South Dakota State University

Open PRAIRIE: Open Public Research Access Institutional Repository and Information Exchange

Electronic Theses and Dissertations

2018

Nonlinear Stochastic Filtering for Online State of Charge and Remaining Useful Life Estimation of Lithium-ion Battery

Suresh Daravath

South Dakota State University

Recommended Citation

Daravath, Suresh, "Nonlinear Stochastic Filtering for Online State of Charge and Remaining Useful Life Estimation of Lithium-ion Battery" (2018). *Electronic Theses and Dissertations*. 2421.
https://openprairie.sdstate.edu/etd/2421

NONLINEAR STOCHASTIC FILTERING FOR ONLINE STATE OF CHARGE AND REMAINING USEFUL LIFE ESTIMATION OF LITHIUM-ION BATTERY

BY

SURESH DARAVATH

A thesis submitted in partial fulfillment of the requirements for the

Master of Science

Major in Mechanical Engineering

South Dakota State University

2018

NONLINEAR STOCHASTIC FILTERING FOR ONLINE STATE OF CHARGE AND REMAINING USEFUL LIFE ESTIMATION OF LITHIUM-ION BATTERY

This thesis is approved as a creditable and independent investigation by a candidate for the Master of Science in Mechanical Engineering degree and is acceptable for meeting the thesis requirements for this degree. Acceptance of this thesis does not imply that the conclusions reached by candidate are necessary the conclusions of the major department.

Huitian Lu, Ph.D.
Thesis Advisor Date

Kurt Bassett, Ph.D.
Head, Department of Mechanical Engineering Date

Dean, Graduate School Date

ACKNOWLEDGMENTS

I would like to express my gratitude to my thesis advisor, Professor Huitian Lu, for his supervision, advice, mentor, and guidance and professional assistance through my course of study. Dr. Lu has been always inspiration for me and contributed significantly through my thesis work and my academic career development from the early stage of this research to until the end.

I would like to thank all the faculty and staff of Mechanical Engineering Department for their consistent support.

Most of all, I would like to thank my family, my father Balu, my mother Pathali and my brother Vinod for their endless love, encouragement, support, and absolute confidence in me throughout my life. Without them I could not have finished this Thesis.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

NONLINEAR STOCHASTIC FILTERING FOR ONLINE STATE OF CHARGE AND REMAINING USEFUL LIFE ESTIMATION OF LITHIUM-ION BATTERY

SURESH DARAVATH

2018

Battery state monitoring is one of the key techniques in Battery Management System (BMS). Accurate estimation can help to improve the system performance and to prolong the battery lifetime. The main challenges for the state online estimation of Li-ion batteries are the flat characteristic of open circuit voltage (OCV) with the function of the state of charge. Hence, the focus of this thesis study is to estimation of the state of charge (SOC) of Li-ion with high accuracy, more robustness.

A 2nd order RC equivalent circuit model is adapted to battery model for simulation, mathematical model analysis, and dynamics characteristic of battery study. Model parameters are identified with MATLAB battery model simulation. Although with more lumped RC loaders, the model is more accurate, high computation with a higher nonlinear function of output will be. So, a discrete state space model for the battery is developed.

For a complex battery model with strong nonlinearity, Sequential Monte Carlo (SMC) method can be utilized to perform the on-line SOC estimation. An SMC integrates the Bayesian learning methods with sequential importance sampling. SMC approximate the posterior density function by a set of particles with associated weights, which is developed in MATLAB environment to estimate on-line SOC. A comparison is presented with Kalman Filtering and Extended Kalman Filtering to validated estimation results with

SMC. Finally, the comparison results provide that SMC method is more accurate and robust then KF and EKF.

Accurately prediction of Remaining Useful Life of Li-ion batteries is necessary to reliable system operation and monitoring the BMS. An empirical model for capacity degradation has been developed based on experimentally obtained capacity fade data. A nonlinear, non-Gaussian state space model is developed for empirical model. The obtained empirical model used in Sequential Monte Carlo (SMC) framework is to update the on-line state and model parameters to make a prediction of remaining useful life of a Li-ion battery at various lifecycle.

1. Introduction

1.1 Thesis Motivation

The motivation for the thesis is ambitious by the world's need to reduce the carbon emission and use to sustainable and highly reliable of renewable energy sources. Importance consideration for the spacecrafts, electric vehicles, and satellites are energy storage and Battery Management System are considered to this research.

A Comprehensive Literature of previous research work on the estimation of the State of Charge and Remaining Useful Life of Li-ion batteries has been revealed that a further investigation of on this topic is needed.

Although previous research work has proposed numerous estimation methods for SOC, most of them are a simple model with Simple algorithms are proposed with offline technique. However, the accuracy of these methods is relatively low. The Alternative, more precise adaptive algorithms are highly depending on the adopted dynamic battery model and computationally intensive onboard systems. Therefore, an accurate and fast Online SOC estimation method is needed.

Another important for Li-ion batteries is predicting the remaining useful life (RUL) have become increasingly important. Prognostics and Health Management (PHM) has as one of the keys enables to improve system safety, increase system operations reliability, system life cycle cost, and prevent catastrophic failure.

1.2 Thesis Objectives and Scope

The aim of the thesis is to model a second order equivalent circuit model and to understand the dynamic system of the model to online SOC estimation Li-ion battery.

The research objectives can be stated as follows:

The first specific objective is to model the Li-ion battery using Equivalent circuit method (ECM) with Thevien 2nd order RC- model is treated as a non-linear dynamic system, with discrete time state-space model. A nonlinear state-space model in presence of Non-Gaussian process and measurement noise. The Particle Filter general frameworks utilize the nonlinear system to assist the online estimation of the state of charge of a Li-ion battery.

The second specific objective is to implement an online particle filter based framework for Remaining Useful Life of Li-ion battery in nonlinear, non-Gaussian systems. Firstly, a new empirical model for capacity degradation was developed based on experimentally obtained capacity fade data. The obtained empirical model used in Sequential Monte Carlo framework to make a prediction of remaining useful life of a Li-ion battery at various lifecycle.

1.3 Organization of the Thesis

This thesis divided into 5 chapters.

Chapter 1: Thesis motivation, Research statement, and Objectives and Scope are discussed.

Chapter 2: A literature review of working principle of Li-ion battery, nonlinear filtering, battery modeling techniques, battery performance online assessment, and remaining useful life of Li-ion battery is described.

Chapter 3: A 2nd order RC Equivalent circuit model is developed, Identified the model parameters with pulse discharge current, implemented state space model, SOC estimation approach with particle filtering, particle filtering flow chart for SOC estimation, and simulation results and case study for the estimated SOC are described.

Chapter 4: Li-ion battery capacity degradation model, remaining useful life online assessment, implemented state space model for empirical data-driven, experimental results and discussion are presented.

Chapter 5: Particle filtering algorithm for estimation SOC and two case study are described two demonstrations of the proposed model. Conclusion based performance and complexity analysis are presented and recommendation for future research works are also provided.

2. Literature Review

2.1 Mathematical Definitions of SOC, SOH, and RUL

- State of charge (SOC)

The SOC of the battery is defined as the ratio of remaining charge capacity $Q(t)$ at any given time t to its total usable capacity Q_{total} when fully charged, and it is represented by

$$SOC(t) = \frac{Q(t)}{Q_{total}} \tag{2.1}$$

Accurate SOC estimation can maximize the performance of the battery and protect the battery to prevent overcharge and over discharge. In an electric vehicle, the parameter is the state of charge (SOC) as it shows the current battery capacity as a percentage of maximum capacity. As such it provides a measure of the amount of electric energy stored in a battery. It is analogous to fuel gauge on a conventional internal combustion engine vehicle [1]. The SOC is a dimensionless number between 0 and 1 representing a percentage. It is worth noting that a zero SOC does not mean that the battery full empty, only that the battery cannot be discharged anymore without causing permanent damage (irreversible chemical reaction) to it [2].

- State of Health (SOH)

The mathematical definition of SOH is not easy and differs for different applications one of the commonly adopted equations is defined as [3]:

$$SOH(t) = \frac{Q_{total}\,(t)}{Q_{new}} \tag{2.2}$$

Where Q_{new} is the capacity of new battery, and $Q_{total}(t)$ is the instantaneous total capacity at any given time t, it starts to decline as a function of time when the battery is aged or being in use. The estimation of Q_{total} over time is not simple, as there are many parameters involved in comprehensive algorithms. The State of Health (SOH) indicates a condition in the battery life between the beginning of life and End of Life in percentage. The beginning of the life of a battery is defined as the point in time when battery life beings. The end of life of a battery is reached when the battery cannot perform according to its predefined minimum requirements.

- Remaining Useful Life

Estimation for the system RUL, which is inherently entangled with the probability of failure time instants. This probability can be obtained from long-term predictions, when the empirical knowledge about critical conditions for the system is included in the form of thresholds for main fault indicators, also referred to as the hazard zones [4]. Defining the critical pdf with lower and upper bounds for the fault indicator (H_{lb} and H_{ub}, respectively). The hazard zone specifies the probability of failure for a fixed value of the fault indicator, and the weights $\{w_{t+k}^{(i)}\}_{i=1,\ldots,N}$ represents the predicted probability for the set of predicted paths, then it is possible to compute the probability of failure at any future time instant (namely the RUL) by applying the law of total probabilities, as shown in Equation (2.3). Once the RUL is computed, combining the weights of predicted trajectories with the hazard zone specifications, it is well known how to obtain prognosis confidence intervals, as well as the RUL expectation.

$$\hat{p}_{TTF}(ttf) = \sum_{i=1}^{N} Pr\,(failure|X = \hat{x}_{ttf}^{(i)}, H_{lb}, H_{ub}) \cdot w_{ttf}^{(i)} \qquad (2.3)$$

Equation (2.3) provides a solution for the RUL estimation problem that is suitable for on-line applications. As it depends on the predicted trajectory weights, though, it is subject to uncertainty and it may be sensitive to modelling errors. Moreover, uncertainty inherent to RUL expectations increases as the prediction horizon grows

2.2 General Operational Principle of Li-ion Battery

A rechargeable battery converts chemical energy into electrical energy and vice versa. The battery cell voltage is calculated by the energy of chemical reaction taking place inside the cell. The basic setup of a battery consists of three main parts: the positive electrode, the separator, and the negative electrode. The positive and negative electrode are referred to as the cathode and anode, as shown in Figure 2.1. The battery is connected to an external load using current collector plates. In case of Li-ion cells, a copper collector is used for the positive electrode [5].

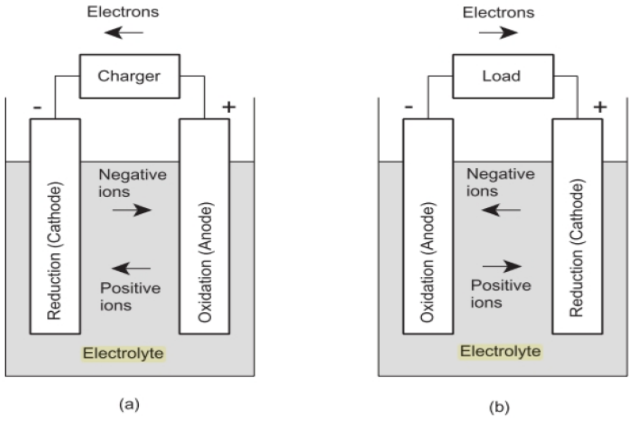


Figure 2. 1: Electrochemical functionality of a battery during charging (a), Discharging (b) [5].

The anode is the electrode capable of supplying electrons to the load. The anode composite material defines the name of the Li-ion battery and is usually made up of a mixture of carbon, while the electrolyte can be made of liquid, polymer, or solid materials. In case of solid or polymer material, the electrolyte will also act also as a separator.

The separator is a porous membrane allowing the transfer of Li-ions only, thus serving as a barrier between electrodes. It prevents the occurrent of short-circuiting and thermal runaway while at the same time offering negligible resistance. The cathode is the electrode usually made of metal oxides (ex. $LiCoO_2$ or $LiMn_2O_4$) as shown in Figure 2.2.

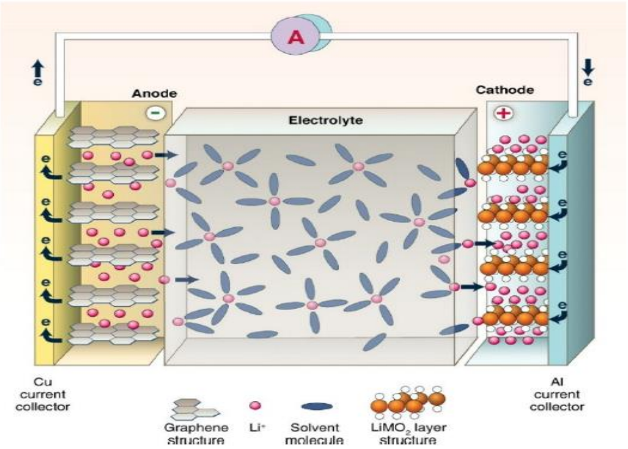


Figure 2. 2: Schematic representation of Li-ion battery discharging [6].

Under the presence of a load current, (Reduction-oxidation) redox reaction occurs. Oxidation reaction takes place at the anode where the trapped lithium particle starts to deintercalated or diffuse towards the electrolyte-solid interface splitting Li-ion into ions and electrons move through the solution due to the potential difference while the electrons moves through the current collector because the electrolyte solution acts as an electronic insulator [2]. Reduction reaction takes place at the cathode where the traveling Li-ion from the anode starts to intercalate and react with the electrode happens without a change in the electrode crystal structure “Intercalation” mechanism. The whole phenomenon of intercalation and deintercalation is reversible as Li-ions pass back and forth between the electrodes during charging and discharging [7]. In theory, this phenomenon could go on infinitely. Unfortunately, due to cell material degradation and other irreversible chemical reactions, the cell capacity and power degrade with the number of cycle and usage [8].

2.3 Nonlinear Stochastic Filtering

Let a system or signal process x_t is a Markov process and observation y_t is given by [9]

$$dy_t = h(\, x_t)dt + d\, W_t \tag{2.4}$$

Generally, $h(\cdot)$ is bounded measurement function. Assume that for each t, x_t and $(W_u - W_v), u, v \geq t$ are independent, which allows for the feedback case. The objective is to calculate in recursive form to estimates of x_t. To do this it is necessary to compute the condition of x_t given

$$y_y = \sigma\{y_s, s \leq t\} \tag{2.5}$$

Nonlinear filtering is a distinguished from the other approaches by its probabilistic nature. It is a field that combines aspects of stochastic analysis, information theory, and statistical inferences. Its generalization to nonlinear systems and nonlinear observations are collectively referred to as nonlinear filtering. To put it clear, nonlinear filtering is an extension of the Bayesian framework to the estimation, prediction, and interpolation of nonlinear stochastic dynamics. Its output is the distribution of the estimated process (the signal) given the data (the observations) available. This distribution is commonly known as the posterior distribution of the estimated process. It is a theoretically optimal algorithm in that it provides the best estimate of the quality of interest, more precisely, it minimizes the mean square error of the estimator.

Nonlinear filters can be classified according to the validity of the estimates within the state space or according to the approximated approach they use for the Bayesian recursive relation (BRR) solution. Firstly, we focus on the validity of the estimates. There are local filters and global filters.

The local filters usually come out of an approximation of the system to allow the Bayesian recursive relation solution for such approximated model. Estimates provided by the local filters are valid within a small neighborhood of a point in the state space. There are two basic approaches to the Bayesian recursive relation solution providing local estimates, the standard local filters, and the new generation derivative-free filters. The analytical approach is based on an approximation of the nonlinear functions in the system and measurement equations by the Taylor series expansion, 1st and 2nd order. They are represented by the extended Kalman filter [10] and its various modifications, e.g. the second order filter [11], the iteration filter etc. The numerical approach is based on Stirling's polynomial interpolation of the nonlinear functions or on the unscented transformation. The approach is represented by the unscented Kalman filter [12] or by the divided difference filters [13].

The global filters aim for the solution of the Bayesian recursive relation even for nonlinear or non-Gaussian systems by an analytical or a numerical approach. They usually approximate the conditional probability density function of the state and provide an estimate which is valid in almost whole state space. This global validity of Noticeably higher computational demands pays the estimate. The analytical approach is based on an approximation of the conditional pdf by e.g. a mixture of Gaussian distributions (the Gaussian sum filter [14-17]. The numerical approach solves the BRR numerically. It is represented by the point-mass methods [18-20] which approximate the state space by a set of isolated grid points and evaluate the conditional probability density function in the grid points only or by the sequential Monte Carlo methods [21-22] which approximate the conditional probability density function by a set of weighted samples.

2.3.1 Monte Carlo Approach

Consider computation of the following integral

$$I(f) = \int f(x)p(x)dx \tag{2.6}$$

where $x \in \chi$ denotes random variable described by the probability density function $p(x)$, $f(x)$ is an arbitrary vector function $f: \chi \rightarrow \mathbb{R}_n$ integral with respect to $p(x)$. The integral (2.6) represents computation of the mean of the function $f(x)$.

Note that the integral

$$I(f) = \int_{\dot{D}} f(x) \tag{2.7}$$

which is an essential part of many scientific problems, is a special case of the integral (2.6) considering the probability density function $p(x)$ to be uniform on D.

Also, note that the pdf $p(x)$ is sometimes known up to a normalization constant only. In such case the relation (2.6) is replaced by the following form

$$I(f) = \frac{\int f(x)p(x)dx}{\int p(x)dx} \tag{2.8}$$

If the pdf is known exactly, the integral in the denominator of (2.8) equals to one. Further, the (2.8) form of the integral $I(f)$ will be used. If it is possible to obtain a large number of samples drawn from the pdf, it is not difficult to approximate the usually intractable integral (2.6). Approximation of the integral in this case is easy to compute because it was given by evaluating the function $f(x)$ at the samples and averaging the results. The procedure represents the main idea of the MC approach. Unfortunately, it is not usually possible to draw the samples from the probability density function $p(x)$, and thus it is necessary to obtain the samples by another way.

Firstly, consider that it is possible to draw the samples from the probability density function $p(x)$ directly than the Monte Carlo approach can be specified as follows:

2.3.1 Perfect Sampling in Monte Carlo Approach

Simulate N independent identical distribution random samples, also named particles $\{X^{(i)}: i = 1,2,3, \ldots, N\}$ according to $p(x)$. Then the $p(x)$ can be approximated by the empirical $P_N(x)$

$$P_N(x) = \sum_{i=1}^{N} \delta(x - x^{(i)}) \qquad (2.9)$$

Where $\delta(x)$ represents Dirac function that has the fundamental property at $\int_{-\infty}^{\infty} f(x)\delta(x-a)dx = f(a)$ and $\delta(x-a) = 0$ for $x \neq a$ the estimation of $I(f)$ given as

$$I_n(f) = \frac{\int f(x)P_N(x)dx}{\int P_N(x)dx} = \frac{\frac{1}{N}\sum_{i=1}^{N} f(x^{(i)})}{\frac{1}{N}\sum_{i=1}^{N} 1} = \frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}) \qquad (2.10)$$

The estimation is unbiased and if the variance $var\{f(x)\}$ is finite, the variance of $I_n(f)$ is given as

$$var\{I_N(f)\} = \frac{var\{f(x)\}}{N} \qquad (2.11)$$

from the strong law of large number

$$I_N(f) \xrightarrow[N\to\infty]{} I(f) \qquad (2.12)$$

and $var\{f(x)\} < \infty$, then

$$\sqrt{N}[I_N(f) - I(f)] \Rightarrow N(f(x): 0, var\{f(x)\}) \qquad (2.13)$$

where $\xrightarrow[N\to\infty]{}$ means the almost sure convergence, the symbol ⇒ denotes convergence in distribution. The advantage of the Monte Carlo method is clear. From the set of random samples $\{X^{(i)}: i = 1,2,3, \dots, N\}$ one can estimate any quantity $I(f)$ and the rate of convergence of this estimate is independent of the dimension of the integral. Note that any deterministic numerical integration method has a rate of convergence that decreases as the dimension of integrand increases.

In the case when the perfect Monte Carlo sampling cannot be utilized because either it difficult to draw the samples from $p(x)$ or the pdf is known up to a normalization constant only, an alternative solution must be used. In the next subsection, such an alternative, the Importance Sampling technique, will be described.

2.3.2 Importance Sampling in Monte Carlo Approach

The importance sampling method [22] is based on so called importance sampling probability density function denoted as $\pi(x)$ which can be arbitrarily chosen provided that the support of $\pi(x)$ includes the support of $p(x)$. Now, the integral (2.6) can be computed as

$$I(f) = \frac{\int f(x)p(x)dx}{\int p(x)dx} = \frac{\int f(x)\frac{p(x)}{\pi(x)}\pi(x)dx}{\int \frac{p(x)}{\pi(x)}\pi(x)dx} = \frac{\int f(x)w(x)\pi(x)dx}{w(x)\pi(x)dx} \tag{2.14}$$

where $w(x) = \frac{p(x)}{\pi(x)}$ will be called importance weight. Now, assume that N samples $\{X^{(i)}: i = 1,2,3, \dots, N\}$ are drawn from the sampling distribution $\pi(x)$. Then the integral $I(f)$ can be estimated as

$$\hat{I}_N(f) \ = \frac{\frac{1}{N}\sum_{i=1}^{N} f(x^{(i)}w(x^{(i)}))}{\frac{1}{N}\sum_{j=1}^{N} w(x^{(j)})} = \sum_{i=1}^{N} f(x^{(i)}w(x^{(i)})) \tag{2.15}$$

Where the normalized weight $\overline{w}(x^{(i)})$are given as

$$\overline{w}\left(x^{(i)}\right) = \frac{w(x^{(i)})}{\sum_{j=1}^{N} w(x^{(j)})} \tag{2.16}$$

Considering N finite, the estimate $\hat{I}_N(f)$ is biased as is given by a ratio of two estimates.

but under the assumptions

$$E_{p(x)}[\overline{w}(x)] = c^{-1} \int \frac{p(x)}{\pi(x)} p(x)dx < \infty \tag{2.17}$$

$$E_{p(x)}[f(x)^2\overline{w}(x)] = c^{-1} \int \frac{f(x)^2p(x)}{\pi(x)} p(x)dx < \infty \tag{2.18}$$

Where $c = \int p(x)dx$, the strong law substantial number can be applied it hold as

$$I_N(f) \xrightarrow[N\to\infty]{} I(f) \tag{2.19}$$

2.3.2 Sequential Monte Carlo Methods

The main idea of the Monte Carlo (MC) method is to approximate an arbitrary pdf by a set of independent and identically distributed $(i.i.d.)$ random samples. The approximation is consequently used in the computation of an integral (e.g. mean value). The SMC method uses the MC method in a sequential framework, i.e. after obtaining new information, the approximation is repeated.

As it is not usually possible to draw samples from the pdf directly (e.g. the pdf is unknown or drawing samples from the pdf are too complex), is necessary to utilize an alternative. The most common alternatives are the importance sampling [23], the accept/reject technique [24], and Markov Chain Monte Carlo (MCMC) [25].

So far, we have considered a vector random variable x only. However, the filtering problem is based on vector random processes treatment, where a vector random process is defined by a set of vector random variables which will be indexed with time instants k. One of the processes describes the evolution of the state x_k and the information about the state is obtained from the second process describing the measurement z_k. Thus, the state segment x^k is given by the conditional $p(x^k|z^k)$.

Therefore, the integral in (2.3) will be considered in the following form

$$I(f_k) = E_{p\left(x^k \middle| z^k\right)}[f_k(x^k)] = \frac{\int f_k(x^k)\frac{p(x^k|z^k)}{\pi(x^k|z^k)}\pi(x^k|z^k)dx^k}{\int \frac{p(x^k|z^k)}{\pi(x^k|z^k)}\pi(x^k|z^k)dx^k} \qquad (2.20)$$

It represents the conditional mean of $f_k(x^k)$ with respect to the conditional probability density function $p(x^k|z^k)$. However, the conditional $p(x^k|z^k)$ is not known and its obtaining is the goal of the filtering problem. Fortunately, it can be expressed by the Bayesian relations and is known up to a normalization constant as

$$p(x^k|z^k) \propto p(x_0) \prod_{j=1}^{k} p(z_j|x_j)p(x_j|x_{j-1}) \qquad (2.21)$$

Calculation of the integral $I(f_k)$ involves generating samples $\{x^{k(i)}, i = 1,2,3,..,N\}$ from the sampling $\pi(x^k|z^k)$ and computing corresponding weights $\{\overline{w}\left(x^{k(i)}\right), i = 1,2,3,\dots,N\}$

$$\overline{w}(x^{k(i)}) \propto \frac{p(x^{k(i)}|z^k)}{\pi(x^{k(i)}|z^k)} \propto \frac{p(x_0^i \prod_{j=1}^{k} p(z_j|x_j^{(i)})p(x_j^i|x_{j-1}^{(i)}))}{\pi(x^{k(i)}|z^k)} \qquad (2.22)$$

The samples and the weights constitute an approximation of the conditional $p(x^k|z^k)$. This approximation can be used either to calculate the integral (2.17) or as a suitable representation of the conditional $p(x^k|z^k)$. Note that the filtering $p(x_k|z^k)$ is a marginal of $p(x^k|z^k)$ and is approximated by the samples $\{x^{k(i)}, i = 1,2,3,\dots,N\}$ and by the weights $\{\overline{w}\left(x^{k(i)}\right), i = 1,2,3,\dots,N\}$, where $\overline{w}\left(x_k^{(i)}\right) = \overline{w}\left(x^{k(i)}\right)$.

As the current time, instant increases, usage of the introduced importance sampling method leads to rising computational complexity because the importance weights m computed over the whole state trajectory at each time instant. A straightforward solution is to set up a sequential scheme for the importance sampling method. The main idea of the sequential scheme is to draw samples of the current state x_k only and to attach the samples to the past generated trajectories as $x^{(i)} = \{x^{k-1(i)}, x_k^{(i)}\}$. Precondition of this sequential manner is splitting of the posterior probability density function as follows

$$p(x^k|z^k) = p(x_k|x^{k-1}, z^k)p(x^{k-1}|z^{k-1}) \tag{2.23}$$

After applying the procedure k times, the following form of the conditional pdf can be found

$$p(x^k|z^k) = p(x_0)\prod_{j=1}^{k} p(x_j|x_{j-1,}z_j) \tag{2.24}$$

Note that $p\left(x_k\middle|x_{k-1,}z_k\right) \propto p(z_k|x_k)p(x_k|x_{k-1})$ because

$$p(x_k, z_k|x_{k-1}) = p\left(x_k\middle|x_{k-1,}z_k\right)p(z_k|x_{k-1}) = p(z_k|x_k)p(x_k|x_{k-1})$$

and consequently

$$p(x_k|x_{k-1}, z_k) = \frac{p(z_k|x_k)p(x_k|x_{k-1})}{p(z_k|x_{k-1})} \tag{2.25}$$

Then the following relation

$$p(x^k|z^k) \propto p(x_0)\prod_{j=1}^{k} p(z_j|x_j)p(x_j|x_{j-1}) \tag{2.26}$$

which is equal to (2.18), holds. The sampling $\pi(x^k|z^k)$ can be written similarly to the conditional in (2.25) as

$$\pi(x^k|z^k) = \pi(x_0)\prod_{j=1}^{k} \pi(x_j|x_{j-1}, z_j) \tag{2.27}$$

As we want to draw samples for the current state x_k only, the sampling pdf $\pi(x^k|z^k)$ will be replaced by a product of the sampling pdf's $\pi(x_k|x_{k-1}, z_k)$. Now, the relation for computing the weights (2.19) can be written as

$$\overline{w}(x^{k(i)}) \propto \frac{p(x^{k(i)}|z^k)}{\pi(x^{k(i)}|z^k)} \propto \frac{p(x_0^i \prod_{j=1}^{k} p(z_j|x_j^{(i)})p(x_j^i|x_{j-1}^{(i)}))}{\pi(x_0^{(i)})\prod_{j=1}^{k} \pi(x_j^{(i)}|x_{j-1}^{(i)}, z_j)} \tag{2.28}$$

The importance weight can be consequently evaluated recursively as

$$\overline{w}(x^{k(i)}) \propto \overline{w}(x^{k-1(i)}) \frac{p(z_k|x_k^{(i)})p(x_k^{(i)}|x_{k-1}^{(i)})}{\pi(x_k^{(i)}|x_{k-1}^{(i)}, z_k)} \qquad (2.29)$$

The sampling $\pi(x_k^{(i)}|x_{k-1}^{(i)}, z_k)$ together with the relation (2.29) for weights evaluation make up the sequential MC method which is an essence of the particle filters. The particle filters mostly differ in choice of the sampling pdf. That the simplest choice of the sampling $\pi(x_k|x_{k-1}^{(i)}, z_k)$ is the transition $p(x_k|x_{k-1})$.

The problem encountered when running the sequential MC method is that after a few times step the weight of a sample is close to one and the weight of the other samples are zero. That means that in this case, the sequential MC method is rather inefficient because only one sample cannot effectively represent the empirical distribution. This problem can be resolved by introducing a resampling step that transforms a set of weighted samples into a set of unweighted samples. Each sample $x_k^{(i)}$ in the original set is transformed into N_i samples of the same value in the resampled set where the quantity *Ni* is proportional to the weight $\overline{w}(x_k^{(i)})$.

2.4 Battery Modeling Techniques

Battery modeling is an important and challenging consideration in battery management systems. To fully understand the operation of a battery different approaches must be taken, as the problems cover many fields of science. The choice between these model is a trade-off between model complexity, starting from parameterization effort. To help solve these problems several model types are created, amongst which the most common are:

- Electrochemical models
- Physical models

- Equivalent circuit models

Each of the presented models gives different perspective into an explanation of the behavior of the battery from their respective field of science. Such separation is created so that knowledge from just one of the areas is sufficient to understand the processes taking place inside the battery.

2.4.1 Electrochemical Models

Electrochemical models are focus mostly on the chemical reactions taking place inside the battery captured using partial differential equations (PDE). This type of battery model finds its use in construction and design of internal electrochemical dynamics of the cell allowing trade-off analysis and high accuracy. A well-known early model with a high accuracy of 2% was originally developed by Doyle, Fuller, and Newman [26,27]. Since electrochemical models use particle differential equations with typically numerous unknown parameters, they are significantly more complicated and computationally expensive than others, making their use in a real-time application for battery management systems (BMS) almost impractical. Moreover, many parameters of the battery are very hard to describe using these models, such as internal resistance, which makes them not feasible to represent the dynamically changing key variables describing the battery behavior. For real-time applications, the electrochemical model reduction is mandatory. Several approaches for electrochemical model reduction have been proposed in the literature. It was observed that much of the computational complexity involved in electrochemical models comes from solving PDFs for Li-ion concentration in the solid particles of the electrodes. A common strategy is to make approximation and simplifications for this calculation [28]. However, the dynamic properties of the battery can be accomplished by analyzing the consistency of the substances taking part in the electrochemical reaction caused by connection of electrodes to an external circuit.

2.4.2 Physical Models

This model represents the operation of the battery through mathematical and physical equations. Two main methods used for the creation of those models can be distinguished, which are the finite number and the computational fluid dynamics technology. These methods allow deep understanding of the fluid and mass flow as well as heat transfer which are important for the operation of the battery. However, high computational power is required due to many complex calculations. Moreover, the process requires a lot of time which deems the model unusable for the purposes of the project presented in this paper

2.4.3 Equivalent Circuit Models

Equivalent circuit based-models uses simple elements such as resistors and capacitors to model the charging and discharging behavior of Li-ion batteries. This model is simple to implement, computationally efficient and simple for implementing parameter and model identification. Therefore, equivalent circuit model can easily have implemented in real-time onboard system microcontroller. However, the model has little or no physical meaning which makes them restrictive for the state of health estimation [29].

The Equivalent circuit model approach in battery management system has been extensively researched [30]. This choice is due to the early population of BMS for portable electronics, where the approximation of battery model with an equivalent circuit model is adequate. The equivalent circuit models represent the electrochemical parameters and the behavior of the system through the creation of simplified, equivalent circuit consisting of electrical elements. The simplicity of these models can vary greatly depending on the required level of precision. These models are easily adjustable to specific requirements while maintaining the lowest possible level of complexity. A disadvantage of equivalent circuit models is that these models are unable to measure underlying physical behavior like power fading, capacity fading, and aging effect. The main advantage is the ability to be implemented in a real-time application with an acceptable range of performance. Equivalent circuit model is chosen for the modeling of Li-ion battery, due to its ability to follow the dynamically changing variables with reasonable computational power requirement.

2.4.3.1 Simple Battery Model

A simple battery model consists of only linear, passive elements, created using open-circuit voltage ideal battery and constant internal resistance, the model in Figure 2.3 is a Simple battery model

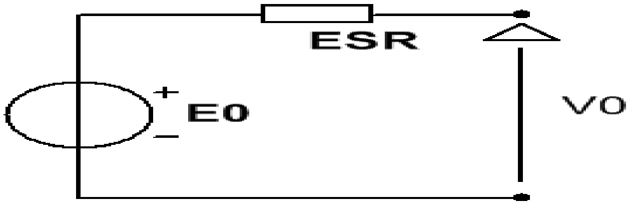


Figure 2. 3: Simple battery model [25].

Here, *ESR* is the internal series resistance, V_0 is the terminal voltage of the battery, and E_0 is open circuit voltage. The model is mostly used in systems where the battery doesn't have too high of an influence on the circuit. It is incapable of describing the battery behavior due to the lack of the relation of internal resistance in different states of charge (SOC).

2.4.3.2 Advanced Simple Battery Model

An advanced simple battery model is an improved version of the simple battery model through the addition of the dependence of internal resistance on the SOC. The configuration of this model is the same as the simple battery model, presented in Figure 2.3. The relation between the internal resistance and the SOC is represented by the equation:

$$ESR = \frac{R_0}{SOC} \tag{2.30}$$

Here, R_0 is the resistance of the fully charged battery, *SOC* is the state of charge of the battery, and *k* is capacity coefficient.

2.4.3.4 The 1st Order RC Model

The OCV-R-RC model is simplest equivalent circuit model and is selected to approximate the electrical performance of the battery as shown in Figure 2.4. It consists of three parts (1) open circuit voltage *OCV*, (2) Internal Resistances representing the ohmic resistances and (3) capacity.

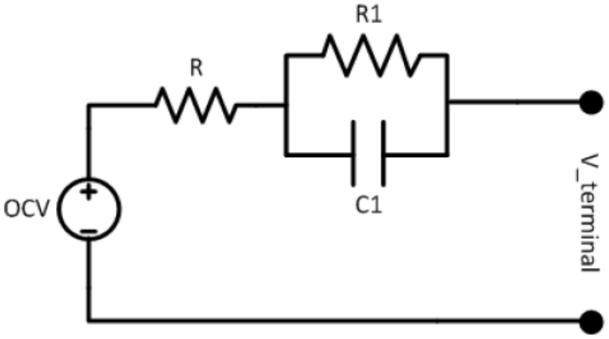


Figure 2. 4: Schematic diagram for R-RC model [25].

The model can capture the battery dynamic and can be easily implemented in the real-time application [30]. The R-RC model can be represented as follows

$$\begin{bmatrix} V_{1,k+1} \\ Z_{k+1} \end{bmatrix} = \begin{bmatrix} 1-\frac{\Delta t}{R_1 C_1} & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} V_{1,k} \\ Z_k \end{bmatrix} + \begin{bmatrix} \frac{\Delta t}{C_1} \\ -\frac{\eta_t \Delta t}{Q} \end{bmatrix} [i_k] \tag{2.31}$$

$$y_k = OCV(Z_k) - Ri_k - V_{1,k} \tag{2.32}$$

Where Z_k is the state of charge, OCV is the open circuit voltage, Q is the battery nominal voltage capacity, R is the battery ohmic resistance, R_1C_1 are RC pair and they represents the polarization time constant, $V_{1,k}$is a state represents the voltage across the capacitor. The state of systems is Z_k, $V_{1,k}$. the model has one output y_k, which is terminal voltage, the current i_k is input.

2.4.3.5 The 2nd Order RC Model

The OCV-R-RC-RC model is shown in Figure 2.5, [30]. The model is able to imitate fast and slow time constants for the voltage recovery of the battery.

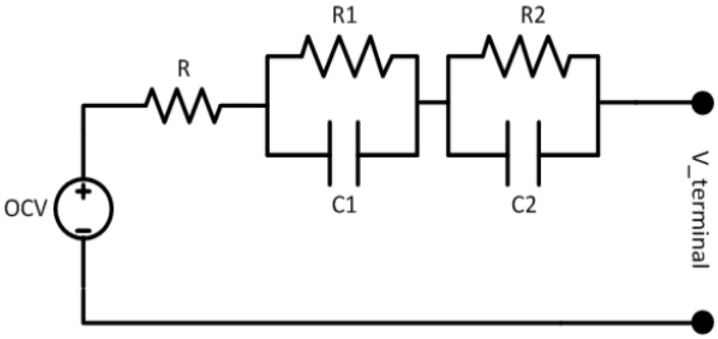


Figure 2. 5: Schematic diagram for R-RC-RC battery model [25]

This model can accurately capture the battery dynamics and it can be easily implemented in real-time applications. the model can be represented as follows

$$\begin{bmatrix} V_{1,k+1} \\ V_{2,k+1} \\ Z_{k+1} \end{bmatrix} = \begin{bmatrix} 1-\frac{\Delta t}{R_1 C_1} & 0 & 0 \\ 0 & 1-\frac{\Delta t}{R_2 C_2} & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V_{1,k} \\ V_{2,k} \\ Z_k \end{bmatrix} + \begin{bmatrix} \frac{\Delta t}{C_1} \\ \frac{\Delta t}{C_2} \\ -\frac{\eta_t \Delta t}{Q} \end{bmatrix} [i_k] \tag{2.33}$$

$$y_k = OCV(Z_k) - Ri_k - V_{1,k} - V_{2,k} \tag{2.34}$$

Where R_1C_1 is the fast polarization time constants, R_2C_2 represent the slow polarization time constant, $V_{1,k}$ is a state variable and represent the voltage across the first capacitor, $V_{2,k}$ is a state variable and represent the voltage across the second capacitor. The state variable of the systems is Z_k, $V_{1,k}$, $V_{2,k}$. The model has one output y_k, which is the terminal voltage, the current i_k is the input. The parameters vectors to be optimized for this model is θ=$[R_0, R_1, R_2, C_{1,}C_2]$. Model identification required significant computing time and power. However, these added parameters increase the model accuracy in real time application.

2.5 Battery Performance Online Assessment

Battery performance online assessment is a measure of battery life, which can quantify the in several ways. As the number of charge and discharge cycle until the end of useful life. The performance which depends on the state of charge, state of health, capacity, *C*-rate, and temperature. In which SOC and capacity are more important for battery performance assessment. Which describes the following two sections.

2.5.1 State of Charge (SOC) Estimates

The state of charge estimation is an important function of Battery Management System (BMS), and it is defined as the ratio of remaining charge capacity $Q(t)$ at any given time t to its total usable capacity Q_{total} when fully charged, and it is represented by

$$SOC(t) = \frac{Q(t)}{Q_{total}} \tag{2.35}$$

Accurate SOC estimation can maximize the performance of the battery and protect the battery to prevent overcharge and over discharge. However, it is difficult to measure SOC directly and it is typically estimated from direct measurement variables. Some approaches have been tested and initiate to provide a precise estimation of battery SOC, but these methods are prolonged, costly, and interrupt main battery performance. It is impossible to make intuitive SOC value measurements. Although SOC value exhibits a monotonous relationship with the battery open circuit voltage (OCV), the SOC value is very sensitive to the change of battery voltage, and even small voltage changes will translate to significant

changes in the SOC value. Overall, it is a significant challenge to obtain an accurate value of SOC. For this reason, estimation of the SOC value is a preferred approach.

In literature has been proposed many methods for SOC estimation, such as the Coulomb counting method (ampere-hour (Ah) integration method) [31-33], the open circuit voltage method [33,34], the BP (back-prorogation) neural network algorithm [35], neural network model methods (NN) [36], support vector regression methods(SVR) [37] and Fuzzy logic methods [38] but they are all computationally expensive and needs a lot of data for training.

Kalman filtering algorithm [39,40], Extended Kalman filtering, Unscented Kalman filter (UKF) [41-44], the strong tracking cubature Kalman filter (STCKF) [45], based on the Gaussian distribution noise, have been widely used for SOC estimation. Several other powerful yet challenging methods utilized to estimate SOC are open circuit voltage method (OCV) [46].

The Kalman filter (KF) is an autoregressive optimal data processing algorithm proposed by Kalman in 1960 [47]. Its core idea is to make the best estimate of the minimum variance in the system state. The KF algorithm overcomes the error accumulation effect of the coulomb counting method that occurs with increased time. The KF algorithm does not depend on an accurate initial SOC value but can improve the SOC value accuracy. However, the accuracy of this method depends on the establishment of a battery equivalent model, and some physical properties of the battery model are nonlinear. The EKF algorithm [48,49] and the UKF algorithm are improved KF algorithms. The EKF algorithm implements recursive filtering by linearizing nonlinear functions [50], and the UKF algorithm applies nonlinear system equations to the standard Kalman filter system by means of unscented transformation (UT). UT is a mathematical function used to estimate the result of applying a given nonlinear transformation to a probability distribution that is characterized by a finite set of statistics. Compared with the EKF algorithm, the UKF algorithm exhibit higher accuracy and has a wider application range, making it well-suited for solving nonlinear problems [51].

The Particle Filtering (PF) or Sequential Monte Carlo method is a random sampling-based filtering method used to solve non-linear non-Gaussian problems [52,53]. The rationale of this method is to use a series of weighted random sample sets (particles) in the state space to approximate the posterior probability density function of the system states. PF based estimator can be utilized for SOC estimation dealing with both the Gaussian and non- Gaussian distributed noise models. PF utilizes the particles (weighted random samples) to approximate the posterior distribution sampled by Monte-Carlo Methods. In these this thesis for SOC estimation SMC for PF algorithm is introduced.

2.5.2 Capacity Estimates

Online capacity estimation, which is a direct fading indicator for assessing the state of health (SOH) of a battery and remaining useful life of the battery. The method for the online capacity estimation of a single battery cell is presented. The stored charge Q (t) in a battery cell referred to the total capacity c_{max} is defined as the state of charge.

$$SOC= \frac{Q(t)}{C_{max}} \tag{2.36}$$

Therefore, *SOC* = 1 when the battery cell is fully charged and *SOC* = 0 when the battery cell is completely discharged. During charging/discharging [54], between times t_k and t_{k+1}, the stored charge is altered from Q_k to

$$Q_{k+1} = Q_k - \Delta Q_{k,k+1} = Q_k - \int_{t_k}^{t_{k+1}} I(t)dt \tag{2.37}$$

Where, I is the positive current during discharging and stored capacity changes to Q_k to Q_{k+1}, at same manner *SOC* changes to SOC_k to SOC_{k+1} and the total capacity of the battery calculated with

$$C_{max} = C_{k,k+1} = \frac{Q_k - Q_{k+1}}{SOC_k - SOC_{k+1}} \tag{2.38}$$

$$= \frac{\int_{t_k}^{t_{k+1}} I(t)dt}{SOC(t_k) - SOC(t_{k+1})} \tag{2.39}$$

Gradual deterioration of battery performance is caused by irreversible chemical reactions and leads to capacity fading and degradation, and which effects on Remaining useful life (RUL) of Li-ion battery. it noticed that RUL prediction is important to lifetime cycle, reliability, and prevent the catastrophic failure.

2.6 Remaining Useful Life Prediction

Li-ion batteries have been widely used in many fields, like electric vehicles, spacecraft, marine systems, aircrafts, satellites, consumer electronics, etc., due to their high-power density, low weight, keep a long lifetime, low self-discharge rate, no memory effect, and other advantages [55,56]. The demand for Li-ion batteries proves the necessity to evaluate their reliability. Failure of Li-ion batteries could lead to performance degradation, operational impairment, and even catastrophic failure [57-59]. To illustrate, in 2006, the National Aeronautics and Space Administration's Mars Global Surveyor stopped working due to the failure of batteries [60]. In 2013, all Boeing 787 Dreamliner's were indefinitely grounded due to battery failures that occurred on two planes [61]. Therefore, monitoring the degradation process, evaluating the state of health and predicting the remaining useful life (RUL) have become increasingly important for Li-ion batteries. Prognostics and Health Management (PHM) has as one of the keys enables to improve system safety, increase system operations reliability, and mission availability, predicting unnecessary maintenance actions, and reduce system life-cycle costs [62,63]. As a very important step of PHM, the RUL prediction based on the condition monitoring (CM) information plays a significant role in maintenance strategy selection, inspection optimization, and spare parts provision [64].

The probability of failure at any future time instant (namely the RUL) by applying the law of total probabilities, as shown in Equation (2.40). Once the RUL is computed, combining the weights of predicted trajectories with the hazard zone specifications [65], it is well known how to obtain prognosis confidence intervals, as well as the RUL expectation.

$$\hat{p}_{TTF}(ttf) = \sum_{i=1}^{N} Pr\,(failure|X = \hat{x}_{ttf}^{(i)}, H_{lb}, H_{ub}) \cdot w_{ttf}^{(i)} \tag{2.40}$$

If the RUL can be predicted accurately, predictive maintenance of the system or equipment can be implemented. Preventive maintenance before degradation is helpful to reduce failure rates and maintenance costs. Therefore, RUL prognostics has become a focus of researchers globally. RUL prognostics methodologies can be divided into the mechanism analysis method and the data-driven method [66]. The degradation of Li-ion batteries is a nonlinear and time-varying dynamic electrochemical process. Though mechanism analysis is clear in physical significance and concepts, it involves a lot of parameters and complex calculations for accurate modeling. In consequence, it is not suitable for real-time monitoring, which severely limits general applicability of the mechanism model. Instead, mechanism analysis is used more in theoretical research and battery designation than in practical engineering [67].

Data-driven techniques extract features from performance data such as current, voltage, capacity and impedance, and thus they are less complex than the Physics of failure-based approaches. The current research about the RUL prediction of Li-ion batteries focuses mainly on data-driven approaches.

The data-driven method of modeling batteries does not require an accurate mechanism of the system. Data-driven methods use the battery state of health data, which can be measured through advanced sensor technology. These methods extract effective feature information and construct the degradation model to predict RUL. These methods are able to describe degradation-inherent relationships and trends based on data [68]. Therefore, data-driven methods have become the focus of RUL prediction in the world [69]. Data-driven RUL prediction methods can be divided into three groups based on the artificial intelligence, filtering techniques, and stochastic process degradation, respectively.

2.7 Summary

Battery modeling is for behavior dynamic characteristic and the state of charge estimation are a very important aspect that can improve the performance of the system and improve the reliability of the system in Battery management systems. The literature review provides different battery modeling techniques were presented and ECM are selected between them because of model complexity, accuracy, and parameterization. For estimating SOC with ECM is very computational, time complexity but it's very accurate to estimate because of a SOC a nonlinear behavior of the battery with open circuit voltage, so nonlinear filtering for Sequential Monte Carlo method which based on Monte Carlo methods are discussed. Remaining useful life prediction based on data-driven techniques for an empirical model are discussed.

3. Online State of Charge Estimation

The Particle filter was developed based on state-space equations of the system and its accuracy is highly dependent on the accuracy of the system model. Thus, a battery model must be constructed to estimate the SOC using Particle filter-based framework. There are two basic requirements on a battery model for SOC estimation. Firstly, it can well simulate the dynamic behaviors of the battery. Secondly, the state-space equations can be easily derived according to the model. In Section 3.3 explains 2nd order ECM model that well meets the above two requirements are the equivalent circuit model (ECM) with lumped parameters.

3.1 Battery Modelling with Second Order Equivalent Circuit Model

Battery equivalent circuit model is commonly used for model-based state estimation design as shown in Figure 3.1. The dynamic cell behavior is described by an impedance model which includes an ohmic resistance R_0 with a two set of resistors R_1 and the capacitor C_1, resistor R_2 and the capacitor C_2 in parallel in the circuit. In this model, the circuit elements are both functions of *SOC* and consumed life. For example, if the consumed life is expressed in terms of the number of full charges or discharge cycle *N*, and expressed in remaining capacity *Qc* (see Figure. 3.1) a circuit element is the function of *SOC* and Q_c. The state space model is obtained based on circuit and takes the voltage across the 2nd RC ladder as $V_{oc}(SOC)$. Defined the state vector variable as *SOC*, U_1, and U_2.

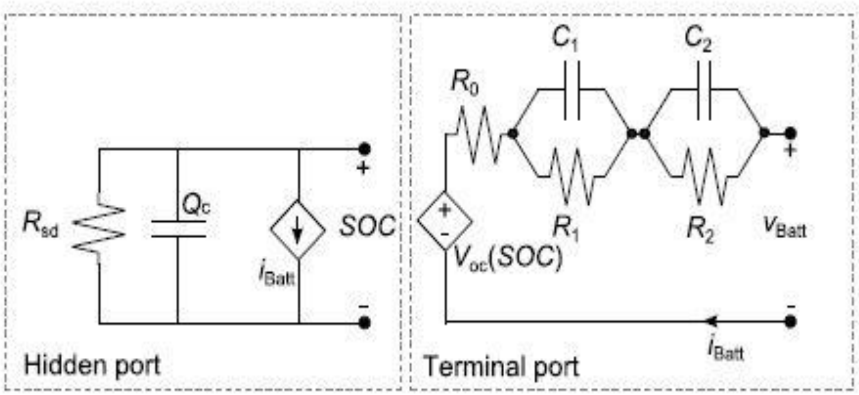


Figure 3. 1: Schematic representation of 2-RC ECM model [44].

The hidden port of the model consists of cell capacity Q_c, represented by a capacitor, self-discharge resistor R_{sd}, and controlled current source *SOC*. The loss of charge when the battery is in open circuit condition is typically negligible for most commercial Li-ion batteries, and R_{sd} can be safely ignored and assume that R_{sd}→∞. Because cell capacity fades as aging, it can be used as a direct measure of consumed life. As a result, assessment consists of two steps: first, the terminal voltage V_{Batt} and terminal current i_{Batt} are used to estimate the circuit components of the terminal port of Figure 3.1,V_{oc}, R_0, R_1, R_2, C_1, and C_2; and second, the parameters of the terminal port are used to estimate the component of the hidden port SOC, Q_c.

SOC is usually defined by equation

$$soc_k = soc_{k-1} - \left(\frac{\eta\Delta t}{Q_c}\right) i_{bat,k} \Rightarrow S\dot{O}C = \frac{i_{bat}}{Q_c} \tag{3.1}$$

Where soc_k and soc_{k-1} represents *SOC* value at times k and $k-1$, respectively; $i_{bat,k}$ represents the value of the current at time *k*; Q_c indicates the rated capacity of the battery. $S\dot{O}C$, is the derivative of *SOC*.

According to Kirchhoff's law, the following equations are obtained from the second-order RC equivalent circuit model:

$$\frac{U_1}{R_1}+C_1\frac{dU_1}{dt}=i_{bat}\ \Rightarrow \dot{U}_1=\frac{i_{bat}}{C_1}-\frac{U_1}{R_1C_1} \tag{3.2}$$

$$\frac{U_2}{R_2}+C_2\frac{dU_2}{dt}=i_{bat}\Rightarrow \dot{U}_2=\frac{i_{bat}}{C_2}-\frac{U_2}{R_2C_2} \tag{3.3}$$

$$V_{Batt}=V_{oc}(SOC)-U_1-U_2+R_0i_{bat} \tag{3.4}$$

U_1 And U_2 denote the terminal voltage of C_1 and C_1 respectively; $\dot{U}_1,\dot{U}_2$ are the derivatives of U_1 U_2 and respectively; V_{Batt} and i_{bat} represent the value of the terminal voltage and current, respectively. $V_{oc}(SOC)$ Indicates the open circuit voltage of the battery (under the same environmental conditions, the open-circuit voltage value and the SOC value are monotonous.)

3.2 Experimental and Identification of Model Parameters

A battery test experiment and the battery performance data identify the model parameters. The test profile is generally as follows: (1) the battery is firstly charged to the fully charged state with 0.1C standard charging method at the room temperature, and then it is left in open circuit condition for 5 hours; (2) the battery terminal voltage is measured and the measured voltage is regarded as the equilibrium potential because the battery is assumed to reach the steady state; (3) the battery is discharged with a constant current of 0.1C by 10% of SOC, and then left in open circuit condition for 2 hours; and (4) steps (2) and (3) are repeatedly performed until the battery reaches fully discharge state. In this model, a typical pulse discharging current point is employed, and the corresponding voltage profile is in Figure 3.2, where the battery discharge with 5A current. The second order system model of the battery is used for SOC and capacity estimation, some parameter in the model must be identified in advance, including open circuit voltage

$V_{oc}(SOC)$ and the value of {R_0, R_1, R_2, C_1, C_2} with the off-line method. Under the same temperature conditions.

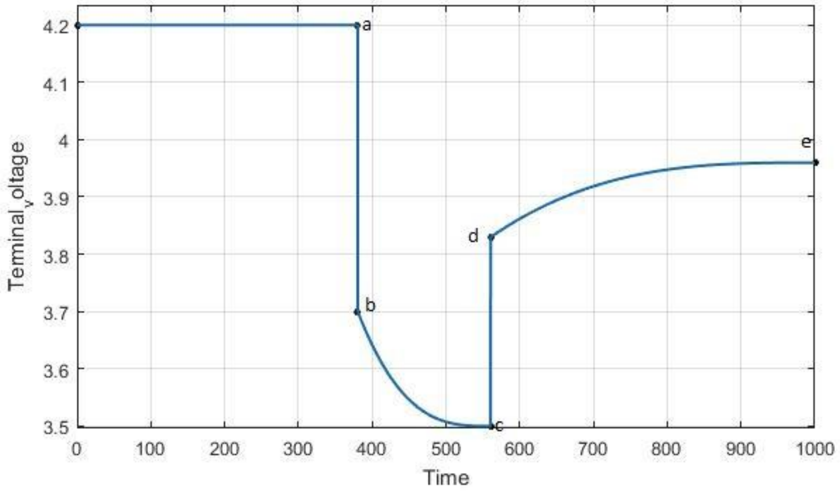


Figure 3. 2: Pulse discharge process with 5A current [46].

(1) Identification parameter R_0: For the 2nd order RC model shown in Figure 3.2, once the discharging current executed or stopped, the terminal voltage will drop immediately. Notice that the voltage U_1 and U_2 of the capacitors C_1 and C_2 would not be suddenly changed at the moment of starting discharging. Then, ohmic resistance R_0 could be found from numerous of the terminal voltage at the moment of starting discharging. Therefore, the ohmic resistance R_0 can be calculated by:

$$R_0 = \frac{|V_T(t_b) - V_T(t_a)| + |V_T(t_d) - V_T(t_c)|}{2|I_T|} \tag{3.5}$$

(2) Identify parameters R_0, R_1, R_2, C_1, and C_2: The identification of the parameters R_0, R_1, R_2, C_1, and C_2 is divided into two steps. The first step is to identify the time constant $\tau_1 \cong R_1C_1$ and$\tau_1 \cong R_1C_1$. Based on the identified time constant, the details identification of the R_0, R_1, R_2, C_1, and C_2 is introduced at another step. In addition, the response of the

1st order RC circuit with resistance *R*, capacitance *C*, and a constant current I is critical for identification, which is given by:

$$U(t) = U(t_0)e^{-\frac{t-t_0}{\tau}} + IR(1 - e^{-\frac{t-t_0}{\tau}}) \qquad (3.6)$$

Where $\tau = RC$ is the initial time constant.

Step 1. Identify the time constant τ_1 and τ_2 during the relaxation process *c*-*d*-*e*: note that the current equal zero during the relaxation process. Then according to Equation $U(t)$, the voltage U_1 and U_2 can be calculated by:

$$U_1(t) = U_1(t_c)e^{-\frac{t-t_C}{\tau_1}} \qquad (3.7)$$

$$U_2(t) = U_2(t_c)e^{-\frac{t-t_C}{\tau_1}} \qquad (3.8)$$

From the output equation i.e., terminal voltage is:

$$V_{Batt}(t) = V_{oc}(SOC) - U_1(t_c)e^{-\frac{t-t_C}{\tau_1}} - U_2(t_c)e^{-\frac{t-t_C}{\tau_1}} \qquad (3.9)$$

Which is rewrite as:

$$V_{Batt}(t) = \alpha_1 - \alpha_2 e^{-\frac{t-t_C}{\beta_1}} - \alpha_3 e^{-\frac{t-t_C}{\beta_2}} \qquad (3.10)$$

Here, $\alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2$ are the unknown coefficients. Obviously, we see $\alpha_1 = U_T(\infty)$ that are measured at the end of the relaxation process, i.e., the point *e* by using the MATLAB function “Custom Equation” in the curve fitting toolbox, the optimal coefficients $\alpha_2, \alpha_3, \beta_1, \beta_2$ can be obtained. Therefore, the time constants τ_1, τ_2 and the voltage $U_1(t_c)$, $U_2(t_c)$ are identified.

Step 2: Identify parameters R_1, R_2, C_1, and C_2 during the discharging process *a*-*b*-*c*: Note that the point a is the end of the previous relaxation process. Then, $U_1(t_a) = 0$ and$U_2(t_a) = 0$. It follows from Equation (3.6) that:

$$U_1(t) = I_T R_1(1 - e^{-\frac{t-t_a}{\tau_1}}) \tag{3.11}$$

$$U_2(t) = I_T R_2(1 - e^{-\frac{t-t_a}{\tau_2}}) \tag{3.12}$$

Hence, the resistance R_1, R_2 are determined by the following equations:

$$R_1 = \frac{U_1(t_c)}{I_T(1-e^{-\frac{t-t_a}{\tau_1}})} \tag{3.13}$$

$$R_2 = \frac{U_2(t_c)}{I_T(1-e^{-\frac{t-t_a}{\tau_2}})} \tag{3.14}$$

Where, $\tau_1, \tau_2, U_1(t_c)$ and $U_2(t_c)$ have been calculated at the above step 1. Since $\tau_1 = R_1C_1,\ \tau_2 = R_2C_2$, we can get, $C_1 = \frac{\tau_1}{R_1}$, $C_2 = \frac{\tau_2}{R_2}$. Therefore, the parameter identification is completed and shown Table 3.1.

Table 3.1: Identified parameters

R_0	R_1	R_2	C_1	C_2
0.0717 Ω	0.0310 Ω	0.0277 Ω	8437 μF	91,401 μF

(3) Identify the non-linear function $V_{oc}(SOC)$: The curve fitting method is used to identify the

Nonlinear function $V_{oc}(SOC)$. here, relatively accurate discharging experiments are carried out to reduce the fitting error of the curve fitting method, in which the discharging current pulse is set to be 5 A. The lasting time of the discharging current pulse is 380 s, which is utilized to achieve the 10% decline of SOC. Moreover, the battery is rest for about 30 minutes after a discharging period to ensure the end of the relaxation process. In order to accurately fit the measurement data, the sixth-order polynomial equation is employed as the nonlinear relationship between the *OCV* and *SOC*, which is given by:

$$V_{oc}(SOC) = 14.795(SOC)^6 - 36.612(SOC)^5 + 29.235(SOC)^4 - 6.281(SOC)^3 - 1.647(SOC)^2 + 1.286(SOC) + 3.404 \quad (3.15)$$

Finally, the validation of the above polynomial equation shown in Figure 3.3.

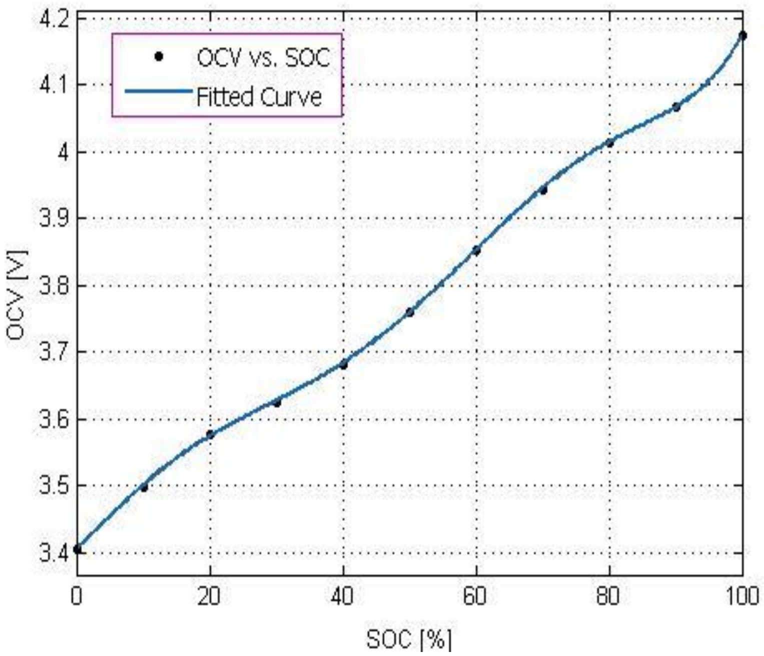


Figure 3. 3: Measured and fitted open circuit voltage (OCV) vs. state of charge.

3.3 State Space Model for 2nd Order ECM

State-space models are a very popular class of time series models presented in section 2.2. Formally, two stochastic processes define a state-space model $\{X_n\}_{n\geq0}$ and $\{Y_n\}_{n\geq0}$.

The differential equations of the second-order RC equivalent circuit model shown in Figure 3.1 can be derived as Equation 3.1,3.2, 3.3, 3.4

$$\begin{cases} S\dot{O}C = \frac{i(k)}{Q_c} \\ \dot{U}_1 = \frac{i_{bat}}{C_1} - \frac{U_1}{R_1 C_1} \\ \dot{U}_2 = \frac{i_{bat}}{C_2} - \frac{U_2}{R_2 C_2} \end{cases} \tag{3.16}$$

$$V_{Batt} = V_{oc}(SOC) - U_1 - U_2 + R_0 i_{bat} \tag{3.17}$$

U_1 And U_2 denote the terminal voltage of C_1 and C_1 respectively; $\dot{U}_1, \dot{U}_2$ are the derivatives of U_1 U_2 and respectively; V_{Batt} and i_{bat} represent the value of the terminal voltage and current at current time k, respectively. $V_{oc}(SOC)$ Indicates the open circuit voltage of the battery, which is varied with the change of SOC value.

So, the discrete state space equation of the battery 2nd order ECM discretized by the system is:

$$\begin{pmatrix} SOC_k \\ U_{1,k} \\ U_{2,k} \end{pmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 - \frac{\Delta t}{C_1 R_1} & 0 \\ 0 & 0 & 1 - \frac{\Delta t}{C_2 R_2} \end{bmatrix} \begin{pmatrix} SOC_{k-1} \\ U_{1,k-1} \\ U_{2,k-1} \end{pmatrix} + \begin{bmatrix} -\frac{\Delta t}{Q_c} \\ \frac{\Delta t}{C_1} \\ \frac{\Delta t}{C_2} \end{bmatrix} i_{bat,k} \tag{3.18}$$

Where k is the discrete-time index, Δt is the sample time and Q_c the discharge capacity of the battery. Thereby, the cell terminal voltage is observed as the value, obtain the observed equation:

$$V_{Batt,k} = V_{oc}(SOC_k) - R_0 i_{bat,k} - U_{1,k} - U_{2,k} \tag{3.19}$$

By selecting the $x = [SOC, U_{1,}U_2]^T$as the state vector, and considering the current i_{bat} and voltage V_{Batt} as the input and output variable respectively, the discrete time state equation of the 2nd RC ECM model can be; State variable $x_k = [SOC_k, U_{1,k}, U_{2,k}]^T$where, SOC_k is the state of charge, $U_{1,k}$ and $U_{2,k}$ are two terminal voltages of R_1C_1 and R_2C_2 circuit in state space at time k. Considering the current i_{bat} is defined as the system input and the terminal voltage $V_{Batt,k}$ is defined as the system output.

3.4 SOC Estimation Approach with Particle Filtering

For complex battery like ECM with strong non-linearity PF can be utilized to perform the SOC estimation. PF is class of Monte Carlo methods also known as Sequential Monte Carlo method that integrates the Bayesian filtering method with sequential importance sampling (SIS) and resampling.

For non-linear system

$$x_k = f(x_{k-1}, u_{k-1}) + \omega_{k-1} \quad \leftrightarrow \quad p(x_k|x_{k-1}) \tag{3.20}$$

$$y_k = g(x_k, u_k) + \vartheta_k \qquad \leftrightarrow \quad p(y_k|x_k) \tag{3.21}$$

Where x_k, stands for the immeasurable state vector at time step k, $u_k(= i(k))$ stand for the input vector, and $y_k(= V_{Batt}(k))$ is the measurement output. ω_{k-1} and ϑ_k are the processes and measurement non-Gaussian noise. $f(\cdot)$ and $g(\cdot)$ indicates the process and measurement function, respectively. Generally, $f(\cdot)$ is linear while $g(\cdot)$ is nonlinear function due to the nonlinear relationship between the OCV and SOC which is presented Equation (3.15).

From a Bayesian perspective, the estimating SOC state is to recursively calculate some degree of belief in the state x_k at time k, taking different values, given the data $y_{1:k}$ up to time k. Thus, it is required to construct the pdf of $p(x_k|y_{1:k})$. It is assumed that the initial pdf of $p(x_0|y_0) = p(x_0)$ of the state vector, which is also known as the prior distribution, is available (y_0 being the set of no measurements). Then, in principle, the $p(x_k|y_{1:k})$ may be obtained, recursively, into two stages: prediction and update.

Suppose that the required pdf $p(x_{k-1}|y_{1:k-1})$ at time $k-1$ is available. The prediction stage involves using the system model (3.20) to obtain the prior of the state at time k via the Chapman-Kolmogorov equation

$$p(x_k|y_{1:k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|y_{1:k-1})dx_{k-1} \tag{3.22}$$

Note that in (3.22), use has been made of the fact that $p(x_k|x_{k-1}, y_{1:k-1}) = p(x_k|x_{k-1})$ as (3.20) describes a Markov process of order one. The probabilistic model of the state evolution $p(x_k|x_{k-1})$ is defined by the system equation (3.20) and known input current of u_{k-1}. At time step k, a measurement y_k becomes available, and this may be used to update the prior distribution via Bayes' rule

$$p(x_k|y_{1:k}) = \frac{p(y_k|x_k)p(x_k|y_{1:k-1})}{p(y_k|y_{1:k-1})} \tag{3.23}$$

Where the normalizing constant

$$p(y_k|y_{1:k-1}) = \int p(y|x_k)p(x_k|y_{1:k-1})dx_k \tag{3.24}$$

Depends on likelihood function $p(y_k|x_k)$ defined by the measurement model (3.21). in the update state (3.23), the measurement y_k is used to modify the prior density to obtain the required posterior density of the current state.

The recurrence relations (3.22) and (3.23) form the basis for the optimal Bayesian solution. This recursive propagation of the posterior density is only a conceptual solution in that in general, it cannot be determined analytically. Solutions do exist in Particle filtering approximation the optimal Bayesian solution.

The Sequential Monte Carlo approach is known variously as bootstrap filtering, the condensation algorithm, particle filtering, and interactive particle approximation. The sequential importance sampling algorithm is an MC method that forms the basis of SMC. It is a technique for implementing a recursive Bayesian filter by MC simulations. The key idea is to represents the required posterior density function by a set of random samples (particles) with associated weight and to compute estimates based on these samples and weights. As the number of samples becomes very large, this MC characteristic becomes available an equivalent representation to the useful functional description of the posterior pdf, and the SIS filter approaches optimal Bayesian estimate.

In order to develop the detail algorithm, let $\left\{x_{0:k}^{(i)}, w_k^i\right\}_{i=1}^{N}$ denotes a random measure that characterizes the posterior $p(x_{0:k}|y_{1:k})$, where $\left\{x_{0:k}^i, i = 0,1,2,\dots,N_s\right\}$ is set of particles with associated weights $\left\{w_k^i, i = 0,1,2,\dots,N_s\right\}$ and $x_{0:k} = \left\{x_j, j = 0,1,2,\dots,k\right\}$ is the set of particles for all states up to time k. The weights are normalized such that $\sum_i^N w_k^i = 1$. Then the posterior density at k can be approximated as

$$p(x_{0:k}|y_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \,\delta(x_{0:k} - x_{0:k}^i) \qquad (3.25)$$

therefore, have a discrete weighted approximation to the true posterior, $p(x_{0:k}|y_{1:k})$. The weights are chosen using the principle of importance sampling. This principle relies on the follows. Suppose $p(x) \propto \pi(x)$ is a probability density from which it is difficult to draw sample but for which $\pi(x)$ can be evaluated. In addition, let $x^i \sim q(x), i =$

$1,2,3,\ldots,N_s$ be samples that are easily generated from the proposal $q(\cdot)$ Called an importance density. Then a weighted approximation to the density $p(.)$ Is given by

$$p(x) \approx \sum_{i=1}^{N_s} w^i\, \delta(x - x^i) \tag{3.26}$$

Where,

$$w^i \propto \frac{\pi(x^i)}{q(x^i)} \tag{3.27}$$

is the normalized weight of the i^{th} particle.

Therefore, if the particles $x_{0:k}^i$ were drawn from an importance density $q(x_{0:k}|y_{1:k})$, then the weights in (3.25) are defined by

$$w_k^i \propto \frac{p(x_{0:k}^i|y_{1:k})}{q(x_{0:k}^i|y_{1:k})} \tag{3.28}$$

Returning to the sequential case, at each iteration, one could have particles constituting an approximation to $p(x_{0:k-1}|y_{1:k-1})$ and want to approximate $p(x_{0:k}|y_{1:k})$ with the new set of particles.

If the importance density is chosen to factor such that

$$q(x_{0:k}|y_{1:k}) = q(x_k|x_{0:k-1}, z_{1:k})q(x_{0:k-1}|y_{1:k-1}) \tag{3.29}$$

Then one can obtain sample particles $x_{0:k}^i \sim q(x_{0:k}|y_{1:k})$ by augmenting each of the existing sample particles $x_{0:k-1}^i \sim q(x_{0:k-1}|y_{1:k-1})$ with the new states $x_k^i \sim q(x_k|x_{0:k-1}, z_{1:k-1})$. To derive the weight update equation, $p(x_{0:k}|y_{1:k})$ is first expressed in term of $p(x_{0:k-1}|y_{1:k-1})$, $p(y_k|x_k)$, and $p(x_k|y_{k-1})$. Note that (3.23) can be derived by integrating (3.30)

$$p(x_{0:k}|y_{1:k}) \propto p(y_k|x_k)p(x_k|x_{k-1})p(x_{0:k-1}|z_{1:k-1})$$

By substituting (3.29) and (3.30) into (3.28)

$$w_k^i \propto \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)p(x_{0:k}^i|y_{1:k-1})}{q(x_{0:k}^i|x_{0:k-1}^i, y_{1:k})q(x_{0:k-1}^i|y_{1:k-1})}$$

$$= w_{k-1}^i \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_{0:k}^i|x_{0:k-1}^i, y_{1:k})} \quad (3.30)$$

Furthermore, if $q(x_k|x_{0:k-1}, y_{1:k}) = q(x_k|x_{k-1}, y_k)$, then the importance density becomes only dependent on x_{k-1} and y_k. This is particularly useful in the common case when only a filtered estimate of $p(x_k|y_{1:k})$ is required at each time step. In such scenarios, only x_k^i need to be stored; therefore, one can discard the path $x_{0:k-1}^i$ and history of measurement $y_{0:k-1}^i$. The modified weight is then

$$w_k^i \propto w_{k-1}^i \frac{p(y_k|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, y_k)} \quad (3.31)$$

and the posterior filtered density $p(x_k|y_{1:k})$ can be approximated as

$$p(x_k|y_{1:k}) \approx \sum_{i=1}^{N_s} w_k^i \, \delta(x_k - x_k^i) \quad (3.32)$$

Where the weights are defined in (3.31). it can be shown that as $N_s \to \infty$, the approximation (3.32) approaches the true posterior density $p(x_k|y_{1:k})$.

However, a widespread problem with the SIS particle filtering is the degeneracy phenomenon, where after a few iterations, all but one particle will have negligible weight. This degeneracy implies that a large computation to the approximation to $p(x_k|y_{1:k})$ is almost zero. A suitable measure of degeneracy of the algorithm is the effective sample size N_{eff} introduced and defined as

$$N_{eff} = \frac{N_s}{1+var(w_k^{*i})} \quad (3.33)$$

Where $w_k^{*i} = p\left(x_k^i \middle| y_{1:k}\right)/q(x_k^i|x_{k-1}^i, y_k)$ is referred to as the true weight. This cannot be evaluated exactly, but an estimate $\widehat{N}_{eff}$ of N_{eff} can be obtained by

$$\widehat{N}_{eff} = \frac{1}{\sum_{i=1}^{N_s}(w_k^i)^2} \quad (3.34)$$

Where, w_k^i is the normalized weight obtained using (3.30). notice that $N_{eff} \leq N_s$, and small N_{eff} indicates severe degeneracy. Clearly, the degeneracy problem is an undesirable effect in particle filters. The basic force approaches to reducing its effect is to use large N_s. This is often impractical; therefore, it relies on resampling method.

The basic idea of resampling use is to eliminate particle that has small weight and to concentrate on the particle with large weighs. The resampling set involves generating a new set particle $\left\{x_k^{*(i)}\right\}_{i=1}^{N_s}$ by resampling (with replacement) N_s times from approximate discrete representation of $p(x_k|y_{1:k})$ given by (3.32). the resulting sample is in fact as independent identical distribution. sample from the discrete distribution (3.32); therefore, the weights are now reset to $w_k^i = \frac{1}{N_s}$. It is possible to implement this resampling procedure operations by sample particles of N_s order uniforms using an algorithm based on order statistics.

3.5 Flow Chart of Particle Filtering to Estimate SOC

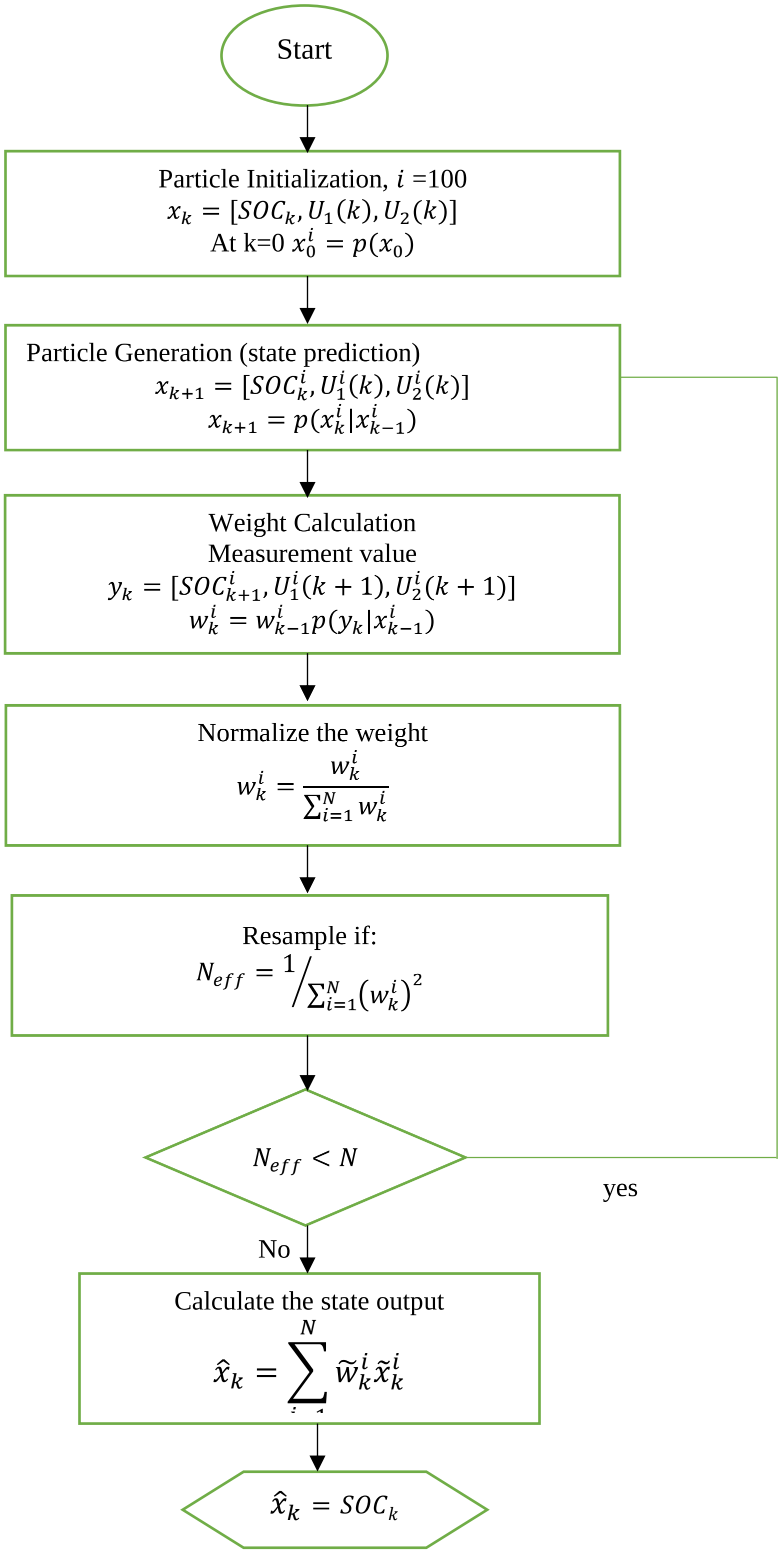

Start
Particle Initialization, i =100
$x_k = [SOC_k, U_1(k), U_2(k)]$
At k=0 $x_0^i = p(x_0)$
Particle Generation (state prediction)
$x_{k+1} = [SOC_k^i, U_1^i(k), U_2^i(k)]$
$x_{k+1} = p(x_k^i | x_{k-1}^i)$
Weight Calculation
Measurement value
$y_k = [SOC_{k+1}^i, U_1^i(k+1), U_2^i(k+1)]$
$w_k^i = w_{k-1}^i p(y_k | x_{k-1}^i)$
Normalize the weight
$w_k^i = \frac{w_k^i}{\sum_{i=1}^N w_k^i}$
Resample if:
$N_{eff} = {}^{1}\!/\!{}_{\sum_{i=1}^N (w_k^i)^2}$
$N_{eff} < N$
yes
No
Calculate the state output
$\hat{x}_k = \sum^N \widetilde{w}_k^i \tilde{x}_k^i$
$\hat{x}_k = SOC_k$

Figure 3. 4: Flow chart of particle filtering for SOC estimation.

3.6 Particle Filtering Algorithm

The particles drawn from the distribution $p(x_k|y_{0:k})$ would represent samples ideally. However, it is often impossible to sample directly from the true posterior density. It is necessary for researchers to find an alternative easy-to-sample proposal distribution $q(x_k|y_{0:k})$. Sequential importance sampling (SIS) and resampling form the bases of the standard PF algorithm. The standard PF is described as follows.

(1) Initialization

Set $k = 0$ and draw particles $x_0^i \sim p(x_0), i = 1,2, \dots, N$.

(2) Importance sampling and weights calculation

For $i = 1,2, \dots, N$, drawn $x_k^i \sim q(x_k^i|x_{0:k-1}^i, y_{0:k})$. In standard SMC, define $q\left(x_k^i\middle|x_{0:k-1}^i, y_{0:k}\right) = p(x_k^i|x_{k-1}^i)$. Assign the particle weight according to

$$w_k^i = w_{k-1}^i p\left(y_k\middle|x_{k-1}^i\right) = w_{k-1}^i \frac{p(y|x_k^i)p(x_k^i|x_{k-1}^i)}{q(x_k^i|x_{k-1}^i, y_k)} \tag{3.35}$$

Normalize weights

$$w_k^i = {w_k^i}\Big/{\sum_{i=1}^N w_k^i} \tag{3.36}$$

(3) Re-sampling

If the effective sample size N$_{\text{eff}}$ is below the given threshold N^{th}, do the re-sampling procedure. Generally, let $N^{\text{th}} = \frac{2}{3}N$

$$N_{eff} \approx 1\Big/{\sum_{i=1}^N \left(w_k^i\right)^2} \tag{3.37}$$

Draw N particles from the current particle set $\widetilde{x_k^i}$ and replace the current set with the new one

$$\widetilde{w}_k^i = 1/N \tag{3.38}$$

(3.22) State prediction Calculate the state by the equation

$$\hat{x}_k = \sum_{i=1}^{N} \widetilde{w}_k^i \tilde{x}_k^i \tag{3.39}$$

If $k \leq T$ (T is the number of the measurements), let $k = k + 1$, turn to step 2; else, end the prediction.

3.7 SOC Estimation Approach with Extended Kalman Filtering

The Extended Kalman Filter is a method for system state estimation in real time. In this application, to estimate the SOC during discharge, the EKF can be constructed in the following steps.

State space representation (3.18) and (3.19) can be shortly expressed in (3.20) and (3.21) for non-linear systems:

$$x_k = f(x_{k-1}, u_{k-1}) + \omega_{k-1} \tag{3.22}$$

$$y_k = g(x_k, u_k) + \vartheta_k \tag{3.23}$$

$$\omega_{k-1} \sim (0, Q_k = diag(Q_{SOC_k}, Q_{U_{1,k}}, Q_{U_{2,k}})), \quad \vartheta_k = (0, R_k)$$

Where $x_k,$ stands for the immeasurable state vector at time step k, u_k(=$i(k)$) stand for the input vector, and $y_k(= V_{Batt}(k))$ is the measurement output. ω_{k-1} and ϑ_k are the processes and measurement Gaussian noise with covariance matrix $Q_k = diag\left(Q_{SOC_k}, Q_{U_{1,k}}, Q_{U_{2,k}}\right)$ and R_k. $f(\cdot)$ and $g(\cdot)$ indicates the process and measurement function, respectively. Generally, $f(\cdot)$ is linear while $g(\cdot)$ is nonlinear function due to the nonlinear relationship between the OCV and SOC which is presented equation (3.15). as for the Q_k, the $Q_{SOC_k}, Q_{U_{1,k}}, Q_{U_{2,k}}$ are the covariance of the SOC and dynamic voltages U_1 and U_2 respectively.

Compute the particle derivative matrices:

$$A_{k-1} = \left.\frac{\partial f}{\partial x}\right|_{\hat{x}_{k-1,}u_{k-1}} , \ H_k = \left.\frac{\partial h}{\partial x}\right|_{\hat{x}_{k|k-1}} \quad (3.24)$$

The initialization can be given by:

For *k*=0, set

$$\hat{x}_0^+ = E[x_0] = x_0 \quad (3.25)$$

$$\hat{P}_0^+ = E[(x_0 - \hat{x}_0^+)(x_0 - \hat{x}_0^+)^T] = P_{x_0} \quad (3.26)$$

Where, P_0^+ is the prediction error covariance matrix.

For *k* = 1, 2,… the following steps are performed

Step 1: Perform the time update of the state estimate and estimation error covariance:

$$\text{State estimation time update: } \hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}) \quad (3.27)$$

$$\text{Error covariance matrix time update: } P_k^- = A_k P_{k-1} A_k^T + w_k Q_{k-1} w_k^T \quad (3.28)$$

Step 2: Compute the Kalman gain matrix:

$$\text{Kalman gain matrix: } Gain_k = P_k^- H_k^T (H_k P_k^- H_k^T + v_k R_k v_k^T)^{-1} \quad (3.29)$$

Step 3: Measurement update:

$$\text{State estimation measurement update: } \hat{x}_k = \hat{x}_k^- + Gain_k(y_k - g(\hat{x}_k, 0)) \ (3.30)$$

$$\text{Error covariance measurement update: } P_k = (1 - Gain_k H_k) P_k^- \quad (3.31)$$

The process of the EKF algorithm is summarized in Figure 3.5. the iterative process between time update and measurement update starts after the initialization. In this way, SOC can be obtained based on the information of battery terminal voltage, V_{Batt} , and input vectors, i_{bat}.

3.8 Extended Kalman Filter Flow Chart

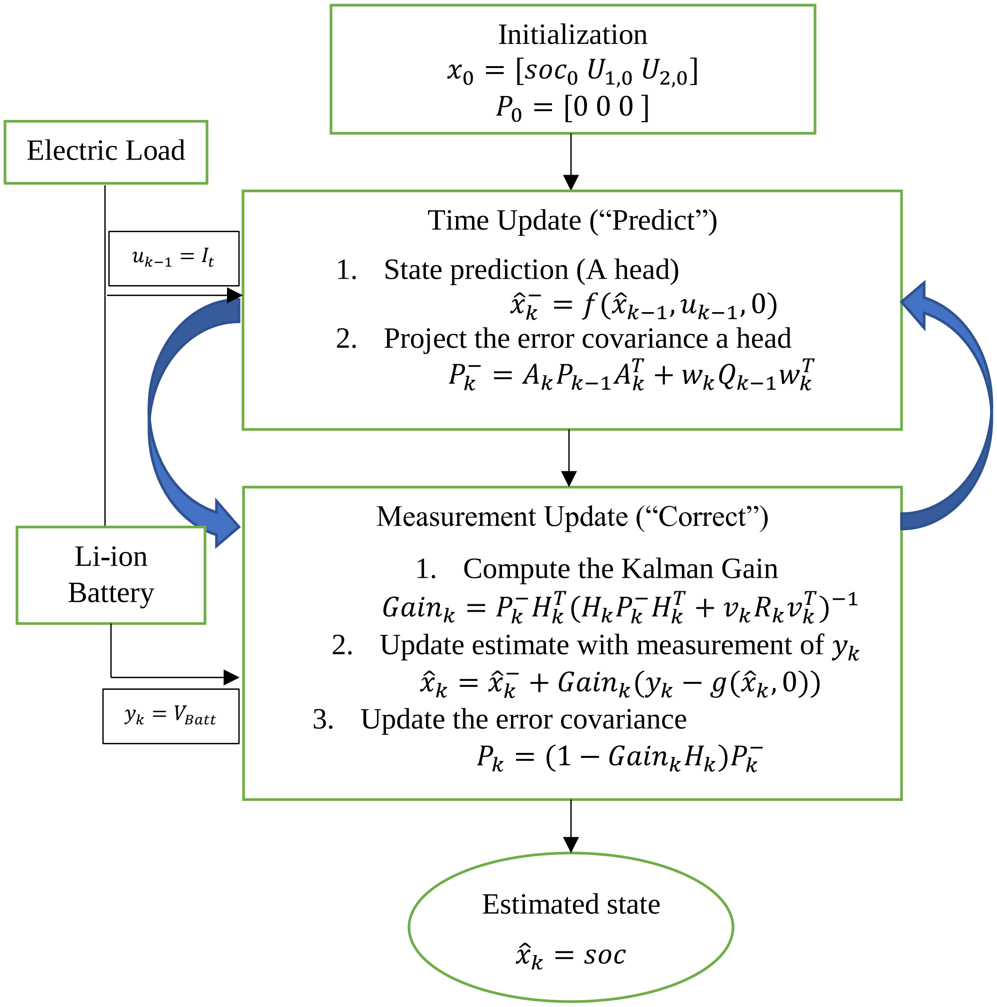

Initialization
$x_0 = [soc_0\ U_{1,0}\ U_{2,0}]$
$P_0 = [0\ 0\ 0\ ]$
Electric Load
$u_{k-1} = I_t$
Time Update ("Predict")
1. State prediction (A head)
$\hat{x}_k^- = f(\hat{x}_{k-1}, u_{k-1}, 0)$
2. Project the error covariance a head
$P_k^- = A_k P_{k-1} A_k^T + w_k Q_{k-1} w_k^T$
Li-ion Battery
$y_k = V_{Batt}$
Measurement Update ("Correct")
1. Compute the Kalman Gain
$Gain_k = P_k^- H_k^T (H_k P_k^- H_k^T + v_k R_k v_k^T)^{-1}$
2. Update estimate with measurement of y_k
$\hat{x}_k = \hat{x}_k^- + Gain_k(y_k - g(\hat{x}_k, 0))$
3. Update the error covariance
$P_k = (1 - Gain_k H_k) P_k^-$
Estimated state
$\hat{x}_k = soc$

Figure 3. 5: Flow chart of Extended Kalman Filter for SOC estimation.

3.9 Simulation Results with Comparison and Case Study

To get the pulse discharge curve of Li-ion battery a constant 5A current discharge was used at 25^0 C in Figure 3.2. based on battery 2nd order RC ECM model parameters are estimated shown in Table 4.1 and parameters are assumed as constants from the equation (3.15) OCV is the function of SOC were used to estimate SOC.

Simulation and estimation results are performed in MATLAB 2016 environment. SOC estimation using PF algorithm in Figure 3.4. During the simulation, 100 particles were chosen however it was observed that PF algorithm with several simulations it showed the almost same SOC estimated error with 500 particles at the cost of high computational time. So, which implies that increasing the particles which leads to increase in the PF computational time. The red solid line represents the estimated result of SOC from PF algorithm. According to Equation (3.1), the ground truth or actual SOC values was obtained by integrating the discharge current per second. In particle filtering, initializing the state is very important to get a significant result. So, the initial SOC is chosen uniformly at 0.80 to 0.90 percentage of SOC. At 150 sections the particles are converged significantly and estimate SOC almost close to the true value. The value of process and measurement non-gaussian noise are $\omega_k = 1e - 6$ and $\vartheta_k = 1e - 4$.

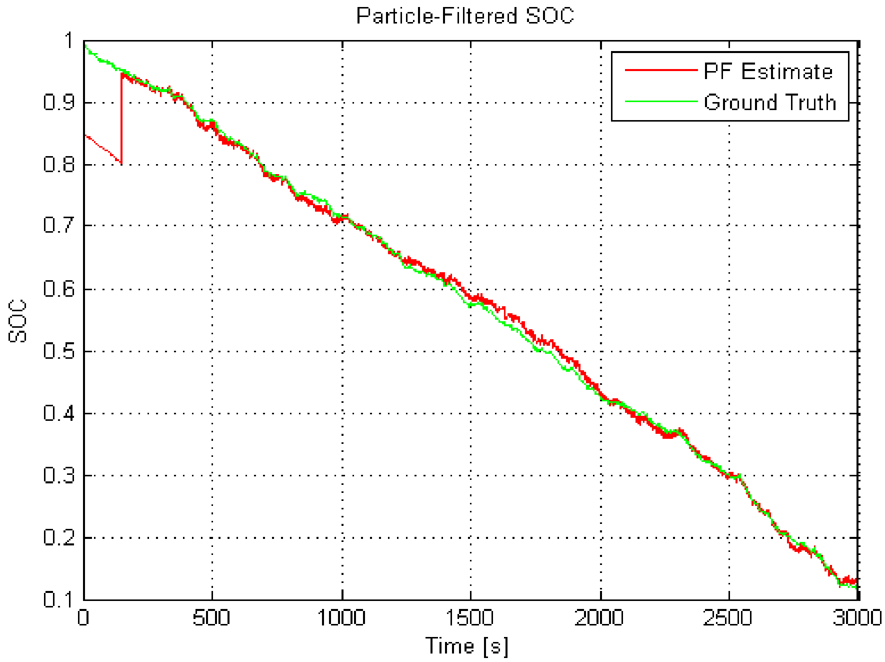


Figure 3. 6: SOC estimation with particle filtering.

In terms of accuracy of SOC estimation, figure 3.6 shows the error in SOC estimation with PF method. The mean error of true state and PF state for SOC estimation is 2.65(%). This demonstrates the significance of the battery dynamic model.

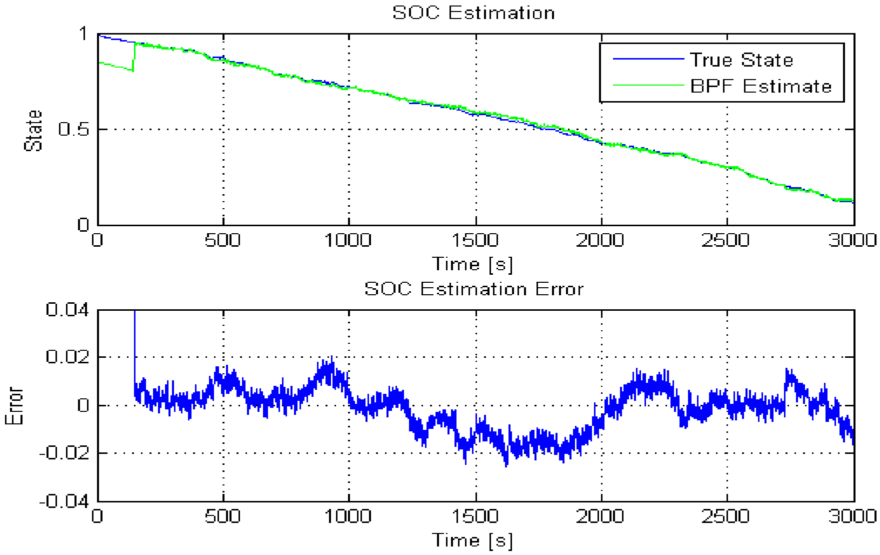


Figure 3. 7: SOC estimation error with particle filtering.

To evaluate the performance of the proposed PF method based SOC estimation algorithm, a comparison with EKF and KF based estimation methods are made. The reason for the comparison is to how effective the proposed model and performance of Li-ion battery for dynamic model.

Estimated results are a comparison with extended Kalman filtering and Kalman filtering algorithms. These algorithms are faster the convergence and lower the accuracy in both EKF and KF. However, the EKF algorithm is nonlinear model observation and faster rate of convergence still not accurate to minimize the error of the SOC estimation in Figure 3.5.

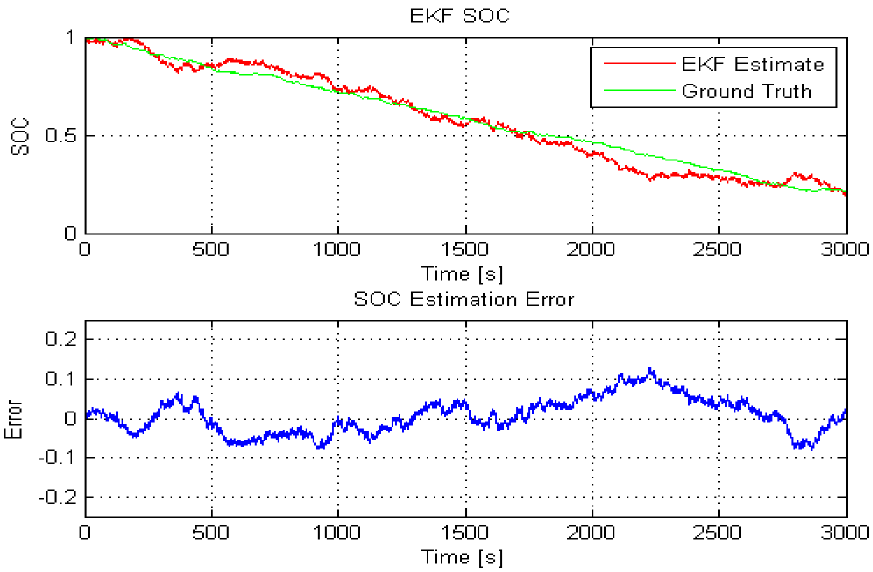


Figure 3. 8: SOC estimation with EKF and estimation error.

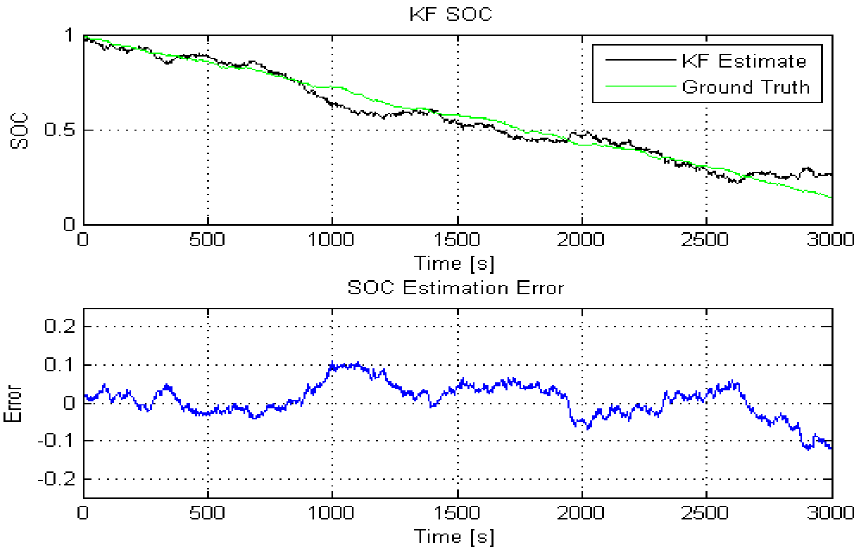


Figure 3. 9: SOC estimation with KF and estimation error.

The comparison results are evidence that PF algorithm is more accurate than EKF and KF algorithm to estimate SOC of Li-ion battery. Figure 3.9, shows that comparison of three filtering algorithms and which evidence that PF algorithm is more accurately estimate SOC online with 2nd RC ECM model. The accuracy is depending on the complexity of the dynamic model of battery.

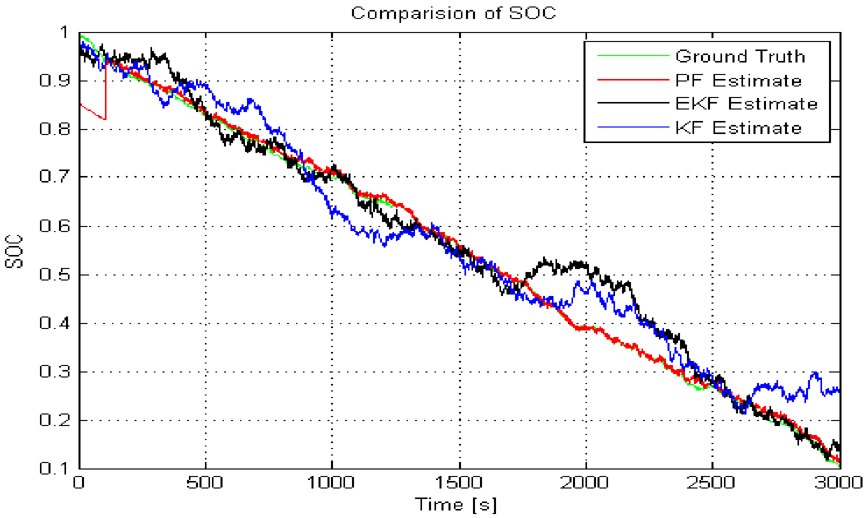


Figure 3. 10: SOC comparison results with PF, EKF, and KF.

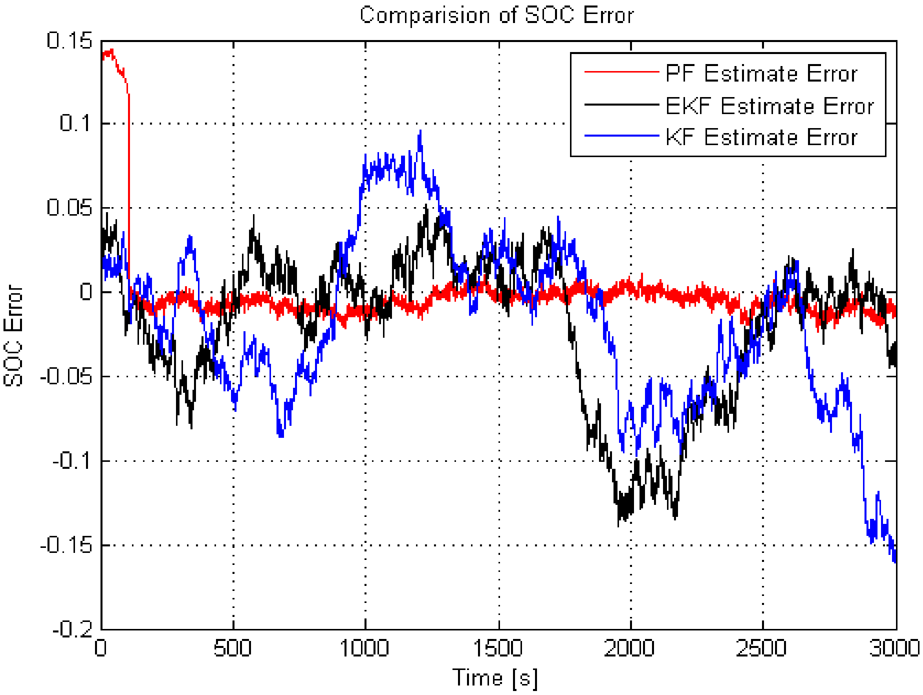


Figure 3. 11: Comparison of error significance with PF, EKF, KF.

Table 3.2: SOC estimation error

Estimation method	PF (%)	EKF (%)	KF (%)
RMSE	5.239	9.429	10.720
Std. RMSE	5.749	10.348	11.764
Error Mean	2.654	4.340	5.103

Case Study:

Let, supposed to have a well know nonlinear system whose discrete time ($\Delta T = 1s$) model followed by [70]

$$x(k) = \frac{1}{2}x(k-1) + \frac{25x(k-1)}{1+x^2(k-1)} + 8\cos\big(1.2(k-1)\big) + w(k-1) \tag{3.40}$$

$$y(k) = \frac{1}{20}x^2 + v(k) \tag{3.41}$$

Where $w \sim \mathcal{N}(0, R_{ww})$, $v \sim \mathcal{N}(0, R_{vv})$ are white gaussian noise. The initial condition is $x(0) \sim \mathcal{N}(0.1, 5)$ and the noise covariances are $R_{vv} = 1$ and $R_{ww} = 10$.

This problem becomes a benchmark for many filtering algorithms. It is highly nonlinear. The case study uses particle filtering algorithm to demonstrate the results. Figure 3.11, represents the simulated state and simulated measurement from the nonlinear state space model

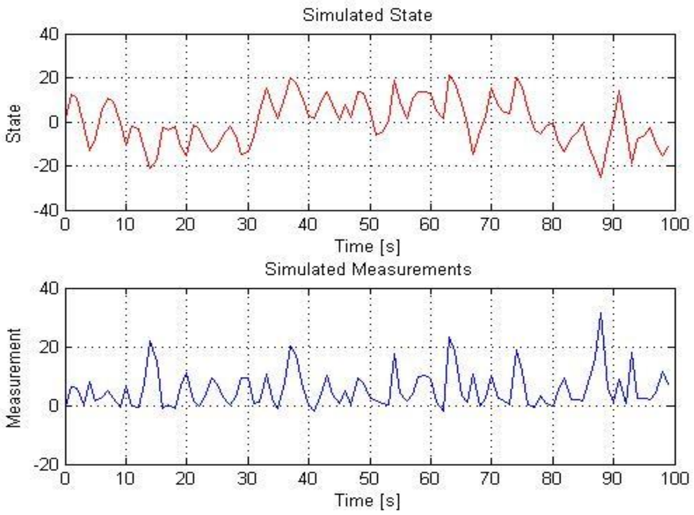


Figure 3. 12: Simulated state and measurement.

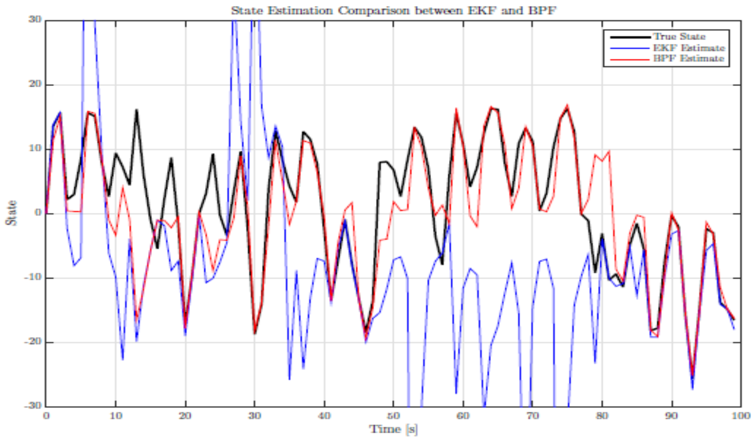


Figure 3. 13: State estimated state comparison between EKF and BPF.

This demonstrates a nonlinear system state estimate with PF. For doing so, 100 particles are initialized with $x(0) \sim \mathcal{N}(0.1, 5)$. These simulation results are evidence that for a complex nonlinear dynamic system with particle filtering using non-gaussian distribution to estimate the state with very accurate and robustness. Figure 3.12, represents the root mean square error for the true state to estimated state with particle filtering and it gives the error of 4.75. this error shows that significant of the particle filtering algorithm.

4. Online Capacity and Remaining Useful Life Assessment

4.1 Li-ion Battery Capacity Degradation Model

The battery capacity data used in this thesis are provided by National Aeronautics and Space Administration (NASA) Ames Prognostics Center of Excellence [71], where 18,650-sized rechargeable Li-ion batteries were tested. Li-ion batteries in batches were run through three different operational profiles: charge, discharge, and impedance, described as follows:

Charge step: charging was conducted at a constant current (CC) level of 1.5 A until the charge voltage reached 4.2 V. Charging was continued in constant voltage (CV) mode until the charge current dropped to 20 mA.

Discharge step: discharging was conducted in CC mode until the discharge voltage reached a predefined cutoff voltage.

Impedance measurement: measurement was performed through an electrochemical impedance spectroscopy (EIS) frequency sweep from 0.1 Hz to 5 kHz. Repeated charge and discharge steps can induce the degradation of Li-ion batteries. Meanwhile, impedance measurements provide insights into internal battery parameters, which vary as degradation progresses. During an entire C-D cycle, charge and discharge steps may be continuous or discontinuous for the impedance measurement. The experiments were terminated when the battery capacity decreased by 30% of original capacity.

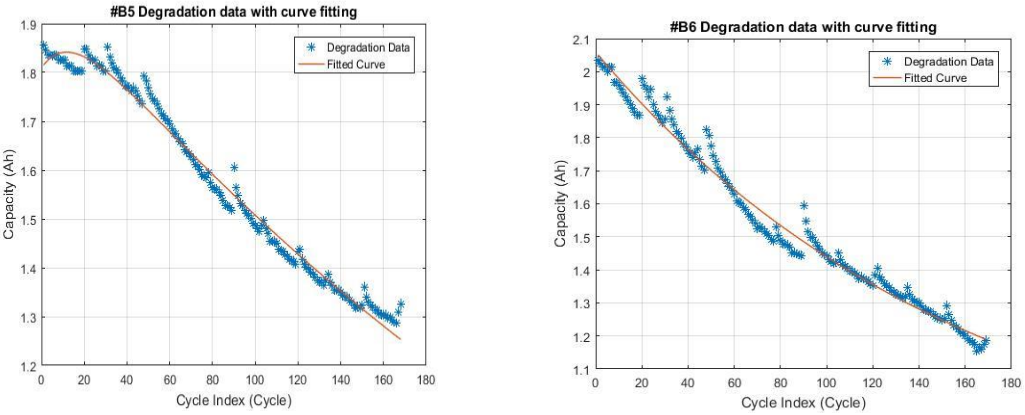
Capacity is the amount of charge a battery holds in its fully charged state and can be described as integrating the current over time.

$$Q = \int_{t_{charge}}^{t_{discharge}} I dt \tag{4.1}$$

Where Q is the battery capacity, I is the current flow of the battery, $t_{discharge}$ is the time at battery fully discharged state, and t_{charge} is the time at battery fully charged state. The capacity will gradually irreversible with the various aging process like SEI layer formation and failure processes like Electrode passivation and corrosion. Generally, for many applications, it is accepted that 80 % of rated capacity is the failure threshold and consider to be End of Life (EOL). Figure 4.1 shows the capacity degradation model with an exponential growth model is used to fit the degradation data. To obtain an accurate exponential model, the Matlab curve fitting toolbox is used to fit the degradation data and the data found that the regression process can be expressed as by an empirical model.

$$Q = a\exp(bk) + c\exp(dk) \tag{4.2}$$

Here, Q is the capacity of the battery, k is the cycle number, and $a, b, c,$ and d are the model parameters. As long as the parameters are accurately estimated, the exponential model can successfully describe the degradation phenomenon of battery B5, B6, B7, and B8. The fitting model unveils the model parameter of the known batteries.



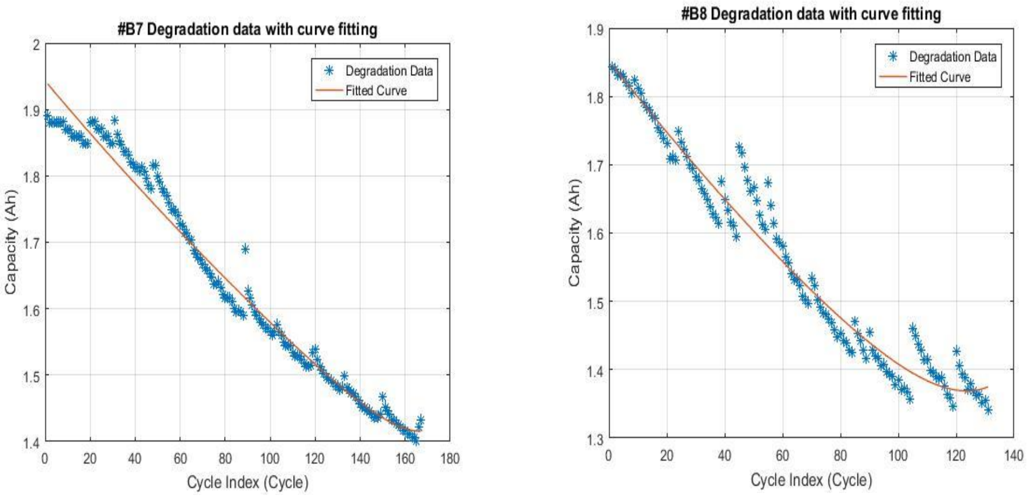


Figure 4. 1: (a), (b), (c), (d) are the battery degradation data with curve fitting.

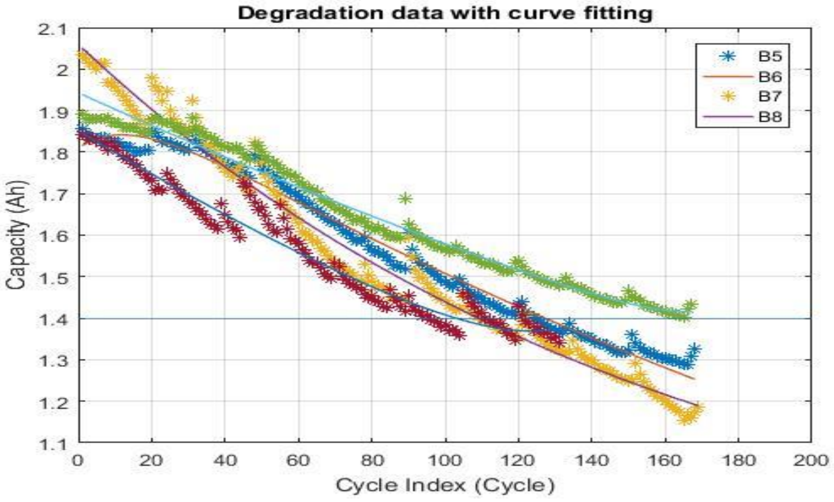


Figure 4. 2: Four batteries capacity degradation data with threshold limit.

Table 4. 1: Identified model parameters

Battery ID	a	b	c	d
B5	1.979	-0.002719	-0.1697	-0.06942
B6	1.338	-0.006239	0.7215	0.00001373
B7	1.943	-0.002074	0.000000256	0.07184
B8	1.852	-0.002914	0.0001881	0.04868

Table 4. 2: Goodness of fit statistic

Battery ID	SSE	R^2	Adjusted R^2	RMSE
B5	0.08368	0.986	0.9859	0.02259
B6	0.2046	0.9807	0.9804	0.03521
B7	0.09072	0.979	0.9786	0.02359
B8	0.1177	0.9614	0.9605	0.03045

The uncertainty of the battery capacity degradation is an exponential model due to repeated cycling up to acceptable threshold limit but, it can also arise from various sources such as ambient temperature, discharge current rate, depth of discharge, and age with time so, to predict the remaining useful life of the battery with SMC algorithm, the B5 battery labeled is chosen because of the goodness of fit in statistic.

4.2 Remaining Useful Life Online Assessment Model

PF is a novel class of nonlinear filters that combines Bayesian learning techniques with importance sampling to provide good state tracking performance while keeping the computational load tractable. The idea is to represent the system state as PDF that is approximated by a set of particles (points) representing sampled values from the unknown state space and set of associated weight denoting discrete probability masses. The particle is generated from a prior estimate of the state PDF, propagated through time using a nonlinear process model, and recursively updated from measurements through a measurement model. the main advantages of PF here is that model parameters can be included as part of a vector to be tracked, thus performing model identification in conjunction with state estimation. After the model has been tuned to reflect the dynamics of the specific system being tracked it can be used to propagate the particle till the failure threshold to give the RUL.

After determining the initial parameters values and collecting the capacity data, the parameters can be updated based on Bayes rule. In order to model the uncertainty, it is assumed that the parameters: *a*, *b*, *c*, and *d* along with the error in the regression model are subjected to a Gaussian distribution.

4.3 State Space Model on Degradation for RUL Assessment

The system transition and measurement function can be written as

$$x_k = [a_k; b_k; c_k; d_k] \tag{4.3}$$

$$\begin{cases} a_k = a_{k-1} + \omega_a & \omega_a \sim N(0, \sigma_a) \\ b_k = b_{k-1} + \omega_b & \omega_b \sim N(0, \sigma_b) \\ c_k = c_{k-1} + \omega_c & \omega_c \sim N(0, \sigma_c) \\ d_k = d_{k-1} + \omega_d & \omega_d \sim N(0, \sigma_d) \end{cases} \tag{4.4}$$

$$Q_k = a_k \exp(b_k k) + c_k \exp(d_k k), \quad n_k \sim (0, \sigma_n) \tag{4.5}$$

Here, Q_k is the capacity measurement at cycle k, $N(0, \sigma_n)$ is the Gaussian noise with zero mean and standard deviation σ. Use the Sequential Monte Carlo in this simulation, the capacity can be estimated by

$$Q_k = \sum_{i=1}^{N} Q_k^i = \sum_{i=1}^{N} \left[a_k^i \cdot \exp\left(b_k^i \cdot k\right) + c_k^i \cdot \exp\left(d_k^i \cdot k\right) \right] \tag{4.6}$$

Then, the *p*-step prediction at cycle *k* can be written as

$$Q_{k+p} = \sum_{i=1}^{N} Q_{k+p}^i \tag{4.7}$$

$$Q_{k+p}^i = a_k^i \exp\left(b_k^i (k+p)\right) + c_k^i \exp\left(d_k^i (k+p)\right) \tag{4.8}$$

The estimated pdf of the prediction is

$$P\left(Q_{k+p}\middle|Q_{0:k}\right) \approx \sum_{i=1}^{N} \omega_k^i \, \delta(Q_{k+p} - Q_{k+p}^i) \tag{4.9}$$

In this capacity degradation data from NASA, 0.73 is the threshold limit is chosen to see the actual failure cycle. So, the life distribution of the RUL prediction at cycle k can be solved by

$$0.73 = a_k^i \exp\left(b_k^i L_k^i\right) + c_k^i \exp\left(d_k^i L_k^i\right) \tag{4.10}$$

$$P(L_k|Q_{0:k}) \approx \sum_{i=1}^{N} \omega_k^i \delta(k - L_k^i) \tag{4.11}$$

Where, *k* is the actual failure cycle number.

4.4 Experimental Results and Discussion

In this section, a case studies are conducted to validate the proposed SMC approach. The data from NASA prognostic center is chosen the for-case studies. Where four batteries B5, B6, B7, and B8 are used to elicit the initial model parameter initialization for the different battery model including initializing the model parameters and their corresponding variance. In section 4.1, a curve fitting model is conducted to choose the best model fit for RUL and B5 battery is chosen for best goodness and statistical fit compared to the reaming batteries so, the model parameters are initialized using the average value through curve fitting based on the battery training samples. Nonlinear least square fitting is performed to initialize the parameters of models then the initial values of parameters *a*, *b*, *c* and *d are* -9.86e-7, 5.752e-2, 8.983e-1 and -8.34e-4, respectively. The battery simulations are performed in MATLAB R2016b environment. In the experiment, the first 50,100,150 cycle capacity measured data points are chosen randomly used to predict the RUL of battery B5 using SMC method. The actual end of life threshold limit of the battery is chosen 80% of the capacity at the beginning of the life of the battery. The results are shown in Figures 4.3, 4.5, and 4.5.

The root means square error (RMSE) gives the standard deviation of the model prediction error. A smaller the value indicates the better model performance. The formula for the RMSE is given as

$$\text{RMSE}=\sqrt{\frac{1}{n}\sum_{k=1}^{n}\left(\hat{Q}_k - Q_k\right)^2} \tag{4.12}$$

The RUL prediction error (E_{RUL}) is the absolute value of the difference between the number of real cycles till 80% of rated capacity in Equation (4.10) and the predicted number of cycles. The formula for the RUL prediction error is given as follows:

$$E_{RUL} = \left|RUL_{real} - RUL_{prediction}\right| \tag{4.13}$$

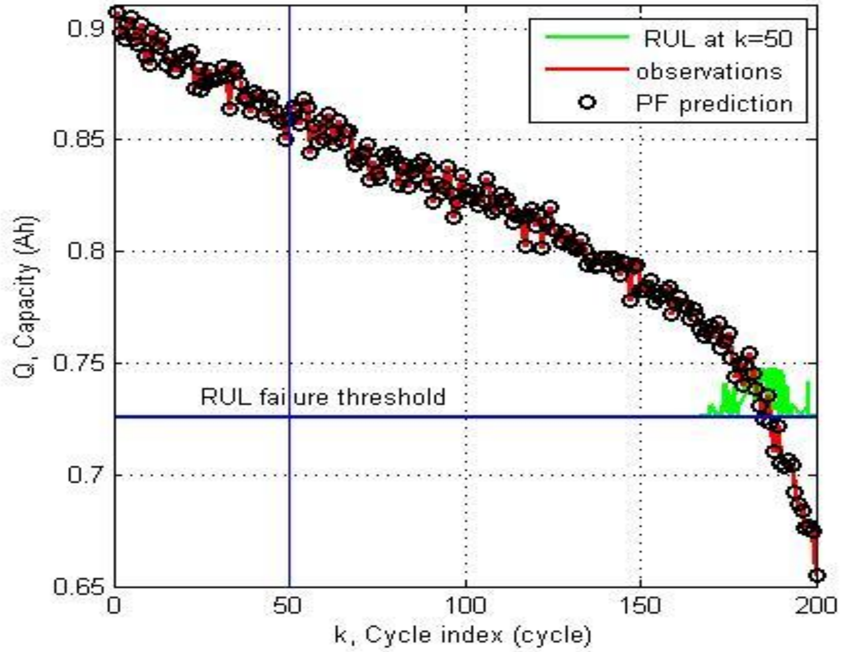


Figure 4. 3: SMC prediction results at 50 cycles for the battery of B5.

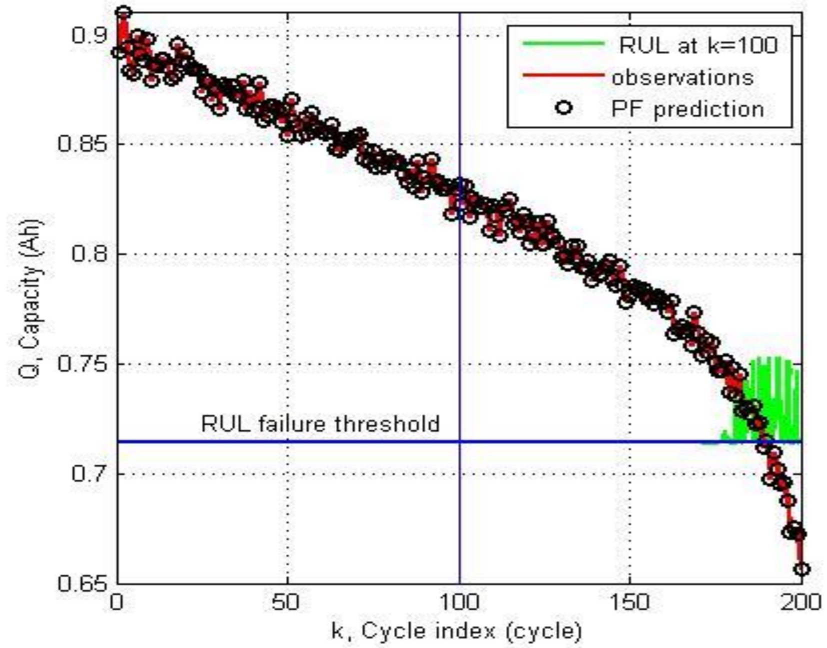


Figure 4. 4: SMC prediction results at 100 cycles for the battery of B5.

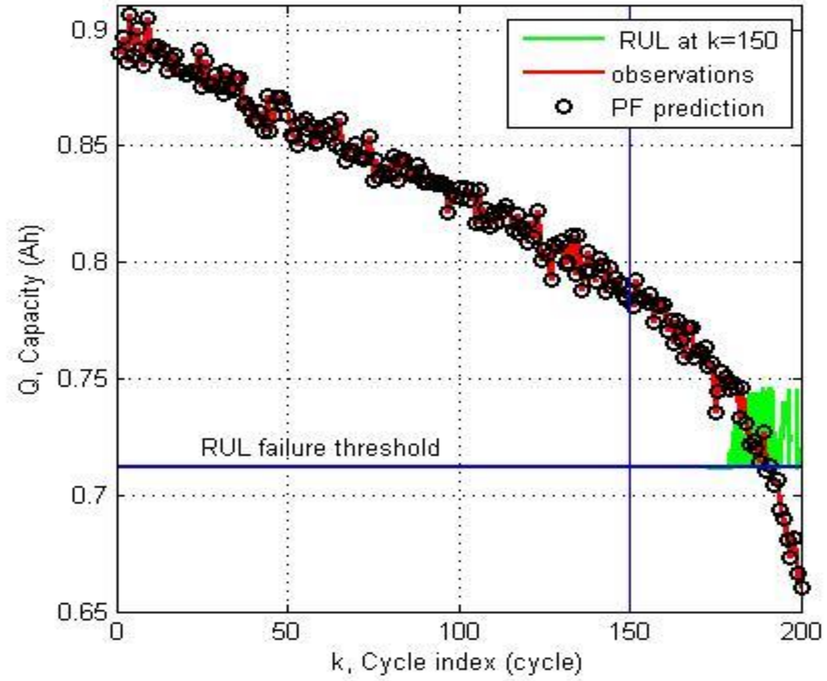


Figure 4. 5: SMC prediction results at 150 cycles for the battery of B5.

Figures (4.3) - (4.5) shows that EOL prediction results for three different prediction cycles i.e., 50th cycle, 100th cycle, and 150th cycle respectively. For each prediction point, the mean value of EOL obtained from the RUL pdfs in Equation 4.11 (shown in green colors) and compared with actual EOL obtained from the normalized experimental B5 battery capacity data. In the simulation work, 100 particles are considered in PF algorithm. A comparison table for the different prediction points are listed in Table 4.3.

Table 4. 3: Comparison of EOL prediction for different cycle

Prediction cycle	Actual EOL (cycle)	Predicted EOL(cycle)	Prediction error (cycle)
K=150	189	189.30	0.30
K=100	189	187.17	1.83
K=50	189	185.27	3.78

From Table 4.3, shows that as the point of prediction approaches the actual EOL, the prediction error reduces gradually. The standard deviation of the EOL prediction is shown in Table 4.4.

Table 4. 4: Comparison of standard deviation EOL prediction at different cycle

Prediction Cycle	Standard Deviation (Cycle)
K=150	7.14
K=100	7.52
K=50	8.28

5. Conclusion and Future Works

5.1 Conclusion

In this thesis, a method for Li-ion battery online SOC estimation using the algorithm of PF is proposed. An accurate 2nd RC ECM with parameters represented by the function of SOC for Li-ion battery is established in MATLAB. A state space model is developed for 2nd RC ECM, that uses the PF algorithm to estimate online SOC. Using EKF and KF algorithm are also presented to estimate online SOC for comparisons and significance of battery model performance. From the simulation results, it demonstrated that PF provides an accurate estimation. In conclusion, the proposed method for battery 2nd RC ECM has superior performance on online SOC estimation for Li-ion battery. This approach uses a statistical characterization of battery profile to estimate the SOC of Li-ion battery.

A new model for capacity degradation of Li-ion battery is proposed. This capacity degradation model is considered as an empirical model because of the capacity degradation is nonlinear so, it is quite capable of nonlinear and easily implemented in PF based framework to make effect RUL prediction for Li-ion batteries. The prediction results obtained so far have been quite satisfactory; however, there is still a lot of considerable room for improvement. The prediction of RUL has been obtained using PF algorithm based on capacity degradation estimation. Then the predicted RUL has been validated with measured RUL from the given threshold limit experimental data. All the test until the end of the life cycle has been carried out at the same discharge *C*-rate. The model accuracy can be still improved by incorporating the influence of various different parameters like *C*-rate, temperature, changes in DOD, and impedance etc.

5.2 Future Works

In future works, ECM model can be developed with consideration of SOC, SOH, and Temperature to standard PF algorithm and Unscented Particle Filtering is the idea of a combination of PF and UKF. It utilizes new coming measurement in the prediction and the prediction accuracy.

The RUL prediction work presented here is an initial investigation of Li-ion battery prognostic health management system. The battery capacity data is used to model the battery degradation trend as an empirical model. however, in future work the ECM battery dynamic can be important to implement for onboard BMS. An Unscented Particle Filtering can have to introduce in future work to estimate SOC and RUL of Li-ion battery because of PF resampling causes particle improvement in the application, choosing reasonable proposal distribution becomes a promising choosing to solve the have higher degeneracy problem.

References

1. MIT Electric Vehicle Team. (2008, December) A Guide to Understanding Battery Specifications. [Online]. http://mit.edu/evt/summary_battery_specifications.pdf.
2. J. Chiasson and B. Vairamohan, "Estimating the state of charge of a battery," *Proceedings of the 2003 American Control Conference*, vol. 4, no. 2, pp. 2863-2868, 2003.
3. K. Smith and Chao-Yang Wang, "Solid-state diffusion limitations on pulse operation of a lithium ion cell for hybrid electric vehicles," *Journal of Power Sources*, vol. 161, no. 1, pp. 628-639, 2006.
4. Marcos E. Orchard and George J. Vachtsevanos, “A particle-filtering approach for on-line fault diagnosis and failure prognosis,” *Transactions of the Institute of Measurement and Control* 2009.
5. V. Pop, H.J. Bergveld, D. Danilov, P.P.L. Regtien, and P.H.L. Notten, *Battery Management Systems*.: Springer, 2008.
6. Bruce Dunn, Haresh Kamath, and Jean-Marie Tarascon, "Electrical Energy Storage for the Grid: A Battery of Choices," *Science*, vol. 334, pp. 928-935, 2011.
7. M. Barak, *Electrochemical Power Sources: Primary and Secondary Batteries*. Stevenage, UK: IET, 1980.
8. Thomas Reddy, *Handbook of batteries*.: McGraw-Hill Professional; 4th edition, 2010.
9. M.H.A Davis, Lectures on stochastic control and nonlinear filtering, Tata institute of fundamental research, *Springer-Verlag 1984.*
10. Jazwinski, A.H. (1970). Stochastic Processes and filtering Theory. Academic Press.
11. Athans, M., R.P. Wisher and A. Bertolini (1968). Suboptimal State Estimation for Continuous-Time Nonlinear Systems with Discrete Noisy Measurements. IEEE Transactions on Automatic Control (5), 504–514.
12. Julier, S. J. and J. K. Uhlmann (2004). Unscented filtering and nonlinear estimation. IEEE review 92(3), 401–421.
13. N¨orgaard, M., N.K. Poulsen and O. Ravn (2000b). New developments in state estimation for nonlinear systems. Automatica 36(11), 1627–1638.
14. Sorenson, H.W. and D.L. Alspach (1970). Approximation of Density Function by a Sum of Gaussians for Nonlinear Bayesian Estimation. In: Proceedings of the 1st Symposium for Nonlinear Estimation Theory and Its Applications. San Diego. pp. 88–90.
15. Sorenson, H.W. and D.L. Alspach (1971). Recursive Bayesian Estimation Using Gaussian Sums. Automatica 7, 465–479.
16. Simandl, M. (1996). State Estimation for Non-Gaussian Models. In: ˇ Preprints of the 13th IFAC World Congress. Vol. 2. IFAC. Elsevier Science. San Francisco, USA. pp. 463–468.
17. Bucy, R.S. (1969). Bayes’ Theorem and Digital Realization of Nonlinear filters. J. Astronaut. Sci. 87, 493–500.
18. Bucy, R.S. and K.D. Senne (1971). Digital Synthesis of Non-linear Filters. Automatica 7, 287– 298.

19. Simandl, M. and O. Straka (2002). Nonlinear estimation by particle filters and Cram´er-Rao ˇ bound. In: Proceedings of the 15th triennial world congress of the IFAC. Barcelona. pp. 79–84.
20. Gordon, N., D. Salmond and A.F.M. Smith (1993). Novel approach to nonlinear/ non- Gaussian Bayesian state estimation. IEE proceedings-F 140, 107–113.
21. Doucet, A., de Freitas, N. and Gordon, N., Eds.) (2001). Sequential Monte Carlo Methods in Practice. Springer.
22. Geweke, J. (1989). Bayesian inference in econometric models using Monte Carlo Integration. Econometrika 24, 1317–1399.
23. Marshall, A. (1956). The use of multi-stage sampling schemes in Monte Carlo computation. In: Symposium on Monte Carlo methods (M. Meyer, Ed.). New York. pp. 123–140.
24. Handshin, J.E. and D.Q. Mayne (1969). Monte Carlo techniques to estimate the condidional expectation in multi-stage non-linear filtering. International Journal of Control 9, 547– 559.
25. Metropolis, N. and S. Ulam (1949). The Monte Carlo method. jasa 44, 335–341.
26. Marc Doyle, Thomas F Fuller, and John Newman, "Modeling of Galvano static Charge and Discharge of the Lithium / Polymer / Insertion Cell," *Journal of The Electrochemical Society*, vol. 140, no. 6, pp. 1526-1533, 1993.
27. Thomas F Fuller, Marc Doyle, and John Newman, "Simulation and Optimization of the Dual Lithium Ion Insertion Cell," *Journal of The Electrochemical Society*, vol. 141, no. 1, pp. 1-10, 1994.
28. James L. Lee, Andrew Chemistruck, and Gregory L. Plett, "One-dimensional physics-based reduced-order model of lithium-ion dynamics," *Journal of Power Sources*, vol. 220, no. 0, pp. 430-448, 2012.
29. Antoni Szumanowsk and Yuhua Chang, "Battery Management System Based on Battery Nonlinear Dynamics Modeling," *IEEE Transaction on Vehicular Technology*, vol. 57, no. 3, pp. 1425-1432, 2008.
30. W.Y. Sean, J.C. Ke Y.H. Chiang, "A comparative study of equivalent circuit models for Li-ion batteries," *J. Power Sources*, pp. 3921–3932, 2011.
31. Ng, K.S.; Moo, C.-S.; Chen, Y.-P.; Hsieh, Y.-C. Enhanced coulomb counting method for estimating state-of-charge and state-of-health of lithium-ion batteries. Appl. Energy 2009, 86, 1506–1511.
32. Zhang, H.; Zhao, L.; Chen, Y. A lossy counting-based state of charge estimation method and its application to electric vehicles. Energies 2015, 8, 13811–13828.
33. Feng, F.; Lu, R.G.; Zhu, C.B. A combined state of charge estimation method for lithium-ion batteries used in a wide ambient temperature range. Energies 2014, 7, 3004–3032.
34. Lee, S.; Kim, J.; Lee, J.; Cho, B.H. State-of-charge and capacity estimation of lithium-ion battery using a new open-circuit voltage versus state-of-charge. J. Power Sources 2008, 185, 1367–1373.
35. Dang, X.; Yan, L.; Xu, K.; Wu, X.; Jiang, H.; Sun, H. Open-circuit voltage-based state of charge estimation of lithium-ion battery using dual neural network fusion battery model. Electrochim. Acta 2016, 188, 356–366.
36. Neeta Khare, Rekha Govil. “Modeling Automotive Battery Diagnostics”. Power Electronics Technology. 2008. pp.36-41.

37. T. Hansen, C. Wang,” Support vector based battery state of charge estimator” J. Power Sources 141 (2005) 351-358.
38. A. Zenati, P. Desprez, and H. Razik, “Estimation of the SOC and the SOH of li-ion batteries, by combining impedance measurements with the fuzzy logic inference,” IECON 2010 - 36th Annual Conference on IEEE Industrial Electronics Society, pp. 1773-1778, 2010.
39. Barbarisi, O.; Vasca, F.; Glielmo, L. State of charge Kalman filter estimator for automotive batteries. Control Eng. Pract. 2006, 14, 267–275.
40. Sepasi, S.; Roose, L.R.; Matsuura, M.M. Extended Kalman filter with a fuzzy method for accurate battery pack state of charge estimation. Energies 2015, 8, 5217–5233.
41. Tian, Y.; Xia, B.; Sun, W.; Xu, Z.; Zheng, W. A modified model based state of charge estimation of power lithium-ion batteries using unscented Kalman filter. J. Power Sources 2014, 270, 619–626.
42. Shi, P.; Zhao, Y.W.; Shi, P. Application of unscented Kalman filter in the SOC estimation of Li-ion battery for Autonomous Mobile Robot. In Proceedings of the 2006 IEEE International Conference on Information Acquisition, Weihai, China, 20–23 August 2006; pp. 1279–1283.
43. Fei, Z.; Guangjun, L.; Lijin, F. Battery state estimation using unscented Kalman filter. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, 12–17 May 2009; pp. 1863–1868.
44. Zhiwei, H.; Mingyu, G.; Caisheng, W.; Leyi, W.; Yuanyuan, L. Adaptive state of charge estimation for Li-ion batteries based on an unscented kalman filter with an enhanced battery model. Energies 2013, 6, 4134–4151.
45. Xia, B.Z.;Wang, H.Q.;Wang, M.W.; Sun,W.; Xu, Z.H.; Lai, Y.Z. A new method for state of charge estimation of lithium-ion battery based on strong tracking cubature Kalman filter. Energies 2015, 8, 13458–13472.
46. H. W. He, X. W. Zhang, R. Xiong, and Y. L. Xu, “Online model-based estimation of state-of-charge and open-circuit voltage of lithium-ion batteries in electric vehicles,” Energy, vol. 39, pp. 310–318, Mar. 2012.
47. Kalman, R.E., A new approach to linear filtering and prediction problems. J. Fluids Eng. 1960, 82, 35–45.
48. Xiong, B.; Zhao, J.;Wei, Z.; Skyllas-Kazacos, M. Extended Kalman filter method for state of charge estimation of vanadium redox flow battery using thermal-dependent electrical model. J. Power Sources 2014, 262, 50–61.
49. Wang, S.; Fernandez, C.; Shang, L.; Li, Z.; Li, J. Online state of charge estimation for the aerial lithium-ion battery packs based on the improved extended Kalman filter method. J. Energy Storage 2017, 9, 69–83.
50. Wan, E.A.; van der Merwe, R. The unscented Kalman filter for nonlinear estimation. In Proceedings of the IEEE 2000 Adaptive Systems for Signal Processing, Communications, and Control Symposium (Cat. No. 00EX373), Lake Louise, AB, Canada, 4 October 2000; pp. 153–158.
51. Mazaheri, A.; Radan, A. Performance evaluation of nonlinear Kalman filtering techniques in low speed brushless DC motors driven sensor-less positioning systems. Control Eng. Pract. 2017, 60, 148–156.

52. Ye, M.; Guo, H.; Cao, B. A model-based adaptive state of charge estimator for a lithium-ion battery using an improved adaptive particle filter. Appl. Energy **2017**, 190, 740–748.
53. Wang, Y.; Zhang, C.; Chen, Z. A method for state-of-charge estimation of LiFePO4 batteries at dynamic currents and temperatures using particle filter. J. Power Sources 2015, 279, 306–311.
54. Markus, Einhorn., Fiorentino, Valero., Christain, Kral. A Method for Online Capacity Estimation of Lithium ion Battery Cell Using the State of Charge and Transferred charge. *IEEE Transaction of Industrial Application, Vol, 48, No.2, March 2012.*
55. He, Z.; Gao, M.; Wang, C.; Wang, L.; Liu, Y. Adaptive state of charge estimation for Li-ion batteries based on an unscented kalman filter with an enhanced battery model. *Energies* 2013, *6*, 4134–4151.
56. Williard, N.; He, W.; Osterman, M.; Pecht, M. Comparative analysis of features for determining state of health in lithium-ion batteries. *Int. J. Progn. Health Manag.* 2013, *1*, 1–7.
57. Zhang, J.; Lee, J. A review on prognostics and health monitoring of Li-ion battery. *J. Power Sources* 2011, *196*, 6007–6014.
58. Liu, D.; Wang, H.; Peng, Y.; Xie, W.; Liao, H. Satellite lithium-ion battery remaining cycle life prediction with novel indirect health indicator extraction. *Energies* 2013, *6*, 3654–3668.
59. Xing, Y.; Ma, E.W.M.; Tsui, K.L.; Pecht, M. Battery management systems in electric and hybrid vehicles. *Energies* 2011, *4*, 1840–1857.
60. Goebel, K.; Saha, B.; Saxena, A.; Celaya, J.R.; Christophersen, J.P. Prognostics in battery health management. *IEEE Instrum. Meas. Mag.* 2008, *11*, 33–40.
61. Williard, N.; He, W.; Hendricks, C.; Pecht, M. Lessons learned from the 787 Dreamliner issue on lithium-ion battery reliability. *Energies* 2013, *6*, 4682–4695.
62. Pecht, M.G. *Prognostics and Health Management of Electronics*; John Wiley & Sons, Ltd.: New York, NY, USA, 2008.
63. Bo, S.; Shengkui, Z.; Rui, K.; Pecht, M.G. Benefits and challenges of system prognostics. *IEEE Trans. Reliab.* 2012, *61*, 323–335.
64. Camci, F.; Chinnam, R.B. Health-state estimation and prognostics in machining processes. *IEEE Trans. Autom. Sci. Eng.* 2010, *7*, 581–597.
65. Escobar. L.A, Meeker. W.Q, Luis. A, William. E, Meeker. Q. A review of accelerated test models. *Stat.Sci.2006,21,552-577.*
66. Chen. C, Pecht.M. Prognostics of Lithium-Ion batteries using Model Based and data-Driven Methods. *In Proceedings of the 2012 Prognostic systems and Health management conference, Beijing, China, 23-25 May 2012*.
67. Lin. C, Tang. A, Wang. W. A review of SOH estimation methods in Lithium-ion Batteries in Electric Vehicles Applications. *Energy Proceeding,2015, 75,1920-1925.*
68. Liu. D, Luo. Y, Liu. J, Peng. Y, Guo. L, Pecht. M. Lithium-ion Battery Remaining Useful Life Estimation Based on Fusion Nonlinear Degradation AR model and RPF Algorithms. *Neural Comput. Appl. 2013,25, 557-572.*

69. Si, X.S, Wang, W, Hu, C.H, Zhou, D.H. Remaining useful life estimation -A review on statistical data driven approaches. *Eur. J. Oper. Res. 2011, 213, 1-14.*

70. N. Tapia Rivas, “Bayesian estimation classical and particle filtering methods,” Department of electrical engineering university of Chile, January 2017.

71. Battery Data Set NASA Ames Prognostic Data Repository

Appendix

MATLAB Codes SOC Estimation using PF

```
% particle filtering for SOC estimation
% using 2nd RC model battery
% SURESH
% M.S, THESIS,2017
%% extract the data and determine the parameters
close all;
clear all;

% Battery Specification
i=5; % Discharge Current Amp
Qc=5*3600; %Battery Capacity 5Ah
dt=1.08; %discrete time interval
time=1:3000; % length of discharge in sec

%Identified parameter from 2nd RC ECM model
R0=0.0717;
R1=0.0310;
R2=0.0277;
C1=8437;
C2=91401;

% initial parameters for the 2nd RC ECM Model
U1(1)=0.08387;
U2(1)=0.0189;
Voc(1)=4.175;
SOC(1)=0.99;
V_teri(1)=4.430;
V_meas(1)=4.3749;

L=3000;  % number of iterations
s=rng;
rng(s);
% Noise
W_k = 1e-6;     % Process Noise
V_k = 1e-4;     % Measurement Noise

for k=2:L

    % State space model
    % process equations (states)
    SOC(k)=SOC(k-1)-i*dt/Qc+ sqrt(W_k)*randn;                % first
State SOC, ****(IMP)if add noise the states are little discrete graphs
    U1(k)=(1-(dt/(R1*C1)))*U1(k-1)+dt/C1+sqrt(W_k)*randn;    % Second
state RC-terminal voltage
    U2(k)=(1-(dt/(R2*C2)))*U2(k-1)+dt/C2+sqrt(W_k)*randn;    % third
state, second RC-terminal voltage
    % Open circuit voltage
    Voc(k)= 14.79*(SOC(k)).^6-36.612*(SOC(k)).^5+29.235*(SOC(k)).^4-
6.281*(SOC(k)).^3....
           -1.647*(SOC(k)).^2+1.286*(SOC(k))+3.404;
    % Measurement Equation (Terminal Voltage)
       V_teri(k)=Voc(k)*SOC(k)+R0*i-U1(k)-U2(k);  % true Terminal
Voltage

       V_meas(k)=V_teri(k)+sqrt(V_k)*randn;        %Measured terinal
Voltage

end


figure
subplot(2,1,1)
plot(time,V_teri,'b');
hold on
plot(time,V_meas,'g')
grid, xlabel('Time [s]'), ylabel('Voltage [V]'), title('Simulated True
Discharge Voltage');
legend('True Voltage','Measured Voltage')
subplot(2,1,2)
plot(time,SOC);
grid, xlabel('Time [s]'), ylabel('SOC'), title('Simulated True SOC');
legend('True SOC')


%% %%
%---------------Particle Filtering---------------
n_part = 100;  % Number of Particles

% intial State particle
Zp = 0.08387;
Zn=0.0189;

%Noise filter
W_k = 1e-6;    % 8and 5
V_k = 1e-5;      %4
w_1 = W_k;
w_2 =V_k ;
Rnn = 1e-5;  %actual 2

%Resampling
N_t = n_part;

%Particle initialization
particle = zeros(n_part,3);  % it creats three zero coloms with 1000*3
matrics
particle_pred =  zeros(n_part,3); % it creats three zero coloms with
1000*3 matrics
particle(:,1) = ones(n_part,1)*Zp; % it creats three coloms one with
mutiple of Zp other two are zero 1000*3
particle(:,2) = ones(n_part,1)*Zn; % it creats three coloms second with
mutiple of Zp other two are zero 1000*3
```

```
particle(:,3) = unifrnd(0.80,0.90, n_part, 1); %soc it creats three
coloms thrid uniform no 0.8 to 0.99 with mutiple of Zp other two are
zero 1000*3
weight = ones(n_part,1)/n_part;   % it creats one colom with 1/1000 i.e
1000*1 matrcis

%Estimators
x1_est_mmse = zeros(L,1); %it creats zero of 1500 rows i.e L so 1500*1
fro state one
x1_est_mmse(1) = mean(particle(:,1)); % it can calculate mean of Zp for
all 1000*0.2/1000=0.2 L of length
x2_est_mmse = zeros(L,1); % it creats zero of 1500 rows for state two
i.e 1500*1 for state two
x2_est_mmse(1) = mean(particle(:,2)); %it creats mean of Zp for all
1000*0.2/1000 for L length
x3_est_mmse = zeros(L,1); %it creats zero of 1500 rows i.e L so 1500*1
for state three
x3_est_mmse(1) = mean(particle(:,3)); % it can calculate mean of 0.8 to
0.9 for all then we have 0.85 for  L of length

v_model = zeros(n_part,1); % it creats zero coloms with 1000*1 matrics

for k=2:L

    for t=1:n_part
        % Measured voltage pattern to the previous particle (k-1)
        v_model(t)=(Voc(k))*particle(t,3)+(R0*i)-particle(t,1)-
particle(t,2);

        %Importance sampling (prediccion from k-1 to k)
        r1 = sqrt(w_1)*randn;
        r2 = sqrt(w_2)*randn;
        particle_pred(t,1) = particle(t,1)+ r1;   % added to i*dt/C1
        particle_pred(t,2) = particle(t,2)+ r1;    %added to i*dt/C2
        particle_pred(t,3) = particle(t,3) - i*dt/Qc + r2;
%%v_model(t)*i*dt
         if particle_pred(t,3)<0
            particle_pred(t,3) = 0;
         end

         %Weight update (value measuremnt in k)
          v_model(t)=(Voc(k))*particle_pred(t,3)+(R0*i)-
particle_pred(t,1)-particle_pred(t,2);

         innov = V_meas(k) - v_model(t); % innovation
         weight(t) = exp( -log(sqrt(2*pi*Rnn)) -(( innov )^2)/(2*Rnn)
);
    end
    if sum(weight)==0
        %Display (Weight)
        disp(innov);
        disp('Error');
        disp(k);
        disp(t);
    end
    % Normalizes Weight
     weight = weight/sum(weight);
     N_eff = 1/( sum(weight.^2) );

     if N_eff < N_t
          %Resampling
        cdf = cumsum(weight);
         %Systematic resampling
         sam = rand/n_part;
         for t=1:n_part
             samInd = sam + (t-1)/n_part;
             ind = find( samInd<=cdf ,1);
            particle(t,:) = particle_pred(ind,:);

         end

     else
        for t=1:n_part
            particle(t,:) = particle_pred(t,:);
        end


     end
     x1_est_mmse(k) = mean(particle(:,1));
     x2_est_mmse(k) = mean(particle(:,2));
     x3_est_mmse(k) = mean(particle(:,3));

     error = SOC'- x3_est_mmse;
     rmse_pf(k)=sqrt(sum(((SOC'- x3_est_mmse).^2))/L);


end

mean_rmse_pf = mean(rmse_pf);
std_rmse_pf = std(rmse_pf);
mean_error=mean(error);

fprintf('mean rmse pf: %f \n',mean_rmse_pf);
fprintf('std rmse pf: %f \n',std_rmse_pf);
fprintf('% of Error: %f \n',mean_error);


figure, plot(time,x3_est_mmse,'r')
hold on
plot(time, SOC,'g')
grid, xlabel('Time [s]'), ylabel('SOC'), title('Particle-Filtered SOC
');
legend('PF Estimate','Ground Truth');

figure
subplot(2,1,1)
plot(time,SOC);
hold on
```

```
plot(time,x3_est_mmse,'g');
grid, xlabel('Time [s]'), ylabel('State'), title('SOC Estimation');
legend('True State','BPF Estimate');


subplot(2,1,2)
error = SOC'- x3_est_mmse;
plot(time,error);
ylim([-4e-2,4e-2]);
grid, xlabel('Time [s]'), ylabel('Error');
title('SOC Estimation Error');
```

SOC Estimation with EKF method

```
close all;
clear all;

%Extended Kalman filtering for SOC estimation
% Using a 2nd RC ECM model
% Suresh
% M.S, Thesis 2017
%% extract the data and determine the parameters

% Battery Specification
i=5; % Discharge Current Amp
Qc=5*3600; %Battery Capacity 5Ah
dt=1.08; %discrete time interval
time=1:3000; % length of discharge in sec

%Identified parameter from 2nd RC ECM model
R0=0.0717;
R1=0.0310;
R2=0.0277;
C1=8437;
C2=91401;

% initial parameters for the 2nd RC ECM Model
U1(1)=0.08387;
U2(1)=0.0189;
Voc(1)=4.175;
SOC(1)=0.99;
V_teri(1)=4.430;
V_meas(1)=4.3749;

L=3000;  % number of iterations
s=rng;
rng(s);

% Noise
W_k = 1e-6;     % Process Noise
V_k = 1e-4;     % Measurement Noise
```

```
for k=2:L

    % State space model
    % process equations (states )
    SOC(k)=SOC(k-1)-i*dt/Qc+ sqrt(W_k)*randn;                % first
State SOC, ****(IMP)if add noise the states are little discrete graphs
    U1(k)=(1-(dt/(R1*C1)))*U1(k-1)+i*dt/C1+sqrt(W_k)*randn;    % Second
state RC-terminal voltage
    U2(k)=(1-(dt/(R2*C2)))*U2(k-1)+i*dt/C2+sqrt(W_k)*randn;    % third
state, second RC-terminal voltage
     %Open circuit voltage
    Voc(k)= 14.79*(SOC(k)).^6-36.612*(SOC(k)).^5+29.235*(SOC(k)).^4-
6.281*(SOC(k)).^3....
           -1.647*(SOC(k)).^2+1.286*(SOC(k))+3.404;
    % Measurement Equation (Terminal Voltage)
       V_teri(k)=Voc(k)*SOC(k)+R0*i-U1(k)-U2(k);  % true Terminal
Voltage

      V_meas(k)=V_teri(k)+sqrt(V_k)*randn;        %Measured terinal
Voltage

end



%figure
%subplot(2,1,1)
%plot(time,V_teri,'b');
%hold on
%plot(time,V_meas,'g')
%grid, xlabel('Time [s]'), ylabel('Voltage [V]'), title('Simulated True
Discharge Voltage');
%legend('True Voltage','Measured Voltage')
%subplot(2,1,2)
%plot(time,SOC);
%grid, xlabel('Time [s]'), ylabel('SOC'), title('Simulated True SOC');
%legend('True SOC')


%% EKF based ground vehicle navigation

F=[1-i*dt/Qc 0 0; 0 (1-(dt/(R1*C1)))+i*dt/C1 0; 0 0 (1-
(dt/(R2*C2)))+i*dt/C2];
%E=[i*dt/Qc; (i*dt)/C1; (i*dt)/C2]
%D=F+E; %state space model
Q=diag([1e-5 1e-6 1e-6]);  %process noise covariance
R=1e-4;  % measure noise covariance

x=[0.97; 0.08387; 0.0189 ]; % intial state
xhatplus=x; %intial state estimate
Pplus=diag([0 0 0]); % intial estimation error covariance

% intialize arrays
xArr=x;
xhatArr=xhatplus;
```

```
for k=2:L

    % system simulation
    x=F*x+ sqrt(Q)*randn(3,1);
    y=Voc(k-1)*x(1)+R0*i-x(2)-x(3)+sqrt(R)*randn; %v(k)

    % EKF time update
    Pminus=F*Pplus*F'+Q;
    xhatminus=F*xhatplus;

    %EKF measurment update
    H=zeros(1,3);
    SOChat=xhatminus(1);
    U1hat=xhatminus(2);
    U2hat=xhatminus(3);
    temp=Voc(1)*SOChat+R0*i-U1hat-U2hat;
    H(1,1)=88.74*(SOChat).^5-183.06*(SOChat).^4+116.94*(SOChat).^3-
18.843*(SOChat).^2....
           -3.294*(SOChat)+1.286;   %i/Qc;
    H(1,2)=-U1hat; %(i/C1)-(U1(k)/(R1*C1));
    H(1,3)=-U2hat; %(i/C2)-(U2(k)/(R2*C2));

    K=Pminus*H'*inv(H*Pminus*H'+R);
    yhat=Voc(k-1)*SOChat+R0*i-U1hat-U2hat; %v(k)
    xhatplus=xhatminus+K*(y-yhat);
    Pplus= Pminus-K*H*Pminus; %(eye(3)-K*H)*Pminus or (1-K*H)*Pminus;

    xArr=[xArr x];
    xhatArr=[xhatArr xhatplus];

end

% figures
figure
subplot(2,1,1)
plot(time,xArr(1,:),'r')
hold on
plot(time, SOC,'g')
grid, xlabel('Time [s]'), ylabel('SOC'), title('EKF SOC ');
legend('EKF Estimate','Ground Truth');

subplot(2,1,2)
error=SOC-xArr(1,:);
plot(time, error)
ylim([-2.5e-1,2.5e-1]);
grid, xlabel('Time [s]'), ylabel('Error');
title('SOC Estimation Error');

% Compute experimental Standard Deviation of Estimation Error
Eststd=std(xArr(1,:)-xhatArr(1,:));
fprintf('Std of EKF: %f \n',Eststd);
```

MATLAB Code for SOC Comparison Results for PF, EKF, and KF

```
close all;
clear all;

% load the data
load('matlab_originalsoc.mat');
load('matlab_PFsoc.mat');
load('matlab_EKFxhat.mat');
load('matlab_KFxArr.mat', 'xArr');
%load('matlab_Xhatplus.mat');


t=1:3000;

figure
plot(t,SOC,'g');
grid on;
hold on;
plot(t,x3_est_mmse,'r');
plot(t,xhatArr(1,:),'k')
plot(t, xArr(1,:),'b')
xlabel('Time [s]'), ylabel('SOC'), title('Comparision of SOC ');
legend('Ground Truth','PF Estimate', 'EKF Estimate','KF Estimate');

error_1=(SOC-x3_est_mmse');
error_2=(SOC-xhatArr(1,:));
error_3=(SOC-xArr(1,:));

figure
plot(t,error_1,'r');
grid on;
hold on;
plot(t,error_2,'k');
plot(t,error_3,'b');
xlabel('Time [s]'), ylabel('SOC Error'), title('Comparision of SOC
Error');
legend('PF Estimate Error', 'EKF Estimate Error','KF Estimate Error');

for t=1:3000

   rmse_pf(t)=sqrt(sum(((SOC-x3_est_mmse').^2))/t);
   rmse_ekf(t)=sqrt(sum(((SOC- xhatArr(1,:)).^2))/t);
   rmse_kf(t)=sqrt(sum(((SOC- xArr(1,:)).^2))/t);

   mean_rmse_pf = mean(rmse_pf);
   std_rmse_pf = std(rmse_pf);

   fprintf('rmse mean of PF: %f \n',mean_rmse_pf);
   fprintf('rmse std of PF: %f \n',std_rmse_pf);

   mean_rmse_ekf = mean(rmse_ekf);
   std_rmse_ekf = std(rmse_ekf);
   fprintf('rmse mean of ekfF: %f \n',mean_rmse_ekf);
   fprintf('rmse std of ekfF: %f \n',std_rmse_ekf);

   mean_rmse_kf = mean(rmse_kf);
   std_rmse_kf = std(rmse_kf);
   fprintf(' rmse mean of kF: %f \n',mean_rmse_kf);
   fprintf('rmse std of kF: %f \n',std_rmse_kf);
end

PFstd=std(error_1);
EKFstd=std(error_2);
KFstd=std(error_3);

fprintf('std of PF: %f \n',PFstd);
fprintf('std of EKF: %f \n',EKFstd);
fprintf('std of KF: %f \n',KFstd);
```

MATLAB code: Remaining Useful Life Estimation (RUL) using PF framework

```
clear all
close all
% RUL Estimation using NASA data Prognostic center
% Suresh Daravath
% M.S. thesis SMC

%% PF model
%Load the data set.
load(['C:\Users\Suresh\Desktop\RUL\nasa prognastic model source
code\NASA progostic center data\BatteryAgingARC-FY08Q4\B0005.mat']);
load(['C:\Users\Suresh\Desktop\RUL\nasa prognastic model source
code\NASA progostic center data\BatteryAgingARC-FY08Q4\B0007.mat']);

theta=[-9.86e-7,5.752e-2,8.983e-1,-8.34e-4]';
first_batt = (-9.86e-7) * exp(5.752e-2 * (1:200)) + (8.983e-1) ...
* exp((-8.340e-4) * (1:200)) + 0.005*randn(1,200);
second_batt = (-9.86e-7) * exp(5.752e-2 * (1:200)) + (8.983e-1) ...
* exp((-8.340e-4) * (1:200)) + 0.005*randn(1,200);

%PF
theta_set=repmat(theta,1,100);
theta_set(1,1:100) = theta(1) + theta(1)/10 * (0.5-rand(100,1));
theta_set(2,1:100) = theta(2) + theta(2)/10 * (0.5-rand(100,1));
theta_set(3,1:100) = theta(3) + theta(3)/10 * (0.5-rand(100,1));
theta_set(4,1:100) = theta(4) + theta(4)/10 * (0.5-rand(100,1));
weights = 0.01 * ones(1,100);
tic
for j = 1:100
choose_par(j,:) = theta_set(1,j) * exp(theta_set(2,j) * ...
(1:250)) + theta_set(3,j) * exp(theta_set(4,j)*(1:250));
RULs(j) = find(choose_par(j,:) <= 0.8*(second_batt(1)),1);
end
toc

tic
```

```
sigma = 0.1;
for i = 1:200
if i ==50
weights_50 = weights;
end
if i == 100
weights_100 = weights;
end
if i == 150
weights_150 = weights;
end
% Get the likelihood
likelihood = 1/(sigma*sqrt(2*pi)) * exp(-1/2 * ...
((second_batt(i)) - (theta_set(1,:) .* exp(theta_set(2,:)...
* i) + theta_set(3,:) .* exp(theta_set(4,:) * i))).^2 /...
sigma^2);

% Update the weights
weights = weights .* likelihood;
weights = weights / sum(weights);
end
toc
[RULs, ind] = sort(RULs);
weights_50s = weights_50(ind);
weights_100s = weights_100(ind);
weights_150s = weights_150(ind);
figure
xlabel('k, Cycle index (cycle)')
ylabel('Capacity (Ah)')
axis square
hold on
grid on
plot(RULs', weights_150s + 0.8*second_batt(1),'g', 'linewidth', 2)
%plot(RULs', weights_100s + 0.8*second_batt(1),'r', 'linewidth', 2)
%plot(RULs', weights_50s + 0.8*second_batt(1),'r', 'linewidth', 2)

plot(1:length(second_batt), second_batt,'r','linewidth',1.5) %%
plot(1:length(second_batt), second_batt,'ko','linewidth',1.5) %% si

Life_RUL=mean(RULs')
s_deviation=std(RULs')

EOL=RULs';
norm=normpdf(RULs',Life_RUL,s_deviation);
plot(EOL,norm)
legend(' RUL at k=150', 'observations','PF prediction')
plot([1,200],second_batt(1)*0.8*[1,1],'b','linewidth',1.5)
text(25,second_batt(1)*0.81,'RUL failure threshold')
axis([0 200 0.65, 0.91])
line([150 150],[0.65 0.91]);
set(gca,'YLim',[0.65 0.91])
%title('PF tracking four states, five percent particle variation')
xlabel('k, Cycle index (cycle)')
ylabel('Q, Capacity (Ah)')
axis square
box on
```

```
err_early = sum(weights_50s.*RULs)-190
err_late = sum(weights_100s.*RULs)-190
err_final = sum(weights_150s.*RULs)-190
sig_early = sqrt(sum(weights_50s.*(RULs - (err_early + 190)).^2) )
sig_late = sqrt(sum(weights_100s.*(RULs - (err_late + 190)).^2) )
sig_final = sqrt(sum(weights_150s.*(RULs - (err_final + 190)).^2) )
```