BVM13TMO Semestral Report

Vojtěch Michal, michavo3

Abstract

This report presents all my work for the course Battery testing, modelling and state estimation during the semester. It is composed of 12 individual papers, each presenting a particular topic. Papers are organized chronologically, which corresponds roughly to the complexity of presented topics. The first part deals with the basics of batteries, their physical testing and measurements, followed by the presentation of various dynamical models of batteries, their implementation and identification. The next several parts are dedicated to the online estimation of SoC and SoH using various algorithms including Coulomb Counting and OCV method, culminating in the presentation of the (Extended) Kalman Filter and its use for estimation of hidden states as well as slow-changing parameters. The topic of online parameter estimation also covers recursive least squares. The end of the report deals with data science and machine learning methods and their use in battery management systems or the electrochemical impedance spectroscopy (EIS).

I. Revisions of individual reports

Based on the assessment and comments received during the semester, some reports were updated, corrected or extended. Most reports were also improved graphically, especially the placement of figures, which is now much more consistent, and they are collected into side-by-side pairs of figures to save space.

- The chapter 2 was extended to provide more conclusions, e.g. an analysis of design decisions an engineer may have to make when developing a BMS balancing module.
- The chapter 3 was augmented with better descriptions of displayed plots. A plot of cell voltages during individual cycles was added.
- The chapter 6 was extended to contain a more elaborate description of the parameter identification procedure.
- The chapter 9 was partially rewritten – due to an incorrect dicrete-time jacobian used in one of Extended Kalman Filters in dual EKF SoH estimation, results submitted during the semester were incorrect. When the bug was found and corrected, the algorithm now yields expected correct results.
- The chapter 13 was extended to include the formula for parameterized impedance used to fit the measured data, as well as an explanation of individual fitted parameters.

Contents

1 BVM13TMO Semestral Report **1**

I Revisions of individual reports **1**

2 Week 2 – BMS Temperature Stability **4**

I Experimental setup **4**

II Experimental results and conclusions **4**

3 Week 3 – MATLAB **5**

I Experimental results **5**

4 Week 4 – Battery testing **7**

I Experimental procedure **7**
I-A Charging pulse . . . 7
I-B Discharging pulse . . . 7
I-C Automated experiment . . . 7
I-D Parameter estimation . . . 8

II Conclusions **8**

5 Week 5 – Implementation of Mathematical Models **9**

I Simple battery model **9**

II Improved battery model **9**

6 Week 6 – Parametrization and Validation of Models **11**

I Open circuit voltage test **11**

II Identification of equivalent circuit model **11**

III Verification on dynamic discharge profile **12**

7 Week 7 – SOC Estimation, part 1 **13**

I Cell model **13**

II Algorithm implementation **13**
II-A Coulomb counting . . . 14
II-B Open circuit voltage lookup . . . 14
II-C Coulomb counting with reset . . . 15

III Discussion **16**

8 Week 8 – SOC Estimation – Extended Kalman Filter **18**

I Model implementation and validation **18**

II EKF Implementation **20**

III Discussion **21**

9 Week 9 – SOH Estimation **23**

I Model implementation **23**

II Experimental results **24**

10 Week 10 – Online Parameter Estimation **26**

I Recursive Least Squares **26**
I-A RLS for battery parameters 27

II Extended Kalman Filter **28**
II-A Hongwen model 28
II-B Michal model 28

III Estimation results **29**

11 Week 12 – Data-Driven Methods **31**

I Familiarization with the environment **31**

II Training the regressor **31**
II-A Selection of features 32
II-B Evaluation of the Regressor 32

12 Week 13 – Overcurrent Protection For A Modular System **34**

I Problem statement **34**

II Analytical solution **34**

III Feedback algorithm **35**

IV Conclusions **36**

13 Week 14 – Offline SoH Estimation **38**

I EIS **38**

II IC/DV Analysis **39**

14 Conclusions **41**

References **41**

Week 2 – BMS Temperature Stability

Abstract

The laboratory task focuses on the temperature stability of the voltage threshold of three distinct cell balancing circuits. Their U-I characteristics were measured at various temperatures to assess the variability of voltage thresholds in real applications.

I. Experimental setup

THE set of three cell balancing circuits – devices under test – with cell connection pads exposed as marked wires was enclosed into a box with a heating element and a thermometer. The voltage-current characteristics of individual devices were measured at several temperatures. Each measurement was performed manually using a programmable power supply used in the constant current operating mode. For each specified value of the discharging current, the corresponding voltage reading was recorded. This method suffered from inaccuracy by neglecting some factors, such as the wire resistance. Furthermore, the power supply only provides voltage readings with a resolution of 1 mV, which was insufficient to obtain more detailed U-I characteristics in the proximity of threshold voltage. Therefore the U-I characteristic is assumed to be piecewise linear

$$i(u) = \frac{\max(u - u_{\mathrm{T}}, 0)}{R}, \tag{1}$$

where u_{T} is the threshold voltage.

II. Experimental results and conclusions

Measured voltage-current characteristics of sample A for various temperatures are shown in Fig. 1 together with their corresponding linear approximations. The threshold voltage is calculated as the x-intercept of the linear approximation. It is clear that the threshold voltage significantly varies with temperature approximately according to

$$u_{\mathrm{T}}(\theta) \approx -0.0026\ \theta + 3.8695\ \mathrm{V}, \tag{2}$$

where θ is the temperature in °C. On the other hand the finer division of the horizontal axis in Fig. 2 hints that the circuit B is far more stable than sample A. The approximate temperature dependence of the threshold voltage is

$$u_{\mathrm{T}}(\theta) \approx -0.0002\ \theta + 3.9481\ \mathrm{V}, \tag{3}$$

i.e. more than an order of magnitude more stable (due to the smaller linear coefficient).

No characteristics could be measured for the sample C, as it employed digital control by an integrated circuit, switching the discharge resistor at irregular intervals according to the BMS internal logic. This made it impossible to measure any static characteristic. For practical implementation, the implementation of balancing considers the costs and precision. Even imprecise circuit A may be appropriate for some applications, since it probably relies only on simple components (Zener diode, comparator, resistor and MOSFET) and needs no integrated circuits, communication buses, clocks or voltage regulators. The digital control of sample C, on the other hand, enables more advanced logic, such as overtemperature protection of the balancing resistor by adjusting the intensity of discharging.

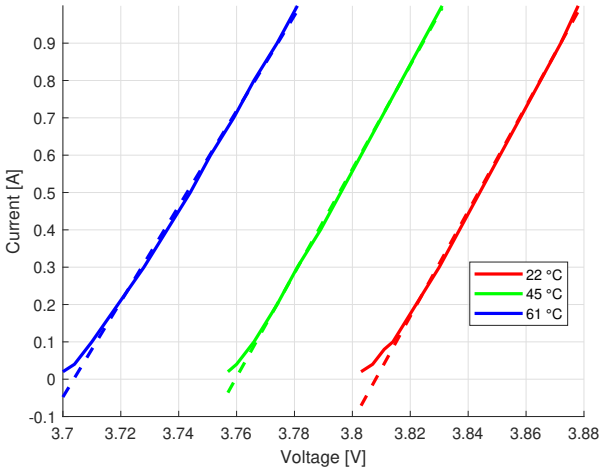


Fig. 1: U-I characteristic of sample A

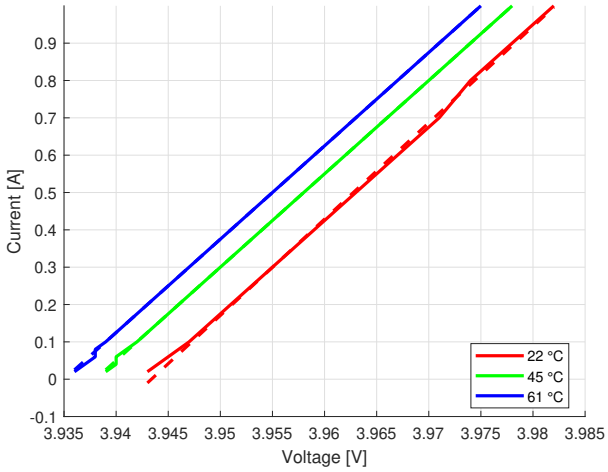


Fig. 2: U-I characteristic of sample B

Week 3 – MATLAB

Abstract

This report presents the results of an exercise focused on the basics of the MATLAB environment and its utilization for processing of large datasets. As an illustration, the report demonstrated results from processing a dataset of several cell charge/discharge cycles.

I. Experimental results

Table I lists discharge capacities recorded during individual cycles.

cycle ID	1	2	3	4	5	6
Discharging capacity Q [Ah]	3.163	3.1505	3.1311	3.1164	3.1049	3.2172

TABLE I: Discharging capacity available at each recorded cycle

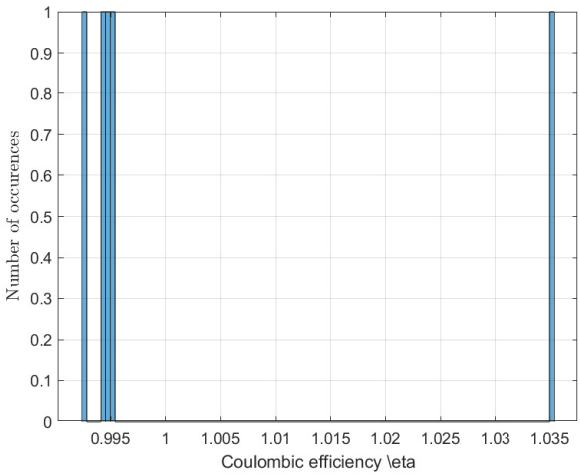


Fig. 3: Histogram of Coulombic efficiencies η from completed cycles.

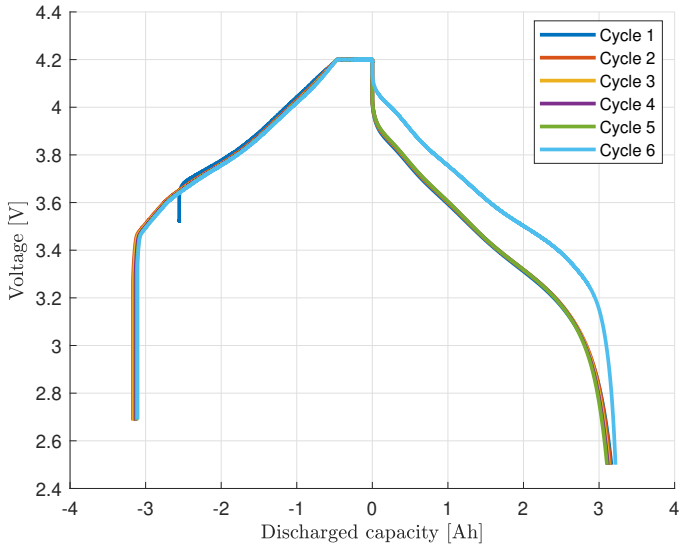


Fig. 4: Voltage/charge curves from individual full cycles.

Voltage measurements collected during individual cycles are shown in Fig. 4. Since individual cycles were performed consecutively (cell aging is negligible on such short time span) and it is reasonable to assume very similar temperatures, all cycles are very similar except for two outliers:

- the cycle 1 charging initiated from higher SoC, hence the voltage curve in Fig. 4 starts ”later”.
- the last cycle used a lower discharge current of 1 A (compared to 3 A used in other cycles), decreasing the voltage drop across the internal resistance and hence increasing the measured terminal voltage, ultimately causing the cell to provide slightly more charge before reaching the cutoff voltage.

Histogram of coulombic efficiencies recorded for individual cycles is show in Fig. 3. It only shows data from complete cycles (i.e. without the first cycle). An interesting detail is the occurrence of number greater than one – this η corresponds to the already discussed last cycle already, where lower discharge current allowed to cell to squeeze out slightly more charge before ending the discharge cycle and this extra charge exceeded the charge stored in the cell during the preceding charging cycle. Otherwise, the efficiency can’t be higher than 1. The state of charge at each time during the experiment is shown in Fig. 5.

Cell data from the second charge-discharge cycle are shown in Fig. 6, clearly showing the CC-CV charging mode and the subsequent CC discharge.

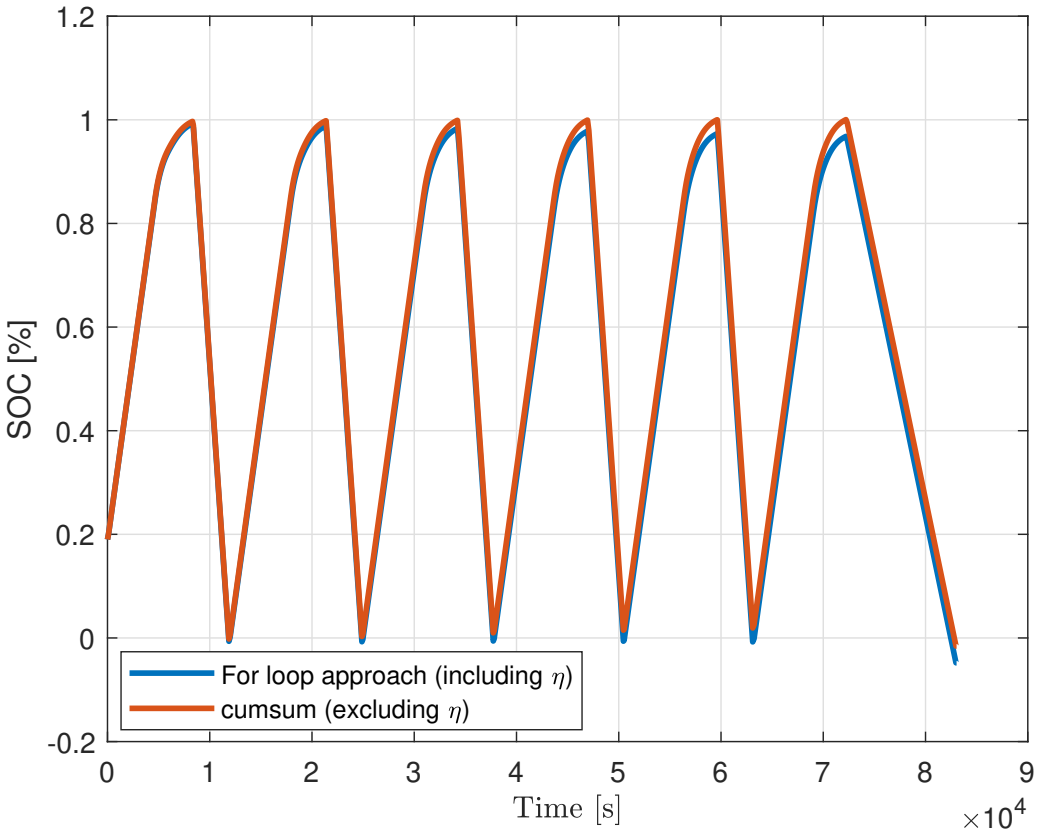


Fig. 5: State of charge calculated using two different methods

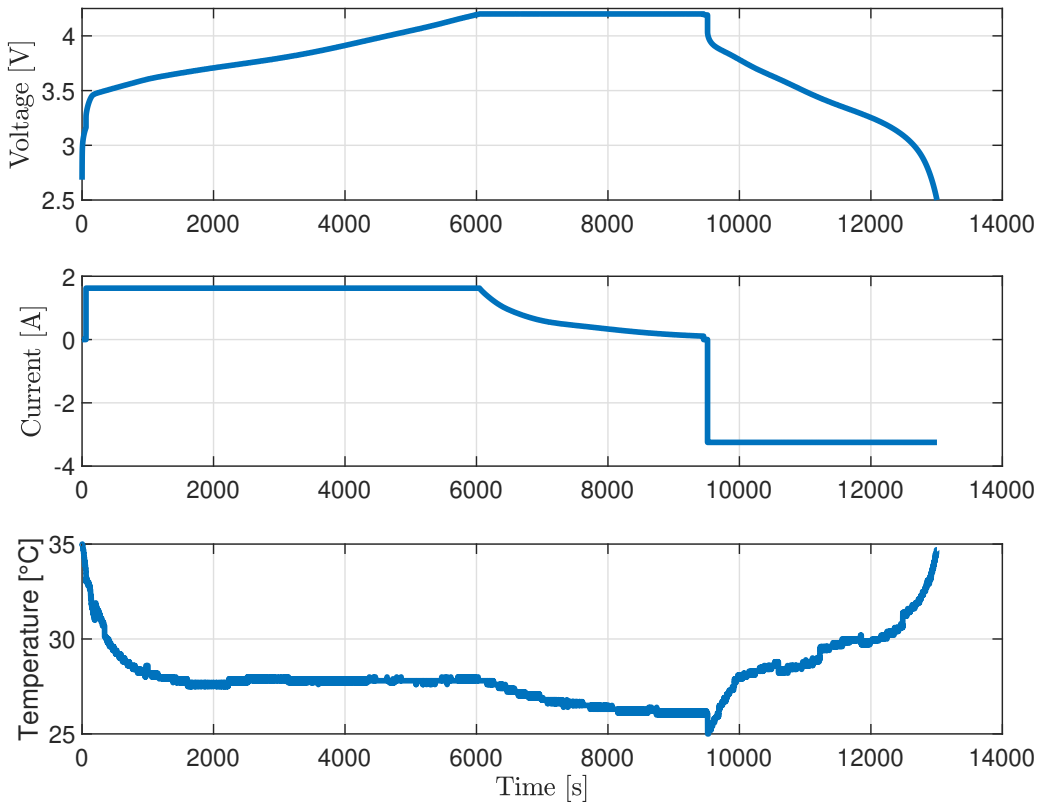


Fig. 6: Voltage, current and temperature of cell during second charge-discharge cycle

Week 4 – Battery testing

Abstract

This report presents the results of simple battery testing procedure. Constant current charging and discharging pulses were applied to the cell under test in order to estimate basic parameters of the equivalent circuit model using the measured voltages. This procedure was repeated twice, once ”manually” using a programmable power supply and DC load and once by programming a dedicated battery tester.

I. Experimental procedure

A. Charging pulse

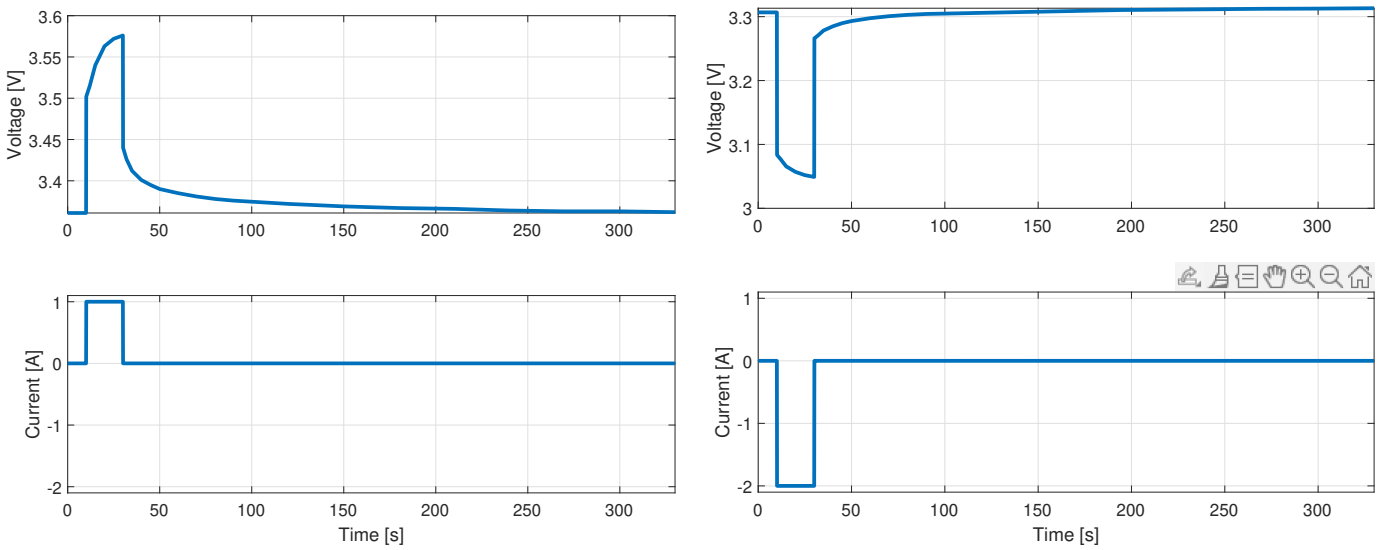
During the charging experiment, the cell was charged by a constant current of 1 A for 20 seconds. Afterwards the cell was allowed to rest for 5 minutes to restore open circuit voltage. Charging was performed using the laboratory power supply *BK PRECISION 9205*. No dedicated voltage measurement was provided, instead only readings from the instrument's display with resolution of 1 mV were recorded. Readings inherently contain voltage drop across the leads as well. Furthermore it is impossible to correctly record fast transients when enabling/disabling the power supply. This significantly reduced the reliability of equivalent circuit parameters R_0, R_1 and C_1 estimated in section I-D. The voltage profile together with the applied current is shown in Fig. 7(a).

B. Discharging pulse

In this experiment, the cell was discharged by a constant current of 2 A for 20 seconds and allowed to rest for 5 minutes afterwards. Discharging was performed using the DC programmable load *BK PRECISION 8601*. Since no dedicated voltage measurement was provided, measurements from this experiment suffer from the same errors as discussed for the charging experiment. Measurements were recorded using the instrument's display with resolution of 0.1 mV. The voltage profile together with the applied current is shown in Fig. 7(b).

C. Automated experiment

An automated battery tester *Newell BT-4008Tn-5V12A-S1* was programmed to perform a complete experiment containing 20 seconds of (dis)charging interleaved by relaxation time. Since the instrument (and the experimental setup overall) is professional, it can be assumed that some lead resistance is compensated in the recorded data. The instrument was configured to save quantities with sampling period 100 ms to correctly record fast transients when applying new value of current. The voltage profile together with the applied current is shown in Fig. 8.



((a)) Charging pulse experiment

((b)) Discharging pulse experiment

Fig. 7: Voltage-current profiles recorded during individual experiments

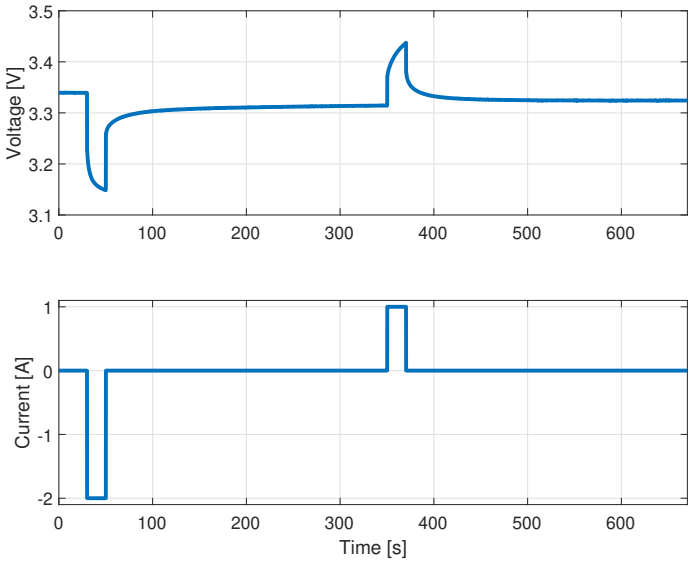


Fig. 8: Voltage profile during the automated experiment

D. Parameter estimation

Data recorded during three experiments were used to estimate parameters of the equivalent circuit model with series impedance R_0 and one RC element R_1, C_1. Estimated values are shown in table II. It can be concluded that although the experimental setup for manual charging/discharging was very simple, it yielded reasonable estimates that are at least within the order of magnitude from the expected value. Automated measurements performed by the programmable battery tester are far more accurate. This manifests especially in the case of parameter R_0, whose value is almost equal for both automated discharge as well as charge experiment. It is not possible to give a certain estimate of R_0 for either of the manual experiments due to low refresh rate of instruments' displays. Recording a video of instrument's screen also isn't a particularly error resistant and high sampling rate method for data processing.

Experiment	R_0 [Ω]	R_1 [Ω]	C_1 [F]
Manual discharge	1.116e-01	2.355e-02	2.547e+03
Manual charge	1.410e-01	7.801e-02	7.6897e+02
Automated discharge	5.664e-02	2.825e-02	2.124e+03
Automated charge	5.701e-02	5.539e-02	1.083e+03

TABLE II: Estimated parameters of the cell model

II. Conclusions

The given cell was subjected to charging and discharging pulses specified manually using a bench power supply and load and then using an automated measurement system. Automated measurement is far superior with respect to the required human time – preparation of the test scenario took a few minutes once in the beginning, allowing an unlimited number of subsequent executions, even on multiple cells in parallel. The automated system also yields far more repeatable results thanks to better uniformity of sampling intervals.

Week 5 – Implementation of Mathematical Models

Abstract

This report presents the Matlab implementation of a dynamic battery model based on an equivalent circuit. A battery cell is first modelled as a voltage source with internal impedance. Subsequently, two RC elements in series with internal impedance are added. Temperature and SOC dependence of individual parameters is considered as well. The cell model was fully parameterized in the assignment. Equations governing the behaviour of the given model were implemented in Matlab code to facilitate simulation.

I. Simple battery model

The first implemented model considered only an ideal voltage source U_{OC} with impedance R_0 in series. The only state is the state of charge (here given in percent rather than in the range $[0, 1]$. The flowing current I is the model input and the voltage across battery terminals U_{bat} is the only output. This model is is governed by the state space

$$\begin{aligned} S\dot{O}C &= -\eta\frac{1}{C}I, \\ U_{bat} &= U_{OC} - IR_0. \end{aligned} \tag{4}$$

The model was simulated over a time interval of 500 seconds with results shown in Fig. 9. Since the battery has total capacity of 3 Ah and the discharging current averages to 1 A, simple calculation predicts decrease of SOC by roughly 4.6 %. Since this prediction matches the simulation result, one can conclude that the simulation yields sane results.

II. Improved battery model

The second implemented model added two RC elements with parameters R_i and C_i on top of the already considered simple model. This modification adds two states – voltage drop U_i across each RC element. This model is is governed by the state space

$$\begin{aligned} S\dot{O}C &= -\eta\frac{1}{C}I, \\ \dot{U}_1 &= -\frac{1}{R_1C_1}U_1 + \frac{1}{C_1}I, \\ \dot{U}_2 &= -\frac{1}{R_2C_2}U_2 + \frac{1}{C_2}I, \\ S\dot{O}C &= -\eta\frac{1}{C}I, \\ U_{bat} &= U_{OC} - IR_0 - U_1 - U_2. \end{aligned} \tag{5}$$

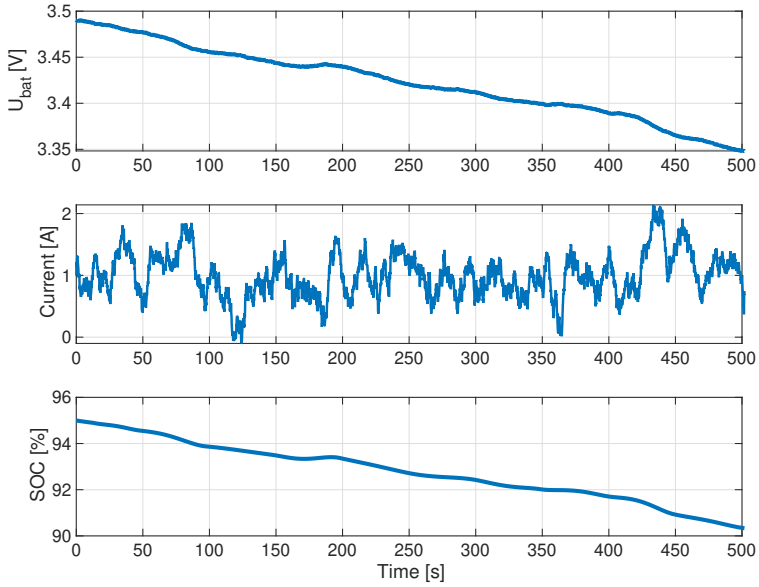


Fig. 9: Simulation results for simple battery model

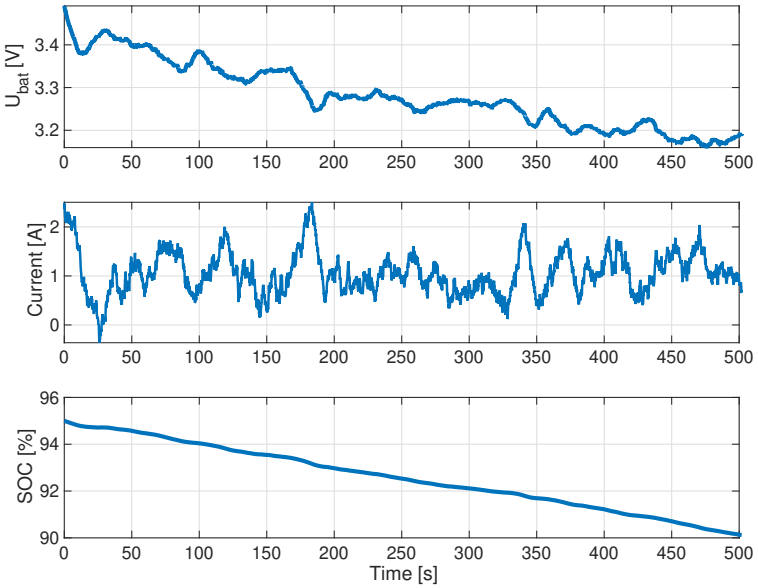


Fig. 10: Simulation results for improved battery model with two RC elements

whose parameters are all temperature-dependent.

The model was simulated over a time interval of 500 seconds using temperature shown in Fig. 11 with results shown in Fig. 10. The discharged SOC again matches the expected value of roughly 5 %. The terminal voltage U_{bat} is generally lower than in case of the simple model since both models use identical values of the internal impedance R_0 and the improved model adds additional voltage drops. These observations prove the sanity of the implemented model and obtained results.

Individual voltage drops across each RC element as well as the internal impedance R_0 are shown in detail in Fig. 12. Note how the voltage drop across R_0 is negligible compared to the drop across each RC element – R_0 is at least an order of magnitude lower than R_1 or R_2. This also explains why the terminal voltage U_{bat} in Fig. 9 is much smoother than U_{bat} in Fig. 10.

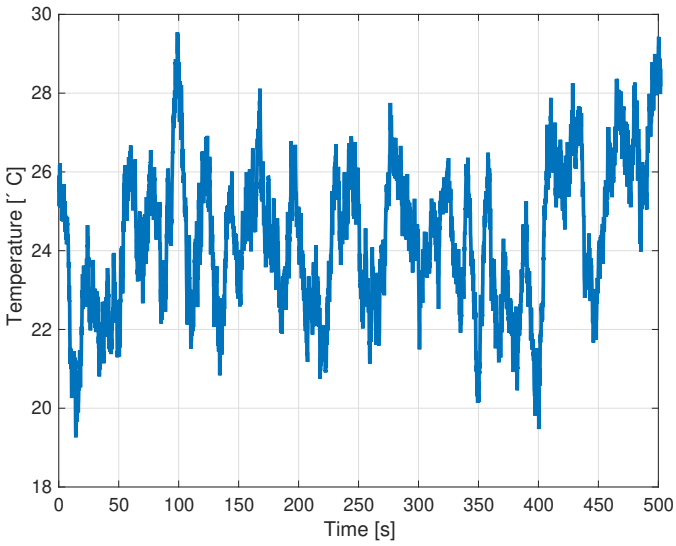


Fig. 11: Temperature used for improved model simulation

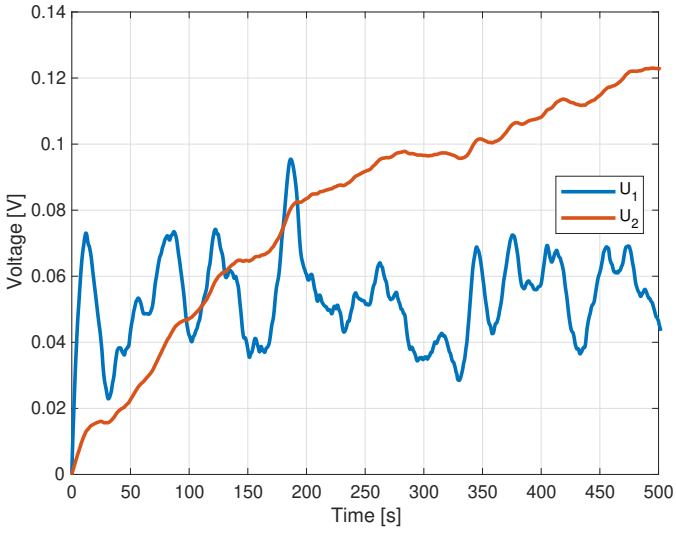


Fig. 12: Simulated voltage drops across each component of the improved model

Week 6 – Parametrization and Validation of Models

Abstract

This report presents results of parametrization and subsequent validation of a battery model using measured experimental data. Experimental recordings of slow charge and discharge cycle are used to estimate the open-circuit voltage U_{OC} as a function of SOC. A pulse test is used to identify equivalent circuit model parameters R_0, R_1 and C_1. All estimates are verified using a testing data set with dynamic discharge profile.

I. Open circuit voltage test

During the identification experiment, the cell went through a full charge-discharge cycle using a small current of 120 mA. First, I wanted to include identification of hysteresis (both maximal magnitude as well as its dynamics represented by the decay rate γ), but I have given up on this task due to the limited amount of available data. Although it would be possible to estimate the maximal magnitude of hysteresis by simply subtracting the charging and discharging voltage curve and compensating the voltage drop on internal resistance, this piece of information would be worthless. The training data set lacks any sufficiently dynamic experiment where the hysteresis decay rate could be reliably identified, hence I would have to tune it using the validation data, which is against the point of validation.

Hence only the OCV curve as a function of SOC was obtained by averaging the voltage curve from charge and discharge half-cycle. All relevant curves can be seen in Fig. 13.

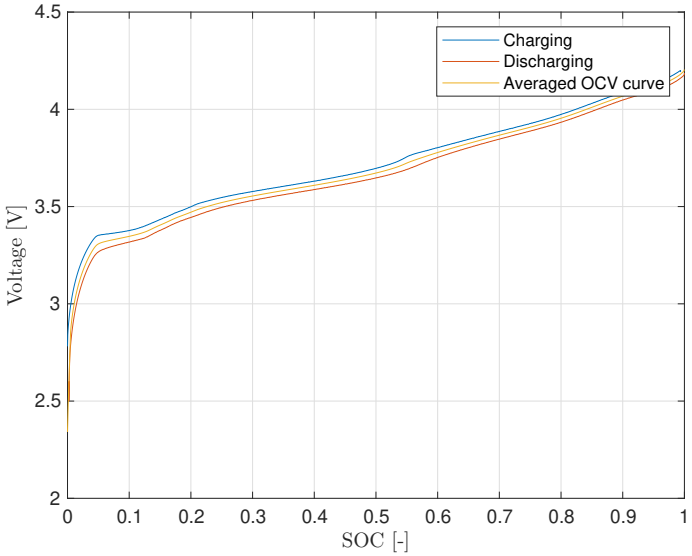


Fig. 13: Open circuit voltage as a function of SOC

II. Identification of equivalent circuit model

Parameters were determined by using the *System Identification Toolbox*, specifically the function `nlgreyest` required due to the nonlinearity of the battery output equation. It accepts a function evaluating the system's discrete-time outputs and state evolution (in this case the 1 RC ECM) and a set of input-output data to fit. Individual parameters are assigned values based on an initial guess. Estimated battery parameters are shown in Tab. III. The model achieved RMSE as low as 2.24 mV across the pulse experiment. This, combined with the degree of fit to training data more than 86 % shown in Fig. 14, proves that the model indeed converged to reasonable parameters – at least under the restriction of only 1 RC element. Adding more complexity to the model would improve its descriptive power and it would better match the system behavior.

The manual implementation of the discretized model was validated with results shown in Fig. 15. Since the results match the automatic comparison performed by Matlab, the model is implemented correctly.

Parameter	R_0	R_1	C_1
Value	63.4e-3 Ω	22.6e-3 Ω	2.2e4 F

TABLE III: Cell ECM parameters identified by the optimization procedure

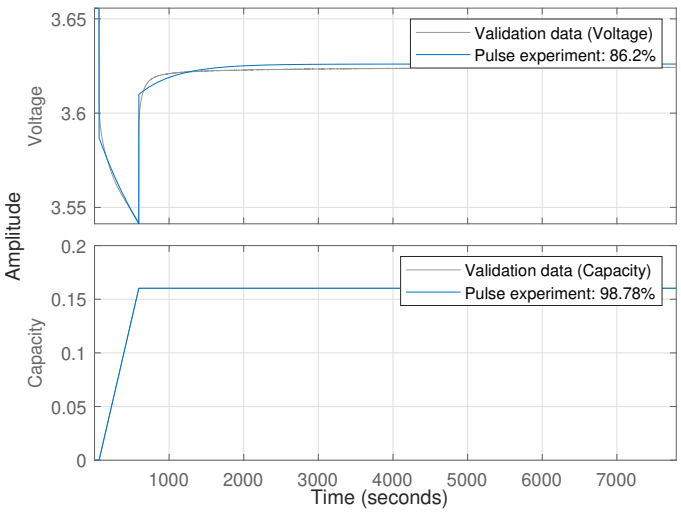


Fig. 14: Assesment of model fit quality

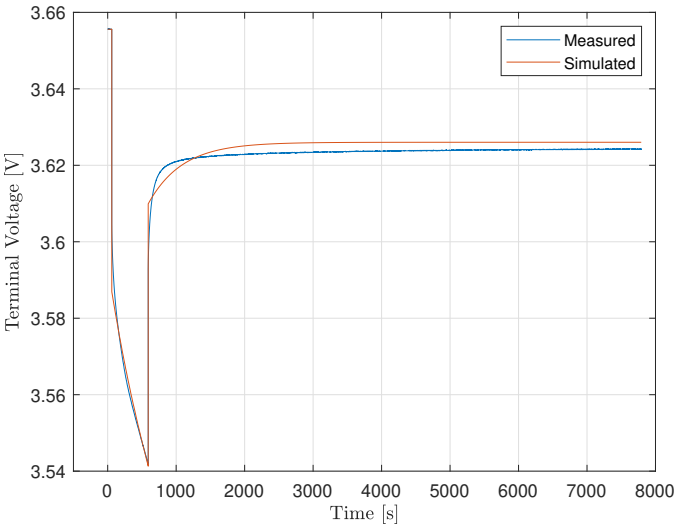


Fig. 15: Manual comparison of model outputs and measurements

III. Verification on dynamic discharge profile

A provided DDP data set was used to verify model parameters identified in previous sections. The DDP consists of rapid discharge by constant 5.2 A, followed by discharge by 1.3 A and short charging by 2.5 A; everything in quick succession in 35 seconds. The model achieved RMSE of 47.3 mV across the whole experiment, where clearly most of the error has accumulated at $t > 9000\,\mathrm{s}$, probably due to significantly different values of ECM parameters at low SoC. Fig. 16 shows that the model is indeed capable of capturing most of the cell dynamics, including gradual decrease of open circuit voltage. Some significant inaccuracy is still present however, as can be seen when zooming in on individual cycles of the DDP, as shown in Fig. 17. It is clear that the model failed to capture all system dynamics – this is however expected since we have only used 1 RC element in the model. Adding a second one would improve the model fit to training data and decrease the RMSE on verification data.

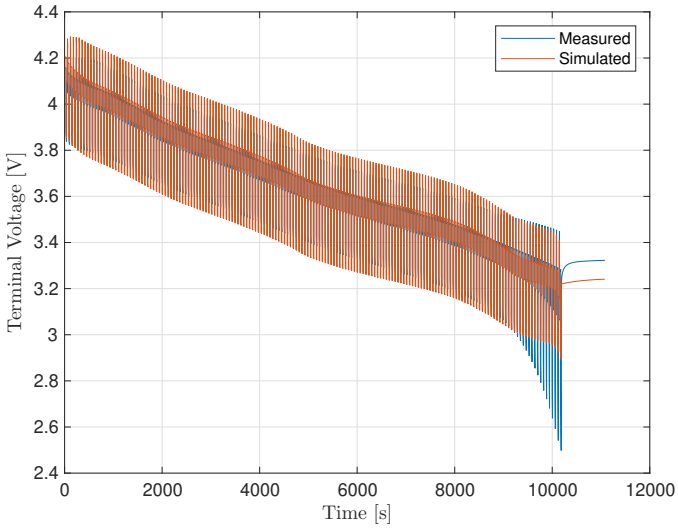


Fig. 16: Comparison of measurements with simulated model outputs

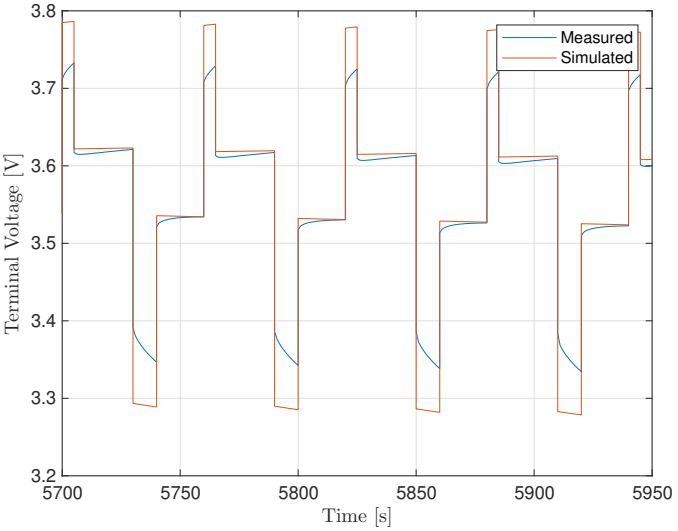


Fig. 17: Verification of identified model on detail of several DDP cycles

Week 7 – SOC Estimation, part 1

Abstract

This report presents an implementation of several basic algorithms used for state-of-charge estimation. Using noisy input-output data from a simulation of a 2RC equivalent circuit model, three methods of SoC determination – namely the Coulomb counting, lookup using filtered terminal voltage and the combination of both – were implemented and validated against reference data. All algorithms are executed for various values of hyperparameters to assess their robustness and errors caused by gradual drift or deviation in initial conditions. The accuracy of estimation is measured using the root-mean-square error.

I. Cell model

The chosen battery model considers only the dynamics of the SoC and otherwise represents the cell as a static system

$$\begin{aligned} \dot{SoC} &= -\frac{1}{C} i, \\ U_{\text{bat}} &= U_{\text{OC}}(SoC) - iR_0(SoC), \end{aligned} \tag{6}$$

with nonlinear output equation, where the system input i denotes the flowing current, state $SoC \in [0, 1]$ is the state of charge, U_{bat} is the battery terminal voltage (system output) and the total capacity C, open circuit voltage $U_{\text{OC}}(SoC)$ and internal resistance $R_0(SoC)$ are (possibly SoC-dependent) model parameters. Numeric values of all model parameters are provided as a part of the assignment. Forward Euler discretization of (6) with sampling period $T_s = 0.1\,\text{s}$ yields a simple system

$$\begin{aligned} SoC(k+1) &= SoC(k) - \alpha \frac{T_s}{C} i(k), \\ U_{\text{bat}}(k) &= U_{\text{OC}}(SoC(k)) - i(k) R_0(SoC(k)), \end{aligned} \tag{7}$$

appropriate for the implementation in code. The state equation was extended with a scaling factor α that handles the conversion of capacity C from ampere-hours to coulombs and the transformation of SoC from $[0, 1]$ to percentage.

II. Algorithm implementation

The implementation was verified by simulating the model using provided parameter values and several sets of initial conditions and comparing them against waveforms obtained during the reference experiment. As shown in Fig. 18, the reference current represents a highly dynamic profile that combines high charging and discharging currents interleaved by several minutes-long rest periods with no current flow. This waveform is appropriate for demonstrating various SoC estimation algorithms as it excites both long-term and short-term cell dynamics.

Since the model (7) is very simple, large discrepancies between the reference waveform and simulation outputs are expected, as shown in Fig. 19. Simulated terminal voltages differ by more than 80 mV using the RMSE metric, as they do not follow

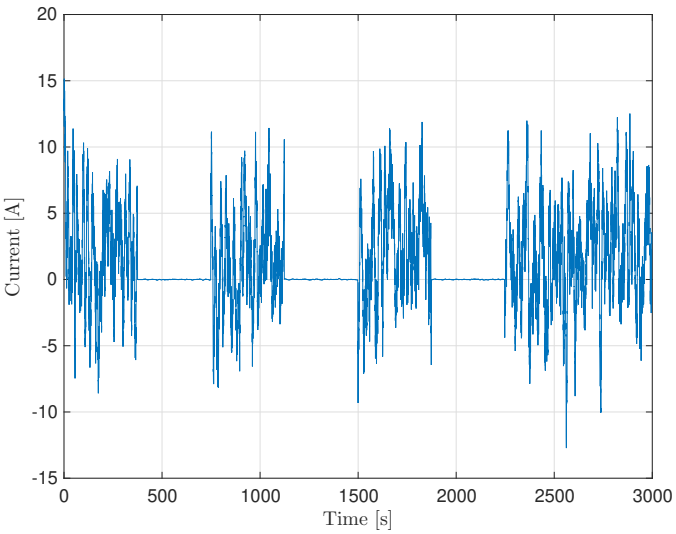


Fig. 18: Current waveform used during all simulations.

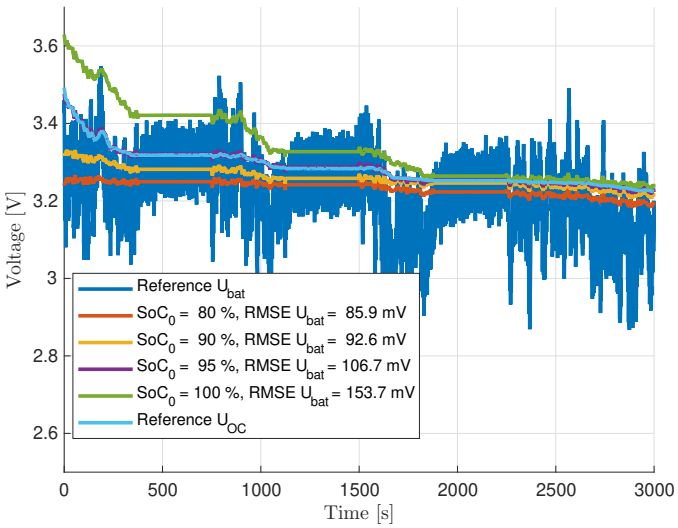


Fig. 19: Comparison of battery terminal voltages during simulation and the reference experiment.

relatively fast dynamics observable in the reference waveform. The reason becomes apparent on closer inspection – reference waveforms were generated using the 2RC equivalent circuit model, whereas the simulation only uses R_0 (neglecting R_1 and R_2). Neglecting the voltage drop across these resistances results in a far lower magnitude of voltage ripple in response to dynamically varying current.

A. *Coulomb counting*

The aforementioned systematic error is not a major issue for simple estimation of SoC, which, in the simple case without state observers, is completely unaffected by incorrect model impedance.The *Coulomb counting* algorithm evaluates the state equation in (7), performing an inherently open-loop integration of the flowing current. This way SoC can be calculated without any knowledge about U_{bat}, making this method useful when only a low-fidelity model is available.

This is illustrated in Fig. 20 – since the SoC evolution only depends on the accuracy of current measurement i and cell capacity C (and the coulombic efficiency η when not neglected), estimates of SoC for various initial conditions evolve "in parallel" and the error neither shrinks nor grows over time. However, even when the algorithm is initialized very close to the true initial state of charge (roughly 95 %), it tends to drift away over time. Open-loop integration accumulates errors and has no means of self-correction.

This behaviour can be observed in detail on Fig. 21. Several possible factors may cause the simulated waveform to drift away from the reference SoC over the course of the experiment, as shown in Fig. 21(a). Some insight into probable causes of the drift can be gained by looking at Fig. 21(b), where the error is plotted against the percentage of capacity discharged. Observing the highly linear dependence (linear fit achieves RMSE only 0.016 % SoC), one can conclude that the error is purely multiplicative and hence could be fixed by recalibrating the current sensor, recalculating the cell capacity C or possibly by extending the model with coulombic efficiency η, as the simulated output tends to overestimate the amount of charge remaining in the battery.

B. *Open circuit voltage lookup*

In contrast to Coulomb counting utilizing the measurement of battery current only, an alternative method uses only the measurement of battery terminal voltage U_{bat}. Calculating a moving average of length T_{avg} over the recorded U_{bat} yields an approximation of the U_{OC} that is subsequently used as an argument for table lookup. Fig. 22 shows how well the algorithm approximates the actual U_{OC} with various setting of the hyperparameter T_{avg}. A clear trend manifests, i.e. when T_{avg} is shorter, the algorithm is incapable of rejecting fast voltage drops when the cell is loaded, but it recovers quite quickly during the resting period. On the other hand, using a long averaging period T_{avg} results in stable U_{OC} estimates (dynamically varying current goes through heavier filtering), but the approximation never approaches the actual U_{OC} since averaging is longer than the resting period.

The same conclusion can be made when looking at Fig. 23 presenting SoC estimates for various values of the hyperparameter T_{avg}. Shorter T_{avg} yields an estimate with a larger variance that recovers quickly once the cell is allowed to rest. Longer T_{avg} ensures proper filtration of the estimate but simultaneously causes the estimate's mean value to deviate from the actual SoC. In this case, the algorithm underestimates the actual SoC since the reference waveform reaches higher discharge currents than charging currents; therefore, the mean of U_{bat} is lower than U_{OC}.

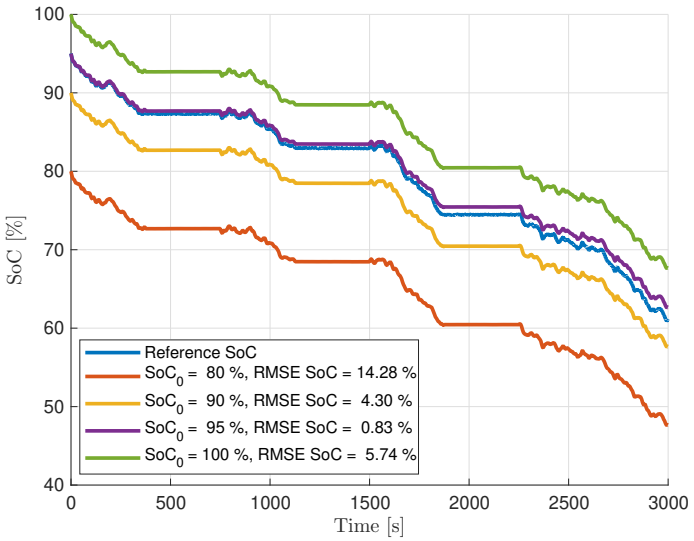
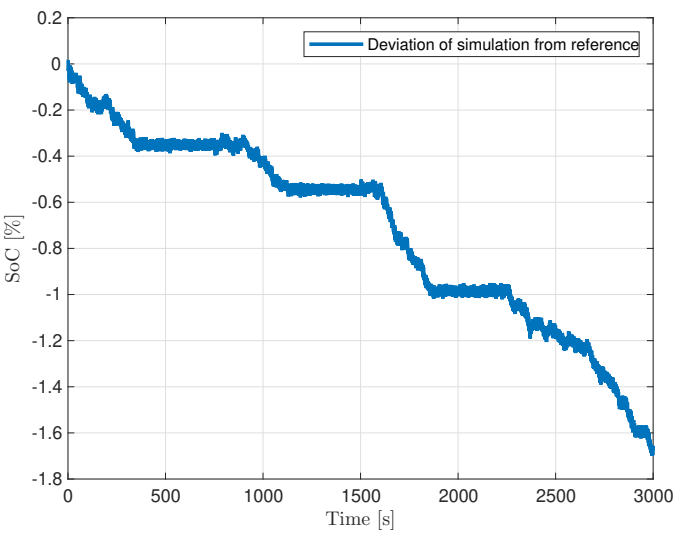
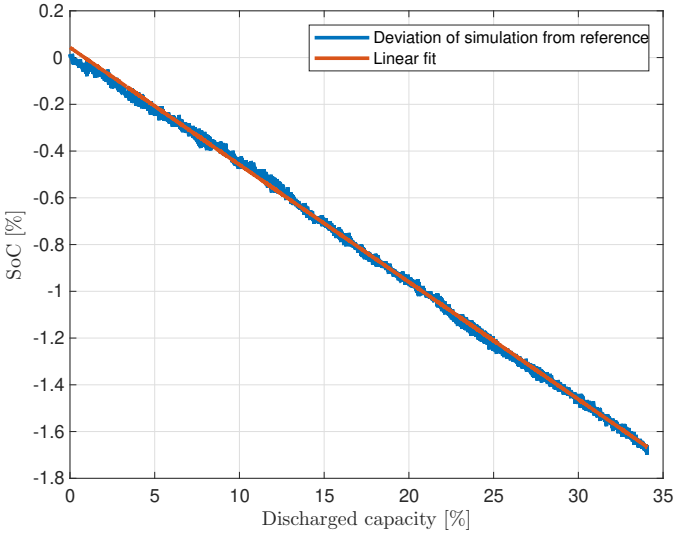


Fig. 20: Comparison of simulated SoCs with the reference experiment.



((a)) Plotted over time.



((b)) Plotted w.r.t the Depth of Discharge (DoD).

Fig. 21: Gradual deviation of the reference and simulated SoC.

C. *Coulomb counting with reset*

Advantages of both previously analyzed algorithms can be merged to achieve the great stability of Coulomb counting and the robustness to incorrect initial conditions characteristic to the U_{OC} lookup. The combined algorithm, from now on denoted as *Coulomb counting with reset*, integrates the flowing current just like normal Coulomb counting. But whenever a resting period longer than T_{delay} is detected, the measured terminal voltage U_{bat} is used as U_{OC} to correct the SoC estimate, removing any accumulated error.

This method has a hyperparameter T_{delay}, whose influence is illustrated in Fig. 24. When the current flow is interrupted at $t \approx 380\,\text{s}$ and the cell is allowed to rest, T_{delay} in some sense determines the amount of time given to the terminal voltage U_{bat} to converge to U_{OC}. After this delay elapses, the algorithm assumes that the U_{bat} has settled and uses its value to reset the integration. The general trend regarding T_{delay} can be inferred from Fig. 24(b) – when the value gets larger, the SoC estimate is smoother (more filtered and less prone to short-term random walks), but extending T_{delay} to infinity is not useful, as once it becomes longer than any reasonable resting period, the algorithm degenerates back to pure Coulomb counting without resets. On the other hand, shorter T_{delay} results in more frequent resets of the coulomb counting algorithm that occur even when U_{bat} is not fully settled, causing unwanted downwards spikes visible across the whole experiment in Fig. 24(a).

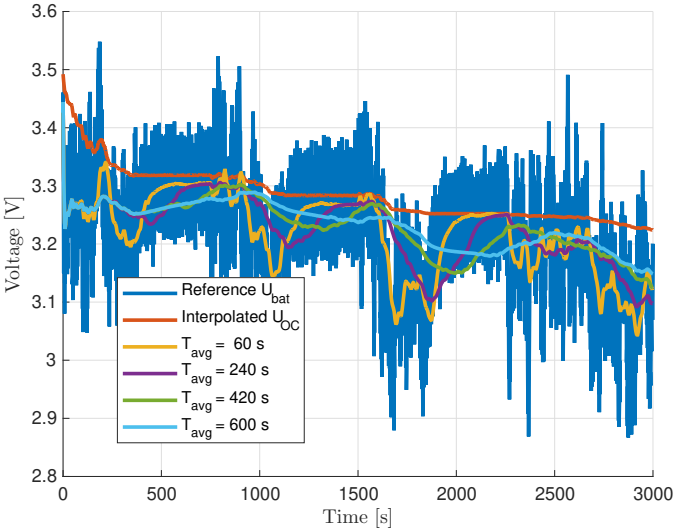


Fig. 22: U_{OC} estimates for various values of T_{avg}.

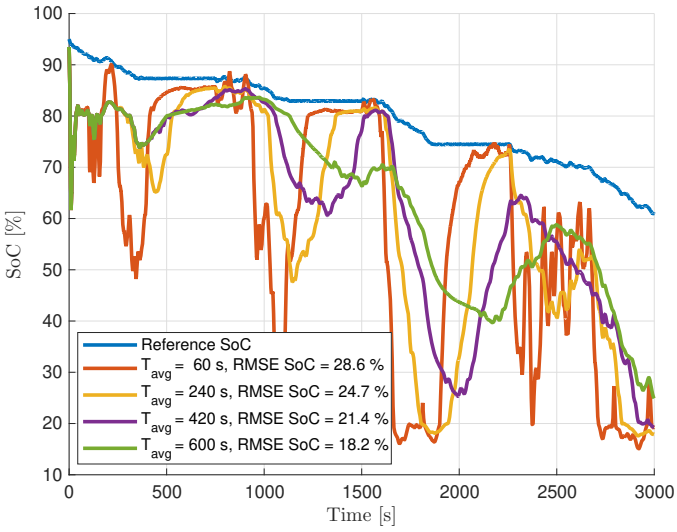
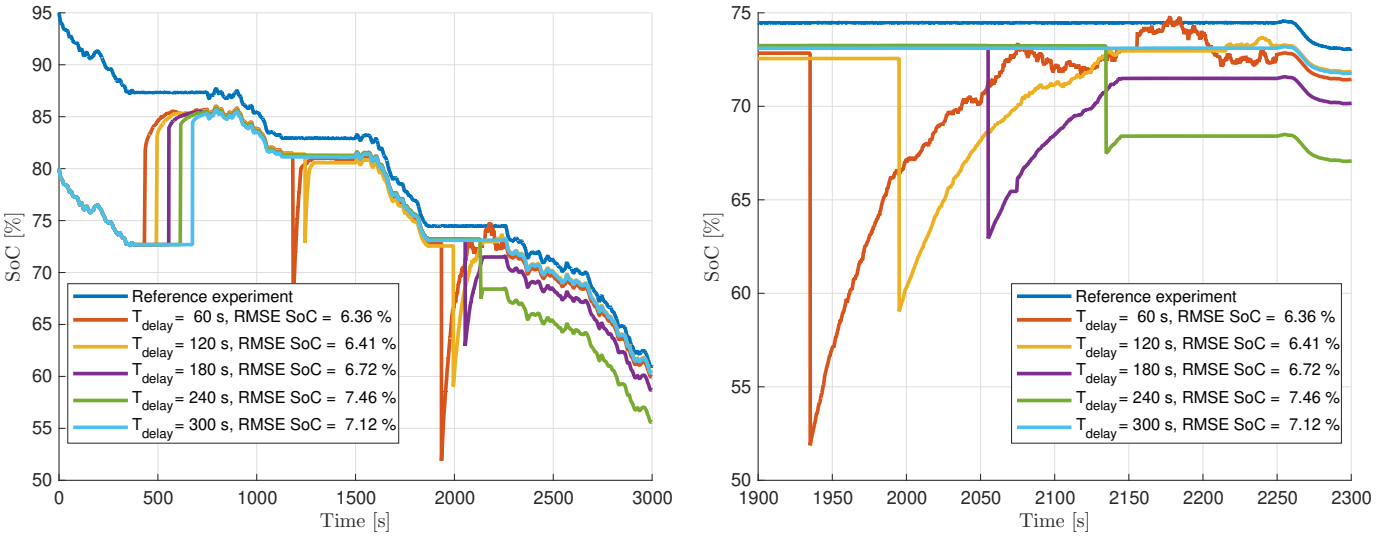


Fig. 23: SoC estimates for various values of T_{avg}.



((a)) SoC estimates for various hyperparameter values T_{avg}.

((b)) Detailed demonstration of reset behavior for various T_{avg}.

Fig. 24: Demonstration of the *Coulomb counting with reset* algorithm.

III. Discussion

In this homework, three simple methods of state-of-charge estimation were implemented. The first one – Coulomb counting – directly repeatedly evaluates the state equation for SoC using the current measurement i. The second method approximates U_{OC} by filtering terminal voltage measurement U_{bat} and subsequently looks the SoC up in a static table. The third method combines both approaches by performing Coulomb counting with occasional resets of the integrator with a value approximated using the U_{OC}.

Key differences between methods can be identified by inspecting Fig. 25, where each method is represented by just one waveform. Each selected waveform does not necessarily minimize the SoC RMSE, but it displays characteristic behavior and disadvantages inherent to the used method. The OCV method, as implemented in Sec. II-B, leads generally to the "wildest" (and worst) estimate from the customer's point of view. Coulomb counting analyzed in Sec. II-A leads to very accurate estimates of ΔSoC over short time periods before the integration accumulates significant errors. It is however incapable of yielding absolute SoC, since the initial condition for integration is unknown in real applications. Clearly the best performance is achieved by the combined algorithm discussed in Sec. II-C, because it combines infrequent updates about the absolute SoC from the OCV method with high accuracy relative ΔSoC calculated by the Coulomb counting.

This conclusion is supported by comparison of root-mean-square errors of estimates; as shown in Fig. 20, Coulomb counting is capable of achieving very low RMSE when given exact knowledge of the initial value. The OCV method achieves RMSE of

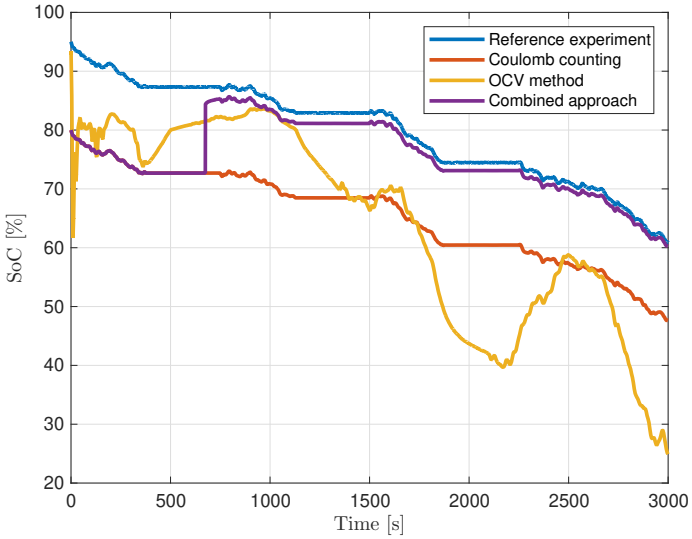


Fig. 25: Performance comparison of presented estimation algorithms.

roughly 18 % or more, as shown in Fig. 23, but it does not depend on the algorithm's initialization. Finally the values in Fig. 24 show what the combined approach achieves errors as low as 7 % in spite of very wrong initial estimate of SoC. Coulomb counting with reset combines low RMSE of Coulomb counting and robustness to wrong initial conditions characteristic to the OCV method.

Week 8 – SOC Estimation – Extended Kalman Filter

Abstract

This report presents an implementation of an EKF-based algorithm for state-of-charge estimation and compares its performance with several simple methods implemented in the previous assignment. Using noisy input-output data from a simulation of a 2RC equivalent circuit model, the nonlinear discrete-time 2RC equivalent circuit model is validated. This model is subsequently linearized analytically around a general operating point to obtain matrices $\mathbf{A}$ and $\mathbf{C}$ needed by the Extended Kalman Filter. Tuning noise covariance matrices $\mathbf{Q}$ and $\mathbf{R}$ is discussed. The performance of EKF is compared to simpler methods, namely the Coulomb counting, table lookup using filtered terminal voltage and a combination of both. Methods are compared visually as well as quantitatively using the root-mean-square error.

I. Model implementation and validation

The chosen implemented battery model is the standard 2RC equivalent circuit model

$$\dot{U_1} = -\frac{1}{R_1 C_1} U_1 + \frac{1}{C_1} i, \tag{8}$$

$$\dot{U_2} = -\frac{1}{R_2 C_2} U_2 + \frac{1}{C_2} i, \tag{9}$$

$$\dot{SoC} = -\frac{100}{C} i, \tag{10}$$

$$U_{\text{bat}} = U_{\text{OC}}(SoC) - U_1 - U_2 - R_0 i, \tag{11}$$

with a nonlinear output equation, where the system input i denotes the flowing current, states U_1 and U_2 are voltage drops across the two RC elements, $SoC \in [0, 100]$ % is the state of charge, U_{bat} is the battery terminal voltage (system output) and the total capacity C, open circuit voltage $U_{\text{OC}}(SoC)$, resistances $R_{0,1,2}$ and capacitances $C_{1,2}$ are (possibly SoC-dependent) model parameters with numeric values provided as a part of the assignment. To facilitate a simple iterative simulation of system behavior, the system (8)-(11) was discretized using the Forward Euler method as

$$\begin{aligned} x(k+1) &= x(k) + T_s f(x(k), u(k)), \\ y(k) &= g(x(k), u(k)), \end{aligned} \tag{12}$$

where f and g are vector functions of right-hand sides of (8)-(11) and $x(k)$, $u(k)$ and $y(k)$ are the system state, input and output vector at sample k, respectively.

The same current waveform shown in Fig. 26 that was recorded during the reference experiment was used for all simulations. Results of simulation with (12) are shown in Fig. 27, specifically Fig. 27(a) shows the evolution of system state SoC whereas

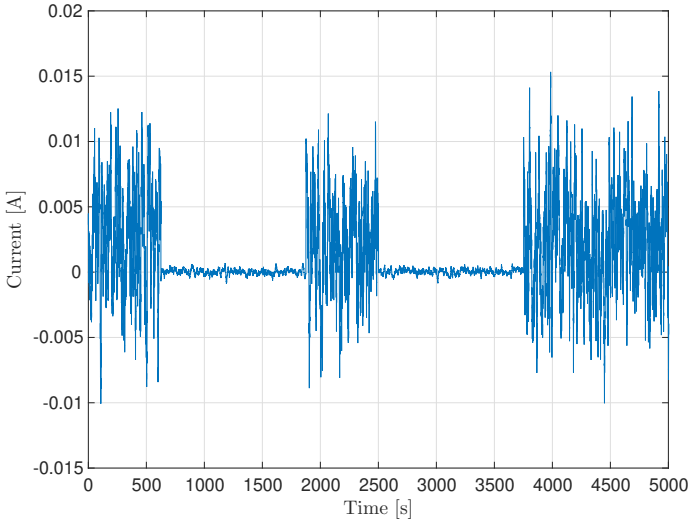
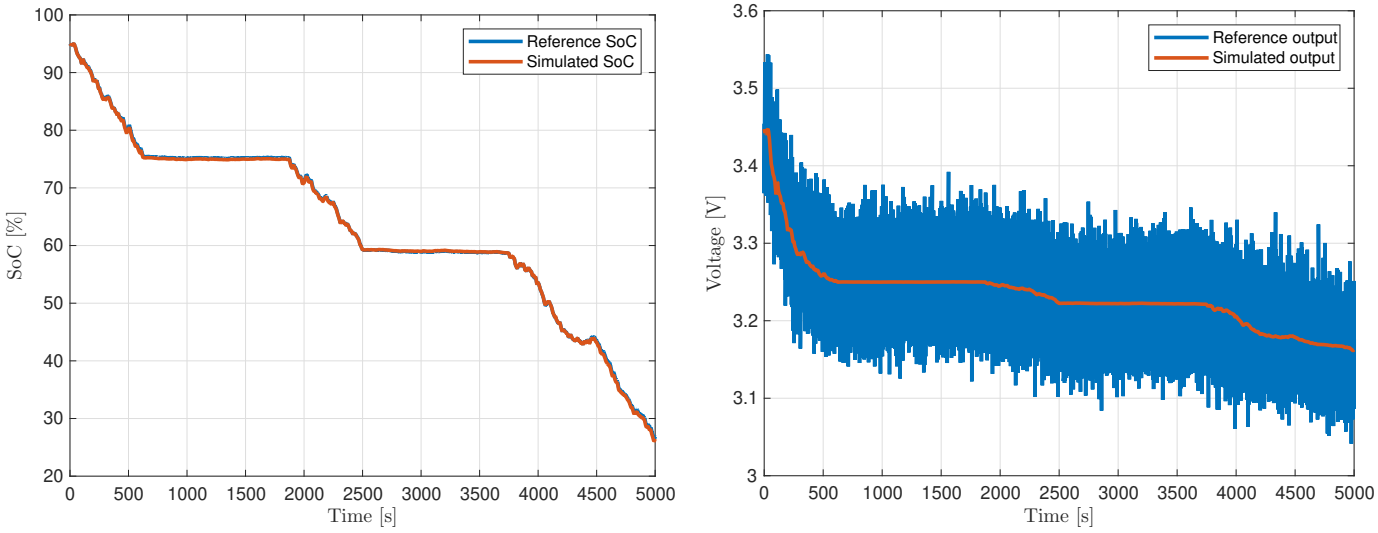


Fig. 26: Current waveform used during all simulations.



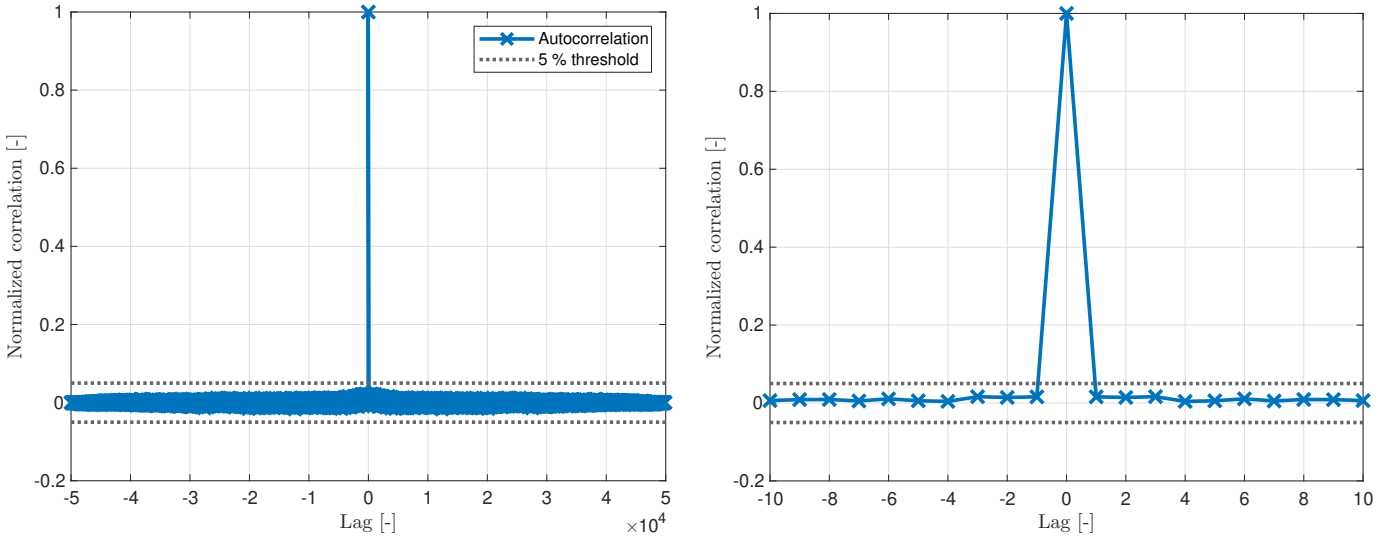
((a)) State of charge SoC.

((b)) Terminal voltage U_{bat}.

Fig. 27: Comparison of reference and simulated waveforms.

Fig. 27(b) compares the reference and simulated system output. Results confirm the expected near-perfect match of reference and simulated signals, since the reference experiment itself was a simulation, presumably using identical model parameters.

Assuming that the discrepancy between the simulated and reference U_{bat} visible in Fig. 27(b) is purely an additive noise sequence e, its parameters can be extracted. The random sequence e has zero mean and variance $\sigma^2 \approx 0.0013$. To assess its whiteness, its normalized autocorrelation function was evaluated and plotted in Fig. 28. Since the correlation for any lag other than 0 samples is far below the 5 % threshold, the noise sequence e can be assumed white. This result is relevant for the theoretical justification of the correctness of results obtained by the Kalman Filter.



((a)) For all possible shifts of the signal

((b)) Detail of the autocorrelation for low values of the signal lag

Fig. 28: Autocorrelation of the prediction error.

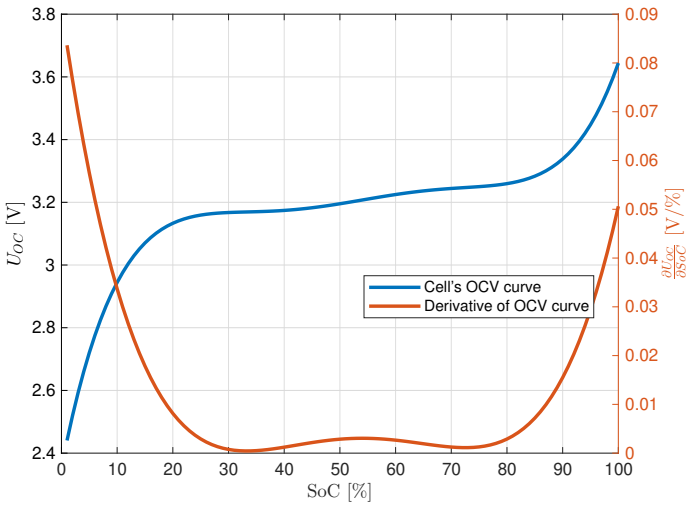


Fig. 29: Polynomial approximations of the U_{OC} curve and its derivative.

II. EKF Implementation

Building on top of the discrete-time model (12), the Extended Kalman Filter was implemented. Apart from the state evolution and output functions, the algorithm requires their derivatives w.r.t. individual states to form time-varying Jacobian matrices

$$\mathbf{C}(k) = \left.\frac{\partial y(k)}{\partial x(k)}\right|_{x(k),u(k)} = \left[-1 \quad -1 \quad \frac{\partial U_{\text{OC}}(SoC(k))}{\partial SoC}\right]\Bigg|_{SoC(k)}, \tag{13}$$

$$\mathbf{A}(k) = \left.\frac{\partial x(k+1)}{\partial x(k)}\right|_{x(k),u(k)} = \mathbf{I} + T_s \left.\frac{\partial f(x(k),u(k))}{\partial x(k)}\right|_{x(k),u(k)} = \begin{bmatrix} 1-\frac{T_s}{R_1 C_1} & 0 & 0 \\ 0 & 1-\frac{T_s}{R_2 C_2} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{14}$$

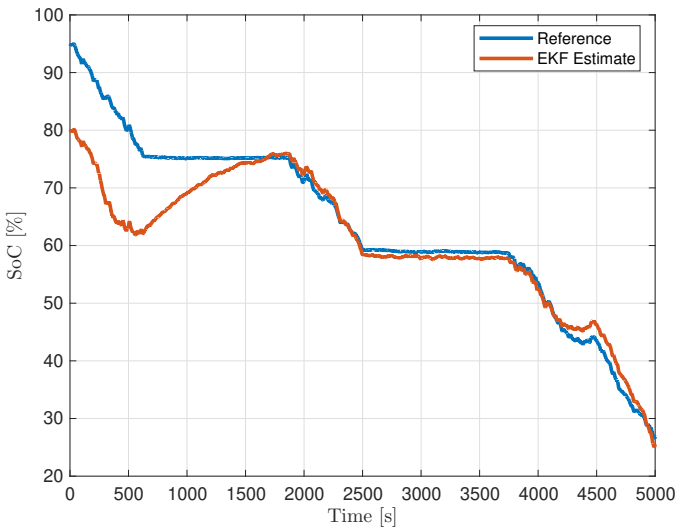
Although the state matrix $\mathbf{A}$ is time invariant, the last element of the output matrix $\mathbf{C}(k)$ is the derivative of the cell's OCV curve that is highly dependent on the immediate $SoC(k)$. The necessity of existence of the partial derivative $\frac{\partial U_{\text{OC}}(SoC(k))}{\partial SoC}$ makes the lookup table implementation of the OCV curve disadvantageous and calls for a polynomial fit – in this case approximation by a polynomial of the 5th degree shown in Fig. 29 together with its derivative.

The single element of measurement noise covariance matrix $\mathbf{R}$ was already determined during the analysis of the additive measurement noise e in Section I as $\mathbf{R} = \sigma^2 \approx 0.0013$. Tuning the process noise covariance matrix $\mathbf{Q}$ required several iterations and resulted in

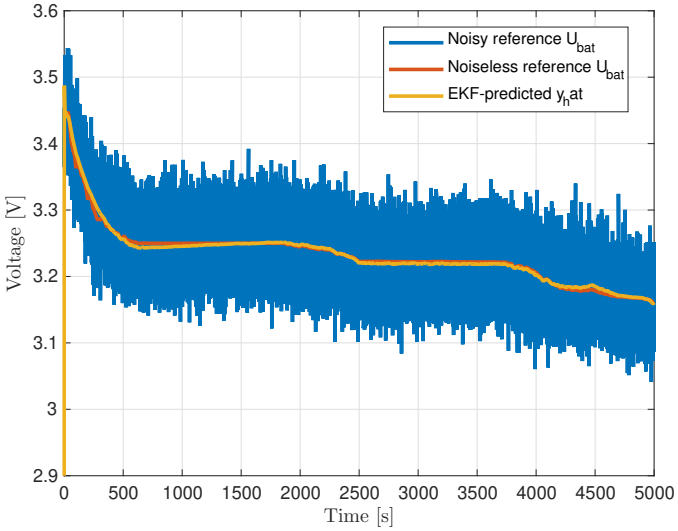
$$\mathbf{Q} = \begin{bmatrix} 10^{-12} & 0 & 0 \\ 0 & 10^{-12} & 0 \\ 0 & 0 & 10^{-4} \end{bmatrix}. \tag{15}$$

Considering the range of reasonable values of system states and other quantities, elements of $\mathbf{Q}$ seem very small and this fact could lead to deterioration of computation accuracy. Should this ever become a problem, one could normalize system variables to obtain a more robust numerical stability. Nevertheless in this case one can easily justify matrix elements this small – there is no reason to consider any noise apart from the additive measurement noise e characterized by σ^2 that was calculated exactly using provided data. The process noise only has to cover unmodelled system dynamics. Due to symmetry, there is no reason to choose different values of variances corresponding to U_1 and U_2 and both should be far lower than the variance corresponding to SoC. This is because we want the EKF to primarily adjust the SoC estimate to explain the observed output, whereas estimates of both polarization voltages U_i should rely mostly on the model and should not be varied by the EKF to explain observed terminal voltages.

Results (estimates) given by the EKF are shown in Fig. 30. Great gradual convergence to the true state of charge is shown in Fig. 30(a), whereas Fig. 30(b) shows great rejection of the additive measurement noise, as the predicted output voltage almost perfectly matches the noiseless reference output with RMSE as low as 16 mV when calculated globally and only 4 mV when calculated from $t = 2000$ onwards, where the filter has perfectly converged to the true state.



((a)) State of charge SoC.



((b)) Terminal voltage U_{bat}.

Fig. 30: Comparison of reference waveforms with EKF estimates/predictions.

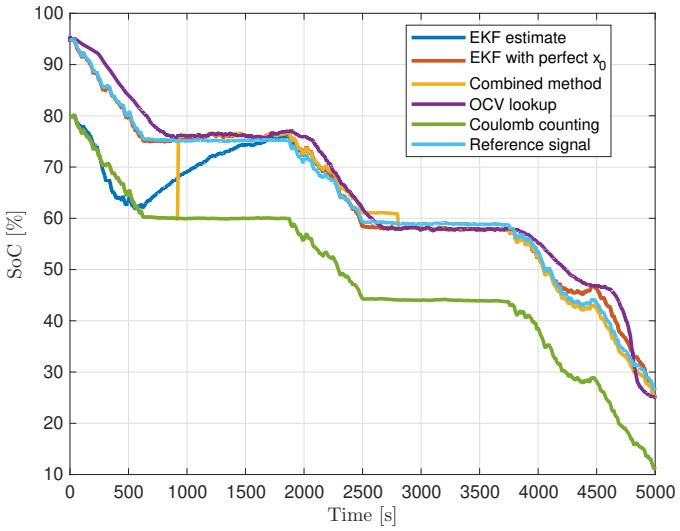
III. Discussion

This homework is a continuation of the previous report (HW7) that presented implementation and results for three simple methods of SoC estimation. It presented many results and observations that will not be repeated, the reader is therefore advised to read its conclusions before proceeding.

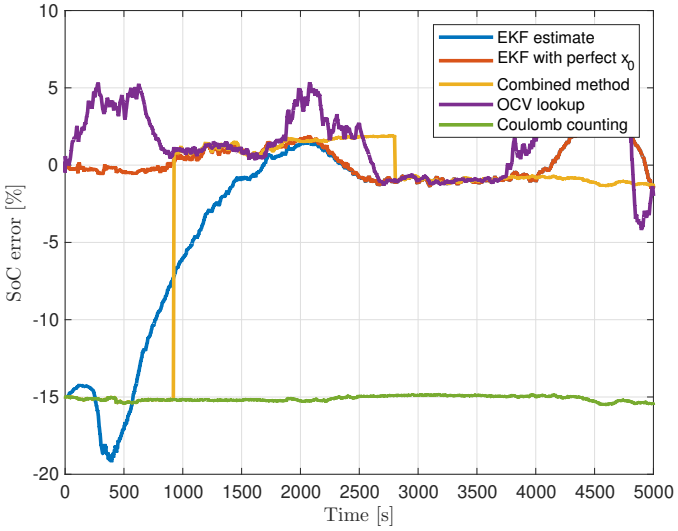
This report added a new SoC estimation method based on the Extended Kalman Filter. All methods are compared in Fig 31 with results summarized in IV. Compared to simple methods from the previous assignment, estimation by the Extended Kalman Filter enjoys a balance of robustness to the selection of initial conditions and accuracy of predicted system behavior. Observing Fig. 31(b), one may conclude that

1) Even when the EKF is started with incorrect initial conditions, it eventually converges to the correct solution (unlike the Coulomb counting method), and
2) once the EKF converges to the reference value, it never deviates far (unlike the U_{OC} lookup method).

Although the OCV-lookup method achieves the lowest RMSE on the whole experiment (with the exception of EKF with perfect knowledge of initial conditions), this is caused by its ultimate lack of dynamics – if the mean of terminal voltage U_{bat} was subject to more dynamical changes (e.g. due to higher current i), the OCV lookup estimate would quickly deteriorate as the approximated U_{OC} would be incorrect. On the other hand, the Coulomb counting has the clear disadvantage of no



((a)) Absolute values of SoC estimates.



((b)) Deviations of SoC estimates from the reference.

Fig. 31: Comparison of estimation performance of individual algorithms.

Method	SoC RMSE [%]	SoC RMSE [%] for $t > 2000$ s
Extended Kalman Filter	6.45	1.38
Coulomb Counting	15.11	15.04
U_{OC} lookup	2.93	3.14
Reset-Coulomb counting	6.62	1.23
Extended Kalman Filter w/ perfect ICs	1.19	1.40

TABLE IV: Errors of estimates yielded by various methods.

correction for imprecise knowledge of initial conditions. Coulomb counting with reset solves this problem, but the correction around $t \approx 900\,\mathrm{s}$ is too sudden and would certainly lead to customer dissatisfaction if it happened e.g. in a smartphone or an electric vehicle. Additionally the correction performed by the reset is only as good as the OCV-lookup method, disadvantages of which were already discussed in detail in the previous report.

For these reasons, the EKF algorithm is the winner in terms of quality of the resulting estimate. One should note however that the implementation of EKF is significantly more complex and computationally demanding than any of the simpler previously discussed methods. The EKF estimate could be further improved e.g. by using a higher-order fit of the cell's U_{OC} curve, since the currently used model seems to deviate a little from the true waveform for $SoC < 50$ %.

Week 9 – SOH Estimation

Abstract

This report presents an implementation of a dual-Extended Kalman Filter architecture used for state of health (SoH) estimation. An EKF for SoC estimation using measured terminal voltage presented in the previous lab report is complemented by a second two-state EKF used for the estimation of battery capacity C used to assess the state of health.

I. Model implementation

The SoH estimation architecture requires two separate models – the battery and the slow aging process. Just like in the previous assignment, the battery model was chosen as the standard 2RC equivalent circuit model

$$\dot{U_1} = -\frac{1}{R_1 C_1} U_1 + \frac{1}{C_1} i, \tag{16}$$

$$\dot{U_2} = -\frac{1}{R_2 C_2} U_2 + \frac{1}{C_2} i, \tag{17}$$

$$\dot{SoC} = -\frac{100}{C} i, \tag{18}$$

$$U_{\text{bat}} = U_{\text{OC}}(SoC) - U_1 - U_2 - R_0 i, \tag{19}$$

with a nonlinear output equation, where the system input i denotes the flowing current, states U_1 and U_2 are voltage drops across the two RC elements, $SoC \in [0, 100]$ % is the state of charge, U_{bat} is the battery terminal voltage (system output) and the open circuit voltage $U_{\text{OC}}(SoC)$, resistances $R_{0,1,2}$ and capacitances $C_{1,2}$ are (possibly SoC-dependent) model parameters with numeric values provided as a part of the assignment. In this task the total capacity C is not assumed constant and is estimated by the second EKF. To facilitate a simple iterative simulation of system behavior, the system (16)-(19) was discretized using the Forward Euler method as

$$\begin{aligned} x(k+1) &= x(k) + T_s f(x(k), u(k)), \\ y(k) &= g(x(k), u(k)), \end{aligned} \tag{20}$$

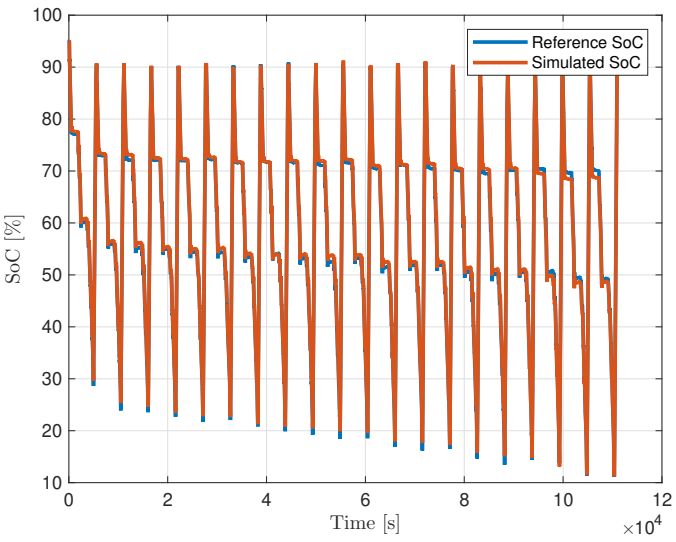
where f and g are vector functions of right-hand sides of (16)-(19) and $x(k)$, $u(k)$ and $y(k)$ are the system state, input and output vector at sample k, respectively.

Battery aging is modelled as a gradual decrease of the available capacity C. There are no degradation dynamics modelled

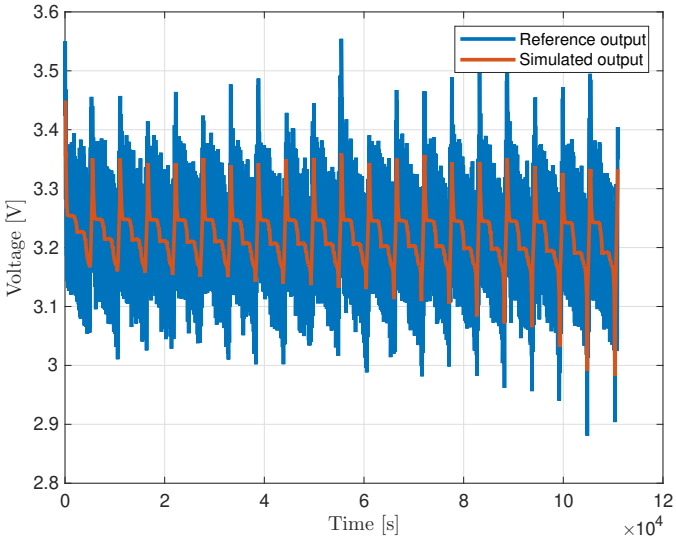
$$C(k+1) = C(k) + v_1(k), \tag{21}$$

$$\hat{SoC}(k+1) = \hat{SoC}(k) - \frac{1}{C(k)} i(k) + v_2(k), \tag{22}$$

hence any change in the battery capacity C originates from the influence of process noise $v(k)$. The quantity $\hat{SoC}$ is a secondary estimate of battery SoC calculated by the SoH-EKF. It acts as an output of the SoH-EKF that is compared to the SoC estimate given by SoC-EKF to obtain the prediction error. This prediction error then drives the SoH-EKF to estimate the correct battery capacity.



((a)) State of charge SoC.



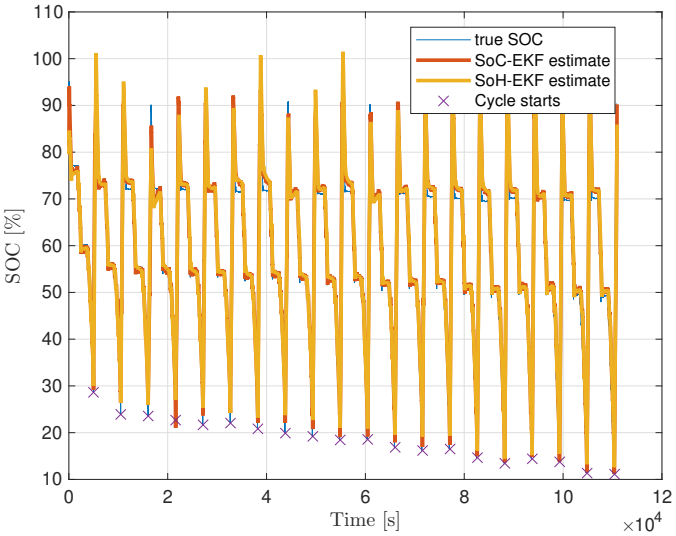
((b)) Terminal voltage U_{bat}.

Fig. 32: Comparison of reference and simulated waveforms.

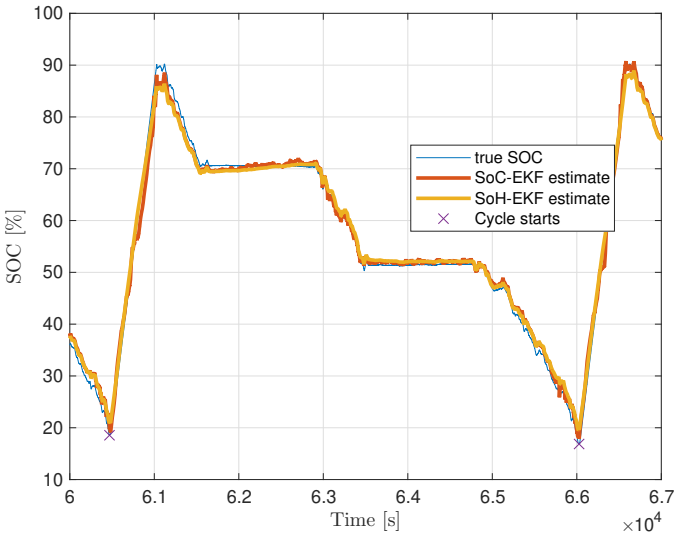
II. Experimental results

Equations were implemented in code to verify that the model and provided parameters work correctly and can reproduce the reference waveforms. A comparison of reference (given) and my simulated signals is in Fig. 32. Signals match quite well, indicating that the most important most of the system dynamics are captured. There are still some uncertainties; for example, it is unclear from the assignment where within a cycle is the reference battery capacity (provided as one sample per charge-discharge cycle) recorded. Apart from that, it is unspecified whether the capacity in the reference simulation is piecewise constant or linearly decreasing during the cycle. These uncertainties about the provided data may result in significant errors.

The dual EKF architecture was implemented regardless of the lack of information. Best results achieved using covariance matrices $\mathbf{R}_{\text{SoH}} = 10^{-3}$, $\mathbf{R}_{\text{SoC}} = 0.05$, $\mathbf{Q}_{\text{SoC}} = diag(10^{-7}, 10^{-6}, 3 \cdot 10^{-2})$ and $\mathbf{Q}_{\text{SoH}} = diag(4 \cdot 10^{-13}, 10^{-10})$ are shown in Fig. 33. Both EKFs track the reference SoC with RMSE of 2.4 % over the whole experiment. The cell capacity C estimated by SoH-EKF is shown in Fig. 34. The EKF captured the slope of overall decrease quite well, but it could still use some further tuning to smoothen the estimated capacity. A possible improvement could introduce a new state, effectively applying a lowpass filter with specified time constant on the estimate of battery capacity. Either way, one can see the theoretical strenght of the dual EKF architecture for SoH estimation since the estimated capacity decreases with the same gradient as the reference.



((a)) SoC For the whole experiment.



((b)) Detail of a single discharge-charge cycle.

Fig. 33: SoC estimates given by the dual EKF architecture.

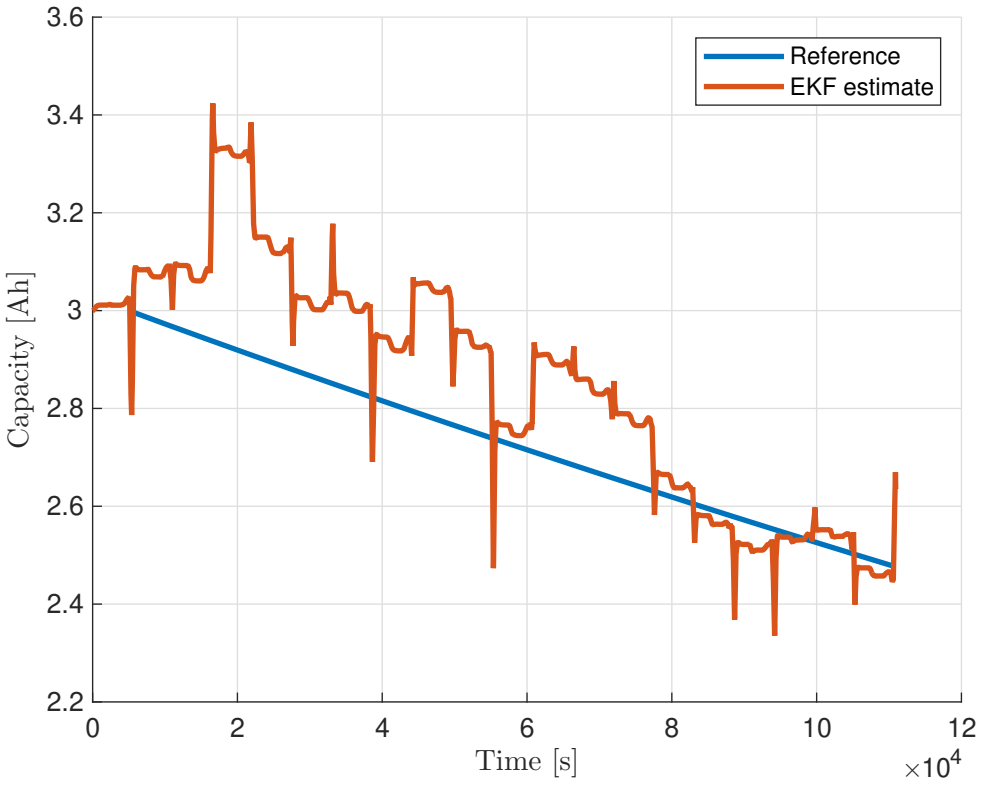


Fig. 34: Estimated and reference battery capacity

Week 10 – Online Parameter Estimation

Abstract

This report presents an implementation of two methods for online estimation of battery model parameters. Recursive Least Squares (RLS) take a discrete-time transfer function and search for values of its coefficients. An Extended Kalman Filter (EKF) on the other hand considers the system in the state-space augmented by auxiliary states representing the – possibly time-varying – system parameters and estimates those. Both methods were derived for the battery equivalent circuit model with a single RC element and tested using a recording of a dynamic discharge profile.

I. Recursive Least Squares

For simplicity, consider the RLS algorithm for a simple ARX model, i.e. a general discrete-time transfer function in the time-domain delay operator d

$$G(d) = \frac{Y(d)}{U(d)} = \frac{b_0 + b_1 d + \cdots + b_{n_b} d^{n_b}}{1 + a_1 d + \cdots + a_{n_a} d^{n_a}} = \frac{\sum_{i=0}^{n_b} b_i d^i}{\sum_{i=0}^{n_a} a_i d^i}, \tag{23}$$

usually normalized to unity a_0, i.e. the coefficient belonging to $y(k)$ in the corresponding difference equation

$$y(k) + \sum_{i=1}^{n_a} a_i y(k-i) = \sum_{i=0}^{n_b} b_i u(k-i) + e(k) \tag{24}$$

relating past samples of the input u and output y up to the discrete time k. The variable $e(k)$ encapsulates stochastic phenomena such as noise, as well as unmodelled system dynamics.

The most recent output $y(k)$ can be predicted from (24) using past inputs and outputs and system parameters θ as

$$\hat{y}(k) = -\sum_{i=1}^{n_a} a_i y(k-i) + \sum_{i=0}^{n_b} b_i u(k-i), \tag{25}$$

that can be seen as a dot product

$$\hat{y}(k) = \varphi^T(k)\theta(k) \tag{26}$$

of the regressor vector

$$\varphi(k) = \begin{bmatrix} -y(k-1) & -y(k-2) & \dots & -y(k-n_a) & u(k) & u(k-1) & \dots & u(k-n_b) \end{bmatrix}^T \tag{27}$$

containing past inputs and outputs and the vector of parameters

$$\theta(k) = \begin{bmatrix} a_1 & a_2 & \dots & a_{n_a} & b_0 & b_1 & \dots & b_{n_b} \end{bmatrix}^T. \tag{28}$$

At each iteration k of the RLS algorithm, the actual measured output $y(k)$ is compared with the prediction $\hat{y}(k)$ from (26) to find an error $e(k)$. Then the Kalman gain K is calculated using the current estimate covariance matrix $\mathbf{P}$ and finally both the estimated system parameters $\hat{\theta}$ as well as the covariance matrix $\mathbf{P}$ are updated using K and e. The algorithm can be extended by a real forgetting factor $\lambda \in [0, 1]$ used to artificially inflate the covariance matrix $\mathbf{P}$ to facilitate tracking of time-varying parameters.

The algorithm can be generalized and applied far beyond the limitation of linear ARX model. To use the RLS algorithm for any general problem, one needs to

1) express the prediction task using an equation linear-in-parameters ,
2) determine what expressions are to be estimated and belong to the parameter vector θ
3) determine what expressions are known and belong to the regressor φ,
4) find the (generally non-linear) mapping between parameters θ and physically meaningful parameters useful to the application.

A. RLS for battery parameters

The continuous-time 1 RC equivalent circuit model reads

$$U_{\text{bat}} = U_{\text{OC}} - i\,R_0 - \underbrace{i\frac{R_1}{C_1\,R_1\,s+1}}_{U_1}, \tag{29}$$

where i is the flowing current, U_{bat} is the measured terminal voltage, U_{OC} is the cell's open-circuit voltage, R_j, C_j are parameters of the model and s is the Laplace transform operator. By relabeling $E = U_{\text{bat}} - U_{\text{OC}}$, (29) can be rewritten as a transfer function

$$G_{\text{cont}}(s) = -R_0 - \frac{R_1}{C_1\,R_1\,s+1} \tag{30}$$

from the flowing current i to the voltage drop E. The Tustin (bilinear) discretization of $G_{\text{cont}}(d)$ with step T gives

$$G(d) = \frac{(R_0\,T + R_1\,T - 2\,C_1\,R_0\,R_1)\,d + R_0\,T + R_1\,T + 2\,C_1\,R_0\,R_1}{(2\,C_1\,R_1 - T)\,d - T - 2\,C_1\,R_1}. \tag{31}$$

Coparing (31) to a general first-order discrete-time transfer function

$$\frac{b_1 d + b_0}{a_1 d + 1} \tag{32}$$

yields substitutions

$$a_1 = \frac{T - 2\,C_1\,R_1}{T + 2\,C_1\,R_1}, \tag{33}$$

$$b_1 = -\frac{R_0\,T + R_1\,T - 2\,C_1\,R_0\,R_1}{T + 2\,C_1\,R_1}, \tag{34}$$

$$b_0 = -\frac{R_0\,T + R_1\,T + 2\,C_1\,R_0\,R_1}{T + 2\,C_1\,R_1}. \tag{35}$$

Using the discretized transfer function $G(d)$, (29) can be rewritten in several steps

$$U_{\text{bat}} - U_{\text{OC}} = iG(d), \tag{36}$$

$$(U_{\text{bat}} - U_{\text{OC}})(a_1 d + 1) = i(b_1 d + b_0), \tag{37}$$

$$U_{\text{bat}} + U_{\text{bat}} a_1 d - U_{\text{OC}}(a_1 d + 1) = i(b_1 d + b_0), \tag{38}$$

$$U_{\text{bat}} = U_{\text{OC}}(a_1 d + 1) - U_{\text{bat}} a_1 d + i(b_1 d + b_0). \tag{39}$$

Finally, assuming that the ΔU_{OC} is negligible on each sampling interval and therefore $U_{\text{OC}}(k) \approx U_{\text{OC}}(k-1)$, (39) can be converted from the d-operator domain to index shifts

$$U_{\text{bat}}(k) = (a_1 + 1)U_{\text{OC}} - a_1 U_{\text{bat}}(k-1) + b_0 i(k) + b_1 i(k-1). \tag{40}$$

Equivalently in the short vector notation

$$U_{\text{bat}}(k) = \varphi^T(k)\theta, \tag{41}$$

where

$$\theta = \begin{bmatrix}(a_1+1)U_{\text{OC}} & a_1 & b_0 & b_1\end{bmatrix}^T, \tag{42}$$

$$\varphi(k) = \begin{bmatrix}1 & -U_{\text{bat}}(k-1) & i(k) & i(k-1)\end{bmatrix}^T. \tag{43}$$

The problem at hand was successfully described using an equation (40) that is linear in parameters θ and hence can be subject to RLS. Equations (42) and (43) describe the role of individual variables (θ are estimated) and equations (34) through (35) describe the nonlinear mappings from physically meaningful battery parameters to elements of θ with the inverse mapping given by

$$R_0 = \frac{b_0 - b_1}{a_1 - 1}, \tag{44}$$

$$R_1 = \frac{2\,(b_1 - a_1\,b_0)}{{a_1}^2 - 1}, \tag{45}$$

$$C_1 = -\frac{T\,{a_1}^2 - 2\,T\,a_1 + T}{4\,(b_1 - a_1\,b_0)}. \tag{46}$$

Performance of the estimation algorithm is presented in Section III.

II. Extended Kalman Filter

The EKF was discussed multiple times in previous reports, mainly as a great tool for SoC estimation. This time, it is used to estimate states of a state-space model augmented with model parameters. The 1 RC ECM from (29) can be rewritten as a state-space

$$\dot{U_1} = -\frac{1}{R_1 C_1} U_1 + \frac{1}{C_1} i \tag{47}$$

$$U_{\text{bat}} = U_{\text{OC}} - iR_0 - U_1 \tag{48}$$

with input i, output U_{bat} and state U_1 (the polarization voltage). To estimate some model parameter M, it must be added to the list of system states. Without any apriori knowledge of parameter evolution, $\dot{M} = 0$ is assumed such that the parameter evolution is driven only by a random walk of the process noise. Two distinct cell state-space representations based on the 1 RC equivalent circuit model were implemented and tested – the reference model "Hongwen" with six states from [1] and my own manually derived model "Michal" with five states. Both models have the flowing current i as input u and the terminal voltage U_{bat} as output y.

A. Hongwen model

The Hongwen model considers six states

$$x = \begin{bmatrix} U_{\text{OC}} & U_{\text{bat}} & U_1 & \frac{1}{C_1} & \frac{1}{R_1} & R_0 \end{bmatrix}^T \tag{49}$$

with state equations

$$\begin{aligned} \dot{x_1} &= 0, \\ \dot{x_2} &= x_1 x_4 x_5 - x_2 x_4 x_5 - (x_4 x_5 x_6 + x_4) u - x_6 \dot{u}, \\ \dot{x_3} &= x_4 u - x_3 x_4 x_5, \\ \dot{x_4} &= 0, \\ \dot{x_5} &= 0, \\ \dot{x_6} &= 0. \end{aligned} \tag{50}$$

This model includes the terminal voltage U_{bat} as state x_2, making the output equation trivial

$$y = x_2. \tag{51}$$

This model is quite complex and requires a non-standard addition of $\dot{u}$ (the derivative of input) that may introduce complications of its own.

B. Michal model

The Michal model eliminates the terminal voltage U_{bat} and considers only five states

$$x = \begin{bmatrix} U_{\text{OC}} & U_1 & \frac{1}{C_1} & \frac{1}{R_1} & R_0, \end{bmatrix}^T \tag{52}$$

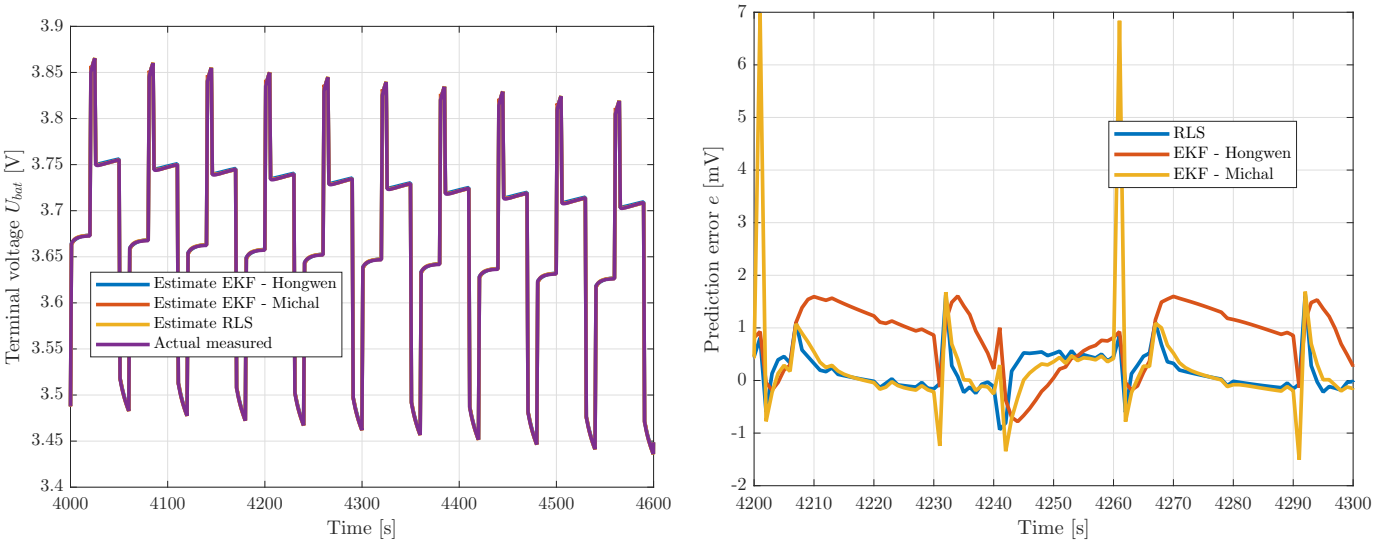
with state equations

$$\begin{aligned} \dot{x_1} &= 0, \\ \dot{x_2} &= x_3 u - x_2 x_3 x_4, \\ \dot{x_3} &= 0, \\ \dot{x_4} &= 0, \\ \dot{x_5} &= 0 \end{aligned} \tag{53}$$

and a non-linear output equation

$$y = x_1 - x_2 - u x_5. \tag{54}$$

This model has fewer states, does not require the numeric differentiation of input u and is generally much simpler than the Hongwen model. Results of online parameter estimation with both models are presented in Section III.



((a)) Over several DDP cycles.

((b)) Detail of one DDP period.

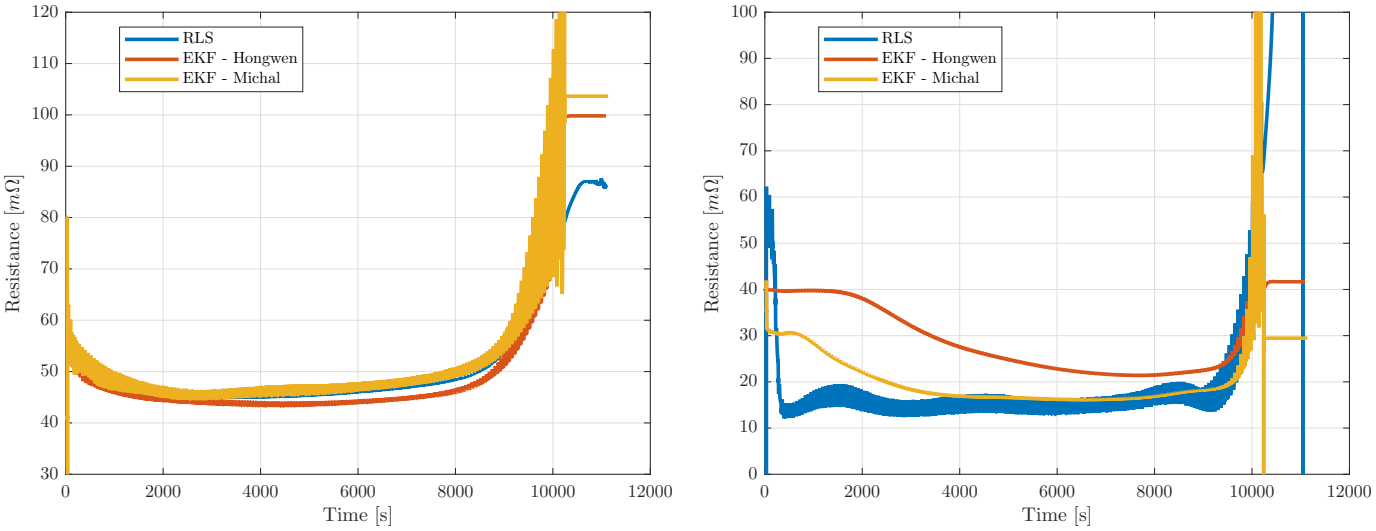
Fig. 35: Comparison of terminal voltages predicted by individual algorithms.

III. Estimation results

All three methods (RLS estimation from Section I, EKF for Hongwen model from Section II-A and EKF for Michal model from II-B) were compared using waveforms from a dynamic discharge profile. The achieved prediction error is shown in Fig. 35; Fig. 35(a) shows predictions throughout several DDP cycles whereas Fig. 35(b) focuses on one DDP period. Low prediction error less than 10 mV confirms good convergence of model parameters to the behavior of the actual battery, although there is certainly some room for improvement (e.g. by introducing a second RC element).

Estimates of both components of the ohmic (real) impedance R_0 and R_1 are shown in Fig. 36(a) and Fig. 36(b), respectively. Estimates of R_0 yielded by various algorithms are quite similar and they demonstrate the expected (approximately) flat plateau roughly in the range of 80 % to 20 % of SoC with growing value towards extremes. On the other hand, estimates of R_1 shown in Fig. 36(b) did not converge as satisfactorily with a significant discrepancy between both EKFs.

The estimated capacitance of the RC element shown in Fig. 37 can be used to demonstrate an important difference between the RLS- and EKF-based parameter estimation. The EKF method allows the user to directly specify variance for individual physically meaningful parameters (albeit possibly inverted, such as C_1^{-1}). In contrast, the RLS works in terms of some artificial parameters b_i and a_i, whose dependence on physically meaningful parameters is non-linear even for a 1 RC ECM and becomes



((a)) Estimates of internal impedance R_0.

((b)) Estimates of RC element resistance R_1.

Fig. 36: Estimates of resistive ECM parameters.

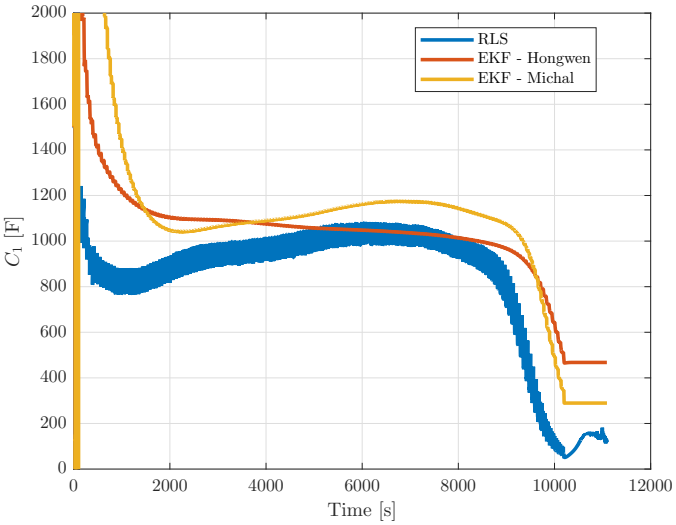


Fig. 37: Estimates of RC element capacitance C_1.

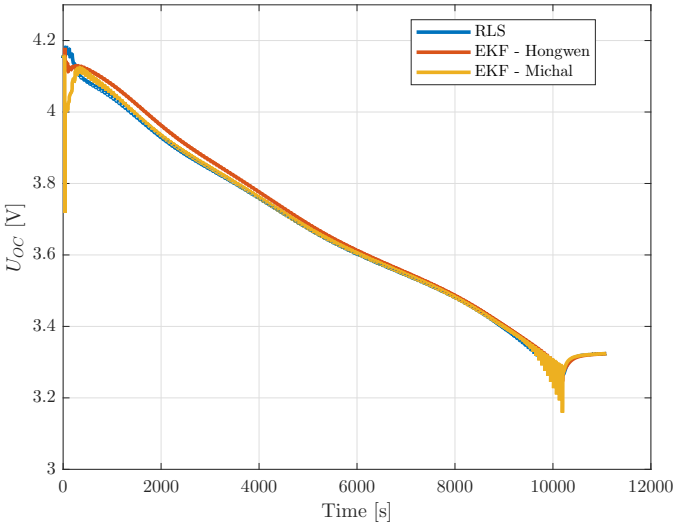


Fig. 38: Estimates of the open circuit voltage U_{OC}.

increasingly more entangled with the growing number of parameters. Therefore, it is possible to get a smooth waveform for some given parameter (here C_1) with the EKF simply by reducing the parameter's variance. When estimating using RLS on the other hand, such fine control is very hard to do and the resulting estimate tends to be more noisy. A similar phenomenon can be observed in Fig. 36(b).

All three methods estimate the U_{OC} (shown in Fig. 38) similarly well. The main difference from the implementation point of view is that the EKF method was far simpler to set up and debug owing to the tight bound of individual states to underlying physics. In contrast, the RLS method uses artificial (physically meaningless) parameters whose values must be post-processed to obtain any physically useful information. Typos such as an incorrect sign in some of the RLS expressions may cause a lot of headaches. Nevertheless, it is computationally less demanding than the EKF estimation and the forgetting factor λ allows an intuitive configuration of the steady-state covariance matrix.

It is still unknown to me why the authors of [1] decided to use an unnecessarily complicated state space model for the parameter estimation task. Despite my best efforts in hyperparameter tuning (especially tuning the process noise covariance matrices Q), the simpler state-space model Michal achieved more credible results (closer matching the RLS estimates that I see as the best available reference in the absence of some ground truth parameter values) while being easier to debug and implement.

Week 12 – Data-Driven Methods

Abstract

This report presents results of an experiment with data-driven methods and their use in battery modelling and prediction tasks. Using a dataset of 380 charging and discharging cycles from accelerated cell aging, a Random Forest regressor is trained to predict the remaining capacity of the cell (essentially predicting the SOH-C) using only a few pieces of information (features) from the full charging cycle.

I. Familiarization with the environment

In contrast to previous lab reports relying heavily on Matlab live scripts, this one was prepared using python in the Jupyer notebook environment. All tasks are performed on a dataset from the University of Michigan[1]. It contains measurements such as cell voltage, current or temperature from 380 full battery cycles periodically sampled every 10 seconds. Since the task only deals with charging half-cycles, all time instants with current $i < 0\,\mathrm{A}$ are removed from the working dataset.

Most of the processing was done using the python library `pandas`. It provides means of loading the dataset using `pd.read_csv`, accessing individual signals using `dataset[signal_name]` and performing logical indexing (slicing) by `dataset[boolean_mask]`. This is especially useful for operations such as extracting a particular full cycle or detecting discharging instants. Additionally, `matplotlib.pyplot` was used for visualization of signals.

The learning part of this lab was performed using the popular python package `scikit-learn`. It provides interface to many machine learning constructs such as metrics (loss functions, e.g. `mean_squared_error`), utilities such as `train_test_split` for simple creation of evaluation datasets for cross-validation, or the actual trainable models, such as the ensemble model `RandomForestRegressor` used in this task.

II. Training the regressor

To verify that the provided dataset can be used for the training of an SOH-C predictor, one can plot and analyze measurements from several cycles. Fig. 39 through 41 show the recorded cell voltage, flowing current and capacity, respectively. The charging mode is clearly CC-CV with current limit of 1000 mA and target voltage of 4.2 V. It is apparent from all figures/12 that with a growing number of elapsed cycles, the cell ages and loses capacity, making subsequent cycles shorter (the voltage in Fig. 39 rises faster and the current in 40 drops earlier). Additionally, the total capacity gradually decreases, as shown in Fig. 42. There are several outlier cycles that were interrupted early for unspecified reasons.

Out of the total of 380 recorded cycles, 19 were randomly selected for final testing. The rest was divided into the training and validation dataset according to the 80:20 Pareto rule. Instead of using all collected samples (roughly 2000 of them per signal per cycle) as inputs to the machine learning algorithm, information from each cycle is condensed into a small vector of features. The Random Forrest is an ensemble algorithm for supervised learning that separately trains many decision trees on subsets of the training data and then combines (e.g. averages) their outputs when asked to predict the output for previously unseen data. In this task, the number of trees in the forest (an algorithm hyperparameter) was chosen as 100.

[1] available at https://deepblue.lib.umich.edu/data/downloads/gq67jr501

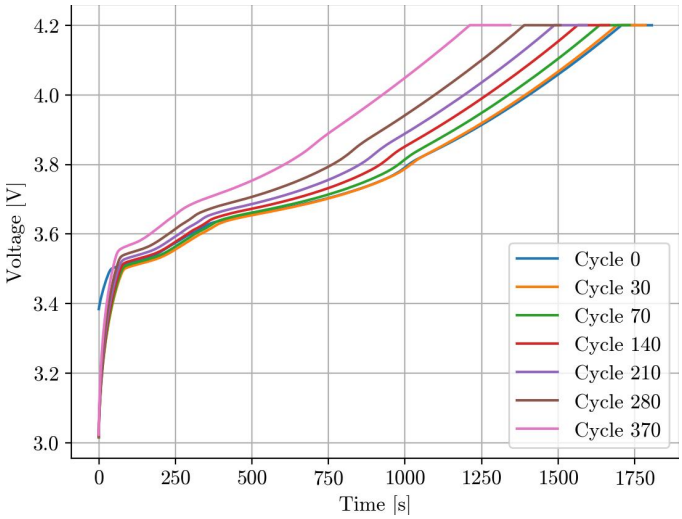


Fig. 39: Voltages during several full battery cycles.

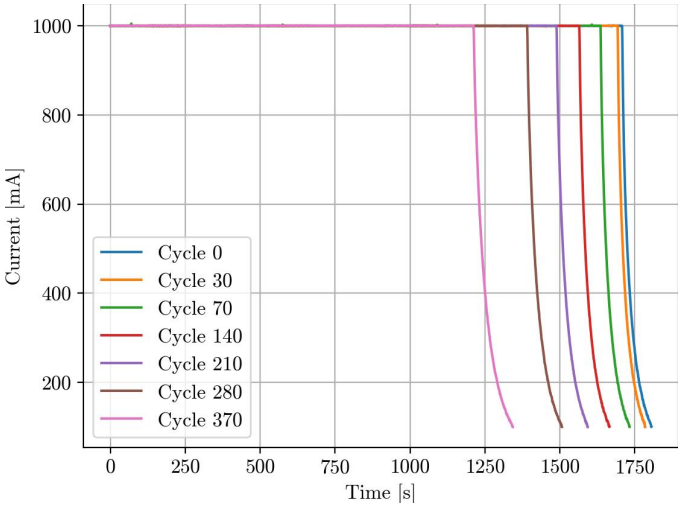


Fig. 40: Currents during several full battery cycles.

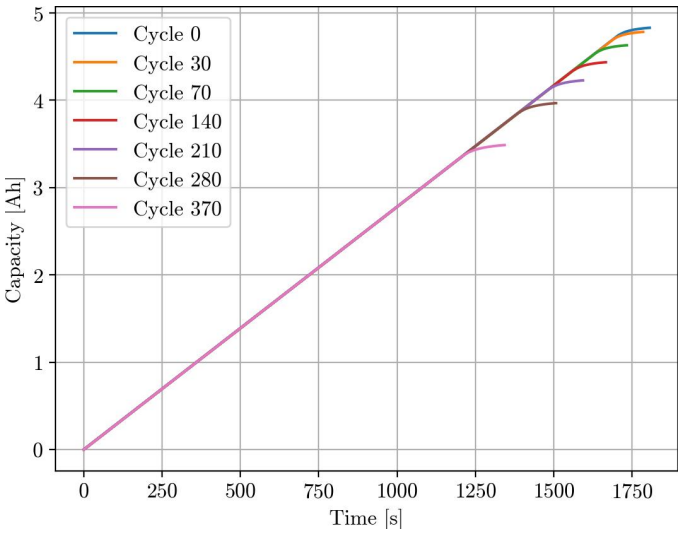


Fig. 41: Cell capacity during several full battery cycles.

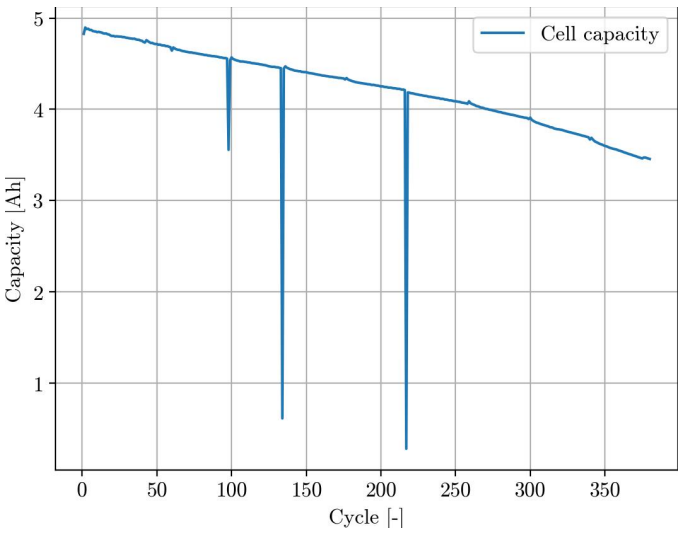


Fig. 42: The loss of total cell capacity due to aging.

Feature name	Mean importance	Standard deviation of importance
Temperature Delta	0.4250	0.3142
Number of Samples	0.3646	0.2996
Total Charge	0.1022	0.1987
Average Current	0.0411	0.1223
Maximal V	0.0360	0.1129
Current Standard Deviation	0.0216	0.0830
Average Temperature	0.0048	0.0329
Minimal V	0.0047	0.0333

TABLE V: Importance of individual features for individual trees in the ensemble model

A. *Selection of features*

The best performance (the lowest error on the test set) was achieved using the features listed in Table V in the decreasing order of mean importance. Said simply[2], the *Mean Decrease in Impurity (MDI)* used by `scikit` measures the importance of a feature as the number of decision nodes that branch paths based on said feature weighted by the probability of reaching said nodes. High feature importance means that many decision tree nodes are branching the path based on said feature and that they reside in busy subtrees of the whole tree hit by many input samples. On the other hand, negligible feature importance shows that the tree did not need to incorporate said feature into its decision nodes during training. Since this metric is calculated for each tree separately, the third column of Table V shows the standard deviation of importances across the whole forest. Note that many of these standard deviations are quite significant, hinting that indeed each tree puts emphasis on entirely different features.

The presence of the number of samples collected during the charging cycle among the most important features is expected, as it indirectly expresses the duration of charging half-cycle and that is significantly influenced by the cell's capacity when using CC-CV scheme, as shown e.g. in Fig. 40. However what is not expected at all is the high importance of the temperature delta. In order to explain it, measurements of the cell's internal resistance available in the dataset have to be used. The evolution of the real part of cell's impedance as a function of the available capacity over the course of the experiment is shown in Fig. 43. The ohmic resistance clearly grows (yet again proving that the dataset indeed describes a noticeably aging cell, in this case with respect to SOH-R), causing increased losses due to constant charging current, eventually resulting in greater temperature delta.

Some features on the other hand were identified by the algorithm as mostly irrelevant, e.g. the minimal and maximal voltage. This can be explained intuitively as well - all charging half-cycles end at 4.2 V (giving no useful information about the cell capacity) and the starting voltage is subject to slight unpredictable variation caused by transition from one cycle to the next.

B. *Evaluation of the Regressor*

The model's performance after training was evaluated using the test set. True cell capacities and the corresponding model predictions are shown in Fig. 44. The histogram of prediction errors shown in Fig. 45 suggests that the model tends to underestimate the true capacity slightly (by less than 15 mAh) with the exception of two outliers with grater prediction error.

[2]More detailed explanation of the feature importance can be found at https://medium.com/the-artificial-impostor/feature-importance-measures-for-tree-models-part-i-47f187c1a2c3 and in resources linked there.

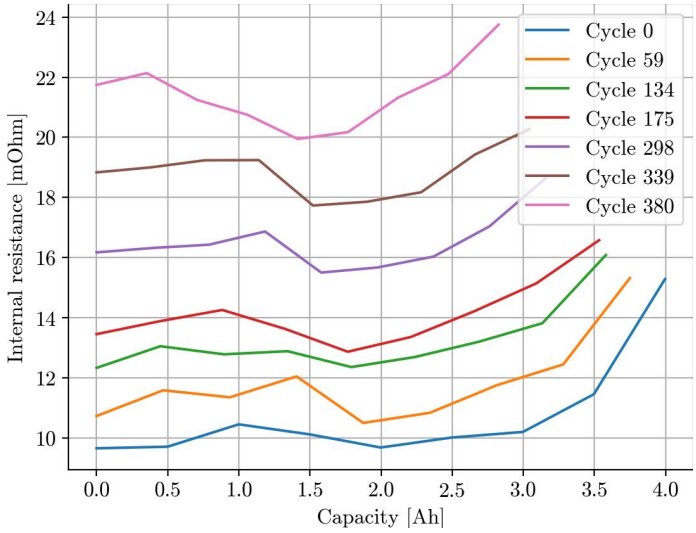


Fig. 43: Real part of the cell impedance during aging

Error metric	Magnitude
Maximal error	2.29e-02
Mean Absolute Deviation (MAD)	6.93e-03
Mean squared error (MSE)	8.35e-05

TABLE VI: Prediction error achieved on the test set

The algorithm's performance was assessed using several metrics listed in Table VI. Their values are derived from the prediction error expressed in Ah, i.e. the greatest single prediction error was 22.9 mAh and the average prediction error was 6.93 mAh.

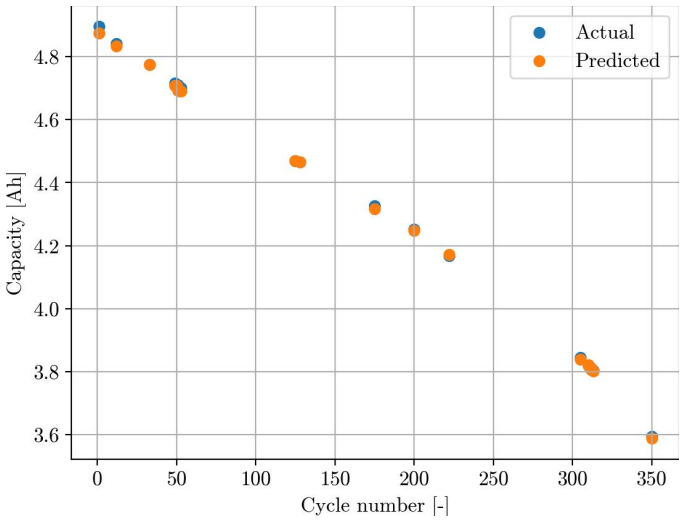


Fig. 44: Regressor performance on the test set.

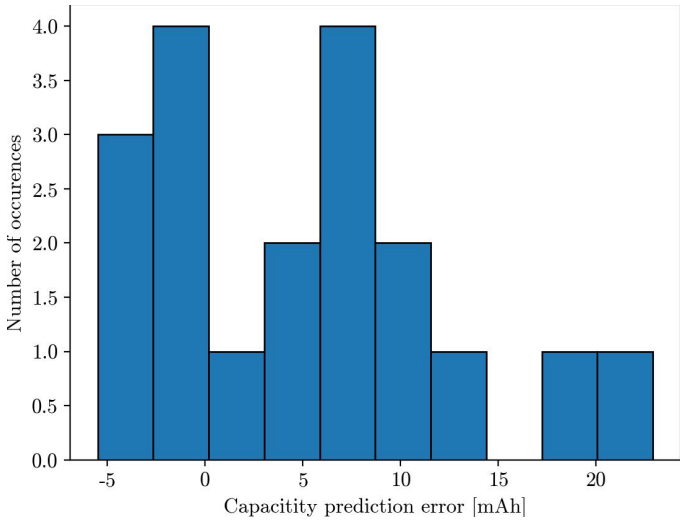


Fig. 45: Capacity prediction error on the test set.

Week 13 – Overcurrent Protection For A Modular System

Abstract

This report presents results of tuning a high-level supervisory algorithm that calculates a system-level current limit respecting the current limitation of individual modules connected in parallel. Different initial conditions of individual modules are assumed for generality. Two solutions are considered – a feedforward approach based on the closed-form expression for current upper and lower bound using the provided voltage, current and internal resistance of each module, and a feedback solution using a modified PID controller regulating the module current towards the module current limit.

I. Problem statement

The given Simulink model contains a model of two battery modules connected in parallel, each consisting of a 1RC equivalent circuit model (ECM). Each module is given a different initial SoC, as shown in fig. 47 and also different parameters of the ECM, such as internal impedance R_0 shown in Fig. 46. This alone results in some current asymmetry between both modules due to the flow of an equalizing current shown in Fig. 48. Since both modules are connected in parallel, their terminal voltages must be equal by definition. This is verified in Fig. 49, showing maximal voltage deviation in the order of 10^{-4} V.

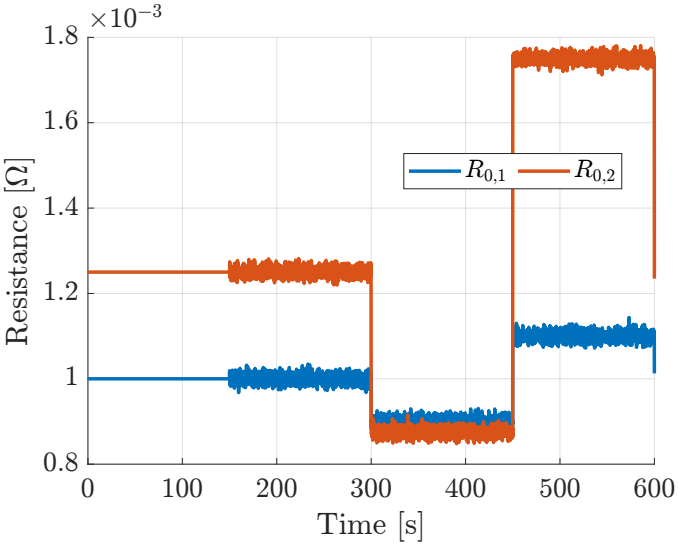


Fig. 46: Estimates of the module internal impedance available to the algorithm.

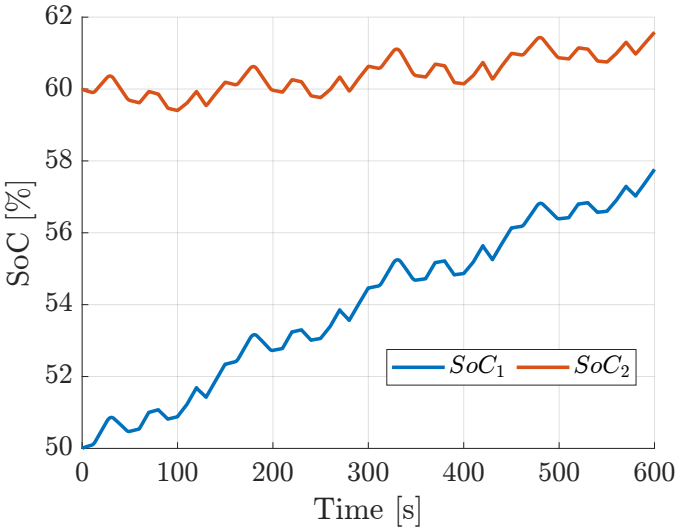


Fig. 47: Module state of charge (SoC) during the simulation.

Each module is given a current limit of 50 A for both charging and discharging. During the experiment, some connected system (e.g. the motor controller of an EV) requests current for vehicle propulsion and the battery system must restrict the current flow to a safe system-wide limit such that neither of the modules is overloaded.

The task is complicated by the measurement noise that is gradually added to the quantities provided to the control algorithm. Furthermore, the estimate of module impedance is subject to significant disturbance, resulting in large variations of the estimate of R_0 available for the current limit calculation, as shown in Fig. 46.

II. Analytical solution

The first solution utilized an algorithm based on a closed-form expression for the current limit. Measured module voltages, currents and impedances are used to calculate the equalizing current I_E. Then, considering the current divider formed by impedances $R_{0,1}$ and $R_{0,2}$, a set of inequalities for the upper bounds of the system current is constructed. The lowest bound is then used to prevent exceeding the current limit fo any module.

Results obtained by this algorithm are shown in Fig. 50 and 51. The algorithm has the potential to work very well when provided with correct information (as shown for $t < 150$ s), where the current through the first module is saturated perfectly at the module current limit. Nevertheless, as soon as the estimate of R_0 becomes noisy or completely wrong, the inherently

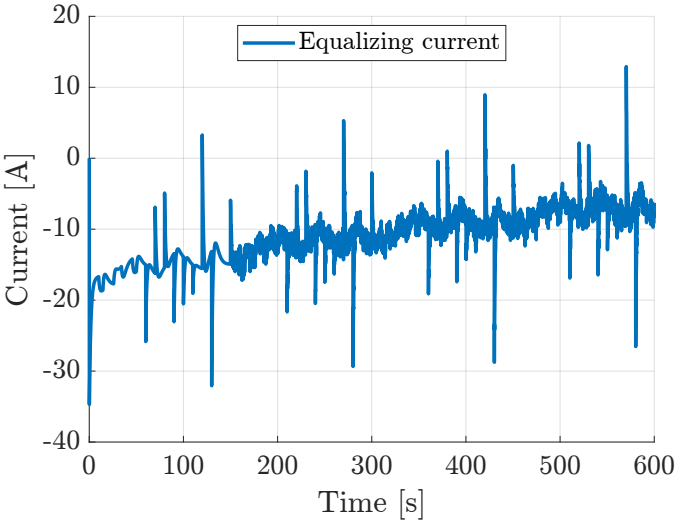


Fig. 48: Equalizing current flowing from module 1 to module 2.

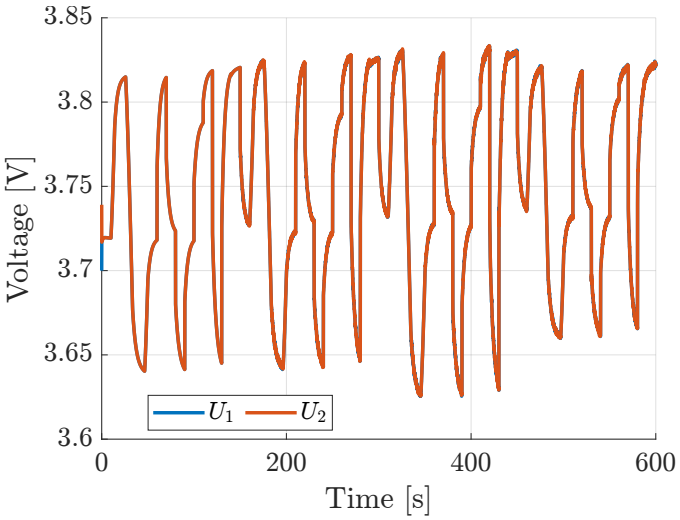


Fig. 49: Module terminal voltages.

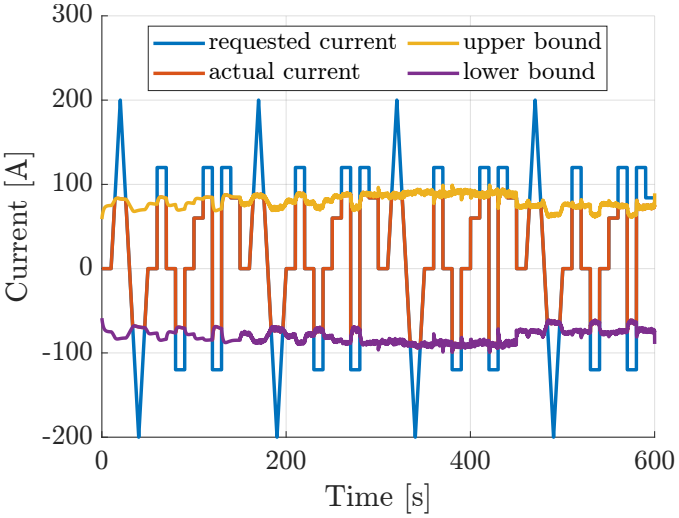


Fig. 50: System currents using the feedforward control algorithm.

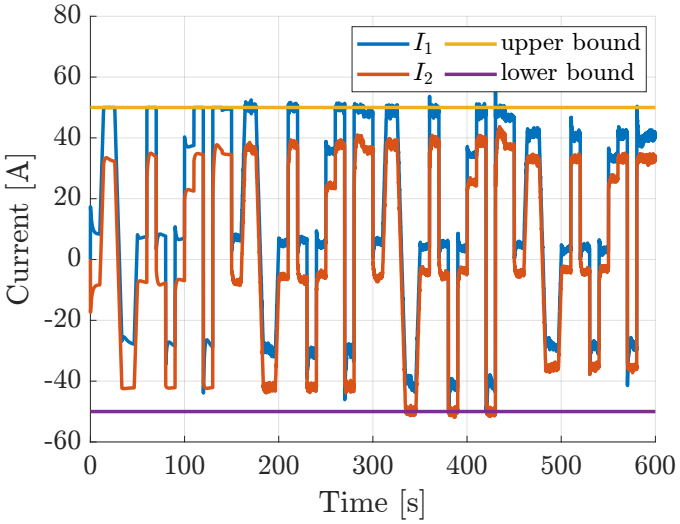


Fig. 51: Subsystem currents using the feedforward algorithm.

open-loop solution can't optimally deliver the requested current. This is illustrated for $t > 450$ s, when the system-wide current limit abruptly decreases due to the error of resistance estimates from Fig. 46.

Due to this inherent limitation, the algorithm was not even completely implemented and yielded incorrect results when approaching the negative lower bound. The alternative feedback solution achieved much better performance; hence, it was not worthwhile to attempt to fix the feedforward algorithm.

III. Feedback algorithm

The feedback algorithm uses a modified PID controller to determine the system-wide current limit. First, since the module current limit is symmetric for charging and discharging, the maximum of absolute values of both module currents is taken to simplify calculations by considering only the positive branch. This maximal current is subtracted from the module current limit to get an error that is fed to the discrete-time PID controller with filtered derivative. The output of the PID controller was saturated at twice the module current limit – that is the optimistic solution in case both modules are balanced and there is no equalizing current.

To further improve the behavior of the PID controller, the following trick was introduced – the error is scaled down by a factor of 1000 when it is positive. This way, the current limit is quickly adjusted down to prevent stress to battery modules, while the growth of the current limit is very slow to ensure smooth waveform.

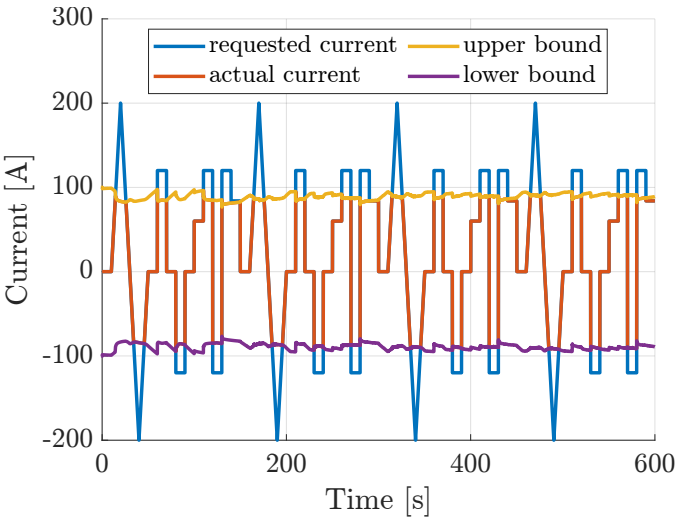


Fig. 52: System current using the feedback control algorithm.

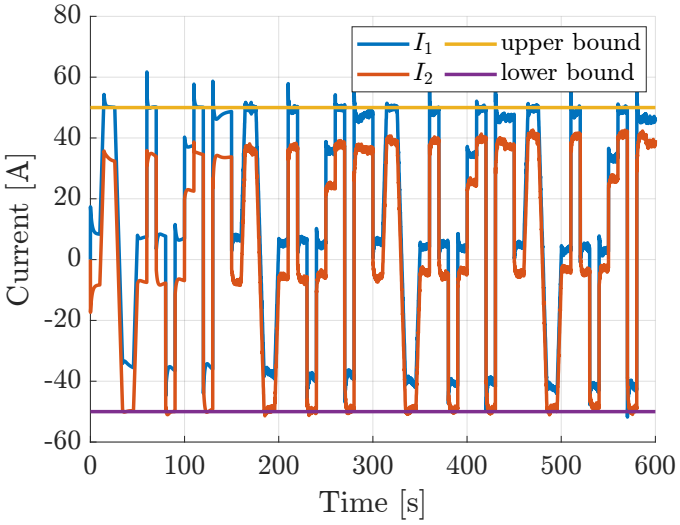


Fig. 53: Subsystem currents using the feedback algorithm.

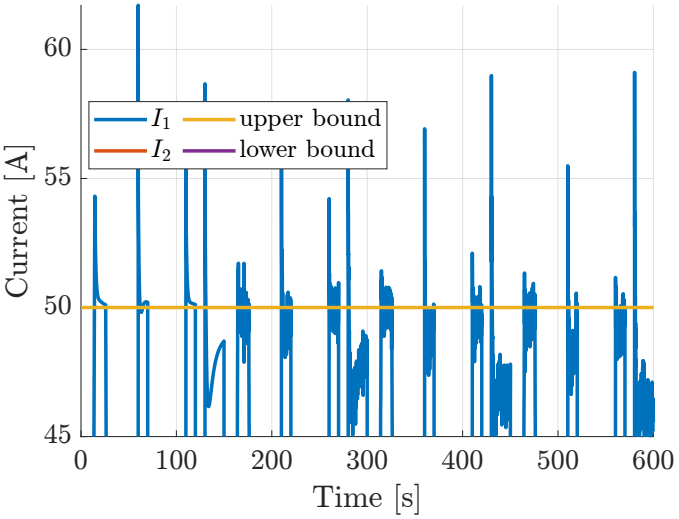


Fig. 54: Overshoots of the module current limit

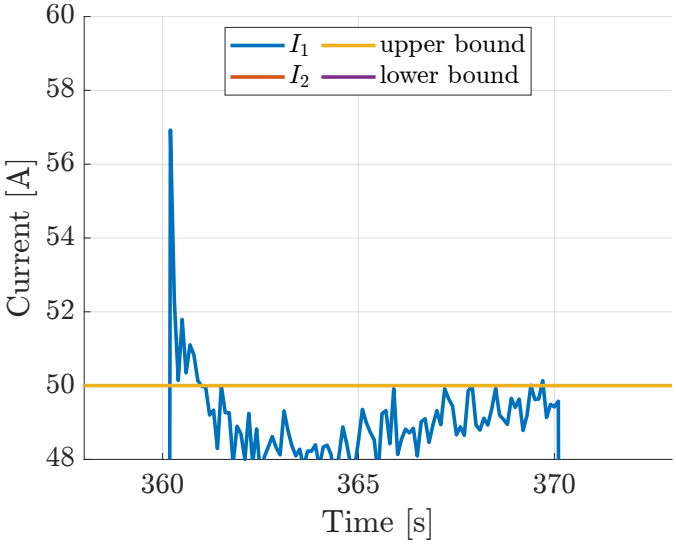


Fig. 55: Detail of one particular overshoot of the OCP limit.

The performance of this algorithm is shown in Fig. 52 and 53. It is clear that the algorithm asymptotically delivers the maximal possible current whilst achieving overshoots of both low height as well as area, as shown in Fig. 54 and 55. Typical overshoot is less than 8 A in height and lasts for less than 400 ms.

Fig. 56 illustrates that a step of the requested system current from 0 to 120 A results in a decrease of the current limit from 94 A to 86 A. This was the worst case (greatest in magnitude) sudden change observed in the simulation outputs.

IV. Conclusions

The achieved performance could be further improved by giving the algorithm access to the requested current. That way, the algorithm could identify time instants when the flow of current is saturated and only then activate the integrator to slowly increase the system-wide current limit. Without this information the algorithm has to slowly increase the current limit at all times in order to be able to quickly deliver the maximal possible current.

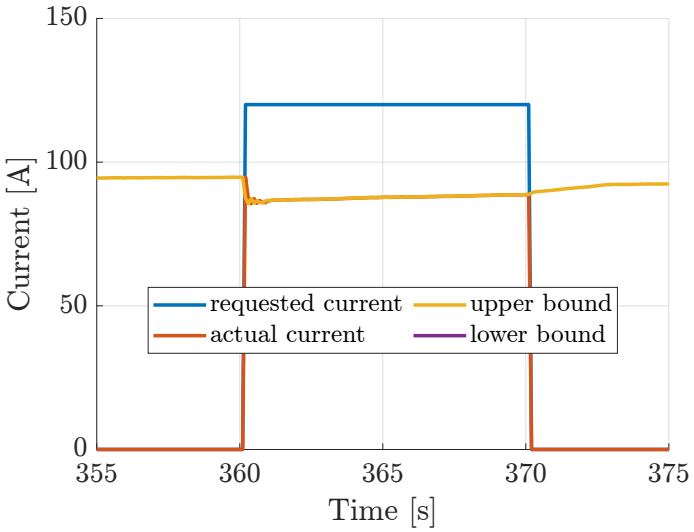


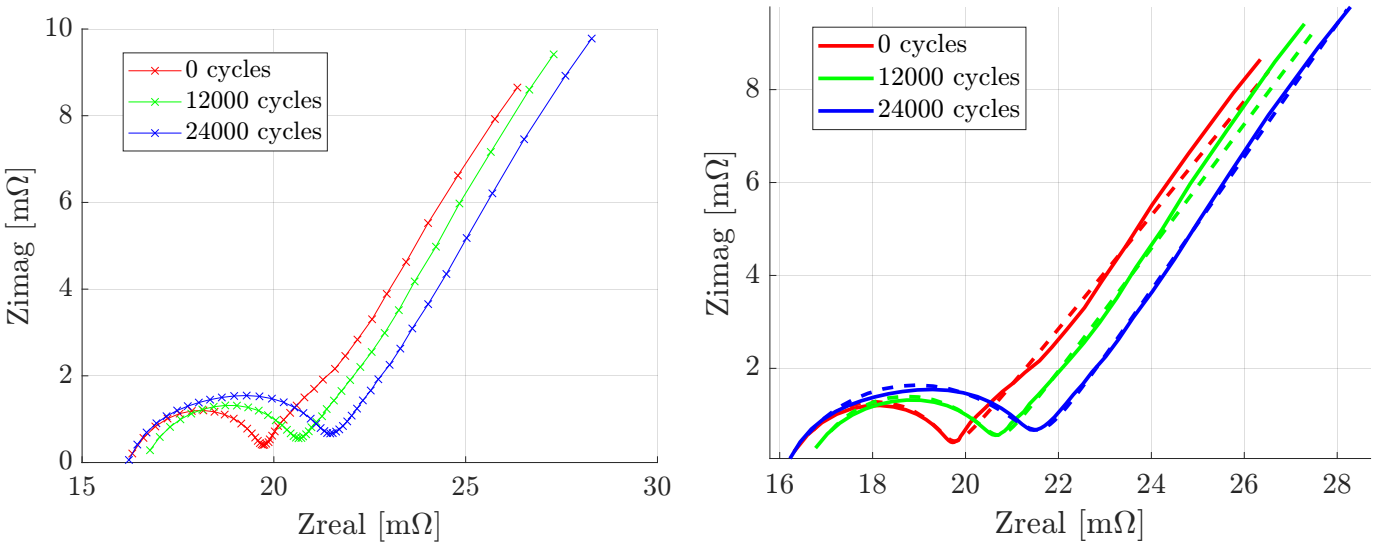
Fig. 56: Response of the system-wide current limit to a step of requested current.

Week 14 – Offline SoH Estimation

Abstract

This report presents an implementation and results of two offline state of health evaluation methods. The assignment provided measured data from a single cell after 0, 12000 and 24000 cycles, namely the impedance data from Electrochemical Impedance Spectroscopy (EIS) and the voltage-capacity dependence for calculation of Differential Voltage (DV) and Incremental capacity (IC) curves. Visualized samples from individual experiments were compared both qualitatively and quantitatively.

I. EIS



((a)) Measured samples.

((b)) Measured and fitted curves.

Fig. 57: Evolution of EIS curves as the cell under test ages.

All provided EIS data correspond to SoC of 60 %. The data was preprocessed according to the standard method – first remove samples with $\Im Z > 0$ (as the indictive character is more influenced by parasitic properties of wires and cell tabs rather than the battery itself) and then flip the sign of $\Im Z$ in the remaining samples to get a plot in the first quadrant. The remaining measured samples are shown in Fig. 57(a). There is a noticeable trend that the inflexion point moves right to higher real parts of the total impedance. It should be noted that the available data does not show a noticeable change in R_0 over the cell's lifetime. This indicates that the resistance of conductors stayed mostly unchanged, and rather only the electrochemical part was subject to degradation.

To fit the provided EIS curves, the parameterized impedance

$$Z(\omega) = R_0 + R_{SEI} || \frac{1}{Q_{SEI}(j\omega)^{n_{SEI}}} + \left(R_{ct} + \frac{A_2}{\sqrt{\omega}(1-j)} \right) || \frac{1}{Q_{ct}(j\omega)^{n_{ct}}}, \tag{55}$$

where $||$ denotes the parallel connection of impedances, of an equivalent circuit proposed in [2] was calculated. It contains a total of eight parameters organized in a series interconnection of blocks

1) real impedance R_0,
2) R_{SEI} in parallel to a constant phase element (CPE) with impedance $(Q_{SEI}(j\omega)^{n_{SEI}})^{-1}$,
3) R_{ct} in series with semi-infinite Warburg $A_2(\sqrt{\omega}(1-j))^{-1}$, the whole branch in parallel to a second CPE with impedance $(Q_{ct}(j\omega)^{n_{ct}})^{-1}$.

Number of cycles	R_0 [mΩ]	R_{SEI} [mΩ]	R_{ct} [mΩ]	R_W [mΩ]	CL [%]	LLI [%]	LAM [%]
0	16.36	3.28	3.91	5.57	0.00	0.00	0.00
12000	16.71	3.93	6.19	5.98	2.13	40.82	7.38
24000	16.21	5.32	4.80	8.08	-0.94	40.87	45.15

TABLE VII: Equivalent circuit parameters fitted from EIS measurements.

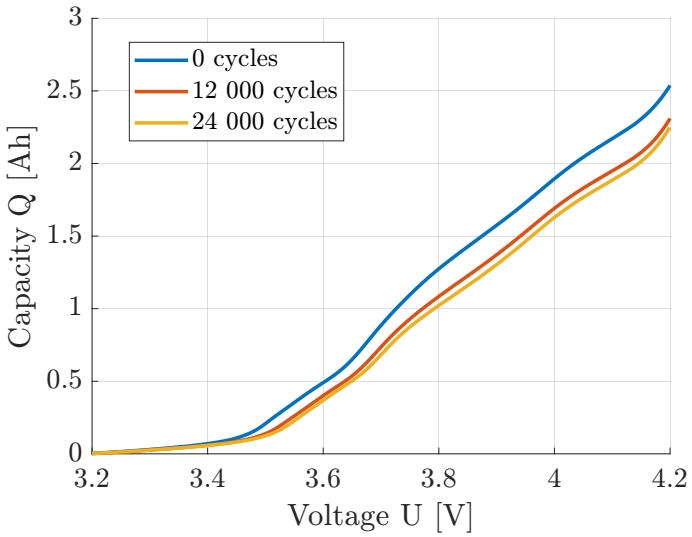


Fig. 58: Collected voltage-capacity data from three charging half-cycles.

Fitted curves are shown in Fig. 57(b) and parameters are important parameters are listed in Table VII. The last three columns additionally include the evaluated "growth in percentage" G_{EIC} metric for individual degradation modes, namely conductivity loss (CL), Lithium loss inventory (LLI) and loss of active material (LAM). Analyzing the table more closely, due to the negligible variance of R_0, the conductivity loss is a very unlikely degradation mode. On the other hand the growth of SEI (Solid Electrolyte Interface) resistance R_{SEI} and charge transfer resistance R_{ct} hint at significant loss of Lithium inventory (LLI). Finally, the increasing Warburg resistance R_W hints at gradual loss of active material (LAM) degradation mode.

II. IC/DV Analysis

A similar analysis can be performed using the time-domain data from a charging half-cycle – ideally with low charging current to obtain pseudo OCV curves, but the assignment only provided CC-CV charging with 1.5 A of current. The measured relation between cell terminal voltage and the current capacity is shown in Fig. 58 for three distinct charging half-cycles throughout the cell's lifetime. The first obvious observation is that as the cell ages, the total capacity decreases, hence the CC-CV charging ends earlier at lower capacity.

To facilitate the calculation of Incremental capacity and Differential Voltage curves, the data was preprocessed to suppress measurement noise using a Savitzky-Golay Filter of order 3 and frame length 501 samples. Subsequently duplicated neighbouring measurements resulting in division by zero were eliminated. Both curves for all experiments are shown in Fig. 59 and

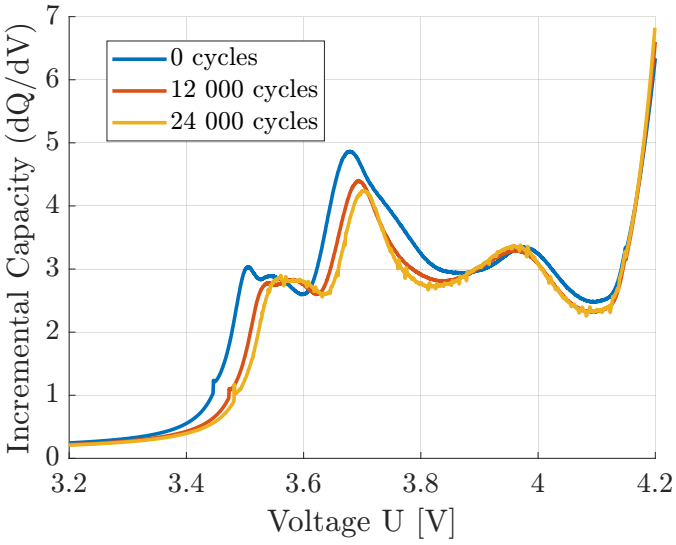


Fig. 59: Evolution of Incremental capacity (IC) during cell's lifetime.

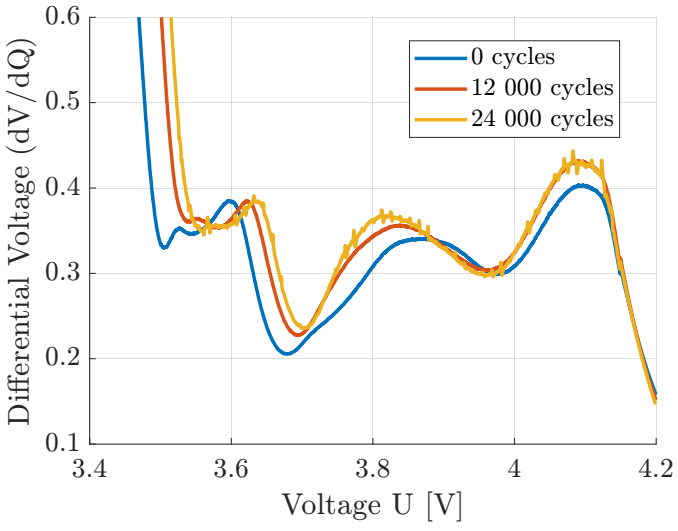


Fig. 60: Evolution of Differential voltage (DV) during cell's lifetime.

Number of cycles	CL [%]	LLI [%]	LAM [%]
0	0.00e+00	0.00	0.00
12000	2.72e-05	8.98	9.64
24000	1.53e-02	11.42	12.89

TABLE VIII: Degradation modes identified using IC/DV curves.

60, respectively. Again, there are some clear differences between the reference curves corresponding to 0 cycles and curves corresponding to points later in the cell's life – for example the DV curve in Fig. 60 gets overall higher whilst the IC curve in 59 overall moves down since as the total capacity decreases with cell's age, lower amount of charge is needed to increase the terminal voltage.

Using formulae from [2], degradation mode indicators for CL, LLI and LAM listed in Table VIII were calculated. Obtained results roughly match the conclusion from EIS (see VII) – the cell aged in a way that did not influence the resistance of conductors, whereas both LLI and LAM are noticeable.

Conclusions

This report comprises of 12 individual papers, each presenting a particular algorithm of battery testing, modelling or state and parameter estimation. The semester started with familiarization with batteries, the measurement hardware involved, and the MATLAB environment used for data postprocessing; these topics are covered in sections 2 through 4. Sections 5 and 6 then covered the basics the dynamical systems, their implementation and identification. These sections introduced equivalent circuit models of various complexities (from the combination of voltage source and internal resistance only to 2 RC ECM), that would be subsequently relied on in sections 7 through 9 introducing the (Extended) Kalman Filter and other method of estimation of unmeasurable quantities. Section 10 expanded the scope of estimation from states to parameters as well with the introduction of recursive least squares. The semester concluded with modern machine learning methods of supervised learning described in section 11, integration of multi-module systems in section 12 and long-term state of health monitoring in section 13.

With the exception of the machine learning task in section 11, which was implemented in Python, all calculations were performed using Matlab or Simulink. This results in a sizeable repository of implemented battery models, model identification routines or state estimation algorithms that can be used to tackle future real-world problems involving batteries.

References

[1] Hongwen He, et al. *"Online estimation of model parameters and state-of-charge of LiFePO4 batteries in electric vehicles"*, Applied Energy, Volume 89, Issue 1, 2012, https://doi.org/10.1016/j.apenergy.2011.08.005.

[2] Carlos Pastor-Fernández et al., *"A Comparison between Electrochemical Impedance Spectroscopy and Incremental Capacity-Differential Voltage as Li-ion Diagnostic Techniques to Identify and Quantify the Effects of Degradation Modes within Battery Management Systems"*, Journal of Power Sources, 2017, https://doi.org/10.1016/j.jpowsour.2017.03.042, Available online at https://www.sciencedirect.com/science/article/pii/S0378775317303269