Contents lists available at ScienceDirect

Journal of Energy Storage

journal homepage: www.elsevier.com/locate/est

Research papers

Remaining discharge energy estimation for lithium-ion batteries using pattern recognition and power prediction

Ollie Hatherall [a,*], Mona Faraji Niri [a], Anup Barai [a], Yi Li [b], James Marco [a]

[a] Warwick Manufacturing Group, University of Warwick, Coventry, CV4 7AL, United Kingdom
[b] Vertical Aerospace Group Ltd., Unit 1, Camwal Court, Chapel Street, Bristol, BS2 0UW, United Kingdom

ARTICLE INFO

Keywords:
Electric vehicle
Lithium ion
Remaining energy
Pattern recognition

ABSTRACT

The remaining discharge energy (RDE) of a battery is an important value for estimating the remaining range of a vehicle. Prediction based methods for calculating RDE have been proven to be suitable for improving energy estimation accuracy. This paper aims to further improve the estimation accuracy by incorporating novel load prediction techniques with pattern recognition into the RDE calculation. For the pattern recognition, driving segment data was categorised into different usage patterns, then a rule-based logic was designed to recognise these, based on features from each pattern. For the power prediction, a clustering and Markov modelling approach was used to group and define power levels from the data as states and find the probabilities of each state-to-state transition occurring. This data was defined for each pattern, so that the logic could inform what data should be used to predict the future power profile. From the predicted power profile, the RDE was calculated from the product of the predicted load and the predicted voltage, which was obtained from a first-order battery model. The proposed algorithm was tested in simulation and real-time using battery cycler data, and compared against other prediction-based methods. The proposed method was shown to have desirable accuracy and robustness to modelling errors. The primary conclusion from this research was using pattern recognition can improve the accuracy of RDE estimation.

1. Introduction

As electric vehicles (EVs) are growing in popularity, the accuracy of the estimation of the remaining distance that can be travelled during a trip is still a high priority for drivers to reduce range anxiety, which is the fear of fully discharging the batteries in the middle of a journey [1]. To predict the remaining range of a vehicle, remaining discharge energy (RDE) is used [2]. RDE refers to the amount of energy that can be discharged before a battery reaches its lower cut-off voltage [2], which is usually set by the manufacturer to ensure safe and reliable operation. RDE is different from state-of-charge (SoC), which is a very common research topic [3], as SoC is a measure of the remaining capacity and not remaining energy [2].

RDE can be calculated using

$$RDE = \int_{t}^{EoDT} U_t(\tau) \cdot I(\tau) \cdot d\tau, \tag{1}$$

where t is the current time, $EoDT$ is the end of discharge time, U_t is the battery terminal voltage as a function of time τ, and I is the current as a function of time τ [2,4]. Since this equation relies on the a priori values for voltage and current, it cannot be used onboard and hence, methods for RDE estimation have been proposed [4]. These can be grouped into several categories: the direct calculation method, model-based estimation methods using filters, and prediction-based methods.

The direct calculation method, given by Eq. (2), uses the current SoC of the battery, to calculate RDE.

$$RDE = Q \cdot U_{nom} \cdot (SOC_t - SOC_{EoDT}), \tag{2}$$

where Q is the battery's rated energy capacity, U_{nom} is the terminal nominal voltage, SOC_t is the state-of-charge value at the current time, and SOC_{EoDT} is state-of-charge value at the end-of-discharge time [2, 5]. Although this method is easy to implement and has low computational costs, which is advantageous for real-time implementation, the nominal voltage used cannot represent the complex, non-linear voltage responses of the system, thus will cause large errors [4]. Additionally, capacity is a function of discharge rate and temperature, so using a simplified measure of rated capacity can produce further inaccuracies [6].

Model-based methods often use the metric state-of-energy (SoE) to define a battery's remaining energy as a ratio of its total available

* Corresponding author.
E-mail addresses: ollie.hatherall@warwick.ac.uk (O. Hatherall), mona.faraji-niri@warwick.ac.uk (M.F. Niri), A.Barai@warwick.ac.uk (A. Barai), yi.li@vertical-aerospace.com (Y. Li), James.Marco@warwick.ac.uk (J. Marco).

https://doi.org/10.1016/j.est.2023.107091
Received 9 November 2022; Received in revised form 1 February 2023; Accepted 6 March 2023
Available online 22 March 2023

energy. These methods use algorithms to estimate SoE, in a similar manner to SoC estimation [7]. SoE is calculated using

$$SoE_t = SoE_{t_0} - \int_{t_0}^{t} P(\tau)d\tau / E_N, \tag{3}$$

where SoE_t is the SoE value at the current time t, SoE_{t_0} is the SoE value at the initial time t_0, P is the power output as a function of time τ, and E_N is the total available energy [4]. This calculation relies on accurate estimations of the energy removed from the battery, which can be affected by measurement errors, and an accurate estimation of the maximum available energy, which changes based on future conditions and the health of the battery [5,8]. Additionally, model-based estimations are often based on equivalent circuit models with OCV vs SoE curves [9], which cannot account for dynamic condition operating conditions [4] and variations due to battery degradation [10].

Prediction-based methods obtain an RDE estimate by aiming to predict future battery states. Since RDE can be calculated using Eq. (1), predicting the future current and voltage can produce very accurate RDE results. In [2], a method that uses real-time battery model parameter estimation and future battery model parameter prediction for RDE estimation was proposed. This method showed a good agreement between the simulated voltage response of the battery and the measured voltage, and also showed the proposed method had a much greater level of accuracy than the direct calculation method, for dynamic stress test (DST) and the Federal Urban Driving Schedule (FUDS) profiles. In [4], future temperature as well as SoC dependent model parameters are considered to further improve the voltage prediction accuracy.

In these examples, the integration of load prediction techniques were not investigated, and the future load used was a priori known, but for transportation applications this would not be the case. From Eq. (1), it can be seen that RDE is directly related to the future load applied to the battery [5]. Additionally, the future load directly affects the results for the future voltage response from the battery model. Due to these factors, investigating and improving load prediction methods can be valuable for increasing RDE accuracy and therefore, understanding the remaining energy that can be drawn from the battery for a given application.

Methods for load prediction are discussed in [5,11]. The fundamental framework involves using a moving window to collect load data, which can be represented as a sequence of distinct states and used to forecast future load profiles. This approach requires two algorithms, a method for grouping the data set and a method to predict a future load profile, in particular, clustering and Markov modelling have been used previously [5]. Clustering algorithms are a type of unsupervised learning method for grouping unlabelled data [12–14]. A cluster is defined as a subset of the total data set which contains data with similar characteristics, where the cluster centre can be used to represent the values of the data in the cluster. For this application, these are used to give an indication of the common load values in a set of load profiles. A clustered load profile can be used to find the probability of transitioning from one load value to another, which is the basis of Markov modelling. For RDE estimation, the future load profile is predicted using the cluster values and the Markov models until the end-of-discharge, so no static prediction horizon is needed. The future voltage can then be predicted by using the predicted load as an input to a battery model. Thus, the RDE can be calculated by multiplying the predicted voltage and predicted load vectors together, then summing the result, as shown in the equation

$$RDE = \sum_{t_0}^{t_{end}} U_{t,pred} \cdot I_{pred}, \tag{4}$$

where t_0 is the time at the start of the prediction, t_{end} is the time when the end-of-discharge is predicted, and $U_{t,pred}$ and I_{pred} are the vectors for the terminal voltage and current, respectively. The frequency of the RDE calculation is a parameter that can be changed based on the accuracy and computational efficiency requirements.

In previous work [15], an offline-training method to estimate RDE was proposed. This method differs from the previous RDE estimation methods as all the data processing for load prediction was performed offline instead of using an online moving window. It was shown that if the training data was similar to the test profile, the RDE results can be more accurate than using the moving window method, but if the data was vastly different to the test profile then the moving window approach was better [15]. This means that the RDE estimation is most accurate when the predicted load profile was similar to the load profile for the current operation. A power level recognition algorithm was used to determine whether to use the offline data or the moving window method. This algorithm could be seen as a limited version of driving pattern recognition (DPR).

Many examples of DPR can be found in the literature. In [16], fuzzy logic pattern recognition was outlined and used to assess EV performance. In [17], features derived from velocity data were chosen based on dependencies between other features and having a high correlation with fuel consumption. The chosen features were clustered to define different types of driving profiles. More recently, DPR has been used as the basis for the design of energy management systems [18–20], where in [18], Markov chains were used to predict future velocity (in a similar manner to how they are used for RDE estimation), with a neural network used to recognise the current driving pattern. Neural networks were also used in [19,20]. In these examples of DPR, the choice of features and the number of driving scenarios used were not explored, referencing other literature for their choices. Therefore, an examination of how the choice of features and scenarios affect the DPR accuracy was performed here for completeness.

This paper aims to extend and improve the previous work by combining prediction-based methods for RDE estimation with DPR. The performance of the proposed algorithm was assessed here and compared to previous methods. In addition, a methodology for selecting the number of features and scenarios for an online DPR algorithm was outlined, as well as a rule-based logic for online pattern recognition. The algorithms for RDE estimation were validated for both simulation and on real-time hardware. The control architecture used is shown in Fig. 1. This diagram shows the key components of the control system, including which parts are performed offline and online.

Table 1 shows a comparison of the RDE estimation methods discussed here as well as the proposed method, showing the advantages and disadvantages of each method.

The contributions of this paper are as follows. (1) A novel RDE estimation method which incorporates DPR into the design of a load and power prediction framework, which to the best of the authors' knowledge has not been implemented before. (2) Performance analysis and comparison of existing and the proposed prediction-based RDE methods in simulation and real-time operation, to demonstrate the superior accuracy of the proposed method. (3) A method for determining the optimal number of clusters for a driving cycle data set, which is compared to another commonly used metric to show the improved repeatability of the results and accuracy for densely packed data points. (4) A new methodology to optimise the pattern recognition accuracy based on a feature selection approach, using a large set of real-world EV driving profiles. (5) Hardware-in-the-loop verification of the proposed algorithm.

The structure of this paper is as follows: in Section 2, the methodology of the DPR and load prediction methods used are proposed. In Section 3, the methodology of the cell characterisation and modelling is outlined. Section 4 shows optimisation of the DPR and load prediction algorithms, and the RDE estimation results from simulations and real-time implementation. In Section 5, conclusions about the findings are made, with possible further directions proposed.

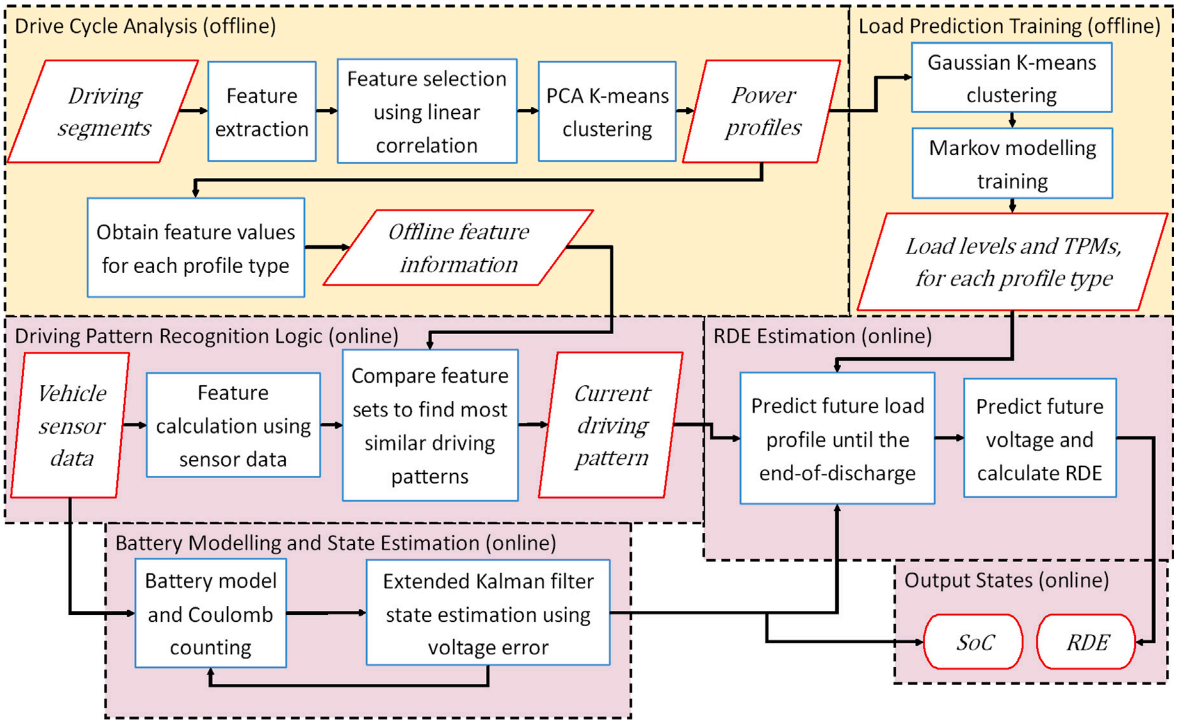

Drive Cycle Analysis (offline)
Driving segments
Feature extraction
Feature selection using linear correlation
PCA K-means clustering
Power profiles
Obtain feature values for each profile type
Offline feature information
Load Prediction Training (offline)
Gaussian K-means clustering
Markov modelling training
Load levels and TPMs, for each profile type
Driving Pattern Recognition Logic (online)
Vehicle sensor data
Feature calculation using sensor data
Compare feature sets to find most similar driving patterns
Current driving pattern
RDE Estimation (online)
Predict future load profile until the end-of-discharge
Predict future voltage and calculate RDE
Battery Modelling and State Estimation (online)
Battery model and Coulomb counting
Extended Kalman filter state estimation using voltage error
Output States (online)
SoC
RDE

Fig. 1. Flowchart of the control architecture proposed here, showing which components are performed online and offline. The blue rectangle blocks represent processes/algorithms, the red parallelogram blocks represent input and output between processes, and the red oval/pill blocks represent outputs of the RDE algorithm. The yellow bounded areas represent offline sections of the algorithm and the purple bounded areas represent online sections of the algorithm. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 1
Table summarising the advantages and disadvantages of the RDE estimation methods discussed [4].

Estimation method	Advantages	Disadvantages
SoC direct calculation	Low computational cost, easily implemented	Cannot represent complex voltage responses, uses a constant rated capacity value
Model-based methods	Accurate, robust to measurement errors	Does not consider future operating conditions, estimates total available energy percentage not RDE
Prediction-based methods	Accurate, considers future battery states	Requires accurate load prediction, relies on accurate SoC and capacity values
Proposed method	Accurate, produces current driving scenario information, performs data processing offline	Increased computational intensity, reduced accuracy for rapidly changing driving conditions

2. Methodology

2.1. *Driving pattern recognition*

DPR uses vehicle measurement data to identify the current driving scenario and style [18] and has been used successfully to underpin the design of the energy management algorithms in many applications [19, 20]. DPR contains two key parts, drive cycle analysis and the pattern recognition logic. Drive cycle analysis is used to process and group the data set into different patterns based on available measurement data, such as velocity and power. The recognition logic aims to determine the current driving pattern and is defined based on the different patterns found previously, with the goal of real-time implementation.

The EV data used throughout this paper was obtained by recording the usage of commercially available passenger EVs. The vehicles were driven over a period of 12 months to capture their operation during both hot and cold climates. In addition, the vehicles were driven in a combination of urban and highway driving. Data was recorded directly from the vehicle's controller area network or CAN bus. In total around 4600 vehicle trips were recorded, representing around 18,000 miles of operation.

2.1.1. *Drive cycle analysis*

To define different driving cycles, a feature selection and clustering framework was required. The EV driving data trips were split into smaller sections called microtrips by considering when the driving segments were at zero velocity for an extended period of time (5 s). The purpose of this was to remove large sections of when the vehicle was stationary. Here, a minimum of 5 s was used to define a new microtrip, but this can be seen as an arbitrary choice. Features based on each microtrip's velocity and power measurements were defined, which provided a summary of each microtrip's characteristics. This process was used in [21]. The 24 features used are shown in Table 2, where acceleration is the rate of change of speed, and jerk is the rate of change of acceleration. These were chosen as they can be easily identified from vehicle measurements.

Table 2
Table showing feature definitions and symbols.

Feature name	Equation	Symbol
Mean velocity	$\frac{\Sigma_{i=1}^{n} v_i}{n}$	$\bar{v}$
Maximum velocity	$max(v_i)$	v_{max}
Minimum velocity	$min(v_i)$	v_{min}
Variance of velocity	$\frac{\Sigma_{i=1}^{n}(v_i-\bar{v})^2}{n-1}$	v_{var}
Mean acceleration	$\frac{\Sigma_{i=1}^{n} a_i}{n}$	$\bar{a}$
Maximum acceleration	$max(a_i)$	a_{max}
Minimum acceleration	$min(a_i)$	a_{min}
Variance of acceleration	$\frac{\Sigma_{i=1}^{n}(a_i-\bar{a})^2}{n-1}$	a_{var}
Mean positive acceleration	$\frac{\Sigma_{i=1}^{n} a_i}{n}$, for $a_i > 0$	$\bar{a}_+$
Mean negative acceleration	$\frac{\Sigma_{i=1}^{n} a_i}{n}$, for $a_i < 0$	$\bar{a}_-$
Mean jerk	$\frac{\Sigma_{i=1}^{n} j_i}{n}$	$\bar{j}$
Maximum jerk	$max(j_i)$	j_{max}
Minimum jerk	$min(j_i)$	j_{min}
Variance of jerk	$\frac{\Sigma_{i=1}^{n}(j_i-\bar{j})^2}{n-1}$	j_{var}
Mean positive jerk	$\frac{\Sigma_{i=1}^{n} j_i}{n}$, for $j_i > 0$	$\bar{j}_+$
Mean negative jerk	$\frac{\Sigma_{i=1}^{n} j_i}{n}$, for $j_i < 0$	$\bar{j}_-$
Mean power	$\frac{\Sigma_{i=1}^{n} P_i}{n}$	$\bar{P}$
Maximum power	$max(P_i)$	P_{max}
Minimum power	$min(P_i)$	P_{min}
Mean temperature	$\frac{\Sigma_{i=1}^{n} T_i}{n}$	$\bar{T}$
Maximum temperature	$max(T_i)$	T_{max}
Minimum temperature	$min(T_i)$	T_{min}
Mean temperature change	$\frac{\Sigma_{i=2}^{n} T_i - T_{i-1}}{n}$	$\bar{\Delta T}$
Maximum temperature change	$max(T_i - T_{i-1})$	ΔT_{max}
Minimum temperature change	$min(T_i - T_{i-1})$	ΔT_{min}

After the features were obtained, ones with a high correlation with each other were removed. This was to reduce the total number of features by removing redundant features [22]. The metric used here to find the correlation was the linear correlation coefficient. This method was selected as it provides an efficient way determine if there is a linear relationship between two variables [22]. The linear correlation coefficient of two vectors can be defined as [23],

$$r(X,Y) = cov(X,Y)/(\sigma(X)\sigma(Y)) \tag{5}$$

where r is the correlation coefficient of vectors X and Y, $cov(X,Y)$ is the covariance of vectors X and Y, and $\sigma(X)$ is the variance of vector X, where X and Y are the values of features for all microtrips. The value of r will be between −1 and 1 [22], where a value of 1 indicates that the vectors are fully correlated.

A threshold value needed to be chosen to determine if two features are dependent. In [17], a threshold value of 0.8 was used. When examining the features, different threshold values were used to see how they would affect the features removed. When 0.8 was used, 5 features were removed, these being v_{max}, a_{max}, a_{var}, $\bar{T}$, and T_{max}. When the value was raised to 0.9, only v_{max}, $\bar{T}$, and T_{max} were removed, but when the value was 0.75, 8 features were removed. These were $\bar{v}$, v_{max}, a_{max}, a_{var}, $\bar{a}_+$, j_{var}, $\bar{T}$, and T_{max}. The feature for mean velocity has a strong relation with the power consumption of the vehicle, so it would be ideal if this feature was kept. Therefore, a value of 0.8 was deemed to be a suitable value for use here.

To identify which features had high correlation with $\bar{P}$, the linear correlation coefficient was used in conjunction with reviewing scatter plots of the each feature against $\bar{P}$, where this process was performed in [17]. These scatter plots are shown in Fig. 2 and the values from the linear correlation coefficients are shown in Fig. 3. Features such as $\bar{a}_+$ and $\bar{\Delta T}$ have a small correlation coefficient, so they should not be included when clustering. Therefore, another threshold value was defined to determine what features will be selected for clustering. A clear example of correlation is shown in Fig. 2(a), where it shows that overall, when $\bar{v}$ increases, $\bar{P}$ increases. Figs. 2 (q) and (r) do not show a correlation, by inspection, and therefore, the threshold value should be greater than their correlational coefficient, 0.35 as shown in Fig. 3. So, an initial threshold value of 0.4 was used to for clustering and the DPR logic. The resultant features are $\bar{v}$, v_{var}, a_{min}, $\bar{P}$, and P_{min}.

Table 3
Table showing the number of principle components and their respective percentage contribution to the total variance. The cumulative percentage is also calculated.

Principle component	Percentage of variance (%)	Cumulative percentage (%)
1	73.2	73.2
2	10.7	83.9
3	8.0	91.9
4	5.2	97.1
5	2.9	100

To group the microtrips into different driving patterns, clustering needed to be applied to the data in the feature set. Since the total feature set may be very large, it was important to implement clustering algorithms that deal well with high dimensional data to avoid ‘the curse of dimensionality’, which causes data points to become sparser as the number of dimensions is increased [24]. One method used to avoid this problem is using principal component analysis (PCA) to reduce the dimensionality of the data set before clustering. Examples of this technique can be found in [18,25]. Here, the data was first processed using Z-score normalisation, as described in [25,26]. This reduces the weighting that features with large values will have on the clustering. After performing PCA, the number of principle components (PCs) used to represent the dimensions for clustering needed to be determined by choosing the ones which contribute the most variance, which would result in a minimum loss of information [27].

For this example, five features were used, so five PCs were found. Their variance percentages and the cumulative percentages are shown in Table 3. In [18], the number of PCs used was decided by the component that had more than 80% cumulative percentage.

The clustering algorithm used here was K-means clustering, where each principle component represented the different dimension. K-means clustering is a popular choice of clustering algorithm [28], that uses distance calculations to partition a set of points. It is easily implemented and has a high computational speed, which allows for large data sets to be processed [29,30], so it was suitable for this application. A major issue with K-means clustering is choosing the optimal number of clusters. Similar processes for grouping driving segments can be found in [17,18,31,32]. In [18,31], no rationale was given to deciding the number of clusters used. In [32], the choice was made by inspecting the clustering results. In [17], the silhouette coefficient was used to determine the optimal number of clusters, which is defined by

$$S(p) = \frac{b(p) - a(p)}{max\{a(p), b(p)\}}, \tag{6}$$

where $S(p)$ is the silhouette coefficient for a point p, $a(p)$ is the average distance from point p to all other points within the same cluster, $b(p)$ is the average distance from point p to all points within the next nearest cluster [17,33]. The silhouette coefficient is a common method for deciding the optimal number of clusters [33,34]. It determines how well separated the clusters are and gives a value between −1 and 1. The higher the silhouette coefficient is, the better defined the clusters are [35]. If a data point has a negative silhouette coefficient, this means it is likely assigned to the wrong cluster. When a data set contains many densely packed points, the silhouette coefficient cannot reliably

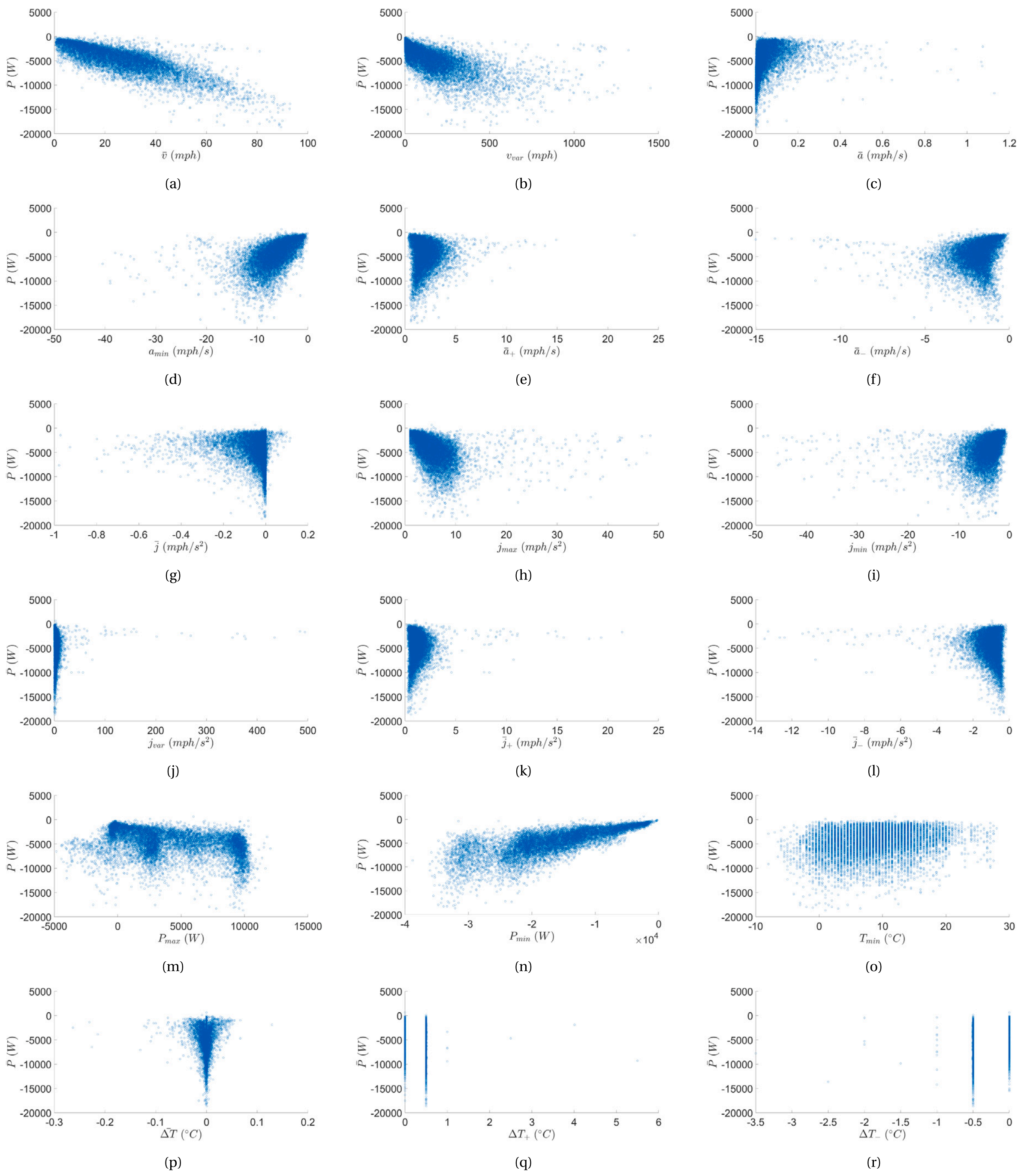


Fig. 2. Scatter plots of the feature set against the mean power. (a) mean velocity ($\bar{v}$), (b) variance of velocity (v_{var}), (c) mean acceleration ($\bar{a}$), (d) minimum acceleration (a_{min}), (e) mean positive acceleration ($\bar{a}_+$), (f) mean negative acceleration ($\bar{a}_-$), (g) mean jerk ($\bar{j}$), (h) maximum jerk (j_{max}), (i) minimum jerk (j_{min}), (j) variance of jerk (j_{var}), (k) mean positive jerk ($\bar{j}_+$), (l) mean negative jerk ($\bar{j}_-$), (m) maximum power (P_{max}), (n) minimum power (P_{min}), (o) minimum temperature (T_{min}), (p) mean temperature change ($\Delta\bar{T}$), (q) maximum temperature change (ΔT_+), (r) minimum temperature change (ΔT_-).

determine the optimal number of clusters. This will be shown in the results section.

Therefore, a new approach was proposed here, which calculates the average standard deviation in the size of the clusters from multiple restarts. The aim of this was to reveal the natural groupings in the data by evaluating how the initial conditions for the cluster centres affect the final number of points in each cluster. This was achieved by clustering the data set with a predetermined number of clusters and ordering the clusters based on their mean power, then calculating the percentage of the total amount of data points in each. This process can be repeated to give a large set of percentage values, so that the difference in the sizes of each cluster can be assessed by calculating the standard deviation of the cluster sizes. By comparing the results when different numbers of clusters are used, the optimal number of clusters can be determined based on the highest cluster number before a significant change in standard deviation occurs. This is because a large change in the standard deviation would suggest that there is a dependence on the initial locations for the clustering outcome.

2.1.2. *Pattern recognition logic*

To use the defined profile types, a method was implemented to recognise the driving pattern based on the current driving conditions.

```
Data: $sensorData$, $offlineFeature(i, j)$
$i$ ← Number of features;
$j$ ← Number of driving patterns;
for $i$ do
    $sensorFeature(i)$ ← calculate feature $i$ from $sensorData$;
    $featureIndex(i)$ ← index $j$ of $min(abs(sensoreFeature(i) - offlineFeature(i, j)))$;
end
Current driving pattern ← $mode(featureIndex)$
```

Algorithm 1: DPR logic pseudocode.

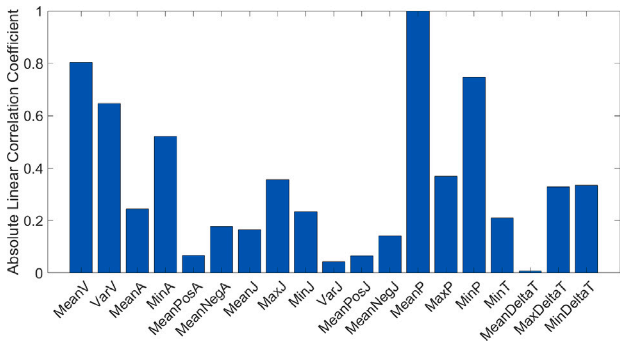


Fig. 3. A bar chart showing the linear correlation coefficient between certain features and mean power.

By using the pre-PCA feature set values of the microtrip segments, the feature values for the different profile types were obtained. A simple rule-based logic was devised by calculating the same features from sensor data and comparing it to the offline features. The input, processing and output of the logic are shown in the Driving Pattern Recognition Logic block in Fig. 1, and the pseudocode for the logic is shown in Algorithm 1. For this method, a moving window that gathers measurement data was implemented. The length initially used was 500 s, but this value was optimised later.

2.2. Power prediction

This section outlines how power prediction is used for RDE estimation, using the driving patterns defined previously. The method used here follows similar methods outlined in [5]. There are two key components to the proposed power prediction method, a clustering algorithm to obtain distinct power levels and the use of Markov modelling to predict future power levels. The difference for the method proposed here is that the relevant information is obtained offline, from a training data set of EV data, for prediction online. This method was used to investigate whether there was a greater potential of increasing the RDE estimation accuracy when used with DPR.

Performance analysis between different prediction-based methods was also performed here. Therefore, the same power prediction method was implemented for all methods. The other methods implemented were an offline approach was used without DPR, where the Markov model was trained from the whole microtrip data set, and an online moving window method, which was used in [5]. The moving window method gathers data during a cycle and performs clustering and the Markov model training online. For the purpose of comparison, these methods used the same clustering and Markov modelling approach. Hence, an efficient clustering and Markov modelling method needed to be implemented, so that is could be performed online in real-time. The K-means algorithm for clustering meets such requirements.

2.2.1. K-means clustering with Gaussian distribution

This application of clustering has a different purpose to the previous implementation. Here, the goal was to find power levels within the data, which were used to predict the future power profile. For the implementation of clustering algorithms for power prediction, Gaussian mixture modelling (GMM) was used in [5] to represent load levels as a Gaussian distribution [36], whereas K-means clustering was used in [11], where the load value is represented as the value of the cluster centre. Since the power draw for EVs can be very transient, using the cluster centres from K-means clustering algorithm results in a loss of accuracy, instead of a distribution of possible values. In contrast, an unaltered GMM will result in computational times many orders of magnitude greater than K-mean clustering [37,38], which will be problematic for clustering large data sets.

Therefore, the method used here involved using K-means to find the cluster centres of the data set and then calculating the variance of the points in each cluster, so that the cluster could be represented as a Gaussian distribution. The information obtained was used online to calculate power values. This was done by randomly generating a number between 0 and 1 and using this as the probability value for a Gaussian inverse calculation, along with the information obtained from clustering.

2.2.2. Markov modelling

Through clustering, power profiles can be described as a sequence of distinct states, which represent certain power levels. This information has been used to form a Markov model [39], with the purpose of forecasting future states [40]. Markov models use probabilities to predict the next state, depending on the current state. The probability information is stored in a transitional probability matrix (TPM). This information is used to predict all future states, where only the current state is needed to predict all future states [5,41].

The TPM is an n-by-n matrix, where n is the number of clusters. The rows represent the current state, and the columns represent the possible future states, so the probability of transitioning from state i to state j is given by M_{ij}. Note that the rows sum to 1. The TPM is calculated by counting the number of times each transition between specific states for i and j occurs, then dividing it by the total number of times i occurs.

For the method proposed here, the TPM was calculated offline from the microtrip data, where cluster information and TPMs were defined for the different driving scenarios. The DPR logic was used to determine which set of data was be used to predict the future profile, under the assumption that the current scenario will continue until the end-of-discharge.

3. Cell characterisation and modelling

To implement the RDE estimation algorithm as well as for real-time implementation, a battery model was needed. This section outlines the experimental approach used to characterise a set of cells, and how parameters for the battery model are identified and validated.

3.1. Experimental characterisation

Five new Samsung 21,700 48X cells were characterised using a MACCOR battery cycler. The tests used here consisted of a capacity test, a discharge capacity test and an open circuit voltage hybrid pulse power characterisation (OCV-HPPC) test, as described in [42,43]. These tests were performed at four different temperatures, 0 °C, 15 °C, 25 °C and 40 °C, with 4 h used for temperature stabilisation. For these cell, the 1C discharge rate was 4.8 A. Additionally, the charge and discharge cut-off voltage for the cells followed the data sheet limits of 4.2 V and 2.5 V, respectively.

The capacity test used here consists of three C/3 constant current constant voltage (CCCV) charge and discharge cycles, with a cut-off current in the CV phase of C/20, to test the stability of the cell's capacity [42]. The discharge capacity tests measured the energy capacity at different discharge rates, these being C/3, 1C, 2C, 3C, 5C and 8C. The OCV-HPPC test that was used combined discharge pulses with long rest periods. The OCV was measured from the voltage values before a 10 s 2C discharge pulse was applied, with a rest period of 4 or 6 h used. An incremental OCV test was used to obtain the OCV values, as it was shown in [44] that this will produce more accurate measurements, with SoC breakpoints of 1% increments between 100% and 91% SoC with 6 h rest periods, 4% increments between 90% and 22% SoC with 4 h rest periods, and 1% increments between 21% and 1% SoC with 6 h rest periods. The longer rest periods were used in the more non-linear regions, with the aim of getting more accurate results.

3.2. Cell modelling

The choice of model used here was a first order equivalent circuit model (ECM) [45]. This was chosen as a computationally efficient model was needed for predicting the future battery voltage and real-time implementation. To identify the model parameters, a least-squares method was used. The voltage response of the model was described by the following discrete form state equations [46],

$$U_{p,k+1} = U_{p,k} e^{\Delta t/\tau} + (1 - e^{\Delta t/\tau}) R_p I_k, \tag{7}$$

$$U_{T,k} = U_{OC,k} + U_{p,k} + R_0 I_k, \tag{8}$$

where k is the current time step, U_p is the polarisation voltage, Δt is the change in time (set to 10 ms), τ is the time constant, which is equal to the polarisation resistance, R_p, multiplied by the polarisation capacitance, C_p; U_T is the terminal voltage, U_{OC} is the OCV, R_0 is the internal resistance, I is the current, where negative current is discharge current.

To identify the relevant model parameters, data from the OCV-HPPC test was used. Firstly, the internal resistance ($R0$) was calculated from the voltage drop when the pulse was applied, using Ohm's law [47,48]. This point was determined when the measured current was first equal to the designed pulse current, this was the second sample point, with a sample time of 0.1 s. To identify the values of the RC pair, the *lsqcurvefit* library was used, which is part of the Mathworks optimisation toolbox [49]. This enabled the values of R_p and C_p to be optimised against the experimental data. Eqs. (7) and (8) were used as the function that needs to be optimised with R_p and C_p as the unknown values. The extracted parameters are shown in Fig. 4.

To validate the model parameters obtained, a predefined discharge cycle was used on a fresh set cells of the same type used for parametrisation to measures the voltage response and determine the accuracy of the battery model. The profile was chosen as it has both regions of higher current pulse and regions of constant current. Fig. 5 shows a validation cycle for one of the cells used. The voltage responses and errors are shown. Overall, three cells were used for validation at four temperatures. It was found that the RMSE of the model was 20 mV for all tests.

4. Results and discussion

4.1. Driving pattern recognition

In order to optimise the DPR, various values outlined in the methodology could be adjusted to maximum the pattern recognition accuracy. This involved clustering the data set to obtain distinct power profiles and determining the which features to used for the rule-based pattern recognition logic.

Table 4
Results of silhouette coefficients for different cluster numbers with multiple runs of the clustering algorithm, where k refers to the number of clusters.

Run	2 k	3 k	4 k	5 k	6 k	7 k
1	0.34	0.35	0.33	0.36	0.39	0.28
2	0.34	0.36	0.37	0.31	0.34	0.38
3	0.34	0.36	0.33	0.36	0.34	0.44
4	0.34	0.35	0.33	0.36	0.34	0.45
5	0.34	0.35	0.33	0.36	0.39	0.34
Mean	0.34	0.35	0.34	0.35	0.36	0.38

4.1.1. Drive cycle analysis

Before the DPR can be optimised, the microtrips needed to be grouped using K-means clustering. For the data set here, the mean silhouette coefficient for all points was calculated from multiple runs and these results are shown in Table 4. The higher cluster numbers showed far more variance in the values of their silhouette coefficients, such as 7 clusters having the largest and smallest coefficient values. Therefore, this method was deemed not to produce a clear optimal number of clusters.

An example of how the initial cluster centre locations affect the final number of points assigned to each cluster are shown in Fig. 6. Here, there were distinct power levels for both 3 and 4 clusters, but the size of the clusters had a high variability with 4 clusters. Hence, the proposed metric proposed in Section 2.1.1 was used.

For the proposed metric, the optimal number of clusters would be the largest number of clusters before a significant change in the standard deviation of the cluster size, as a high standard deviation would show that the size of the clusters is significantly affected by the initial positions. For the current data set, this metric was calculated from 50 restarts for each number of clusters. This number was chosen to give a balance between a large enough same size and the computational time, as a negative of this method was the large amount of time needed. The results from this are shown in Fig. 7, where it is shown that the best number of clusters to use in this scenario is 3.

From clustering, an example of each type of profile are shown in Fig. 8, which were labelled low power, medium power, and high power. The differences between the types of profiles are observable, the low power profile had a has low charge and discharge power; the medium power had a much larger range of power value but was a lot more transient than the other profiles; and the high power profile had regions of sustained power discharge, as well as high charge regions.

4.1.2. Pattern recognition logic

For testing the accuracy of the logic, the data set was randomly split into 70% training data and 30% testing data. The accuracy was defined as the percentage of correct identifications, where the test data was formed into three profiles based on the different clustered driving microtrips.

To validate the feature selection and clustering approach, in terms of DPR accuracy, the logic was tested with and without feature selection before clustering and with different number of PCs used for clustering. An additional metric used here was the percentage of split cases, which was where a clear driving pattern cannot be determined, e.g., if the calculated features correspond to clusters [1, 1, 2, 2, 3]. The results are shown in Table 5. The findings showed that there was no noteworthy change in the accuracy or number of split cases when the number of principle components was changed. The results also showed a significant drop in accuracy when a larger feature set was used. The cause of this was the inclusion "irrelevant" data, which makes recognition more difficult. Furthermore, the difference in accuracy between clustering with 3 and 24 features was significant and could be caused by clustering a high dimensional data set.

From these findings, there was a correlation between the features used and the accuracy of the DPR logic. Therefore, the threshold value

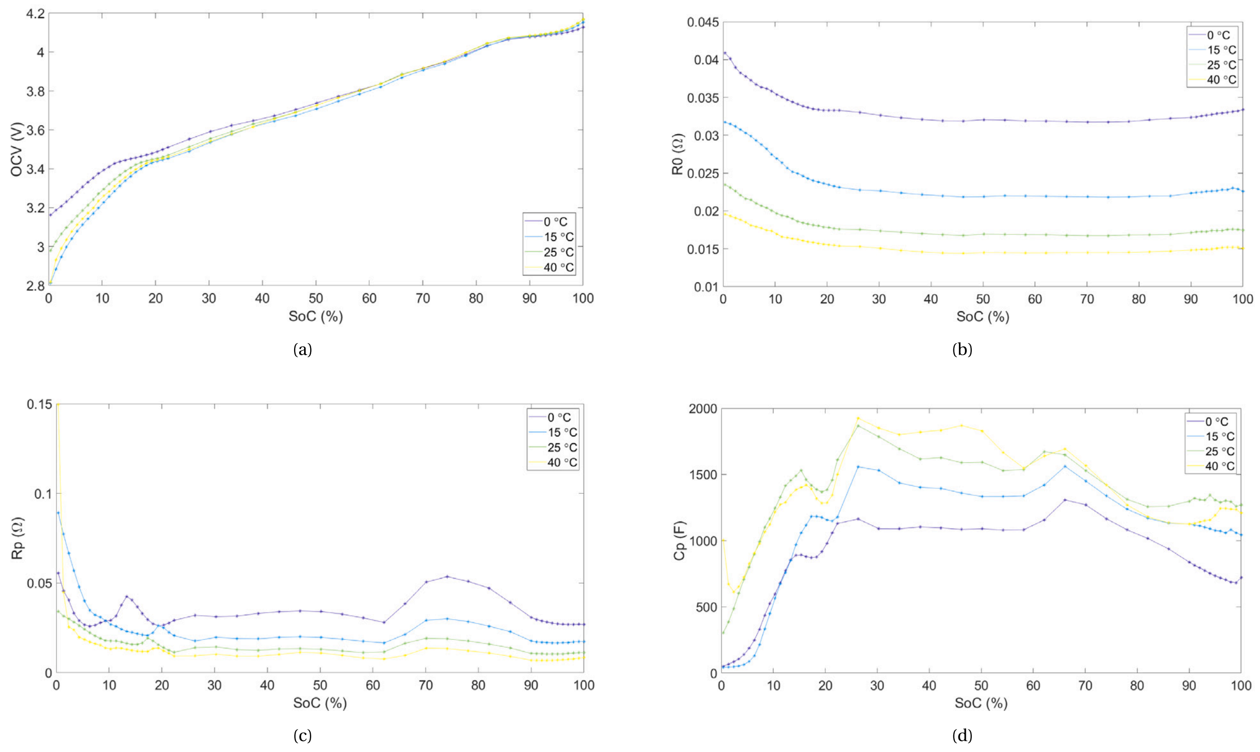


Fig. 4. Parameters identified from the HPPC-OCV test, plotted against SoC at all temperatures used. For modelling, parameters have been reinterpolated so the SoC breakpoints are the same. (a) shows OCV data, (b) shows internal resistance data, (c) shows polarisation resistance data, and (d) shows polarisation capacitance data.

Table 5
Table showing accuracy of the recognition logic and the number of split cases (where the mode has two values), for PCA K-mean clustering with different numbers of PCs, against an approach that does not use any feature selection before the PCA clustering is performed, for 3 and 24 PCs. A moving window length of 300 s was used. The features used here are $\bar{v}$, v_{var}, a_{min}, $\bar{P}$, and P_{min}.

	With feature selection (5 total features)					Without feature selection (24 total features)	
Principle components	1	2	3	4	5	3	24
Accuracy (%)	85.4	85.5	85.2	85.2	85.6	73.6	61.6
Percentage of splitcases (%)	5.0	5.0	5.2	5.0	5.0	9.6	7.9

for the power correlation was also optimised. This involved finding the threshold value that produced the highest DPR accuracy. Since largest correlation value was 1, by definition, the threshold values that were tested were 0.9, 0.8, 0.7, 0.6 and 0.5, which corresponded to the different number of features being used for clustering.

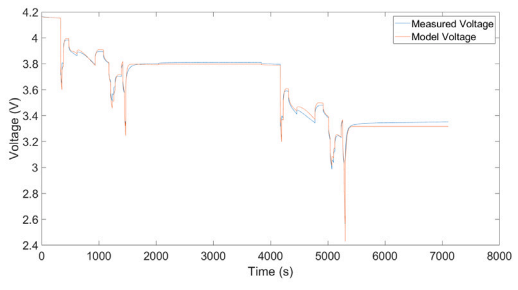
The results from this are shown in Table 6, with the corresponding number of features for each threshold value also shown. The threshold value which had the best accuracy was 0.7, which also had a low percentage of split cases. The accuracy for clustering with one feature was high as well, and by nature had no split cases. As expected, the larger threshold values produced reduced levels of accuracy by up to 7%, as the less relevant features were given the same weighting as the most relevant. From these results, the threshold was set at 0.7 for this data set.

Additionally, the length of the historical window was optimised to produce the best accuracy. To test this, the length of the moving window was changed and the accuracy was calculated. For comparison, a neural network (NN) was created with the MATLAB nftool [50], which is part of the MATLAB Deep Learning Toolbox. Here, the NN input was the relevant feature values of each microtrip, and the target data was the corresponding cluster number. The data for training, validation and testing were divided based on 70%, 15% and 15% of the total data set, respectively. For the training, 10 hidden neurons were used in hidden layer and the Levenberg–Marquardt algorithm was used to train the network. The trained NN was deployed as a function, which used the features values of the measurements in the moving window to output the cluster number, which corresponded to a certain driving pattern.

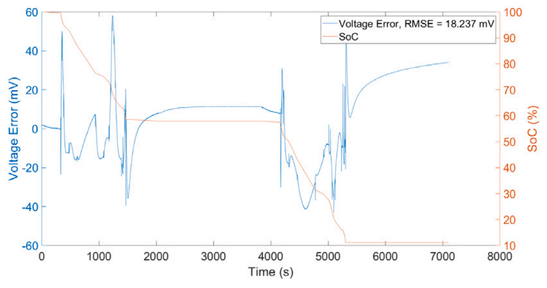
Table 6
Table showing the accuracy of the power pattern recognition accuracy for different values of the power correlation threshold, which determine the features used for clustering. Note, the maximum power correlation value is 1. The length of the moving window used was 300 s.

	Mean power correlation threshold value				
	0.9	0.8	0.7	0.6	0.5
Number of features	1	2	3	4	5
Accuracy (%)	89.7	87.3	92.7	87.9	85.5
Percentage of split cases (%)	0	14.9	1.1	14.4	5.0

The comparison between the accuracy of the rule-based logic and the NN are shown in Fig. 9. For short window lengths, the NN had better identification accuracy, but the results converge when the window length was around 600 s. The computational time was also calculated for the data set, where the rule-based logic took 32 s and the NN took 41 s. This shows a trade-off between the accuracy and computational complexity for short window lengths, but for long window lengths, the proposed logic had comparable accuracy for a smaller computational



(a)



(b)

Fig. 5. Example of results from validation tests, performed at 25 °C. (a) shows the experimental voltage that was measured from a cell, the model voltage using the battery model previously defined. (b) shows the voltage error results, the SoC from the model and the voltage error RMSE.

cost. Therefore, the rule-based logic, with a long historical window was used.

4.2. Power prediction

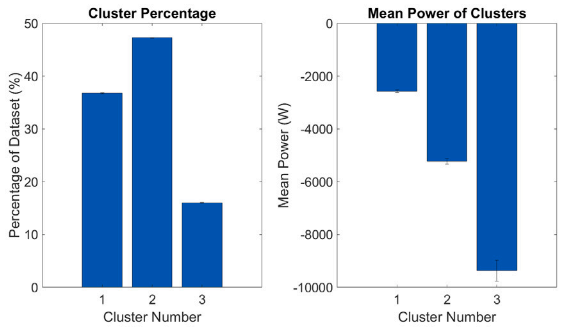
The goal of the power prediction method here was to provide a process for predicting future battery states in real-time. Because of this, the method aims to produce power profiles with the same key characteristics of the training data. Therefore, metrics need to be defined for comparing the profiles from the measurement data to the predicted profiles.

In [51], a metric was defined as the ratio of mean power to max power and was used measure how close the mean power was to the peak power for a given driving profile. An altered metric was implemented here, such that the max power was defined as a percentage of the maximum charge or discharge values, defined as

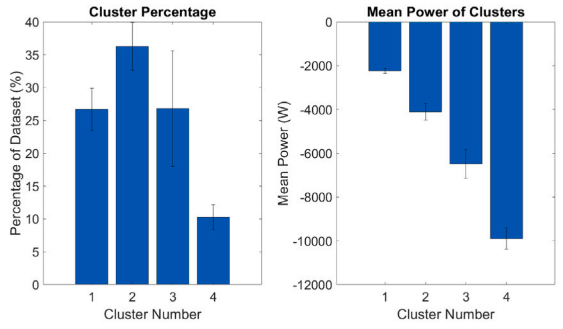
$$\Phi_{n\%} = \left| \frac{Total\ Mean\ Power}{Max\ n\%\ Power} \right|. \tag{9}$$

This change was justified as the average length of the profiles defined here was 7200 s, whereas, in [51], the average length was 1200 s.

The results for this metric are shown in Fig. 10, when the percentage of maximum power values was 1 and 10. Here, the data split into 70% training and 30% testing, so that the power prediction was compared against unseen data. The metric is used for both the charge and discharge components of the profiles. Here, it was shown that there was good agreement, in most cases, between the test data, which consisted of the real-world measurements, and the power predicted profiles. Overall, $\Phi_{10\%}$ performed better with the max discharge data and $\Phi_{1\%}$ performed better with the max charge data. The results that did not have good agreement were the charging parts of the low power profile, and to a lesser extent, the charging parts of the high power profile. This did not have a significant effect on the overall accuracy of the predicted profile, as there were four to five times more discharging points compared to charging points for the profiles present here.



(a) Results from three clusters used.



(b) Results from four clusters used.

Fig. 6. Bar plots of cluster percentage and mean power of the data points within each cluster, when different numbers of clusters are used. The error bars represent the standard deviation between the results from 50 runs of the clustering algorithm. The features present for this clustering are $\bar{v}$, v_{var}, a_{min}, $\bar{P}$, and P_{min}.

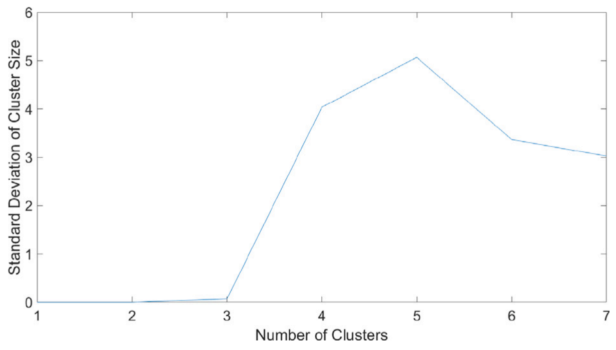


Fig. 7. Plot of the proposed cluster metric, standard deviation of cluster size, against the number of clusters. The features present for this clustering are $\bar{v}$, v_{var}, a_{min}, $\bar{P}$, and P_{min}.

4.3. Single cell simulation

For RDE estimation simulation, the obtained battery model was used to represent an actual battery's voltage response, therefore, there was no voltage or SoC error present. Before the simulations were performed, the power was scaled from the EV pack level to an appropriate cell level. With the scaled data, the high-power profile took one hour for a full discharge, the medium power profile took two hours for a full discharge, the low power profile took three hours for a full discharge. These profiles were generated and used for simulation, as well as an additional mixed power case which were generated from randomly selecting microtrip segments for the purpose of testing the algorithm when the power level do not remain constant.

For the four methods used here, various parameters were calibrated to optimise the RDE result. These included the number of clusters used to represent the different power levels and the length of the moving window. The metric used to justify the parameter choices was the

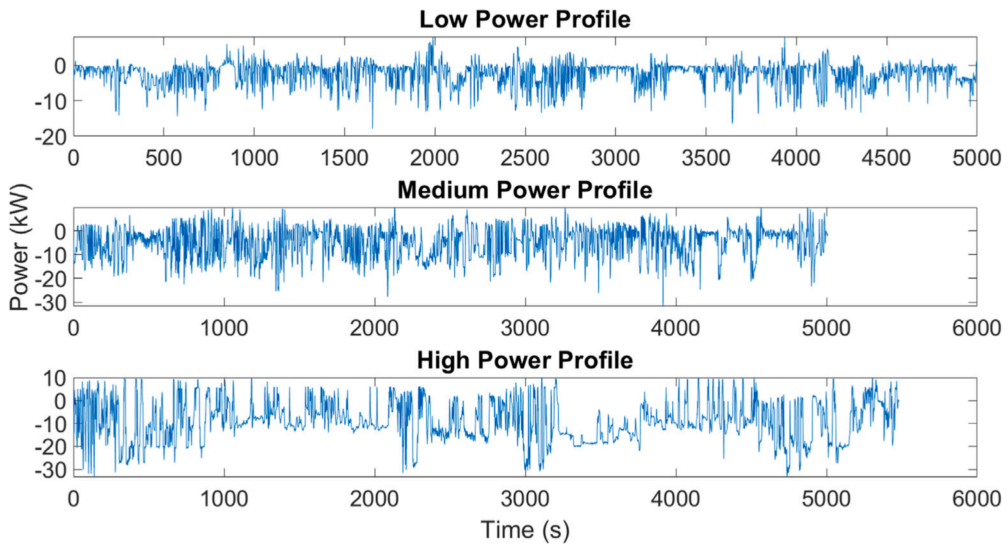


Fig. 8. Example power profiles generated from microtrips in each cluster. The plots are in ascending order of average power.

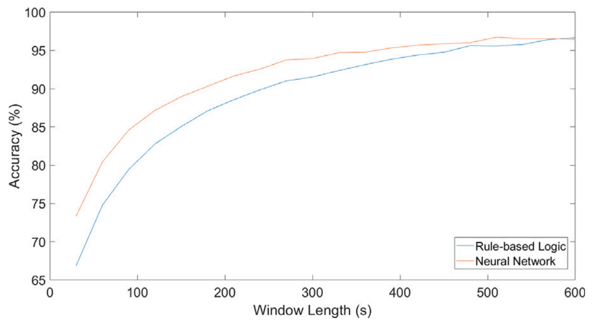


Fig. 9. Plot of the accuracy of the proposed logic and a NN against different historical window lengths, where the sampling time was 1 s. The computational time for the whole test data set for the rule-based logic was 32 s and the NN was 41 s.

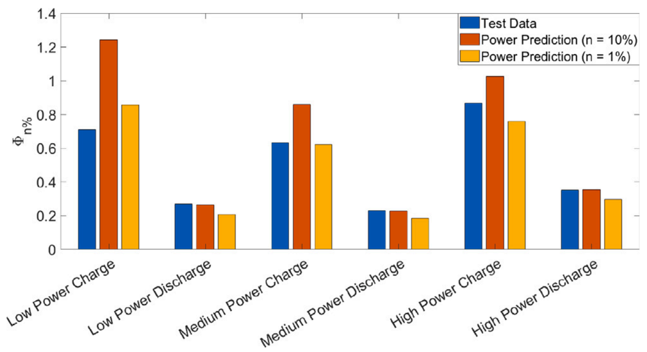


Fig. 10. Plot of $\Phi_{n\%}$ for 10% and 1%, comparing the results from the test data to the power prediction results, for each power case.

RMSE, as this indicated how accurate the method was for the whole profile. Here, the error was defined as the difference between the reference RDE and the RDE estimate. The reference RDE was calculated from the energy removed from a cell (using Eq. (1)), so that the RDE at the start was equal to the total energy removed. The number of clusters used to predict the future power was found to have little impact on the overall results, where two clusters was found to be the most suitable value, as shown in Table 7, for the proposed method. This trend was also present for the other methods. An additional test was also performed to justify both the use of representing the power levels as clusters and using Gaussian distributions. For this, when one cluster was used with no Gaussian distribution, the RMSE is 0.57 Wh, whereas when two clusters are used with Gaussian distribution, the RMSE is 0.32 Wh.

Table 7
Table showing the data from simulations using different cluster numbers for the load levels for load prediction in the RDE estimation, with the proposed DPR algorithm.

	Number of clusters						
	2	3	4	5	6	7	8
Low power RMSE (Wh)	0.76	0.83	0.88	0.86	0.88	0.89	0.89
Medium power RMSE (Wh)	0.35	0.41	0.46	0.45	0.46	0.47	0.51
High power RMSE (Wh)	0.28	0.29	0.29	0.30	0.32	0.31	0.33
Mixed power RMSE (Wh)	0.38	0.40	0.43	0.43	0.46	0.45	0.48
Overall RMSE (Wh)	0.44	0.48	0.51	0.51	0.53	0.53	0.55

For the length of the moving window, it was found that increasing the number of data points collected, the accuracy would increase, as well as decrease the total computational time, so a moving window of 1000 s was used.

For each of the driving patterns, five profiles were generated. These were used to determine which method had the greatest accuracy and the highest computational efficiency. The computational efficiency was determined by the total run time of the algorithm for a whole power profile and was based on a computer with an Intel(R) Core(TM) i7-8665U CPU @ 1.90 GHz and 16 GB of RAM. Table 8 shows the data from these simulations. These results come with three key points. (1) The offline-training method, without pre-processing, can provide RDE values from the initial time, but the offline-training method with DPR, can only provide reliable RDE values after time equal to the pattern recognition window has been reached, this was also the case for the moving window method. So, the RMSE values were calculated with RDE values from the same starting point in time. (2) The computational time shown was for the whole RDE estimation time for each implemented method, but for the moving window method, the direct calculation method was used before the window length time has passed, which took significantly less time than using power prediction. (3) The DPR provides information that can be useful for other areas of the control system, such as energy [31] and thermal management [52,53], at the cost of computational efficiency. Therefore, these should be considered when deciding the optimal accuracy vs relative computational intensity method.

As discussed previously, the direct calculation method has very poor accuracy when compared to prediction based methods. This is expected

Table 8
Table showing the data from simulations using different RDE estimation algorithms. Five different profiles for each profile type were tested, with the mean RMSE for each type shown, as well as the standard deviation in these results and the total computational time shown.

		Direct calculation	Moving window	Offline-training without pre-processing	Offline-training with DPR
Profile type					
Low power	Mean RMSE (Wh)	0.51	0.04	0.73	0.06
	Standard deviation (Wh)	0.01	0.00	0.02	0.04
	Total computational time (s)	34.41	101.94	48.20	97.45
Medium power	Mean RMSE (Wh)	0.95	0.26	0.34	0.26
	Standard deviation (Wh)	0.23	0.13	0.12	0.10
	Total computational time (s)	17.92	22.06	24.55	29.94
High power	Mean RMSE (Wh)	1.26	0.50	0.23	0.28
	Standard deviation (Wh)	0.18	0.20	0.09	0.06
	Total computational time (s)	7.94	7.45	11.61	12.51
Mixed power	Mean RMSE (Wh)	1.37	0.56	0.39	0.53
	Standard deviation (Wh)	0.42	0.27	0.22	0.25
	Total computational time (s)	14.48	16.74	20.35	26.00
Final mean RMSE (Wh)		1.02	0.34	0.42	0.28

since the equation used to calculate RDE did not take into account the current conditions of the cell. From the results shown, the accuracy of this method decreased when higher power profiles were used. This was attributed to the increased difference between the nominal voltage (used in the direct calculation equation) and the terminal voltage of the cell. Overall, the direct calculation method will only be suitable for application that require high computational speeds.

When comparing the prediction-based methods, there was no method that was universally the most accurate across all use cases. For the low power case, the offline-training method with DPR and the moving window method both had similar accuracy, which was much greater than the offline-training method with no data pre-processing. This was caused by the low power having a small range of possible values, which was unrepresentative of the entire data set. For the medium power profiles, the results were much closer, and for the high power profiles, the moving window method was the least accurate and the offline-training method with no pre-processing was the most accurate. This was because the possible range of values is very high. As for the mixed power case, the offline-training method and the moving window method were not suited to this type of profile, as the same power level was not sustained for a significant period of time. The method with no pre-processing provided the greatest accuracy for mixed power profiles. Therefore, two cases were defined, a constant power case and a mixed power case, where different methods were preferable for each case. For a specific use case where the current conditions are present for the whole cycle, the power pattern recognition method would be preferable, but if the conditions rapidly change unpredictably, the method where the whole training set is used for the prediction would be favoured.

For medium power profiles, since the accuracy between the DPR method and the moving window was the same, the standard deviation was used to select the most appropriate method. Since the standard deviation was smallest for the offline-training, with pattern recognition, this was the best choice for medium power. For high power profiles, the two offline-training methods showed similar accuracy, and when considering the standard deviation in the results, the choice of the best method became less significant. Therefore, the choice of the most suitable method should be made according to the specific application, and whether more accuracy is needed, or more repeatable results are needed, as well as the type of power profiles that will occur.

In terms of the total computational time, the direct calculation method was very efficient, especially when longer profiles were used. The moving window method was also efficient for all cases except the low power profiles. For the pattern recognition method, the additional computation cost came from the recognition component, compared to the other offline-training method. The significance of this time difference would be reduced if pattern recognition is necessary for the control system, or if it is implemented to improve the accuracy of other parts of the system. In addition, options for pattern recognition algorithms were not explored here, in terms of accuracy and computational efficiency, so this time difference could be reduced in the future work.

4.4. Real-time implementation

For the simulations, the testing conditions were made so the battery model producing a voltage response was the same as the model used for the RDE estimation. It is noteworthy that this was an idealised case, where no voltage error was present. So, it is important to test the proposed algorithm in real-time implementation (RTI) scenario using data from real cells. The cells used for validation were used to gather data based on the different driving profiles.

The hardware setup is shown in Fig. 11. Due to the limitations of the cycler, the setup used for this implementation involved obtaining data using a Maccor multi-range cycler from three 21,700 48X cells, which were previously used for validation. A low, a medium and a high power profile that were used for the simulations, were used as an input for the cycler. To avoid cycler errors from the inability to switch between very low current and very high current with high frequency, the profiles were re-interpolated at ten second intervals. The difference between the profiles is shown in Fig. 12(a) and Fig. 13(a). The voltage and current data was then used as an input for the algorithm. The algorithm was implemented on dSPACE Scalexio system and executed in real-time, with a sample time of 10 ms and a fixed-step discrete solver. Since three cells were used to obtain the data, three instances of the each RDE prediction-based algorithm were also used and performed at the same time. The algorithm also included an extended Kalman filter

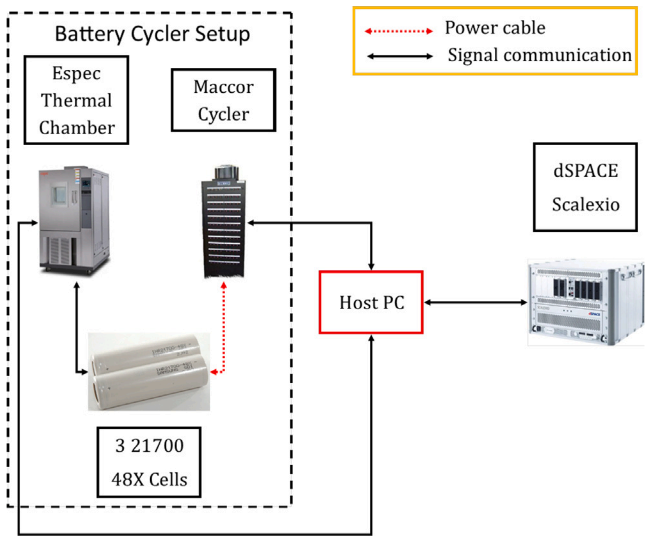


Fig. 11. Hardware diagram of real-time implementation setup, showing the components that were used and the connections between them.

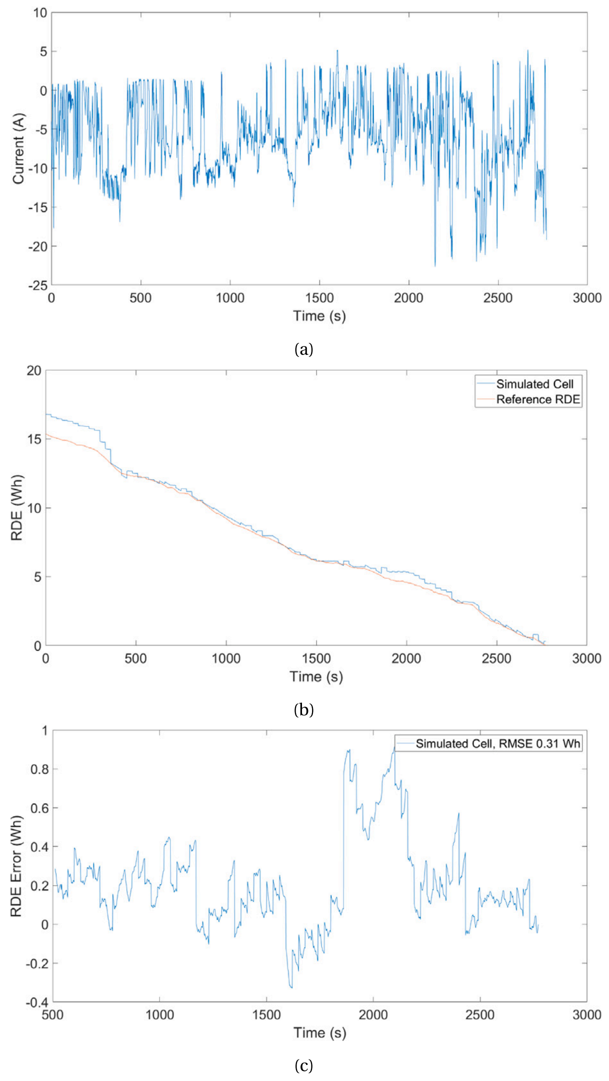


Fig. 12. Data from RDE simulations using a high power profile. (a) shows the corresponding current profile. (b) shows the RDE against time results, with the reference RDE regarded as the true value for RDE. (c) shows the RDE error and the calculated RMSE.

for SoC estimation, refer to [46,54] for the algorithm definition and implementation.

Figs. 12 and 13 show RDE results for the simulation and the RTI for the proposed method, respectively. The figures also show the current profile used, which were based on the same original profile. Once again, the reference RDE was calculated from the energy removed from a cell. In this case, the voltage and current values were taken from the battery cycler measurements. For the RTI, Fig. 13 shows the difference between the cells for RDE estimation. The differences were most likely caused by different random number generator seeds being used, as the cells used did not have significant cell-to-cell when characterised. Although, cell ageing was not examined here, the degradation caused by performing multiple high current validation cycles may have increased the cell-to-cell variations. When comparing the simulation result to the RTI results, all cells had similar RMSE as the simulation results. This was overall a positive result, as it shows the algorithm could achieve the same accuracy, even when in the presence of modelling errors, as well as add validity to the simulation results.

The three prediction-based methods were tested with high, medium and low power profiles, in real-time. All the results are shown in Table 9. In general, the moving window and offline-training with DPR methods performed well, achieving higher accuracy than in the simulations for the high and medium power profiles used. For these methods, the accuracy of the low power profile was significantly lower. This could have been caused by the modelling errors. The opposite trend occurred for the offline-training method without pre-processing, where it performed better than the simulations for the low power profile, but worse for the high power profile. Overall, the offline-training method produced slightly more accurate results that the moving window method in real-time.

5. Conclusion

Using power prediction methods to estimate RDE have been shown to be more accurate than alternative methods, such as SoC direct calculation and model-based methods. Here, previously implemented methods for power prediction were compared to the proposed DPR power prediction approach. The novelty of the proposed method was the combination of power prediction with DPR, as well as an investigation into how feature selection affected DPR accuracy. Firstly, the DPR approach was outlined, including methods for feature selection and clustering to group the microtrips into different driving categories. Each step was justified with numerical simulations to determine its effect on the recognition accuracy. Furthermore, a new method for determining the optimal number of clusters was proposed, which is effective for dense data sets. The simulations were then performed using the DPR approach and comparing the results to a moving window method and an offline training method, with no DPR. For four different types of power profile, the DPR approach was the most accurate, at the cost of computational speed. This approach was best suited for power profiles that maintain a comparable power level for a significant period of time. Next, the prediction-based algorithms were tested in real-time, using actual cell data. The results were compared showing the DPR method and the moving window method achieved a similar level of accuracy for the three cases, but with the DPR method producing information about the current driving conditions. Compared to the simulations, the methods showed an acceptable level of accuracy, considering the modelling errors present.

There are multiple areas of future work for this research. These can be split into two categories, system implementation and algorithm improvements. For the system implementation, the main component of this would include BMS integration, which would be achieved by testing the algorithm on an electronic control unit (ECU). In addition, testing the transferability of the algorithm to different battery

Table 9
Results of real-time implementation (RTI) using three prediction-based RDE estimation methods, for the same profiles, with the average RMSE from each profile calculated. The results for the same profiles in simulation are also shown.

		Simulation RMSE (Wh)	Cell 1 RTI RMSE (Wh)	Cell 2 RTI RMSE (Wh)	Cell 3 RTI RMSE (Wh)	Average RTI RMSE (Wh)
Moving window	High Power	0.31	0.27	0.30	0.29	0.29
	Medium Power	0.22	0.24	0.22	0.25	0.24
	Low Power	0.03	0.24	0.26	0.22	0.24
Offline-training without pre-processing	High Power	0.18	0.51	0.54	0.53	0.53
	Medium Power	0.45	0.43	0.40	0.46	0.43
	Low Power	0.76	0.60	0.55	0.62	0.59
Offline-training with DPR	High Power	0.31	0.29	0.30	0.29	0.29
	Medium Power	0.23	0.23	0.22	0.24	0.23
	Low Power	0.04	0.21	0.23	0.20	0.21

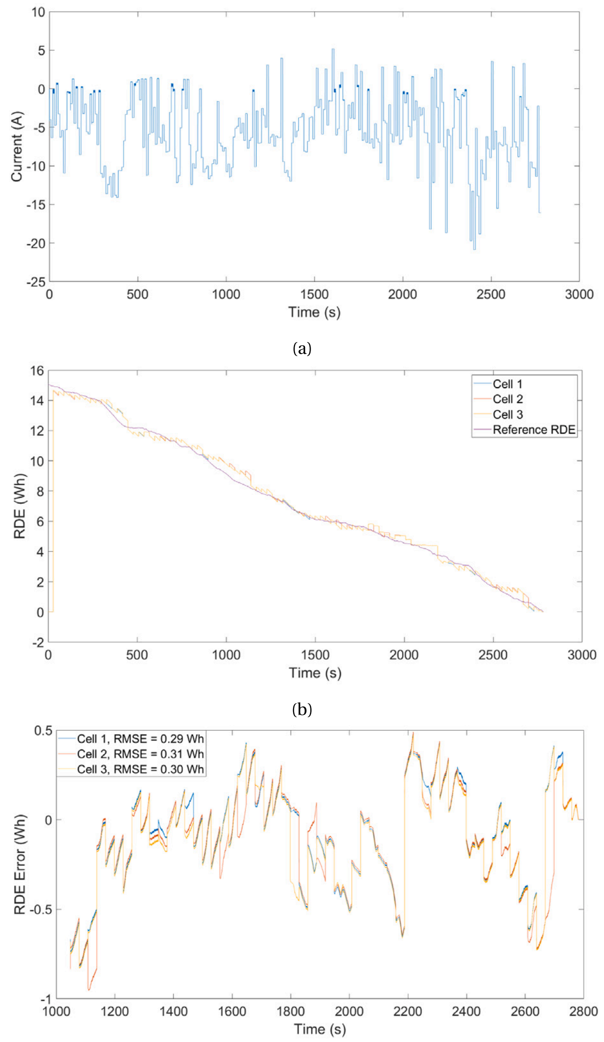


Fig. 13. Data from real-time RDE tests using the proposed DPR method for a high power profile, using three cells. (a) shows re-interpolated current profile that was used for simulation. (b) shows the RDE against time results, with the reference RDE regarded as the true value for RDE. (c) shows the RDE error and the calculated RMSE.

chemistries, form-factors and load profiles, which will require additional data sets. The algorithm improvements would include the implementation different power prediction techniques to improve RDE estimation accuracy, especially with computationally expensive data processing, as this can be performed offline when using the proposed RDE estimation method. Other improvements would also include implementation of a thermal model, which would aim to predict the future temperature to improve the RDE estimation accuracy, as well as scaling the RDE estimation algorithm up from a single cell to multiple cells.

CRediT authorship contribution statement

Ollie Hatherall: Conceptualization, Methodology, Software, Validation, Writing – original draft. **Mona Faraji Niri:** Conceptualization, Writing – review & editing, Supervision. **Anup Barai:** Writing – review & editing, Supervision. **Yi Li:** Writing – review & editing. **James Marco:** Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgement

This research was undertaken in collaboration Vertical Aerospace Group through the Warwick Collaborative Postgraduate Research Scholarship Scheme.

References

[1] J. Neubauer, E. Wood, The impact of range anxiety and home, workplace, and public charging infrastructure on simulated battery electric vehicle lifetime utility, J. Power Sources 257 (2014) 12–20, http://dx.doi.org/10.1016/j.jpowsour.2014.01.075.
[2] G. Liu, M. Ouyang, L. Lu, J. Li, J. Hua, A highly accurate predictive-adaptive method for lithium-ion battery remaining discharge energy prediction in electric vehicle applications, Appl. Energy 149 (2015) 297–314, http://dx.doi.org/10.1016/j.apenergy.2015.03.110.
[3] P. Shrivastava, T.K. Soon, M.Y.I.B. Idris, S. Mekhilef, Overview of model-based online state-of-charge estimation using Kalman filter family for lithium-ion batteries, Renew. Sustain. Energy Rev. 113 (2018) (2019) 109233, http://dx.doi.org/10.1016/j.rser.2019.06.040.
[4] D. Ren, L. Lu, P. Shen, X. Feng, X. Han, M. Ouyang, Battery remaining discharge energy estimation based on prediction of future operating conditions, J. Energy Storage 25 (July) (2019) 100836, http://dx.doi.org/10.1016/j.est.2019.100836.
[5] M.F. Niri, T.M. Bui, T.Q. Dinh, E. Hosseinzadeh, T.F. Yu, J. Marco, Remaining energy estimation for lithium-ion batteries via Gaussian mixture and Markov models for future load prediction, J. Energy Storage 28 (January) (2020) 101271, http://dx.doi.org/10.1016/j.est.2020.101271.
[6] A. Barai, K. Uddin, W.D. Widanalage, A. McGordon, P. Jennings, The effect of average cycling current on total energy of lithium-ion batteries for electric vehicles, J. Power Sources 303 (2016) 81–85, http://dx.doi.org/10.1016/j.jpowsour.2015.10.095.
[7] K. Mamadou, E. Lemaire, A. Delaille, D. Riu, S.E. Hing, Y. Bultel, Definition of a State-of-Energy Indicator (SoE) for Electrochemical Storage Devices: Application for Energetic Availability Forecasting, J. Electrochem. Soc. 159 (8) (2012) A1298–A1307.

[8] R. Xiong, Y. Zhang, H. He, X. Zhou, M.G. Pecht, A double-scale, particle-filtering, energy state prediction algorithm for lithium-ion batteries, IEEE Trans. Ind. Electron. 65 (2) (2017) 1526–1538.
[9] Y.Z. Zhang, H.W. He, R. Xiong, A Data-Driven Based State of Energy Estimator of Lithium-ion Batteries Used to Supply Electric Vehicles, Energy Procedia 75 (2015) 1944–1949.
[10] R. Ahmed S. Rahimifard, S. Habibi, Offline Parameter Identification and SOC Estimation for New and Aged Electric Vehicles Batteries, in: ITEC 2019-2019 IEEE Transportation Electrification Conference and Expo, 2019.
[11] X. Lai, Y. Huang, H. Gu, X. Han, X. Feng, H. Dai, Y. Zheng, M. Ouyang, Remaining discharge energy estimation for lithium-ion batteries based on future load prediction considering temperature and ageing effects, Energy 238 (2022) 121754, http://dx.doi.org/10.1016/j.energy.2021.121754.
[12] B. Basaran, F. Günes, Data clustering, Intell. Multidimens. Data Clust. Anal. 31 (3) (2016) 28–72.
[13] O. Nasraoui, C.-E. Ben N'Cir, Clustering methods for big data analytics, 2019, [Online]. Available: https://www.springer.com/gp/book/9783319978635.
[14] B. Mirkin, Mathematical classification and clustering, 1996.
[15] O. Hatherall, J. Marco, A. Barai, M. Faraji-Niri, Load prediction based remaining discharge energy estimation using a combined online and offline prediction framework, in: 2022 Conference on Control Technology and Applications, No. 2, 2022, pp. 1196–1201.
[16] B.Y. Liaw, M. Dubarry, From driving cycle analysis to understanding battery performance in real-life electric hybrid vehicle operation, J. Power Sources 174 (1) (2007) 76–88.
[17] M. Montazeri-Gh, A. Fotouhi, A. Naderpour, Driving patterns clustering based on driving feature analysis, Proc. Inst. Mech. Eng. C 225 (6) (2011) 1301–1317.
[18] X. Lin, G. Zhang, S. Wei, Velocity prediction using Markov Chain combined with driving pattern recognition and applied to Dual-Motor Electric Vehicle energy consumption evaluation, Appl. Soft Comput. 101 (2021) 106998, http://dx.doi.org/10.1016/j.asoc.2020.106998.
[19] R. Zhang, J. Tao, H. Zhou, Fuzzy optimal energy management for fuel cell and supercapacitor systems using neural network based driving pattern recognition, IEEE Trans. Fuzzy Syst. 27 (1) (2019) 45–57.
[20] C. Wei, Y. Chen, X. Li, X. Lin, Integrating intelligent driving pattern recognition with adaptive energy management strategy for extender range electric logistics vehicle, Energy 247 (2022) 123478, http://dx.doi.org/10.1016/j.energy.2022.123478.
[21] S. Aghabozorgi, A. Seyed Shirkhorshidi, T. Ying Wah, Time-series clustering - A decade review, Inf. Syst. 53 (2015) 16–38, http://dx.doi.org/10.1016/j.is.2015.04.007.
[22] L. Yu, H. Liu, Efficient feature selection via analysis of relevance and redundancy, J. Mach. Learn. Res. 5 (2004) 1205–1224.
[23] G. Chandrashekar, F. Sahin, A survey on feature selection methods, Comput. Electr. Eng. 40 (1) (2014) 16–28, http://dx.doi.org/10.1016/j.compeleceng.2013.11.024.
[24] L. Parsons, E. Haque, H. Liu, Subspace Clustering for High Dimensional Data: A Review, Sigkdd Explorations 6 (1) (2004) 90–105.
[25] B. Dash, D. Mishra, A. Rath, M. Acharya, A hybridized K-means clustering approach for high dimensional dataset, Int. J. Eng. Sci. Technol. 2 (2) (2010) 59–66.
[26] S. García, J. Luengo, F. Herrera, Data preprocessing in data mining, 2015, p. 72.
[27] S.M. Holland, Principal components analysis (PCA), 2019.
[28] K. Wagstaff, C. Cardie, S. Rogers, S. Schrödl, Constrained K-means clustering with background knowledge, in: International Conference on Machine Learning ICML, 2001, pp. 577–584, [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.90.4624&rep=rep1&type=pdf.
[29] N. Dhanachandra, K. Manglem, Y.J. Chanu, Image Segmentation Using K-means Clustering Algorithm and Subtractive Clustering Algorithm, Procedia Comput. Sci. 54 (2015) 764–771, http://dx.doi.org/10.1016/j.procs.2015.06.090.
[30] T. Kodinariya, P. Makwana, Review on determining number of Cluster in K-Means Clustering, Int. J. Adv. Res. Comput. Sci. Manag. Stud. 1 (6) (2013) 90–95.
[31] J. Shi, B. Xu, Y. Shen, J. Wu, Energy management strategy for battery/supercapacitor hybrid electric city bus based on driving pattern recognition, Energy (2021) http://dx.doi.org/10.1016/j.energy.2021.122752.
[32] J. Hu, D. Liu, C. Du, F. Yan, C. Lv, Intelligent energy management strategy of hybrid energy storage system for electric vehicle based on driving pattern recognition, Energy 198 (2020) 117298, http://dx.doi.org/10.1016/j.energy.2020.117298.
[33] S. Aranganayagi, K. Thangavel, Clustering categorical data using silhouette coefficient as a relocating measure, in: Proceedings - International Conference on Computational Intelligence and Multimedia Applications, ICCIMA 2007, Vol. 1, 2008, pp. 13–17.
[34] U. Braga-Neto, Fundamentals of pattern recognition and machine learning, 2020.
[35] T.F. Covões, E.R. Hruschka, Towards improving cluster-based feature selection with a simplified silhouette filter, Inform. Sci. 181 (18) (2011) 3766–3782.
[36] X. He, D. Cai, Y. Shao, H. Bao, J. Han, Laplacian regularized Gaussian mixture model for data clustering, IEEE Trans. Knowl. Data Eng. 23 (9) (2011) 1406–1418.
[37] D. Pandove, S. Goel, R. Rani, Systematic review of clustering high-Dimensional and large datasets, ACM Trans. Knowl. Discov. Data 12 (2) (2018).
[38] G. Singh, A. Panda, S. Bhattacharyya, T. Srikanthan, Vector quantization techniques for GMM based speaker verification, in: ICASSP, IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings, Vol. 2, 2003, pp. 65–68.
[39] L.R. Rabiner, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, Proc. IEEE 77 (2) (1989) 257–286.
[40] T.W. Anderson, L.A. Goodman, Statistical Inference About Markov Chains, Ann. Math. Statist. 28 (03) (1957) 89–110.
[41] J.R. Beck, S.G. Pauker, The Markov Process in Medical Prognosis, Med. Decis. Making 3 (4) (1983).
[42] USABC Electric Vehicle Battery Test Procedures Manual, No. January, United States Department of Energy, Washington, DC, USA, 1996.
[43] INL, U. S Department of Energy Vehicle Technologies Program Battery Test Manual For Electric Vehicles, 2014.
[44] M. Petzl, M.A. Danzer, Advancements in OCV measurement and analysis for lithium-ion batteries, IEEE Trans. Energy Convers. 28 (3) (2013) 675–681.
[45] S.S. Madani, E. Schaltz, S.K. Kær, A Review of Different Electric Equivalent Circuit Models and Parameter Identification Methods of Lithium-Ion Batteries, ECS Trans. 87 (1) (2018) 23–37.
[46] R. Xiong, F. Sun, X. Gong, C. Gao, A data-driven based adaptive state of charge estimator of lithium-ion polymer battery used in electric vehicles, Appl. Energy 113 (2014) 1421–1433, http://dx.doi.org/10.1016/j.apenergy.2013.09.006.
[47] H. Dai, T. Xu, L. Zhu, X. Wei, Z. Sun, Adaptive model parameter identification for large capacity Li-ion batteries on separated time scales, Appl. Energy 184 (2016) 119–131, http://dx.doi.org/10.1016/j.apenergy.2016.10.020.
[48] W. Waag, S. Käbitz, D.U. Sauer, Experimental investigation of the lithium-ion battery impedance characteristic at various conditions and aging states and its influence on the application, Appl. Energy 102 (2013) 885–897, http://dx.doi.org/10.1016/j.apenergy.2012.09.030.
[49] Mathworks, Solve nonlinear curve-fitting (data-fitting) problems in least-square sense - MATLAB lsqcurvefit, 2013, [Online]. Available: mathworks.com/help/optim/ug/lsqcurvefit.html.
[50] Mathworks, Open Neural Net Fitting app - MATLAB nftool. [Online]. Available: https://www.mathworks.com/help/deeplearning/ref/nftool.html.
[51] Q. Kellner, W. Dhammika Widanage, J. Marco, Battery power requirements in high-performance electric vehicles, in: 2016 IEEE Transportation Electrification Conference and Expo, ITEC 2016, No. Table 3, 2016, pp. 1–6.
[52] M. Kandidayeni, A. Macias, L. Boulon, S. Kelouwani, Investigating the impact of ageing and thermal management of a fuel cell system on energy management strategies, Appl. Energy 274 (2019) (2020) 115293, http://dx.doi.org/10.1016/j.apenergy.2020.115293.
[53] J. Jeffs, T.Q. Dinh, W.D. Widanage, A. McGordon, A. Picarelli, Optimisation of direct battery thermal management for evs operating in low-temperature climates, Energies 13 (22) (2020) 1–35.
[54] G.L. Plett, Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs - Part 1. Background, J. Power Sources 134 (2) (2004) 252–261.