



Kaunas University of Technology
Faculty of Mathematics and Natural Sciences

Machine Learning Models Research for Lithium-ion Battery Ageing Monitoring and Reliability Assessment

Master's Final Degree Project

Rasa Grilauskaitė
Project author

Prof. Dr. Robertas Alzbutas
Supervisor

Kaunas, 2024



Kaunas University of Technology
Faculty of Mathematics and Natural Sciences

Machine Learning Models Research for Lithium-ion Battery Ageing Monitoring and Reliability Assessment

Master's Final Degree Project
Applied Mathematics (6211AX006)

Rasa Grilauskaitė

Project author

Prof. Dr. Robertas Alzbutas

Supervisor

Assoc. Prof. Dr. Paulius Palevičius

Reviewer

Kaunas, 2024



Kaunas University of Technology

Faculty of Mathematics and Natural Sciences

Rasa Grilauskaitė

Machine Learning Models Research for Lithium-ion Battery Ageing Monitoring and Reliability Assessment

Declaration of Academic Integrity

I confirm the following:

1. I have prepared the final degree project independently and honestly without any violations of the copyrights or other rights of others, following the provisions of the Law on Copyrights and Related Rights of the Republic of Lithuania, the Regulations on the Management and Transfer of Intellectual Property of Kaunas University of Technology (hereinafter – University) and the ethical requirements stipulated by the Code of Academic Ethics of the University;
2. All the data and research results provided in the final degree project are correct and obtained legally; none of the parts of this project are plagiarised from any printed or electronic sources; all the quotations and references provided in the text of the final degree project are indicated in the list of references;
3. I have not paid anyone any monetary funds for the final degree project or the parts thereof unless required by the law;
4. I understand that in the case of any discovery of the fact of dishonesty or violation of any rights of others, the academic penalties will be imposed on me under the procedure applied at the University; I will be expelled from the University and my final degree project can be submitted to the Office of the Ombudsperson for Academic Ethics and Procedures in the examination of a possible violation of academic ethics.

Rasa Grilauskaitė

Confirmed electronically

Grilauskaitė, Rasa. Machine Learning Models Research for Lithium-ion Battery Ageing Monitoring and Reliability Assessment. Master's Final Degree Project/supervisor Prof. Dr. Robertas Alzbutas; Faculty of Mathematics and Natural Sciences, Kaunas University of Technology.

Study field and area (study field group): Mathematics, Applied mathematics.

Keywords: lithium-ion batteries, Machine Learning, ageing models, reliability assessment methods, State of Charge, State of Health, Neural Network, Regression models, Kalman Filter, data preprocessing.

Kaunas, 2024. 73 p.

Summary

In the energy sector, sustainability encompasses a wide range of aspects. One of them could be the responsible use of resources. Lithium-ion batteries are particularly suitable for achieving responsible use of energy resources. As these batteries are rechargeable and long-lasting, they can be reused. This not only helps reduce the costs for a company or organisation but also protects the environment. Despite their positive properties, lithium-ion batteries age, are not resistant to all conditions and can fail. To identify faults and ageing, it is recommended to perform an ageing simulation and reliability assessment. This study analyses the effectiveness of machine learning models in assessing the internal states of lithium-ion batteries: state of charge and state of health.

A review of the scientific literature revealed that different machine-learning models are combined for the ageing simulation and reliability assessment of lithium-ion batteries. Data is often pre-processed before being fed into the machine learning models.

For this study, synthetic lithium-ion battery charging data with 10,000 data points and five variables: current, voltage, temperature, capacity, and testing time in seconds were generated. Additionally, real lithium iron phosphate battery discharge and charging data were analysed. Three artificial neural networks, one regression model, and one recursive filter were created. These models were refined, and a total of nine models were tested with synthetic and real battery data during the study. A total of 202 experiments were conducted. Of these, 181 experiments were conducted with synthetic data, and 21 experiments were conducted with real data. The results were compared with the actual values of the State of Charge and State of Health.

The suitability of the models was verified with real data. It was observed that neural networks are more suitable than regression models. Based on the study results, an additional application system was created, allowing the user to independently generate synthetic lithium-ion battery charging data, apply a Kalman Filter, and calculate the state of charge of the batteries without programming. The application system also allows the user to upload their data.

Grilauskaitė, Rasa. Mašininio mokymosi modelių tyrimas ličio jonų baterijos senėjimo monitoringui ir patikimumui vertinti. Magistro baigiamasis projektas/vadovas Prof. Dr. Robertas Alzbutas; Kauno technologijos universitetas, Matematikos ir gamtos mokslų fakultetas.

Studijų kryptis ir sritis (studijų kryptių grupė): Matematikos mokslai, taikomoji matematika.

Reikšminiai žodžiai: ličio jonų baterijos, mašininis mokymasis, senėjimo modeliai, patikimumo vertinimo metodai, įkrovos būseną, sveikatos būseną, neuroninis tinklas, regresijos modeliai, Kalmano filtras, išankstinis duomenų apdorojimas.

Kaunas, 2024. 73 p.

Santrauka

Energetikos sektoriaus atveju tvarumas apima labai platų spektrą dalykų. Vienas jų galėtų būti atsakingas išteklių naudojimas. Ličio jonų baterijos yra itin tinkamos siekiant atsakingai naudoti energetinius išteklius. Kadangi šios baterijos yra pakraunamos ir pasižymi ilgaamžiškumu, jas galima naudoti pakartotinai. Tai ne tik padeda mažinti įmonės ar organizacijos išlaidas, bet ir saugo gamtą. Nepaisant teigiamų savybių, ličio jonų baterijos sensta, nėra atsparios visoms sąlygoms ir gali sugesti. Siekiant laiku identifikuoti gedimus ir senėjimą, rekomenduojama atlikti senėjimo modeliavimą ir patikimumo vertinimą. Šiame tyrime analizuojamas mašininio mokymosi modelių efektyvumas atliekant ličio jonų baterijų vidinių būsenų: įkrovimo būsenos ir sveikatos būklės vertinimą.

Atlikus mokslinės literatūros analizę, pastebėta, kad ličio jonų baterijų senėjimo modeliavimui ir patikimumo vertinimui yra derinami skirtingi mašininio mokymosi modeliai. Duomenys dažnai iš anksto apdorojami prieš pateikiant juos kaip įvestį į mašininio mokymosi modelius.

Šiam tyrimui sugeneruoti sintetiniai ličio jonų baterijų įkrovimo duomenys su 10 000 tūkstančių duomenų taškų ir penkiais kintamaisiais: srove, įtampa, temperatūra, talpa ir bandymo laiku sekundėmis. Taip pat buvo išanalizuoti tikri ličio geležies fosfato baterijos iškrovimo ir įkrovimo duomenys. Sukurti trys dirbtiniai neuroniniai tinklai, vienas regresijos modelis ir vienas rekursinis filtras. Šie modeliai buvo tobulinami, todėl tyrimo metu iš viso buvo išbandyti devyni modeliai su sintetiniais ir realiais baterijų duomenimis. Iš viso atlikti 202 eksperimentai. Iš jų 181 eksperimentas atliktas su sintetiniais duomenimis, o 21 eksperimentas atliktas su realiais duomenimis. Rezultatai buvo palyginti su faktinėmis įkrovimo būsenos ir sveikatos būklės vertėmis.

Modelių tinkamumas buvo tikrinamas su realiais duomenimis. Pastebėta, kad neuroniniai tinklai yra tinkamesni nei regresijos modeliai. Atsižvelgiant į tyrimo rezultatus, papildomai sukurta taikymų sistema, leidžianti vartotojui be programavimo savarankiškai generuoti sintetinius ličio jonų baterijų įkrovimo duomenis, pritaikyti Kalmano filtrą ir apskaičiuoti baterijų įkrovimo būseną. Taip pat taikymo sistemoje yra galimybė vartotojui įkelti turimus duomenis.

Content

List of Figures	8
List of abbreviations.....	11
Introduction	12
1. Scientific literature review.....	13
1.1. Lithium-ion batteries and their safety issues	13
1.2. Characterization of the ageing process and reliability.....	17
1.3. Lithium-ion batteries ageing monitoring and reliability assessment.....	20
1.3.1. Most commonly applied ageing models and reliability assessment methods	20
1.3.2. Assessment of various internal states of lithium-ion batteries	23
1.3.3. Machine-learning and deep-learning applications.....	24
2. Research methods and models	28
2.1. Research workflow and synthetic data generation	28
2.2. Mathematical models of states of lithium-ion batteries.....	30
2.3. Signal processing methods for data preprocessing	31
2.4. Machine Learning models and their evaluation metrics	32
2.4.1. Artificial neural network models architectures	32
2.4.2. Regression models and Kalman filter.....	36
2.4.3. Metrics for models performance.....	38
2.4.4. Software used for modelling	39
3. Research results.....	40
3.1. Data overview and preprocessing results.....	40
3.1.1. Synthetic data case.....	40
3.1.2. Real data case	44
3.2. Model training and testing results.....	48
3.2.1. Synthetic data State of Charge assessment results	48
3.2.2. Real data State of Charge assessment results	54
3.2.3. Real data State of Health assessment	59
3.3. Application system	60
Conclusions	64
Scientific literature sources.....	66
Appendixes	74
1 appendix. Changes in the number of publications	74
2 appendix. Literature sources distribution by year.....	75
3 appendix. GRU results for synthetic data SoC	76
4 appendix. LSTM results for synthetic data SoC	77

5 appendix. Publication and presentations.....	78
6 appendix. Python code for synthetic data generation	79
7 appendix. Python code for data processing (example with real data)	80
8 appendix. Python code for training and testing a RNN with synthetic data	81
9 appendix. Python code for Linear Regression model (example with real data)	83
10 appendix. Python code for Ridge Regression model (example with real data)	84
11 appendix. Python code for a Kalman filter (example with synthetic data).....	85

List of Figures

Fig. 1. LIB development progress (modified according to [11])	14
Fig. 2. Examples of (a) LIBs parallel configuration and (b) LIBs series configuration [15].....	15
Fig. 3. Applicable standards for LIBs safety (prepared according to [19]).....	16
Fig. 4. A standard BMS diagram [20]	16
Fig. 5. Main mechanisms of LIBs anode and cathode ageing (prepared according to [30])	18
Fig. 6. Reliability assessment procedure for LIBs in electric vehicles [32].....	19
Fig. 7. Development of ageing models for LIBs	20
Fig. 8. Classification of ageing models and reliability assessment methods, and their examples	21
Fig. 9. LIBs SoC estimation methods (prepared according to [74])	23
Fig. 10. Lithium-ion batteries SOH estimation methods (prepared according to [75])	24
Fig. 11. Example flowchart of ML applications for LIB SoH estimation [76]).....	24
Fig. 12. General research workflow structure	28
Fig. 13. Single Layer Denoising Autoencoder graphical illustration [97]	32
Fig. 14. Basic RNN architecture [98].....	32
Fig. 15. SRNN architecture with input layer (1), first SRNN layer (2), second SRNN layer (3) and dense layer (4) for initial model (a), LSTM architecture with input layer (1), first LSTM layer (2), second SRNN layer (3) and dense layer (4) for initial model (b), GRU architecture with input layer (1), first GRU layer (2), second GRU layer (3) and dense layer (4) for initial model (c).....	33
Fig. 16. ReLu function visualisation [101].....	35
Fig. 17. Standard process of KF [102]	37
Fig. 18. Simplified process of KF	37
Fig. 19. Charge data of synthetic lithium-ion batteries in terms of current (a) and voltage (b).....	40
Fig. 20. Calculated SoC against time (a) and voltage against SoC (b)	40
Fig. 21. Pearson correlation between target variable SoC and features	41
Fig. 22. Real current against the preprocessed current.....	42
Fig. 23. Real voltage against preprocessed current	43
Fig. 24. Real battery data current (a) and voltage (b).....	44
Fig. 25. Calculated real battery SoC over time	45
Fig. 26. Real current against the preprocessed current.....	46
Fig. 27. Real voltage against preprocessed current	46
Fig. 28. Pearson correlation between target variable SoC and features	47
Fig. 29. RNN model training with synthetic data MSE values per epoch, best results (a) and worst results (b).....	49
Fig. 30. RNN model with Bayesian optimisation training with synthetic data MSE values per epoch, best results (a) and worst results (b)	50
Fig. 31. Comparison of best-performing models by training time in seconds and RMSE error, synthetic dataset case.....	51
Fig. 32. Real SoC value and assessed values applying Ridge regression and neural NN, synthetic dataset case	53
Fig. 33. KF results in comparison with real SoC values, synthetic dataset case.....	53
Fig. 34. Neural network models training with synthetic data MSE values per epoch, best results (a) and worst results (b), variables used: voltage, current, time difference in hours, charge Ah and delta Q	54
Fig. 35. Neural network models training with synthetic data MSE values per epoch, best results (a) and worst results (b), variables used: voltage and current.....	55
Fig. 36. Comparison of best and worst performing models by training time in seconds and RMSE error, real battery dataset case	56

Fig. 37. Neural network models training with real dataset worst performing case.....	56
Fig. 38. Neural network models training with real dataset best performing case	57
Fig. 39. ML models training with real battery dataset worst performing case.....	58
Fig. 40. ML models training with real battery dataset best performing case	58
Fig. 41 . KF results in a comparison with real SoC values, real battery dataset case	59
Fig. 42. Obtained SoH using real battery data	59
Fig. 43. SoH estimated by applying the empirical degradation model	60
Fig. 44. Default values for synthetic dataset generation	61
Fig. 45. Application system menu functions.....	61
Fig. 46. Pop-up informational notifications after performing certain actions.....	62
Fig. 47. Application system for the KF	62
Fig. 48. Application system for SoC assessment	63

List of Tables

Table 1. Pre-market safety tests for LIBs (prepared according to [18])	15
Table 2. Advantages and disadvantages of different ageing models	22
Table 3. Reliability assessment methods advantages and disadvantages	22
Table 4. Applications of ML and DL for state estimation of LIBs.....	25
Table 5. LIB data preprocessing techniques	26
Table 6. Applications of filters for direct evaluation of internal states of batteries.....	27
Table 7. Values of parameters used for synthetic dataset generation	29
Table 8. Number of parameters in each neural network	33
Table 9. Number of units and learning rate for initial models after applying Bayesian optimisation	34
Table 10. Models split ratio and performance metrics	35
Table 11. Applied Python libraries according to groups of tasks to be solved.....	39
Table 12. Statistical characteristics and their values	41
Table 13. Variance Inflation Factor (VIF) for synthetic data results.....	43
Table 14. Research battery description.....	44
Table 15. Variance Inflation Factor (VIF) for real battery data results	47
Table 16. Initial RNN model testing results with different numbers of epochs and model input data, synthetic dataset case.....	48
Table 17. RNN model with Bayesian optimisation testing results with different number of epochs and model input data, synthetic dataset case	49
Table 18. Ridge regression optimised alpha values for different variables, synthetic dataset case .	52
Table 19. ML models testing results with different model input data, synthetic dataset case	52
Table 20. Neural network models testing with variables voltage, current, charge Ah, time difference in hours and delta Q results, real battery dataset case, 100 epochs	54
Table 21. Neural network models testing with only variables voltage and current results, real battery dataset case, 100 epochs	55
Table 22. Ridge regression optimised α values for different input variables, real dataset case.....	57
Table 23. ML models testing results with different model input data, real battery dataset case.....	58

List of abbreviations

LIB(s) - Lithium-Ion Battery(ies)
BMS - Battery Management System
ML - Machine Learning
DL - Deep Learning
SoC - State of Charge
SoH - State of Health
WT - Wavelet Transform
DWT - Discrete Wavelet Transform
KF - Kalman Filter
SLDA - Single-layer Denoising Autoencoder
SRNN - Simple Recurrent Neural Network
LSTM - Long Short-Term Memory
GRU - Gated Recurrent Unit
LR - Linear Regression
SGF - Savitzky-Golay Filter
MSE - Mean Squared Error
RMSE - Root Mean Squared Error
SEI - Solid Electrolyte Interphase
ReLu - Rectified Linear Unit

Introduction

Lithium-ion batteries are known for their high energy retention capability, as evidenced by the fact that a mere kilogram of lithium can retain up to 3860 Ah [1]. Lithium-ion batteries have witnessed remarkable advancements over the decades. These advancements include the introduction of organic electrodes and the development of various anodes during the period from 1980 to 1990, as well as the development of advanced battery types such as lithium-air and all-solid-state batteries by 2025 [2]. Lithium-ion batteries are widely used in devices for individual consumers; moreover, they are prominently utilised in electric vehicles and the energy sector in general. For each application of lithium-ion batteries, safety requirements are set. In the case of electric vehicles, it has been established that batteries must maintain their performance capabilities under extreme temperature conditions, specifically between -40°C and 60°C [3]. In addition to this requirement, most manufacturers commit to replacing the batteries if their capacity falls below the forecasted threshold [4].

Over time, lithium-ion batteries age; thus, failures may occur that pose a threat to user safety. To prevent accidents, it is necessary to continuously monitor ageing and perform reliability assessments. There has been a noticeable steady increase in the number of publications related to both ageing monitoring and research individually on reliability assessment during the period from 2003 to 2023 inclusive (see Appendix 1). Machine learning and deep learning offer advanced solutions for monitoring ageing and assessing the reliability of lithium-ion batteries. Incorporating these algorithms enables precise calculations by considering factors such as usage and temperature changes, significantly enhancing battery life and efficiency across various applications.

The main objective of this research is to thoroughly investigate the efficiency of Machine Learning and Deep Learning models for monitoring ageing and reliability assessment of lithium-ion batteries. To achieve this main objective, the following tasks are distinguished:

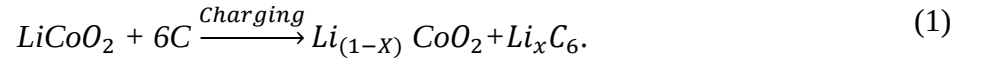
1. Analyse scientific literature related to the ageing of lithium-ion batteries, including their state monitoring and the assessment of reliability metrics;
2. Generate synthetic lithium-ion battery data and collect, analyse and process real lithium-ion battery data;
3. Develop, implement, and evaluate different models based on Machine Learning and Deep Learning methods with synthetic data and compare the results of the best models with the empirical methods used in reliability theory;
4. Confirm the practical applicability of the developed models with real lithium-ion battery data;
5. Create a trial version of the system where users could prepare datasets for training and testing Machine Learning models.

This research, funded by the Research Council of Lithuania, is structured into three main parts. The first part presents an analysis of scientific literature, where the sources analysed range from 1999 to 2024, inclusive (see Appendix 2). The third part introduces the research results. A list of presentations at scientific conferences related to this research is presented in Appendix 5.

1. Scientific literature review

1.1. Lithium-ion batteries and their safety issues

Lithium-ion batteries (further denoted as LIB(s)) are rechargeable batteries. These batteries are composed of five essential parts: primary electrodes, anodes, an insulating barrier layer, connection points, and an enclosing structure [5, 6, 7]. LIBs operate on a principle that involves the movement of lithium ions between the cathode and anode during charging and discharging cycles. During the charging process, lithium ions extraction from the cathode composed of lithium cobalt oxide $LiCoO_2$ is being performed. Furthermore, lithium ions are moved to the anode, made of graphite C_6 . This process is represented by the chemical equation [8]:



Conversely, during discharge, the lithium ions travel back from the graphite anode to the lithium cobalt oxide cathode, allowing the battery to release stored energy and power devices. This reversible process is key to the functionality and rechargeability of LIBs.

To extend the usage duration of LIBs between charges, it is essential to maximize the cell voltage without surpassing the electrolyte stability limits. This process involves calculating the equilibrium voltage for lithium intercalation, a process expressed by a formula [9]:

$$V = -\frac{\mu_{Li}^{cathode} - \mu_{Li}^{anode}}{zF}. \quad (2)$$

In Formula 2, V represents the cell voltage, z represents the number of electrons involved in the reaction, while F is a measure of electric charge per mole of electrons known as the Faraday constant, $\mu_{Li}^{cathode}$ and μ_{Li}^{anode} are the chemical potentials of lithium.

The creation of LIBs is the result of the collective work of many scientists over several decades. During the early phases of LIB research, the main focus was on solid-state physics, but at the beginning of the 20th century, the physical properties of electrode materials such as surface treatment and structural details began to be investigated [10]. The entire development process, marked by key milestones and shifts, is depicted in the image below.

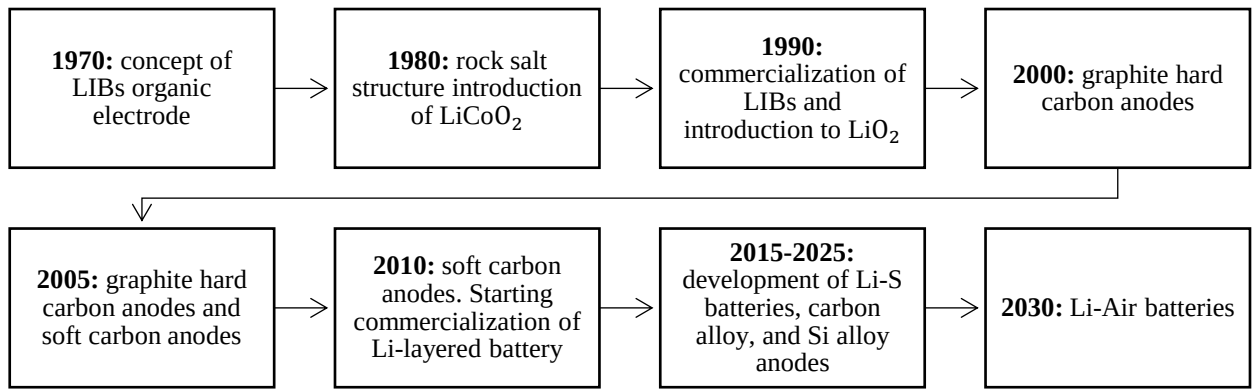


Fig. 1. LIB development progress (modified according to [11])

According to Figure 1, the development of the LIB concept began in the 1970s, emphasizing the use of organic electrodes. The rock salt structure LiCoO₂ was introduced in the 1980s. Battery commercialization started in the 1990s, along with the introduction of LiO₂. The beginning of the new millennium in the 2000s showed progress towards solid graphite carbon anodes, and a later trend in 2005 combined both solid and soft carbon anodes. In the following decade, starting in 2010, preference was given to soft carbon anodes, marking the beginning of commercialization for Li-layered batteries. From 2015 to 2025, progress was highlighted in the development of Li-S batteries, carbon alloy anodes, and Si alloy anodes. Looking further into the future, 2030 promises another innovative evolution with the emergence of Li-Air batteries.

LIB cells are commonly distinguished by their physical design and appearance, falling into two main types: **cylindrical cells**, which are enveloped in hard casings, and **prismatic cells**, which vary, ranging from solid box-like shapes to pliable, pouch-type units [12]. It would be difficult to determine which type of commercially available LIB elements are the best, as the choice depends on various factors. Despite that, all elements can be compared based on the arrangement of electrodes, mechanical stability, heat management, specific energy, and energy density. For example, the cylindrical cell boasts the best mechanical stability due to its wound electrode arrangement but suffers from poor heat management, whereas the prismatic cell offers a good balance with better heat management and energy density, and the pouch cell, despite its poor mechanical stability, provides the best specific energy and a compact, flexible shape, making it suitable for varied applications where space and energy requirements are critical [13].

Furthermore, the elements of LIBs are categorized by connection into single-cell configuration, series configuration, parallel configuration, and series-parallel configuration depending on the required capacity [14]. Examples of parallel and series configuration schemes are indicated in Figure 2, where in a parallel configuration, two cells are connected with both their positive and negative terminals linked to each other, while in a series configuration, two cells are connected in a positive-to-negative sequence.

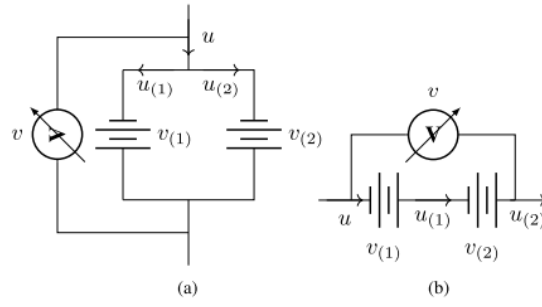


Fig. 2. Examples of (a) LIBs parallel configuration and (b) LIBs series configuration [15]

Overall, within a single-cell configuration, a LIB employs only one cell, delivering its inherent voltage and capacity. On the other hand, the series-parallel configuration combines the characteristics of both series and parallel configurations.

The practical applications of LIBs are broadly divided into two main classes: stationary and portable uses [16]. Consumer electronics and electric vehicles are considered portable uses, while the electricity grid and renewable energy are considered stationary uses. Examining the examples, consumer electronics include mobile phones and laptops; electric vehicles encompass cars and electric bicycles; renewable energy consists of wind turbines and solar power plants; and the electricity grid is made up of power stations [17]. Portable uses are more suited to individual consumers and applications in a domestic environment, which is why LIBs are smaller. Since stationary uses are designed for broader consumption in the manufacturing sector, battery cells are employed.

Before LIBs reach the market, they undergo strict pre-market safety evaluations. These assessments are crucial for ensuring that the batteries operate safely under various conditions and scenarios. The following table (see Table 1) outlines the core attributes and primary evaluations conducted during these safety tests. Each category, ranging from electrochemical resilience to environmental adaptability, is accurately tested to ensure that the batteries meet the highest standards of safety before they are considered suitable for individual use.

Table 1. Pre-market safety tests for LIBs (prepared according to [18])

Core attributes	Primary evaluations
Electrochemical resilience	Tests overcharging, deep discharging, temperature extremes, short circuits, and enforced discharging
Mechanical durability	Assessments include dropping, severe impacts, penetrations tests, vibration, compression, acceleration, and crushing
Thermal stability	Evaluations of response to heating, thermal shocks, and ignition sources
Environmental adaptability	Examinations under low pressure, high altitudes, and water immersion scenarios

As shown in Table 1, the pre-market tests include evaluating the ability to withstand overcharging, deep discharging, exposure to extreme temperatures, several mechanical stress tests, response to temperature changes, as well as capability to adapt to different environments.

Also, there are international standards established by organizations such as the International Organization for Standardization (ISO), the International Electrotechnical Commission (IEC), the Society of Automotive Engineers (SAE), Underwriters Laboratories (UL), and Guobiao (which

stands for Chinese National Standards or GB) for ensuring compliance with LIB safety. Figure 3 outlines the applicable each international organisation standards for LIBs safety.

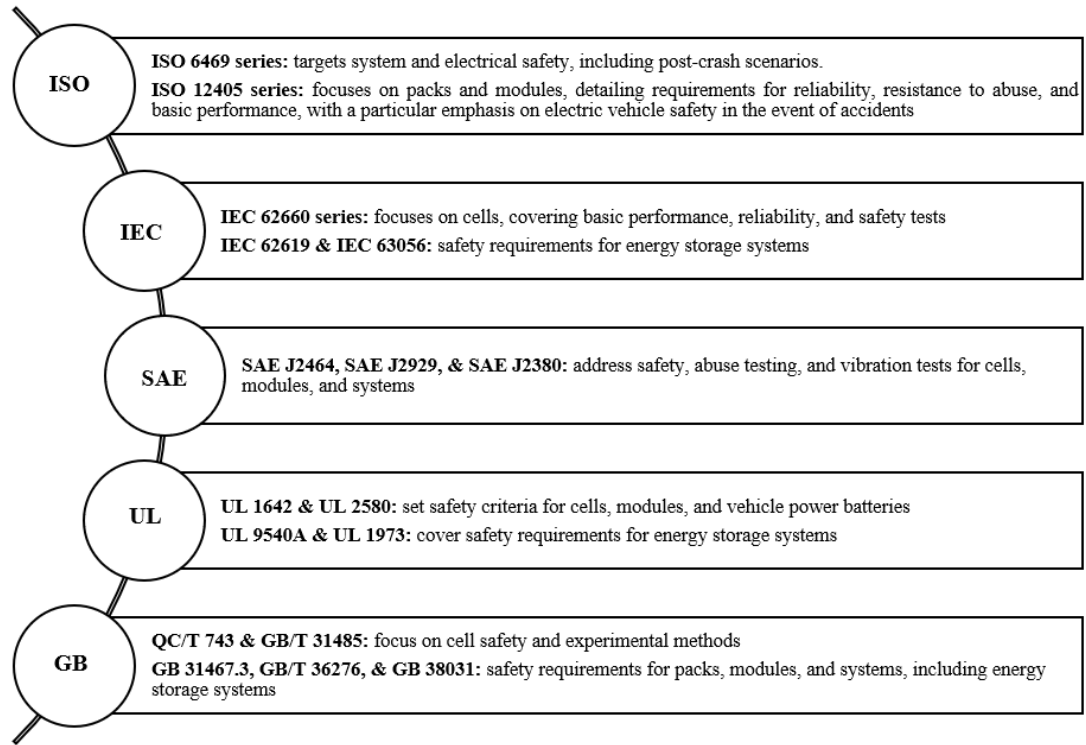


Fig. 3. Applicable standards for LIBs safety (prepared according to [19])

Presented standards cover LIB safety across all levels, from system integrity and crash response to the reliability of packs, modules, and cells, with a strong emphasis on electric vehicle safety during normal use and accidents. However, it is worth mentioning that LIB safety standards are constantly updated.

A Battery Management System (further denoted as BMS) is a system used for rechargeable battery monitoring in real-time. The standard diagram of it is presented in Figure 4. The high voltage multiplexer (HV MUX in Figure 4) at the BMS front end links with the battery, directing cell voltages for the BMS to monitor. As per Figure 4, the Measurement block within the BMS gathers critical metrics. The Management block ensures the maintenance once needed. The Communications block is for connection between the BMS and other devices. The Electronic Control Unit is for the BMS safety commands.

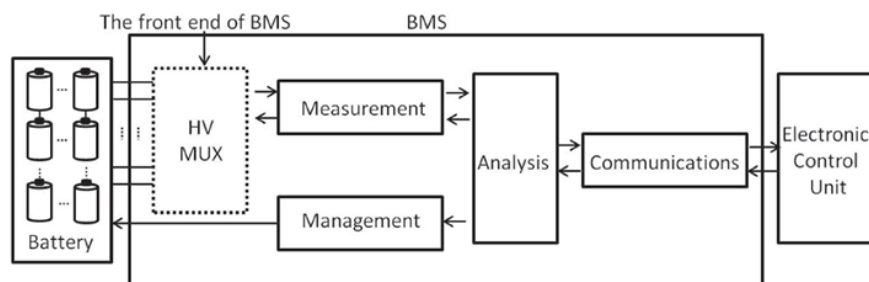


Fig. 4. A standard BMS diagram [20]

The BMS must meet certain requirements. The following aspects are typical of utmost relevance: temperature and voltage data acquisition (covering individual cell voltages, stack voltages, overall pack voltage, and direct current-link voltage), current data acquisition (spanning data from the stack or entire pack), efficient data communication between the BMS master module and slave modules responsible for data acquisition; seamless communication with the battery pack surrounding applications (e.g., vehicles or aircraft), stringent electromagnetic interference (EMI) robustness requirements, the reliability of contactors and system redundancy to enhance functional safety, implementation of galvanic isolation in functional systems, management of balancing mechanisms, optimisation of power consumption, and considerations related to size and weight, among other factors [21].

The integrated BMS for LIBs usually employs various data, such as voltage, current, and temperature of the battery pack, as well as calculated metrics, for example, cell balancing, capability estimation, State of Charge (further denoted as SoC), State of Health (further denoted as SoH), thermal management, and remaining useful life [22]. In addition to the mentioned data, a LIB BMS might also process additional data including the resistance between individual cells, the speed of charging and discharging, electrical resistance within the battery, records of past charge/discharge patterns, the ratio of energy output to input, the flow of electrical power, and comprehensive diagnostic information.

1.2. Characterization of the ageing process and reliability

Two main ageing scenarios for LIBs are identified: calendar ageing, which means that the battery's capacity decreases over time, even without use, and cyclic ageing, which reflects the irreversible reduction in capacity due to constant charging and discharging cycles [23, 24].

Considering the characteristics of degradation over time, both calendar and cyclic ageing can exhibit either linear or non-linear behaviour. It was noticed that in the initial ageing phase, LIB cells demonstrate a linear progression, whereas, towards the later stages, it becomes markedly non-linear, characterized by a swift decline in capacity and an increase in resistance [25].

Analysing key ageing factors, the progression of time, increased temperatures, and a higher state of charge significantly hasten calendar ageing in LIBs; concurrently, cyclic ageing is principally affected by the number of full equivalent cycles, charge-discharge rates, depth of discharge, and the extremities of charge and discharge cut-off voltages [26], however, ageing factors are generally divided into **chemical changes** and **physical mechanisms**. Copper dissolution, dendrite and formation of the Solid Electrolyte Interphase (further denoted as SEI) layer growth, chemical crosstalk, oxygen evolution and gassing, and transition metal dissolution are primarily chemical changes, while pulverisation and electrical isolation, structural disordering, and microcracking and electrolyte penetration represent physical mechanisms; aluminium corrosion entails both chemical and physical processes due to the material's reaction with the environment and the resultant physical degradation [27]. It is observed that temperature, which is neither a chemical change nor a physical mechanism but a physical parameter, also has an impact on the ageing of lithium-ion batteries, exerting a noticeable negative influence on the process. This impact manifests in both time-dependent calendar ageing and usage-related cyclic ageing. Higher temperatures may accelerate secondary reactions within the battery, whereas lower temperatures tend to slow the movement of lithium ions, inducing internal stress. Consequently, fluctuations in temperature can

undermine the battery's thermal stability, further contributing to its ageing. In a scientific study, four groups of commercial lithium-ion batteries were subjected to ageing over 100 cycles at temperatures of 25 °C, 40 °C, 60 °C, and 80 °C respectively, revealing that elevated temperatures result in a diminished battery capacity and a rise in internal impedance [28]. In another study, the effect is emphasized when the temperature increases from 69 °C to over 200 °C. Observed behaviour when increasing temperature [29]:

- temperature equal to 69 °C: each component of the battery remains thermally stable;
- temperature equal to 90 °C: SEI film begins to decompose;
- temperature equal to 120 °C: SEI is destroyed, and the anode reacts with the electrolyte;
- temperature equal to 150 °C: electrodes begin to degrade and release gas;
- temperature equal to 130 °C: separator melts and potentially blocks, risking a short circuit;
- temperature equal to 200 °C: electrolyte burning occurs, leading to battery thermal runaway;
- temperature is above 200 °C: the risk of fire and explosion escalates significantly.

However, it is worth mentioning that ageing factors can be analysed, based on which parts of the LIBs are affected. For example, Vetter et al. [30] described the main mechanisms of ageing in LIB anodes and cathodes (see Figure 5).

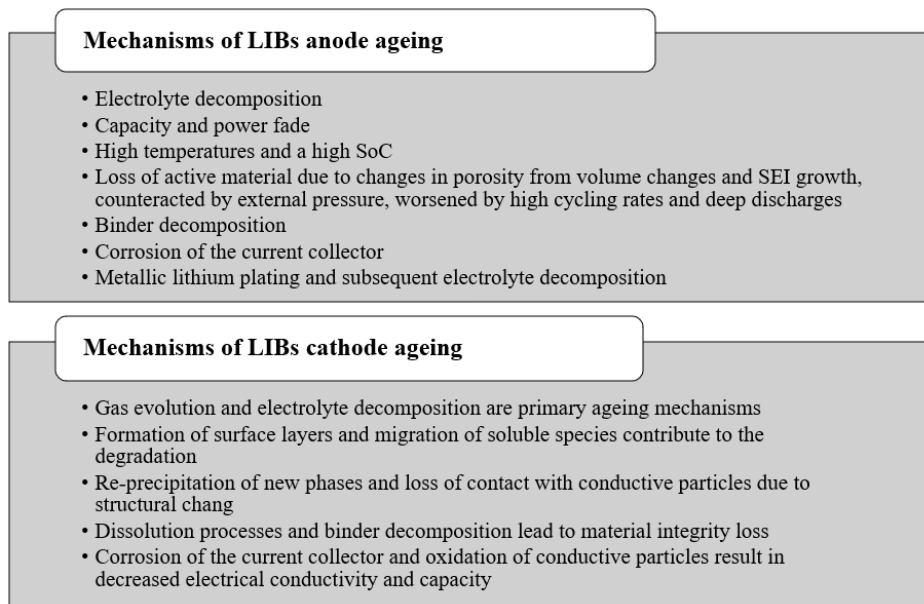


Fig. 5. Main mechanisms of LIBs anode and cathode ageing (prepared according to [30])

According to the information in Figure 5, the anode ageing process is multifaceted, with electrolyte decomposition leading to solid electrolyte interphase formation, which results in lithium loss and increased impedance, ultimately causing capacity and power fade. This can be mitigated by additives that stabilize the SEI, although high temperatures and a high SoC can worsen the issue. Additionally, overcharging may lead to further active material loss due to solvent co-intercalation and gas evolution, which also contributes to capacity fade; this effect can be reduced with a stable SEI or carbon pre-treatment. Other factors influencing anode ageing include changes in porosity from volume changes and solid electrolyte interphase growth, as well as contact loss of active material particles during cycling, both of which can be countered by external pressure but are worsened by high cycling rates and deep discharges.

In general, system reliability refers to the probability that a system will function as intended for a specific period without failing. From a mathematical perspective, the reliability of a system is captured by a function known as $R(t)$:

$$R(t) = P(T > t), \quad (3)$$

indicating the chance that the system will remain functional at a certain moment t . In this context, T symbolizes the variable that measures time until a failure occurs. Typically, the reliability function is related to a concept called the failure rate, symbolized as $\lambda(t)$, which in many reliability models is treated as a constant. This relationship is explained in the equation:

$$R(t) = e^{-\lambda t}, \quad (4)$$

where the exponential term represents the natural decay of system reliability over time.

For complex systems, reliability is determined through a framework that integrates the dependability of each component:

$$R_B = \sum_{j=P-1}^P C_P^j (R_C)^j (1 - R_C)^{P-j}, \quad (5)$$

where R_B represents the overall reliability of the LIB pack, with R_C signifying the reliability of each lithium-ion cell. P stands for the total cell count within the pack, and j counts the specific lithium cell involved [31].

The reliability of LIBs is understood as a structured procedure, which involves both: ageing monitoring and reliability. An example of this is the assessment of the reliability of LIBs within the reliability evaluation of electric vehicles.

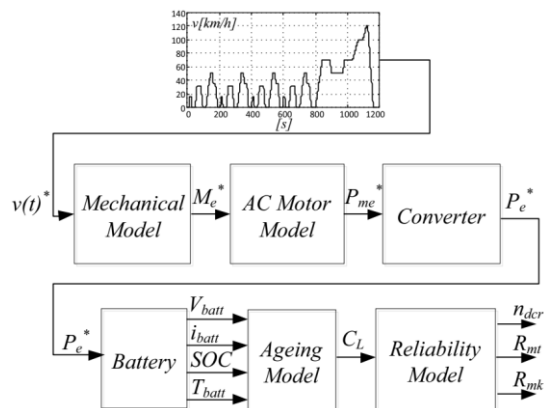


Fig. 6. Reliability assessment procedure for LIBs in electric vehicles [32]

Figure 6 provides a sequence of the complex process for assessing the reliability of LIBs in electric vehicles. Beginning with input from the vehicle's speed, it progresses through a Mechanical Model to an Alternating Current (AC) Motor Model that outlines the power demands. These demands are modulated by a Converter before impacting the battery, characterized by its voltage, current, SoC, and temperature. An ageing model predicts the battery's longevity, and a Reliability Model evaluates the system's reliability in a general context.

1.3. Lithium-ion batteries ageing monitoring and reliability assessment

1.3.1. Most commonly applied ageing models and reliability assessment methods

In the initial stages of LIB ageing research, mainly during the late 1990s and early 2000s, the focus was on explaining degradation processes [33, 34, 35]. These initial models were mostly empirical. The advancement in knowledge regarding the chemistry of LIBs led to the development of more intricate models in the early 2000s. This era marked the emergence of physics-based models that integrated key mechanisms, for example, the growth of the solid electrolyte interphase (SEI) layer [36, 37]. During the 2010s, the application of Machine Learning (further denoted as ML) algorithms became dominant, enabling the examination of large datasets related to battery usage [38, 39]. In recent developments, there has been a shift toward integrating real-life usage data into LIB ageing models to more accurately forecast how batteries will degrade under real-world conditions [40, 41]. In the future, it is expected that LIB ageing models will be increasingly incorporated into BMS [42, 43, 44].

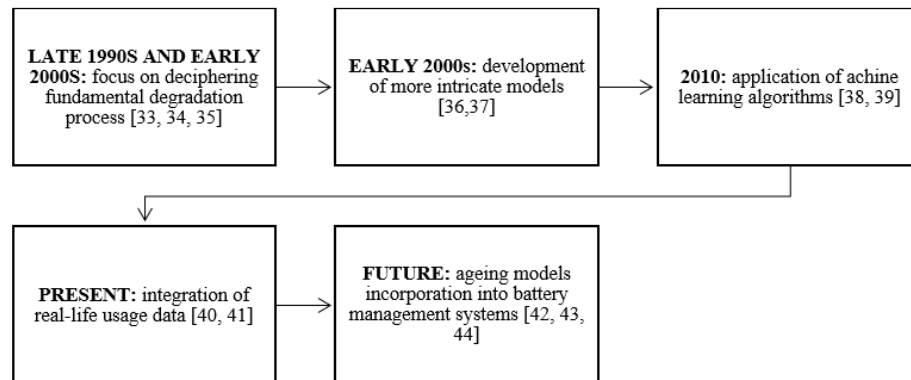


Fig. 7. Development of ageing models for LIBs

Often, when analysing the lifespan of LIBs, ageing models and reliability assessment methods are combined [45]. Thus, they are interconnected and complement each other.

In general, ageing models are classified into Physics-Based Models (PBM), Equivalent Circuit Models (ECM), Empirical and Semi-Empirical Models, and ML models. Reliability methods are classified into Statistical Reliability Analysis, Physics of Failure (PoF) based methods, Probabilistic Risk Assessment (PRA), Health Monitoring and Prognostics, Safety and Risk Analysis, and ML and Deep Learning (further denoted as DL) methods (or models). However, scientific literature may present other groupings as well.

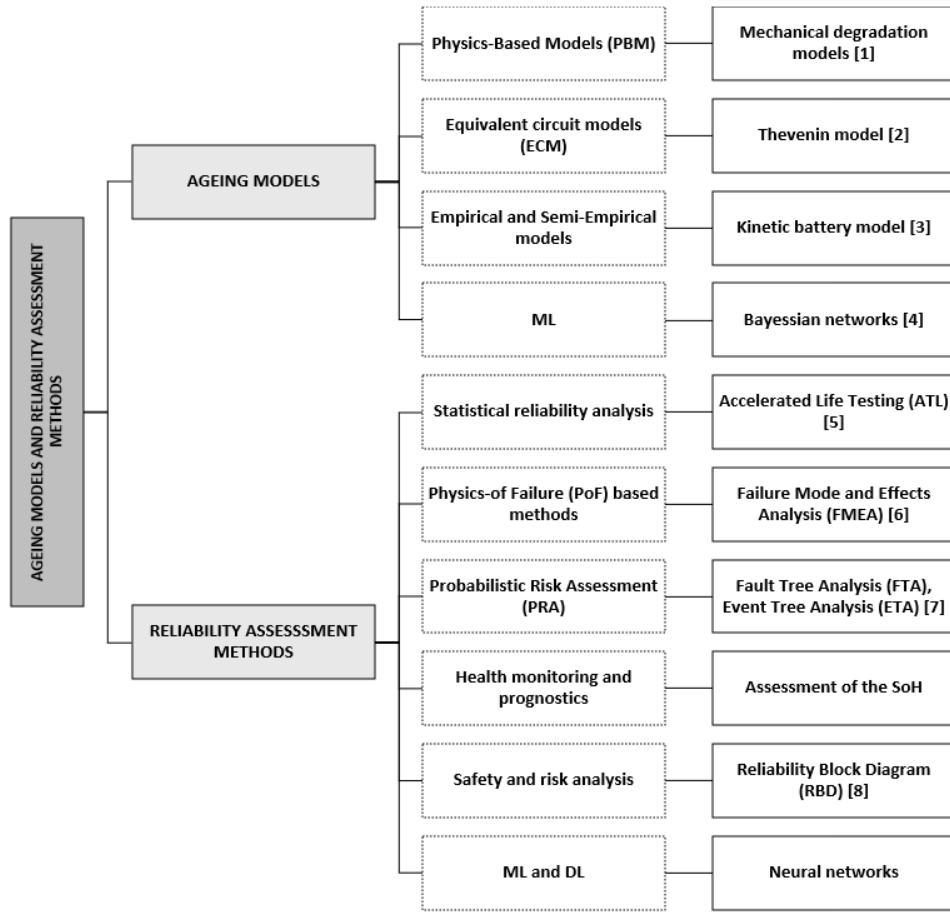


Fig. 8. Classification of ageing models and reliability assessment methods, and their examples

Examining the examples of each of the ageing models, Physics-Based Models (PBM) can be classified as Mechanical Degradation Models, and Equivalent Circuit Models (ECM) can be classified as Thevenin Models. Empirical and Semi-Empirical Models can be classified as Kinetic Battery Models, while an example of ML would be Bayesian Networks. Examining examples of reliability assessment, Statistical Reliability Analysis is classified as Accelerated Life Testing (ALT), Physics-of-Failure (PoF) based methods are classified as Failure Mode and Effects Analysis (FMEA), Probabilistic Risk Assessment (PRA) is classified as Fault Tree Analysis (FTA) and Event Tree Analysis (ETA), Health Monitoring and Prognostics would be internal states such as SoH assessment, while ML and DL are examples of Neural Networks.

Lithium-ion ageing models must satisfy certain conditions. An effective LIB ageing model must comprehensively model both cycle and calendar ageing phenomena, incorporate references to electro-chemo-mechanical-related ageing phenomena, be scalable across multiple hierarchical scales, undergo experimental validation under realistic operating conditions, and be capable of online implementation within a BMS [54]. Additionally, the advantages and disadvantages of each model could be highlighted.

Table 2. Advantages and disadvantages of different ageing models

Model	Advantages	Disadvantages
Physics-Based Model (PBM)	Accurately captures internal battery chemistry, can predict behaviour under varied conditions, useful for understanding fundamental ageing mechanisms [56]	They frequently depend on numerical methods, which can render the models too expensive to deploy in BMS [55]
Equivalent Circuit Model (ECM)	Straightforward parameter determination, minimal computational requirements for model analysis, efficient real-time applicability, and strong adaptability to a wide range of operational conditions [58]	May oversimplify battery behaviour, potentially less accurate for predicting long-term ageing effects [57]
Empirical and Semi-Empirical Model (EM)	Easy to deploy due to the establishment of basic parametric functions that interpolate experimental data [59]	The effectiveness of a model is significantly impacted by its parameters, and an overly complex model with too many parameters can cause mismatches that diminish its accuracy [60]. Also, the empirical nature limits its ability to deliver an in-depth understanding of the battery internal electrochemical interactions, as it is restricted by the scope of its underlying experimental data [61]
ML	Can handle complex, non-linear relationships, adaptable to different types and sizes of data sets [63]	Requires substantial amounts of data for training, can be computationally intensive, and results can be difficult to interpret due to the black box nature [62]

Upon deeper analysis, Physics-Based Models excel in simulating the internal chemistry of LIBs and predicting operational behaviours. Equivalent Circuit Models (ECM) adapt well to various conditions and real-time data but are less precise for long-term ageing. Empirical and Semi-Empirical Models, with their basic parametric functions, can result in low accuracy due to too many parameters and fail to capture internal battery interactions. Conversely, ML offers precision with real data but can be computationally expensive and complex to interpret. Similar to ageing models, reliability assessment methods also have both advantages and disadvantages.

Table 3. Reliability assessment methods advantages and disadvantages

Method	Advantages	Disadvantages
Statistical Reliability Analysis	The capacity to identify failures using historical records [64]	Depend on extensive historical data, which might not reliably forecast future outcomes [65]
Physics-of-Failure (PoF) Based Methods	The ability to analyse potential failure modes specific to the intended environment, identify design improvements, provide realistic forecasts, and quickly assess reliability [67]	It is difficult to apply to all systems [66]
Probabilistic Risk Assessment (PRA)	Efficient for detecting design and operational vulnerabilities [69]	Traditional approaches have restricted capabilities in evaluating and quantifying the dynamic attributes of intricate systems [68]
Health Monitoring and Prognostics	Enable robust, decision-supportive assessments of component health through critical analysis, sensor selection, and methodological evaluations [70]	Are complex to implement [70]
Safety and Risk Analysis	Focuses on identifying hazards and threats [71]	Might not encompass all possible failure situations, particularly in novel or evolving technologies, due to risk factors, including thermal issues, environmental impacts, legal responsibilities, financial concerns, health hazards, and risks of electrocution [72]

Summarising the information presented in the table above, Statistical Reliability Analysis leverages historical data for failure identification, benefiting from data insights but may falter in accuracy with evolving battery technology despite online adaptations. Physics-of-Failure (PoF) methods specialise in failure mode analysis and design optimisation, boosting cost-efficiency and reliability, though they are limited by their focus on known failures and are less applicable to full systems. Probabilistic Risk Assessment (PRA) excels in identifying design and operational flaws but struggles with the dynamics of complex systems. Health Monitoring and Prognostics enhance reliability and reduce costs through detailed health checks, yet require costly, sophisticated technology. Safety and Risk Analysis targets safety risks like thermal runaway effectively but may miss broader failure scenarios in new technology due to diverse risk factors, including environmental and legal issues. The advantages and disadvantages of ML and DL are not discussed, as they are very similar in the case of ageing models.

1.3.2. Assessment of various internal states of lithium-ion batteries

There are many different internal states of lithium-ion batteries, but perhaps the most important ones addressed in research are SoC and SoH. These internal states are interrelated. It has been observed that as a lithium-ion battery is cycled, its SoH decreases. While the SoC is a momentary measure of the battery's current charge level and is independent of SoH, the maximum SoC attainable diminishes over time as the SoH declines [73].

The primary class of SoC assessment methods for LIBs is data-driven methods. These methods are briefly categorized into classification and regression, fuzzy rule-based methods, hybrid algorithms, probabilistic methods, Recurrent Neural Networks (further denoted as RNN), and Feed-Forward Neural Networks [74]. SoC estimation methods for LIBs are more broadly divided into hybrid optimisation-based and hybrid DL.

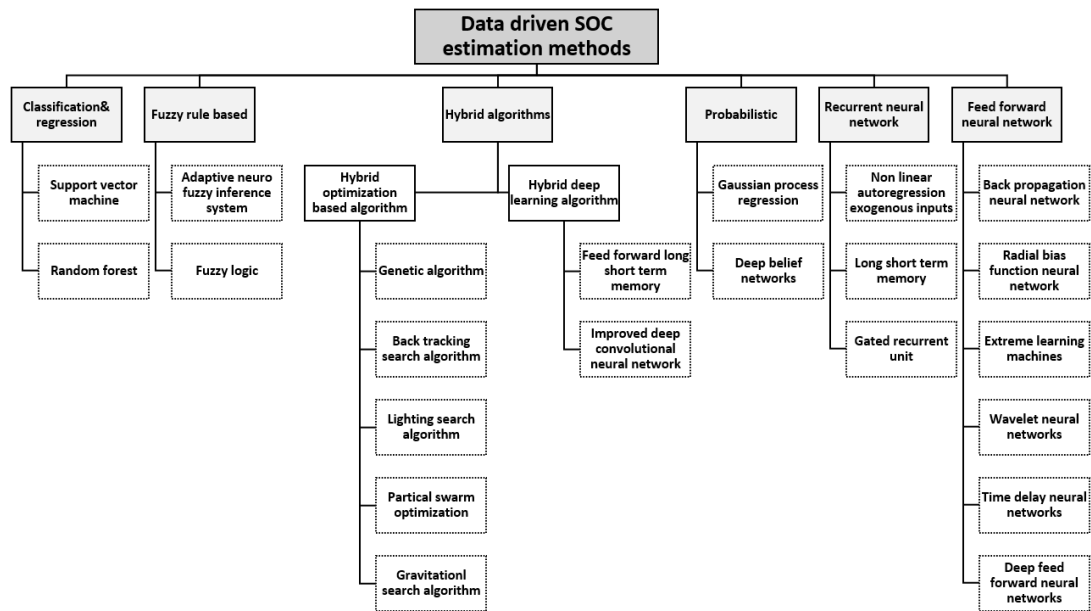


Fig. 9. LIBs SoC estimation methods (prepared according to [74])

On the other hand, SoH estimation methods are classified into direct assessment approach, adaptive approach, data-driven approach and others.

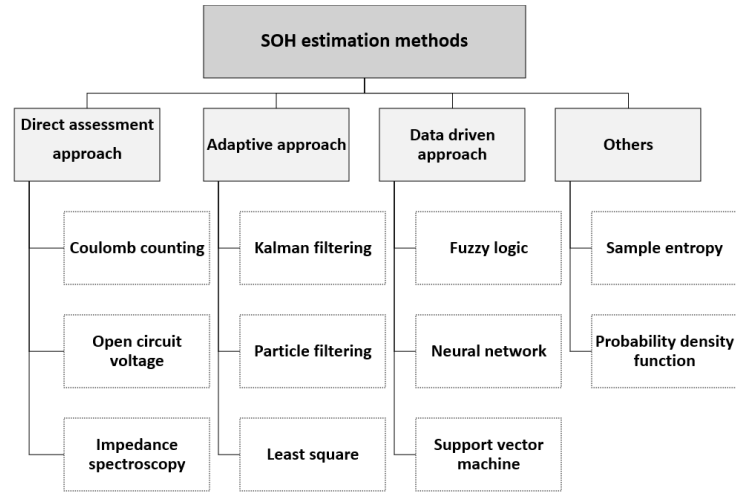


Fig. 10. Lithium-ion batteries SOH estimation methods (prepared according to [75])

The Direct Assessment approach includes Coulomb Counting, Open Circuit Voltage, and Impedance Spectroscopy. The Adaptive approach includes Kalman Filtering, Particle Filtering, and Least Squares. The Data-Driven approach includes Fuzzy Logic, Neural Networks, and Support Vector Machines. Other methods include Sample Entropy and Probability Density Function.

1.3.3. Machine-learning and deep-learning applications

Various advanced algorithms are employed to scrutinise intricate data configurations for lithium-ion batteries, facilitating accurate forecasts regarding their health, efficiency decline, and potential failure dangers. An example of an application for LIB State SoH estimation is visible in Figure 11.

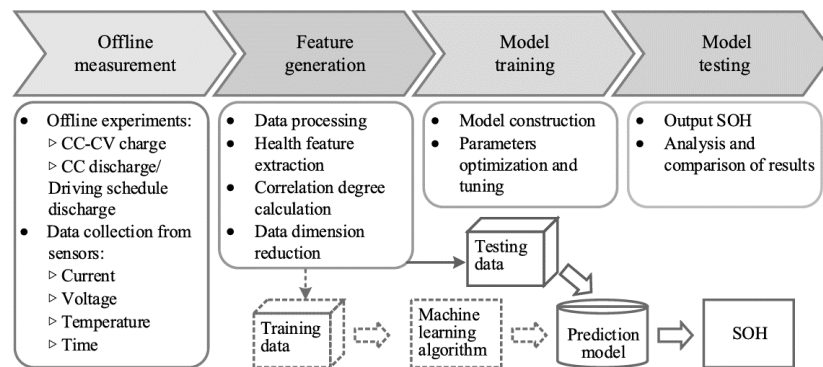


Fig. 11. Example flowchart of ML applications for LIB SoH estimation [76])

The process of applying ML for LIBs SoH estimation starts with collecting offline data, including charge cycles and sensor readings like current and temperature. Next, features are crafted from this data, leading to the construction and refinement of an ML model with training data. Finally, the model is tested, and SoH predictions are made and assessed for their accuracy and reliability. A similar workflow can be applied to SoH assessment.

Further on, ML and DL applications are examined in general, taking into account the internal states and results evaluated (see Table 4). Data preprocessing techniques (see Table 5) and filter applications (see Table 6) were additionally analysed.

It has been observed that different ML and DL models are combined with each other. For example, in the evaluation of SoH, the combined Machine Learning methods such as Support Vector Machine (SVM), k-nearest Neighbours (k-NN), Artificial Neural Networks (ANN), and Linear Regression (further denoted as LR) are used alongside the Deep Learning method Deep Neural Networks (DNN). In many cases, the different results of both axiomatic and Deep Learning methods are tested and compared. For instance, when evaluating only SoC and testing only Deep Learning methods such as the Fully Connected Neural Network (FCNN), Long Short-Term Memory (further denoted as LSTM), Gated Recurrent Unit (further denoted as GRU), and Temporal Convolutional Network (TCN), it was observed that the highest accuracy is obtained with only Convolutional Neural Network (CNN).

Table 4. Applications of ML and DL for state estimation of LIBs

Paper	Internal state		ML model	DL model	Results
	SoC	SoH			
[77]	+		Not applicable	Fully Connected Neural Network (FCNN), LSTM, Gated Recurrent Unit (GRU), Temporal Convolutional Network (TCN)	The most accurate results were obtained by applying Convolutional Neural Network (CNN), while the shortest code execution time was obtained by applying GRU
[78]	+		Not applicable	Long Short-Term Memory (LSTM) – Recurrent Neural Network (RNN) combination	The proposed Long Short-Term Memory (LSTM)/(RNN) model can learn to estimate SoC varying temperatures accurately
[79]		+	K-Nearest Neighbor (KNN), Support Vector Regressor (SVR), Decision Tree (DT), Random Forest (RF) Regressors	Not applicable	The highest accuracy was obtained by applying K-Nearest Neighbor (KNN) and Decision Tree (DT)
[80]	+		Ridge regression, Linear Regression, Lasso, Elastic Net, Linear Regression with L2, K-means	Not applicable	Linear Regression with L2 outperformed other methods in accuracy terms
[81]		+	Not applicable	Feed-Forward Neural Network (FNN), Convolutional Neural Network (CNN), Long Short-Term Memory Neural Network (LSTM)	Better performance was detected with simple LSTM and Feed Forward Neural Network (FNN)
[82]	+	+	Gaussian Process Regression (GPR)	Not applicable	Combining Gaussian Process Regression (GPR) with Improved Firefly Algorithm (IFA), which is an optimisation algorithm, can improve the accuracy of SoC and SoH estimates

A combination of DL models, specifically LSTM with RNN, has made it possible to perform State of Charge (SoC) evaluations at different temperatures. It is also common to apply Machine Learning methods such as K-Nearest Neighbours (KNN), Support Vector Regressor (SVR), Decision Tree (DT), Random Forest (RF) Regressors, and Gaussian Process Regression (GPR), as well as Deep Learning methods including Feed-Forward Neural Networks (FNN), Convolutional Neural Networks (CNN), and LSTM. The analysis performed reveals that testing only one method is not sufficient; a comparison of the results from different methods is necessary to achieve the highest possible accuracy in calculations.

It has been observed that models designed for the evaluation of internal states are being refined using optimisation techniques. In one of the studies, Neural Networks were improved by applying Bayesian optimisation [83].

Performing data preprocessing before evaluating internal states and applying ML and DL models often aims to remove noise or outliers in the data. Filters such as the Extended Kalman filter (EKF), Gauss filter, Savitzky-Golay Filter (further denoted as SGF), Rlowess & Rloess filters, and Median filter are used for that purpose. Signal processing methods are also applied: Wavelet Transform (further denoted as WT), Stacked Denoising Autoencoders (SDAE), and CEEMDAN (Complete Ensemble Empirical Mode Decomposition with Adaptive Noise).

Table 5. LIB data preprocessing techniques

Paper	Internal state		ML and DL models applied (without filters)		Techniques and their applications	
	SoC	SoH	Machine learning models	Deep learning models	Techniques applied	Description of techniques application
[84]	+		Linear Regression, Elastic net, Gradient Descent, Bayesian Ridge, Lasso, RANSAC (Random Sample Consensus) regression, Gradient Boosting Algorithm, Bagging, XGBoost, Histogram Based Gradient Boosting, LightGBM, AdaBoost, tree-based Random Forest, Decision Trees, KNN regression, ANN (Artificial Neural Network), SVR, ExtraTree (Extremely Randomized Trees)	Not applicable	Gauss filter, SGF, Rlowess & Rloess filters, Median filter	Removing outliers in battery data
[85]	+		Not applicable	Feedforward Neural Network (FFNN)	Extended Kalman filter (EKF)	Excluding time constants in battery data
[86]	+		The Support Vector Regression algorithm	Extreme Learning Machine (ELM)	Stacked Denoising Autoencoders (SDAE)	Lithium-ion battery data distribution corrections
[87]		+	Gaussian Process Regression (GPR)	Not applicable	Wavelet Transform (WT)	Smoothing the Differential Thermal Voltammetry (DTV) curves

Applications of filters for direct evaluation of the internal states of batteries are presented in Table 6

Table 6. Applications of filters for direct evaluation of internal states of batteries

Paper	Internal state		Machine learning and deep learning models applied (without filters)		Filters applied
	SoC	SoH	ML models	DL models	
[88]	+	+	Endogenous Immune Algorithm (EIA)	Not applicable	Adaptive dual square root Kalman filtering (ADSRCKF)
[89]	+	+	Improved Firefly Algorithm, Recursive Least Square Method with Forgetting Factor	Not applicable	Particle filter
[90]		+	Recursive Least Square (RLS)	Not applicable	Improved particle filter
[91]	+		Not applicable	Deep Feed-Forward Neural Network (DFNN)	Kalman Filter (KF), Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF)
[92]	+		Regression Analysis, Support Vector Machines (SVM), Random Forests, Ensemble Learning	Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) networks, Autoencoders, Deep Feed-Forward Neural Networks (DFFNs)	Kalman Filter (KF), Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF)

For direct internal states estimation, it is common to apply different variations of Kalman Filter (further denoted as KF) and Particle filters; Adaptive Dual Square Root Kalman Filtering (ADSRCKF), KF, Extended Kalman Filter (EKF), Unscented Kalman Filter (UKF), Particle filter and Improved particle filter. ML models such as Regression Analysis, Support Vector Machines (SVM), Random Forests, Ensemble Learning, Recursive Least Square (RLS), Improved Firefly Algorithm, Recursive Least Square Method with Forgetting Factor and Endogenous Immune Algorithm (EIA) are combined with filters.

2. Research methods and models

2.1. Research workflow and synthetic data generation

This research consists of several experiments as per the general structure workflow shown in the figure below. The research begins with the preparation of the datasets. Synthetic LIB charging data was generated separately and preprocessed. Afterwards, an SoC evaluation was conducted. Real LIB data were also collected, according to which the SoC evaluation was performed, and they were preprocessed analogously to the synthetic data. Further on, ML and DL models were developed. Models on synthetic data were trained and tested with 100, 70, 50, and 10 epochs when applying a specific Neural Network and using both raw and processed data. For the initial and preprocessed real battery data, models were trained and tested. In this case, for Neural Networks, only 10 epochs were set. SoH assessment was not conducted for synthetic data; however, it was carried out for real data. After training and testing machine learning and deep learning models, all results were analysed. Based on research results, the application system for battery monitoring development that ended the entire research followed. In addition, a separate case with KF for only one parameter estimation was tested.

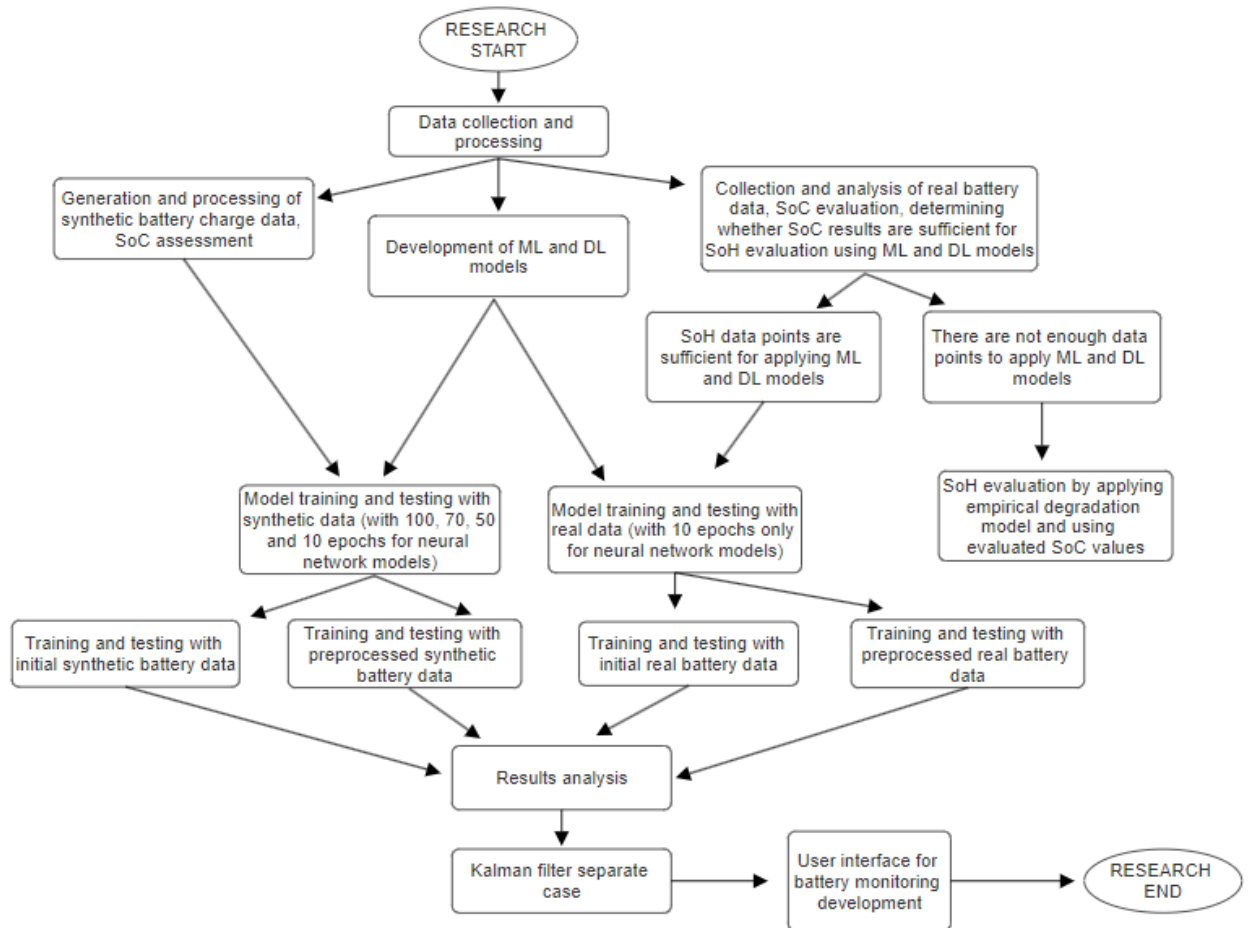


Fig. 12. General research workflow structure

A generated dataset of synthetic LIB charge data has 10,000 data points and five variables: voltage, current, temperature, time (in seconds), and capacity. Data generation begins with parameterisation. All variables were set as an array. To reproduce real data as accurately as possible, step interval, step increase, maximum capacity, voltage start, voltage end, voltage step, current start, and current peak values were predetermined (see Table 7). Voltage was calculated at each time point for the available charge capacity, while the current was set to be calculated based on changes in the charge capacity. The temperature variable was set to start at 25 °C and rise to 40°C, falling to 28 °C. The step interval was set equal to 500 when its increase was set equal to 0.1. Maximum capacity is set at 4.2 (Ah). Voltage starts at 2.5 (V) and ends at 4.2 (V). Per step, it increases by 0.1 (V). Current starts from 0 Amperes (A) with a peak equal to 1 (A).

Table 7. Values of parameters used for synthetic dataset generation

Parameter	Value
Step interval	500
Step increase	0.1
Max capacity	4.2 (Ah)
Voltage start	2.5 (V)
Voltage end	4.2 (V)
Voltage step	0.1 (V)
Current start	0.0 (A)
Current peak	1.0 (A)

The data generation process can also be explained with mathematical formulas. For example, the generation of time in seconds can be expressed as a linear sequence:

$$t_i = i \text{ for } i \in \{0, 1, 2, \dots, n - 1\}, \quad (6)$$

where t_i indicates linearly increasing time for time step I , and n indicates the overall number of data points.

Two conditions were set for the variable capacity. According to one of them, the capacity increases, and according to the other, there is no loss if the first condition remains unchanged:

$$C(i) = \begin{cases} C(i - 1) + \delta, & \text{if } (i \bmod \tau = 0) \text{ and } (C(i - 1) < C_{max}) \\ C(i - 1), & \text{otherwise} \end{cases}, \quad (7)$$

there $C(i)$ is for charge capacity at a certain i time, C_{max} is the maximum capacity, δ indicates the amount for which the battery increases, while τ indicates the time interval for which the battery increases.

The current was calculated as follows:

$$I(i) = \begin{cases} -I_{initial}, & \text{if } (C(i) > C(i-1)) \text{ and } (i = 1) \\ I(i-1) - \text{value in } [0.01, 0.05], & \text{if } (C(i) > C(i-1)) \text{ and } (i \neq 1) \\ I(i-1) + \text{value in } [-0.01, 0.01], & \text{otherwise,} \end{cases} \quad (8)$$

there $I(i)$ is for current at a certain i time, $-I_{initial}$ is initial current at the beginning, $C(i)$ indicates charge capacity at a certain i time, and i indicates time step.

To ensure that the voltage value does not fall below the set initial value and does not exceed the end value (or otherwise voltage end), the following boundaries were set:

$$V(i) \in [V_{start}, V_{end}], \quad (9)$$

there V_{start} voltage at the beginning of the test period and V_{end} is the voltage at the end of the period. Their values are presented in Table 7 as mentioned previously.

2.2. Mathematical models of states of lithium-ion batteries

SoC can be evaluated by integrating ampere-hours. Then the model is being used:

$$SOC_{(t)} = SOC_{(t_0)} - \int_{t_0}^t \frac{k_T i_{(t)}}{k_I C_0} dt, \quad (10)$$

there t_0 is for the initial time, t is the current time, $i_{(t)}$ is current at a certain time, k_T is constant of the temperature effect.

SoC can also be evaluated by taking into account certain factors affecting the battery, such as discharge current, temperature, and cycle times. With such data, the state of charge is calculated according to Equation 11.

$$\begin{aligned} SOC &= \left(SOC_0 - \frac{C_{dis}}{C_{in}} \right) \times 100\% \\ &= \left(SOC_0 - \frac{\int_{t_0}^t l(t) dt}{C_{in}} \right) \times 100\% \\ &= \left(SOC_0 - \frac{\int_{t_0}^t l(t) dt}{(a_0 T^2 + a_1 T + a_2) I^{1-b_0 T^2 - b_1 T - b_2}} \right) \times 100\% \\ &= \left(SOC_0 - \frac{\int_{t_0}^t l(t) dt}{(k_h(a_0 T^2 + a_1 T + a_2) I^{1-b_0 T^2 - b_1 T - b_2})} \right) \times 100\%, \end{aligned} \quad (11)$$

there, C_{dis} is discharge capacity, while C_{in} is charging capacity, $l(t)$ is the current time, t_0 is initial time, T is the temperature. Also, a_0 , a_1 , a_2 , b_0 , b_1 , b_2 , k , and h are coefficients representing the relationship between variables.

For SoH assessment:

$$SoH = \frac{C_a}{C_r} \times 100\%, \quad (12)$$

There C_a is actual (or measured) capacity and C_r is rated capacity (usually determined by the battery manufacturer).

There are other ways to perform SoH assessment. One of them used in this research is the empirical degradation model expressed in equation [93]:

$$h = k_1 C + k_2 e^{\alpha C} + 1 - k_2, \quad (13)$$

there h indicates SoH, k_1 together with k_2 are unknown parameters, $e^{\alpha C}$ is decay factor, C is for the battery cycle.

2.3. Signal processing methods for data preprocessing

Discrete wavelet transform (further denoted as DWT), Single-Layer Denoising Autoencoder (further denoted as SLDA) and SGF were used for data preprocessing.

The DWT decomposition process can be explained as follows [94]:

$$f(t) = \sum_{k=-\infty}^{+\infty} C_{n,k} \phi(2^{-n}t - k) + \sum_{k=-\infty}^{+\infty} \sum_{k=-\infty}^{+\infty} 2^{-\frac{j}{2}} d_{j,k} \psi(2^{-j}t - k), \quad (14)$$

Where $f(t)$ indicates reconstructed signal, 2^{-j} indicates normalization factor, n indicates the highest decomposition level, $\psi(2^{-j}t - k)$ indicates function scaling, $d_{j,k}$ indicates coefficients at scale j , $C_{n,k}$ indicates approximation coefficients, and j indicates the index of scale level. Based on this equation, DWT decomposition is an iterative process.

There are a lot of functions that could be used for DWT. The ones mentioned in the scientific literature are Haar, Daubechies, Biorthogonal, Coiflets, Symlets, Morlet, Mexican Hat, and Meyer functions [95]. For illustrative purposes, the Daubechies function was used in this research.

Another signal processing method applied is SGF, which can be expressed as an equation [96]:

$$S_j^* = \frac{\sum_{i=-m}^{i=m} C_i S_{j+i}}{N}, \quad (15)$$

there C_i is smoothing coefficient, N indicates the amount of data points, stands for initial signal, s^* stands for smoothed signal (output), and m indicates the range of indices.

The figure below provides a graphical illustration of SLDA. There $\tilde{x}$ is noisy input, g_θ is a decoder, q_D is corruption process, f_θ decoder, $L_H(x, z)$ is the loss function, x is the noisy input encoded version, and z is x in latent space.

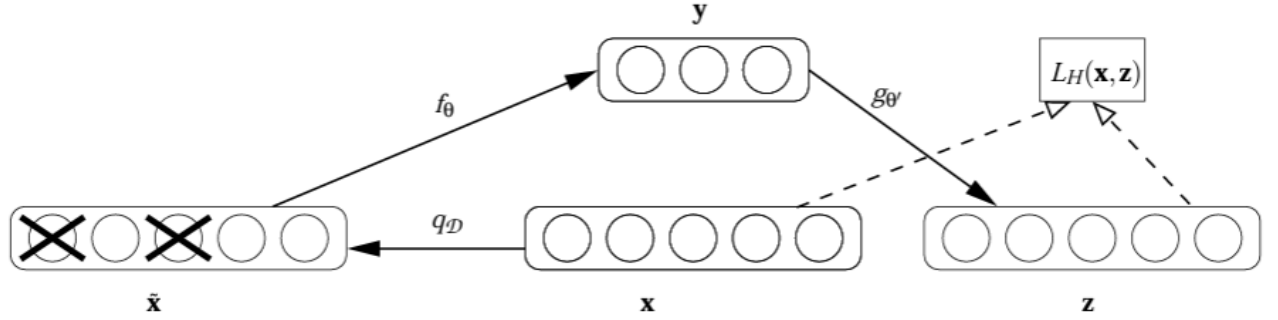


Fig. 13. Single Layer Denoising Autoencoder graphical illustration [97]

According to the figure above, the entire process of SLDA begins with providing input data. Next, the given data is compressed into a latent space. This action is performed by the encoder. After this, the decoder reconstructs the initial data from the compressed latent space. At the end of the process, the loss function is calculated.

2.4. Machine Learning models and their evaluation metrics

2.4.1. Artificial neural network models architectures

In this research, three neural network architectures were created: Simple Recurrent Neural Network (further denoted as SRNN), LSTM, and GRU. These models fall into the DL category, a subfield of ML. Additionally, all of them are types of RNNs. The main idea behind choosing these models was to start with the basic one and observe the impact on results by using more complex ones. The basic RNN architecture is presented in the figure below.

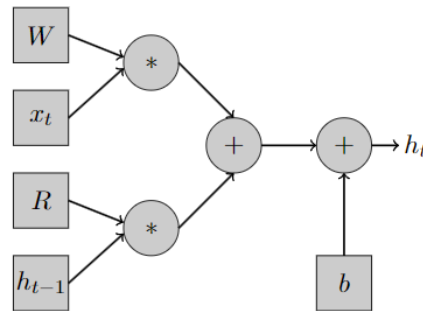


Fig. 14. Basic RNN architecture [98]

In a general sense, in basic RNN architecture, there is an input weight matrix, input, recurrent weight matrix, previous hidden state, bias, and hidden state. It is worth mentioning that it also has an output that is not visible in the figure above. To keep information inside the network, several loops are implemented within the RNN [99].

Only trainable parameters were used for the models. The highest number of parameters, equal to 31,051, was for LSTM, while the lowest number, equal to 7,801, was for SRNN.

Table 8. Number of parameters in each neural network

Neural network	Number of parameters
SRNN	7 801
LSTM	31 051
GRU	23 601

Each neural network created for this research has an input layer, a first and second SRNN (or LSTM/GRU) layer, as well as a dense layer. For all of the model layers, the first "none" indicates that the model can handle any batch size, while the second "none" indicates that the model can handle any length of data. A number equal to 50 indicates the number of features. For the dense layer, "none" indicates batch size, while a number equal to 50 is for input size. A number equal to 1 in dense layers indicates that only one value per data sequence will be retrieved. After each layer in Figure 16, the output layer shows data changes inside the model

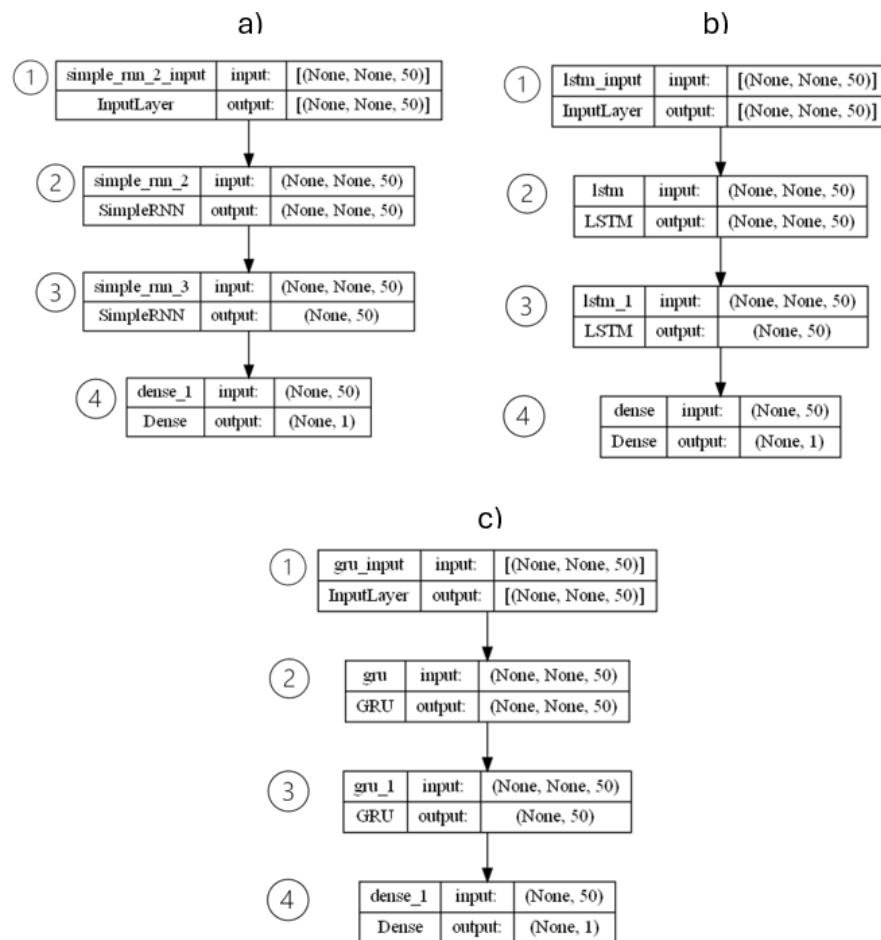


Fig. 15. SRNN architecture with input layer (1), first SRNN layer (2), second SRNN layer (3) and dense layer (4) for initial model (a), LSTM architecture with input layer (1), first LSTM layer (2), second SRNN layer (3) and dense layer (4) for initial model (b), GRU architecture with input layer (1), first GRU layer (2), second GRU layer (3) and dense layer (4) for initial model (c)

The initial neural networks were refined using Bayesian optimisation. In this case, Bayesian optimisation was applied specifically to optimize the learning rate and the number of units. For the initial models, the number of units was randomly chosen to be 50, and the learning rate was set to 0.001. After applying Bayesian optimisation to the SRNN, the number of units did not change, but the learning rate increased to 0.0360. In the case of the LSTM, the number of units increased to 100,

and the learning rate increased to 0.0496. For the GRU model, the number of units also increased to 100, while the learning rate increased to 0.0532.

Table 9. Number of units and learning rate for initial models after applying Bayesian optimisation

Model	Initial values		Values Bayesian optimisation	
	Number of units	Learning rate	Number of units	Learning rate
RNN	50	0.001	50	0.0360
LSTM	50	0.001	100	0.0496
GRU	50	0.001	100	0.0532

When applying Bayesian optimisation, the optimisation problem solved is expressed by the equation:

$$x^* = \underset{x \in X}{\operatorname{argmin}} f(x), \quad (16)$$

there x^* is an optimal solution, argmin is the x value which minimizes the f function, and $f(x)$ is the function that is being minimized.

Thus, according to one of the related studies [100], Bayesian optimisation for an optimal number of units and learning rate can be written as the equation for expected improvement:

$$EI(x) = (\mu(x) - y^*) \Phi(Z) + \sigma(x)\phi(Z), \quad (17)$$

there EI indicates expected improvement, Φ indicates cumulative distribution function, ϕ represents probability density function, y^* indicates the lowest value, μ indicates mean, σ indicates standard deviation, while Z measures standard deviation for y^* . The optimal learning rate was found from the range of 0.001 to 0.1 inclusive, while the optimal number of units was searched by setting the number of units to 10, 50, and 100.

To ensure the learning quality of each neural network model, data normalization was performed:

$$X_{normalized} = \frac{X - X_{minimum}}{X_{maximum} - X_{minimum}}, \quad (18)$$

there X is for original data, while $X_{maximum}$ indicates the maximum value of the original data, and $X_{minimum}$ the minimum value of the original data.

For all machine learning models, a specific data set split ratio for training and testing was applied, and performance evaluation metrics (detailed in Chapter 2.4.4) were used. In the case of neural networks and regression models, the data sets were split into training and testing sets at a ratio of 80:20, indicating that 80% of all data was allocated for training, while 20% was used for testing. For the Kalman filter, the experiment was conducted by allocating 70% of the entire data set for training and 30% for testing. For neural networks, Root Mean Squared Error (further denoted as RMSE), Mean Squared Error (further denoted as MSE), Tweedie error, and R^2 were applied. In the case of regression models and the Kalman filter, Tweedie error was not applied because

experiments with neural networks revealed that the results from this metric were inaccurate (see Chapter 3).

Table 10. Models split ratio and performance metrics

Model	Train: test split ratio	Model performance evaluation metrics used			
		RMSE	MSE	TDE	R^2
RNN	80:20	+	+	+	+
LSTM	80:20	+	+	+	+
GRU	80:20	+	+	+	+
Linear Regression	80:20	+	+		+
Ridge Regression	80:20	+	+		+
Kalman filter	70:30	+	+		+

Many optimisers are used for training RNNs and other models. For demonstration purposes, all created models were trained with the Adam optimiser. Mathematically, the Adam optimiser can be expressed by the following equation:

$$\begin{aligned}
 m_t &\leftarrow \beta_1 m_{t-1} + (1 - \beta_1) g_t & v_t &\leftarrow \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \hat{m}_t \\
 &\leftarrow \frac{m_t}{1 - \beta_1^t} & \hat{v}_t &\leftarrow \frac{v_t}{1 - \beta_2^t} \Delta w_t \\
 &\leftarrow - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \hat{m}_t,
 \end{aligned} \tag{19}$$

there g_t denotes gradient, m_t denotes the first moment, v_t denotes the second moment, β_1 along with β_2 denotes decay rates, Δw_t denotes weight update, ϵ denotes small constant and η is for the learning rate.

Rectified Linear Unit (further denoted as ReLu) was chosen as the activation function for each model case. According to the graph of this function presented in the figure below, it can be observed that from the point (0, 0) the line goes up when input $u \geq 0$, but if input $u < 0$, then zero values are obtained.

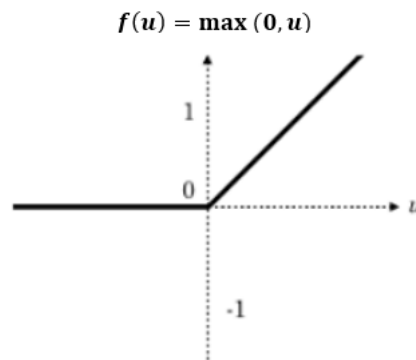


Fig. 16. ReLu function visualisation [101]

2.4.2. Regression models and Kalman filter

LR model for one dependent variable and one independent variable could be expressed in equation form:

$$Y = \beta_0 + \beta_1 X + \epsilon, \quad (20)$$

where parameter β_0 is for intercept, parameter β_1 is for slope, Y is the variable to be predicted/calculated known as the dependent variable, X indicates independent variables and ϵ is known as the error term.

Since in this research, the LR model is applied to one dependent variable with several independent variables, then the multiple linear regression equation can be expressed by the following equation:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_n X_n + \epsilon. \quad (21)$$

Formally Ridge Regression is expressed as an equation:

$$\hat{\beta}_R = (X'X + kI)^{-1}X'y, \quad (22)$$

there $\hat{\beta}_R$ is estimated regression coefficients matrix, X' is the transpose matrix of the X matrix that shows predictions, k is the parameter identifying the penalty, I is for identity matrix and y is the dependent variable vector. Depending on the task this equation can be expressed differently. In the context of this research the following formula was used, where Ridge Regression is formalised as the objective function:

$$J(\beta) = \sum_{i=1}^m (y_i - X_i\beta)^2 + \alpha \sum_{j=1}^n \beta_j^2, \quad (23)$$

where $\sum_{i=1}^m (y_i - X_i\beta)^2$ indicates residuals summation and $\alpha \sum_{j=1}^n \beta_j^2$ indicates penalty. In this research, the optimal α value was found automatically within a range from 0.001 to 100 inclusive.

In this study, the KF was applied to achieve two different objectives. One is battery data evaluation, and the other is solving the same task, but without taking into account all data points. In the second case, the obtained results were used as input for neural network models. In the first case, the whole process is described in the figure below.

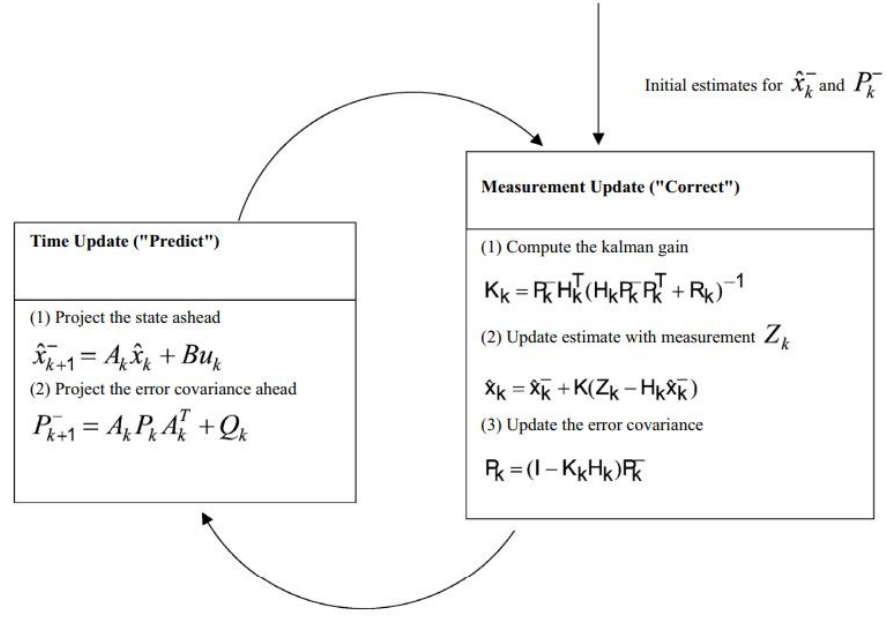


Fig. 17. Standard process of KF [102]

The whole process starts with initial estimates. Based on these initial estimates, a measurement update is performed, which includes steps such as Kalman gain computations and error covariance update. Then comes the time update, which includes projecting the state and error covariance ahead. In the second option, the whole process was somewhat simplified. The simplified process of the KF is presented in the figure below.

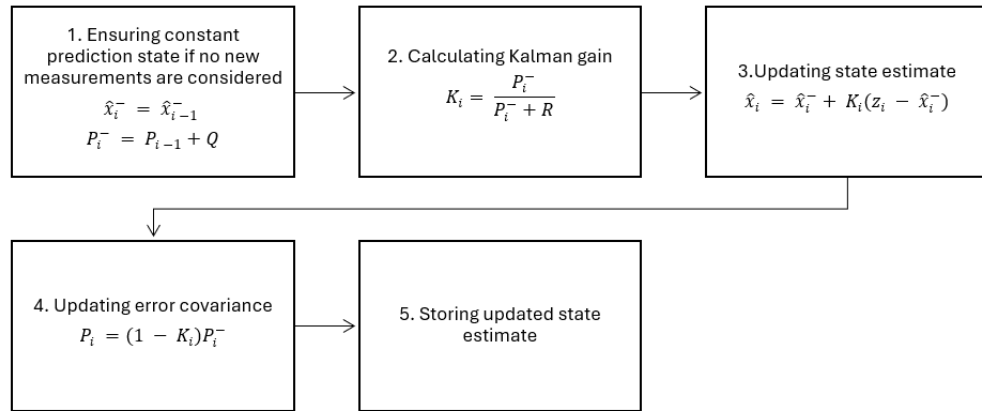


Fig. 18. Simplified process of KF

The process begins by determining the constant prediction state when no new measurements are received. Then, the Kalman gain is calculated, followed by the state estimate and error covariance update. All updated state estimates are stored.

2.4.3. Metrics for models performance

RMSE, MSE, R^2 and Tweedie Deviance Error (TDE) were applied for model performance estimation. Usually, RMSE and MSE are used while solving regression tasks similar to this research, but Tweedie Deviance Error may also be suitable for this task. Several metrics are chosen to ensure robustness in model testing.

Equation for RMSE calculations:

$$\text{RMSE} = \sqrt{\frac{1}{m} \sum_{i=1}^m (X_i - Y_i)^2} \quad (24)$$

there X_i predicted value for each observation, Y_i denotes the real value for each observation, and m denotes the total number of observations.

The equation for MSE calculations:

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{Y}_i - Y_i)^2 \quad (25)$$

there Y_i denotes the real value for each observation, $\hat{Y}_i$ denotes the predicted value for each observation, and n denotes the total number of observations.

Tweedie Deviance Error equation could be expressed as follows:

$$TDE = \begin{cases} (\hat{y} - y)^2, \text{ for } p = 0 \\ 2 \times \left(y \times \log\left(\frac{y}{\hat{y}}\right) + \hat{y} - y \right), \text{ for } p = 1 \\ 2 \times \left(\log\frac{\hat{y}}{y} + \frac{y}{\hat{y}} - 1 \right), \text{ for } p = 2 \\ 2 \times \left(\frac{(\max(y, 0))^2}{(1-p)(2-p)} - \frac{y(\hat{y})^{1-p}}{1-p} + \frac{(\hat{y})^{2-p}}{2-p} \right), \text{ otherwise} \end{cases} \quad (26)$$

In this equation $\hat{y}$ represents the predicted value, y represents the real value, and p represents the power parameter. Also, $(\hat{y} - y)^2, \text{ for } p = 0$ is for Normal distribution, $2 \times \left(y \times \log\left(\frac{y}{\hat{y}}\right) + \hat{y} - y \right), \text{ for } p = 1$ is for Poisson distribution, $2 \times \left(\log\frac{\hat{y}}{y} + \frac{y}{\hat{y}} - 1 \right), \text{ for } p = 2$ is for Gamma distribution, $2 \times \left(\frac{(\max(y, 0))^2}{(1-p)(2-p)} - \frac{y(\hat{y})^{1-p}}{1-p} + \frac{(\hat{y})^{2-p}}{2-p} \right), \text{ otherwise}$ indicates Tweedie distribution, where $1 < p < 2$.

R^2 is usually written as the equation:

$$R^2 = \frac{SS_{Regression}}{SS_{Total}}, \quad (27)$$

there $SS_{Regression}$ is for the explainable sum of squares, while SS_{Total} is for the general amount of summarized squares.

2.4.4. Software used for modelling

The research was conducted using the Python programming language. This choice was influenced by the fact that Python has many packages designed for creating machine learning models, its convenience for developing application systems, and its ability to adapt to fundamentally different personal needs. The Python libraries used are listed according to the tasks addressed in the table below.

Table 11. Applied Python libraries according to groups of tasks to be solved

Library group by task	Library name
Dataset manipulation and visualization	pandas, numpy, matplotlib.pyplot, matplotlib.dates, matplotlib.lines, seaborn
Integration and optimisation	scipy.optimize, scipy.integrate, GPyOpt
Signal processing and statistical analysis	scipy.signal, scipy.stats, statsmodels, pywt
General math functions	math
ML	tensorflow, tensorflow.keras, sklearn.model_selection, sklearn.preprocessing, sklearn.metrics, sklearn.linear_model, time, random, random as, python_random, filterpy.kalman, filterpy.common
Application system development	matplotlib.backends, os, ttkthemes, tkinter

For solving tasks related to dataset manipulation and visualization, the libraries pandas, numpy, and matplotlib.pyplot, matplotlib.dates, matplotlib.lines, and seaborn were applied. For integration and optimisation solutions, the libraries scipy.optimize, scipy.integrate, and GpyOpt were used. During data preprocessing and statistical analysis, the libraries scipy.signal, scipy.stats, statsmodels, and pywt were employed. Additionally, the math library was used for general mathematical tasks. For the creation and testing of machine learning models, the libraries tensorflow, tensorflow.keras, sklearn.model_selection, sklearn.preprocessing, sklearn.metrics, sklearn.linear_model, time, random, random as python_random, filterpy.kalman, and filterpy.common were applied. Finally, for the application system development, the libraries matplotlib.backends, os, ttkthemes, and tkinter were used.

3. Research results

3.1. Data overview and preprocessing results

3.1.1. Synthetic data case

In the case of synthetic data, dynamic current was obtained. It can be seen that the current drops at the end of the tests, which is not at all characteristic of real data for LIBs. The voltage rises at the beginning of the tests until it finally becomes unstable at certain values.

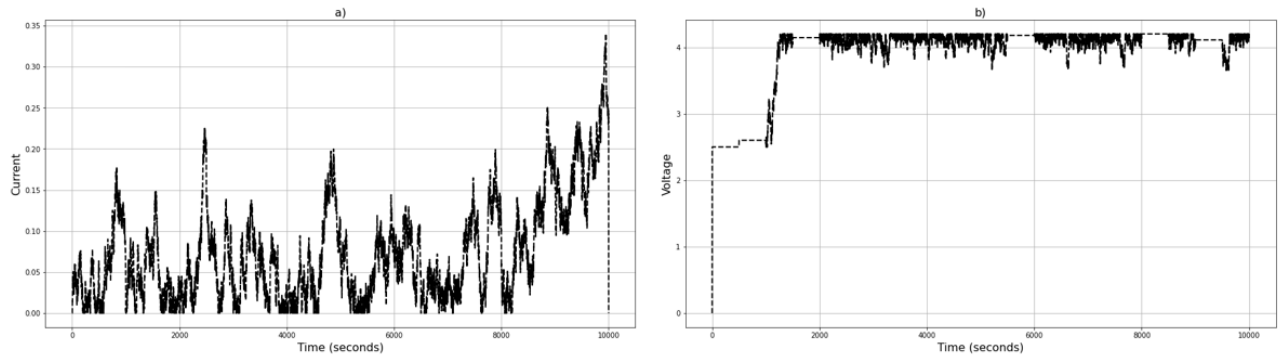


Fig. 19. Charge data of synthetic lithium-ion batteries in terms of current (a) and voltage (b)

Evaluation of SoC was also performed on the synthetic data. The obtained results are indicated in the figure below.

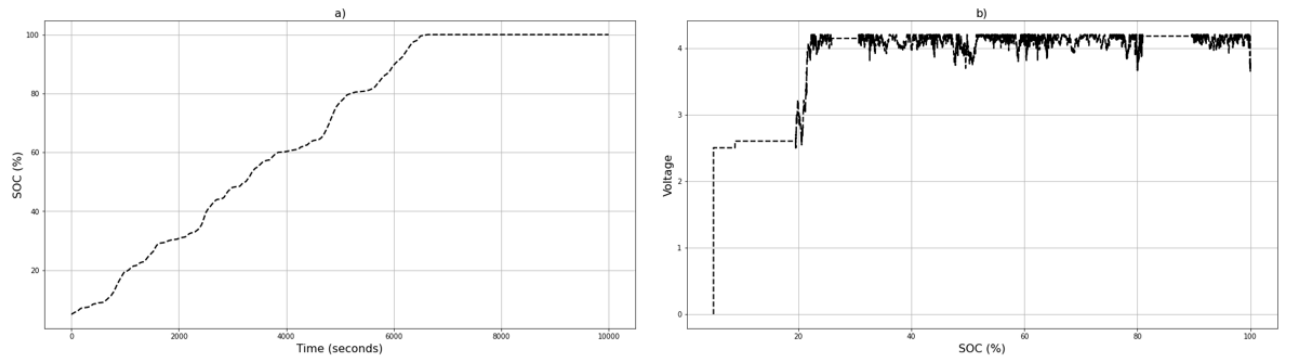


Fig. 20. Calculated SoC against time (a) and voltage against SoC (b)

The resulting SoC value ranges from 0% to 100%, which is typical for all charging data. Given the obtained results, it was decided not to carry out the SoH assessment, as there is simply insufficient data for this, and it would not reflect reality.

In the further stage of the research, the relationship between the data set variables SoC, voltage, current, and temperature was determined. The variable capacity is not included because it was used in the SoC evaluation. Pearson's correlation coefficient was calculated to determine the relationship between these variables.

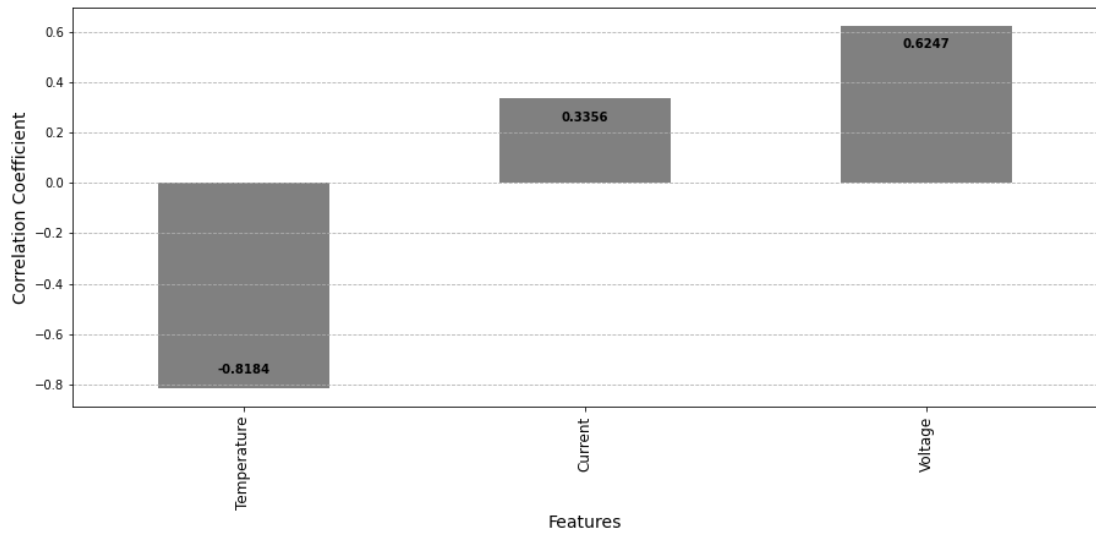


Fig. 21. Pearson correlation between target variable SoC and features

A strong negative correlation of -0.8184 was recorded between SoC and the temperature variable. This result indicates that when one variable increases, the other decreases. A weak positive correlation of 0.3356 was recorded between SoC and current, while a moderately strong correlation of 0.6247 was recorded between SoC and voltage. Both the weak positive and moderately positive correlations indicate that the changing trend of these variables is similar to that of SoC, meaning they change in the same direction.

A separate analysis of the evaluated SoC variable was performed. The analysed characteristics include the mean, standard deviation, minimum and maximum values, quartiles, skewness, and kurtosis.

Table 12. Statistical characteristics and their values

Characteristic	Calculated value
mean	67.376806
std	32.350090
25%	39.322727
50%	77.124959
75%	100.000000
Skewness	-0.4737046681777111
Kurtosis	-1.212478626246607

It was observed that about 25% of the values are below 39.32, while about 75% of the SoC data is at or below 100. According to the calculated standard deviation, it can be said that the data is diverse. The skewness indicates that the data is concentrated on the right side of the mean, and the kurtosis suggests that most of the data has high values. It should be noted that the kurtosis results are self-explanatory because the SoC values are rising.

The results of variable current preprocessing are shown in the figure below. Graph (a) shows the KF results, graph (b) shows the autoencoder results, graphs (c) to (e) inclusive show different wavelet transform results, and graph (f) shows the SGF results.

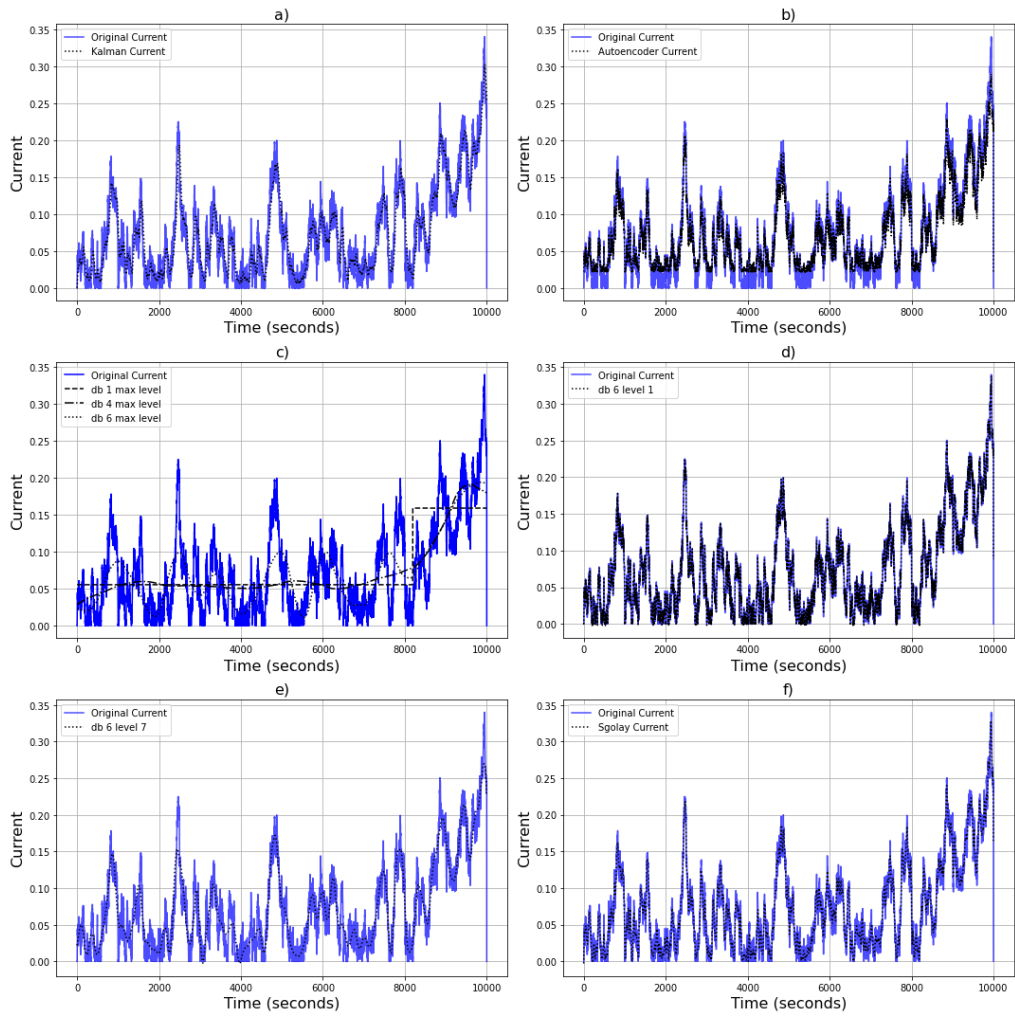


Fig. 22. Real current against the preprocessed current

Since the voltage variable is not as dynamic and stochastic as the current variable, only KF and SGF were used for data processing. In the figure below, **(a)** represents the results with the SGF and **(b)** represents the results with the Kalman filter.

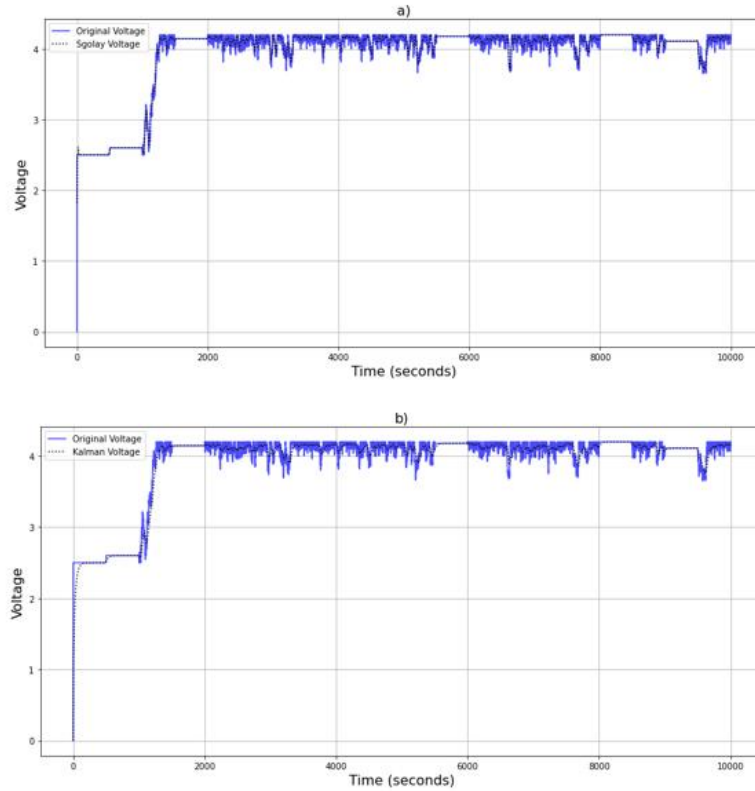


Fig. 23. Real voltage against preprocessed current

Variance Inflation Factor (VIF) was calculated to determine multicollinearity between the variables used to estimate SoC. Both raw and processed data were monitored to investigate whether variable preprocessing can affect the data.

Table 13. Variance Inflation Factor (VIF) for synthetic data results

Feature	Original data VIF	KF preprocessed VIF	SLDA preprocessed VIF	DWT preprocessed VIF
Constant	93.710874	89.078733	94.815059	94.468679
Current	1.142527	1.027039	1.023410	1.024226
Temperature	1.144269	1.159669	1.145891	1.163349
Voltage	1.023531	1.156808	1.147719	1.163449

After analysing the obtained results, multicollinearity between the variables was not determined.

3.1.2. Real data case

In the case of real data, discharge and charge data of a lithium iron phosphate battery were used. A detailed description of the battery is given in the table below.

Table 14. Research battery description

Attribute	Real information
Cell capacity (Ah)	100
Cell chemistry	LiFePO4
Max peak charge rate (C)	3
Max peak discharge rate (C)	10
Max absolute cell voltage (V)	4
Min absolute cell voltage (V)	2.5
Max nominal voltage (V)	4

This battery cell capacity in Ah is equal to 100. Cell capacity can also be calculated by using this equation:

$$\begin{aligned} \text{Max Capacity (Ah)} &= \max \left(\sum_{i=1}^n \left(\text{Current}_i \times \frac{\text{Time Interval}_i}{3600} \right) \right) \\ &= 99.59137065069768 \approx 100 \end{aligned} \quad (28)$$

There are a total of three variables in the dataset. The variable timestamp indicates the time of the test, as well as the variables voltage and current. Current and voltage are shown in the figure below.

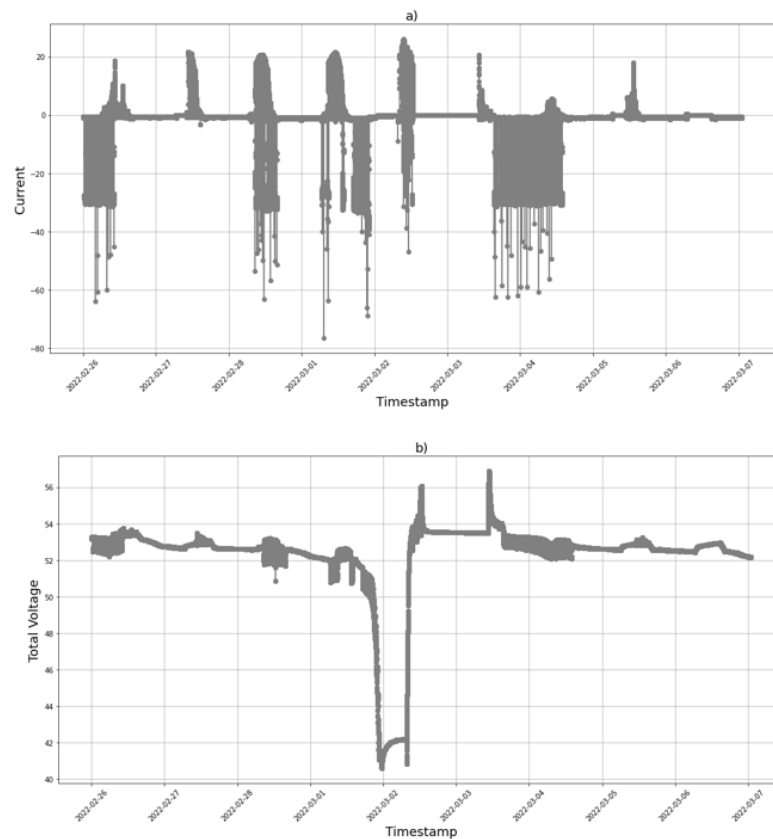


Fig. 24. Real battery data current (a) and voltage (b)

According to the graphs, it is possible to identify that the battery has been discharged and charged several times. The testing period spans from 2022-02-26 to 2022-03-07, with time also displayed in hours, minutes, and seconds. The dataset contains a total of 1,048,575 data points. Three additional variables were calculated. The calculated variable, time difference in hours, indicates the difference between data points in hours:

$$\Delta t_i = \frac{timestamp_i - timestamp_{i-1}}{3600}, \quad (29)$$

there $timestamp_i$ is variable timestamp value at the i -th data point.

The calculated variable Delta Q shows the change in charge per time interval. Variable equation:

$$Delta\ Q = I_i - \Delta t_i, \quad (30)$$

There I_i is for the total current at the time and Δt_i is for time difference in hours.

Another calculated variable is charge Ah which shows charge values in Ah and can be expressed as an equation:

$$Charge\ Ah = I_i \times \Delta t_i. \quad (31)$$

SoC evaluation was performed using the newly calculated and original dataset variables. According to the variation of the SoC, it is observed that the battery was partially discharged, then repeatedly partially discharged. This pattern indicates that the SoH will have just two points.

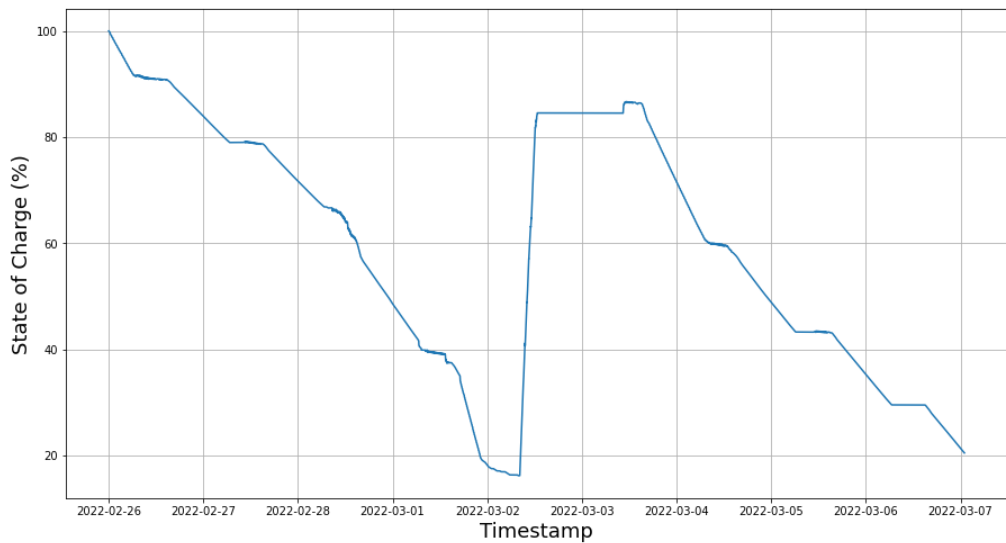


Fig. 25. Calculated real battery SoC over time

Current and voltage preprocessing was performed. Graph (a) shows the autoencoder results, graph (b) shows the KF results, graph (c) shows the SGF results, and graph (d) shows the DWT results.

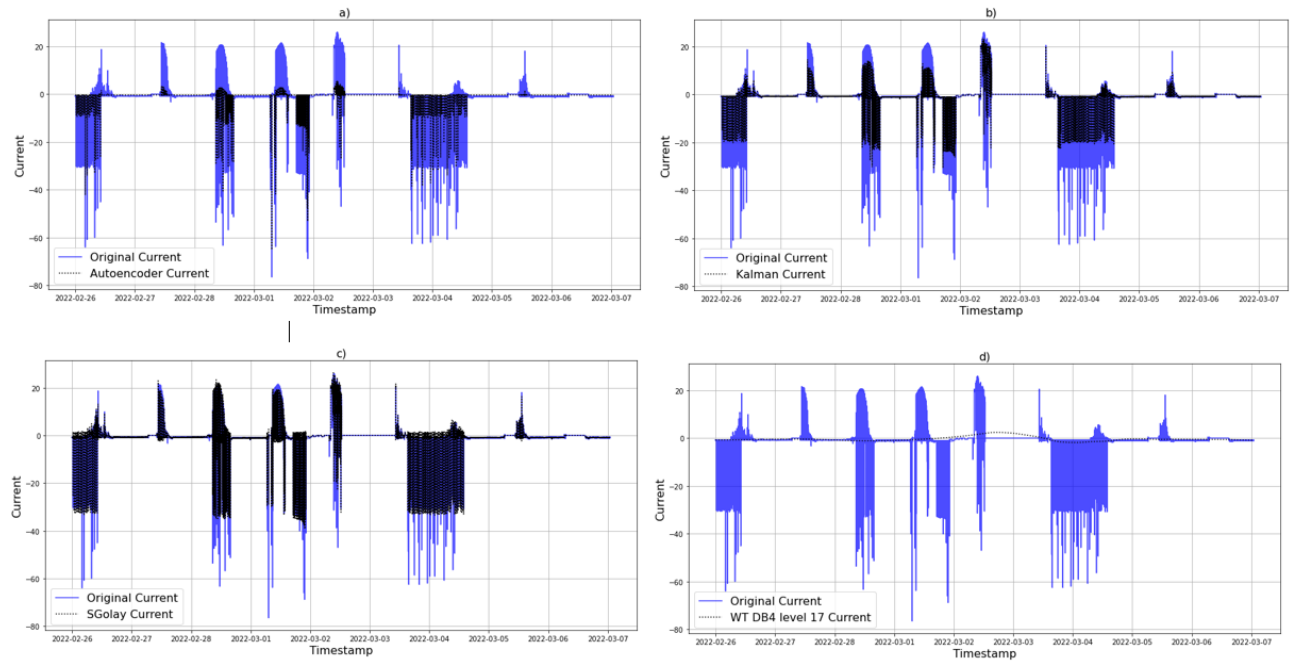


Fig. 26. Real current against the preprocessed current

Similar preprocessing was performed for voltage. Graph (a) shows the autoencoder results, graph (b) shows the KF results, graph (c) shows the SGF results and graph (d) shows the DWT results.

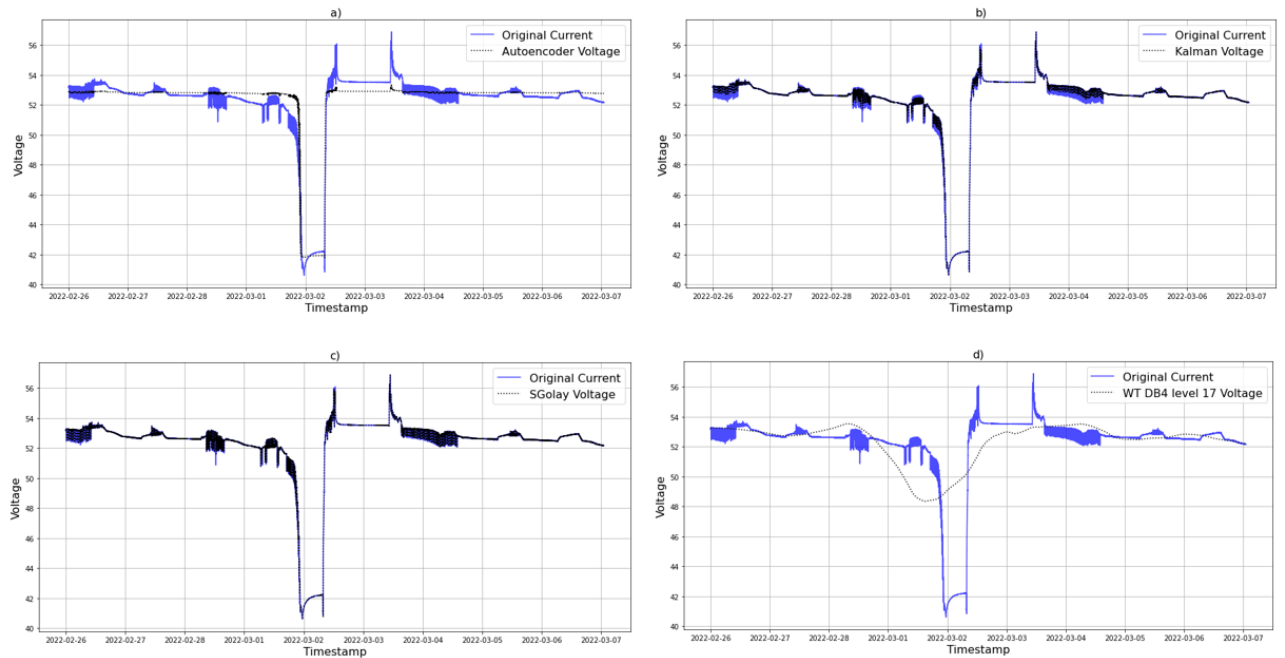


Fig. 27. Real voltage against preprocessed current

In addition, the relationship between the variables in terms of SoC was assessed using the Pearson correlation coefficient. Between SoC and the variables charge Ah, delta Q, time difference in hours, and current, very low coefficient values close to zero were obtained. These results indicate a very weak or even non-existent relationship between SoC and these variables. The relationship between

SoC and voltage, with a coefficient of 0.54, indicates a moderate positive correlation, meaning the values of SoC and voltage change in similar directions.

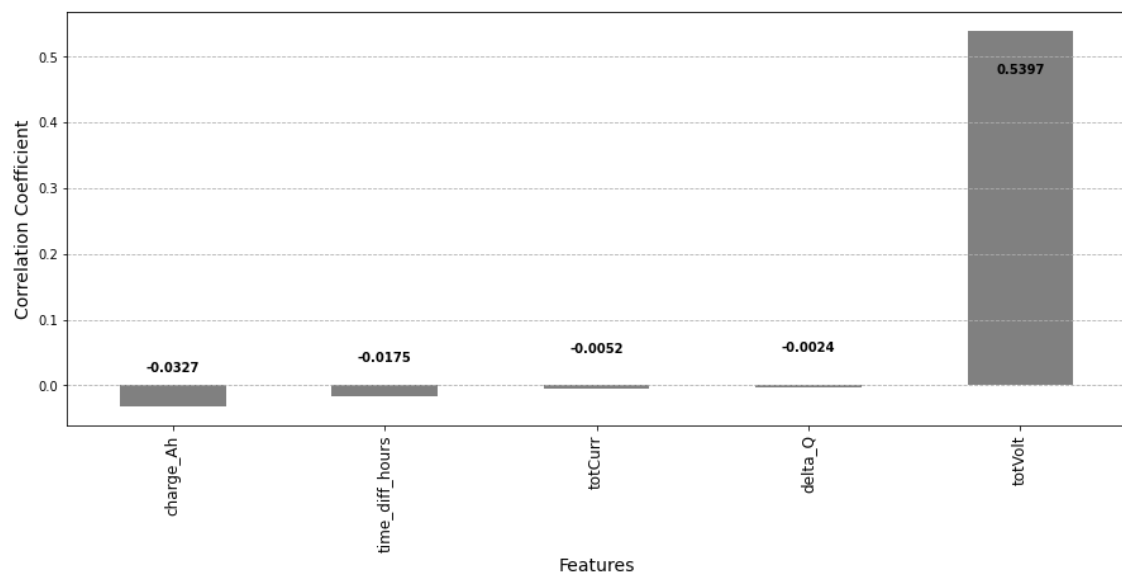


Fig. 28. Pearson correlation between target variable SoC and features

For real data as well as synthetic data, the Variance Inflation Factor (VIF) was performed to determine multicollinearity.

Table 15. Variance Inflation Factor (VIF) for real battery data results

Feature	VIF original	VIF Autoencoder
Consta	527.114731	562.559003
Current	4.702611	2.482382
Voltage	1.009121	1.003387
Time difference in hours	1.059606	1.062455
Change in charge	4.209507	3.771647
The amount of charge in each time interval	2.154814	1.918460

A highly collinear constant feature was obtained. In the case of voltage and time difference in hours, there is no multicollinearity, but in the case of current and charge Ah, moderate multicollinearity was recorded.

3.2. Model training and testing results

3.2.1. Synthetic data State of Charge assessment results

In the case of neural networks, several different scenarios were tested. First, the initial RNN, LSTM, and GRU models were developed. The number of units and the learning rate were optimised using Bayesian optimisation. A total of 144 tests were performed on neural networks. Of these, 96 were performed without Bayesian optimisation, and 48 tests were performed with Bayesian optimisation. The table below shows the results of the initial RNN model testing. The results for LSTM and GRU are shown in the appendices.

Table 16. Initial RNN model testing results with different numbers of epochs and model input data, synthetic dataset case

Model input data	Number of epochs	Train dataset results			Test dataset results		
		MSE	TDE	R^2	MSE	TDE	R^2
Original voltage and current variables with temperature, initial RNN model	100	0.0016	0.0153	0.9858	0.0018	0.0170	0.9852
	70	0.0018	0.0163	0.9843	0.0019	0.0180	0.9837
	50	0.0020	0.0190	0.9827	0.0021	0.0210	0.9823
	10	0.0032	0.0268	0.9720	0.0032	0.0313	0.9725
Original voltage and current variables without temperature variable, initial RNN model	100	0.0588	0.1356	0.4901	0.0598	0.1375	0.4938
	70	0.0596	0.1510	0.4833	0.0603	0.1532	0.4892
	50	0.0603	0.9309	0.4779	0.0608	0.5756	0.4853
	10	0.0615	0.1444	0.4668	0.0615	0.1433	0.4790
Kalman voltage and current variables with temperature, initial RNN model	100	0.0540	0.1165	0.5321	0.0555	0.1206	0.5301
	70	0.0546	0.1177	0.5271	0.0559	0.1211	0.5268
	50	0.0557	0.1201	0.5172	0.0566	0.1240	0.5209
	10	0.0587	0.1846	0.4916	0.0591	0.2263	0.4992
Kalman voltage and current variables without temperature variable, initial RNN model	100	0.0540	0.1165	0.5321	0.0555	0.1206	0.5301
	70	0.0546	0.1177	0.5271	0.0559	0.1211	0.5268
	50	0.0557	0.1201	0.5172	0.0566	0.1240	0.5209
	10	0.0587	0.1846	0.4916	0.0591	0.2263	0.4992
Autoencoder voltage and current variables with temperature, initial RNN model	100	0.0016	0.0130	0.9859	0.0018	0.0142	0.9850
	70	0.0018	0.0653	0.9840	0.0020	0.0578	0.9833
	50	0.0019	0.0153	0.9831	0.0021	0.0171	0.9826
	10	0.0031	0.0251	0.9728	0.0032	0.0294	0.9731
Autoencoder voltage and current variables without temperature variable, initial RNN model	100	0.0594	0.9496	0.4855	0.0601	0.6470	0.4911
	70	0.0601	0.1416	0.4796	0.0607	0.1408	0.4863
	50	0.0605	0.9737	0.4756	0.0611	0.6283	0.4825
	10	0.0616	0.1477	0.4662	0.0616	0.1451	0.4784
DWT DB 6 level 7 voltage and current variables with temperature, initial RNN model	100	0.0016	0.0159	0.9865	0.0018	0.0185	0.9847
	70	0.0017	0.0164	0.9851	0.0019	0.0186	0.9837
	50	0.0019	0.0175	0.9834	0.0021	0.0209	0.9818
	10	0.0034	0.0292	0.9707	0.0034	0.0344	0.9708
DWT DB 6 level 7 voltage and current variables without temperature variable, initial RNN model	100	0.0582	0.1354	0.4959	0.0582	0.1354	0.5071
	70	0.0587	0.1328	0.4909	0.0588	0.1328	0.5018
	50	0.0594	0.8028	0.4856	0.0595	0.5277	0.4962
	10	0.0609	0.1360	0.4720	0.0604	0.1351	0.4888

The lowest MSE error and the highest R^2 were obtained with DWT DB 6 level 7 voltage and current variables with temperature, initial RNN model was tested with 100 epochs. The application of Tweedie deviance error was not appropriate because its values were obtained even with the lowest MSE. This test indicates the importance of choosing appropriate metrics for model testing. The worst test results were obtained after preprocessing the data with a KF. Increasing the number of epochs helps improve model performance in all cases, as well as using the additional temperature variable instead of only using voltage and current.

Additionally, MSE error was observed as the number of epochs increased. Graph (a) shows the MSE graph of the model with the lowest MSE error and the highest R^2 , while graph (b) shows the case of the model with the highest MSE error and the lowest R^2 .

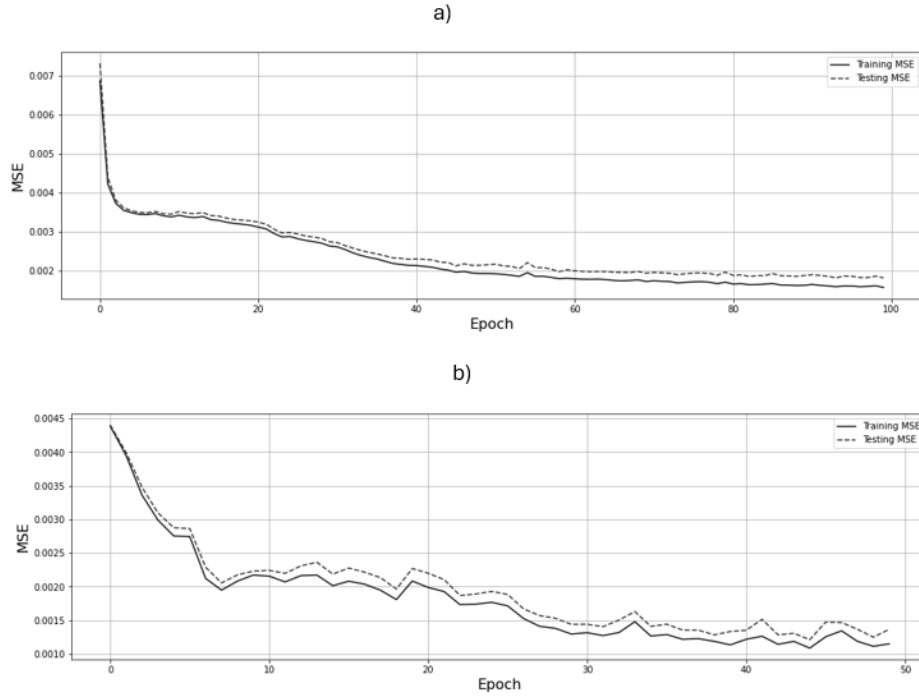


Fig. 29. RNN model training with synthetic data MSE values per epoch, best results (a) and worst results (b)

In both cases, the MSE decreases with increasing epochs. In the case of the best model, this change is more stable, while in the case of the worst model, certain jumps are recorded. Overall, both graphs indicate that the model learns well from new data. Applying Bayesian optimisation to the improvement of neural network models managed to increase R^2 values in all cases, except for KF preprocessed voltage and current with temperature, using 100 epochs.

Table 17. RNN model with Bayesian optimisation testing results with different number of epochs and model input data, synthetic dataset case

Model input data	Number of epochs	Train dataset results			Test dataset results			Initial model performance improvement
		MSE	TDE	R^2	MSE	TDE	R^2	
Original voltage and current variables with temperature	100	0.0025	1.3165	0.9786	0.0026	1.0115	0.9783	Yes
	70	0.0025	2.1496	0.9785	0.0026	1.9668	0.9781	Yes
	50	0.0026	3.8326	0.9775	0.0027	3.8414	0.9768	Yes
	10	0.0027	0.0212	0.9768	0.0028	0.0242	0.9763	Yes
Kalman voltage and current variables with temperature	100	0.0602	0.1372	0.4786	0.0606	0.1405	0.4866	No
	70	0.4855	0.1373	0.4776	0.0608	0.1404	0.4855	No
	50	0.0605	0.1378	0.4761	0.0609	0.1410	0.4840	No
	10	0.0696	0.1622	0.3969	0.0702	0.1665	0.4057	No
Autoencoder voltage and current variables with temperature	100	0.0021	0.0167	0.9822	0.0021	0.0186	0.9820	Yes
	70	0.0022	0.1847	0.9812	0.0023	0.2384	0.9809	Yes
	50	0.0022	0.0133	0.9812	0.0023	0.0180	0.9809	Yes
	10	0.0022	3.2703	0.9814	0.0023	3.2539	0.9807	Yes
DWT DB 6 level 7 voltage and current variables with temperature	100	0.0010	0.0109	0.9914	0.0012	0.0138	0.9900	Yes
	70	0.0011	0.0116	0.9904	0.0013	0.0148	0.9891	Yes
	50	0.0011	0.0118	0.9900	0.0014	0.0156	0.9884	Yes
	10	0.0022	0.0164	0.9812	0.0022	0.0191	0.9811	Yes

However, model performance improvement with Bayesian optimisation was evaluated in terms of R^2 rather than MSE, as it was not possible to reduce the MSE in all cases

For the best model, the MSE decreases with slight spikes as the number of epochs increases, indicating that the model learns well with new data. In the case of the worst model, the MSE decreases at the very beginning and then starts to increase sharply. This may indicate that the learning rate optimised by Bayesian optimisation is incorrect.

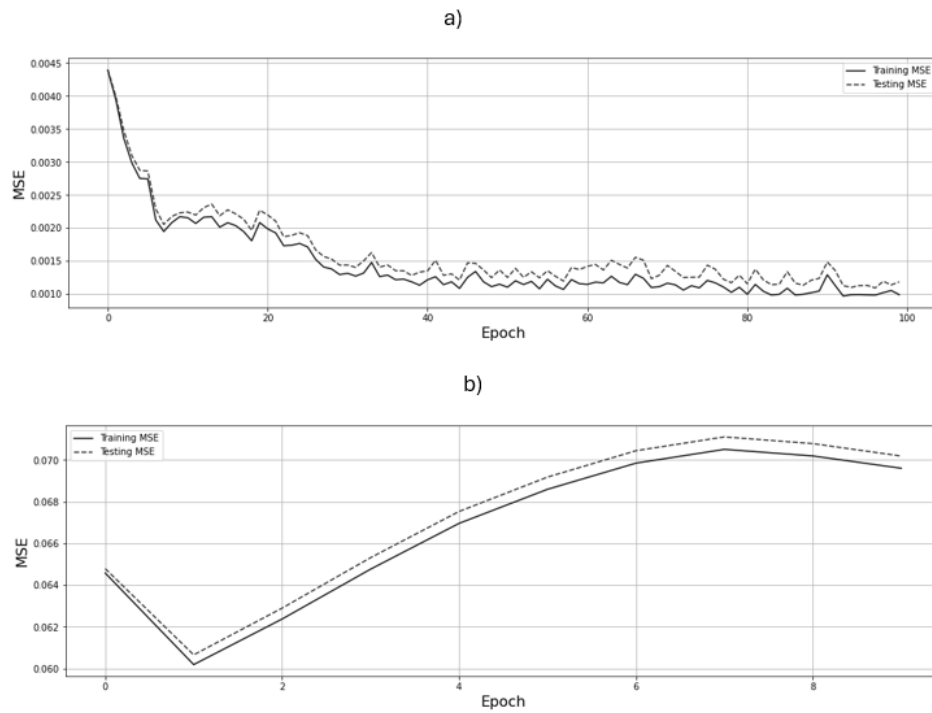


Fig. 30. RNN model with Bayesian optimisation training with synthetic data MSE values per epoch, best results (a) and worst results (b)

Best performing models were compared in terms of training time in seconds and RMSE error.

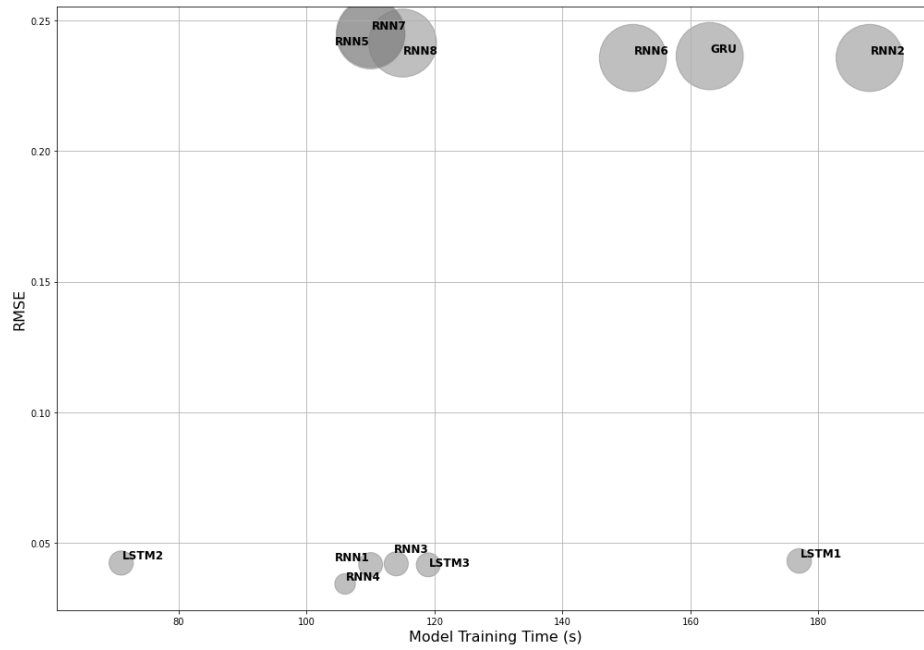


Fig. 31. Comparison of best-performing models by training time in seconds and RMSE error, synthetic dataset case

In the figure above, RNN1 is RNN initial model with initial data and temperature variable, RNN2 is initial RNN model with Kalman filter preprocessed data and temperature variable, RNN3 is initial RNN model with autoencoder preprocessed data and temperature variable, RNN4 is initial RNN model with Wavelet transform preprocessed data with temperature variable, RNN5 is initial RNN model with initial data without temperature variable, RNN6 is initial RNN model with Kalman filter preprocessed data without temperature variable, RNN7 is initial RNN model with autoregression preprocessed data without temperature variable, RNN8 is initial RNN model with Wavelet transform preprocessed data without temperature variable, LSTM1 is Bayesian optimised LSTM model with initial data and temperature variable, LSTM2 is Bayesian optimised LSTM model with autoregression preprocessed data without temperature variable, LSTM3 is Bayesian optimised LSTM model with Wavelet transform preprocessed data with temperature variable, and GRU is initial GRU model with Kalman filter preprocessed data with temperature variable. It can be observed that the training time for all models is quite short. The fact that there are only 10,000 thousand people in the data set can have a big impact on this. data points and also solves the regression task and makes the data easier to process. It can also be seen from the graph that it takes the shortest time to train the LSTM2 model, while the longest training time is obtained with the RNN2 model. The RMSE error of the RNN4 model is the lowest when the training time is relatively short. The highest RMSE error was obtained with RNN5, RNN7 and RNN8 models, so in the case of synthetic data, the initial RNN model with Kalman filter preprocessed data and temperature variable in terms of RMSE error and overall model training time in seconds has the best performance.

In the case of regression models in general 37 experiments were performed. The lowest RMSE and MSE error values and the highest R^2 value for regression models are obtained by applying Ridge Regression with voltage, current and temperature variables, when voltage and current variables are preprocessed with KF. It is difficult to evaluate the suitability of Ridge regression based only on alpha values, because a higher value does not necessarily indicate that the model will be the most

suitable, so the results of Ridge regression are additionally compared to the RMSE and MSE errors and R squared values of linear regression and the Kalman filter.

For ridge regression with a synthetic dataset, the optimised regularization strength values denoted as α are given in the table below.

Table 18. Ridge regression optimised alpha values for different variables, synthetic dataset case

Input variables processing	Variables (input) used			Optimised α
	Current	Voltage	Temperature	
Initial	+	+	+	0.16624716
	+	+		2.0280003
		+		0.30913675
Kalman	+	+	+	2.21035262
	+	+		5.94829519
		+		3.44456253
Autoencoder	+	+	+	0.16624716
	+	+		1.98204987
DWT db6 level 7	+	+	+	0.06047521
	+	+		1.189629

The obtained results also indicate that the introduction and use of an additional variable such as temperature helps to improve the model results. The smallest errors are obtained with the KF. Since the Kalman filter was experimentally applied only to the modelling of the voltage variable, these results indicate that it is potentially possible not to perform the SoC evaluation directly, but to evaluate the battery by modelling the voltage as well.

Table 19. ML models testing results with different model input data, synthetic dataset case

Model	Input variables processing	Variables used			Train dataset results			Test dataset results		
		Current	Voltage	Temperature	RMSE	MSE	R^2	RMSE	MSE	R^2
LR	Initial	+	+	+	0.0792	0.0063	0.9457	0.0809	0.0065	0.9446
		+	+		0.2505	0.0627	0.4564	0.2506	0.0628	0.4681
			+		0.2661	0.0708	0.3863	0.2649	0.0702	0.4057
	Kalman	+	+	+	0.0727	0.0053	0.9542	0.0735	0.0054	0.9542
		+	+		0.2454	0.0602	0.4781	0.2481	0.0615	0.4788
			+		0.2633	0.0693	0.3993	0.2632	0.0693	0.4133
	Autoencoder	+	+	+	0.0792	0.0063	0.9456	0.0809	0.0065	0.9446
		+	+		0.2503	0.0626	0.4572	0.2506	0.0628	0.4683
	WT db6 level 7	+	+	+	0.0791	0.0063	0.9458	0.0809	0.0065	0.9446
		+	+		0.2483	0.0617	0.4657	0.2479	0.0615	0.4796
Ridge Regression	Initial	+	+	+	0.0792	0.0063	0.9457	0.0809	0.0065	0.9446
		+	+		0.2505	0.0627	0.4563	0.2506	0.0628	0.4682
			+		0.2661	0.0708	0.3863	0.2649	0.0702	0.4057
	Kalman	+	+	+	0.0728	0.0053	0.9541	0.0734	0.0054	0.9543
		+	+		0.2456	0.0603	0.4773	0.2479	0.0614	0.4797
			+		0.2634	0.0694	0.3989	0.2631	0.0692	0.4136
	Autoencoder	+	+	+	0.0792	0.0063	0.9456	0.0809	0.0065	0.9446
		+	+		0.2503	0.0627	0.4571	0.2506	0.0628	0.4684
	WT db6 level 7	+	+	+	0.0791	0.0063	0.9458	0.0809	0.0065	0.9446
		+	+		0.2483	0.0617	0.4657	0.2479	0.0615	0.4796
KF	Initial		+		0.0107	>0	0.9979	0.0065	>0	0.9552

When evaluated graphically, the Ridge regression results are very different from the actual SoC values and resemble synthetic voltages in their form. The neural network results are closer to the actual SoC values, but there are some deviations. The results of the neural network shown in the figure are obtained with the original LSTM model when the data is preprocessed with DWT using voltage, current, and temperature variables, and trained with 100 epochs. Ridge regression results are not detailed because similar values were obtained in all cases.

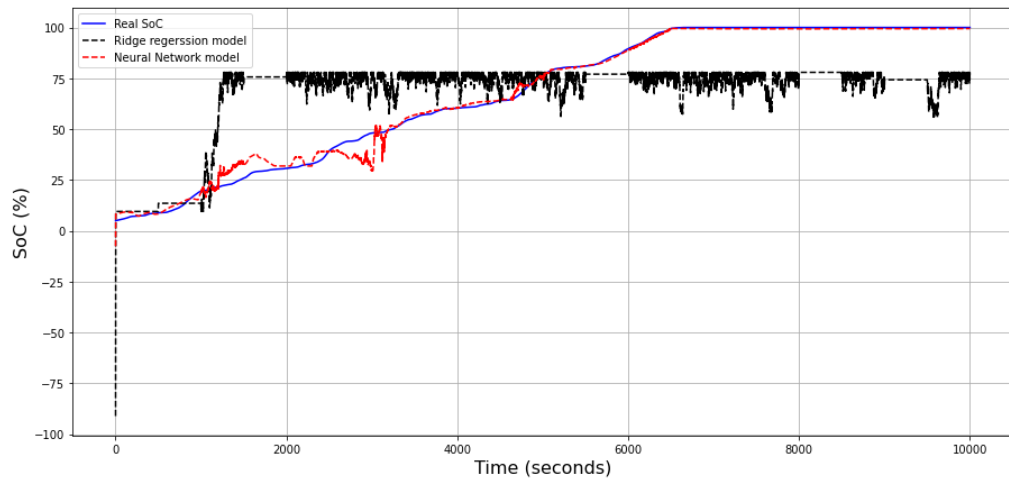


Fig. 32. Real SoC value and assessed values applying Ridge regression and neural NN, synthetic dataset case

The KF perfectly evaluates the voltage values. According to the figure below, it can be seen that the values of the Kalman filter completely match the actual voltage values.

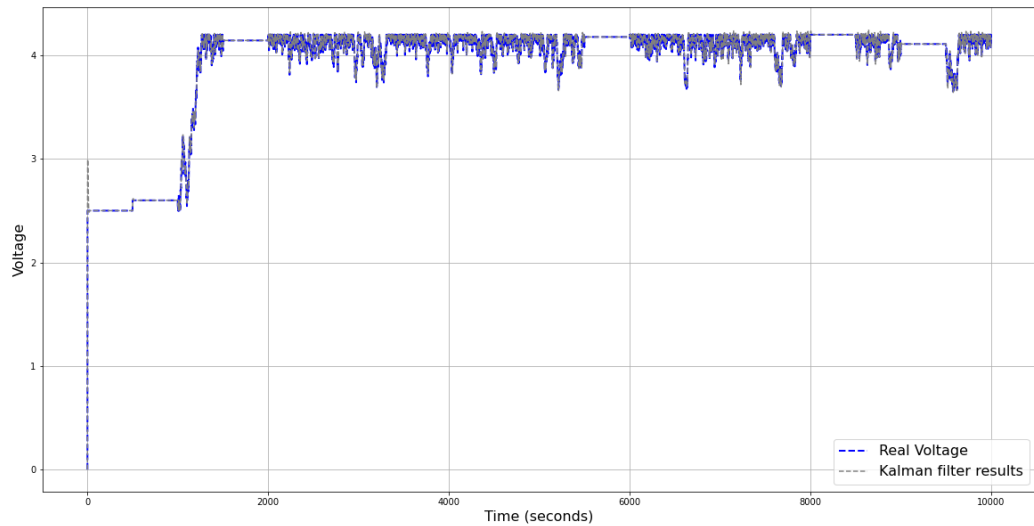


Fig. 33. KF results in comparison with real SoC values, synthetic dataset case

According to the obtained results, regression models are not suitable for SoC evaluation. Neural network models have great potential, but their improvement is needed.

3.2.2. Real data State of Charge assessment results

In the case of real data, only the initial models were trained and tested without applying Bayesian optimisation. It was decided to use only autoencoder preprocessed data to compare the results of the initial and preprocessed input data. When training neural network models with the variable's voltage, current, charge Ah, delta Q, and time difference in hours, the lowest MSE and the highest R^2 value were obtained with the RNN model using raw data as input. Analogous to synthetic data, Tweedie deviance error is not suitable for the performance analysis of these models because, the value of Tweedie deviance error is relatively high, despite low MSE and high R^2 .

Table 20. Neural network models testing with variables voltage, current, charge Ah, time difference in hours and delta Q results, real battery dataset case, 100 epochs

Model	Input variables processing	Train dataset results			Test dataset results		
		MSE	TDE	R^2	MSE	Tweedie error	R^2
RNN	Initial	0.0196	0.1240	0.7452	0.0197	0.1136	0.7433
	Autoencoder	0.0277	0.1169	0.6399	0.0278	0.1092	0.6377
LSTM	Initial	0.0218	1.7323	0.7160	0.0219	1.6738	0.7137
	Autoencoder	0.0327	0.1191	0.5743	0.0329	0.1197	0.5700
GRU	Initial	0.0220	0.0880	0.7139	0.0220	0.0913	0.7128
	Autoencoder	0.0328	0.1762	0.5729	0.0330	0.1689	0.5689

For the best model, the MSE error steadily decreases as the number of epochs increases. The training and testing values are consistent for all epochs. In the case of the worst-case model, the MSE error drops at the beginning and then shows large jumps. In the training itself, the training MSE is higher than the testing MSE, but as the number of epochs increases, it coincides with the testing values. Sharp jumps in the worst case can indicate the sensitivity of the model to certain input variables.

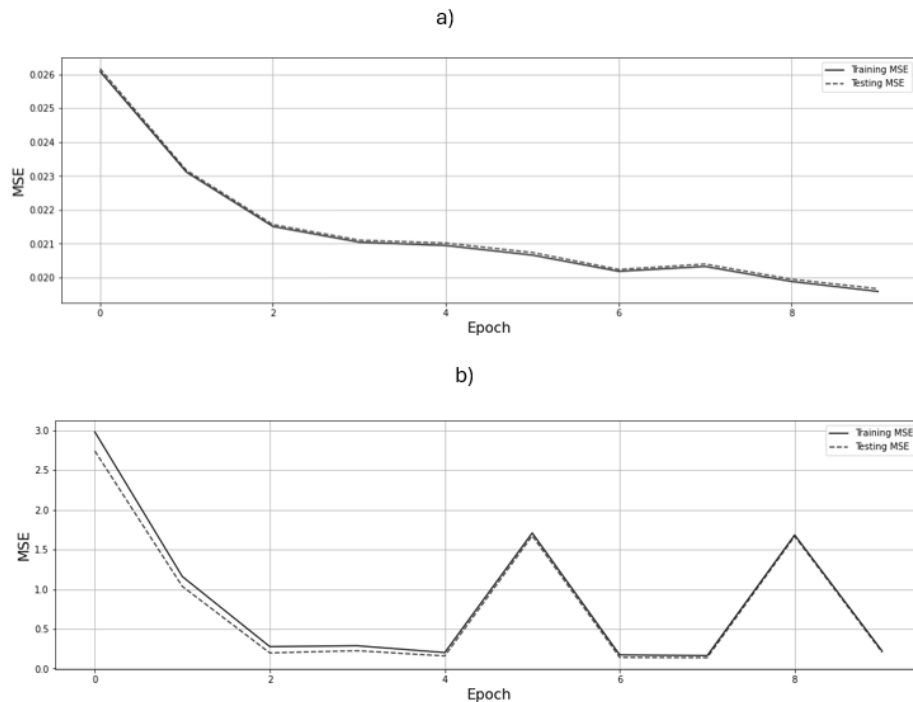


Fig. 34. Neural network models training with synthetic data MSE values per epoch, best results (a) and worst results (b), variables used: voltage, current, time difference in hours, charge Ah and delta Q

When training neural network models with only voltage current variables, the highest R^2 value was obtained with the RNN model as input given raw or raw data. In this case, the data preprocessing had a negative effect on the model results. A higher MSE error and a lower R^2 the value were obtained compared to the case when more variables were used. This confirms that the inclusion of variables other than just voltage and current can improve the model results.

Table 21. Neural network models testing with only variables voltage and current results, real battery dataset case, 100 epochs

Model	Input variables processing	Train dataset results			Test dataset results		
		MSE	TDE	R^2	MSE	TDE	R^2
RNN	Initial	0.0199	1.3507	0.7406	0.0200	1.3045	0.7387
	Autoencoder	0.0303	0.1401	0.6066	0.0303	0.1424	0.6042
LSTM	Initial	0.0202	0.0913	0.7367	0.0203	0.0906	0.7348
	Autoencoder	0.0278	0.1297	0.6381	0.0279	0.1325	0.6354
GRU	Initial	0.0213	0.0889	0.7234	0.0213	0.0922	0.7217
	Autoencoder	0.0282	0.1242	0.6329	0.0283	0.1270	0.6303

In the case of the best model, the MSE error decreases steadily at the beginning, but there is an obvious spike where the error starts to increase and then decreases again. Despite this jump, the model learns well because the training and testing values match over the entire period. In the case of the worst model, the training and testing MSE error values initially do not match but then converge. Very sharp jumps are observed throughout the period, which indicates possible overfitting.

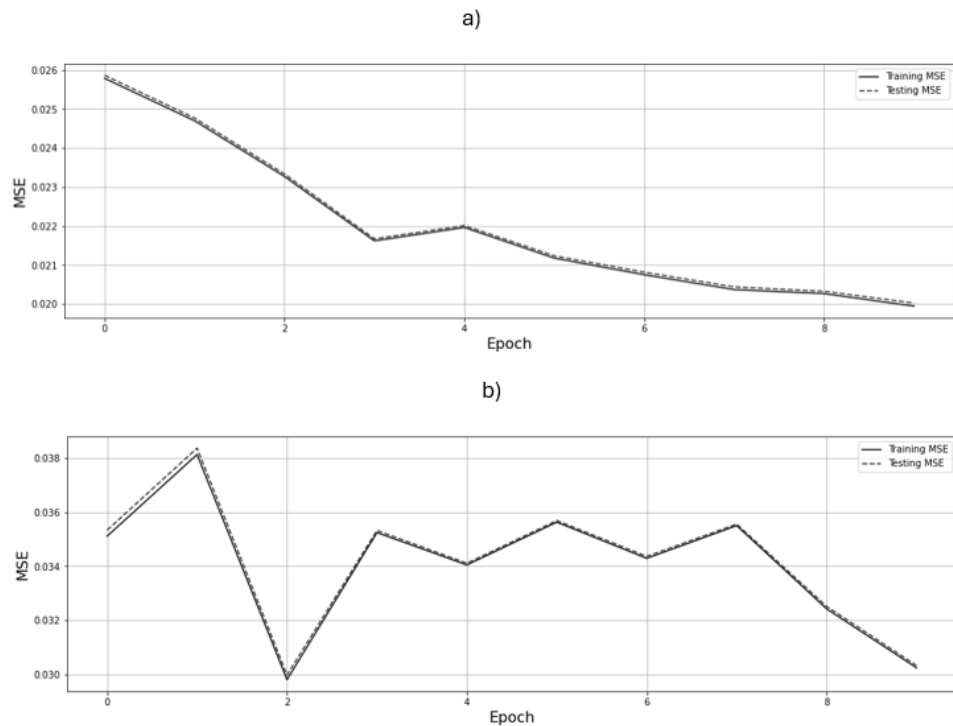


Fig. 35. Neural network models training with synthetic data MSE values per epoch, best results (a) and worst results (b), variables used: voltage and current

Additionally, the best and worst performing models were compared in terms of training time in seconds and RMSE error.

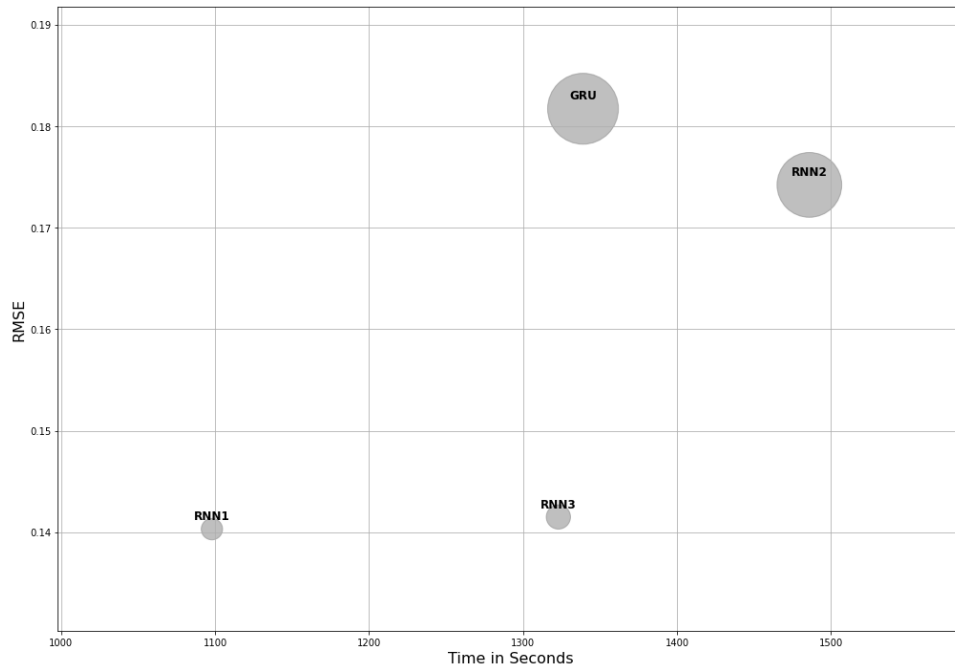


Fig. 36. Comparison of best and worst performing models by training time in seconds and RMSE error, real battery dataset case

In the figure above, RNN1 indicates RNN model with multiple variables with initial data, RNN2 indicates RNN model with autoencoder preprocessed data with current and voltage variables only, RNN3 indicates RNN model with initial data with current and voltage variables only, GRU indicates GRU model with autoencoder preprocessed data with multiple variables. Models RNN1 and RNN3 are considered to be the best-performing ones. It has been observed that the RNN1 model has the shortest training time and the lowest RMSE error, while the RNN2 model has the longest training time and the GRU model has the highest RMSE error.

Graphically, for one of the worst models, the values do not match the true values at all. The bouncing values obtained by the neural network do not reflect the real data.

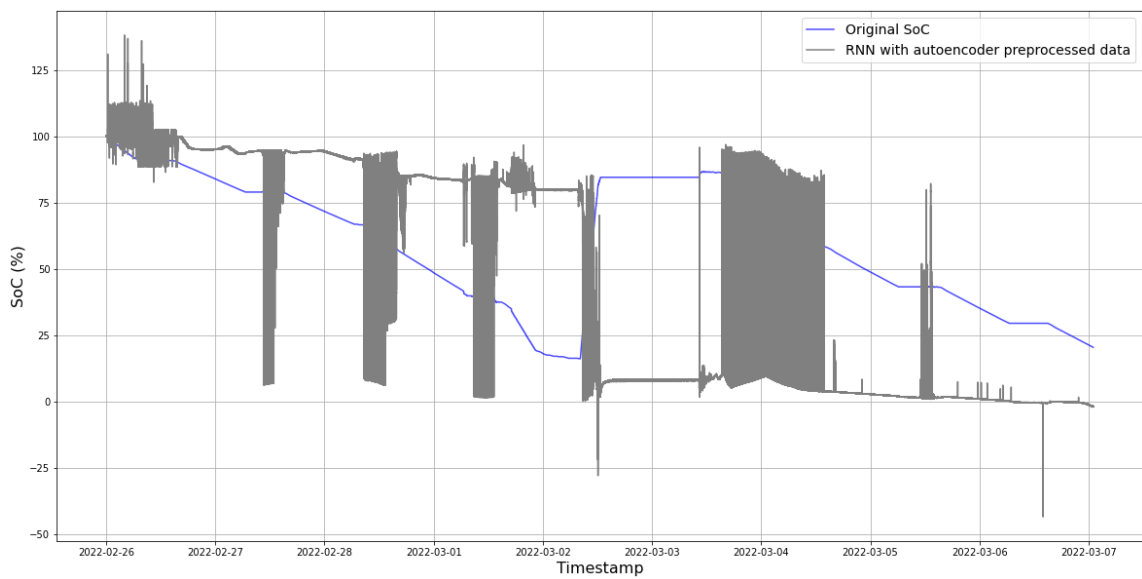


Fig. 37. Neural network models training with real dataset worst performing case

The same was obtained with the best-performing neural network model, although the results are less concentrated. This shows that the developed neural network models are suitable for synthetic data, but for real data, improvements are needed by focusing less on data processing and more on neural network architecture.

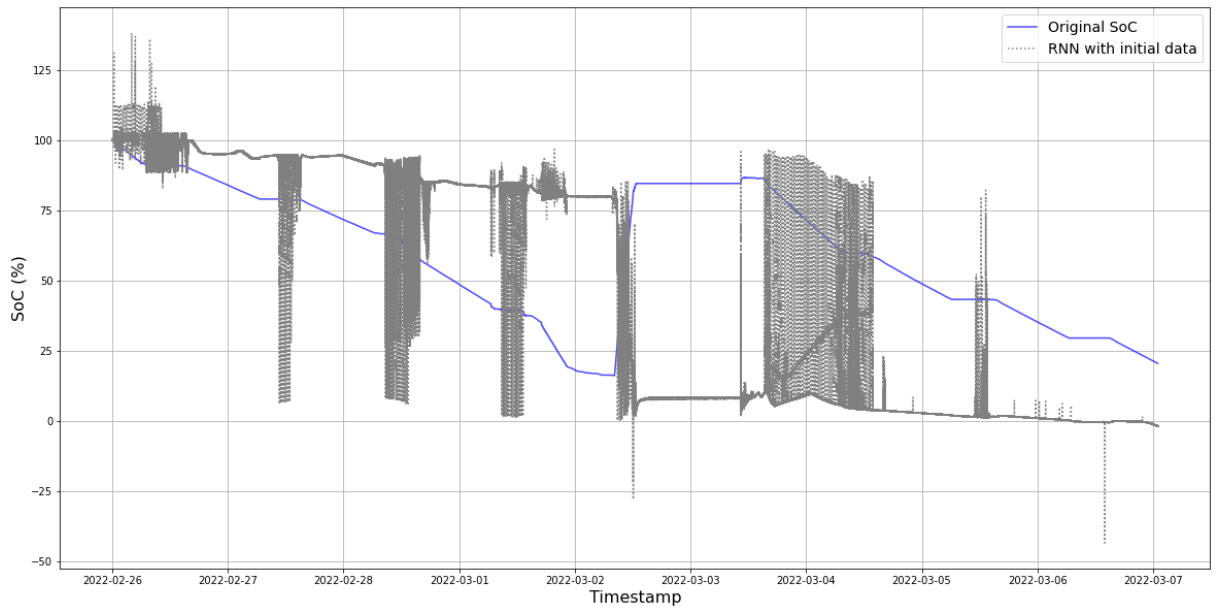


Fig. 38. Neural network models training with real dataset best performing case

Ridge regression α for all real dataset cases was obtained with the same value. Only autoencoder preprocessed data with current and voltage variables have the highest values.

Table 22. Ridge regression optimised α values for different input variables, real dataset case

Input variables processing	Variables used		Optimised α
	Current	Voltage	
Initial	+	+	0.001
		+	0.001
Autoencoder	+	+	9.72632433
		+	0.001

It can be seen that the results obtained in the cases of Linear Regression and Ridge Regression completely match. This may be because the estimated value of α is too small or because there is no obvious multicollinearity in the data. The lowest RMSE and MSE errors and the highest Rs squared value are obtained with both Linear regression and Ridge regression models when voltage and current initial variables are used.

Table 23. ML models testing results with different model input data, real battery dataset case

Model	Input variables processing	Variables used		Train dataset results			Test dataset results		
		Current	Voltage	RMSE	MSE	R^2	RMSE	MSE	R^2
Linear Regression	Initial	+	+	0.2330	0.0543	0.2943	0.2331	0.0543	0.2908
			+	0.2333	0.0545	0.2919	0.2335	0.0545	0.2887
	Autoencoder	+	+	0.2525	0.0637	0.1711	0.2527	0.0638	0.1670
			+	0.2525	0.0638	0.1707	0.2527	0.0639	0.1666
Ridge Regression	Initial	+	+	0.2330	0.0543	0.2943	0.2331	0.0543	0.2908
			+	0.2333	0.0545	0.2919	0.2335	0.0545	0.2887
	Autoencoder	+	+	0.2525	0.0637	0.1711	0.2527	0.0638	0.1670
			+	0.2525	0.0638	0.1707	0.2527	0.0639	0.1666
Kalman filter	Initial		+	0.010740	0.000115	0.99998	0.006481	0.000042	0.99902

Analysing the results, the graphically obtained results also differ significantly from the real SoC values, although they are more generous than in the case of neural networks.

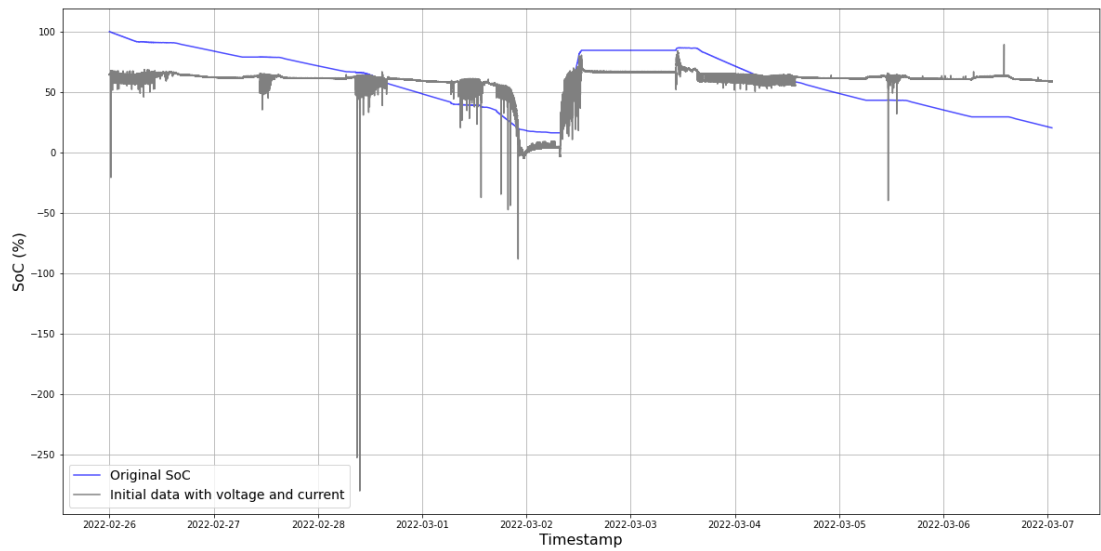


Fig. 39. ML models training with real battery dataset worst performing case

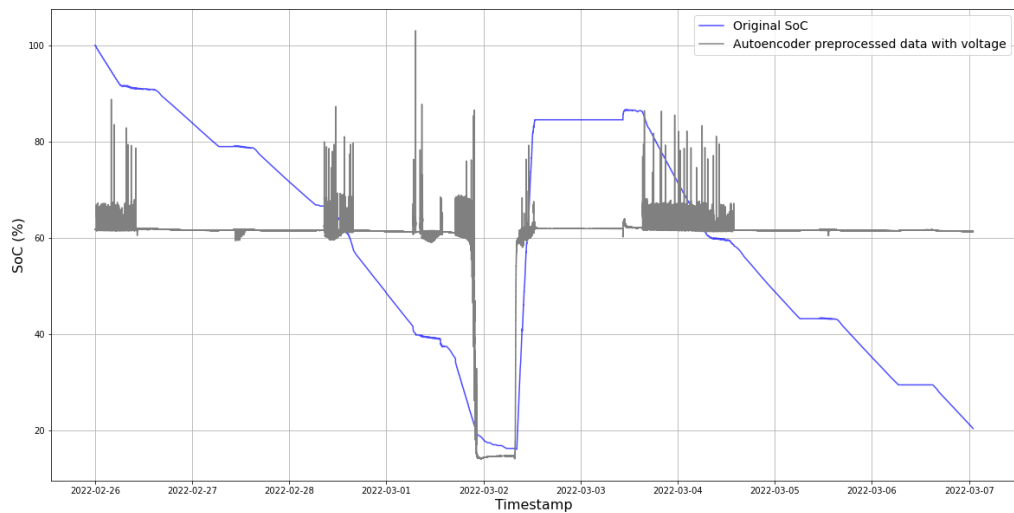


Fig. 40. ML models training with real battery dataset best performing case

For both the best and worst models, the results show spikes in certain places where this is not the case in the real data. Nevertheless, some patterns can be discerned in the data obtained by the regression models that resemble the real data, such as a visible band in the data that can identify the discharge.

In contrast to the cases of neural networks or regression models, applying the KF to the voltage variable gives results analogous to real data.

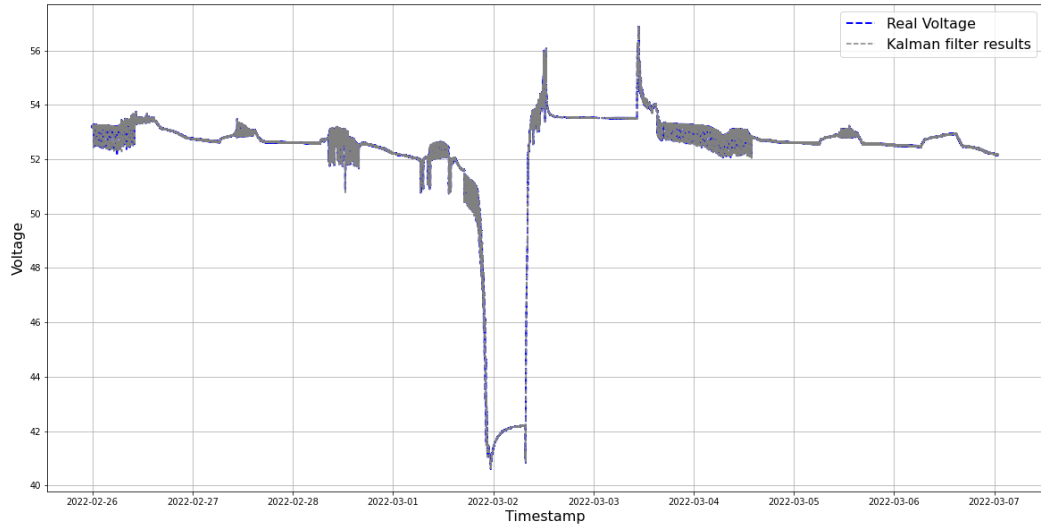


Fig. 41 . KF results in a comparison with real SoC values, real battery dataset case

In the figure above, it can be seen that the voltage obtained by the Kalman filter completely reproduces the real voltage data.

3.2.3. Real data State of Health assessment

According to the calculated SoC of the real data, it was found that the highest SoCs occur at two points. At one point, it is equal to 100%, and at the other, it is equal to 86.44%. The SoH calculated according to Formula 12 is 100% at the first point and 86.44% at the second point. Plotting the curve between these points results in the graph shown in the figure below.

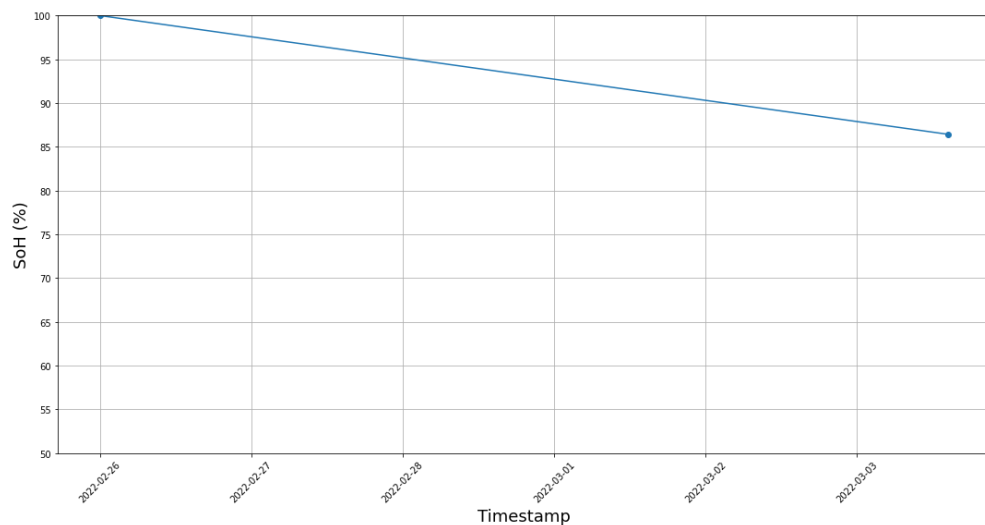


Fig. 42. Obtained SoH using real battery data

In addition, SoH assessment was performed using the empirical degradation model (see Equation 13). For SoH assessment with an empirical degradation model, the values of the already evaluated SoC were used. In this case, Equation 13 was rewritten as:

$$h = k_1 \left(\frac{\sum (\Delta SoCAh_t \times totVolt_t \times 1_{\{\Delta SoCAh_t < 0\}})}{\bar{X}(totVolt)} \right) + k_2 e^{\alpha \left(\frac{\sum (\Delta SoCAh_t \times totVolt_t \times 1_{\{\Delta SoCAh_t < 0\}})}{\bar{X}(totVolt)} \right)} + 1 - k_2, \quad (32)$$

where $\bar{X}(totVolt)$ is voltage mean, $\Delta SoCAh_t$ is for SoC change, $1_{\{\Delta SoCAh_t < 0\}}$ is set to take into account only discharge values.

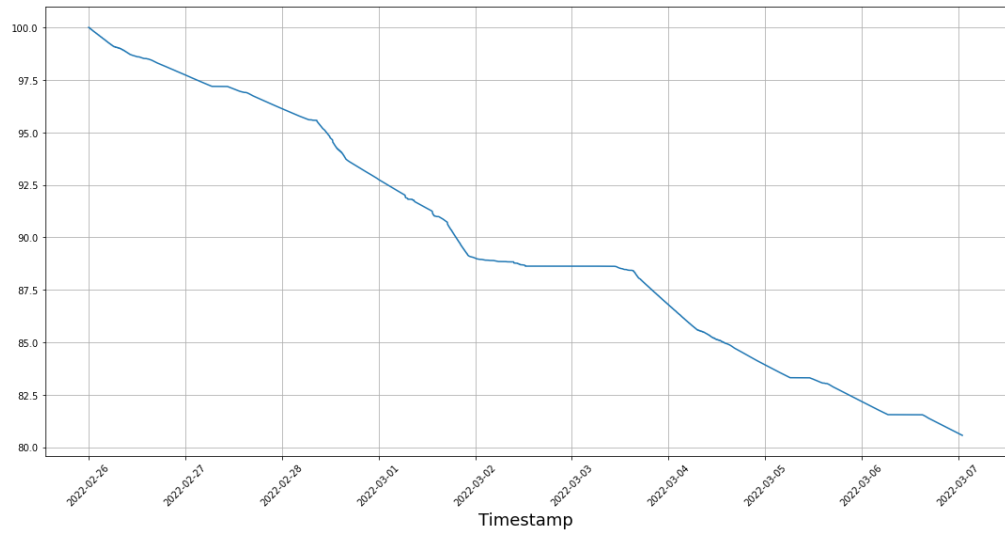


Fig. 43. SoH estimated by applying the empirical degradation model

The resulting SoH decreases similarly to the graph above, but this change is not as sharp. It occurs gradually over time and with a certain stability.

3.3. Application system

An application system was developed that allows the generation of synthetic charging data for lithium-ion batteries. The generated variables include temperature, voltage, current, charge capacity, and time in seconds. The default parameters for these variables are indicated in the figure below. The user can adjust these parameters without needing to perform any coding tasks.

Simulation Parameters

Step Interval: 500

Step Increases: 0.1

Max Capacity: 4.2

Voltage Start: 2.5

Voltage End: 4.2

Voltage Step: 0.1

Current Start: 0.0

Current Peak: 1.0

Confirm

Fig. 44. Default values for synthetic dataset generation

There are three main buttons with a pop-up menu. One button is for dataset preparation. As mentioned previously, the user can set values for variables as well as specify which variables to generate. The default number of data points is set to 10,000, but the user can adjust this as well. There is an option to print the dataset, clear all data, and export the generated dataset to an Excel file. Also, the user can upload their dataset in CSV and Excel formats.

The second button is for dataset analysis. There is an option to plot current and voltage parameters. By clicking these options within the interface environment, current or voltage charts will be generated, where the x-axis will be time in seconds and the y-axis will be current or voltage, optionally. The dataset analysis button includes not only these functions but also an option to calculate SoC for the entire dataset and create an SoC results plot, where the x-axis is time in seconds and the y-axis is SoC values in percentages.

Another button named KF includes options to split the dataset for training and testing by a ratio of 70:30, which means that 70% will be for training and the remaining 30% of the entire dataset will be for testing. KF applications are also included for this button. By clicking Apply Kalman Filter the KF will be applied within the interface environment, generating a results chart where the x-axis is time in seconds and the y-axis shows the real voltage values and the voltage values estimated with the KF.

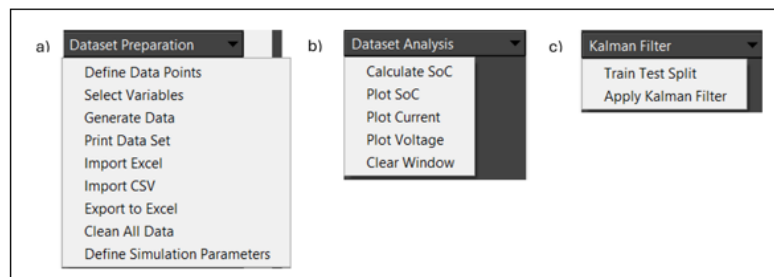


Fig. 45. Application system menu functions

Data generation is performed in the following sequence: choose the dataset preparation button → click on define simulation parameters → confirm simulation parameters → click on select variables → select the required variables → click on define datapoints → determine the required number of data points → click on generate data.

After successful completion of data generation, SoC evaluation, export of data to an Excel file, or splitting of the data set into payment and testing, the user receives a notification about the completed action. The messages are shown in the figure below. There **(a)** indicates a message about a successfully generated data set, **(b)** is a message about a successful SoC evaluation, **(c)** is a message about a successful export of the generated data to Excel format, **(d)** is a message about a successfully split data set into testing and training. Since there is an additional function for removing data, the pop-up message **(e)** is a confirmation of the operation, while **(f)** indicates that the operation was completed successfully.

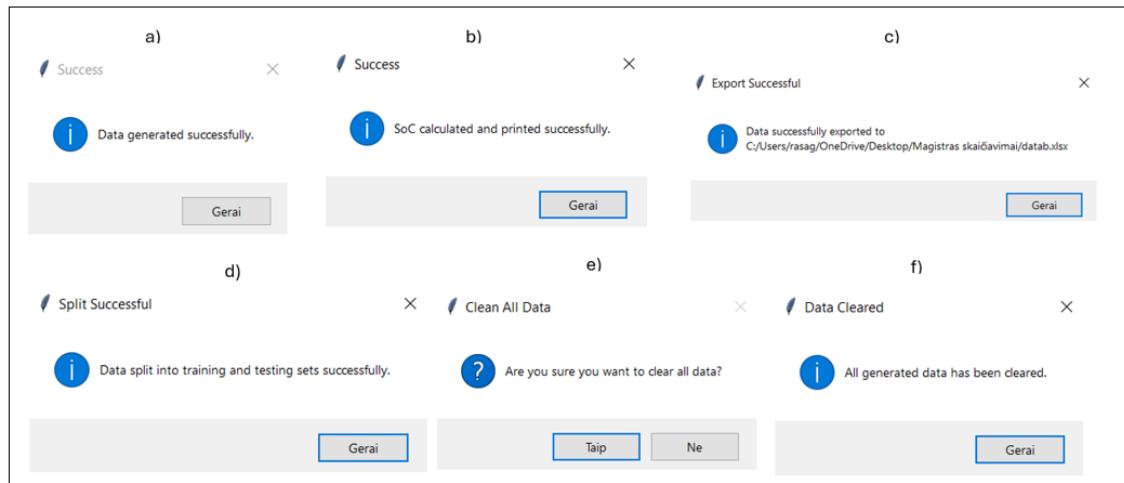


Fig. 46. Pop-up informational notifications after performing certain actions

Below is an application system application with a Kalman filter. The area marked with the number one contains the menu, the area marked with the number two contains the printed data set decomposition results and the RMSE and MSE errors of the Kalman filter evaluation on training and testing and the R squared value, while the results in the area marked with the number three are displayed graphically.

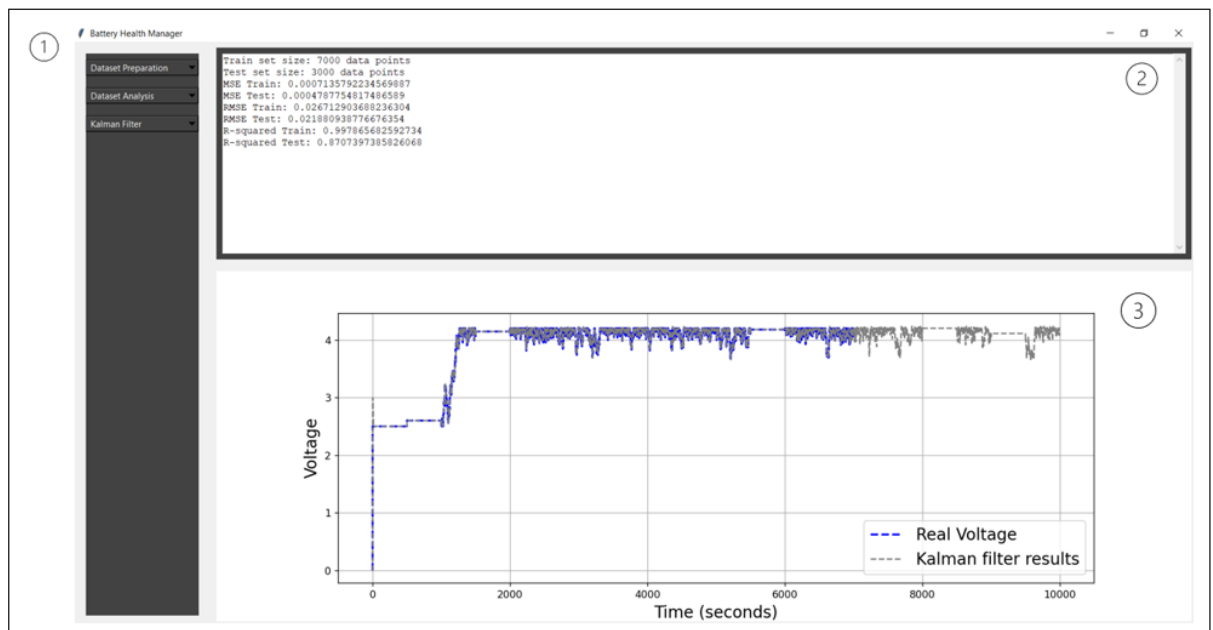


Fig. 47. Application system for the KF

KF is applied in the following sequence: choose the Kalman filter button → click on train test split → click on apply Kalman filter.

Below is an application system application for SoC assessment. The area marked with the number one contains the area, and the column marked with the number two contains the printed data set with the SoC column. Also, the application of the data points is printed as well. The area marked with number three is for the SoC results chart.

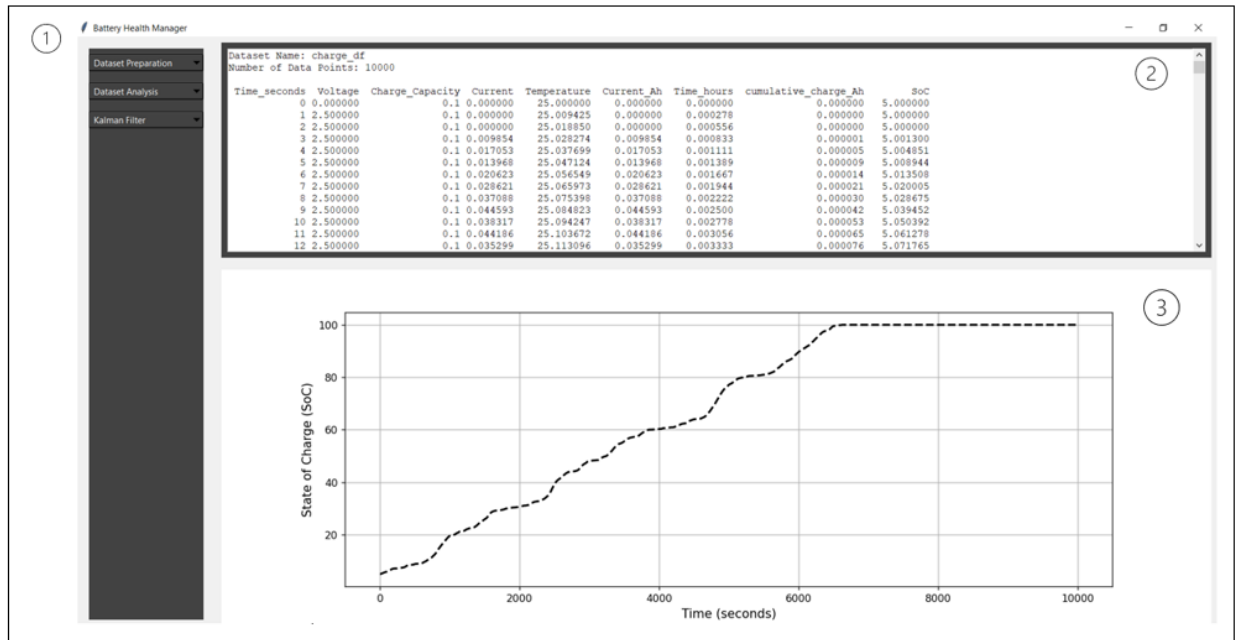


Fig. 48. Application system for SoC assessment

Assessment of SC is performed in the following sequence: generate dataset or import dataset in CSV/Excel format → click on calculate SoC → click on plot SoC.

Conclusions

1. After analysing eighty-eight scientific sources and other practical links, it was found that the ageing mechanisms of lithium-ion batteries are primarily divided into chemical changes and physical mechanisms, with ageing scenarios categorised as calendar ageing and cyclic ageing. This also indicates that the ageing process is inevitable.
2. The development of mathematical models and methods for monitoring lithium-ion battery ageing and assessing reliability can help create a Battery Management System, which allows for the automatic observation of lithium-ion battery ageing and reliability. According to previous research by other scientists, neural networks and regression models are usually used for monitoring lithium-ion battery ageing and reliability assessment. Data are often pre-processed before being fed into Machine Learning models using signal processing methods. Two of the most commonly used states for assessing lithium-ion batteries are State of Charge (SoC) and State of Health (SoH).
3. After generating a synthetic lithium-ion battery charging dataset with 10,000 data points, it was determined that discharge data alone are not suitable for the State of Health (SoH) assessment, as they do not reflect the real situation where data are both discharged and charged multiple times. However, such data are suitable for State of Charge (SoC) assessment. Additionally, having the real discharge and charge data of the lithium iron phosphate battery the comparison of the real data contains more dynamics and noise, whereas the changes in the synthetic data are highly predictable. Therefore, the results of the Machine Learning models with synthetic data were better than with real data.
4. Machine Learning models for synthetic lithium-ion batteries were trained and tested only for State of Charge (SoC) assessment. According to the results obtained, Bayesian optimisation improved the performance of neural network models with both original data used as input and data pre-processed with signal processing methods. Observed R^2 increase in a range from 0.54% to 1.06%, while the observed MSE decrease is in a range from 30.00% to 35.29% indicating Bayesian optimisation potential suitability for neural network models improvement.
5. Comparing the best results of the neural network and regression models on the synthetic data training sample, it was observed that the MSE error of the neural network model was 97.55% lower than that of the regression model, while the R^2 value was 3.19% higher. These results show that regression models compared to the neural network are less suitable for assessing the State of Charge (SoC) of lithium-ion batteries.
6. Comparing the best results of the neural network and regression models on the real data training sample, it was observed that the MSE error of the neural network model was 63.70% lower than that of the regression model, while the R^2 value was 63.72% higher. This confirms that the calculation results with synthetic data are suitable for the case of real data as well, although graphical evaluation shows that the model results are closer to the estimated State of Charge (SoC) with synthetic data than with real data.
7. In the case of real data, the calculated State of Charge (SoC) are based just on the short-term signals. The dynamics of State of Health (SoH) are obtained by applying an empirical degradation model and using the already assessed State of Charge (SoC) values. This indicates that the State of Charge (SoC) evaluated by Machine Learning models can be applied to the assessment of the State of Health (SoH).

8. It has been observed that highly accurate results are obtained using the Kalman filter for both synthetic and real data, suggesting that this method may be suitable not only for lithium-ion batteries but also for other types of batteries.
9. Using preprocessed data as input with more variables than just voltage and current can improve the models performance results for lithium-ion batteries. Also, it was noticed that data preprocessing with Discrete Wavelet Transform Daubechies 6 level 7 helps to decrease MSE by 15% and increase R^2 by 0.22% in comparison with the initial data used as input for the LSTM model.
10. An application system has been developed that allows the user to independently generate synthetic data for lithium-ion batteries, calculate the State of Charge (SoC) of the batteries, and apply the Kalman filter. This system is convenient because the user does not need to do any programming themselves. Additionally, calculations can be performed with any lithium-ion or even other types of battery data, making the developed calculation program widely applicable.

Recommendations and possible directions for further research

Further research could be focused on the applications of mathematics in economics and computer science:

1. **Development of new Machine Learning models for lithium-ion battery ageing simulation and reliability assessment.** In this case, a detailed analysis of the effectiveness of developed models would be required.
2. **Economic studies related to the financial benefits of lithium-ion battery monitoring in the energy sector.** This research could be carried out by analysing available data from the energy sector or by collaborating with business enterprises.
3. **Development of the application system to include more features and cater to user needs.** For this research, it is recommended to conduct an analysis of existing systems and perform a study regarding the user needs.

Scientific literature sources

1. KULOVA, T. L., et al. A brief review of post-lithium-ion batteries. *Int. J. Electrochem. Sci. Interactive*. 15, 2020, 8, 7242-7259. Access via: <https://doi.org/10.20964/2020.08.22>. [viewed 2023-10-22];
2. KIM, T., et al. Lithium-ion batteries: outlook on present, future, and hybridized technologies. *Journal of Materials Chemistry A. Interactive*. 2019, 7, 2942-2964. Access via: <https://doi.org/10.1039/C8TA10513H>. [viewed 2023-10-22];
3. HOU, J., et al. Fundamentals and challenges of lithium-ion batteries at temperatures between 40 and 60° C. *Advanced Energy Materials. Interactive*. 10, 2020, 18, 1-23. Access via: <https://doi.org/10.1002/aenm.201904152>. [viewed 2023-10-22];
4. MASIAS, A; MARCICKI, J; PAXTON, W. A. Opportunities and challenges of lithium-ion batteries in automotive applications. *ACS energy letters. Interactive*. 6, 2021, 2, 621-630. Access via: <https://doi.org/10.1021/acsenenergylett.0c02584>. [viewed 2023-10-22];
5. AL SHDAIFAT, Y., et al. Basics, properties, and thermal issues of EV battery and battery thermal management systems: Comprehensive review. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering. Interactive*. 2, 2023, 3, 295-311. Access via: <https://doi.org/10.1177/09544070221079195>. [viewed 2023-07-16];
6. XIE, Ji; LU, Y. C. A retrospective on lithium-ion batteries. *Nature Communications. Interactive*. 11, 2020, 1, 1-4. Access via: <https://doi.org/10.1038/s41467-020-16259-9>. [viewed 2023-08-10];
7. TOMASZEWSKA, A., et al. Lithium-ion battery fast charging: A review. *Etransportation. Interactive*. 1, 2019, 1-28. Access via: <https://doi.org/10.1016/j.etrans.2019.100011>. [viewed 2023-07-17];
8. ZHANG, J.N., et al. Trace doping of multiple elements enables stable battery cycling of LiCoO₂ at 4.6 V. *Nature Energy. Interactive*. 4, 2019, 7, 594-603. Access via: <https://doi.org/10.1038/s41560-019-0409-z>. [viewed 2024-02-01];
9. URBAN, A; SEO, D. H; CEDER, G. *Computational understanding of Li-ion batteries. npj Computational Materials. Interactive*. 2, 2016, 1, 1-13. Access via: <https://doi.org/10.1038/npjcompumats.2016.2>. [viewed 2024-02-05];
10. LI, M., et al. 30 years of lithium-ion batteries. *Advanced Materials. Interactive*. 30, 2018, 33, 1800561. Access via: <https://doi.org/10.1002/adma.201800561>. [viewed 2023-08-17];
11. RANGARAJAN, S., et al. Lithium-ion batteries—The crux of electric vehicles with opportunities and challenges. *Clean Technologies. Interactive*. 4, 2022, 4, 908-930. Access via: <https://doi.org/10.3390/cleantechnol4040056>. [viewed 2023-08-19];
12. UN, C; AYDIN, K. Thermal runaway and fire suppression applications for different types of lithium ion batteries. *Vehicles. Interactive*. 3, 2021, 3, 480-497. Access via: <https://doi.org/10.3390/vehicles3030029>. [viewed 2023-07-16];
13. GANDOMAN, F. H., et al. Reliability evaluation of lithium-ion batteries for E-mobility applications from practical and technical perspectives: A case study. *Sustainability. Interactive*. 13, 2021, 21, 11688 viewed. Access via: <https://doi.org/10.3390/su132111688>. [viewed 2023-08-07];
14. RAMKUMAR, M. S., et al. Review on Li-Ion Battery with Battery Management System in Electrical Vehicle. *Advances in Materials Science and Engineering. Interactive*. 2022, 1-8. Access via: <https://doi.org/10.1155/2022/3379574>. [viewed 2023-07-16];

15. COUTO, L. D., et al. A Descriptor System Approach for the Nonlinear State Estimation of Li-Ion Battery Series/Parallel Arrangements. *IEEE Transactions on Control Systems Technology*. Interactive. 31, 2022, 2, 825-840. Access via: <https://doi.org/10.1109/TCST.2022.3215102>. [viewed 2023-08-19];
16. BARCELLONA, S., et al. Aging effect on the variation of Li-ion battery resistance as function of temperature and state of charge. *Journal of Energy Storage*. Interactive. 2022, 50: 104658. Access via: <https://doi.org/10.1016/j.est.2022.104658>. [viewed 2024-02-01];
17. FLAMME, B., et al. Guidelines to design organic electrolytes for lithium-ion batteries: environmental impact, physicochemical and electrochemical properties. *Green Chemistry*. Interactive. 19, 2017, 8, 1828-1849. Access via: <https://doi.org/10.1039/C7GC00252A>. [viewed 2024-02-01];
18. CHEN, Y., et al. A review of lithium-ion battery safety concerns: The issues, strategies, and testing standards. *Journal of Energy Chemistry*. Interactive. 59, 2021, 83-99. Access via: <https://doi.org/10.1016/j.jechem.2020.10.017>. [viewed 2024-02-01];
19. LAI, X., et al. A review of lithium-ion battery failure hazards: test standards, accident analysis, and safety suggestions. *Batteries*. Interactive. 8, 2022, 11, 248. Access via: <https://doi.org/10.3390/batteries8110248>. [viewed 2024-02-06];
20. LI, X., et al. A high voltage multiplexer with rail to rail output swing for battery management system applications. *IEICE Electronics Express*. Interactive. 14, 2017, 1, 20161144-20161144. Access via: <https://doi.org/10.1587/alex.13.20161144>. [viewed 2024-02-02];
21. LELIE, M., et al. Battery management system hardware concepts: An overview. *Applied Sciences*. Interactive. 8, 2018, 4, 534. Access via: <https://doi.org/10.3390/app8040534>. [viewed 2024-02-02];
22. HASAN, M. K., et al. Review of electric vehicle energy storage and management system: Standards, issues, and challenges. *Journal of energy storage*. Interactive. 41, 2021, 102940. Access via: <https://doi.org/10.1016/j.est.2021.102940>. [viewed 2024-02-07];
23. CHE, Y., et al. Health prognostics for lithium-ion batteries: mechanisms, methods, and prospects. *Energy & Environmental Science*. Interactive. 2023, 16, 338-371. Access via: <https://doi.org/10.1039/d2ee03019e>. [viewed 2023-07-17];
24. REDONDO-IGLESIAS, E; VENET, P; PELISSIER, S. Modelling lithium-ion battery ageing in electric vehicle applications—calendar and cycling ageing combination effects. *Batteries*. Interactive. 6, 2020, 1, 14. Access via: <https://doi.org/10.3390/batteries6010014>. [viewed 2023-08-09];
25. YANG, X, G., et al. Modeling of lithium plating induced aging of lithium-ion batteries: Transition from linear to nonlinear aging. *Journal of Power Sources*. Interactive. 2017, 360, 28-40. [viewed 2024-02-01]. Access via: <https://doi.org/10.1016/j.jpowsour.2017.05.110>. [viewed 2024-02-01];
26. COLLATH, N., et al. Aging aware operation of lithium-ion battery energy storage systems: A review. *Journal of Energy Storage*. Interactive. 55, 2022, 55, 105634. Access via: <https://doi.org/10.14459/2022m p1652796>. [viewed 2024-02-02];
27. NARAYAN, R., et al. Self-healing: an emerging technology for next-generation smart batteries. *Advanced Energy Materials*. Interactive. 12, 2022, 17, 2102652. Access via: <https://doi.org/10.1002/aenm.202102652>. [viewed 2024-02-01];
28. GAO, T., et al. Effect of aging temperature on thermal stability of lithium-ion batteries: Part A—High-temperature aging. *Renewable Energy*. Interactive. 203, 2023, 592-600 [viewed 2023-08-11]. Access via: <https://doi.org/10.1016/j.renene.2022.12.092>. [viewed 2023-08-11];

29. GAO, F., et al. A review on materials for flame retarding and improving the thermal stability of lithium ion batteries. *International Journal of Electrochemical Science*. Interactive. 15, 2020, 2: 1391-1411. Access via: <https://doi.org/10.20964/2020.02.24>. [viewed 2024-02-06];
30. VETTER, J., et al. Ageing mechanisms in lithium-ion batteries. *Journal of power sources*. Interactive. 147, 2005, 1-2, 269-281. Access via: <https://doi.org/10.1016/j.jpowsour.2005.01.006>. [viewed 2024-02-04];
31. REN, Y., et al. A Comprehensive Review of Key Technologies for Enhancing the Reliability of Lithium-Ion Power Batteries. *Energies*. Interactive. 16, 2023, 17: 6144. Access via: <https://doi.org/10.3390/en16176144>. [viewed 2024-02-06];
32. MICARI, S., et al. Reliability assessment and lifetime prediction of Li-ion batteries for electric vehicles. *Electrical Engineering*. Interactive. 104, 2022, 1, 165-177. Access via: <https://doi.org/10.1007/s00202-021-01288-4>. [viewed 2024-02-06];
33. FELLNER, J, P.; LOEBER, G.,; SANDHU, S, S. Testing of lithium-ion 18650 cells and characterizing/predicting cell performance. *Journal of Power Sources*. Interactive. 81-82, 1999, 867-871. Access via: [https://doi.org/10.1016/S0378-7753\(98\)00238-9](https://doi.org/10.1016/S0378-7753(98)00238-9). [viewed 2024-01-21];
34. MOULIK, B. Battery aging and fuel efficiency as optimisation objectives as part of a real. *Energy*. Interactive. 2000, 1-10. Access via: <https://doi.org/10.12783/shm2017/13919>. [viewed 2024-01-21];
35. BOTTE, G, G. *Thermal stability and modeling of lithium-ion batteries*. Doctoral dissertation. Interactive. University of South Carolina, 2000. Access via: <https://www.proquest.com/>. [viewed 2024-01-21];
36. HATCHARD, T. D., et al. Thermal model of cylindrical and prismatic lithium-ion cells. *Journal of The Electrochemical Society*. Interactive. 148, 2001, 7: A755. Access via: <https://doi.org/10.1149/1.1377592>. [viewed 2024-01-21];
37. LIU, J., et al. An adaptive recurrent neural network for remaining useful life prediction of lithium-ion batteries. In: *Annual Conference of the PHM Society*. Interactive. 2010, 1-9. Access via: <https://doi.org/10.36001/phmconf.2010.v2i1.1896>. [viewed 2024-01-21];
38. ZENATI, A; DESPREZ, P; RAZIK, H. Estimation of the SOC and the SOH of li-ion batteries, by combining impedance measurements with the fuzzy logic inference. In: *IECON 2010-36th Annual Conference on IEEE Industrial Electronics Society*. Interactive. 2010, 1773-1778. Access via: <https://doi.org/10.1109/IECON.2010.5675408>. [viewed 2024-01-21];
39. SARASKETA-ZABALA, E., et al. Realistic lifetime prediction approach for Li-ion batteries. *Applied energy*. Interactive. 162, 2016, 839-852. Access via: <https://doi.org/10.1016/j.apenergy.2015.10.115>. [viewed 2024-01-21];
40. TANG, X., et al. Real-time aging trajectory prediction using a base model-oriented gradient-correction particle filter for Lithium-ion batteries. *Journal of Power Sources*. Interactive. 440, 2019, 227118. Access via: <https://doi.org/10.1016/j.jpowsour.2019.227118>. [viewed 2024-01-21];
41. XIONG, R., et al. Lithium-ion battery health prognosis based on a real battery management system used in electric vehicles. *IEEE Transactions on Vehicular Technology*. Interactive]. 68, 2018, 5, 4110-4121. Access via: <https://doi.org/10.1109/TVT.2018.2864688>. [viewed 2024-01-21];
42. PRASAD, G, K.; RAHN, C, D. Model based identification of aging parameters in lithium ion batteries. *Journal of power sources*. Interactive. 232, 2013, 15, 79-85. Access via: <https://doi.org/10.1016/j.jpowsour.2013.01.041>. [viewed 2024-01-21];

43. WANG, Q-K., et al. Design and management of lithium-ion batteries: A perspective from modeling, simulation, and optimisation. *Chinese Physics B*. Interactive. 29, 2020, 6, 068201. Access via: <https://doi.org/10.1088/1674-1056/ab90f8>. [viewed 2024-01-21];
44. LIU, K., et al. Gaussian process regression with automatic relevance determination kernel for calendar aging prediction of lithium-ion batteries. *IEEE Transactions on Industrial Informatics*. Interactive. 16, 2019, 6, 3767-3777. Access via: <https://doi.org/10.1109/TII.2019.2941747>. [viewed 2024-01-21];
45. CHE, Y., et al. Lifetime and aging degradation prognostics for lithium-ion battery packs based on a cell to pack method. *Chinese Journal of Mechanical Engineering*. Interactive. 35, 2022, 4, 1-16. Access via: <https://doi.org/10.1186/s10033-021-00668-y>. [viewed 2024-01-21];
46. CHEN, Z., et al. Porous Electrode Modeling and its Applications to Li-Ion Batteries. *Advanced Energy Materials*. Interactive. 2022, 2201506. [viewed 2024-01-21]. Access via: <https://doi.org/10.1002/aenm.202201506>. [viewed 2024-01-21];
47. GAO, Q., et al. Impedance modeling and aging research of the lithium-ion batteries using the EIS technique. *SAE Technical Paper*. Interactive. 2019. [viewed 2024-01-21]. Access via: <https://doi.org/10.4271/2019-01-0596>. [viewed 2024-01-21];
48. VARINI, M; CAMPANA, P, E; LINDBERGH, G. A semi-empirical, electrochemistry-based model for Li-ion battery performance prediction over lifetime. *Journal of Energy Storage*. Interactive. 25, 2019, 100819. Access via: <https://doi.org/10.1016/j.est.2019.100819>. [viewed 2024-02-02];
49. VERMEER, W; MOULI, G, R, C; BAUER, P. A comprehensive review on the characteristics and modeling of lithium-ion battery aging. *IEEE Transactions on Transportation Electrification*. Interactive. 8, 2021, 2, 2205-2232. Access via: <https://doi.org/10.1109/TTE.2021.3138357>. [viewed 2024-01-17];
50. KITTANEH, O. On the Theory of the Arrhenius-Normal Model with Applications to the Life Distribution of Lithium-Ion Batteries. *Batteries*. Interactive. 9, 2023, 1, 55. Access via: <https://doi.org/10.3390/batteries9010055>. [viewed 2024-02-06];
51. KIRANA, R, C., et al. Failure assessment in lithium-ion battery packs in electric vehicles using the failure modes and effects analysis (FMEA) approach. *Journal of Mechatronics, Electrical Power, and Vehicular Technology*. Interactive. 14, 2023, 1, 94-104. Access via: <https://doi.org/10.14203/j.mev.2023.v14.94-104>. [viewed 2024-02-06];
52. MENG, H., et al. An integrated methodology for dynamic risk prediction of thermal runaway in lithium-ion batteries. *Process Safety and Environmental Protection*. Interactive. 171, 2023, 385-395. Access via: <https://doi.org/10.1016/j.psep.2023.01.021>. [viewed 2024-02-06];
53. MA, S., et al. Reliability Analysis and Comparison of Lithium Battery System Based on Passive and Active Equalization. In: *2023 6th International Conference on Electronics Technology (ICET)*. IEEE. Interactive. 2023, 454-458 [viewed 2024-02-06]. Access via: <https://doi.org/10.1109/ICET58434.2023.10211542>. [viewed 2024-02-06];
54. YEREGUI, J., et al. State of charge estimation combining physics-based and artificial intelligence models for Lithium-ion batteries. *Journal of Energy Storage*. Interactive. 73, 2023, 73, 108883. Access via: <https://doi.org/10.1016/j.est.2023.108883>. [viewed 2024-01-19];
55. ROJAS, C., et al. A critical look at efficient parameter estimation methodologies of electrochemical models for Lithium-Ion cells. *Journal of Energy Storage*. Interactive. 80, 2024, 110384. Access via: <https://doi.org/10.1016/j.est.2023.110384>. [viewed 2024-01-19];

56. NASCIMENTO, R, G., et al. A framework for Li-ion battery prognosis based on hybrid Bayesian physics-informed neural networks. *Scientific Reports*. Interactive. 13, 2023, 1, 13856. Access via: <https://doi.org/10.1038/s41598-023-33018-0>. [viewed 2024-01-21];
57. THAKKAR, R, R. Electrical equivalent circuit models of lithium-ion battery. In: *Management and Applications of Energy Storage Devices*, pp. 3-9. Interactive. IntechOpen, 2021. ISBN 9781839696459. Access via: <https://books.google.it/>. [viewed 2024-01-21];
58. AUCH, M., et al. Influence of lithium-ion-battery equivalent circuit model parameter dependencies and architectures on the predicted heat generation in real-life drive cycles. *Batteries*. Interactive. 9, 2023, 5, 274. Access via: <https://doi.org/10.3390/batteries9050274>. [viewed 2024-02-02];
59. DONG, G; WEI, J. A physics-based aging model for lithium-ion battery with coupled chemical/mechanical degradation mechanisms. *Electrochimica Acta*. Interactive. 395, 2021. 139133. Access via: <https://doi.org/10.1016/j.electacta.2021.139133>. [viewed 2024-01-20];
60. LAI, X; ZHENG, Y; SUN, T. A comparative study of different equivalent circuit models for estimating state-of-charge of lithium-ion batteries. *Electrochimica Acta*. Interactive. 259, 2018, 566-577. Access via: <https://doi.org/10.1016/j.electacta.2017.10.153>. [viewed 2024-01-19];
61. EDGE, J, S., et al. Lithium ion battery degradation: what you need to know. *Physical Chemistry Chemical Physics*. Interactive. 23, 2021, 14, 8200-8221 [viewed 2024-01-20]. Access via: <https://doi.org/10.1039/d1cp00359c>. [viewed 2024-01-20];
62. RICHARDSON, R, R.; OSBORNE, M, A.; HOWEY, D, A. Battery health prediction under generalized conditions using a Gaussian process transition model. *Journal of Energy Storage*. Interactive. 23, 2019, 320-328. Access via: <https://doi.org/10.1016/j.est.2019.03.022>. [viewed 2024-01-20];
63. WAAG, W; FLEISCHER, C; SAUER, D, U. Critical review of the methods for monitoring of lithium-ion batteries in electric and hybrid vehicles. *Journal of Power Sources*. Interactive. 258, 2014, 321-339. Access via: <https://doi.org/10.1016/j.jpowsour.2014.02.064>. [viewed 2024-01-20];
64. MA, M., et al. Fault detection of the connection of lithium-ion power batteries in series for electric vehicles based on statistical analysis. *Energy*. Interactive. 164, 2018, 745-756. Access via: <https://doi.org/10.1016/j.energy.2018.09.047>. [viewed 2024-02-04];
65. KHALEGHI, S., et al. Developing an online data-driven approach for prognostics and health management of lithium-ion batteries. *Applied Energy*. Interactive. 308, 2022, 308, 118348. viewed [viewed 2024-02-04]. Access via: <https://doi.org/10.1016/j.apenergy.2021.118348>. [viewed 2024-02-04];
66. SNOOK, I; MARSHALL, J, M.; NEWMAN, R, M. Physics of failure as an integrated part of design for reliability. *Annual Reliability and Maintainability Symposium, 2003, IEEE Interactive*, 2003, 46-54. Access via: <https://doi.org/10.1109/RAMS.2003.1181901>. [viewed 2024-02-04];
67. MATIC, Zoran; SRUK, Vlado. The physics-of-failure approach in reliability engineering. *ITI 2008-30th International Conference on Information Technology Interfaces, IEEE*. Interactive. 2008, 745-750. Access via: <https://doi.org/10.1109/ITI.2008.4588504>. [viewed 2024-02-04];
68. PARHIZKAR, T., et al. Dynamic Probabilistic Risk Assessment. In: *Online Probabilistic Risk Assessment of Complex Marine Systems: Principles, Modelling and Applications*, pp. 17-39. Interactive. Springer, 2022, ISBN 9783030880989. Access via: https://link.springer.com/chapter/10.1007/978-3-030-88098-9_2. [viewed 2024-02-04];

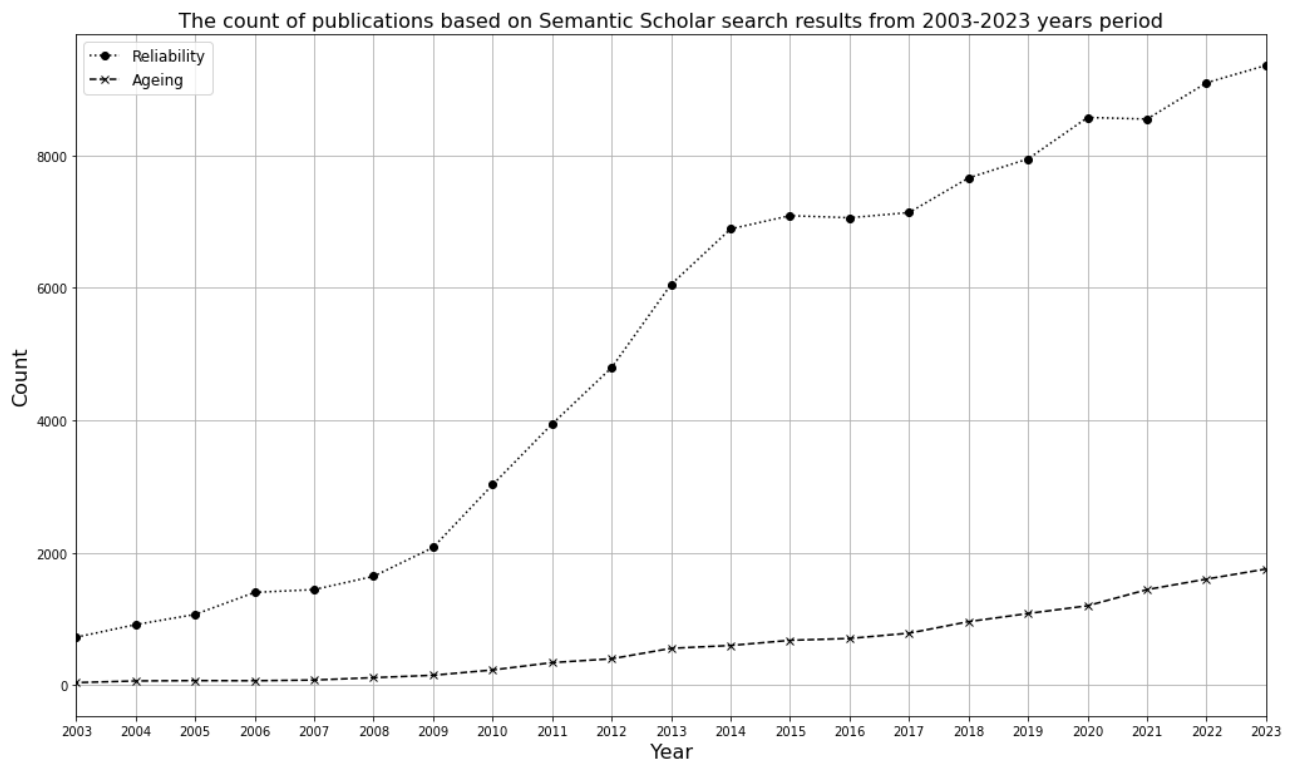
69. DENNING, R. S.; BUDNITZ, R. J. Impact of probabilistic risk assessment and severe accident research in reducing reactor risk. *Progress in Nuclear Energy*. Interactive. 102, 2018, 90-102. Access via: <https://doi.org/10.1016/j.pnucene.2017.05.021>. [viewed 2024-02-04];
70. ATAMURADOV, V., et al. Prognostics and health management for maintenance practitioners-Review, implementation and tools evaluation. *International Journal of Prognostics and Health Management*. Interactive. 8, 2017, 3, 1-31. Access via: <https://doi.org/10.36001/ijphm.2017.v8i3.2667>. [viewed 2024-02-04];
71. FENG, X., et al. Investigating the thermal runaway mechanisms of lithium-ion batteries based on thermal analysis database. *Applied Energy*. Interactive. 246, 2019, 53-64. Access via: <https://doi.org/10.1016/j.apenergy.2019.04.009>. [viewed 2024-02-04];
72. CHRISTENSEN, P, A., et al. Risk management over the life cycle of lithium-ion batteries in electric vehicles. *Renewable and Sustainable Energy Reviews*. Interactive. 148, 2021, 111240. Access via: <https://doi.org/10.1016/j.rser.2021.111240>. [viewed 2024-02-04];
73. ZHAO, J., et al. Review of State Estimation and Remaining Useful Life Prediction Methods for Lithium-Ion Batteries. *Sustainability*. Interactive. 15, 2023, 6, 5014. Access via: <https://doi.org/10.3390/su15065014>. [viewed 2024-02-04];
74. SESIDHAR, D. V. S. R.; BADACHI, C; GREEN II, R.C. A review on data-driven SOC estimation with Li-Ion batteries: Implementation methods & future aspirations. *Journal of Energy Storage*. Interactive. 72, 2023, 108420. Access via: <https://doi.org/10.1016/j.est.2023.108420>. [viewed 2024-02-04];
75. LIPU, MS H., et al. A review of state of health and remaining useful life estimation methods for lithium-ion battery in electric vehicles: *Challenges and recommendations*. *Journal of cleaner production*. Interactive. 205, 2018, 115-133. Access via: <https://doi.org/10.1016/j.jclepro.2018.09.065>. [viewed 2024-02-04];
76. SHU, X., et al. State of health prediction of lithium-ion batteries based on machine learning: Advances and perspectives. *Iscience*. Interactive. 24, 2021,11. Access via: <https://doi.org/10.1016/j.isci.2021.103265>. [viewed 2024-02-06];
77. GUO, S; MA, L. A comparative study of different deep learning algorithms for lithium-ion batteries on state-of-charge estimation. *Energy*. Interactive. 263, 2023, 125872. Access via: <https://doi.org/10.1016/j.energy.2022.125872>. [viewed 2024-02-06];
78. CHUNG, D, W. KO, J, H; YOON, K, Y. State-of-charge estimation of lithium-ion batteries using LSTM deep learning method. *Journal of Electrical Engineering & Technology*. Interactive. 17, 2022, 3, 1931-1945. Access via: <https://doi.org/10.1007/s42835-021-00954-8>. [viewed 2024-02-06];
79. DAS, K; KUMAR, R; KRISHNA, A. Analysing electric vehicle battery health performance using supervised machine learning. *Renewable and Sustainable Energy Reviews*. Interactive. 189, 2024, 113967. Access via: <https://doi.org/10.1016/j.rser.2023.113967>. [viewed 2024-02-06];
80. HUANG, B., et al. Prediction and evaluation of health state for power battery based on Ridge linear regression model. *Science Progress*. Interactive. 104, 2021, 4, 00368504211059047. Access via: <https://doi.org/10.1177/00368504211059047>. [viewed 2024-02-06];
81. KHUMPROM, P; YODO, N. A data-driven predictive prognostic model for lithium-ion batteries based on a deep learning algorithm. *Energies*. Interactive. 12, 2019, 4, 660. Access via: <https://doi.org/10.3390/en12040660>. [viewed 2024-02-06];

82. KAUR, K., et al. Deep learning networks for capacity estimation for monitoring SOH of Li-ion batteries for electric vehicles. *International Journal of Energy Research*. Interactive. 45, 2021, 2, 3113-3128. Access via: <https://doi.org/10.1002/er.6005>. [viewed 2024-02-06];
83. WANG, C., et al. Enhanced state-of-charge and state-of-health estimation of lithium-ion battery incorporating machine learning and swarm intelligence algorithm. *Journal of Energy Storage*. Interactive. 83, 2024, 110755. Access via: <https://doi.org/10.1016/j.est.2024.110755>. [viewed 2024-02-06];
84. KORKMAZ, M. SoC estimation of lithium-ion batteries based on machine learning techniques: A filtered approach. *Journal of Energy Storage*. Interactive. 72, 2023, 108268. Access via: <https://doi.org/10.1016/j.est.2023.108268>. [viewed 2024-02-06];
85. CHEN, C., et al. State-of-charge estimation of lithium-ion battery using an improved neural network model and extended Kalman filter. *Journal of Cleaner Production*. Interactive. 234, 2019, 1153-1164. Access via: <https://doi.org/10.1016/j.jclepro.2019.06.273>. [viewed 2024-02-06];
86. XIE, Q., et al. Residual life prediction of lithium-ion batteries based on data preprocessing and a priori knowledge-assisted CNN-LSTM. *Energy*. Interactive. 281, 2023, 128232. Access via: <https://doi.org/10.1016/j.energy.2023.128232>. [viewed 2024-02-06];
87. LI, S; HE, H; LI, J. Big data driven lithium-ion battery modeling method based on SDAE-ELM algorithm and data pre-processing technology. *Applied energy*. Interactive. 242, 2019, 1259-1273. Access via: <https://doi.org/10.1016/j.apenergy.2019.03.154>. [viewed 2024-02-06];
88. LI, Z., et al. Novel method for monitoring and adaptive estimation for SOC and SOH of lithium-ion batteries. *Journal of Energy Storage*. Interactive. 62, 2023, 106927. Access via: <https://doi.org/10.1016/j.est.2023.106927>. [viewed 2024-02-06];
89. WU, T., et al. SOC and SOH joint estimation of lithium-ion battery based on improved particle filter algorithm. *Journal of Electrical Engineering & Technology*. Interactive. 17, 2022, 1, 307-317. Access via: <https://doi.org/10.1007/s42835-021-00861-y>. [viewed 2024-02-06];
90. HONG, S., et al. State-of-health estimation and remaining useful life prediction for lithium-ion batteries based on an improved particle filter algorithm. *Journal of Energy Storage*. Interactive. 64, 2023, 107179. Access via: <https://doi.org/10.1016/j.est.2023.107179>. [viewed 2024-02-06];
91. MURAWWAT, S., et al. State of charge estimation and error analysis of lithium-ion batteries for electric vehicles using Kalman filter and deep neural network. *Journal of Energy Storage*. Interactive. 72, 2023, 108039. Access via: <https://doi.org/10.1016/j.est.2023.108039>. [viewed 2024-02-06];
92. DU, Z., et al. Data-driven estimation of remaining useful lifetime and state of charge for lithium-ion battery. *IEEE Transactions on Transportation Electrification*. Interactive. 8, 2021, 1, 356-367. Access via: <https://doi.org/10.1109/TTE.2021.3109636>. [viewed 2024-03-16];
93. JIANG, Y., et al. State of health estimation for lithium-ion battery using empirical degradation and error compensation models. *IEEE*. Interactive. 8, 2020, 123858-123868. Access via: <https://doi.org/10.1109/ACCESS.2020.3005229>. [viewed 2024-03-16];
94. ALTURKI, Fahd A., et al. EEG signal analysis for diagnosing neurological disorders using discrete wavelet transform and intelligent techniques. *Sensors*. Interactive. 20, 2020, 9, 2505. Access via: <https://doi.org/10.3390/s20092505>. [viewed 2024-03-16];
95. CAMPO, D; QUINTERO, O. L.; BASTIDAS, M. Multiresolution analysis (discrete wavelet transform) through Daubechies family for emotion recognition in speech. *Journal of Physics: Conference Series*. IOP Publishing. Interactive. 2016, 012034. Access via: <https://doi.org/10.1088/1742-6596/705/1/012034>. [viewed 2024-03-16];

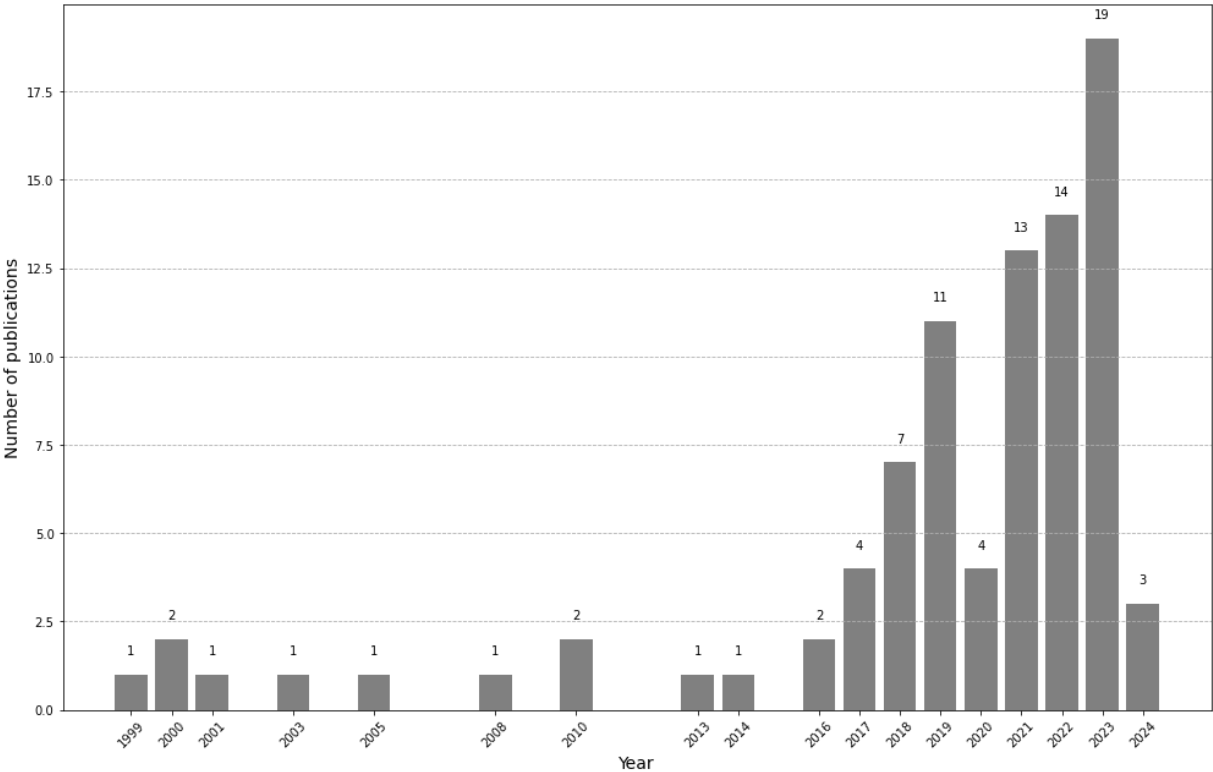
96. LIU, Y., et al. Applications of savitzky-golay filter for seismic random noise reduction. *Acta Geophysica*. Interactive. 64, 2016, 101-124. Access via: [https://doi.org/ 10.1515/acgeo-2015-0062](https://doi.org/10.1515/acgeo-2015-0062). [viewed 2024-03-16];
97. VINCENT, P., et al. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*. 11, 2010, 12. [viewed 2024-03-16];
98. BANERJEE, K., et al. Optimizing deep learning rnn topologies on intel architecture. *Supercomputing Frontiers and Innovations*. Interactive. 6, 2019, 3, 64-85. Access via: <https://doi.org/10.14529/jsfi190304>. [viewed 2024-03-16];
99. KATTE, T. Recurrent neural network and its various architecture types. *International Journal of Research and Scientific Innovation*. 5, 2018, 3, 124-129. [viewed 2024-03-16];
100. VICTORIA, A. H; MARAGATHAM, G. Automatic tuning of hyperparameters using Bayesian optimisation. *Evolving Systems*. Interactive. 12, 2021, 1, 217-223. Access via: <https://doi.org/10.1007/s12530-020-09345-2>. [viewed 2024-03-16];
101. PAULY, L., et al. Deeper networks for pavement crack detection. *Proceedings of the 34th ISARC*. 2017, 479-485. Access via: <https://doi.org/10.22260/isarc2017/0066>. [viewed 2024-03-16];
102. KHODARAHMI, M; MAIHAMI, V. A review on Kalman filter models. *Archives of Computational Methods in Engineering*. Interactive. 30, 2023, 1, 727-747. Access via: <https://doi.org/10.1007/s11831-022-09815-7>. [viewed 2024-03-16].

Appendixes

1 appendix. Changes in the number of publications



2 appendix. Literature sources distribution by year



3 appendix. GRU results for synthetic data SoC

Model input data & model specifications	Number of epochs	Train dataset results			Test dataset results		
		MSE	Tweedie error	R^2	MSE	Tweedie error	R^2
Original voltage and current variables with temperature, initial GRU model	100	0.0016	0.0111	0.9860	0.0018	0.0129	0.9849
	70	0.0018	0.0136	0.9843	0.0020	0.0158	0.9833
	50	0.0021	0.0186	0.9820	0.0022	0.0218	0.9811
	10	0.0032	1.3308	0.9723	0.0033	0.6593	0.9723
Original voltage and current variables with temperature, GRU model with Bayesian optimisation	100	0.0023	0.8873	0.9797	0.0025	0.5683	0.9785
	70	0.0020	0.0198	0.9826	0.0022	0.0207	0.9812
	50	0.0018	0.0107	0.9848	0.0019	0.0124	0.9837
	10	0.0021	0.0116	0.9817	0.0028	0.0126	0.9760
Original voltage and current variables without temperature variable, initial GRU model	100	0.0608	0.1403	0.4731	0.0613	0.1402	0.4807
	70	0.0611	0.1356	0.4707	0.0614	0.1360	0.4798
	50	0.0614	0.1358	0.4679	0.0615	0.1362	0.4788
	10	0.0626	0.1428	0.4577	0.0621	0.1420	0.4738
Kalman voltage and current variables with temperature, initial GRU model	100	0.0575	0.1710	0.5020	0.0580	0.1723	0.5086
	70	0.0581	0.1761	0.4965	0.0586	0.2005	0.5040
	50	0.0586	0.1793	0.4920	0.0591	0.2263	0.4999
	10	0.0596	0.1491	0.4833	0.0600	0.1864	0.4923
Kalman voltage and current variables with temperature, GRU model with Bayesian optimisation	100	0.0550	0.1223	0.5237	0.0558	0.1258	0.5273
	70	0.0565	0.1228	0.5105	0.0573	0.1261	0.5149
	50	0.0571	0.1257	0.5049	0.0579	0.1291	0.5096
	10	0.0584	0.1316	0.4937	0.0590	0.1351	0.5003
Kalman voltage and current variables without temperature variable, initial GRU model	100	0.0575	0.1710	0.5020	0.0580	0.1723	0.5086
	70	0.0581	0.1761	0.4965	0.0586	0.2005	0.5040
	50	0.0586	0.1793	0.4920	0.0591	0.2263	0.4999
	10	0.0596	0.1491	0.4833	0.0600	0.1864	0.4923
Autoencoder voltage and current variables with temperature, initial GRU model	100	0.0016	0.0093	0.9863	0.0019	0.0105	0.9843
	70	0.0017	0.0208	0.9849	0.0020	0.0506	0.9828
	50	0.0019	0.0124	0.9833	0.0022	0.0144	0.9813
	10	0.0030	0.5705	0.9736	0.0031	0.3472	0.9735
Autoencoder voltage and current variables with temperature, GRU model with Bayesian optimisation	100	0.0017	1.4085	0.9851	0.0018	1.1434	0.9844
	70	0.0426	0.5400	0.9842	0.0020	0.5560	0.9833
	50	0.0018	1.6249	0.9844	0.0019	1.3633	0.9837
	10	0.0023	0.2130	0.9803	0.0023	0.1714	0.9804
Autoencoder voltage and current variables without temperature variable, initial GRU model	100	0.0608	0.1388	0.4727	0.0616	0.1389	0.4788
	70	0.0611	0.1352	0.4704	0.0616	0.1357	0.4786
	50	0.0614	0.1354	0.4679	0.0616	0.1359	0.4783
	10	0.0625	0.1407	0.4584	0.0621	0.1402	0.4742
DWT DB 6 level 7 voltage and current variables with temperature, initial GRU model	100	0.0016	0.0178	0.9864	0.0021	0.0203	0.9825
	70	0.0019	0.0201	0.9832	0.0021	0.0227	0.9822
	50	0.0021	0.0186	0.9820	0.0021	0.0216	0.9821
	10	0.0035	0.4325	0.9698	0.0036	0.6581	0.9693
DWT DB 6 level 7 voltage and current variables with temperature, GRU model with Bayesian optimisation	100	0.0019	0.0168	0.9834	0.0021	0.0209	0.9824
	70	0.0050	0.0570	0.9571	0.0051	0.0601	0.9570
	50	0.0021	4.5044	0.9821	0.0023	4.3842	0.9809
	10	0.0022	0.0132	0.9807	0.0024	0.0156	0.9794
DWT DB 6 level 7 voltage and current variables without temperature variable, initial GRU model	100	0.0586	0.1303	0.4919	0.0583	0.1311	0.5063
	70	0.0594	0.1347	0.4856	0.0592	0.1350	0.4987
	50	0.0603	0.1352	0.4777	0.0600	0.1352	0.4915
	10	0.0611	1.4571	0.4706	0.0605	1.6457	0.4874

4 appendix. LSTM results for synthetic data SoC

Model input data & model specifications	Number of epochs	Train dataset results			Test dataset results		
		MSE	Tweedie error	R^2	MSE	Tweedie error	R^2
Original voltage and current variables with temperature, initial LSTM model	100	0.0019	0.0686	0.9837	0.0020	0.0943	0.9831
	70	0.0021	0.3091	0.9820	0.0022	0.1051	0.9815
	50	0.0022	0.0407	0.9808	0.0023	0.0213	0.9805
	10	0.0039	0.0495	0.9666	0.0039	0.0523	0.9667
Original voltage and current variables with temperature, LSTM model with Bayesian optimisation	100	0.0018	0.0218	0.9847	0.0019	0.0227	0.9842
	70	0.0020	0.0121	0.9830	0.0020	0.0138	0.9828
	50	0.0018	0.0162	0.9841	0.0019	0.0176	0.9840
	10	0.0022	0.0464	0.9809	0.0022	0.0574	0.9810
Original voltage and current variables without temperature variable, initial LSTM model	100	0.0615	0.1431	0.4669	0.0615	0.1414	0.4794
	70	0.0630	0.1464	0.4545	0.0624	0.1446	0.4715
	50	0.0631	0.1424	0.4532	0.0625	0.1424	0.4706
	10	0.0633	0.1451	0.4515	0.0627	0.1462	0.4691
Kalman voltage and current variables with temperature, initial LSTM model	100	0.0596	0.1861	0.4840	0.0599	0.2272	0.4931
	70	0.0597	0.1866	0.4826	0.0600	0.2277	0.4919
	50	0.0597	0.1896	0.4825	0.0600	0.2276	0.4917
	10	0.0598	0.1783	0.4822	0.0601	0.2045	0.4912
Kalman voltage and current variables with temperature, LSTM model with Bayesian optimisation	100	0.0550	0.1207	0.5238	0.0562	0.1242	0.5241
	70	0.0548	0.1224	0.5254	0.0562	0.1272	0.5243
	50	0.0558	0.1261	0.5162	0.0570	0.1307	0.5173
	10	0.0579	1.1066	0.4979	0.0586	1.4886	0.5040
Kalman voltage and current variables without temperature variable, initial LSTM model	100	0.0596	0.1861	0.4840	0.0599	0.2272	0.4931
	70	0.0597	0.1866	0.4826	0.0600	0.2277	0.4919
	50	0.0597	0.1896	0.4825	0.0600	0.2276	0.4917
	10	0.0598	0.1783	0.4822	0.0601	0.2045	0.4912
Autoencoder voltage and current variables with temperature, initial LSTM model	100	0.0019	0.0129	0.9832	0.0022	0.0153	0.9817
	70	0.0022	0.9957	0.9812	0.0024	1.2300	0.9800
	50	0.0023	0.6661	0.9804	0.0024	0.6101	0.9795
	10	0.0038	0.0460	0.9667	0.0039	0.0504	0.9668
Autoencoder voltage and current variables with temperature, LSTM model with Bayesian optimisation	100	0.0017	1.1194	0.9850	0.0018	0.8420	0.9844
	70	0.0017	1.7583	0.9855	0.0018	1.5478	0.9845
	50	0.0017	1.0561	0.9853	0.0018	0.7146	0.9848
	10	0.0023	4.4117	0.9799	0.0043	4.2881	0.9634
Autoencoder voltage and current variables without temperature variable, initial LSTM model	100	0.0616	0.1514	0.4664	0.0615	0.1491	0.4792
	70	0.0628	0.1427	0.4555	0.0623	0.1416	0.4722
	50	0.0630	0.1413	0.4543	0.0624	0.1412	0.4714
	10	0.0633	0.1446	0.4516	0.0627	0.1458	0.4686
DWT DB 6 level 7 voltage and current variables with temperature, initial LSTM model	100	0.0016	0.0194	0.9860	0.0017	0.0304	0.9853
	70	0.0017	0.0171	0.9852	0.0018	0.0187	0.9848
	50	0.0020	0.3577	0.9829	0.0021	0.1577	0.9826
	10	0.0038	0.0416	0.9675	0.0038	0.0464	0.9675
DWT DB 6 level 7 voltage and current variables with temperature, LSTM model with Bayesian optimisation	100	0.0022	7.0057	0.9811	0.0023	7.3948	0.9802
	70	0.0017	1.3977	0.9849	0.0019	1.1146	0.9839
	50	0.0020	0.2554	0.9831	0.0021	0.3147	0.9822
	10	0.0022	3.3726	0.9814	0.0022	3.2505	0.9814
DWT DB 6 level 7 voltage and current variables without temperature variable, initial LSTM model	100	0.0586	0.1303	0.4919	0.0583	0.1311	0.5063
	70	0.0594	0.1347	0.4856	0.0592	0.1350	0.4987
	50	0.0603	0.1352	0.4777	0.0600	0.1352	0.4915
	10	0.0611	1.4571	0.4706	0.0605	1.6457	0.4874

5 appendix. Publication and presentations

1. Grilauskaitė, R. & Alzbutas, R. Assessment of Parallel Network System Reliability Using a Recurrent Neural Network. Proceedings of Conference on *Mathematics and Natural Sciences: Theory and Application*. 1-9 p. 2023 May 12;
2. Grilauskaitė, R. & Alzbutas, R. Research on Machine Learning Models for Lithium-Ion Battery Aging Monitoring and Reliability Assessment. Conference *Mathematics and Natural Sciences: Theory and Application*. 2024, May 17;
3. Grilauskaitė, R. & Alzbutas, R. Applications of Machine Learning and Deep Learning Methods in Lithium-Ion Battery Aging Modeling and Reliability Assessment. Lithuanian Research Council Conference *The New Generation of Scientists*. 2024, May 17;
4. Grilauskaitė, R. & Alzbutas, R. Lithium-Ion Batteries Ageing Modelling and Reliability Assessment Methods. International Conference of Young Scientists on Energy and Natural Sciences Issues (*CYSENI*). 2024, May 21-23.

6 appendix. Python code for synthetic data generation

```
def simulate_battery_charge_step(num_points, step_interval, step_increase, max_capacity):
    time_seconds = np.arange(0, num_points)
    charge_capacity_data = np.zeros(num_points)
    step = 0
    for i in range(num_points):
        if (i % step_interval == 0) and (step < max_capacity):
            step += step_increase
            charge_capacity_data[i] = step
    return time_seconds, charge_capacity_data

def simulate_voltage(charge_capacity, voltage_start, voltage_end, voltage_step, step_interval):
    voltage_data = np.zeros_like(charge_capacity)
    voltage = voltage_start
    last_step_change_index = 0
    voltage_increase = True

    for i in range(1, len(charge_capacity)):
        if charge_capacity[i] > charge_capacity[i-1] and voltage_increase:
            voltage += voltage_step
            last_step_change_index = i
            voltage_increase = False
        elif (i - last_step_change_index) >= step_interval:
            if np.random.rand() > 0.5:
                voltage -= np.random.uniform(0.02, 0.05)
                voltage_increase = True
            else:
                voltage += voltage_step / 2
                voltage_increase = False

        voltage = max(min(voltage, voltage_end), voltage_start)
        voltage_data[i] = voltage

    voltage_data[-1] = voltage_end
    return voltage_data

def simulate_current(charge_capacity, current_start, current_peak, step_interval):
    current_data = np.zeros_like(charge_capacity)
    current = current_start
    for i in range(1, len(charge_capacity)):
        if charge_capacity[i] > charge_capacity[i-1]:
            current = -current_peak if i == 1 else current - np.random.uniform(0.02, 0.05)
        else:
            current += np.random.uniform(-0.01, 0.01)

        current = max(current, current_start)
        current_data[i] = current

    current_data[-1] = current_start
    return current_data

def simulate_smooth_temperature_profile(num_points, start_temp=25, peak_temp=40, fall_temp=28,
stable_temp=20):
    rise_time = int(0.25 * num_points)
    fall_time = int(0.25 * num_points)
    fall_to_stable_time = int(0.25 * num_points)
    stable_time = num_points - rise_time - fall_time - fall_to_stable_time

    temp_rise = start_temp + (peak_temp - start_temp) * np.sin(np.pi * np.arange(rise_time) / (2
* rise_time))
    temp_fall = peak_temp - (peak_temp - fall_temp) * np.sin(np.pi * np.arange(fall_time) / (2 *
fall_time))
    temp_fall_to_stable = fall_temp - (fall_temp - stable_temp) *
(np.arange(fall_to_stable_time) / fall_to_stable_time)
    temp_stable = np.full(stable_time, stable_temp)

    temperature_data = np.concatenate((temp_rise, temp_fall, temp_fall_to_stable, temp_stable))
    return temperature_data
```

7 appendix. Python code for data processing (example with real data)

```
##### Single-Layer Denoising Autoencoder

scaler = MinMaxScaler()
df26['Scaled_Current'] = scaler.fit_transform(df26[['totCurr']])

def create_sdae(input_dim, encoding_dim=64):
    model = Sequential()
    # Encoder
    model.add(GaussianNoise(0.1, input_shape=(input_dim,)))
    model.add(Dense(encoding_dim, activation='relu'))
    # Decoder
    model.add(Dense(input_dim, activation='sigmoid'))
    model.compile(optimizer=Adam(), loss='mean_squared_error')
    return model

sdae = create_sdae(1)
sdae.fit(df26[['Scaled_Current']], df26[['Scaled_Current']], epochs=100, batch_size=256,
        shuffle=True, verbose=1)

encoded_data = sdae.predict(df26[['Scaled_Current']])
df26['Autoencoder_Current'] = scaler.inverse_transform(encoded_data)

##### Savitzky-Golay Filter

window_length = 51
poly_order = 3

df26['Sgolay_Current'] = savgol_filter(df26['totCurr'], window_length, poly_order)

##### Discrete Wavelet Transform

totCurr = df26['totCurr']

wavelet_name = 'db4'
wavelet = pywt.Wavelet(wavelet_name)
max_level = pywt.dwt_max_level(data_len=len(totCurr), filter_len=wavelet.dec_len)
print("Maximum level of decomposition:", max_level)

current_signal = df26['totCurr'].values

wavelet = 'db4'
max_level = 17

coeffs = pywt.wavedec(current_signal, wavelet, level=max_level)

cA = coeffs[0]
coeffs_detail_none = [None] * len(coeffs[1:])

reconstructed_signal = pywt.waverec([cA] + coeffs_detail_none, wavelet)

if len(reconstructed_signal) > len(current_signal):
    reconstructed_signal = reconstructed_signal[:len(current_signal)]
elif len(reconstructed_signal) < len(current_signal):
    pad_length = len(current_signal) - len(reconstructed_signal)
    reconstructed_signal = np.pad(reconstructed_signal, (0, pad_length), 'constant',
    constant_values=(0))

df26['WT_Current_db4'] = reconstructed_signal
```

8 appendix. Python code for training and testing a RNN with synthetic data

```
def train_and_evaluate(epochs, model_name):
    start_time = time.time()

    np.random.seed(123)
    random.seed(123)
    tf.random.set_seed(123)

    model = Sequential([
        SimpleRNN(units=50, activation='relu', return_sequences=True,
input_shape=(X_train.shape[1], X_train.shape[2])),
        SimpleRNN(units=50, activation='relu'),
        Dense(1)
    ])

    model.compile(optimizer='adam', loss='mean_squared_error')

    train_rmse_per_epoch = []
    test_rmse_per_epoch = []
    train_mse_per_epoch = []
    test_mse_per_epoch = []

    for epoch in range(epochs):
        model.fit(X_train, y_train, epochs=1, batch_size=32, verbose=0)

        y_pred_train = model.predict(X_train)
        y_pred_test = model.predict(X_test)

        train_mse = mean_squared_error(y_train, y_pred_train)
        test_mse = mean_squared_error(y_test, y_pred_test)
        train_rmse = np.sqrt(train_mse)
        test_rmse = np.sqrt(test_mse)

        train_rmse_per_epoch.append(train_rmse)
        test_rmse_per_epoch.append(test_rmse)
        train_mse_per_epoch.append(train_mse)
        test_mse_per_epoch.append(test_mse)

    predicted_soc_scaled = model.predict(X)
    predicted_soc = scaler_target.inverse_transform(predicted_soc_scaled)
    df[model_name] = predicted_soc.flatten()

    plt.figure(figsize=(17, 5))
    plt.plot(train_rmse_per_epoch, 'b', label='Training RMSE')
    plt.plot(test_rmse_per_epoch, 'r--', label='Testing RMSE')
    plt.title(f'Training and Testing RMSE over Epochs ({model_name})')
    plt.xlabel('Epoch', fontsize=16)
    plt.ylabel('RMSE', fontsize=16)
    plt.legend()
    plt.grid(True)
    plt.show()

    plt.figure(figsize=(17, 5))
    plt.plot(train_mse_per_epoch, 'b', label='Training MSE')
    plt.plot(test_mse_per_epoch, 'r--', label='Testing MSE')
    plt.title(f'Training and Testing MSE over Epochs ({model_name})')
    plt.xlabel('Epoch', fontsize=16)
    plt.ylabel('MSE', fontsize=16)
    plt.legend()
    plt.grid(True)
    plt.show()
```

```

end_time = time.time()
execution_time = end_time - start_time
minutes, seconds = divmod(execution_time, 60)

return {'train_mse': train_mse, 'test_mse': test_mse, 'train_rmse': train_rmse, 'test_rmse':
test_rmse,
        'train_tweedie': mean_tweedie_deviance(y_train, np.clip(y_pred_train, 1e-6, None),
power=1.5),
        'test_tweedie': mean_tweedie_deviance(y_test, np.clip(y_pred_test, 1e-6, None),
power=1.5),
        'r2_train': r2_score(y_train, y_pred_train), 'r2_test': r2_score(y_test,
y_pred_test),
        'execution_time': f"{int(minutes)} minutes and {int(seconds)} seconds"}

epoch_configs = {100: 'RNN_InitialbeT_100', 70: 'RNN_InitialbeT_70', 50: 'RNN_InitialbeT_50',
10: 'RNN_InitialbeT_10'}

for epochs, model_name in epoch_configs.items():
    metrics = train_and_evaluate(epochs, model_name)

    print(f"Results for {model_name}:")
    print(f"Training MSE: {metrics['train_mse']:.4f}, RMSE: {metrics['train_rmse']:.4f}, "
          f"Tweedie: {metrics['train_tweedie']:.4f}, R-squared: {metrics['r2_train']:.4f}")
    print(f"Testing MSE: {metrics['test_mse']:.4f}, RMSE: {metrics['test_rmse']:.4f}, "
          f"Tweedie: {metrics['test_tweedie']:.4f}, R-squared: {metrics['r2_test']:.4f}")
    print(f"Execution time: {metrics['execution_time']}")
    print("\n")

```

9 appendix. Python code for Linear Regression model (example with real data)

```
model = LinearRegression()

start_time = time.time()

model.fit(X_train, y_train)

end_time = time.time()

y_pred_train_scaled = model.predict(X_train)
y_pred_test_scaled = model.predict(X_test)

train_mse = mean_squared_error(y_train, y_pred_train_scaled)
test_mse = mean_squared_error(y_test, y_pred_test_scaled)
train_rmse = np.sqrt(train_mse)
test_rmse = np.sqrt(test_mse)

total_time_seconds = end_time - start_time
minutes, seconds = divmod(total_time_seconds, 60)
print(f"Total training time: {int(minutes)} minutes and {seconds:.2f} seconds")

r2_train = r2_score(y_train, y_pred_train_scaled)
r2_test = r2_score(y_test, y_pred_test_scaled)

print(f"Final Training MSE: {train_mse:.4f}, RMSE: {train_rmse:.4f}, R-squared: {r2_train:.4f}")
print(f"Final Testing MSE: {test_mse:.4f}, RMSE: {test_rmse:.4f}, R-squared: {r2_test:.4f}")

predicted_soc_scaled = model.predict(scaled_features)
predicted_soc = scaler_target.inverse_transform(predicted_soc_scaled)

df['SoC_LR_Initial_Visi'] = predicted_soc
```

10 appendix. Python code for Ridge Regression model (example with real data)

```
np.random.seed(42)

def ridge_regression_alpha_optimisation(params):
    alpha = params[0][0]

    model = Ridge(alpha=alpha)
    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)
    mse = mean_squared_error(y_test, y_pred)

    return mse

bounds = [{'name': 'alpha', 'type': 'continuous', 'domain': (0.001, 100)}]

optimizer = GPyOpt.methods.BayesianOptimisation(f=ridge_regression_alpha_optimisation,
                                                domain=bounds,
                                                acquisition_type='EI',
                                                exact_feval=True,
                                                maximize=False,
                                                random_state=1)

optimizer.run_optimisation(max_iter=50)

alpha_opt = optimizer.x_opt
print(f'Optimised alpha: {alpha_opt}')

start_time = time.time()

model_opt = Ridge(alpha=alpha_opt)
model_opt.fit(X_train, y_train)

y_pred_train_opt = model_opt.predict(X_train)
y_pred_test_opt = model_opt.predict(X_test)

train_mse_opt = mean_squared_error(y_train, y_pred_train_opt)
test_mse_opt = mean_squared_error(y_test, y_pred_test_opt)
train_rmse_opt = np.sqrt(train_mse_opt)
test_rmse_opt = np.sqrt(test_mse_opt)
r2_train_opt = r2_score(y_train, y_pred_train_opt)
r2_test_opt = r2_score(y_test, y_pred_test_opt)

print(f"Optimised Training MSE: {train_mse_opt:.4f}, RMSE: {train_rmse_opt:.4f}, R-squared: {r2_train_opt:.4f}")
print(f"Optimised Testing MSE: {test_mse_opt:.4f}, RMSE: {test_rmse_opt:.4f}, R-squared: {r2_test_opt:.4f}")

y_train_actual = scaler_target.inverse_transform(y_train)
y_test_actual = scaler_target.inverse_transform(y_test)
y_pred_train_inv = scaler_target.inverse_transform(y_pred_train_opt.reshape(-1, 1))
y_pred_test_inv = scaler_target.inverse_transform(y_pred_test_opt.reshape(-1, 1))

end_time = time.time()

execution_time = end_time - start_time
minutes = execution_time // 60
seconds = execution_time % 60

print(f"Code execution time: {int(minutes)} minutes and {seconds:.2f} seconds")
```

11 appendix. Python code for a Kalman filter (example with synthetic data)

```
start_time = time.time()

kf = KalmanFilter(dim_x=2, dim_z=1)

kf.F = np.array([[1., 1.],
                 [0., 1.]])

kf.H = np.array([[1., 0.]])

kf.x = np.array([df['totVolt'].iloc[0], 0])

kf.P *= 1000.

kf.R = np.array([[5]])

kf.Q = Q_discrete_white_noise(dim=2, dt=1, var=0.1)

filtered_voltage = np.zeros(len(df))
filtered_soc = np.zeros(len(df))

filtered_voltage = [0] * len(df)
filtered_soc = [0] * len(df)

for i in range(len(df)):
    kf.predict()
    kf.update(df['totVolt'].iloc[i])
    filtered_voltage[i] = kf.x[0]
    filtered_soc[i] = kf.x[1]

df['Filtered_Voltage'] = filtered_voltage
df['Filtered_SoC'] = filtered_soc

mse = mean_squared_error(df['totVolt'], filtered_voltage)
rmse = sqrt(mse)
r_squared = r2_score(df['totVolt'], filtered_voltage)

end_time = time.time()
execution_time = end_time - start_time

execution_time_minutes = int(execution_time // 60)
execution_time_seconds = int(execution_time % 60)

print(f'MSE: {mse:.4f}')
print(f'RMSE: {rmse:.4f}')
print(f'R-squared: {r_squared:.4f}')

if execution_time_minutes > 0:
    print(f'Code execution time: {execution_time_minutes} minutes and
{execution_time_seconds:.2f} seconds')
else:
    print(f'Code execution time: {execution_time_seconds:.2f} seconds')
```