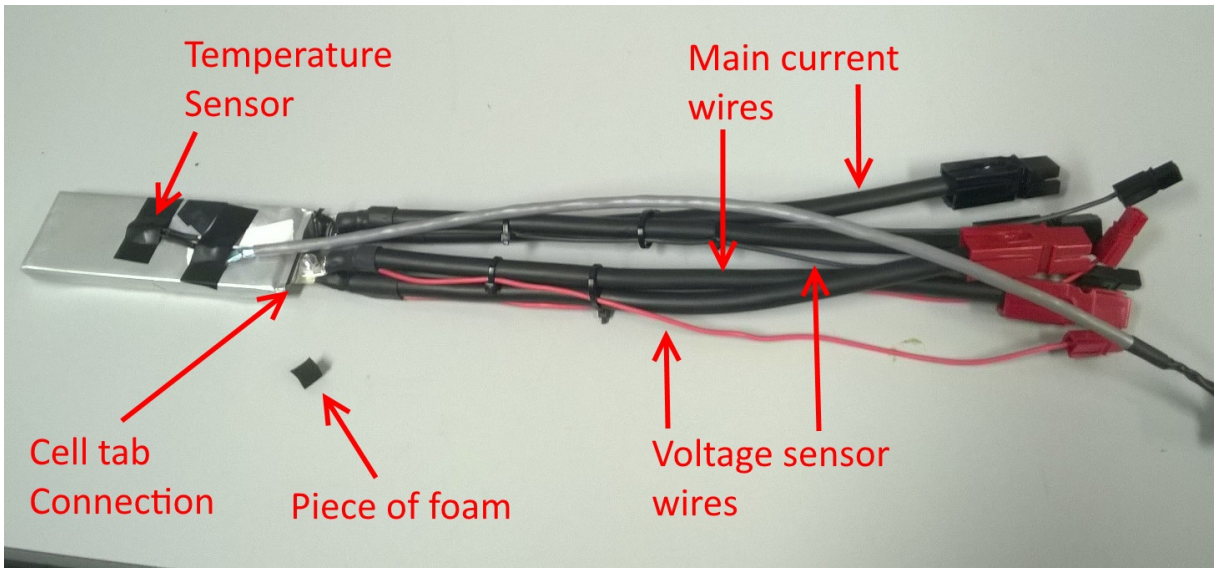


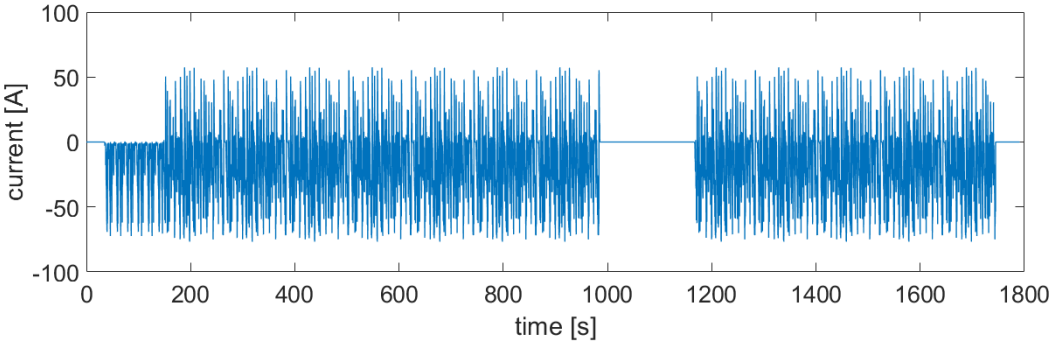
Figure 5.1: Cell test setup

The current waveform was constructed by extracting the current profile logged by a data logger installed in Revolve NTNUs 2016 car 'Gnist' at the Formula Student Austria 2016 Endurance event. From this two separate sample drive cycles were extracted and repeated to create the current waveform shown in Figure 5.2(a) [12]. When applied to the cell the voltage response shown in Figure 5.2(b) was obtained.

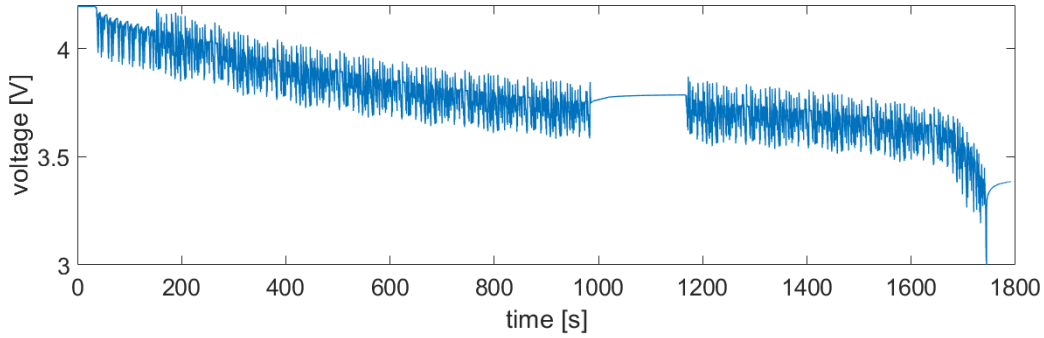
5.2 Selected discrete time cell model

The cell model selected is a simple 2RC ECM. Two RC-elements (R1/C1, R2/C2) are used to approximate the combined effects of activation and concentration polarization voltage. One resistance (R0) approximates the effect of ohmic polarization. The SoC-OCV relationship is modelled as a SoC-controlled ideal voltage supply with the SoC defined as in Equation (3.4). η is assumed to have little impact on the SoC estimate, and is therefore left out of the SoC model. Hysteresis is not modelled, since previous research by Hu et al. [43] indicate that including hysteresis does not necessarily increase accuracy. The continuous ECM is shown schematically in Figure 5.3.

Using straightforward circuit analysis techniques the resulting continuous model is arrived at:



(a) Current waveform for test



(b) Voltage response

Figure 5.2: Current waveform and voltage response

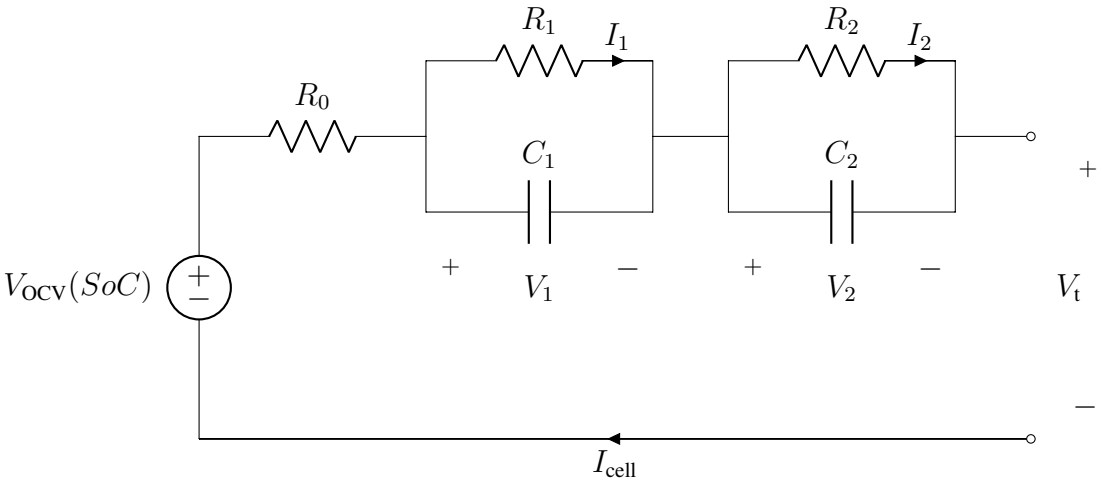


Figure 5.3: Selected cell model - continuous time ECM cell model with 2RC blocks.

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{SoC} \\ \dot{I}_1 \\ \dot{I}_2 \end{bmatrix} = \begin{bmatrix} \frac{I_{\text{cell}}}{Q_{\text{max}}} \\ \frac{1}{R_1 C_1} + \frac{1}{R_1 C_1} I_{\text{cell}} \\ \frac{1}{R_2 C_2} + \frac{1}{R_2 C_2} I_{\text{cell}} \end{bmatrix} \tag{5.1}$$

$$y = OCV(SoC) - R_0 I_{\text{cell}} - R_1 I_1 - R_2 I_2 \tag{5.2}$$

where T is the fixed time step of the discrete system.

Note that all three states are independent and thus can be discretized separately. Applying Eulers method for the SoC state and assuming the current input to be fixed between time steps for the current states I_1, I_2, the system can be linearized as follows [44, p.109-110]:

$$\boldsymbol{x}_n = f_c(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) = \begin{bmatrix} SoC_n \\ I_{1,n} \\ I_{2,n} \end{bmatrix} = \begin{bmatrix} SoC_{n-1} - \frac{T}{3600\text{Cap}} I_{\text{cell},n-1} \\ I_{1,n-1} e^{-\frac{T}{R_1 C_1}} + (1 - e^{-\frac{T}{R_1 C_1}}) I_{\text{cell},n-1} \\ I_{2,n-1} e^{-\frac{T}{R_2 C_2}} + (1 - e^{-\frac{T}{R_2 C_2}}) I_{\text{cell},n-1} \end{bmatrix} \tag{5.3}$$

$$y_n = h_c(\boldsymbol{x}_n, \boldsymbol{u}_n) = OCV(SoC_n) - R_0 I_{\text{cell,n}} - R_1 I_{1,n} - R_2 I_{2,n} \tag{5.4}$$

where T is the sampling period of the system and the subscript c indicates that the $f_c(\cdot)$ and $h_c(\cdot)$ are the cell model functions.

5.3 Augmented cell model

The values of the cell parameters (such as the resistance and capacitance values) of an ECM tends to change with temperature and SoC [45]. A consequence of this is that a model with fixed parameters might be unable to describe the cell throughout a full discharge profile. To counteract this it is possible to create look-up tables for various conditions (such as different temperatures and states of aging) and then optimize the parameters for the various conditions. The problem with this approach is that the system essentially becomes unable to adapt to situations it has not been trained for. Another, alternative approach is to allow the estimator to adapt the cell parameters on-line. In the context of KFs and SMC this is typically done by adding the parameters to the state vector (known as *augmenting* the state vector). This is often referred to as *joint estimation* [9], owing to the fact that the estimator jointly estimates states and parameters.

In this work, the state vector is augmented to include important cell parameters (i.e. capacitance and resistance in RC blocks and the cell capacity) to allow the estimator to adapt

the parameters of the model during operation. In addition to increasing the adaptability of the estimator, this also represents a more realistic use case. The state vector is augmented so that:

$$\boldsymbol{x}_a = \begin{bmatrix} \boldsymbol{x}_s \\ \boldsymbol{\theta}_p \end{bmatrix} \tag{5.5}$$

with the *states* $\boldsymbol{x}_s$ and the *parameters* $\boldsymbol{\theta}_p$ defined to be

$$\boldsymbol{x}_s = \begin{bmatrix} SoC & I_1 & I_2 \end{bmatrix}^T \tag{5.6}$$

$$\boldsymbol{\theta}_p = \begin{bmatrix} R_0 & R_1 & R_2 & C_1 & C_2 & \text{Cap} \end{bmatrix}^T \tag{5.7}$$

The parameters are modelled as integrated white noise, giving the parameter model:

$$\boldsymbol{\theta}_{p,n} = \boldsymbol{\theta}_{p,n-1} + \mathcal{N}(0, \sigma_\theta) \tag{5.8}$$

The parameters are assumed to be constant for a single time step, so the state update function is not changed and the parameters are only actuated by white gaussian noise. By combining Equation (5.3) and (5.4) with Equations (5.6)-(5.8) the cell model used in the estimators is given as:

$$\boldsymbol{x}_{a,n} = \boldsymbol{f}_a(\boldsymbol{x}_{a,n-1}, \boldsymbol{u}_{n-1}) = \begin{bmatrix} SoC_{n-1} - \frac{T}{3600\text{Cap}_{n-1}} I_{\text{cell},n-1} \\ I_{1,n-1} e^{-\frac{T}{R_{1,n-1}C_{1,n-1}}} + (1 - e^{-\frac{T}{R_{1,n-1}C_{1,n-1}}}) I_{\text{cell},n-1} \\ I_{2,n-1} e^{-\frac{T}{R_{2,n-1}C_{2,n-1}}} + (1 - e^{-\frac{T}{R_{2,n-1}C_{2,n-1}}}) I_{\text{cell},n-1} \\ R_{0,n-1} \\ R_{1,n-1} \\ R_{2,n-1} \\ C_{1,n-1} \\ C_{2,n-1} \\ \text{Cap}_{n-1} \end{bmatrix} \tag{5.9}$$

$$h_a(\boldsymbol{x}_n, \boldsymbol{u}_n) = OCV(SoC_n) - R_{0,n} I_{\text{cell,n}} - R_{1,n} I_{1,n} - R_{2,n} I_{2,n} \tag{5.10}$$

Where the subscript a indicates the augmented model. Note that the explicit state names are used above, for greater clarity of presentation. In actual implementation the state names are replaced with their state vector value.

5.4 Determining the SoC-OCV relationship

The SoC-OCV relationship was determined by first charging the cell to full capacity using Constant Current / Constant Voltage (CC/CV) charging, and then fully discharged at a constant rate of C/30 A[1].

After full discharge, the cell was recharged at a constant current of C/30 ampere. The SoC-OCV relationship was then extracted as the average of these two curves. The resulting curve is shown in Figure 6.1. The SoC-OCV relationship was implemented as a lookup-table with linear interpolation and linear extrapolation in MATLAB.

5.5 Determining the reference SoC

The true SoC was established by first assuming that the cell was fully charged at the start of the test $Q_{\text{start}} = Q_{\text{max}}$, and thus SoC is 1. The discharge capacity was taken to be the integrated current of the discharge part of the SoC-OCV test from Section 5.4. The charge capacity was taken to be the integrated current of the charge part of the SoC-OCV test from Section 5.4.

A discrete integration with a triangular approach was selected. For the reference SoC, the SoC model with η included (given in Equation (3.6)) was selected for maximum accuracy. The full method is summarized in Table 5.1

for $n = 1$**:**
$$SoC_n = 1;$$
for $n \geq 2$**:**
if $I > 0$ $$SoC_n = SoC_{n-1} - \frac{T}{3600\text{Cap}} \frac{I_{n-1} + I_n}{2}$$ else $$SoC_n = SoC_{n-1} - \frac{\eta T}{3600\text{Cap}} \frac{I_{n-1} + I_n}{2}$$ Where Cap is the discharge capacity given in $\mathrm{A\,h}$, η is calculated from Equation (3.5), T is the time step of the system and I is the cell current.

Table 5.1: Method for determining reference SoC

[1] 1C is equal to the current required to empty the cell in one hour. For a cell with a capacity of $6.55\,\mathrm{A\,h}$, 1C = $6.55\,\mathrm{A}$, C/2 = $3.275\,\mathrm{A}$ etc.

Parameter settings			
Parameter	starting value	Upper bound	Lower bound
$R_0[\mathrm{m}\Omega]$	2.1	10	1
$R_1[\mathrm{m}\Omega]$	2.5	10	1
$R_2[\mathrm{m}\Omega]$	0.797	10	1
$C_1[\mathrm{F}]$	11500	10^6	10^3
$C_2[\mathrm{F}]$	2821	10^8	10^3
Capacity [A h]	6.86	7.3	6.5
Solver settings			
Algorithm	Trust-region-reflective		
$\boldsymbol{x}_0$	$\begin{bmatrix} SoC & I_1 & I_2 \end{bmatrix}^T = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$		
Function tolerance	10^{-6} (default)		

Table 5.2: Settings for the lsqcurvefit(·) matlab function used to fit cell parameters to cell data.

5.6 Determining starting parameters for cell model

Since the main emphasis of this thesis is not on parameter estimation, the estimators were supplied with reasonable starting estimator values. To determine these values, the 2RC cell model was run in a Least Squares curve-fitting algorithm to solve for the optimal parameter settings. The MATLAB-provided curve-fitting function lsqcurvefit(·) was used with the settings given in Table 5.2. The resulting open-loop voltage output is plotted in Figure 6.2.

5.7 Implemented Estimators

In this section the implementation and tuning settings of the implemented estimators are discussed in detail.

The Joint Central Difference Kalman Filter (JCDKF)

The JCDKF follows the implementation given in Table 4.1, and is given the settings shown in Table 5.3.

System functions

$$\begin{aligned} \boldsymbol{f}(\boldsymbol{x}_n) &= \boldsymbol{f}_a(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) \\ h(\boldsymbol{x}_n) &= h_a(\boldsymbol{x}_n, \boldsymbol{u}_n) \end{aligned}$$

Estimator tuning variable settings

$$\boldsymbol{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2.1 \cdot 10^{-3} \\ 2.7 \cdot 10^{-3} \\ 5.3. \cdot 10^{-4} \\ 7.2 \cdot 10^{3} \\ 2.85 \cdot 10^{3} \\ 6.85 \end{bmatrix} \quad \sqrt{\boldsymbol{P}_0} = diag\left\{ \begin{bmatrix} 1 \\ 10^{-4} \\ 10^{-4} \\ 1.2 \cdot 10^{-4} \\ 10^{-4} \\ 10^{-4} \\ 10 \\ 1 \\ 10^{-3} \end{bmatrix} \right\} \quad \sqrt{\boldsymbol{Q}_n} = diag\left\{ \begin{bmatrix} 10^{-4} \\ 10^{-3} \\ 10^{-6} \\ 10^{-9} \\ 10^{-10} \\ 10^{-10} \\ 10^{-1} \\ 2 \cdot 10^{-7} \\ 10^{-11} \end{bmatrix} \right\}$$

Where $diag\{\}$ is a function that creates an $n \times n$ matrix from the $n \times 1$ input vector. The vector elements are placed along the diagonal of the matrix, and all other elements are set to zero. The noise matrix elements are given as standard deviation for simpler comparison with the PFs.

$$\sqrt{\boldsymbol{R}_n} = 0.2 \qquad h = \sqrt{3}$$

$$w_0^{\mu} = w_0^{C} = \frac{h^2 - L}{h^2} \qquad w_i^{\mu} = w_i^{C} = \frac{1}{2h^2}$$

Where L is the number of elements in $\boldsymbol{x}_a$ and $i = 1...2L$.

Table 5.3: Joint Central Difference Kalman Filter (JCDKF) estimator settings

The Joint Bootstrap Particle Filter (JBPF)

If the proposal density in the Particle Filter is chosen to be equal to the process likelihood so that

$$q(x_n|x_{n-1}) = f(x_n|x_{n-1}) \tag{5.11}$$

then the weight update function is reduced to a very simple form:

$$w_n^i = w_{n-1}\frac{f(X_n^i|X_{n-1}^i)g(y_n|X_n^i)}{q(X_n^i|X_{n-1}^i)} = w_{n-1}\, g(y_n|X_n^i) \tag{5.12}$$

The resulting filter is known as the Bootstrap Particle Filter (BPF) [37, p.271]. Using the BPF with the augmented model given in Section 5.3 yields the JBPF. The JBPF implemented here is simple and straightforward, using the same model and initialization distributions as the Kalman Filter. In practice, this means that all likelihood distributions are selected to be Normal distributions. The process equation was selected as the linear combination of the augmented cell model given in Section 5.3 and discrete draws from a normal Gaussian noise of variance $\boldsymbol{\sigma}_p$ to jitter the particles and provide sample diversity.

The measurement likelihood model was selected as a propagation of particles through the measurement function and evaluation by a normal distribution. The initial state distribution likelihood was selected as a Normal distribution for all the states. The settings selected for the JBPF are summarized in Table 5.4.

Likelihood functions

$$
\begin{aligned}
p(x_0) &= \mathcal{N}(\boldsymbol{x}_0, \boldsymbol{\sigma}_0) \\
f(x_n|x_{n-1}) &= \boldsymbol{f}_a(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) + \mathcal{N}(0, \boldsymbol{\sigma}_p) \\
g(y_n|x_n) &= \mathcal{N}(\, h_a(\boldsymbol{x}_n, \boldsymbol{u}_n)\,; V_{\text{meas,n}}, \sigma_m) \\
q(x_n|x_{n-1}) &= f(x_n|x_{n-1})
\end{aligned}
$$

Estimator tuning variable settings

$$
\boldsymbol{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2.1 \cdot 10^{-3} \\ 2.7 \cdot 10^{-3} \\ 5.3. \cdot 10^{-4} \\ 7.2 \cdot 10^{3} \\ 2.85 \cdot 10^{3} \\ 6.85 \end{bmatrix} \qquad \boldsymbol{\sigma}_0 = \begin{bmatrix} 1 \\ 10^{-3} \\ 10^{-3} \\ 10^{-4} \\ 10^{-4} \\ 10^{-4} \\ 100 \\ 100 \\ 10^{-3} \end{bmatrix} \qquad \boldsymbol{\sigma}_p = \begin{bmatrix} 2 \cdot 10^{-4} \\ 10^{-3} \\ 10^{-3} \\ 10^{-8} \\ 10^{-8} \\ 10^{-6} \\ 10^{-8} \\ 10^{-6} \\ 10^{-6} \end{bmatrix} \qquad \sigma_m = 0.2
$$

The number of particles $N = 100$.

Table 5.4: Joint Bootstrap Particle Filter (JBPF) estimator settings

The Joint Imprecise Bootstrap Particle Filter (JIBPF)

The JIBPF uses the same setup as the JBPF but exchanges the Gaussian measurement function for the imprecise likelihood function given in Equation (4.40). The measurement function is adapted to fit three imprecision parameters, one for each resistor. This is intended to reflect the fact that there might be an imprecision in the way the resistors affect the voltage output. Additionally, adding the imprecision to the resistor means that when the cell is at rest and $I_{\text{cell}} = I_1 = I_2 = 0$, the voltage imprecision will effectively go to zero. This plays into the ideal situation when the cell is at complete rest and only the SoC-OCV relationship is important[2]. The settings and equations used are given in Table 5.5.

[2]This is of course only true if one can ignore any hysteresis effects, which is assumed here.

Likelihood functions

$$\begin{aligned} p(x_0) &= \mathcal{N}(\boldsymbol{x}_0, \boldsymbol{\sigma}_0) \\ f(x_n|x_{n-1}) &= \boldsymbol{f}_a(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) + \mathcal{N}(0, \boldsymbol{\sigma}_p) \\ g(y_n|x_n) &= CDF(V_{\text{meas,n}}; \underline{h}_x, R) - CDF(y; \overline{h}_x, R) \\ q(x_n|x_{n-1}) &= f(x_n|x_{n-1}) \end{aligned}$$

Where $\underline{h}_x$ and $\overline{h}_x$ are defined as in Equation 4.41, and $g(\boldsymbol{x}; \boldsymbol{\theta}_i)$ is defined as

$$g(\boldsymbol{x}_n, \boldsymbol{u}_n; \boldsymbol{\theta}_i) = OCV(SoC_n) - (R_0 + \theta_{i,1})I_{\text{cell,n}} - (R_1 + \theta_{i,2})I_{1,n} - (R_2 + \theta_{i,3})I_{2,n}$$

Estimator tuning variable settings

$$\boldsymbol{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2.1 \cdot 10^{-3} \\ 2.7 \cdot 10^{-3} \\ 5.3. \cdot 10^{-4} \\ 7.2 \cdot 10^{3} \\ 2.85 \cdot 10^{3} \\ 6.85 \end{bmatrix} \qquad \boldsymbol{\sigma}_0 = \begin{bmatrix} 1 \\ 10^{-3} \\ 10^{-3} \\ 10^{-4} \\ 10^{-4} \\ 10^{-4} \\ 100 \\ 100 \\ 10^{-3} \end{bmatrix} \qquad \boldsymbol{\sigma}_p = \begin{bmatrix} 2 \cdot 10^{-4} \\ 10^{-3} \\ 10^{-3} \\ 10^{-8} \\ 10^{-8} \\ 10^{-6} \\ 10^{-8} \\ 10^{-6} \\ 10^{-6} \end{bmatrix} \qquad \boldsymbol{\sigma}_m = 0.2$$

$$\underline{\boldsymbol{\theta}} = \begin{bmatrix} -5 \cdot 10^{-}4 \\ -1 \cdot 10^{-4} \\ -1 \cdot 10^{-4} \end{bmatrix} \qquad \overline{\boldsymbol{\theta}} = \begin{bmatrix} 5 \cdot 10^{-}4 \\ 1 \cdot 10^{-4} \\ 1 \cdot 10^{-4} \end{bmatrix} \qquad R = 0.2$$

The number of particles $N = 100$.

Table 5.5: Joint Imprecise Bootstrap Particle Filter (JIBPF) estimator settings

The JAPF

When using the APF with the augmented model from Section 5.3, the resulting estimator is the JAPF. The JAPF implemented here shares the basic densities with the JBPF, specifically it uses the same definitions for $q(x_0)$, $f(x_n|x_{n-1}), g(y_n|x_n)$ and $q(x_n|x_{n-1})$ as the JBPF. This leaves the predictive densities $\tilde{p}(y_{n+1}|x_n)$ to be chosen. A variety of strategies exist for constructing these densities (see for example [32], [39], [46]). Under the assumption that the estimator can be delayed by one time step[3], the following method has been applied here.

To evaluate $\tilde{p}(y_{n+1}|X_n^i)$, first project the particles X_n^i through the process likelihood function $\tilde{f}(x_{n+1}|X_n^i)$. $\tilde{f}(x_{n+1}|X_n^i)$ is equal to the usual process likelihood function, but does not add noise to the particles. Secondly, apply weighting based on the modified measurement likelihood function $\tilde{g}(y_{n+1}|\tilde{X}^{(i)})$. $\tilde{g}(y_n|x_n)$ is a direct application of the Cauchy Lorentz (CL) distribution. The CL distribution is symmetrical, but has fatter tails than the Normal distribution, meaning it spreads out farther from the mean than the Normal distribution. Using it here helps prevent a situation where an outlier particle is given a weight of zero, which could trigger numerical issues. Its use here is inspired by Schwunk et al [13].

In summary, the modified process and measurement likelihood distributions are written as

$$\tilde{X}_n^{(i)} = \tilde{p}(x_n|X_{n-1}^i) = \boldsymbol{f}_a(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) \tag{5.13}$$

$$\tilde{g}(y_{n+1}|X_n^i) = CL(\boldsymbol{h}_a(\tilde{X}_n^i, \boldsymbol{u}_n); \mu, s) \tag{5.14}$$

where

$$CL(x; \mu, s) = \frac{1}{\pi} \frac{s}{s^2 + (x-\mu)^2} \tag{5.15}$$

is the evaluation of the Cauchy Lorentz likelihood function.

The JAPF is then implemented according to the algorithm given in Table 4.5, with the likelihood distributions and settings given in Table 5.6.

[3]Given the high update frequency of 10 Hz and comparatively slow change of SoC (SoC changes from 1 to zero in approximately 30 minutes), a single time step delay is seen as inconsequential. Note however that this need not be the case for all applications.

Likelihood functions

$$
\begin{aligned}
p(x_0) &= \mathcal{N}(\boldsymbol{x}_0, \boldsymbol{\sigma}_0) \\
f(x_n|x_{n-1}) &= \boldsymbol{f}_a(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) + \mathcal{N}(0, \boldsymbol{\sigma}_p) \\
g(y_n|x_n) &= \mathcal{N}(h_a(\boldsymbol{x}_n, \boldsymbol{u}_n); V_{\text{meas,n}}, \sigma_m) \\
q(x_n|x_{n-1}) &= f(x_n|x_{n-1}) \\
\\
\tilde{p}(x_n|x_{n-1}^i) &= \boldsymbol{f}_a(\boldsymbol{x}_{n-1}, \boldsymbol{u}_{n-1}) \\
\tilde{X}_n^{(i)} &= \tilde{p}(x_n|x_{n-1}^i) \\
\tilde{g}(y_{n+1}|X_n^i) &= CL(h_a(\tilde{X}_n^i, \boldsymbol{u}_n); V_{\text{meas,n}}, s_{\text{CL}})
\end{aligned}
$$

Estimator tuning variable settings

$$
\boldsymbol{x}_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 2.1 \cdot 10^{-3} \\ 2.7 \cdot 10^{-3} \\ 5.3. \cdot 10^{-4} \\ 7.2 \cdot 10^{3} \\ 2.85 \cdot 10^{3} \\ 6.85 \end{bmatrix} \qquad \boldsymbol{\sigma}_0 = \begin{bmatrix} 1 \\ 10^{-3} \\ 10^{-3} \\ 10^{-4} \\ 10^{-4} \\ 10^{-4} \\ 100 \\ 100 \\ 10^{-3} \end{bmatrix} \qquad \boldsymbol{\sigma}_p = \begin{bmatrix} 2 \cdot 10^{-4} \\ 10^{-3} \\ 10^{-3} \\ 10^{-8} \\ 10^{-8} \\ 10^{-6} \\ 10^{-8} \\ 10^{-6} \\ 10^{-6} \end{bmatrix} \qquad \sigma_m s_{\text{CL}} = 0.2
$$

The number of particles $N = 100$.

Table 5.6: Joint Auxiliary Particle Filter (JAPF) estimator settings

5.8 Estimating Confidence bounds in the filters

Section 2.8 discusses briefly how to construct confidence intervals from the estimators. The algorithms for establishing the confidence bounds of the state estimates are given in Table 5.7 and 5.8

1. Sort particles by value along the dimension of interest (e.g. SoC or I_1 or R_0 etc). Implemented using the sort($\cdot$) function in MATLAB.
2. Calculate the cumulative sum of the weights along the dimension of interest. Implemented using the cumsum($\cdot$) function in MATLAB.
3. Evaluate the inverse cumulative sum of the weights at the confidence bounds of interest. Implemented using the interp1($\cdot$) function in MATLAB. When a requested value falls between two values in the cumulative sum, linear interpolation is used.

Note that for the APF, the weights $\tilde{w}$ must be used.

Table 5.7: Evaluating the confidence bounds for the Particle Filter

1. Let i be the number of the state of interest (e.g. SoC, I_1 etc)
2. Evaluate the normal inverse cumulative density function with $\mu = \hat{x}_i$ and $\sigma = \sqrt{\hat{P}_{i,i}}$ at confidence bounds of interest. Implemented using the norminv($\cdot$) function in MATLAB.

Table 5.8: Evaluating the confidence bounds for the Kalman Filter

5.9 Determining the effect of the system nonlinearity

As previously discussed, a fundamental result underlying the KF is that linear transformations of Gaussian distributions are also Gaussian distributed, per the reproductive property given in Table 2.1. This is the fundamental property that allows the KF to simply propagate means and variances. However if parts of the system is not linear, as is the case in the LiBs, this is no longer strictly valid. Thus, when evaluating the estimator schemes presented in this thesis, a point of particular interest is the ability of each estimator to handle the nonlinearities presented by this particular use case, the battery cell model.

In order to evaluate the nonlinearities present in the system, several approaches are possible. If an analytic expression is available for the transformations, an attempt can be made to derive analytic expressions for the transformed variables. Alternatively, a Monte Carlo style method can be applied to explore the system behaviour. This is the approach chosen in this thesis as it is very straightforward to implement.

The method proceeds as follows. N_{mc} state vectors are constructed by drawing N_{mc} random numbers for each of the states and parameters. Each state and parameter has a separate distribution from which its random values are drawn. To facilitate easy comparison with the CDKF way of calculating the distribution, the Normal distribution is selected as the starting distribution. The N_{mc} state vectors are then propagated once through the process equation and once through the measurement equation of the system. The output of this is N_{mc} voltage values, which can be used to estimate a PDF of the voltage state given the (Normally distributed) starting state vector. Knowing the 'true' distribution, the distribution metrics can be calculated, and equivalent estimates can be generated from all the estimators presented in this thesis. The algorithm is summarized in Table 5.9.

1. Draw N_{mc} state vectors from the Normal distribution. Each of the states of the state vector have separate settings for their mean μ and variance σ^2, given in Table 5.10.
2. Propagate the N_{mc} vectors through the process and measurement functions and obtain N_{mc} voltage values.
3. Form an estimate of the PDF of the measurement from the N_{mc} using the Kernel estimate method. Implemented using the ksdensity(·) function in MATLAB.
4. Calculate the properties of the resulting PDF
   a) Calculate skew and kurtosis by applying formula (2.9)-(2.8). $E(x)$ is calculated with the mean(·) function in MATLAB.
   b) Calculate confidence bounds for the skew and kurtosis using the pre-made MATLAB bootci(·) function with the settings given in Table 5.10
5. run a CDKF for one iteration starting with $\boldsymbol{P}_0 = diag(\boldsymbol{\sigma}_0)$, $R = 0$, $\boldsymbol{Q} = \begin{bmatrix}\mathbf{0}\end{bmatrix}$ and $\boldsymbol{x}_0$ given in Table 5.10 and obtain $\hat{y}$, P_y.
6. Plot the Normal distribution estimated by the CDKF (defined by $\mathcal{N}(\hat{y}, \sqrt{P_y})$ alongside the Kernel density estimate of the true distribution.

Table 5.9: Nonlinearity analysis description

Estimator tuning variable settings

$$\boldsymbol{x}_0 = \begin{bmatrix} SoC_{\text{start}} \\ 0 \\ 0 \\ 0.0021 \\ 0.0025 \\ 7.97 \cdot 10^{-4} \\ 1.15 \cdot 10^{4} \\ 2.8 \cdot 10^{3} \\ 6.86 \end{bmatrix} \qquad \boldsymbol{\sigma}_0 = \begin{bmatrix} 0.01 \text{ or } 0.03 \\ 10^{0} \\ 10^{0} \\ 10^{-4} \\ 10^{-4} \\ 10^{-4} \\ 100 \\ 100 \\ 10^{-1} \end{bmatrix} \qquad I_{\text{cell}} = 0$$

where

$$SoC_{\text{start}} = 0.01, 0.02, ...1$$

The number of particles $N_{\text{par}} = 15000$, and the number of bootstrap resamplings were set to $N_{\text{boot}} = 2000$.

Table 5.10: Nonlinearity analysis settings

Chapter 6

Results of the study

6.1 Main findings

The nonlinearity analysis conducted in this thesis show that the SoC-OCV relationship of the tested cell introduces a nonlinearity that might have a noticeable impact on the SoC-estimate at SoCs below approximately 12%. However, applying various PFs and a CDKF to a real current and voltage waveform, the practical difference between the two approaches is not obvious. The JCDKF, JBPF, JIBPF and the JAPF all have similar performance in all ranges of SoC. It is even observed that the CDKF seems to outperform the PFs. Furthermore, the parameters of the cell is seen to behave erratically and randomly, as if the feedback from the measurement of the cell is weak when estimated using Joint Particle Filters. The problem might be few particles or some other issue. On the other hand it is also observed that the likelihood distributions for the parameters seem to be distinctly non-symmetric which might indicate that the PF approach might be useful for parameter estimation if the feedback issue can be solved.

6.2 SoC-OCV relationship found

The method outlined in Section 5.4 was used to determine the SoC-OCV relationship at 25 °C The resulting curve is given in 6.1.

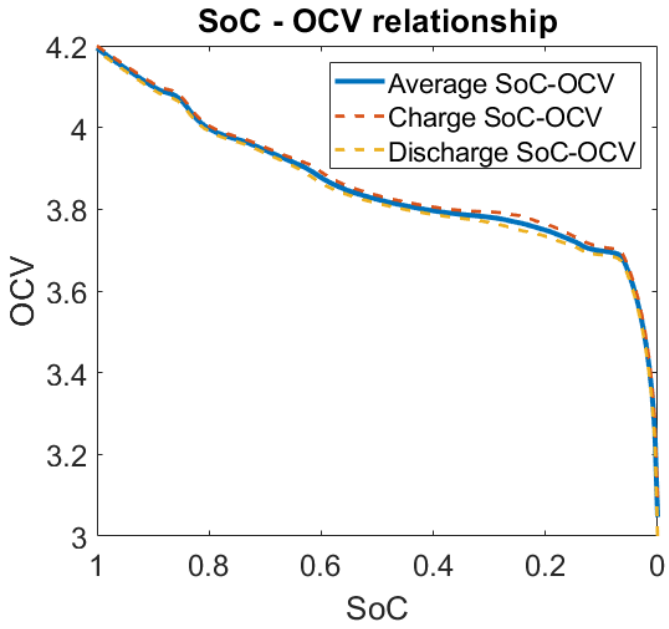
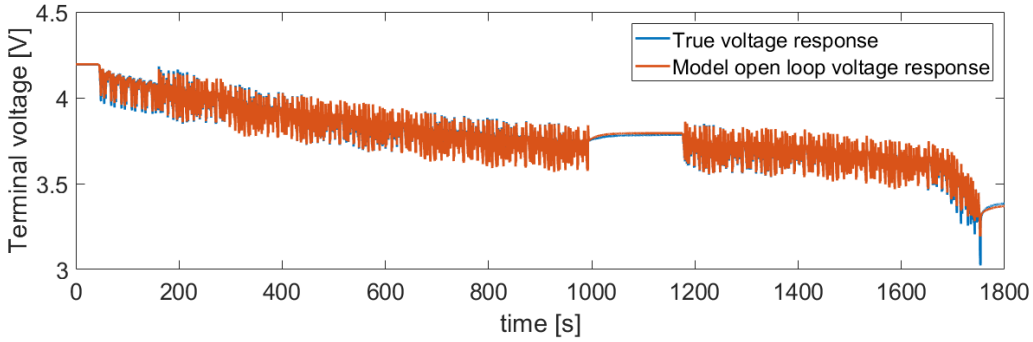


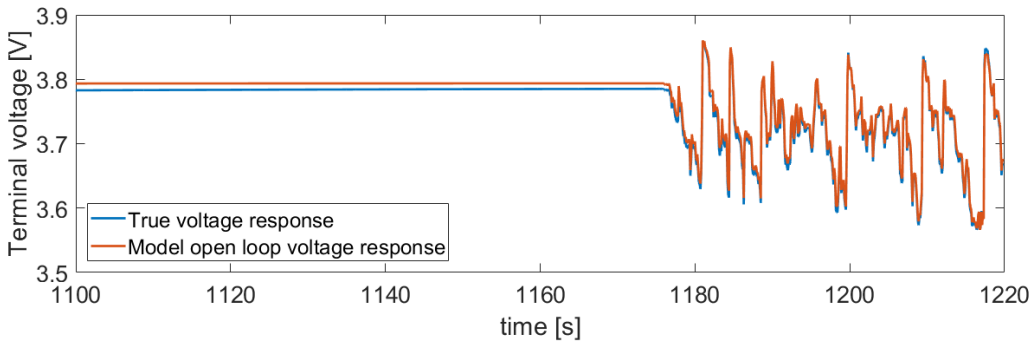
Figure 6.1: SoC-OCV relationship found for the tested cell

6.3 Cell model open loop response

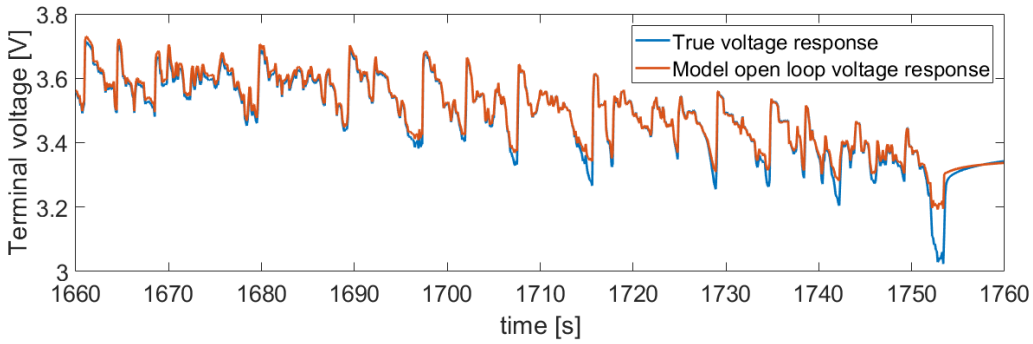
The cell model was excited in open loop using the current profile given in Figure 5.2(a). The main purpose of this is to evaluate whether or not the cell model is able to approximate the actual voltage response of the cell. In Figure 6.2, the voltage response is given both as the full waveform and with zoom on specific regions.



(a) Full waveform



(b) Zoom on middle period



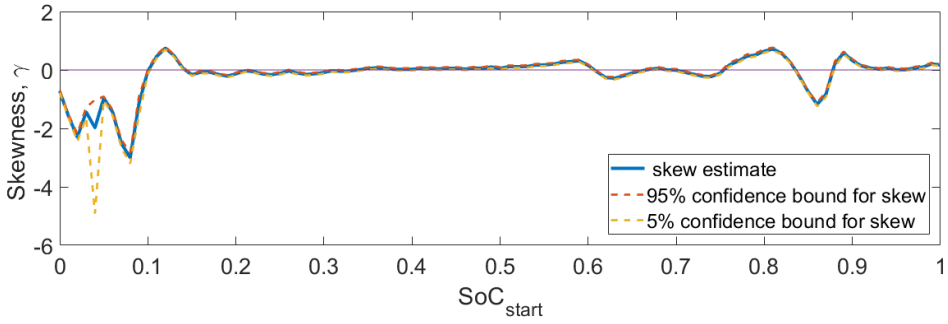
(c) Zoom on late period

Figure 6.2: Open loop voltage response of cell model compared to true cell voltage

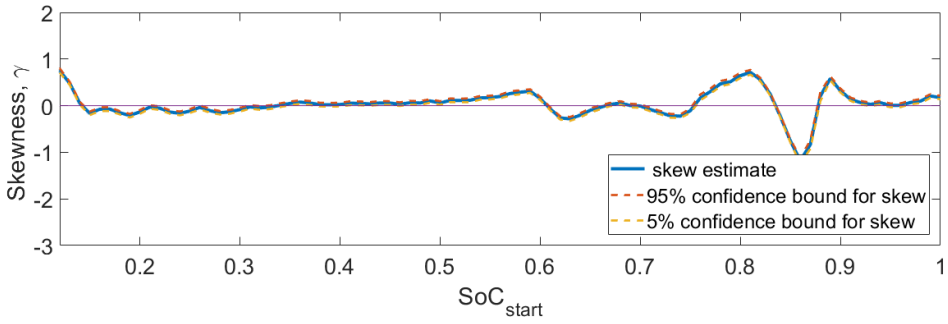
6.4 Results of model nonlinearity analysis

The Nonlinearity analysis was applied to the fully augmented 9-state model with the settings given in Table 5.10. At each value of SoC_{start}, estimates and confidence bounds of the skew and kurtosis was calculated and the results plotted with SoC_{start} on the horisontal axis and the respective estimates and confidence bounds on the vertical axis in for 3% standard deviation on the SoC state in Figure 6.4 and 1% standard deviation on the SoC state in Figure 6.4.

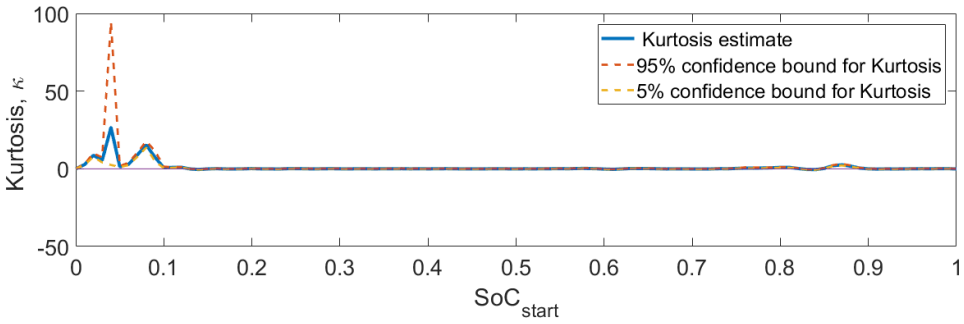
Furthermore, as a reference and for direct comparison, the CDKF method was used to perform an equivalent estimate of the voltage distribution, as described in Table 5.9. The corresponding skew and kurtosis estimates and confidence bounds were computed using the Bootstrap method of 2.6 In the interest of compactness, the resulting likelihood distributions are plotted only for $SoC_{\text{start}} = \{0.1, 0.5, 0.85\}$. Figure 6.6 show the results for a SoC standard deviation of 3%, while Figure 6.5 shows the results for a SoC standard deviation of 1%.



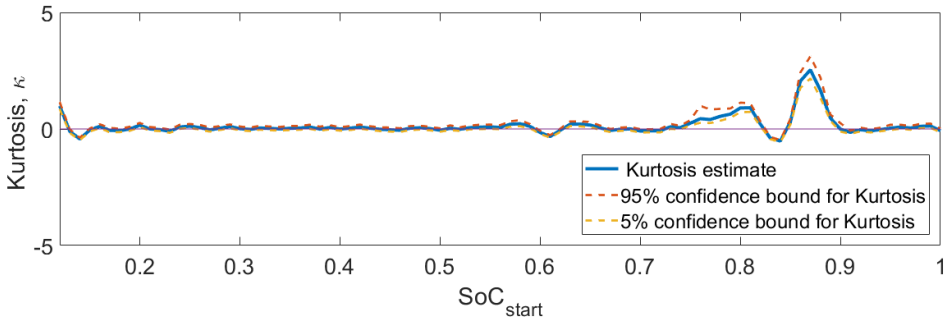
(a) Skew plotted against SoC



(b) Skew plotted against SoC, zoom on $SoC > 0.10$

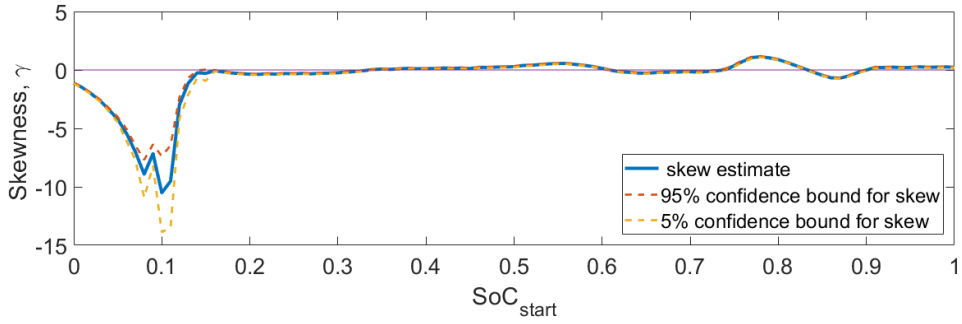


(c) Kurtosis plotted against SoC

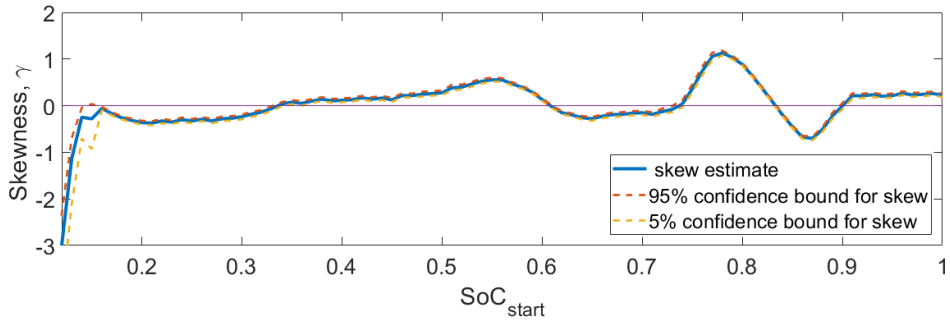


(d) Kurtosis plotted against SoC, zoom on $SoC > 0.10$

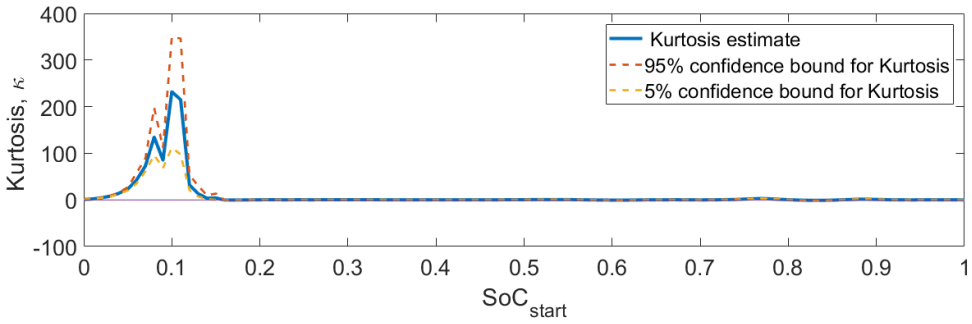
Figure 6.3: Skew and kurtosis from nonlinearity nonlinearity analysis with 0.01 standard deviation in the SoC



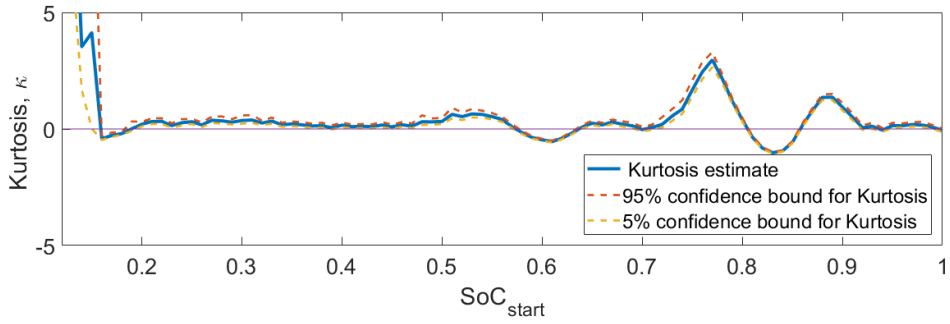
(a) Skew plotted against SoC



(b) Skew plotted against SoC, zoom on $SoC > 0.10$

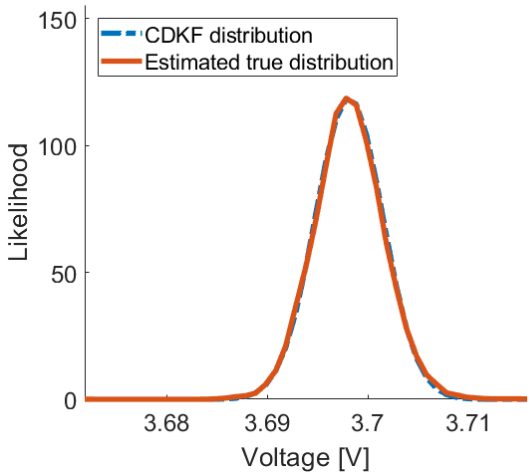


(c) Kurtosis plotted against SoC

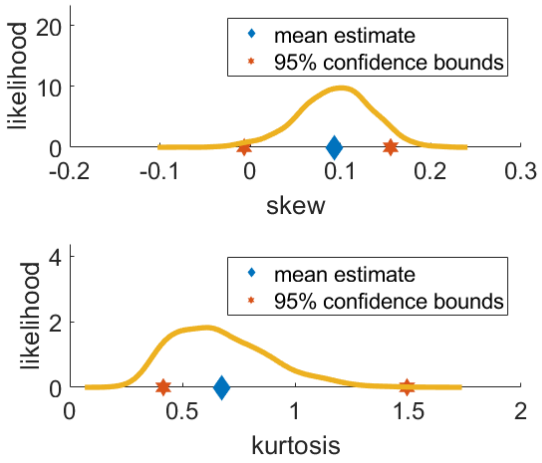


(d) Kurtosis plotted against SoC, zoom on $SoC > 0.10$

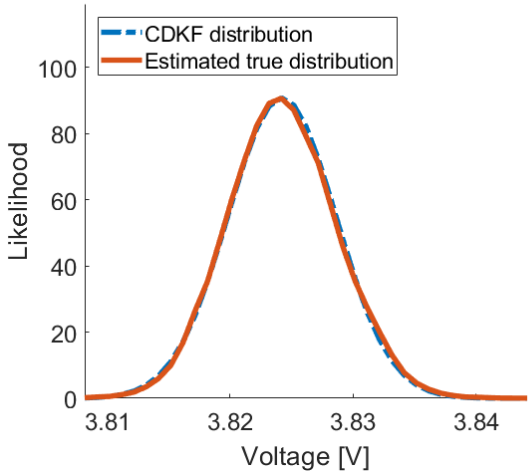
Figure 6.4: Skew and kurtosis from nonlinearity nonlinearity analysis with 0.03 standard deviation in the SoC



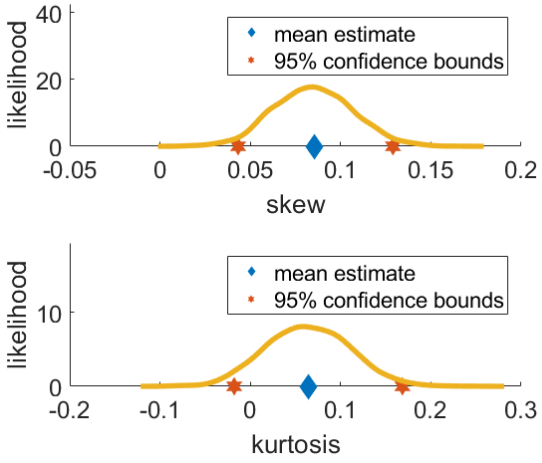
(a) SoC = 0.1, distribution



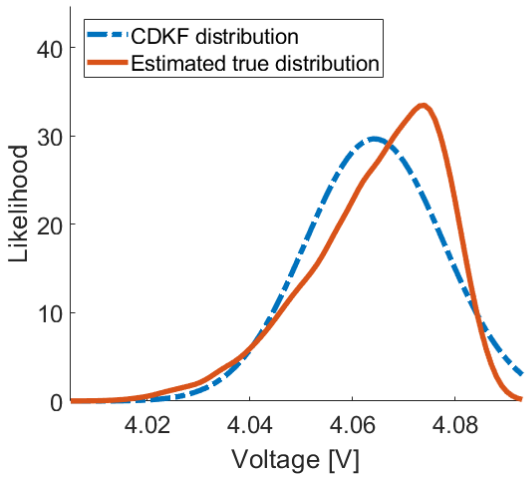
(b) SoC = 0.1, kurtosis and skew estimates



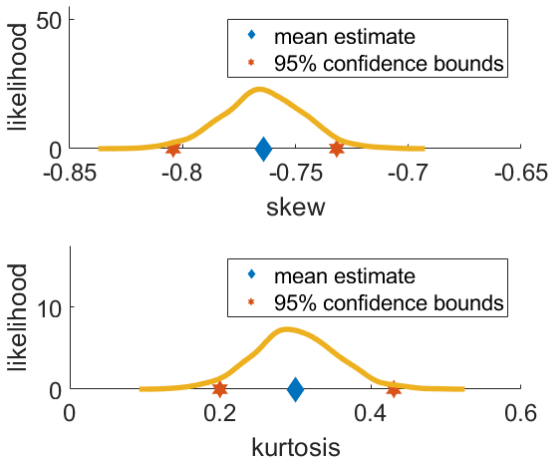
(c) SoC = 0.5, distribution



(d) SoC = 0.5, kurtosis and skew

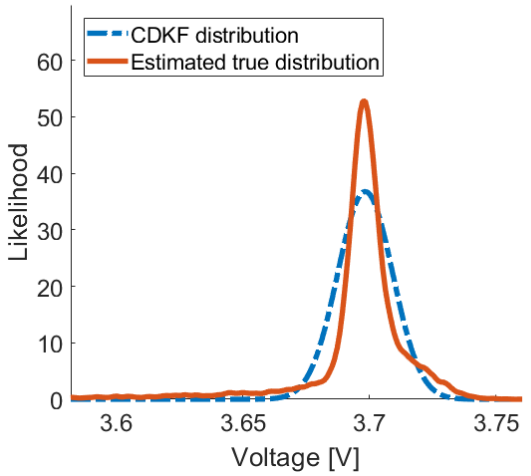


(e) SoC = 0.85, distribution

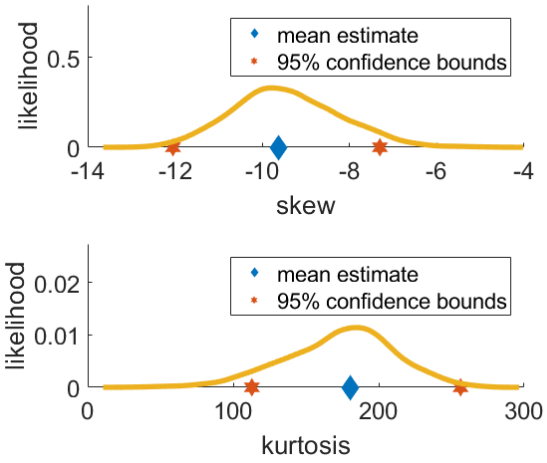


(f) SoC = 0.85, kurtosis and skew

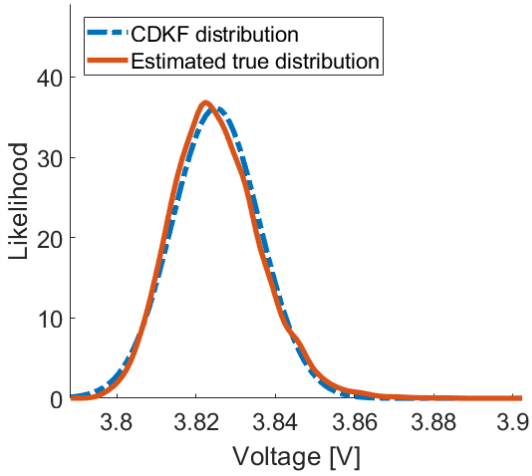
Figure 6.5: Distribution comparisons between 'true' distribution and CDKF estimated distribution.SoC standard deviation of 1%



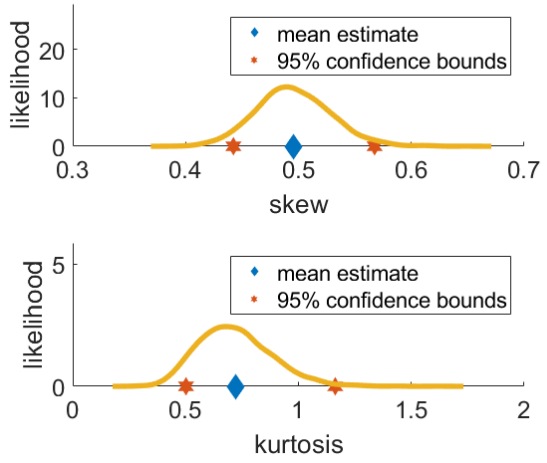
(a) SoC = 0.1, distribution



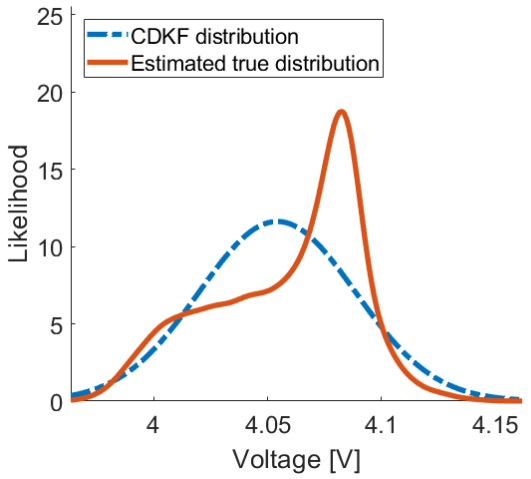
(b) SoC = 0.1, kurtosis and skew estimates



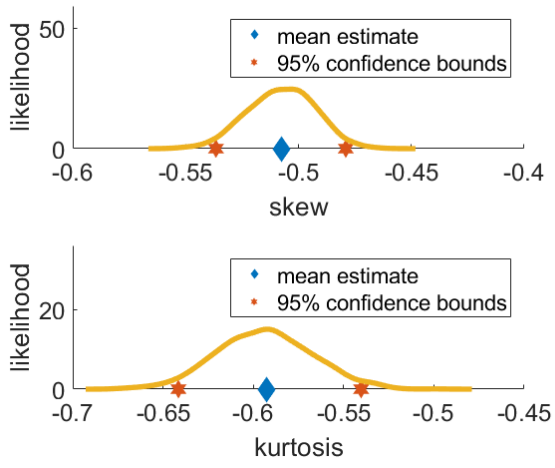
(c) SoC = 0.5, distribution



(d) SoC = 0.5, kurtosis and skew



(e) SoC = 0.85, distribution

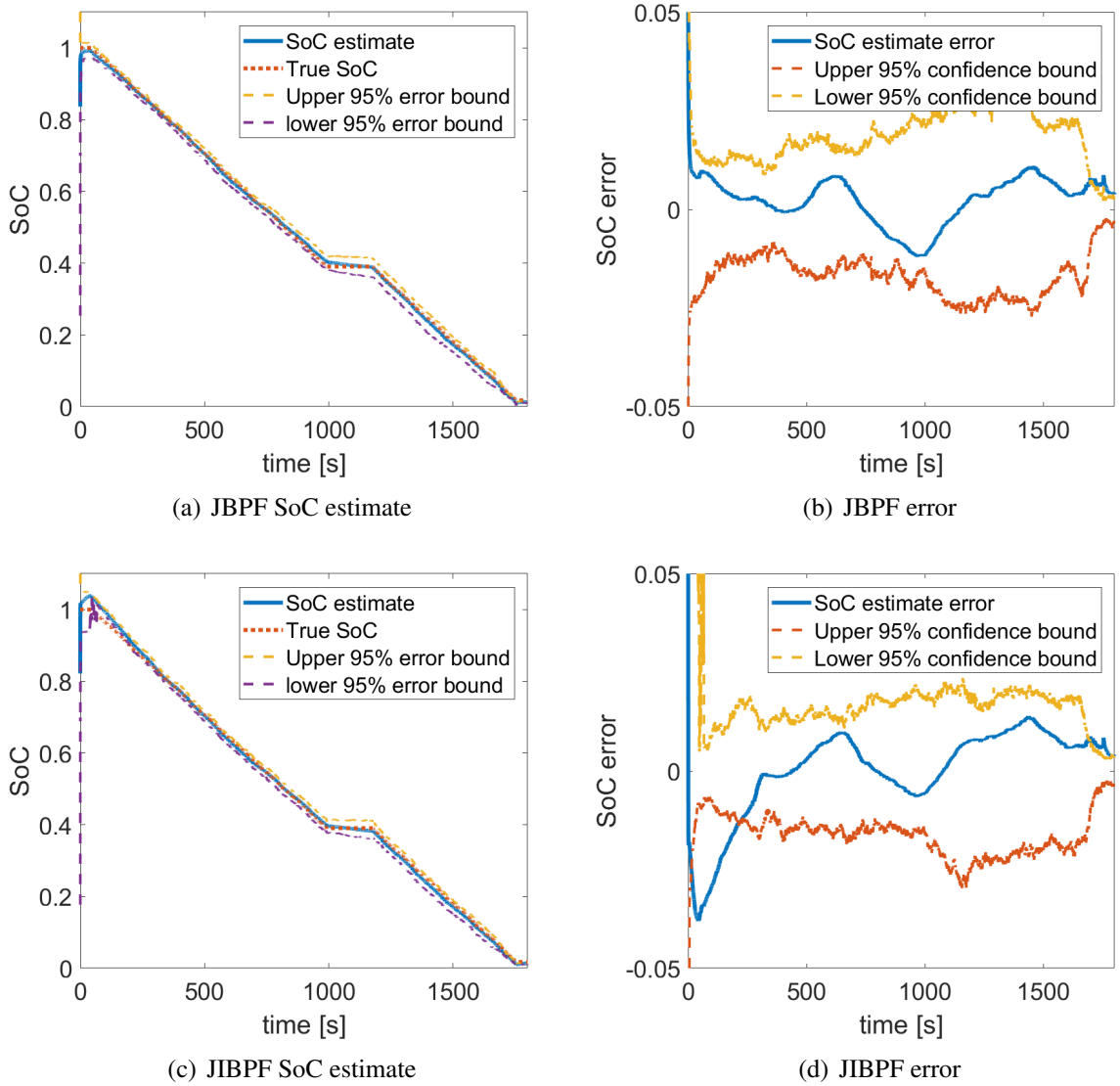


(f) SoC = 0.85, kurtosis and skew

Figure 6.6: Distribution comparisons between estimated true distribution and CDKF estimated distribution. SoC standard deviation of 3%

6.5 Estimator single run behaviour

The estimators were run once and the estimated values and their 95% error bounds were calculated according to the algorithms given in Table 5.7 and 5.8 for all states and parameters. The full state output is given in Appendix C. To emphasize the performance in terms of SoC estimation, the SoC output and error is given for each of the filters in Figure 6.7 and 6.8. The corresponding confidence bounds are given to illustrate how the error bound are related to the error.



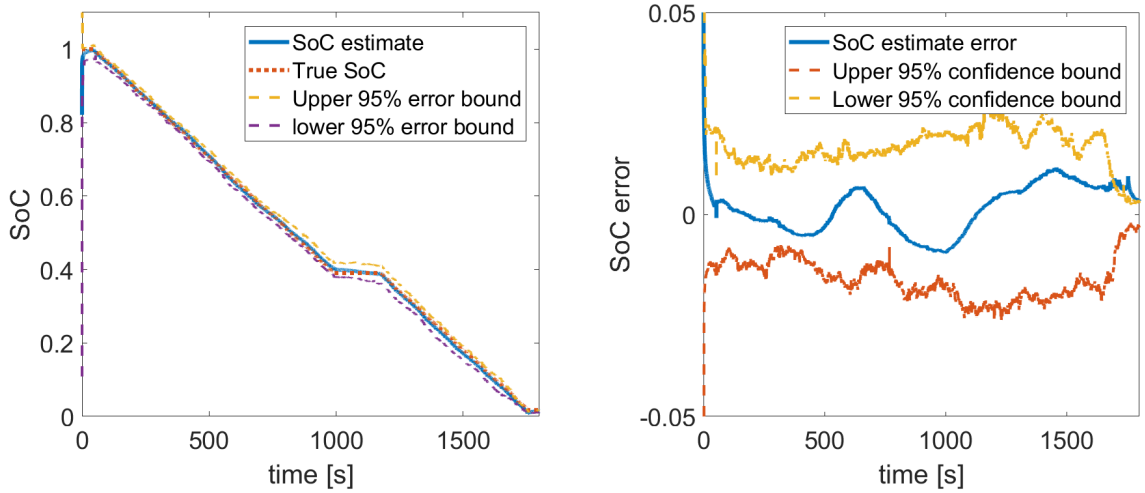
(a) JBPF SoC estimate

(b) JBPF error

(c) JIBPF SoC estimate

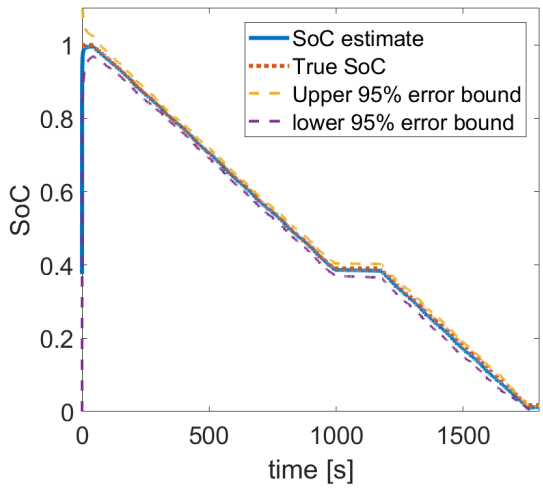
(d) JIBPF error

Figure 6.7: State of charge estimate and error with 95% error bounds, JBPF and JIBPF

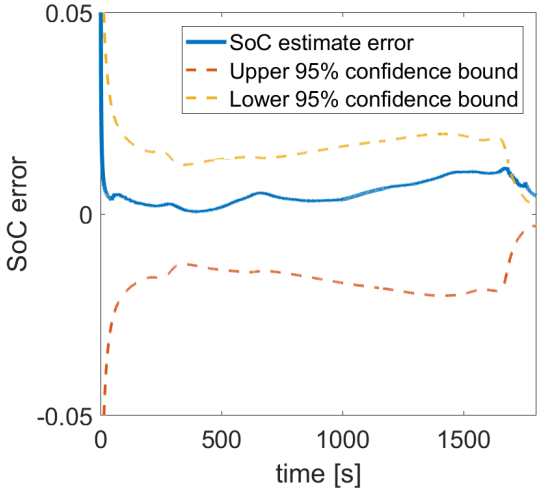


(a) JAPF SoC estimate

(b) JAPF error



(c) JCDKFSoC estimate

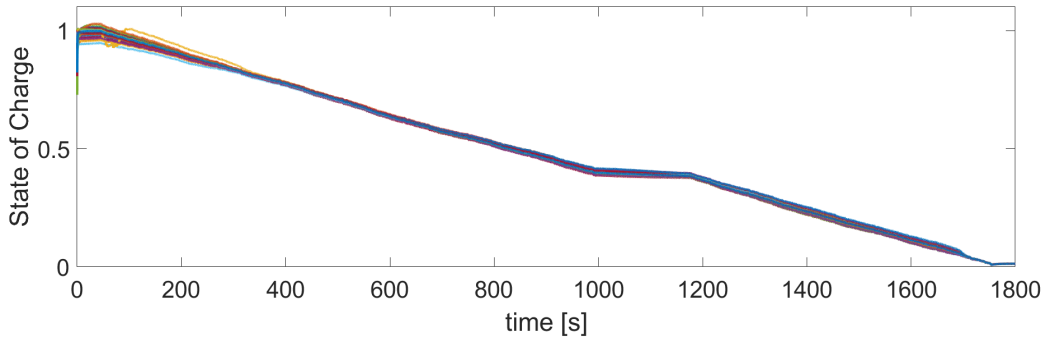


(d) JCDKF error

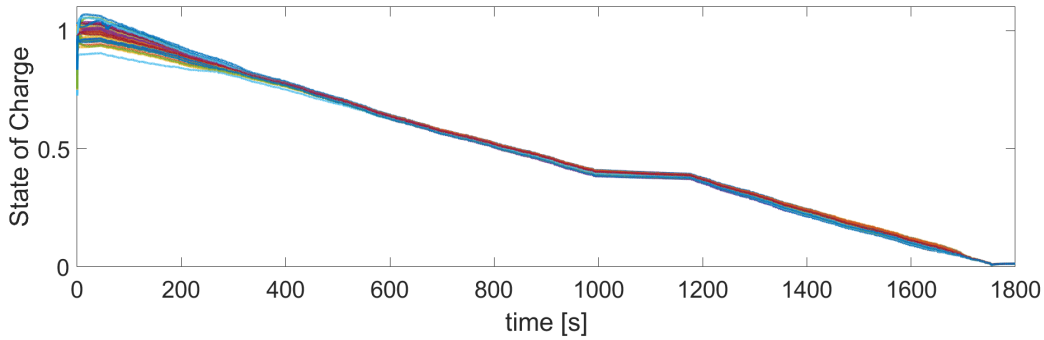
Figure 6.8: State of charge estimate with 95% error bounds

6.6 Estimators multiple run behaviour

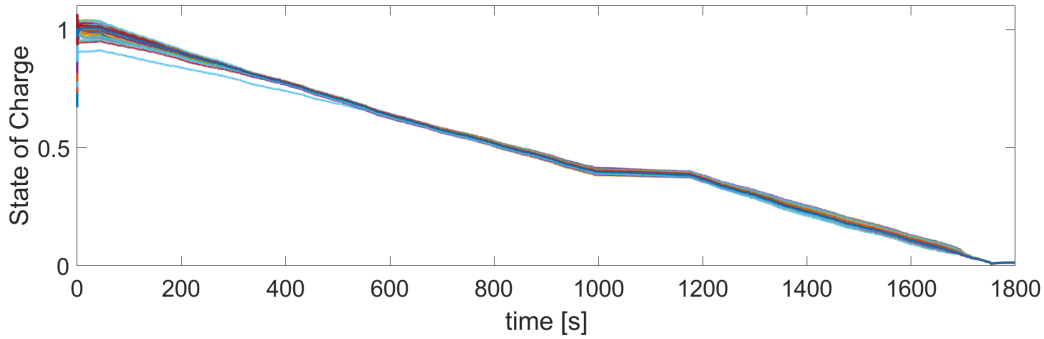
Due to the random nature of the MC method, the estimators are expected to perform differently when run multiple times. To explore this effect, the three PFs were run a total of 50 times. In Appendix D, the state outputs are shown for 20 of these runs (only 20 runs are plotted in order to preserve clarity). In Figure 6.9 however, the full 50 runs are shown for the SoC state in all the PFs.



(a) JBPF



(b) JIBPF

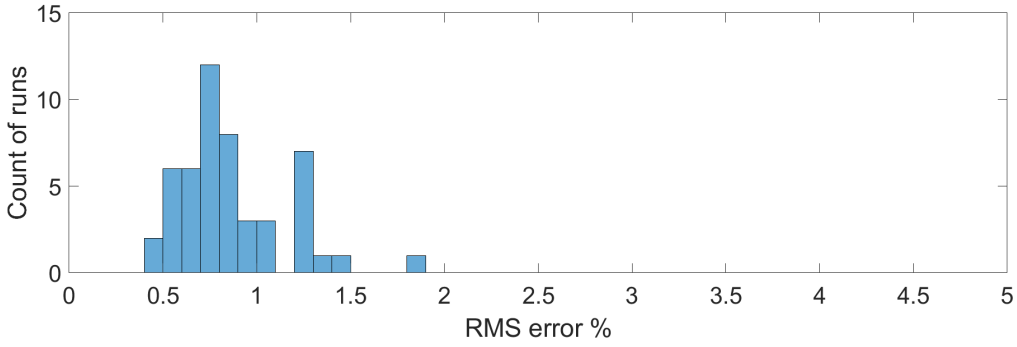


(c) JAPF

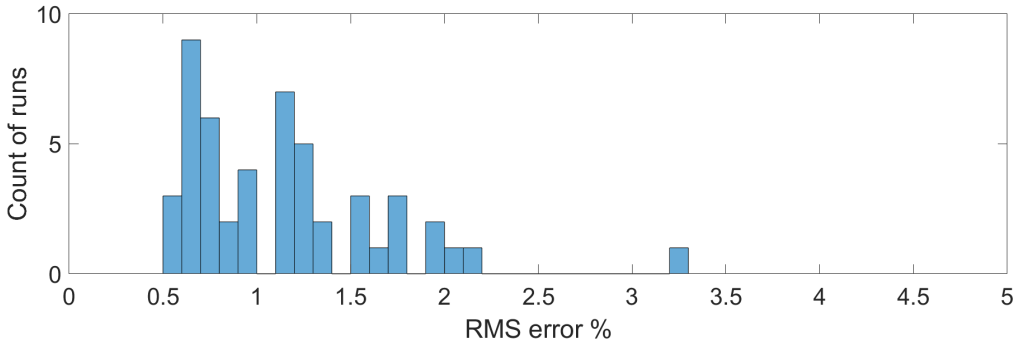
Figure 6.9: Plot of 50 different trajectories of the PFs

RMS error histogram

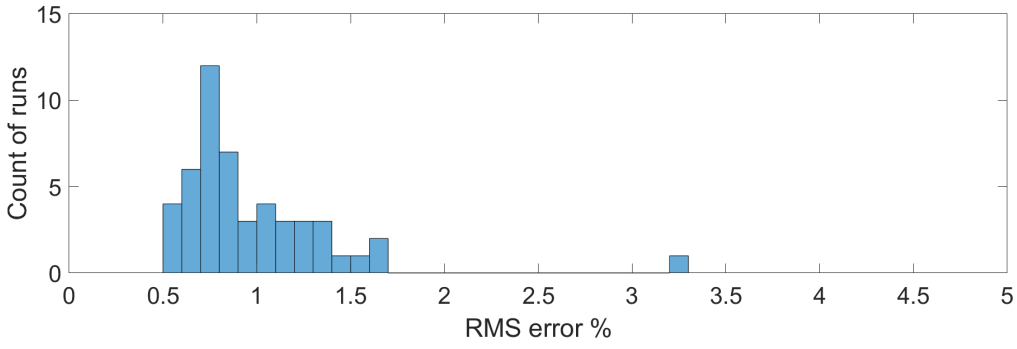
The SoC RMS error of the individual filters were calculated, and are shown in a histogram in 6.10.



(a) JBPF



(b) JIBPF

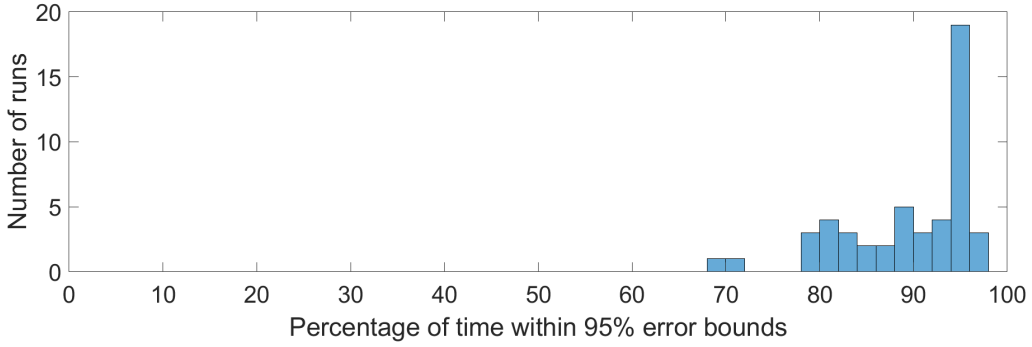


(c) JAPF

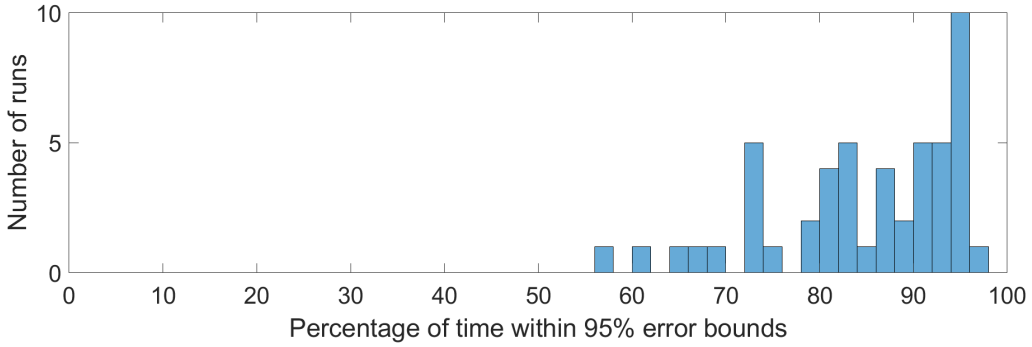
Figure 6.10: Histogram of RMS Error for 50 runs of the PFs

Error bounds correctness histogram

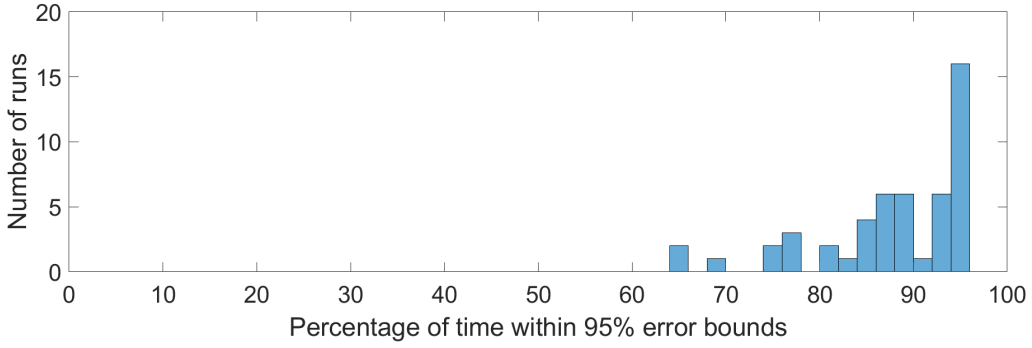
For each run of the estimators the true SoC was compared to the estimated error bounds. The percentage of time that the error bounds contained the true SoC was calculated. A percentage of 90% indicates that the error bounds contained the true SoC for 90% of the time. In Figure 6.11 the results of this calculation is shown as a histogram for each of the estimators evaluated.



(a) JBPF



(b) JIBPF

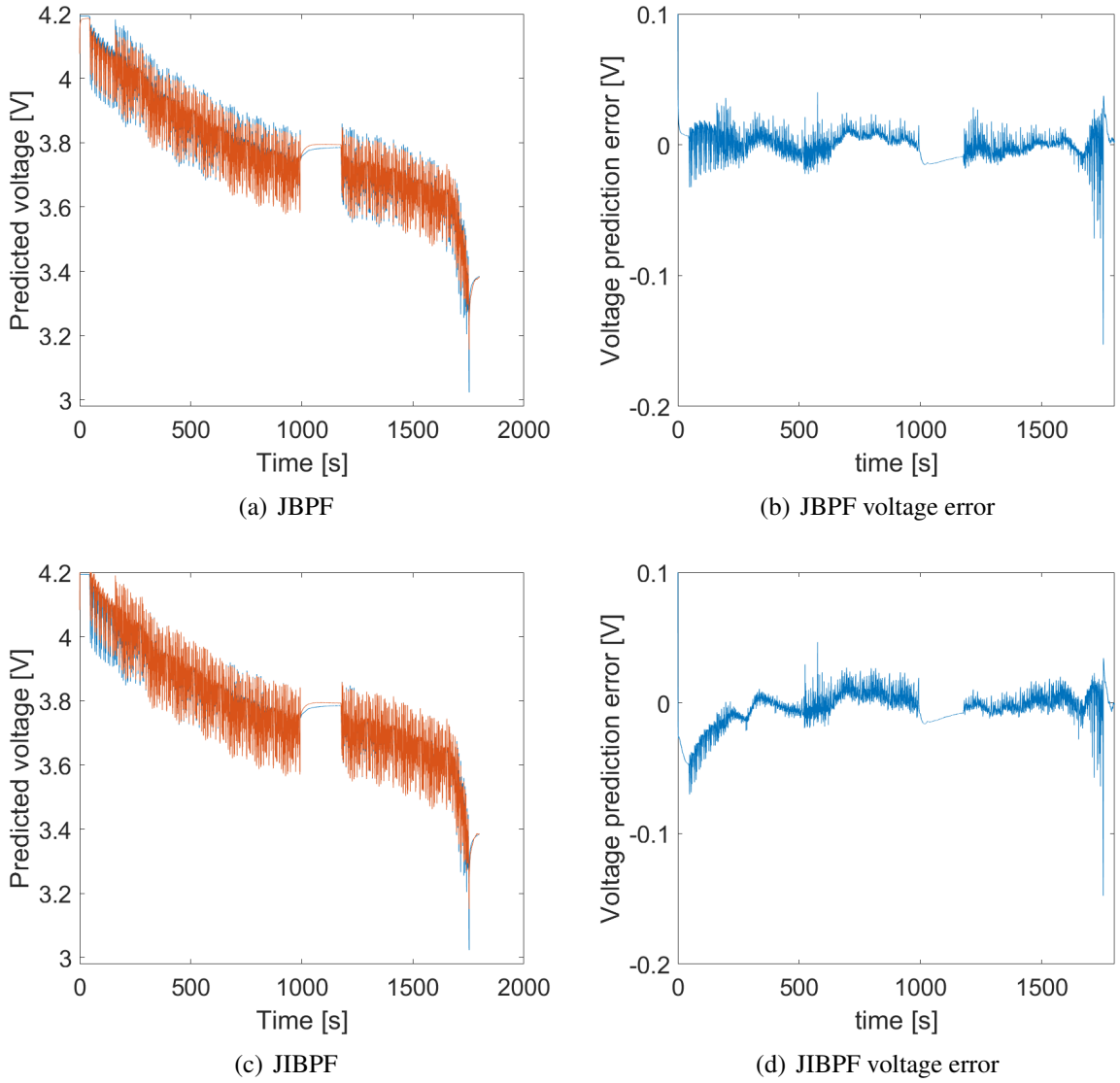


(c) JAPF

Figure 6.11: Histograms of error percentage for the implemented SMC methods

6.7 Voltage prediction performance

The estimated voltage was extracted from the filters, and the voltage and error plot is shown in Figures 6.12 and 6.13.



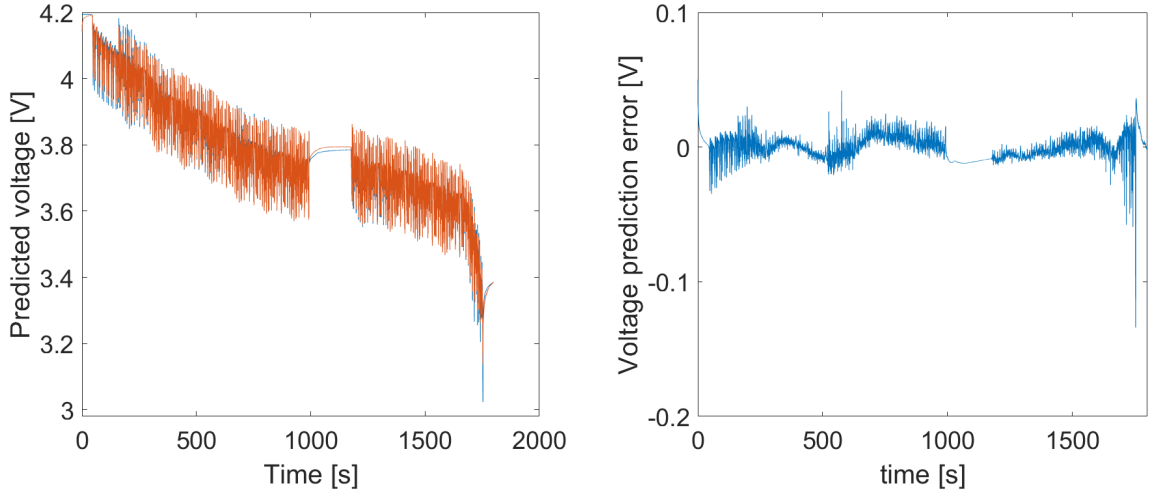
(a) JBPF

(b) JBPF voltage error

(c) JIBPF

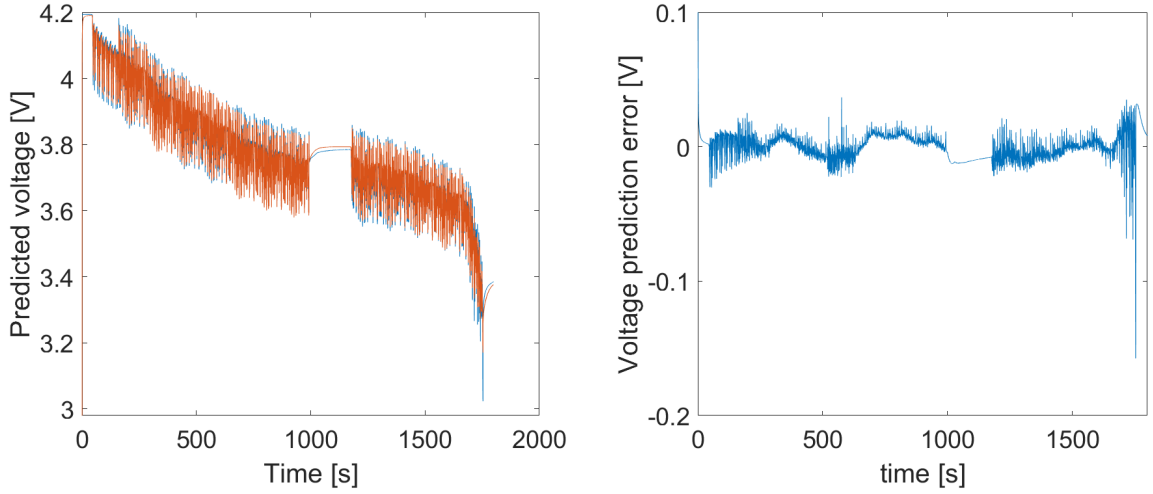
(d) JIBPF voltage error

Figure 6.12: Example voltage prediction output for JBPF and JIBPF



(a) JAPF

(b) JAPF voltage error



(c) JCDKF

(d) JCDKF voltage error

Figure 6.13: Example voltage prediction output for JAPF and JCDKF

6.8 Summarized performance all filters

Table 6.1 show the calculated average errors from running all filters 50 times and calculating the max, min and mean of the RMS error of SoC and RMS error of the predicted voltage. The JCDKF is only run once due to its deterministic behaviour.

Error statistic	**JCDKF**	**JBPF**	**JIBPF**	**JAPF**
max RMS error SoC	0.88	1.864	3.25	3.25
mean RMS error SoC	0.88	0.87	1.16	0.97
min RMS error SoC	0.88	0.49	0.56	0.52
Max RMS error V_{pred} [mV]	11.07	16.60	28.88	32.22
Mean RMS error V_{pred} [mV]	11.07	9.98	11.91	10.49
Min RMS error V_{pred} [mV]	11.07	8.29	8.23	8.322
Max % in 95% error bounds	94.53	96.84	96.22	95.46
Mean % in 95% error bounds	94.53	89.62	84.66	87.84
Min % in 95% error bounds	94.53	68.63	57.90	64.81

Table 6.1: Performance summary for the estimators summarized

Chapter 7

Discussion

It is clear from the contents of Chapter 4 that while the nonlinear KF and the SMC/PF both solve the same problem, they go about it in very different ways. While the KF assumes Gaussian distributions and thereby greatly simplifies the problem, the SMC methods use random sampling techniques to essentially allow any distribution to be used. Additionally, the SMC methods open up a wide variety of options for models to be used, as opposed to the KF which in comparison is much more restrictive. Consequently, for a given application the central question is: Is it necessary or beneficial to describe non-Gaussian distributions?

The answers to this questions will heavily depend on the specific application being discussed. In this Chapter, this is discussed on the basis of the SoC estimation problem, and the results presented in Chapter 6.

7.1 Impact of nonlinearity in the battery cell model

The properties of the system transfer and measurement likelihood functions will be of crucial importance when evaluating the impact of the nonlinearity. This is clearly seen in the example given in Figure 4.1, and is also reflected in the skew and kurtosis calculation presented in Figures 6.3 and 6.4. When comparing the skew and kurtosis plots to the SoC-OCV curve found in Figure 6.1, there is a clear connection between the smoothness of the curve and the amount of skew and kurtosis present. In particular, the SoC-OCV curve has a minor bump around SoC=85%, and a major bump around SoC=10%. These bumps correspond well with the peaks of the skew and kurtosis estimates.

From this observation, it seems apparent that the dominating nonlinearity in this application is the relationship between the SoC and the OCV. The nonlinearities from the parameters does not seem to affect the output voltage much, although a more specific study might be necessary to more firmly establish this.

It is also observed that the width of the distribution being propagated through the system has

a major impact on the distortion caused. This is clearly seen when comparing the nonlinearity analysis outputs in Figure 6.3 and 6.4, as well as comparing the plot for $SoC_{\text{start}} = 10\%$ in Figure 6.5 and 6.6. The peaks of the kurtosis and skew estimates are far more pronounced for higher values of SoC variance. In other words, for the same system the Gaussian assumption might be *valid* if the standard deviation of the important states can be assumed to be small enough but *not valid* if the standard deviations are likely to be significant. Of course, in this context it is the relative relation between 'how much' nonlinearity is present and the distributions that is important.

7.2 Weaknesses and strengths of the nonlinearity analysis approach

In the case of the analysis of nonlinearity, a major potential difficulty is the use of the MC approach for exploring the system behaviour. While this is a powerful method it relies heavily on random sampling. If the number of samples is insufficient, the results retrieved might not accurately reflect the true behaviour of the system. This is one of the reasons why the use of the Bootstrap method for determining confidence intervals was included in the analysis. However, the Bootstrap method itself also relies on random sampling and thus the fundamental problem persists: Since the Bootstrap assumes that the available set is the best one that can be had, it is also effectively limited by the (random) dataset. Given the relatively high dimensionality of the problem (9 states including states and parameters), it is likely that a very high number of particles is required to fully describe the resulting voltage distribution. This can quickly increase the computational demand. The number of particles chosen for the analysis reflects a tradeoff between computational time and assumed accuracy, but a higher number of particles might still be desirable.

Another fundamental issue is that even if given perfect knowledge of parameters such as the skew and kurtosis it is difficult to get a real sense of how much the values actually impact the performance of the estimator. In other words, at what threshold does the skew and kurtosis deviations become significant? While the method is able to give an intuitive and visual indication of the nonlinearity, it is still difficult to quantify the impact nonlinearity.

The present method of analysis only compares the output of a CDKF to the estimated true distribution, but does not show how a practical PF compares. It is not immediately clear that a practical PF with a limited number of particles will be able to perform better than a comparable KF. It is worth noting at this point that a limited number of particles might turn out to be a genuine problem in the relatively high-dimensional Joint PFs presented here. Furthermore, the analysis only shows how the KF assumption fares in light of the model, but cannot account for how this affects the feedback mechanisms in the KF (or the PF for that matter). Thus based

on the present analysis, it is difficult to conclude that the PF necessarily must outperform the KF in a Joint estimating setting. This is further complicated by the fact that the results from the various PFs does not seem to indicate any significant performance gains for SoC estimation when compared to the KF. In fact, as seen in Section 6.5 and 6.6 and Table 6.1, the presented JCDKF is seen to perform at least as well as all presented variations of the PF for both voltage prediction and SoC estimation.

Even so, the method as presented has power as a tool for visualization of model nonlinearity. The visually apparent distorting of the Normal distribution is an intuitively appealing indicator of nonlinearity. As such it might be a significant aid in gaining an intuitive grasp on the problem of nonlinearities. The method might also be able to effectively point to candidate areas of the model in which one would expect to see error being introduced.

7.3 Comparison of the KF and the PF

The single run outputs of the KF and PFs are very similar, particularly if one only considers the states of SoC and currents I_1, I_2. In terms of the states however (seen in Appendix C), the parameter estimates are quite dissimilar. Note also that the parameters vary wildly between separate runs of the same PFs, as clearly seen in Appendix D. Ignoring the intra-run variations of the PFs, The JCDKF outputs are still much smoother and less twitchy than the ones produced by the PFs. This might be a tuning issue, but the PF implementations have generally been observed to produce less smooth confidence bounds than the KF.

In general, both the PFs and KF have been observed to be very sensitive to the tuning parameters, and tuning of the PFs has turned out to be very challenging. A major reason for this is the randomized behaviour of the estimator. When tuning, a single run of the estimator is not sufficient to comment on the behaviour of the estimator in general. The estimator must be run multiple times for each tuning setup, which has made tuning very time consuming and difficult. Ideally, the PFs RMS error outputs should also be subjected to hypothesis testing in order to establish whether the estimated mean values are significant. This has not been performed in this thesis, but is suggested as an improvement for later work.

The confidence bounds of the SoC state is generally seen to be approximately symmetric, which gives some credit to the notion of assuming Gaussian noise. However, it is interesting to note that the 95% confidence bounds on the *parameter values* are generally non-symmetric. This indicates that the estimated distributions of the parameters are non-Gaussian. This is interesting because it suggests that non-Gaussian methods might have extra merit when doing parameter estimation.

In terms of average results the KF and the PFs have similar performance. As seen in Table 6.1, the JBPF has the lowest average SoC error, with the KF at a very similar error value. Similar comments apply for the voltage prediction, although the JCDKF har slightly more error in this

case. However, when considering the best and worst cases of the PFs, the situation is more nuanced. While all the PFs have minimimum RMS errors below the error of the JCDKF, they all have worst case error higher than the JCDKF. This is important, because it demonstrates that the PFs are less reliable, in the sense that the random factor in the algorithm leads to a lot of noise. Since the estimators are tested with the same waveform each time, the only source of this error is in the estimator itself.

7.4 Comment on the tuning variable settings

It is important to note that the tunings used by the estimators in this thesis are conservative, in the sense that the process noise on the SoC state is quite low. This implies that the SoC model is highly trusted, and it will only weakly be corrected from the measurement. This leaves the system less adaptable to a difference between the observed and estimated voltage, and also leaves the system less able to correct its output voltage. In other words, the estimator tunings emphasize preserving the SoC state model over matching the voltage output. This is the case for both the presented KF and PFs, and is particularly visible in the voltage outputs in Section 6.7 in the final parts of the discharge cycle. The voltage error increases noticeably in the final 100 seconds for all the estimators. It is likely that the true parameters of the real cell change very quickly towards the end, a change that the estimator is unable to follow correctly. To compensate, it is possible to increase process noise on the SoC state, but this invariably makes the estimators overcompensate on the SoC state. Another subtlety of the conservative tuning is that this leaves little room for state space exploration in the estimators. This in turn reduces the estimator ability to adapt the parameters.

There could be a variety of reasons for this problem. The 2RC ECM model might not be the best fit for this test cast, or it could be that the SoC-OCV model used is not correct enough. The cell temperature has been assumed to be constant, but in fact it rises 10-15 degrees during the test. This might impact the SoC-OCV relationship. In Figure 6.2(b), one can see a small but still significant stationary offset between the model output voltage and the true voltage. This could be the result of incorrectly modelled polarization voltage, but it might also be the result of modelling elements that are entirely missing. An obvious candidate is hysteresis, mentioned briefly in Section 3.4. A rudimentary model of hysteresis was implemented in the preliminary work on this thesis. This was ultimately rejected because the model used was not able to increase the accuracy over the simpler 2RC model. This could indicate that a different model is required for the cell altogether to achieve better prediction performance.

Increasing noise on the SoC state also tends to lead to very noisy SoC estimates in the PFs. This might be caused by the choice of the proposal density $q(x_n|x_{n-1})$, which is simply the previous state with added random noise. Increasing this noise makes the particles jump around, creating the noisy SoC. Other ways of formulating $q(x_n|x_{n-1})$ might be desirable to handle this

problem. One such approach is the Unscented Particle Filter, which uses several parallel KFs to generate the proposal density. A variant of the Unscented Particle Filter was implemented early in the course of this work but was not included here as it was very computationally demanding and did not appear to provide any significant benefit. However, it was only applied to a state vector without the augmented parameters. The benefit might have been bigger in the higher-dimensional augmented system.

7.5 Evaluation of the parameter estimation approach

The main focus of this thesis has been on the behaviour of the PF when compared to the non-linear KF when used for SoC estimation. Parameter estimation was included mainly to provide a more realistic case for comparison. The major weakness with the approach is the lack of any knowledge of what the 'true value' of the parameters should optimally be. As noted in Section 3.5, it can be difficult to draw parallels between the parameters and the real condition of the cell, so the 'true values' might instead be gathered by looking at the behaviour of the cell in short windows of SoC. Determining the parameter values would require extensive testing, which unfortunately was not possible during the course of this thesis work. Thus, the specific estimated value of the parameters are of much less interest than the overall behaviour of the KFs.

7.6 Evaluation of the imprecise modelling approach

The inclusion of the JIBPF in this thesis was motivated by a desire to illustrate a different approach to measurement modelling than what has traditionally been used in PFs for SoC estimation. In practice, the JIBPF is seen to perform worse than the JBPF when the process noise is tuned identically. Again tuning is key to the performance of the estimators and a major weakness of the imprecise modelling approach in this context is that it adds another set of tuning variables that must be carefully considered. Given the lack of added performance, it is difficult to directly recommend this approach in the SoC estimation problem, but it might prove to be far more useful in other applications.

Chapter 8

Further work

This work has focused on comparing the Particle Filter and the Kalman Filter for a comparatively simple use case: SoC estimation with an electrical Equivalent Circuit Model. Augmentation of the state vector was added to increase the realism of the use case, but the scope has been deliberately kept simple. As such, there are numerous other avenues that can be considered.

One alternative is to further explore (joint) parameter estimation with PFs. Such work could also emphasize cell modelling and might involve using more advanced models such as the one proposed by Ørjan Gjengedal [12] using either complex or simple variants of PFs. Using MC methods to estimate the true distribution of parameters might also reveal whether or not the KF approach has merit for parameter estimation.

It could also be interesting to investigate Particle Markov chain Monte Carlo (pMCMC) methods for parameter estimation. pMCMC is a relatively new MC method proposed by Andrieu, Doucet and Holenstein [47] that combines Markov Chain Monte Carlo (MCMC) and SMC. It might be more suited for high-dimensional problems than pure SMC.

Note also that temperature and current dependence of the resistance elements at lower temperatures remain unexplored in this work. This still provides an interesting test case for a comparison between a PF and a KF if the proper testing can be performed.

More advanced approaches such as the one proposed by Tulsyan et. al [30] can also be explored. This involves applying the PF directly to a reduced-order electrochemical cell model. While potenetially very accurate, this requires a fairly deep exploration of electrochemical modelling of battery cells.

Chapter 9

Conclusion

In this thesis, the problem of estimating SoC in a LiBs under a highly dynamic current load has been examined. The basics of battery cell chemistry and cell modelling techniques were reviewed, and a simple 2RC ECM was developed for use in an estimator. After a review of Bayesian estimation, the fundamentals of the nonlinear KF and SMC methods was presented and contrasted. An imprecise modelling technique was also reviewed in some detail.

The Lithium Ion battery system exhibit varying amounts of nonlinearity, depending on how much uncertainty is used when propagating the particles. The nonlinearities are most pronounced for state vectors with SoC values below approximately 12%. Based on the nonlinearities observed, the JCDKF was expected to perform worse than the PFs, but in testing the KF performed similarly to the PFs. This might be attributed to tuning issues, and testing with a more advanced model could still reveal greater differences.

The augmented state and parameter JBPF, JIBPF, JAPF and JCDKF was tested on a current and voltage waveform imitating a real-life use case from a Formula Student race car. While all the PFs have single runs with higher accuracy than the JCDKF, on average the JCDKF performs similarly to the PF variants. The JIBPF is less accurate than the simpler JBPF in terms of both SoC estimation and voltage prediction. The difference can partially be attributed to tuning but it still indicates that there might not be much to gain from introducing non-standard measurement models in the SoC estimation problem. The tunings presented may be too conservative, and a more accurate model might be required to further compare the methods.

The parameter estimates of the implemented PFs are seen to act randomly. One possible reason for this could be that the estimators are run with a relatively low number of particles. This might be too few particles to be able to accurately represent the likelihoods of all 9 states in the augmented state vector. Due to the distinctly nonsymmetric parameter confidence bounds produced by the PFs, it is hypothesized that the Monte Carlo approach might be better suited for parameter estimation than the KF. However, this might require a different approach than the one taken in this thesis.

Bibliography

[1] Yoshio Nishi. "Past, Present and Future of Lithium-Ion Batteries: Can New Technologies Open up New Horizons?" In: *Lithium-Ion Batteries*. Ed. by Gianfranco Pistoia. Amsterdam: Elsevier, 2014, pp. 21–39. ISBN: 978-0-444-59513-3. DOI: 10.1016/B978-0-444-59513-3.00002-9.

[2] Eirik Børsheim and Martin Standal Skåravik. "Energy Storage, and Design of Tractive System for EV Application". Master Thesis. Norwegian University of Science and Technology, 2014.

[3] Martin Winter and Ralph J. Brodd. "What Are Batteries, Fuel Cells, and Supercapacitors?" In: *Chemical Reviews* 104.10 (2004), pp. 4245–4270. DOI: 10.1021/cr020730k. pmid: 15669155.

[4] G. L. Plett. "High-Performance Battery-Pack Power Estimation Using a Dynamic Cell Model". In: *IEEE Transactions on Vehicular Technology* 53.5 (Sept. 2004), pp. 1586–1593. ISSN: 0018-9545. DOI: 10.1109/TVT.2004.832408.

[5] Wladislaw Waag, Christian Fleischer, and Dirk Uwe Sauer. "Critical Review of the Methods for Monitoring of Lithium-Ion Batteries in Electric and Hybrid Vehicles". In: *Journal of Power Sources* 258 (Supplement C July 15, 2014), pp. 321–339. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2014.02.064.

[6] Wladislaw Waag and Dirk Uwe Sauer. "Adaptive Estimation of the Electromotive Force of the Lithium-Ion Battery after Current Interruption for an Accurate State-of-Charge and Capacity Determination". In: *Applied Energy* 111 (Supplement C Nov. 1, 2013), pp. 416–427. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2013.05.001.

[7] Gregory L. Plett. "Extended Kalman Filtering for Battery Management Systems of LiPB-Based HEV Battery Packs: Part 1. Background". In: *Journal of Power Sources* 134.2 (Aug. 12, 2004), pp. 252–261. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2004.02.031.

[8] Gregory L. Plett. “Extended Kalman Filtering for Battery Management Systems of LiPB-Based HEV Battery Packs: Part 2. Modeling and Identification”. In: *Journal of Power Sources* 134.2 (Aug. 12, 2004), pp. 262–276. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2004.02.032.

[9] Gregory L. Plett. “Extended Kalman Filtering for Battery Management Systems of LiPB-Based HEV Battery Packs: Part 3. State and Parameter Estimation”. In: *Journal of Power Sources* 134.2 (Aug. 12, 2004), pp. 277–292. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2004.02.033.

[10] Gregory L. Plett. “Sigma-Point Kalman Filtering for Battery Management Systems of LiPB-Based HEV Battery Packs: Part 1: Introduction and State Estimation”. In: *Journal of Power Sources* 161.2 (Oct. 27, 2006), pp. 1356–1368. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2006.06.003.

[11] Gregory L. Plett. “Sigma-Point Kalman Filtering for Battery Management Systems of LiPB-Based HEV Battery Packs: Part 2: Simultaneous State and Parameter Estimation”. In: *Journal of Power Sources* 161.2 (Oct. 27, 2006), pp. 1369–1384. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2006.06.004.

[12] Ørjan Gjengedal. “Real Time Impedance Identification of Li-Polymer Battery with Kalman Filter”. Master Thesis. Norwegian University of Science and Technology.

[13] Simon Schwunk, Nils Armbruster, Sebastian Straub, et al. “Particle Filter for State of Charge and State of Health Estimation for Lithium–iron Phosphate Batteries”. In: *Journal of Power Sources* 239 (Supplement C Oct. 1, 2013), pp. 705–710. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2012.10.058.

[14] M. F. Samadi, S. M. M. Alavi, and M. Saif. “An Electrochemical Model-Based Particle Filter Approach for Lithium-Ion Battery Estimation”. In: *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*. 2012 IEEE 51st IEEE Conference on Decision and Control (CDC). Dec. 2012, pp. 3074–3079. DOI: 10.1109/CDC.2012.6426009.

[15] V. Pathuri Bhuvana, C. Unterrieder, and M. Huemer. “Battery Internal State Estimation: A Comparative Study of Non-Linear State Estimation Algorithms”. In: *2013 IEEE Vehicle Power and Propulsion Conference (VPPC)*. 2013 IEEE Vehicle Power and Propulsion Conference (VPPC). Oct. 2013, pp. 1–6. DOI: 10.1109/VPPC.2013.6671666.

[16] Kristin L. Sainani. “A Closer Look at Confidence Intervals”. In: *PM&R* 3.12 (Dec. 1, 2011), pp. 1134–1141. ISSN: 1934-1482. DOI: 10.1016/j.pmrj.2011.10.005.

[17] Ronald E. Walpole, Raymond H. Myers, Sharon L. Myers, et al. *Probability & Statistics for Engineers and Scientists*. Ninth international version. Pearson. ISBN: 0-321-74823-9.

[18] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Monographs on statistics and applied probability 26. Chapman & Hall, 1986. ISBN: 0-412-24620-1.

[19] Art B. Owen. *Empirical Likelihood*. Monographs on statistics and applied probability 92. Chapman & Hall, 2001. ISBN: 1-58488-071-6.

[20] Roger W. Johnson. "An Introduction to the Bootstrap". In: *Teaching Statistics* 23.2 (June 1, 2001), pp. 49–54. ISSN: 1467-9639. DOI: 10.1111/1467-9639.00050.

[21] Aditya Tulsyan, R. Bhushan Gopaluni, and Swanand R. Khare. "Particle Filtering without Tears: A Primer for Beginners". In: *Computers & Chemical Engineering* 95 (Dec. 5, 2016), pp. 130–145. ISSN: 0098-1354. DOI: 10.1016/j.compchemeng.2016.08.015.

[22] R. Spotnitz. "Lithium-Ion Batteries: The Basics". In: *Chemical Engineering Progress* 109.10 (2013), pp. 39–43. ISSN: 0360-7275.

[23] Christian Julien, Alain Mauger, Ashok Vijh, et al. "Lithium Batteries". In: *Lithium Batteries*. Springer, Cham, 2016, pp. 29–68. ISBN: 978-3-319-19108-9. DOI: 10.1007/978-3-319-19108-9_2.

[24] Christian Julien, Alain Mauger, Ashok Vijh, et al. "Basic Elements for Energy Storage and Conversion". In: *Lithium Batteries*. Springer, Cham, 2016, pp. 1–27. ISBN: 978-3-319-19108-9. DOI: 10.1007/978-3-319-19108-9_1.

[25] Alexander Farmann and Dirk Uwe Sauer. "A Study on the Dependency of the Open-Circuit Voltage on Temperature and Actual Aging State of Lithium-Ion Batteries". In: *Journal of Power Sources* 347 (Apr. 15, 2017), pp. 1–13. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2017.01.098.

[26] B. Pattipati, B. Balasingam, G. V. Avvari, et al. "Open Circuit Voltage Characterization of Lithium-Ion Batteries". In: *Journal of Power Sources* 269 (Dec. 10, 2014), pp. 317–333. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2014.06.152.

[27] Anup Barai, W. Dhammika Widanage, James Marco, et al. "A Study of the Open Circuit Voltage Characterization Technique and Hysteresis Assessment of Lithium-Ion Cells". In: *Journal of Power Sources* 295 (Supplement C Nov. 1, 2015), pp. 99–107. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2015.06.140.

[28] Tarun Huria, Massimo Ceraolo, Javier Gazzarri, et al. *Simplified Extended Kalman Filter Observer for SOC Estimation of Commercial Power-Oriented LFP Lithium Battery Cells*. SAE Technical Paper 2013-01-1544. Warrendale, PA: SAE International, Apr. 8, 2013. DOI: 10.4271/2013-01-1544.

[29] James L. Lee, Andrew Chemistruck, and Gregory L. Plett. "One-Dimensional Physics-Based Reduced-Order Model of Lithium-Ion Dynamics". In: *Journal of Power Sources* 220 (Supplement C Dec. 15, 2012), pp. 430–448. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2012.07.075.

[30] Aditya Tulsyan, Yiting Tsai, R. Bhushan Gopaluni, et al. “State-of-Charge Estimation in Lithium-Ion Batteries: A Particle Filter Approach”. In: *Journal of Power Sources* 331 (Nov. 1, 2016), pp. 208–223. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2016.08.113.

[31] Xueyan Li, Meng Xiao, and Song-Yul Choe. “Reduced Order Model (ROM) of a Pouch Type Lithium Polymer Battery Based on Electrochemical Thermal Principles for Real Time Applications”. In: *Electrochimica Acta* 97 (Supplement C May 1, 2013), pp. 66–78. ISSN: 0013-4686. DOI: 10.1016/j.electacta.2013.02.134.

[32] Arnaud Doucet and A. M. Johansen. *A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later*. Technical report. Department of Statistics, University of British Columbia, Dec. 2008.

[33] M. S. Arulampalam, S. Maskell, N. Gordon, et al. “A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking”. In: *IEEE Transactions on Signal Processing* 50.2 (Feb. 2002), pp. 174–188. ISSN: 1053-587X. DOI: 10.1109/78.978374.

[34] R. E. Kalman. “A New Approach to Linear Filtering and Prediction Problems”. In: *Journal of Basic Engineering* 82.1 (Mar. 1, 1960), pp. 35–45. ISSN: 0098-2202. DOI: 10.1115/1.3662552.

[35] Rudolph Van der Merwe. “Sigma-Point Kalman Filters for Probabilistic Inference in Dynamic State-Space Models”. In: *Scholar Archive* 8 (Apr. 1, 2004). URL: https://digitalcommons.ohsu.edu/etd/8.

[36] Kristian Roaldsnes. *Investigation into Particle Filter for Battery State of Charge Estimation*. Department of Engineering Cybernetics, Norwegian University of Science and Technology, 2017.

[37] Robert Grover Brown and Patrick Y.C. Hwang. *Introduction to Random Signals and Applied Kalman Filtering*. Fourth Edition. John Wiley & Sons Inc., 2014. ISBN: 978-0-470-60969-9.

[38] Branko Ristic. *Particle Filters for Random Set Models*. New York, NY: Springer New York, 2013. ISBN: 978-1-4614-6316-0. DOI: 10.1007/978-1-4614-6316-0.

[39] R. Van Der Merwe and Arnaud Doucet. *The Unscented Particle Filter*. Techincal report CUED/F-INFENG/TR 380. Cambridge University Engineering Department, Aug. 16, 2000, p. 46.

[40] Randal Douc, Olivier Cappé, and Eric Moulines. “Comparison of Resampling Schemes for Particle Filtering”. In: (July 8, 2005). arXiv: cs/0507025.

[41] B. Ristic. “Bayesian Estimation With Imprecise Likelihoods: Random Set Approach”. In: *IEEE Signal Processing Letters* 18.7 (July 2011), pp. 395–398. ISSN: 1070-9908. DOI: 10.1109/LSP.2011.2152392.

[42] Ronald P.S. Mahler. *Statistical Multisource-Multitarget Information Fusion*. Artech House. ISBN: 978-1-59693-092-6.

[43] Xiaosong Hu, Shengbo Li, and Huei Peng. “A Comparative Study of Equivalent Circuit Models for Li-Ion Batteries”. In: *Journal of Power Sources* 198 (Supplement C Jan. 15, 2012), pp. 359–367. ISSN: 0378-7753. DOI: 10.1016/j.jpowsour.2011.10.013.

[44] Chi-Tsong Chen. *Linear System Theory and Design*. International 4th edition. Oxford University Press, 2014. ISBN: 978-0-19-996454-3.

[45] Wladislaw Waag, Stefan Käbitz, and Dirk Uwe Sauer. “Experimental Investigation of the Lithium-Ion Battery Impedance Characteristic at Various Conditions and Aging States and Its Influence on the Application”. In: *Applied Energy*. Special Issue on Advances in sustainable biofuel production and use - XIX International Symposium on Alcohol Fuels - ISAF 102 (Feb. 1, 2013), pp. 885–897. ISSN: 0306-2619. DOI: 10.1016/j.apenergy.2012.09.030.

[46] Wei Cai and Jun Wang. “Estimation of Battery State-of-Charge for Electric Vehicles Using an MCMC-Based Auxiliary Particle Filter”. In: *2016 American Control Conference (ACC)*. 2016 American Control Conference (ACC). July 2016, pp. 4018–4021. DOI: 10.1109/ACC.2016.7525541.

[47] Andrieu Christophe, Doucet Arnaud, and Holenstein Roman. “Particle Markov Chain Monte Carlo Methods”. In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72.3 (May 20, 2010), pp. 269–342. ISSN: 1369-7412. DOI: 10.1111/j.1467-9868.2009.00736.x.

Appendix A

Linear Kalman Filter algorithm

For $k = 0$

$$\hat{\mathbf{x}}_0^- = E(\mathbf{x}) \qquad \mathbf{P}_0^- = E(\mathbf{P}_0)$$

For $k \geq 1$

Compute Kalman Gain

$$\mathbf{K}_k = \mathbf{P}_k^- \mathbf{C}_{d,k}^T (\mathbf{C}_{d,k} \mathbf{P}_k^- \mathbf{C}_{d,k}^T + \mathbf{R}_k)^{-1}$$

Update estimate with measurement

$$\hat{\mathbf{x}}_k = \hat{\mathbf{x}}_k^- + \mathbf{K}_k (\mathbf{z}_k - \mathbf{C}_{d,k} \hat{\mathbf{x}}_k^-)$$

Compute error covariance for updated estimate

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{C}_{d,k}) \mathbf{P}_k^-$$

Predict ahead

$$\begin{aligned} \hat{\mathbf{x}}_{k+1}^- &= \mathbf{A}_d \mathbf{x} \\ \mathbf{P}_{k+1}^- &= \mathbf{A}_d \mathbf{P}_k \mathbf{A}_d^T + \mathbf{Q}_k \end{aligned}$$

Table A.1: LKF algorithm, adapted from Brown & Hwang [37, p.147].

Appendix B

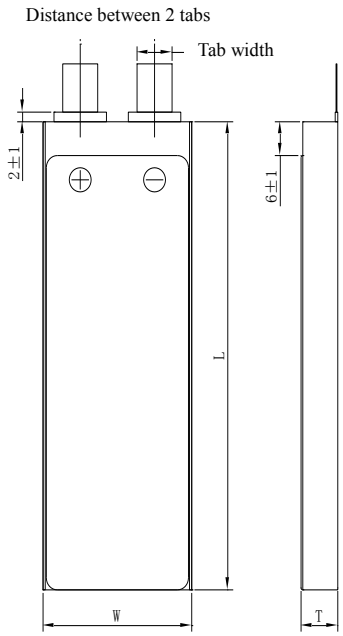
Cell data sheet excerpt

2. 型号 MODEL

SLPBB042126 6550mAh 10C 3.7V

3. 产品规格 SPECIFICATION

单颗电池规格 Specifications of single cell

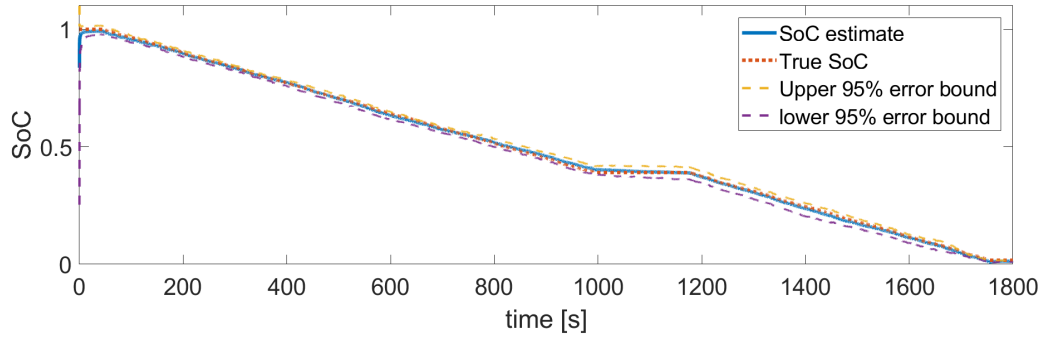


◆标称容量 Typical Capacity①		6.55Ah
◆标称电压 Nominal Voltage		3.7V
◆ 充电条件 Charge Condition	最大电流 Max. Continuous charge Current	13.1A
	峰值充电 Peak charge current	26.2A(≤1sec)
	电压 Voltage	4.2V±0.03V
◆ 放电条件 Discharge Condition	Max Continuous Discharge Current	65.5A
	Peak Discharge Current	98.25A
	Cut-off Voltage	3.0V
◆交流内阻 AC Impedance(mOHM)		<3.0
◆循环寿命【充电:1.0C,放电:10C】 Cycle Life【CHA:1.0C,DCH:10C】		>100cycles
◆使用温度 Operating Temp.	充电 Charge	0℃~45℃
	放电 Discharge	-20℃~60℃
◆ 电芯尺寸 Cell Dimensions	厚度 Thickness(T)	10.7±0.3mm
	宽度 Width(W)	42±0.5mm
	长度 Length(L)	127.5±0.5mm
	极耳间距 Distance between 2 tabs	21±1mm
◆ 极耳尺寸 Dimensions of Cell tabs	极耳宽度 Tab Width	12mm
	极耳厚度 Tab Thickness	0.2mm
	极耳长度 Tab Length	Max 30mm
◆重量 Weight(g)		128.5±3.0
①标称容量：0.5CmA,4.2V~3.0V@23℃±2℃ Typical Capacity:0.5CmA,4.2V~3.0V@23℃±2℃		

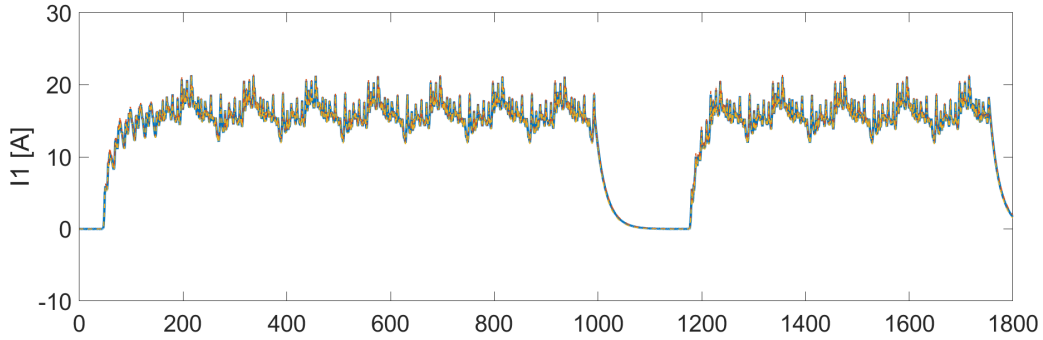
Appendix C

Estimator single run results, all states

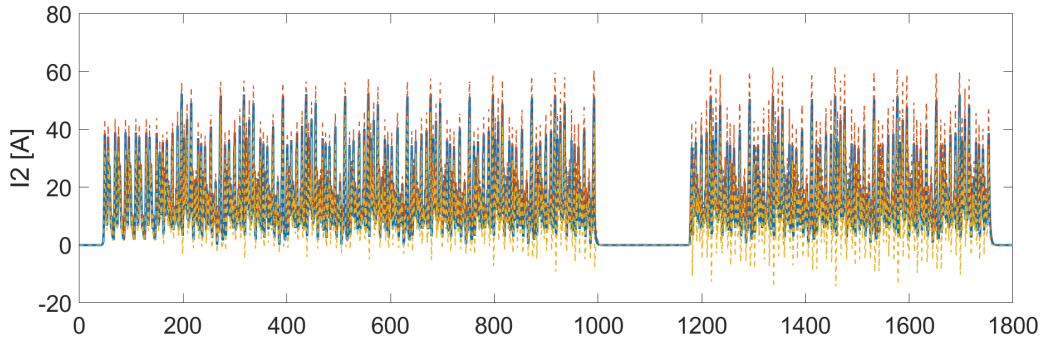
C.1 JBPF single run results



(a) SoC

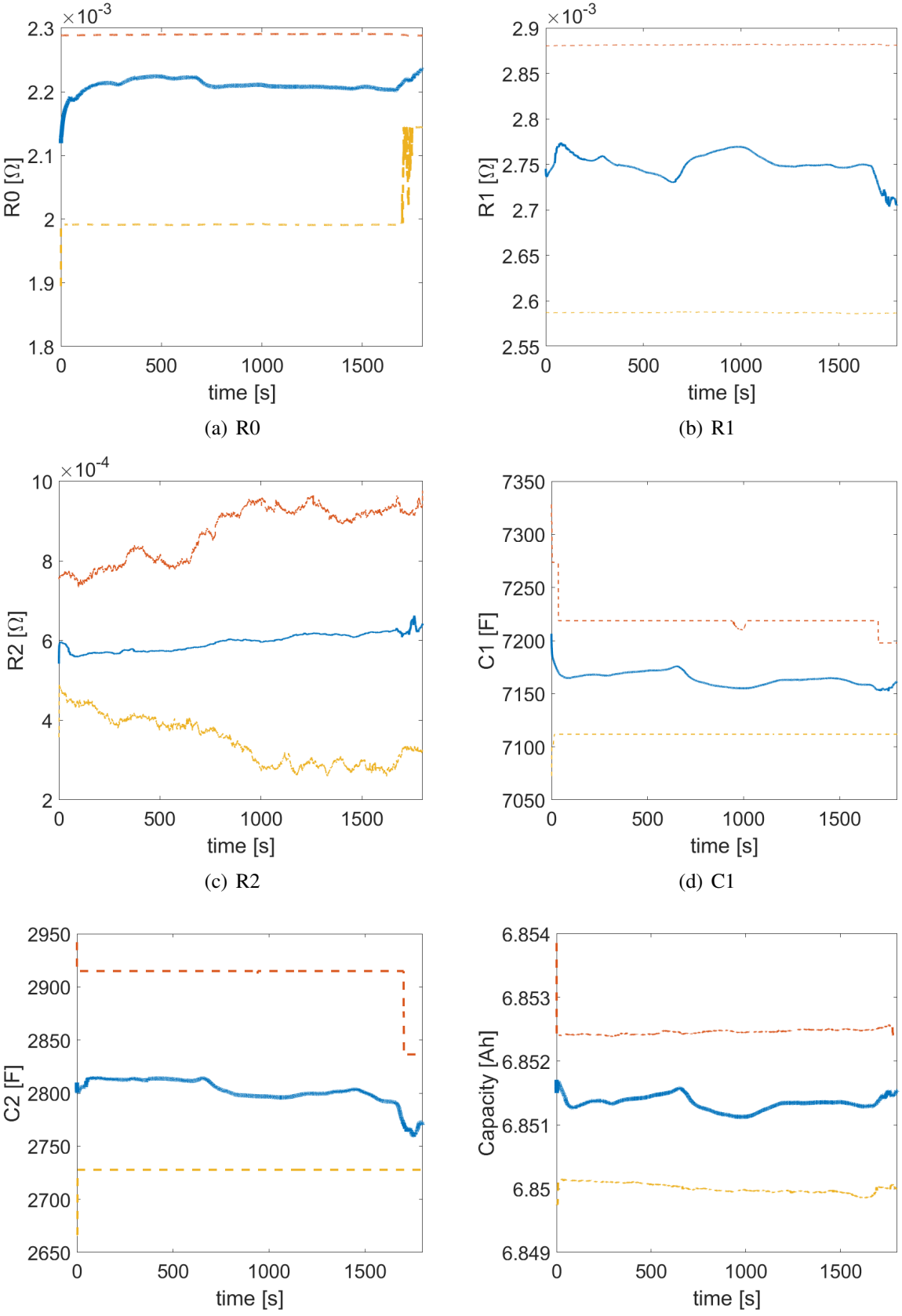


(b) I1



(c) I1

Figure C.1: JBPF single run state results with 95% error bounds on each state



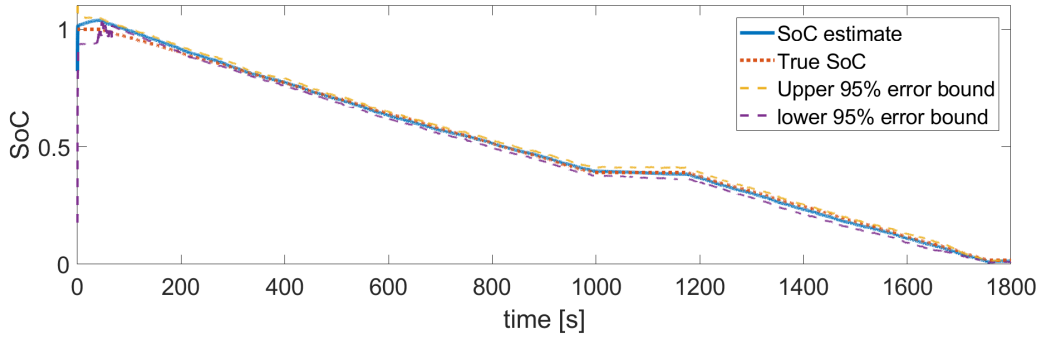
(a) R0 (b) R1

(c) R2 (d) C1

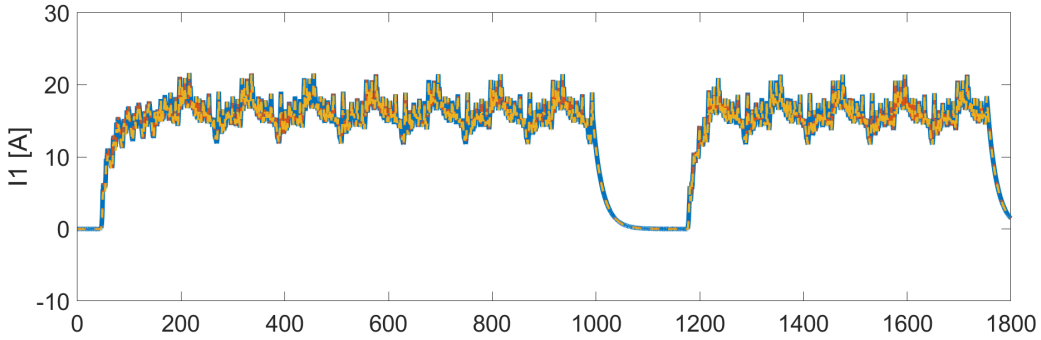
(e) C2 (f) Capacity

Figure C.2: JBPF single run parameter results with 95% error bounds on each state

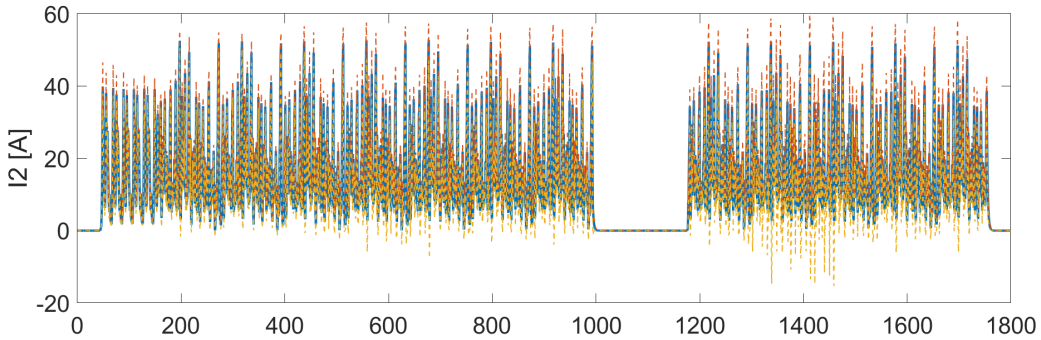
C.2 JIBPF single run results



(a) SoC

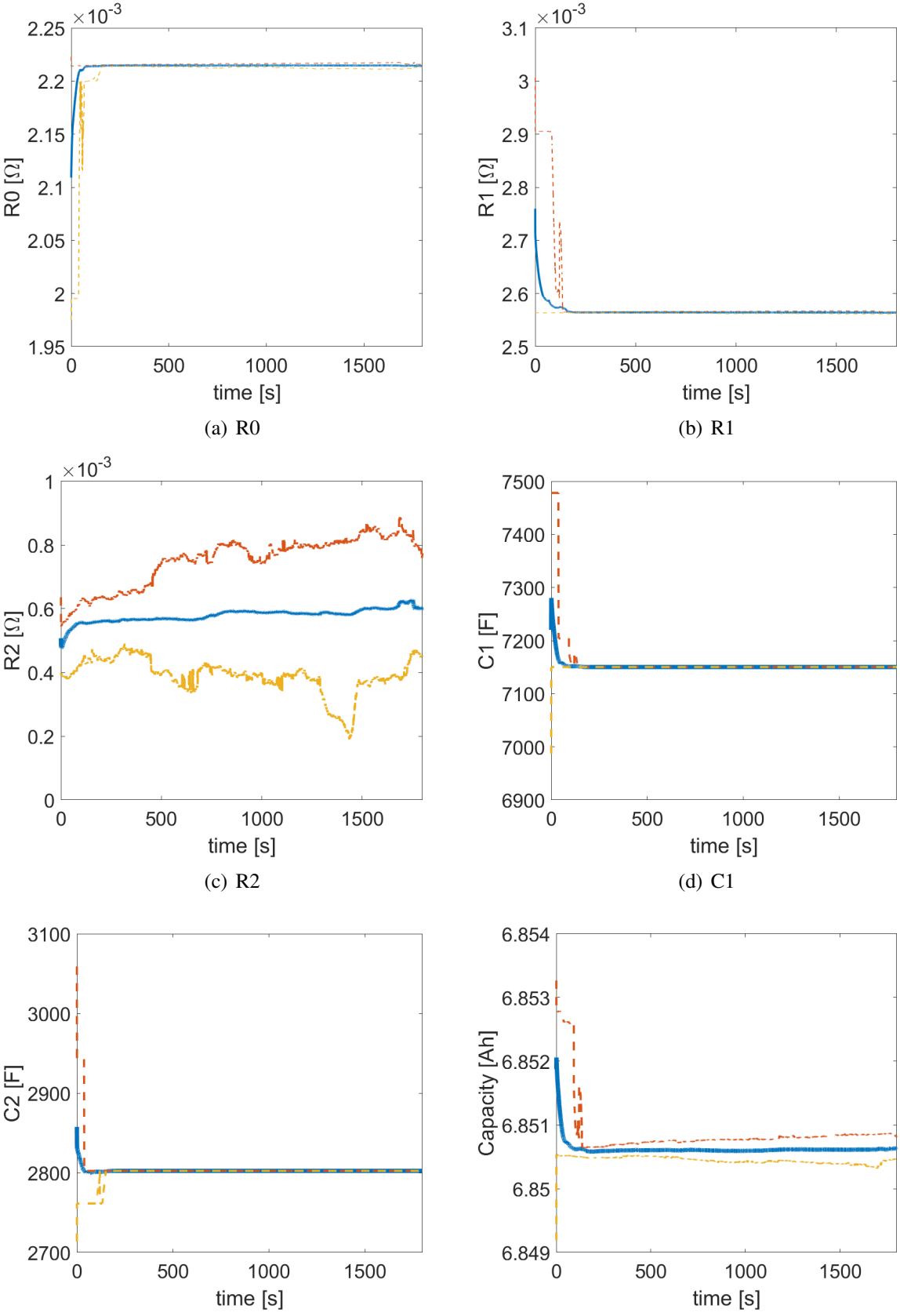


(b) I1



(c) I1

Figure C.3: JIBPF single run state results with 95% error bounds on each state



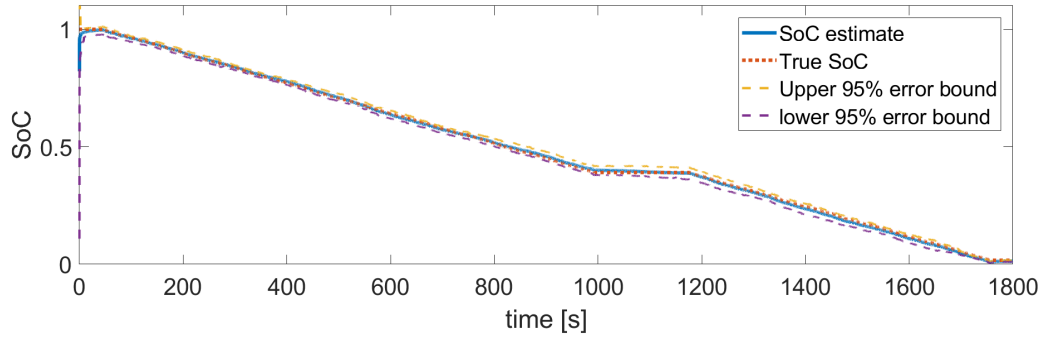
(a) R0 (b) R1

(c) R2 (d) C1

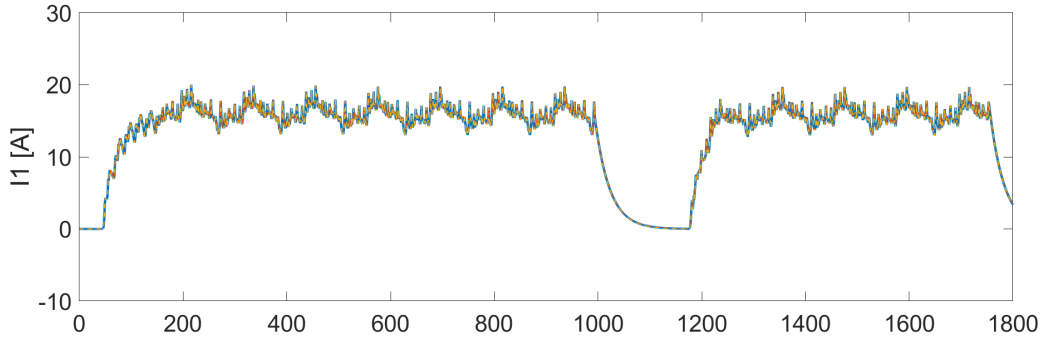
(e) C2 (f) Capacity

Figure C.4: JIBPF single run parameter results with 95% error bounds on each state

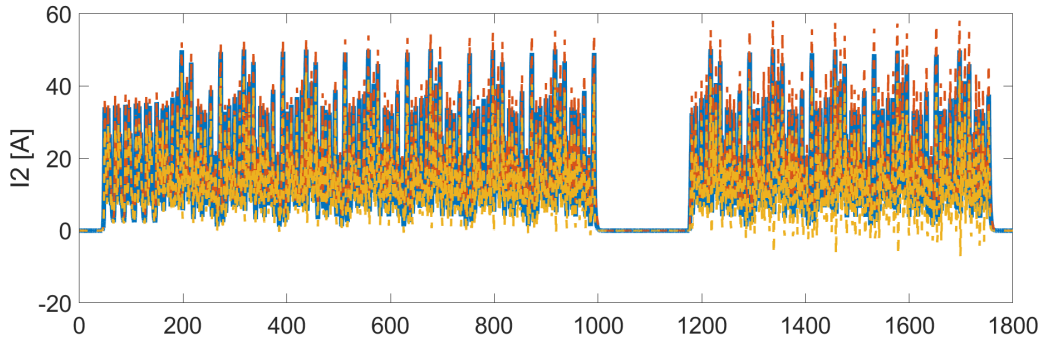
C.3 JAPF single run results



(a) SoC

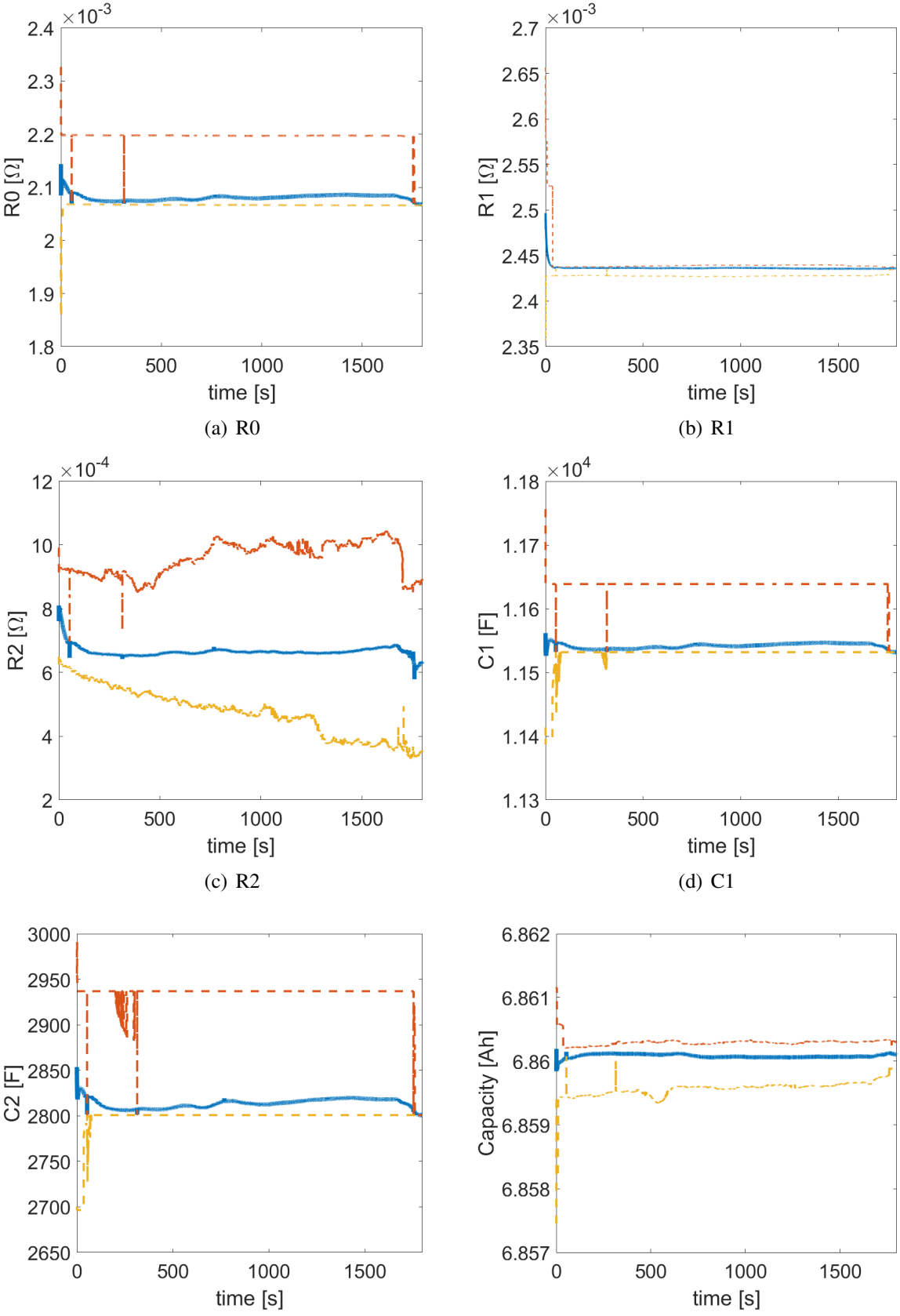


(b) I1



(c) I1

Figure C.5: JAPF single run state results with 95% error bounds on each state



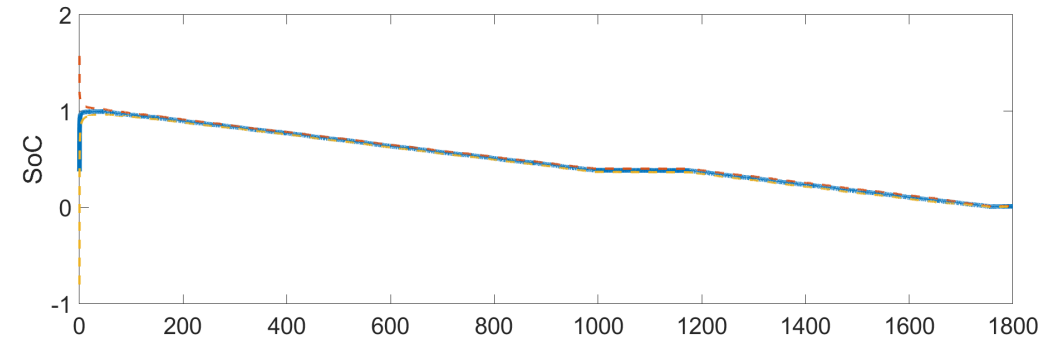
(a) R0 (b) R1

(c) R2 (d) C1

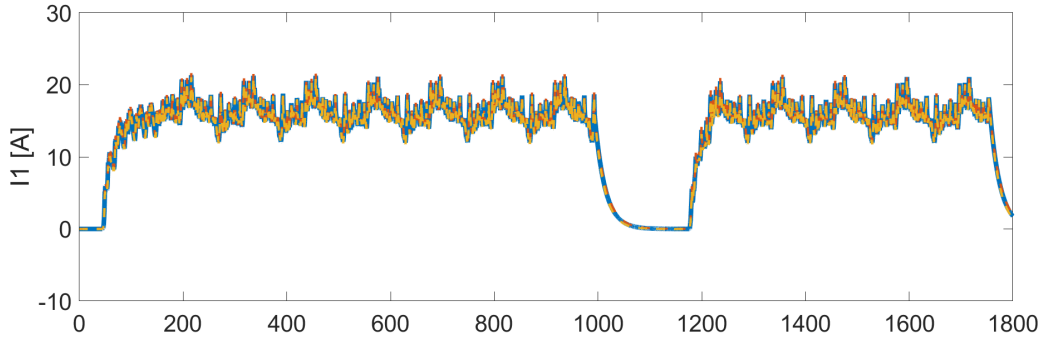
(e) C2 (f) Capacity

Figure C.6: JAPF single run parameter results with 95% error bounds on each state

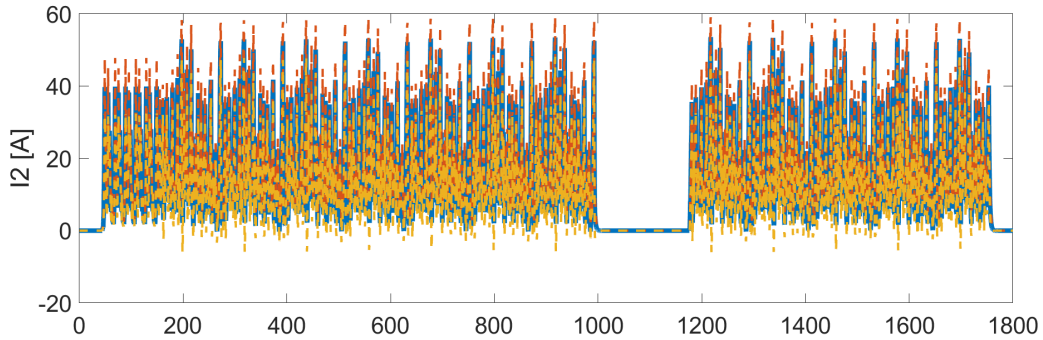
C.4 CDKF single run results



(a) SoC

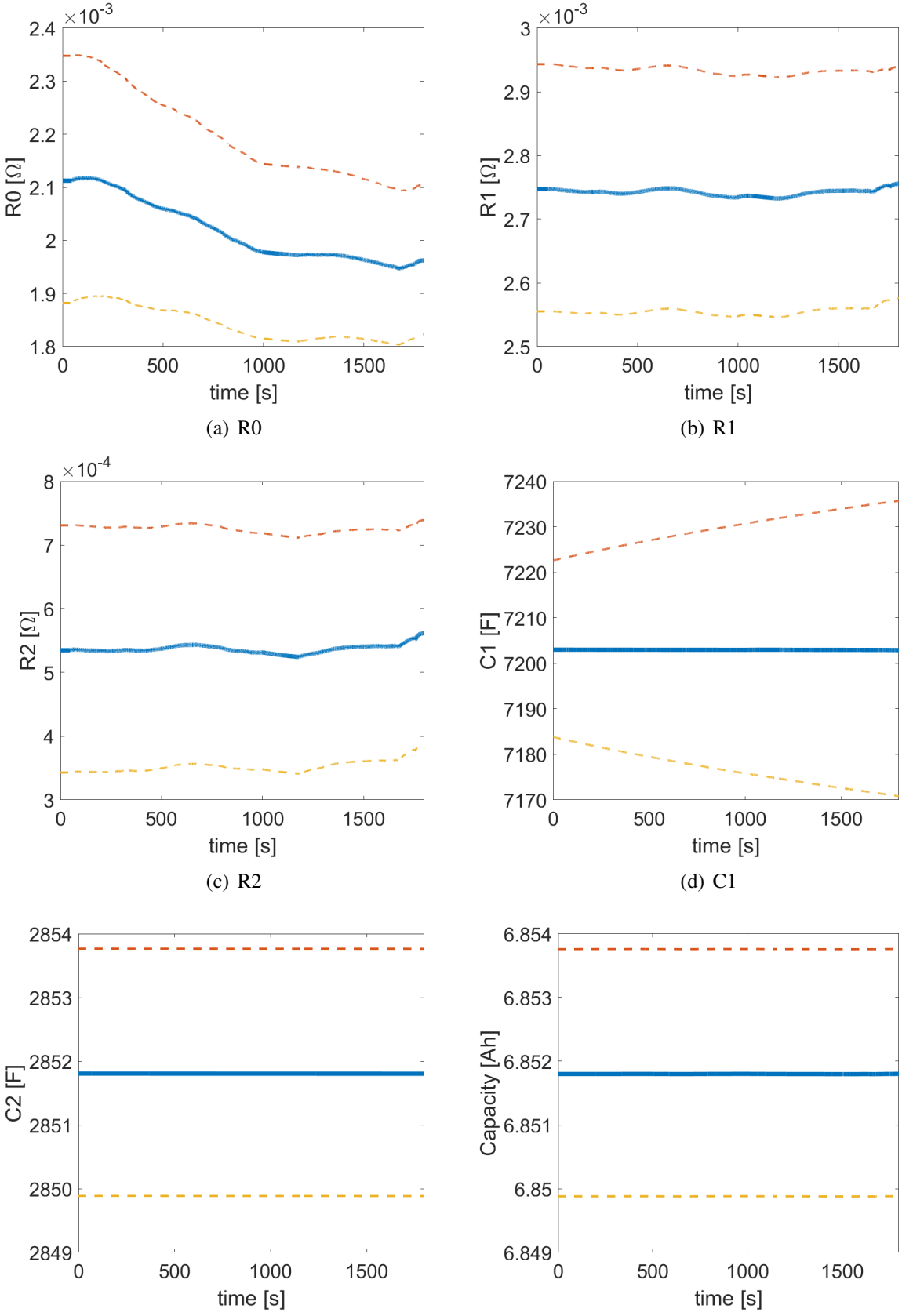


(b) I1



(c) I1

Figure C.7: CDKF single run state results with 95% error bounds on each state



(a) R0 (b) R1

(c) R2 (d) C1

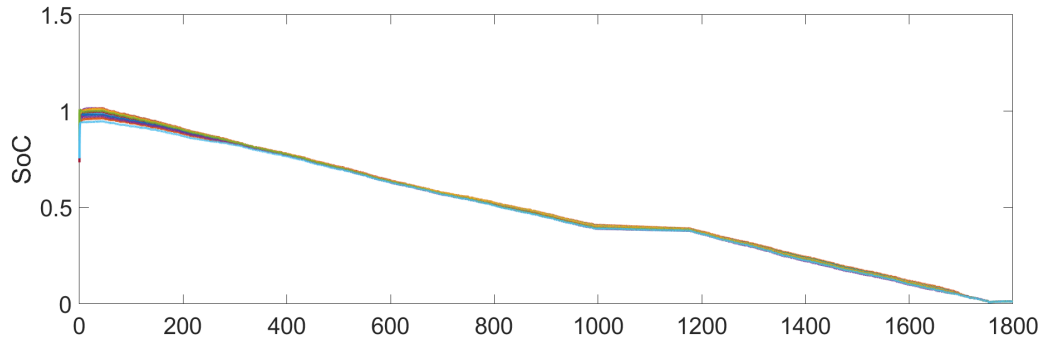
(e) C2 (f) Capacity

Figure C.8: CDKF single run parameter results with 95% error bounds on each state

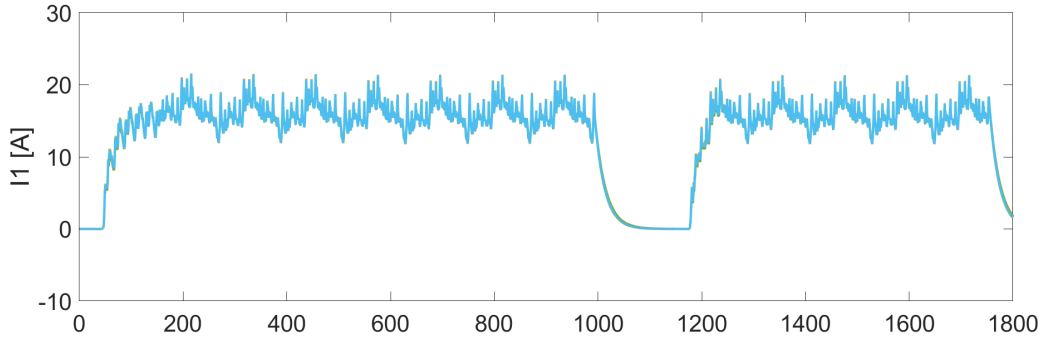
Appendix D

Estimator multiple run results, all states

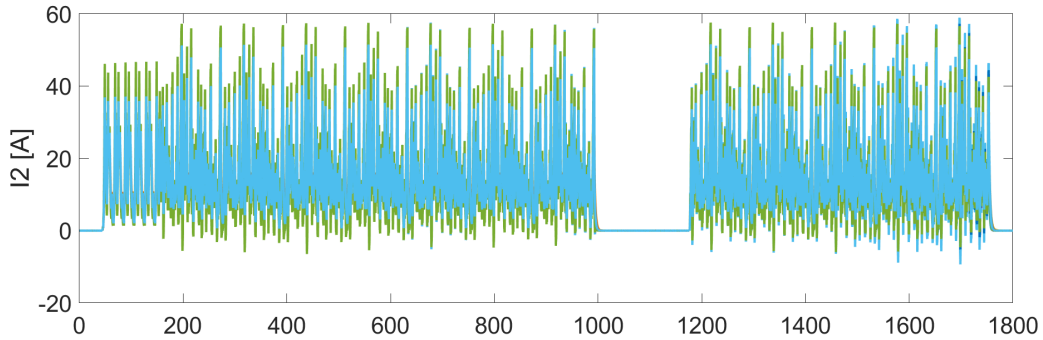
D.1 JBPF multiple run state and parameter outputs



(a) SoC

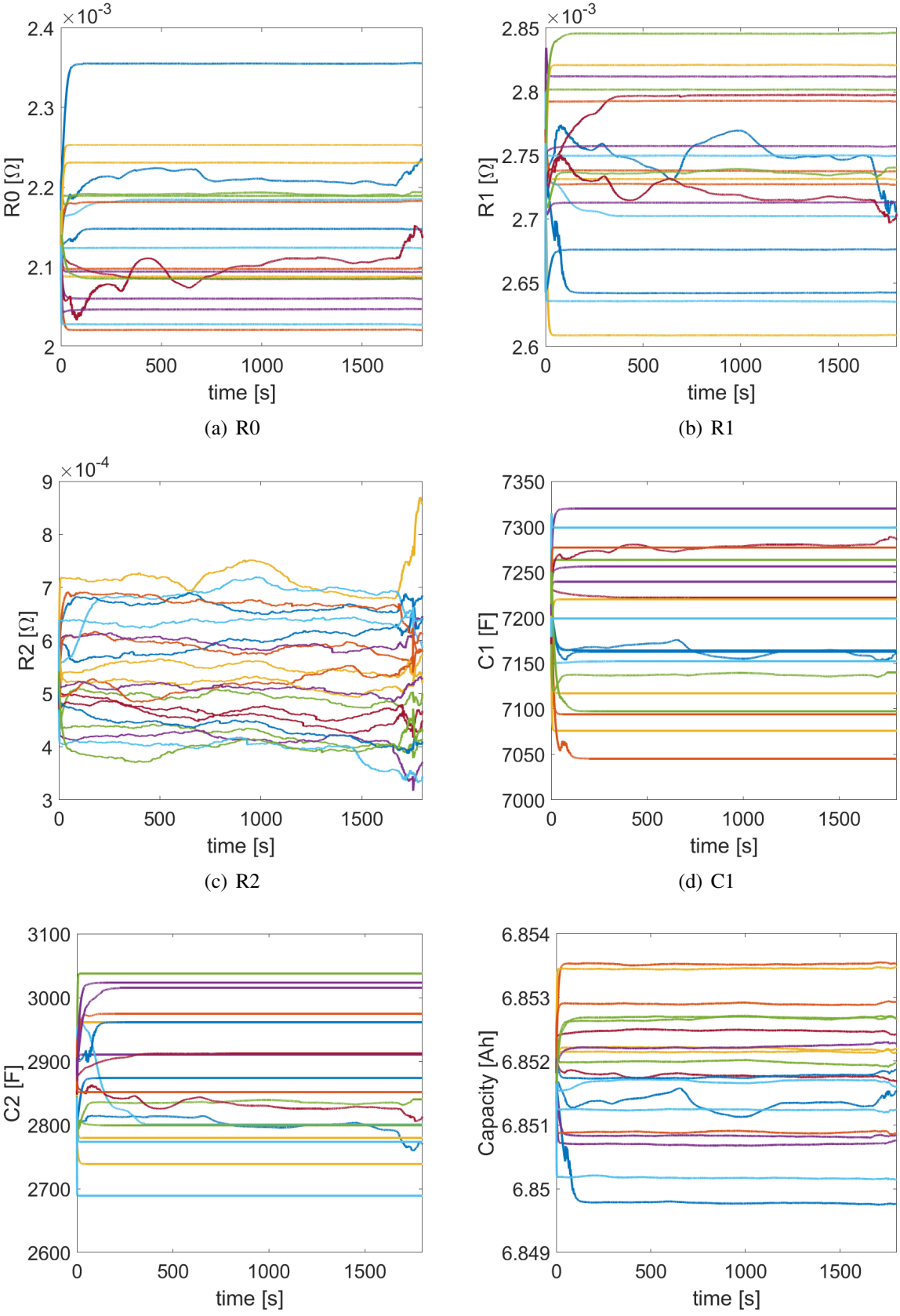


(b) I1



(c) I1

Figure D.1: JBPF 20 runs state results



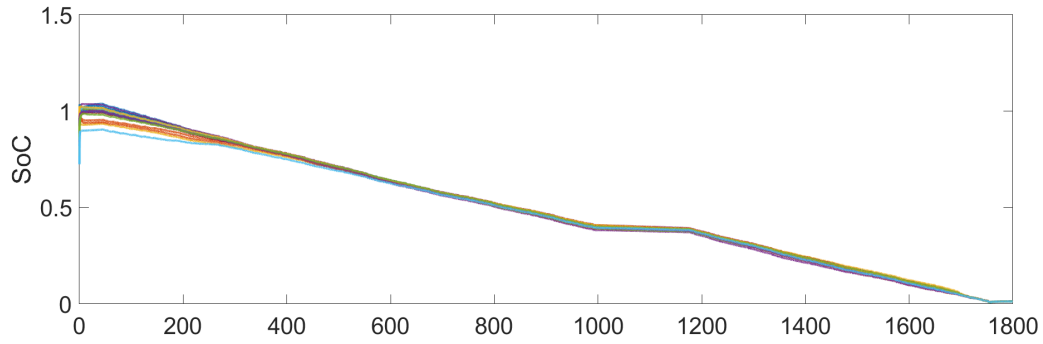
(a) R0 (b) R1

(c) R2 (d) C1

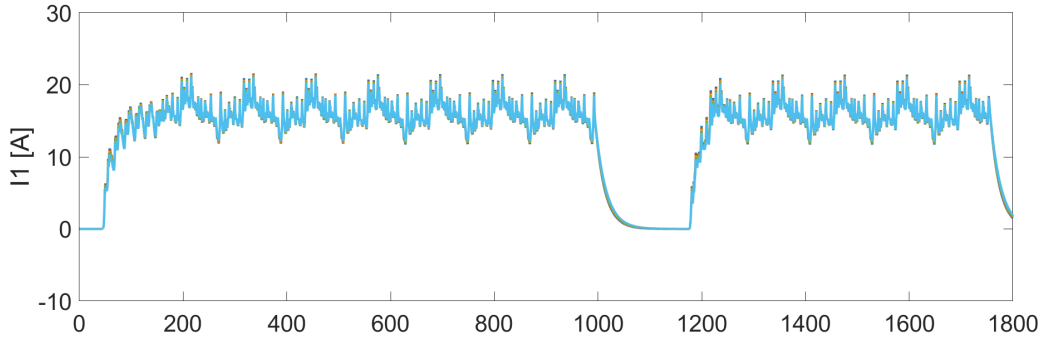
(e) C2 (f) Capacity

Figure D.2: JBPF 20 runs parameter results

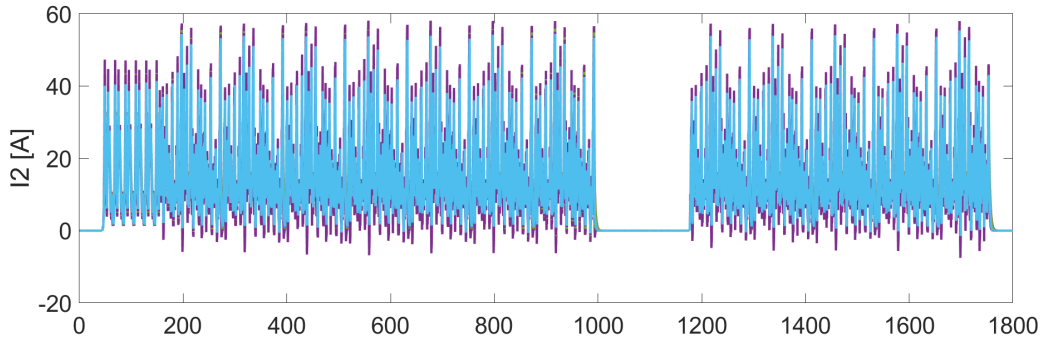
D.2 JIBPF multiple run state and parameter outputs



(a) SoC

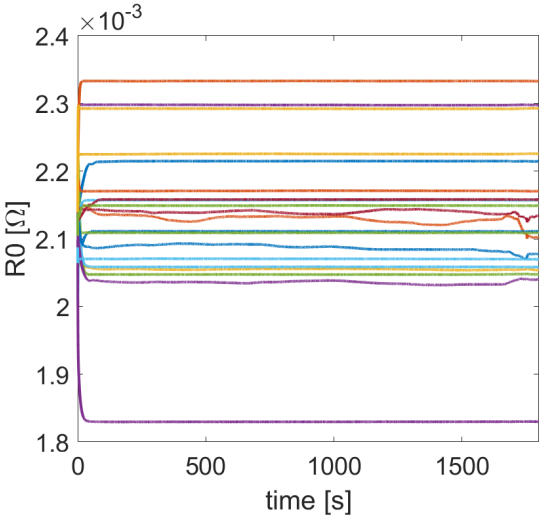


(b) I1

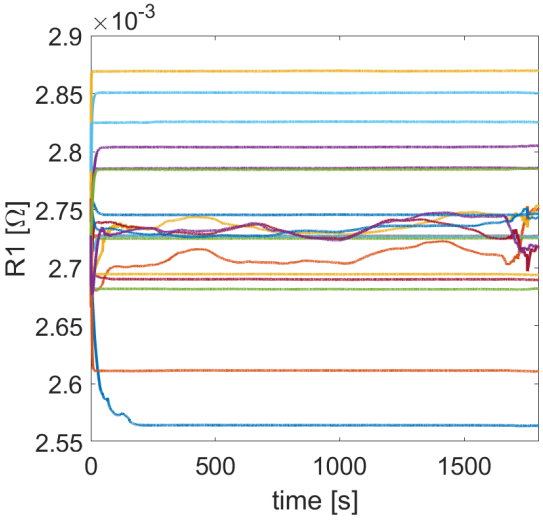


(c) I1

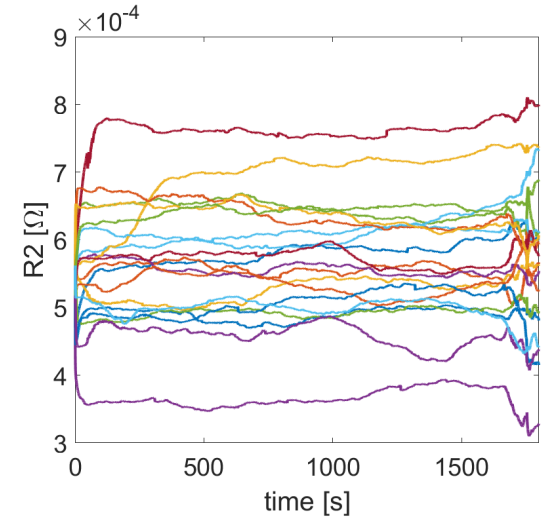
Figure D.3: JIBPF single run state results



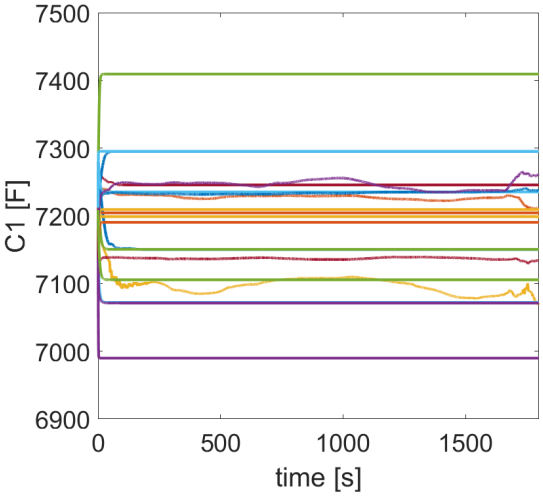
(a) R0



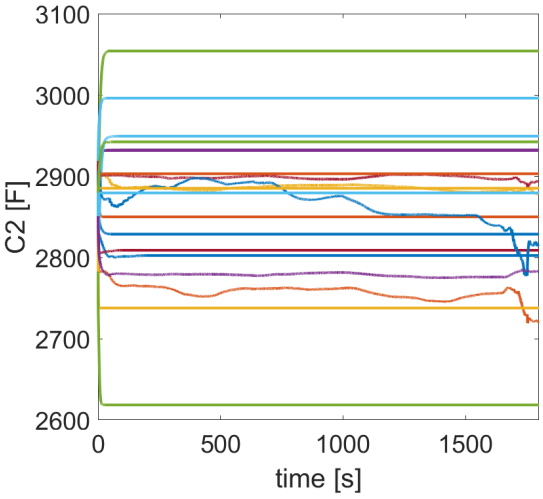
(b) R1



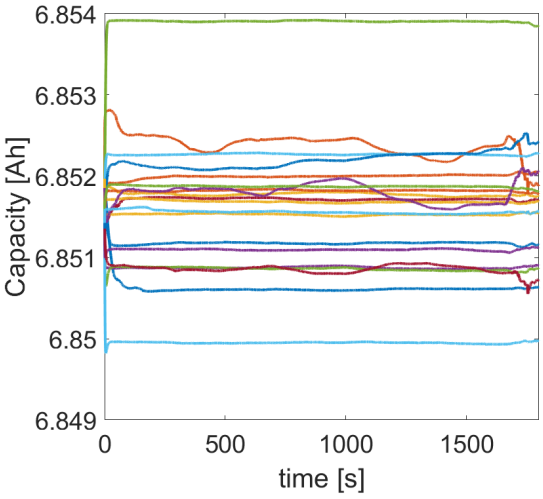
(c) R2



(d) C1



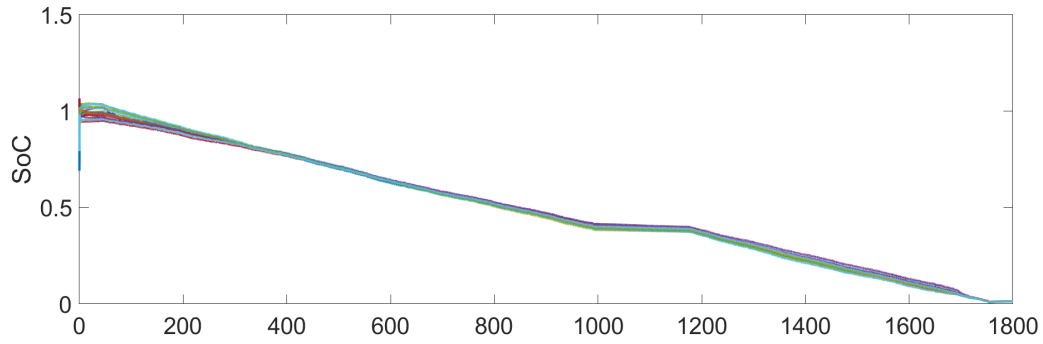
(e) C2



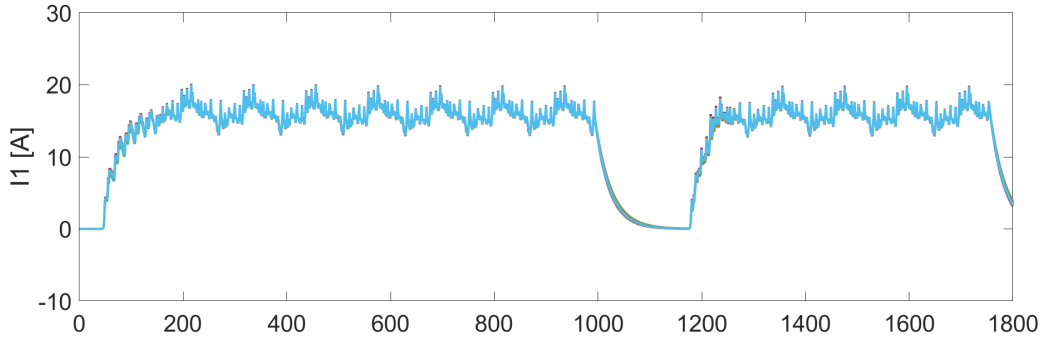
(f) Capacity

Figure D.4: JIBPF 20 runs parameter results

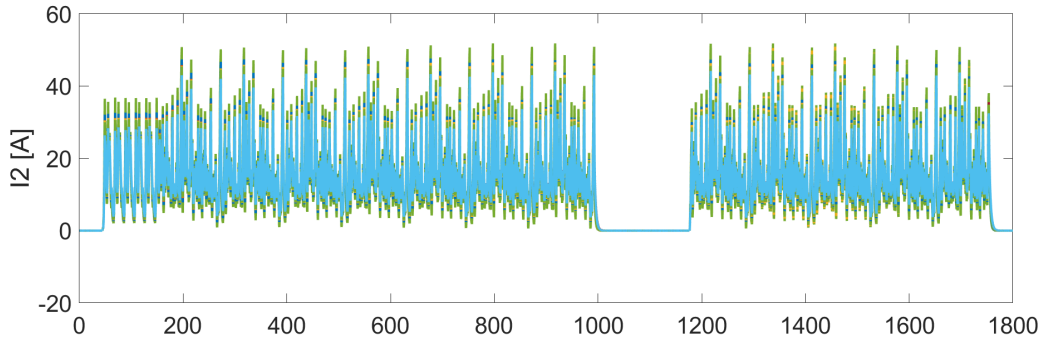
D.3 JAPF multiple run state and parameter outputs



(a) SoC

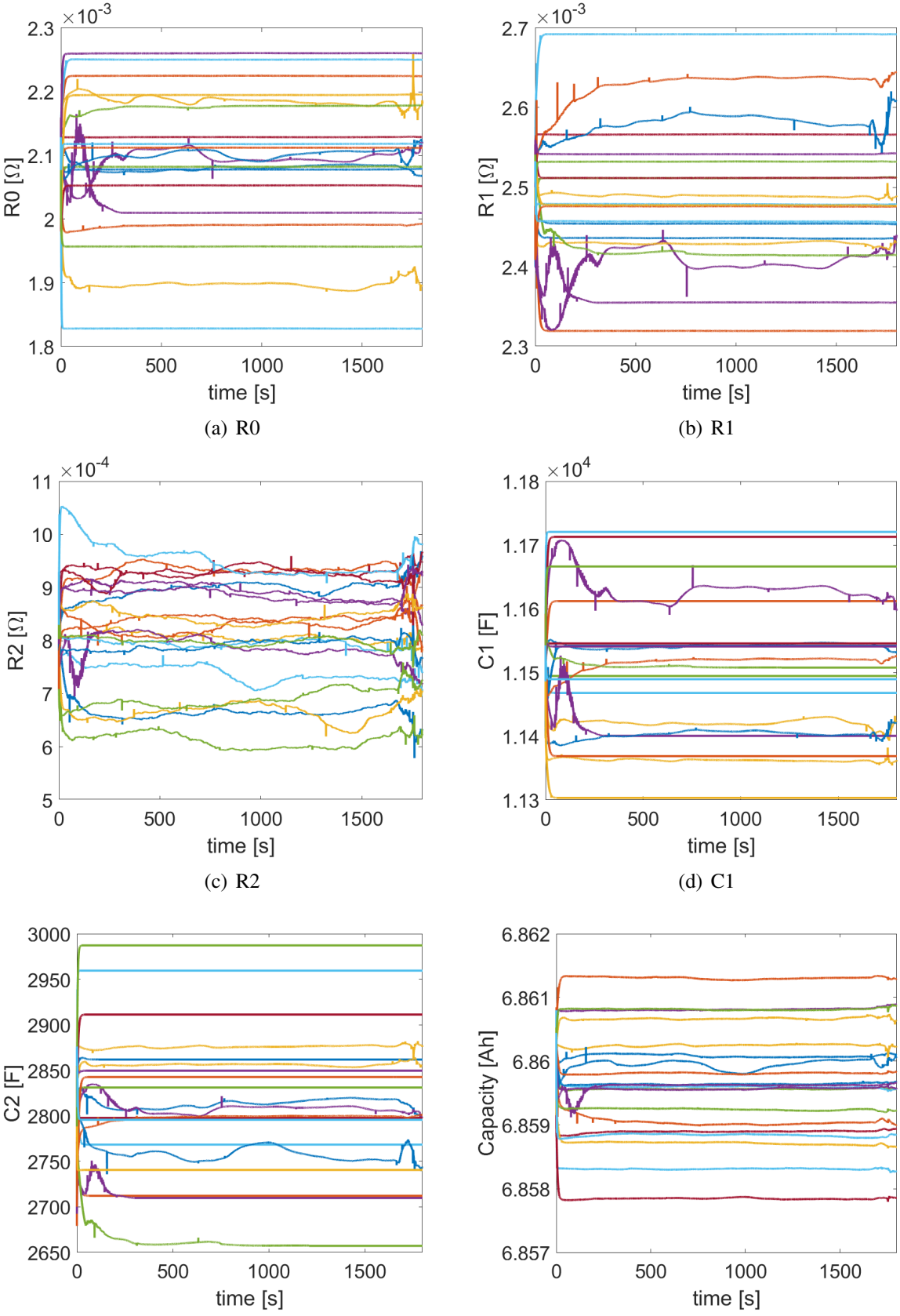


(b) I1



(c) I1

Figure D.5: JAPF 20 runs state results



(a) R0 (b) R1

(c) R2 (d) C1

(e) C2 (f) Capacity

Figure D.6: JAPF 20 runs parameter results