IJNRD.ORG ISSN : 2456-4184
INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD) | IJNRD.ORG
An International Open Access, Peer-reviewed, Refereed Journal

A STATE OF CHARGE ESTIMATION FOR CONNECTED SOLAR BATTERY SYSTEM USING ANFIS MATLAB

[1]Yashaswani Singh,[2] G. S. Sailesh Babu

[1]Student,[2]Professor
[1]Department of Electrical Engineering,
[1]Dayalbagh Educational Institute, Agra, India

Abstract

Estimating the state of charge is critical in all battery-related applications. The process of determining the current state of charge of a battery or energy storage system at any given time is known as real-time state of charge estimation (SoC). It is an essential component of battery management systems because it optimizes battery performance and extends battery life. Real-time SoC estimation has become critical in a variety of applications, such as electric vehicles, renewable energy systems, and portable electronics, where accurate and dependable battery performance is critical. This paper aims to estimate the SoC of a battery bank in real-time using the ANFIS MATLAB tool.

Keywords-Battery,SoC,ElectricVehicles,ANFIS,MATLAB

1. INTRODUCTION

A battery behaves differently since it is an electrochemical product. in a variety of operating and environmental circumstances. The indeterminacy of a battery's performance presents a problem for the execution of these operations.
The state of charge (SoC) of a cell represents the available capacity as a function of the rated capacity. SoC is measured in percentage points ranging from 0% to 100%. A SoC of 100% indicates that the cell is fully charged, whereas a SoC of 0% indicates that the cell is completely discharged. In practical applications, the SoC is not permitted to exceed 50%. Similarly, as a cell ages, it loses the capacity to which it is capable. This means that a 100% SoC for an aged cell is equivalent to a 75%-80% SoC for a new cell. The inverse of SoC is called Depth of Discharge (DoD).The inverse of SoC is called Depth of Discharge (DoD).SOC estimation is a fundamental challenge for battery use. The SOC of a battery, which is used to describe its remaining capacity, is a very important parameter for a control strategy [1].As the SOC is an important parameter, which reflects the battery performance, so accurate estimation of the SOC can not only protect battery, prevent over discharge, and improve the battery life but also allow the application to make rational control strategies to save energy [2].

The traditional methods for estimating the SOC are divided into two categories:
1. Establishing a mathematical relationship between SOC and one or more battery variables, such as battery open-circuit voltage, EMF, electrolyte specific gravity, and DC internal resistance or AC internal impedance.
2. Using the coulomb counting method, measure the energy that enters or exits the battery.
Traditional methods for computing SOC have many drawbacks because they use equivalent circuit battery models, which have many disadvantages, including the
inability to accurately represent the behavior of the battery. The models are not universal because they may produce inaccurate SOC estimates for different battery types and discharge conditions while producing adequate results for other battery types and discharge conditions. Any type of battery has a highly nonlinear characteristic. The other major drawback of these methods is their inability to calculate the SoC in real-time.
Real-Time SoC estimation is crucial for the following reasons:
1. To prolong the lifespan of the battery: Knowing the current SoC helps prevent overcharging and deep discharging, which can damage the battery and reduce its lifespan. Real-time SoC estimation allows for precise management of the charging and discharging of the battery.
2. To optimize battery performance: Real-time SoC estimation helps optimize battery performance by ensuring that the battery is used at its maximum capacity without over-discharging or undercharging.
Battery behavior is strongly influenced by battery age as well as operating conditions such as temperature, discharge rate, and the spread of behavior among batteries of the same type.By updating the training data, ANN can easily adapt to any recent conditions.
In a real-world battery management system, fuzzy logic can handle uncertainties and imprecision.

The SOC estimator based on ANFIS technique, is a combination of ANN and the fuzzy logic system, will present an adaptive and accurate SOC estimator under different battery operating conditions and can easily be implemented by a low-cost microcontroller.
Adaptive Neuro-Fuzzy Inference Systems (ANFIS) are a combination of Artificial Neural Network (ANN) and Fuzzy Logic that can predict the battery's state of charge (SOC).

3. CLASSIFICATION OF SoC ESTIMATION TECHNIQUES

The various mathematical methods of estimation are classified according to methodology. The classification of these SOC estimation methods is different in the various literatures. However, some literatures [3, 4] allow a division into the following categories.The SOC is one of the most important parameters for batteries, but its definition presents many different issues [3].In general, the SOC of a battery is defined as the ratio of its current capacity () to the nominal capacity (). The nominal capacity is given by the manufacturer and represents the maximum amount of charge that can be stored in the battery.

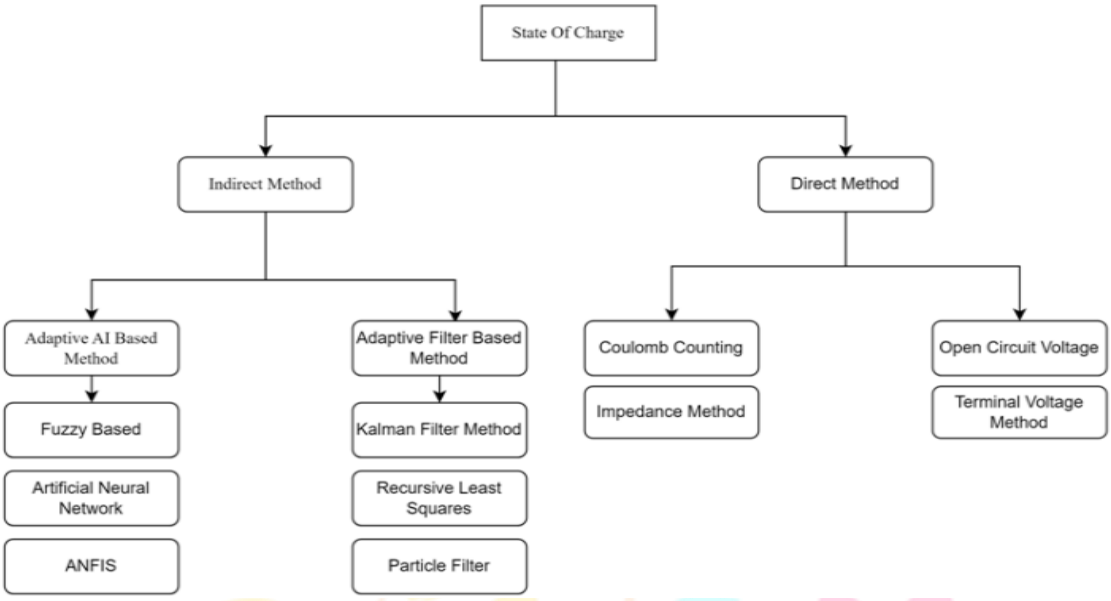


Fig.1; Classification of SoC estimation techniques

2.1 Direct Method

Some physical battery properties, such as terminal voltage and impedance, are measured directly. Several different direct methods have been used: Methods of open circuit voltage, terminal voltage, impedance measurement, and impedance spectroscopy [5].

2.1.1 Coulomb Counting Method

Coulomb counting is a popular method for calculating the SoC of lithium-ion batteries, which are used in electric vehicles, portable electronics, and renewable energy systems. It is a dependable method for tracking battery charge, but it necessitates precise current flow measurement and careful calibration to account for variations in battery capacity over time. The effects of temperature, ageing, and changes in battery chemistry are some of the limitations of Coulomb counting, which can affect the accuracy of the estimate over time.

2.1.2 Open Circuit Voltage

This method necessitates a longer battery resting time, and is thus referred to as offline prediction. When a battery is disconnected from a load, OCV is the voltage difference in electric potential between its terminal

2.2 Indirect Method

These methods rely on mathematical models or algorithms that use measurements of battery voltage, temperature, and other parameters to estimate the SOC, rather than directly measuring the charge that flows into or out of the battery.

2.2.1 Extended Kalman Filter

There are several works in literature attempting to estimate SoC using Kalman Filter; majority of them being based on electrochemical models or ad hoc models[1] .It functions as a prediction correction model that is independent of the precision of the battery model. This estimation employs highly complex matrix operations, which may result in numerical instability.

2.2.2 Recursive Least Square Method

The RLS method for estimating the SOC of a battery works by modeling the battery as a dynamic system that is subject to various inputs and disturbances. The state of the battery can be represented as a vector that includes the SOC and other relevant parameters such as the battery voltage and current.

2.2.3 Fuzzy Logic

Fuzzy logic allows complex systems to be modeled using a higher level of abstraction created from our knowledge and experience,

and it resembles human decision making. Fuzzy logic enables the expression of knowledge using abstract concepts such as big, small, very hot, fast or slow, and so on. It is frequently employed in automatic control systems. It is used to provide a more accurate estimate of a battery's SOC or SOH.

Fuzzy logic method provides a powerful means of modeling nonlinear and complex systems. In [8], a practical method of estimating SOC of battery system has been developed and tested for several systems. The method involves the use of fuzzy logic models to analyze data obtained by impedance spectroscopy and/or Coulomb counting methods. In [9], a fuzzy logic-based SOC estimation method has been developed for lithium-ion batteries for potential use in portable defibrillators. The ac impedance and voltage recovery measurements have been made which are used as the input parameters for the fuzzy logic model.

Singh et al. [10] presented an estimation system which can select features in data base to develop fuzzy logic models for both available capacity and SOC estimation, simply by measuring the impedance at three frequencies. In [11], the SOC is estimated by an improved Coulomb metric method, and the time-dependent variation is compensated by using a learning system. The learning system tunes the Coulomb metric method in such a way that the estimation process remains error free from the time-dependent variation. The proposed learning system uses the fuzzy logic models, which is not used for estimation of SOC but performs as a component of learning system.

The steps to creating a fuzzy rule-based model are as follows:

1. The fuzzy input and output variables are determined first.
2. The fuzzy sets for each variable are then determined.
3. The membership functions of all fuzzy inputs and outputs are generated in the third step.Membership functions come in a variety of shapes and sizes, such as triangular, Gaussian, trapezoidal, and so on. Because the type of membership functions has an impact on the design of the fuzzy logic controller, they must be carefully chosen.
4. Fourth, fuzzy IF-THEN rules are created to connect the input and output variables.
5. Fifth, the inference procedure has been established

The most prevalent FIS types are Sugeno and Mamdani. Sugeno generates linear or constant output, whereas Mamdani is employed for membership functions such as trapezoidal, triangular, and so on. Sugeno is taught with the use of a data collection. Mamdani does not require a data set, instead relying on expert knowledge. The Mamdani type contains the following procedures. The input variables are fuzzified so that the degree to which they fit each of the fuzzy sets is determined using membership functions. To produce a single number, the inputs are joined using a "AND" or "OR" fuzzy operator. The rule's weight is then determined before the implication is applied to each rule. The fuzzy rules are then applied[7].

2.2.4 Neural Network

A neural network is a computing system made up of interconnected nodes that resemble the structure of the human brain. It is intended to learn and recognise patterns in data and then make decisions based on that knowledge. The network is trained on a set of data and can identify and classify patterns in new data using algorithms. Every communication channel is paired with a weight that holds information about the input signal. Because the weight normally excites or prevents the signal from being communicated, this is the most valuable knowledge for neurons to solve a specific problem. Each neuron has an internal state called an activation signal. The output signals generated by combining input signals with the activation rule can be transmitted to other units [12].Image recognition, natural language processing, and prediction models are all examples of how neural networks are used. They are regarded as one of the most powerful machine learning techniques available, but they necessitate a large amount of data and computational power to train and operate efficiently.

4. ANFIS MATLAB

FL systems can generalize any system employing cycle number estimation since it is easier to explain the status of the battery (High, Low) in some battery tests rather than getting a precise figure.By upgrading the training data, ANN can easily adjust to any recent conditions.ANFIS is a hybrid intelligence paradigm that combines the benefits of fuzzy systems and adaptive networks. When the flexibility and subjectivity of fuzzy inference systems are combined with the optimisation power and learning potential of adaptive networks, ANFIS achieves amazing modeling, approximation, nonlinear mapping, and pattern recognition capabilities. ANFIS (Adaptive Neuro Fuzzy Inference System) is a type of artificial intelligence system that combines the power of Neural Networks and Fuzzy Logic for systems modeling and control. MATLAB provides a toolbox for creating ANFIS models for data classification and prediction.

The ANFIS MATLAB toolbox allows you to train and optimize Fuzzy Inference Systems using a hybrid learning algorithm that combines gradient descent and least-squares estimation. The toolbox also provides functions for plotting and testing the ANFIS model, as well as for generating C-code for deployment on embedded systems [7].

In the fuzzy control system, fuzzy reasoning is a map to the relationship of input-output. The neurons can map any function relationships, it also can be used to make the fuzzy inference come true. Also, the neural network can realize fuzziness and non-fuzziness. Hence the neural network can represent all fuzzy control[3]

ANFIS can be used to design input-output mapping based on the initial given fuzzy system and available input-output data pairs. The Mamdani inference system and the Sugeno inference system are two of the most widely used fuzzy inference systems (FIS) in a variety of applications.

In MATLAB, the ANFIS algorithm is implemented as follows:

3.1 Define the input and output data sets

The first step is to specify the data sets that will be used to train the ANFIS network. These data sets should include a sufficient number of samples that represent the nonlinear system under consideration.

3.2 Fuzzification

The input variables are then fuzzified. The process of mapping input variables to fuzzy sets is known as fuzzification. The membership function of these fuzzy sets should be determined by the system under consideration.

3.3 Rule Generation

The rule generation step generates the fuzzy if-then rules that represent the nonlinear system under consideration. The fuzzy sets obtained from define the if-part of each rule.

3.4 Membership computation

The next step is to compute the degree of membership of each fuzzy set's input variables. This is accomplished by utilizing the membership functions defined in step 3.2.

3.5 Rule Evaluation

The fuzzy if-then rules generated in step 3.3 are evaluated in this step based on the degree of membership of the input variables in each fuzzy set.

3.6 Defuzzification

The defuzzification of the output variables is the final step in the ANFIS algorithm. The results of the rule evaluation step are combined to produce crisp outputs.
The preceding steps are repeated until the difference between the predicted and actual output is less than a predefined threshold. The ANFIS algorithm has been shown to be effective in modeling and predicting a wide range of variables.

5. PROPOSED MODEL AND ALGORITHM

The ANFIS MATLAB tool is used to estimate real-time SoC of the solar battery system. The dataset contains the real-time recorded Battery Voltage, Battery Current, Battery temperature for 400AH Lead-Acid Battery collected at an interval of 10 minutes, with the output being Energy in KWH

4.1 Dataset

Every Machine learning algorithm requires a lot of dataset to learn and generalize patterns and relationships in the data. The more data that is available, the more diverse and comprehensive the patterns that can be identified and learned by the algorithm. With more data, the model can better handle complex and noisy data, reducing the impact of outliers or missing values. It can also better capture the various features and nuances of the data, resulting in more accurate predictions and better performance. But , it is not possible to acquire a dataset of this magnitude for a real-time battery system hence interpolation techniques were used.

By using interpolation techniques to generate additional data points, machine learning models can be trained on larger and more complete datasets. When dealing with data that is irregularly spaced or scattered, scattered interpolation is usually preferred. Hence for the original 980 unique data points out of 10,780 , we were able to generate 35,146 , leading to a total of 45,926 datapoints,

4.2 Training Data:

The training data is the largest (in terms of size) subset of the original dataset used to train or fit the machine learning model.The training data for unsupervised learning contains unlabeled data points, which means that the inputs are not tagged with the corresponding outputs. In order to make predictions, models must find patterns in the given training datasets.

The next stage is to investigate the impact of training data size. The trial was divided into four scenarios. We pre-trained and re-trained the model using all (100%) available training data in the first case. The second third and fourth scenarios used three fourth (70%), half of the training data and one-third of the training data, respectively. We recorded the training duration and error data in all circumstances.

To demonstrate the small amount of training time required to obtain low error rates, all training in this section was completed for only 10 epochs.The results are shown in Table 1. We see that when we pre-trained and re-trained the model on all available data (row 1), the error metric is lower than when we did not pre-train the model (row 3).Both took approximately the same duration of time.

Table 1. The influence of pre-training on the training time, training data amount, and cross-dataset generalization. *PT Pre-training, RT Re-training, T Training.

Training Approach	Data	Time Taken (sec)	Train RMSE	Test RMSE
PT+RT	100	26.22	0.15406	0.095218
T	100	14.77	0.15489	1.1249
PT+RT	70	24.03	0.15748	0.09721
T	70	12.07	0.15812	1.1349
PT+RT	50	20.88	0.17437	0.17437
T	50	11.14	0.17524	1.1669
PT+RT	30	18.93	0.1509	1.16978
T	30	10.51	0.15184	1.18619

Unsurprisingly, the highest performing option is when the model is pre-trained and re-trained on 100% datapoints, whereas the worst performing mode is when the model is pre-trained on 30% datapoints. However, when the model was pre-trained and re-trained on 70% of the datapoints, the test error rate and time consumed were practically identical to the top performing model. This shows that pre-training can increase ANFIS model accuracy by providing a better starting point for fine-tuning.

Pre-training produces an initial set of fuzzy rules based on the training data's input-output relationship, which can assist avoid the model from fitting the noise in the data. In the scenario where the model was trained on less data (30% of the training data), we see that the model produces large test errors without pre-training (row 9). Rows 7 and 8 show that pre-training the model reduces the error rate. This demonstrates that, regardless of cell type, pre-training contributes to reducing test set error.

4.2 Test Dataset

During the training phase, the ANFIS algorithm learns patterns in the input data and adjusts its parameters to minimize the error between its predictions and the actual outputs. To ensure that the model is generalizing well and is not over fitting to the training data, we need to test it on a separate dataset that the model has not seen before. By evaluating the model on the testing data, we can estimate its performance on new data, and we can also compare the model's predicted outputs to the actual outputs to measure its accuracy. As a general rule of thumb, it is recommended to use around 20% to 30% of the available dataset for testing purposes.This data set majorly comprised of the recorded dataset along with scattered interpolation generated dataset.

4.3 FIS Generation

FIS generation is a crucial step in the ANFIS (Adaptive Neuro-Fuzzy Inference System) process. It involves generating the Fuzzy Inference System (FIS) that will be used to estimate the system output based on the input data. The FIS is a combination of fuzzy rules, membership functions, and fuzzy logic operators.

The ANFIS FIS Generation component creates a Sugeno fuzzy inference system (FIS) with a single output and tunes the system parameters using the specified input/output training data. Grid partitioning is used to generate the FIS object automatically..

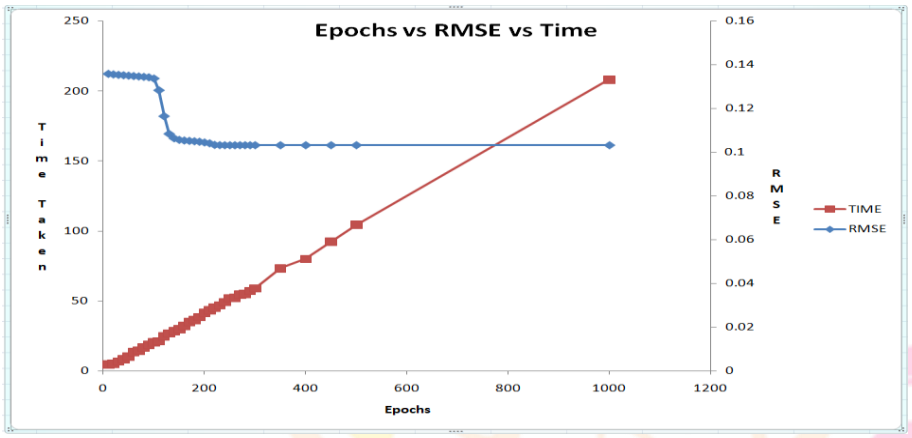
4.4 Optimization Method

The Fuzzy Logic Toolbox software employs a back-propagation algorithm, either alone or in conjunction with a least-squares algorithm, to train a fuzzy system using ANFIS. This training process fine-tunes a FIS's membership function parameters so that the system can model your input/output data.The resulting FIS is capable of accurately modeling complex systems with non-linear relationships between the inputs and outputs.

4.5 Epochs & its Estimation

An epoch is one cycle of training the neural network with all of the training data. During an epoch, we only use all of the data once. A forward and backward pass combined counts as one pass. An epoch is made up of one or more batches in which we train the neural network using a portion of the dataset.
The value of Epochs is determined by test running the training dataset and generating the FIS model by changing the value of epochs and recording the RMSE error along with the time taken to complete the run from epoch = 10 to 1000 progressively.

Fig.2; RMSE vs. Epochs vs. time taken graph



From the experiment it can be observed that the RMSE (Root Mean Square Error) remains constant as 0.10319 after 240 epoch cycles till 1000. In machine learning, it is common for the training process to reach a point where the reduction in error starts to slow down and eventually stops, even after additional epochs. This phenomenon is often referred to as "convergence."The reason why the RMSE error in the model stops reducing after a certain number of epochs is that the model has learned as much as it can from the available data. In other words, the model has reached its optimal level of performance given the training data and cannot further reduce the error.

The second observation from this work is that there is rapid decline in the RMSE error at first from the points where epochs is varied from 90 to 140, but it slows down after 240 to 1000.This phenomenon is due to the fact that the optimization algorithm used in ANFIS, as in most machine learning models, tries to minimize the error iteratively by adjusting the parameters of the model.

At the beginning of the training process, the optimization algorithm makes large adjustments to the model's parameters to quickly reduce the error. However, as the training progresses and the model gets closer to the optimal solution, the optimization algorithm needs to make smaller and more precise adjustments to avoid overshooting the optimal solution.

Therefore, the decrease in the RMSE initially happens more rapidly because the model makes significant progress by making large adjustments to the parameters. As the model gets closer to the optimal solution, the algorithm makes smaller and smaller adjustments, and the decrease in the RMSE slows down.

As observed from the graph the RMSE (Root Mean Square Error) remains constant as 0.10319 after 240 epoch cycles till 1000. Considering the time taken to run the complete dataset and error generated from it, epochs was set as 240.

6. SIMULATED MODEL AND RESULTS

The simulated ANFIS MATLAB model is shown in the following figures. The inputs in1, in2, in3 are real-time recorded Battery Voltage, Battery Current, Battery temperature for 400AH Lead-Acid Battery collected at an interval of 10 minutes, with the output being Energy in KWH

a. Training ANFIS Model

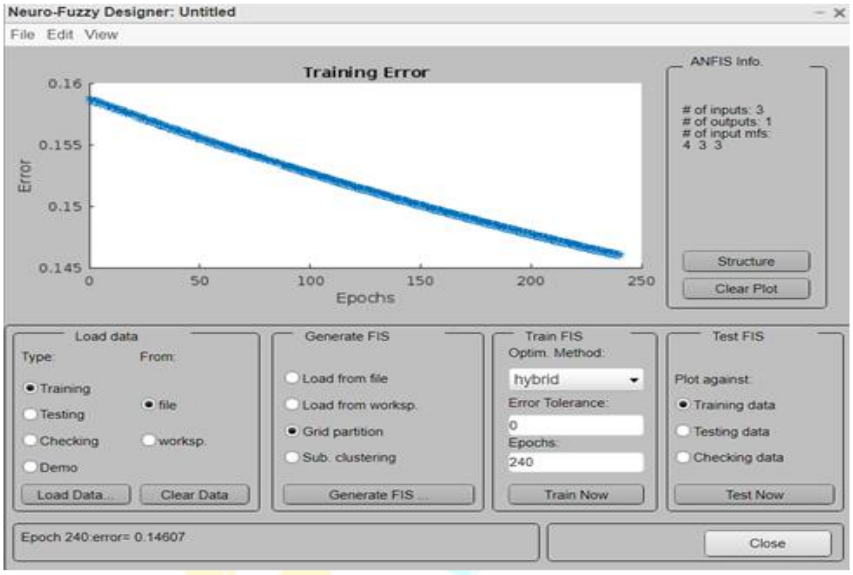


Fig. 3 Training FIS Model with epochs 240

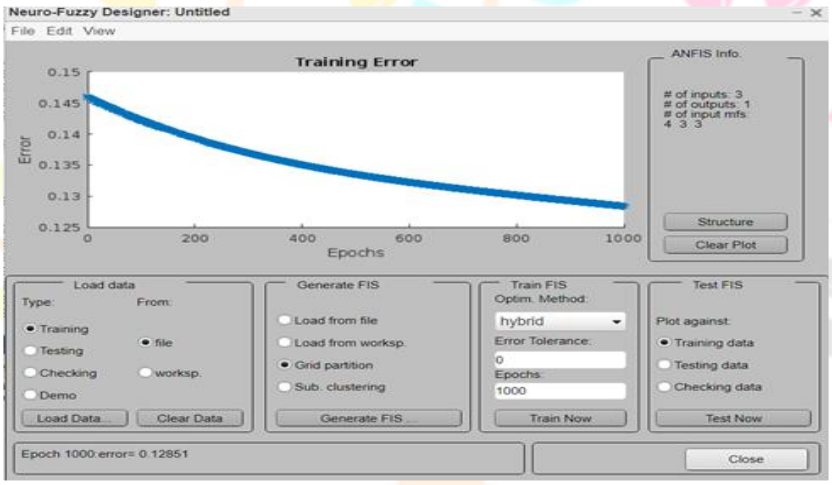


Fig.4: Re-Training Model with epochs 10000

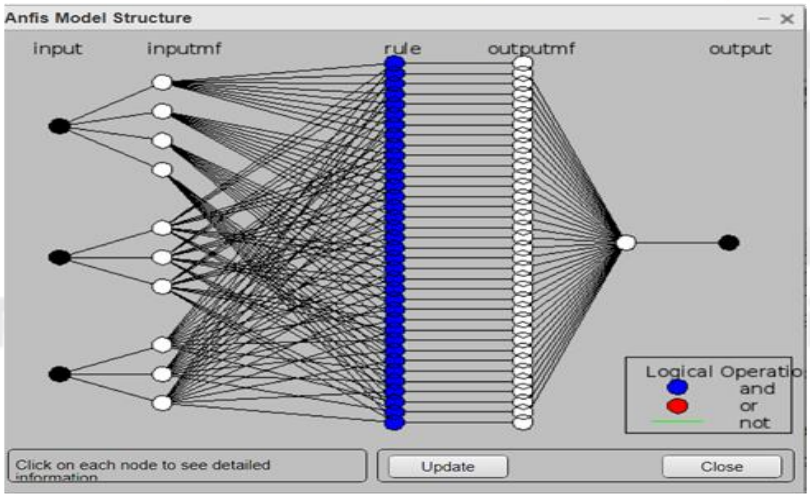


Fig. 5: Model Structure ANFIS

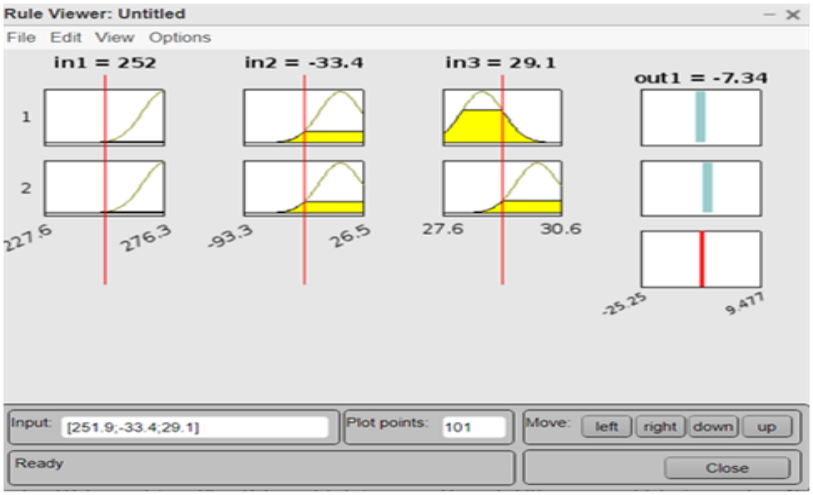


Fig. 6: ANFIS Rule Viewer

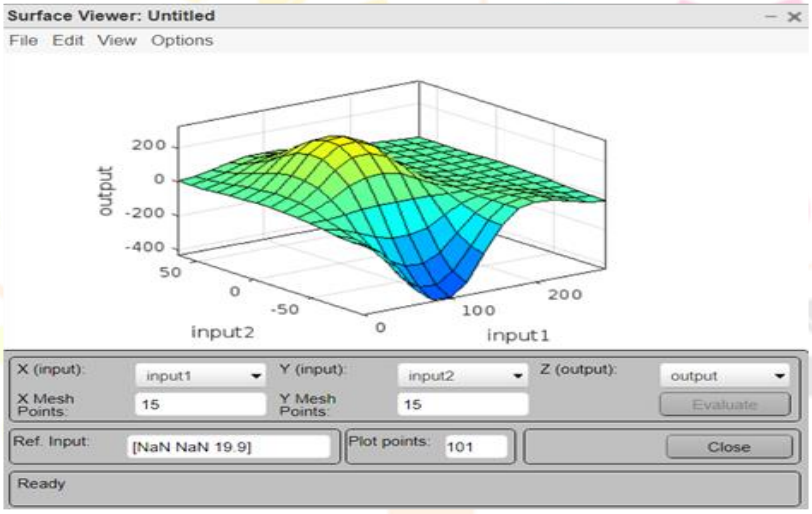


Fig..7: ANFIS Surface View (Voltage vs. Current vs. output)

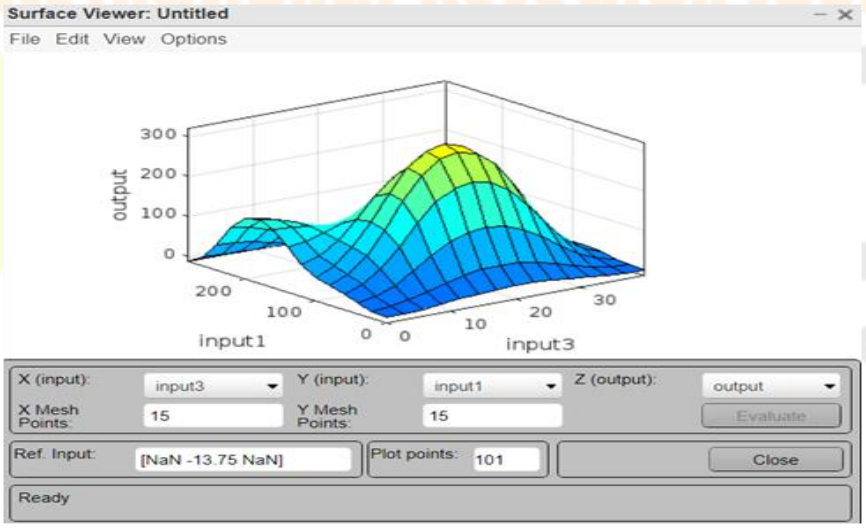


Fig.8: ANFIS Surface View (Voltage vs. Temperature vs. output)

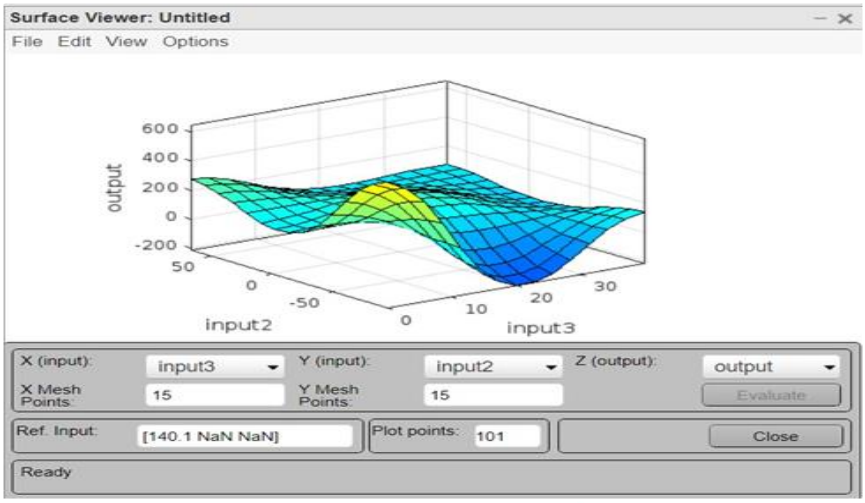


Fig.9: ANFIS Surface View (Current vs. Temperature vs. output)

5.2 Testing ANFIS Model

For the testing of the trained ANFIS Model, a testing dataset was created and plotted against the training data as well as the testing data.

In the first method (Testing Data against Training Data), the model is tested on data that was not used for training, but it is still part of the original dataset. This means that the model has already seen some of the testing data during the training process, and it may have learned to recognize some patterns in it.

In the second method (Testing Data against Testing Data), the model is tested on completely new and unseen data that was not used during the training process. This means that the model is being tested on its ability to generalize to new data and not just memorize the training data.

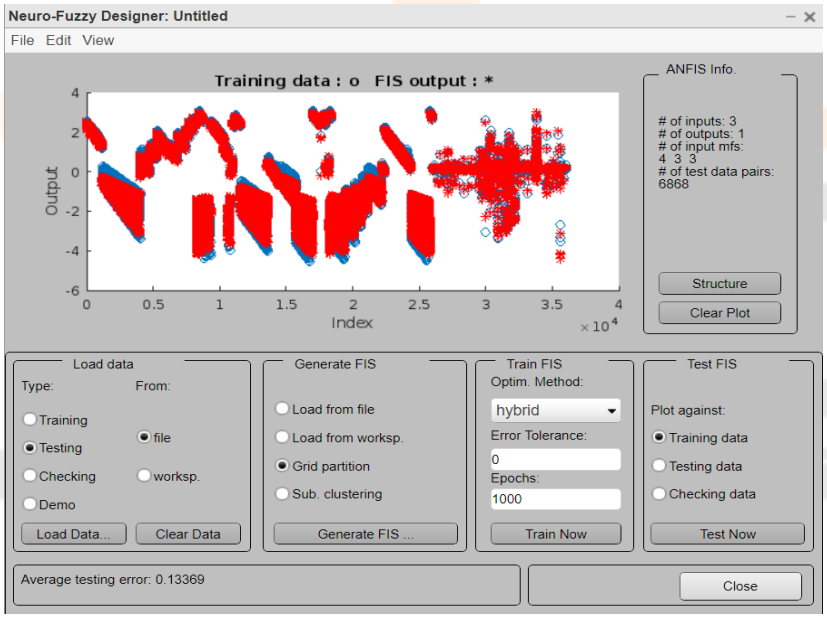


Fig.10: Testing Data against Training Data

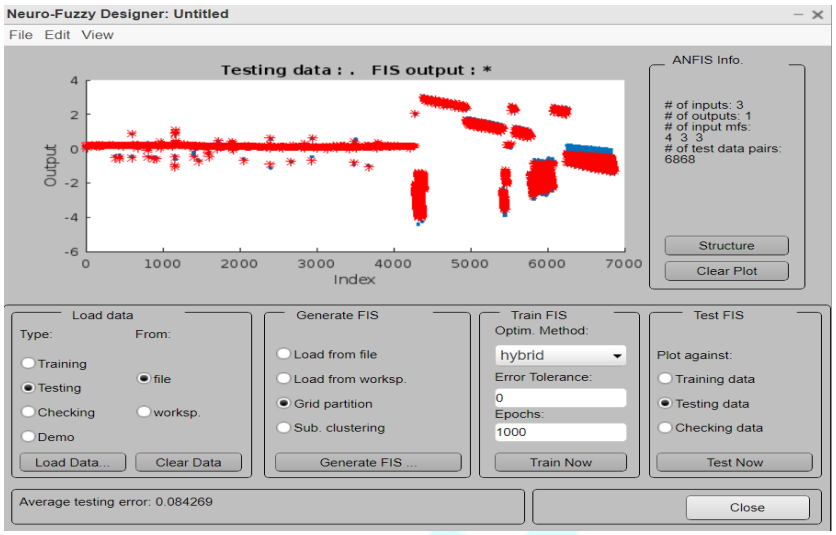


Fig. 11: Testing Data against Testing Data

5.3 Validation of the ANFIS Model

In order to validate the accuracy of the model, a test case consisting of 50 real-time recorded data points was provided as an input to the **evalfis** function, which evaluates the fuzzy inference system for a given input and returns the output, compared the actual output with the expected output and calculate the error metrics.

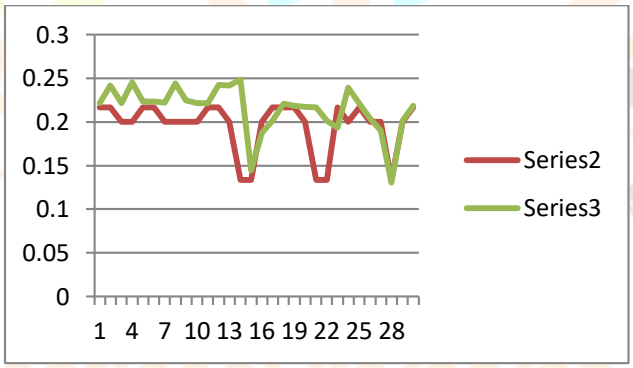


Fig.12: Accuracy of the ANFIS Model (Recorded SoC vs Predicted SoC)

There are various performance metrics that can be used to check the ANFIS model, such as the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and coefficient of determination (R-squared). These metrics can provide insight into the accuracy and reliability of the model.

On calculating the RMSE error as the difference between the predicted output and recorded output it comes out to be

MSE = 0.028791 RMSE = 0.169678

7. CONCLUSION

ANFIS fully utilises fuzzy logic reasoning, which is simple, strong, robust, and accurate, as well as neural networks. The ANFIS model estimated with an average testing error of 0.097398.It can be used in real-time applications to estimate the SOC of a battery in the event of missing or unavailable data. It's considerably easier to set up and delivers a more accurate SOC value. Any battery data, regardless of specifications, can be used to estimate the outcome; this is extremely useful in cases where other approaches produce errors and are application-specific. Different battery conditions can be easily estimated. The outcomes are quite practical. If adequate training data is provided, the artificial neural network can be built for any battery system. The SOC estimator based on the ANFIS approach computes the SOC with a low error.

8. REFERENCES

[1] H. W. He, R. Xiong, and H. Q. Guo, "Online estimation of model parameters and state-of-charge of LiFePO4 batteries in electric vehicles," Applied Energy, vol. 89, no. 1, pp. 413–420, 2012.
[2] Z. H. Cai, G. F. Liu, and J. Luo, "Research state of charge estimation tactics of nickel-hydrogen battery," in Proceedings of the International Symposium on Intelligence Information Processing and Trusted Computing (IPTC '10), pp. 184–187, Huanggang, China, October 2010.
[3] N. Watrin, B. Blunier, and A. Miraoui, "Review of adaptive systems for lithium batteries state-of-charge and state-of-health estimation," in Proceedings of IEEE Transportation Electrification Conference and Expo, pp. 1–6, Dearborn, Mich, USA, June 2012.
[4] V. Prajapati, H. Hess, E. J. William et al., "A literature review of state of-charge estimation techniques applicable to lithium poly-carbon monoflouride (LI/CFx) battery," in Proceedings of the India International Conference on Power Electronics (IICPE '10), pp. 1–8, New Delhi, India, January 2011.
[5] Wen-Yeau Chang, "The State of Charge Estimating Methods for Battery: A Review", International Scholarly Research Notices, vol. 2013, Article ID 953792, 7 pages, 2013.
[6] K. S. Ng, C. S. Moo, Y. P. Chen, and Y. C. Hsieh, "State-of-charge estimation for lead-acid batteries based on dynamic open-circuit voltage," in Proceedings of the 2nd IEEE International Power and Energy Conference (PECon '08), pp. 972–976, Johor Bahru, Malaysia, December 2008.
[7] Tiezhou Wu, Mingyue Wang, Qing Xiao, "The SOC Estimation of Power Li-ion Battery based on ANFIS Model", 2012 Smart Grid andRenewable Energy.
[8] A. J. Salkind, C. Fennie, P. Singh, T. Atwater, and D. E. Reisner, "Determination of state-of-charge and state-of-health of batteries by fuzzy logic methodology," Journal of Power Sources, vol. 80, no. 1-2, pp. 293–300, 1999.
[9] P. Singh, R. Vinjamuri, X. Wang, and D. Reisner, "Design and implementation of a fuzzy logic-based state-of-charge meter for Li-ion batteries used in portable defibrillators," Journal of Power Sources, vol. 162, no. 2, pp. 829–836, 2006.
[10] P. Singh, C. Fennie Jr., and D. Reisner, "Fuzzy logic modelling of state-of-charge and available capacity of nickel/metal hydride batteries," Journal of Power Sources, vol. 136, no. 2, pp. 322–333, 2004
[11] S. Malkhandi, "Fuzzy logic-based learning system and estimation of state-of-charge of lead-acid battery," Engineering Applications of Artificial Intelligence, vol. 19, no. 5, pp. 479–485, 2006.
[12] Iversen, Emil B., Juan M. Morales, and Henrik Madsen. "Optimal charging of an electric vehicle using a Markov decisionprocess."AppliedEnergy123(2014)

International Research Journal
IJNRD
Research Through Innovation