Modeling of electric vehicle batteries using RBF neural networks

Zhang, C., Yang, Z. & Li, K.

Author post-print (accepted) deposited by Coventry University's Repository

Original citation & hyperlink:
https://dx.doi.org/10.1109/ComManTel.2014.6825590

DOI 10.1109/ComManTel.2014.6825590
ISBN 978-1-4799-2903-0

Publisher: IEEE

© 2014 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

Copyright © and Moral Rights are retained by the author(s) and/ or other copyright owners. A copy can be downloaded for personal non-commercial research or study, without prior permission or charge. This item cannot be reproduced or quoted extensively from without first obtaining permission in writing from the copyright holder(s). The content must not be changed in any way or sold commercially in any format or medium without the formal permission of the copyright holders.

This document is the author’s post-print version, incorporating any revisions agreed during the peer-review process. Some differences between the published version and this version may remain and you are advised to consult the published version if you wish to cite from it.
Modeling of electric vehicle batteries using RBF neural networks

Cheng Zhang, Zhile Yang, Kang Li

Abstract—Electric Vehicles (EVs) are promised to significantly reduce the usage of conventional fossil fuels as well as to limit the overwhelming greenhouse gas emissions. An accurate battery model is indispensable for the design of charging and discharging control of EVs. A new Radial Basis Function (RBF) modelling approach, which combines the Levenberg-Marquardt method to tune the non-linear parameters and an input selection approach for refining the input variables is proposed to model the batteries of EVs. Experimental results on modelling Li-ion batteries show that the resultant models have achieved high accuracy and desirable generalization performance.

I. INTRODUCTION

In recent years, Li ion batteries are rapidly gaining popularity in electric vehicle (EV) and hybrid electric vehicle (HEV) applications due to its several advantages, such as high energy density and long service life compared with other types of batteries, such as Lead-Acid, NiMH battery etc. However, particular care should be taken during the operation of Li ion batteries. High voltage and high temperature must be avoided for safety concerns, and over-charge and over-discharge can damage a battery greatly. Therefore, a battery model is essential for proper operation and management of the battery system.

Over the years, researchers have developed many different battery models that are of different levels of accuracy and complexity. Electrochemical models describe the battery electrochemical processes in detail using physical laws, such as concentrated solution theory and porous electrode theory [1, 2]. The models are very accurate and can be considered as white-box modelling method. However, the model complexity is very high and the model also often involves very complex computations of the partial differential equations. Therefore, they are mainly used for battery design purposes.

The equivalent circuit model (ECM) uses equivalent electronic components, such as resistors and capacitors, to model the battery terminal behaviour [3]. The model structure is simple and can be identified using on-line recursive least square method. Therefore, ECM is widely adopted for real-time application. However, the battery behaviour is highly non-linear and non-stationary and depends on the operating conditions, such as temperature and load current rate, which is very difficult for the ECM to capture. Hence, the accuracy of ECM is usually low.

Among all the modelling methods, Radial basis function (RBF) neural network (NN) has been widely adopted due to its simple structure and powerful approximation ability to model non-linear systems. One of the main challenges involved in the construction of RBF neural models is the optimization of non-linear parameters in basis function. The famous Back-Propagation (BP) method for training NN models is actually a version of the gradient-descent method, in which only the first-order derivative information, i.e., the gradient, is used. In this paper, the second-order optimization method, i.e. the Levenberg Marquardy(LM) method, which makes use of the second order derivation information, is adopted. Therefore, the convergence speed is increased. On the other side, the input selection remains a key step and of great challenge in neural network modeling. For actual systems, some training data sets may be irrelevant or having less impact to the output, thus the introduction of them may cause undesirable effects such as over-fitting and high computational cost [4]. Removing these less significant and redundant input terms may significantly increase the accuracy and generalization capacity of the neural network models. Numerous approaches have been proposed to tackle this problem, including model based approaches [5, 6] and model free approaches [7, 8]. Among them, the subset selection methods for 'linear-in-the-parameters' models, such as the orthogonal least squares method (OLS) [9, 10] and forward recursive algorithm (FRA) [11], have also been introduced for NN input selection [12, 13].

The rest of the paper is organized as follows. A brief review of RBF neural networks is discussed in Section II. A fast forward input selection method is then presented in Section III, followed by the integrated parameter training method illustrated in Section IV. After the construction of the proposed algorithm in Section V, experimental results are presented in Section VI. Finally, Section VII concludes the paper and point out the future work.

II. RADIAL BASIS FUNCTION NEURAL NETWORKS

Consider a general multiple-inputs-single-output (MISO) RBF network denoted as

\[ y(t) = \sum_{i=1}^{n} w_i \cdot \psi_i(X(t); \sigma_i) + \varepsilon(t) \]

where \( y(t) \) is the system output at sample time \( t \), and \( w_i \) denotes the linear output weight for the \( i \)-th node in the hidden layer. \( X(t) \in \mathbb{R}^{P} \) is the input vector, and \( \sigma_i^2 \) and \( \varepsilon(t) \) denote the width and the center vectors of the \( i \)th hidden node respectively, while \( \varepsilon(t) \) denotes the network error at sample
time $t$. Finally, the non-linear activation Gauss function $\psi_i$ of input vector $X(t)$ is defined as follows:

$$
\psi_i(X) = \exp\left(-\frac{1}{\sigma_i^2} \|X - c_i\|^2\right), i = 1, 2, ..., n \quad (2)
$$

If the model is trained by a set of $N$ data samples $\{X(t), y(t)\}_{t=1}^N$, then the NN model output can be represented as

$$
\hat{Y} = \Psi \ast W \quad (3)
$$

with

$$
\hat{Y} = [\hat{y}_1, \hat{y}_2, ..., \hat{y}_N]^T
$$

$$
\Psi = [\psi_1, \psi_2, ..., \psi_N]^T
$$

$$
\psi_i = [\psi_{i,1}, \psi_{i,2}, ..., \psi_{i,n}]^T
$$

$$
W = [w_1, w_2, ..., w_n]^T
$$

where $N$ is the number of data samples used for the training of the RBF neural model and $N_n$ is the number of hidden nodes in the RBF model. Let

$$
\theta_{nl} = [c_1^T, \sigma_1^2, ..., c_n^T, \sigma_n^2]^T
$$

denote the non-linear parameters.

Therefore, the modelling error is

$$
err = Y - \hat{Y} = Y - \Psi \ast W \quad (4)
$$

where $Y = [Y_1, Y_2, ..., Y_N]^T$ is the actual system output.

During the training procedure for the RBF neural model, an objective function of the mean square error as in (5) is minimized.

$$
MSE = \frac{1}{N} \text{err}^T \ast \text{err} \quad (5)
$$

When the RBF neural nodes centres and width $\theta_{nl}$ are fixed, the linear RBF network output weights can be optimized by least square method as follows,

$$
\hat{W} = (\Psi^T \Psi)^{-1} \Psi^T \ast Y \quad (6)
$$

Then the modelling error becomes

$$
\text{err} = \Psi (\Psi^T \Psi)^{-1} \Psi^T \ast Y - Y \quad (7)
$$

Note that $\Psi$ consists of the non-linear parameters $c_i, \sigma_i^2$, then

$$
MSE = \frac{1}{N} \text{err}^T \ast \text{err} = \frac{1}{N} Y^T (I - \Psi (\Psi^T \Psi)^{-1} \Psi^T) Y \quad (8)
$$

III. FAST FORWARD INPUT SELECTION

A fast forward input selection method proposed in [13] is introduced in this research to aid the construction of the RBF neural network. Consider a discrete non-linear dynamic system with $m$ inputs and single output denoted as

$$
y(t) = f(y(t - 1), ..., y(t - l_y), u(t - 1), ..., u(t - l_u)) \quad (9)
$$

where $y(t)$ is the system output at sample time $t$, $u(t) = [u_1(t), ..., u_m(t)]$ are the system input vector, $l_y$ and $l_u$ are the maximal orders for the system output and input delay, $f(\cdot)$ denotes some non-linear relationships between the system inputs and output.

Referring to the RBF expression in (1), the model inputs $X$ can be selected straightforwardly from the past system inputs and outputs, i.e., $u_i(t-1), ..., u_i(t-l_u), i = 1, ..., m$ and $y(t-1), ..., y(t-l_y)$. However, if all of these terms are used as the model inputs, the number of the model parameter will be too large and the computational cost extremely high. Furthermore, the delay orders, i.e., $l_u, l_y$, of system are usually unknown. If (9) is approximated by a polynomial 'linear-in-the-parameter' model, the RBF neural model input selection problem could be replaced and solved analogously by model term selection method. Thus the candidate model term pool $\varphi_i$ is formulated by some non-linear functions combining system input vector and output vector at some past time instants, denoted as

$$
\varphi_i(t) = \prod_{j=n_p}^{n_p} y(t - j) \prod_{k=1}^{m} \prod_{l=n_k}^{n_k} u_k(t - d_k - j), i = 1, ..., q \quad (10)
$$

where $0 \leq n_{p1} \leq ... \leq n_{pj} \leq l_y, 0 \leq n_{k1} \leq ... \leq n_{kq} \leq l_u$. To select the variable terms from the pool, a forward selection is introduced by [12, 13]. Define a matrix series as

$$
R_k = I - \Phi_k \Phi_k^T \Phi_k^{-1}, k = 1, ..., p \quad (11)
$$

where $\Phi_k = [\varphi_1, ..., \varphi_k], k = 1, ..., p$ donates the $k$ selected terms, and $\varphi_i = [\varphi_i(1), ..., \varphi_i(N)]^T$. Then the recursive form of $R_{k+1}$ is denoted as

$$
R_{k+1} = R_k - \frac{R_k \varphi_{k+1 \varphi_{k+1}^T R_k}}{\varphi_{k+1 \varphi_{k+1}^T}}, k = 0, 1, ..., p - 1 \quad (12)
$$

The aforementioned matrix series $R_k$ also has the following properties [11]:

1) $R_k^T = R_k, (R_k)^2 = R_k$ \quad (13)

2) $R_k R_j = R_j R_k = R_k$ for all $k \geq j$ \quad (14)

3) $R_k \varphi_i = 0, \ \forall i \in \{1, ..., k\}$ \quad (15)

Suppose $E_k$ is square modelling error after selecting $k$ terms, i.e., $\Phi_k$, and follow the same procedure from (3) to (8), then $E_k$ can be denoted as

$$
E_k = y^T R_k y \quad (16)
$$
On the basis of (16), the cost function reduction after adding $k + 1$ th term, i.e., the contribution of $\phi_k + 1$ can be expressed as

$$
\Delta E_{k+1} = y^T (R_k - R_{k+1}) y = \frac{y R_k \phi_k + 1 \phi_k + 1^T R_k y}{\phi_k + 1 R_k \phi_k + 1}.
$$

(17)

Further simplification may improve the computational speed [12, 13], and it is not included in this paper due to the page limit.

The candidate terms are selected consecutively according to the cost contribution for which the terms that make the maximum contributions would be selected. The selection procedure stops based on the cost function criterion, and the sum squared error (SSE) is introduced as the stop criterion in this study.

IV. LEVENBERG-MARQUARDT METHOD FOR PARAMETER OPTIMIZATION OF RBF NEURAL MODELS

Note that the modelling error in (4) depends linearly on the neural networks weight, $W$, and non-linearly on the node centres and width parameters, i.e., $\sigma_i^2$ and $c_i$. Correspondingly, the parameters to be optimized can be divided into two groups: linear parameters and non-linear parameters. Usually a two-step training method is adopted. Firstly, the linear parameters are optimized by least-square (LS) method, while the non-linear parameters are fixed. Secondly, fix the linear parameters and optimize the non-linear parameters using a non-linear optimization method, such as the deepest-descend method and the Gauss-Newton method. This two-step procedure is repeated until a desired training accuracy is achieved.

In this paper, an integrated training method proposed by Li et al [14, 15] is adopted to optimize the linear and non-linear parameters together simultaneously.

The gradient of $MSE$ in (8) with respect to $\theta_{nl}$ is

$$
G = \frac{\partial SSE}{\partial \theta_{nl}} = \frac{2}{N} err^T \frac{\partial err}{\partial \theta_{nl}}.
$$

(18)

Note that

$$
\frac{\partial err}{\partial \theta_{nl}} = \frac{\partial \{\Psi (\Psi T \Psi)^{-1} \Psi T\}}{\partial \theta_{nl}} Y
$$

$$
= \frac{\{\frac{\partial \Psi}{\partial \theta_{nl}} (\Psi T \Psi)^{-1} \Psi T + \Psi \frac{\partial (\Psi T \Psi)}{\partial \theta_{nl}}\}}{\frac{\partial \Psi}{\partial \theta_{nl}} + \Psi T (\Psi T \Psi)^{-1} \frac{\partial \Psi}{\partial \theta_{nl}}} Y
$$

(19)

Substitute

$$
\frac{\partial (\Psi T \Psi)^{-1}}{\partial \theta_{nl}} = -(\Psi T \Psi)^{-1} * (\frac{\partial \Psi T}{\partial \theta_{nl}} \Psi + \Psi T \frac{\partial \Psi}{\partial \theta_{nl}}) (\Psi T \Psi)^{-1}
$$

into (19), yields

$$
\frac{\partial err}{\partial \theta_{nl}} = (I - \Psi (\Psi T \Psi)^{-1} \Psi T) \frac{\partial \Psi}{\partial \theta_{nl}} \hat{W}
$$

$$
- \Psi (\Psi T \Psi)^{-1} \frac{\partial \Psi T}{\partial \theta_{nl}} err
$$

(20)

As

$$
err^T \Psi = 0
$$

therefore, the gradient is

$$
G = \frac{2}{N} err^T \frac{\partial \Psi}{\partial \theta_{nl}} \hat{W}
$$

(21)

The next step is to calculate the Hessian of $MSE$ with respect to $\theta_{nl}$.

$$
H_{es} = \frac{\partial^2 MSE}{\partial \theta_{nl}^2} = \frac{2}{N} \left( \frac{\partial err^T}{\partial \theta_{nl}} \frac{\partial err}{\partial \theta_{nl}} + err^T \frac{\partial^2 err}{\partial \theta_{nl}^2} \right)
$$

(22)

The modelling error $err$ is assumed to be very small, then

$$
H_{es} \approx \frac{\partial err^T}{\partial \theta_{nl}} \frac{\partial err}{\partial \theta_{nl}} = \frac{2}{N} \{ err^T \frac{\partial \Psi}{\partial \theta_{nl}} (\Psi T \Psi)^{-1} \frac{\partial \Psi T}{\partial \theta_{nl}}
$$

$$
* err + \hat{W} \frac{\partial \Psi T}{\partial \theta_{nl}} (I - \Psi (\Psi T \Psi)^{-1} \Psi T) \frac{\partial \Psi}{\partial \theta_{nl}} \hat{W} \}
$$

(23)

Finally, apply Levenberg Marquardt (LM) method as

$$
\theta_{nl}(s + 1) = \theta_{nl}(s) - (H_{es} + \lambda I)^{-1} G
$$

where $s$ is the iteration step of the optimization procedure, $\lambda$ can be found by golden search method.

It should be noted that the calculation of the derivative $\frac{\partial \Psi}{\partial \theta_{nl}}$ is a key to this optimization procedure. From (2),

$$
\frac{\partial \Psi}{\partial \theta_{nl}} = [\frac{\partial \psi_1}{\partial \theta_{nl}}, \ldots, \frac{\partial \psi_{N_n}}{\partial \theta_{nl}}, 0]
$$

and $\psi_i$ depends only on $c_i, \sigma_i^2$, and

$$
\frac{\partial \psi_{i,j}}{\partial \sigma_i^2} = \psi_{i,j} * \frac{2(u_i - c_i)}{\sigma_i^2}
$$

$$
\frac{\partial \psi_{i,j}}{\partial c_i} = \psi_{i,j} * \frac{\Vert u_i - c_i \Vert^2}{(\sigma_i^2)^2}
$$

V. CONSTRUCTION ALGORITHM

The objective function for the LM optimization process is the mean squared error (MSE) expressed in (5). The implementation of the modeling procedure thus has four major steps illustrated as below.

A. Input selection

According to Section III, a candidate term pool should be formulated first. Some promising system inputs in a past time instant as well as a combination of the input vector based on (10) may be selected. Generally speaking, all the input vectors with past time instants are initially considered and some output vectors at several past time intervals should also be considered. Then, compute the cost contribution of each term in the candidate pool based on (17), select the most significant contributor, and remove it from the candidate term pool. Such process would continue and terms of large impact could be selected iteratively until the SSE criterion achieved. Once the most significant terms are selected, then the corresponding system inputs and system outputs including the associated time lag in these selected terms are used as the inputs for the RBF NN model to be constructed and trained.
B. Determination of Network Structure

Along with the input selection, the RBF network structure including the linear and non-linear parameters of hidden nodes should also be optimized. It should be noted that in this study the network structure is determined on the basis of trial and error method, and further elaborate model structure could be considered in the future work. The values for the non-linear parameters including the center and width of the Gauss function could be generated randomly at the beginning.

C. Network Parameters Optimization Using LM

The dependence of the linear parameters on the non-linear parameters, i.e., as shown in (6) is taken into consideration during the optimization of the non-linear parameters, and therefore the linear parameters vanish in the objective function. Then only the non-linear parameters are left to be optimized, thus reducing the parameter space to search for the optimization solutions. The convergence speed can thus be improved.

The detailed process is demonstrated as follows.

1) Input selection:
   a) formulate a candidate input term pool;
   b) compute the input contribution and select the maximum contributor;
   c) repeat the selection step until the SSE criteria in (16) is achieved.

2) Network Determination:
   a) pre-set the hidden nodes numbers;
   b) randomly assign values for the non-linear parameters.

3) Parameter optimization for the RBF neural model using the proposed algorithm.

VI. EXPERIMENTAL STUDY

A 5Ah LFP battery was tested under constant temperature (25°C) using Arbin BT2000 battery test system. The Federal Urban Drive Schedule (FUDS) test procedure was run on the battery. The data includes 2000 measured current and voltage samples, as shown in Fig. 1. The first 1000 data samples were used for RBF model training and the rest for model validation. A RBF neural model structure with 4 inputs and 12 hidden nodes were selected by trial and error method. The modelling performance of the NN model with FRA input selection method is compared with the NN model with randomly selected inputs. The same optimization procedure was run for both two models for fair comparison, and the iteration number are set to 20 steps.

A. Input selection

A natural selection of the four inputs are \( v_{t-1}, v_{t-2}, I_t, I_{t-1} \), where \( v_{t-1} \) indicates the voltage measurement at the past time instant. Using the FRA method, four inputs were selected, including \( v_{t-1}, I_{t}, I_{t-1}, v_{t-2}^{2} \).

B. Validation Errors

The validation errors of the two methods are shown in Fig. 4 and Fig. 5. The mean square errors are \( 2.37E-5 \) and \( 2.46E-5 \) without input selection. The FRA input selection method achieved an improvement of about 5%. The improvement may not seem significant, and it can be explained by the super modelling performance achieved by neural networks. This also indicates the success of the parameter identification method.
5 for NN with and without FRA input selection. The FRA method achieves about 17% improvement.

Fig. 4. validation error of FRA+NN methods

Fig. 5. validation error of NN methods without input selection

It can be seen from these figures that the training error and validation error for both methods are very satisfactory after the 20 steps of parameter optimization, which indicates the good performance of the proposed integrated parameter optimization method.

C. Discussions

It can be observed that the error spikes occur when the load current changes suddenly. This can be explained by the highly non-linear property of battery behaviour, which is difficult to capture, even for the neural networks modelling method.

Another problem arose with the parameter training is that the parameter optimization iteration number was set to 20. If the model is further trained for more iterations, the training error will keep decreasing. However, the validation error will increase rapidly. This indicates the well-known over-fitting problem.

VII. CONCLUSION AND FUTURE WORK

In this paper, a novel RBF neural network modelling method is proposed and implemented to model the electric vehicle battery. To simplify the model structure and improve the modelling speed, a fast forward input selection approach is implemented that the significant system input vectors with time lags are consequently selected from a candidate term pool. Moreover, to further optimize the non-linear parameters in the model, an integrated parameter optimization method is introduced. The model is applied to a Li-ion battery to predict the voltage output, and trained by the data set from a standard test procedure. The result illustrates that the model has a high training accuracy and an excellent generalization capability.

Future work may concentrate on the further improvements of the model selection method and the parameter refinement procedure. Numerous approaches on the selection of hidden nodes could be introduced in addition to the input selection [4]. On the other side, the Levenberg-Marquardt method is a gradient based optimization which is easy to be trapped within local optimum. Heuristic approach such as Particle Swarm Optimization [16], Differential Evolution, [17], Biogeography-Based optimization [18] and Teaching-learning based optimization (TLBO) [19] can be used to refine the non-linear parameters in the model.

ACKNOWLEDGMENT

The authors would like to thank Prof. Chunbo Zhu and Lei Pei from Harbin Institute of Technology for providing test data. Cheng Zhang would like to thank Chinese Scholarship Council and UK-China Science Bridge project for sponsoring his research.

REFERENCES


