Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 3793492, 13 pages
https://doi.org/10.1155/2018/3793492

Research Article

Development of Accurate Lithium-Ion Battery Model Based on Adaptive Random Disturbance PSO Algorithm

Huang Kai,[1] Guo Yong-Fang,[2] Li Zhi-Gang,[1] Lin Hsiung-Cheng,[3] and Li Ling-Ling[1]

[1] *State Key Laboratory of Reliability and Intelligence of Electrical Equipment, Hebei University of Technology, Tianjin 300130, China*
[2] *School of Computer Science & Engineering, Hebei University of Technology, Tianjin 300130, China*
[3] *Department of Electronic Engineering, National Chin-Yi University of Technology, Taichung 41170, Taiwan*

Correspondence should be addressed to Lin Hsiung-Cheng; hclin@ncut.edu.tw

Received 28 November 2017; Revised 8 May 2018; Accepted 5 June 2018; Published 3 July 2018

Academic Editor: Eric Monfroy

The performance behavior of the lithium-ion battery can be simulated by the battery model and thus applied to a variety of practical situations. Although the particle swarm optimization (PSO) algorithm has been used for the battery model development, it is usually unable to find an optimal solution during the iteration process. To resolve this problem, an adaptive random disturbance PSO algorithm is proposed. The optimal solution can be updated continuously by obtaining a new random location around the particle's historical optimal location. There are two conditions considered to perform the model process. Initially, the test operating condition is used to validate the model effectiveness. Secondly, the verification operating condition is used to validate the model generality. The performance results show that the proposed model can achieve higher precision in the lithium-ion battery behavior, and it is feasible for wide applications in industry.

1. Introduction

A battery management system (BMS) is an electronic system used to manage a rechargeable battery or battery pack, and it is widely applied to many applications that use a battery or batteries, such as portable electronic devices, electric vehicles, and power grids [1]. To ensure safe and efficient operation, it is essential for a BMS to be able to predict the static and dynamic behavior of the batteries.

Currently, there are three general types of battery models available in the literature: the electrochemical battery model, artificial neural network model, and equivalent circuit model (ECM) [2–9]. Among them, ECMs have been extensively researched in recent years owing to their excellent adaptability and simple realization [10, 11].

Parameter identification is an essential step in battery modeling, and its results directly affect the accuracy and reliability of the model. Identification methods are usually divided into two types [12, 13]: online and offline, corresponding to online and offline modeling, respectively.

Online identification methods adjust the parameters of the model in real time based on the condition and current state of the battery. Furthermore, the BMS makes use of such parameters and other information such as current, voltage, and temperature to evaluate the state of charge (SOC), state of health (SOH), and so on [14, 15], which are required for real time control [16, 17]. However, the processor should require higher calculation speed. Offline identifications can adopt mass experimental data that reflects the characteristics of the batteries. As a result, the identified parameters have higher precision and adaptability, and this makes such offline identifications more suitable for battery or battery pack modeling.

At present, there are two major types of offline identifications. The first type of methods is generally called traditional identification methods, like fitting method based on Least Squares [8], subspace identification [16], multiple linear regression method [18], and so on. This type of methods is simple and intuitive; however, the identified parameters have larger errors, and hence it is usually used in applications with lower accuracy demand [19, 20]. The second type of methods is generally called bionic intelligent optimization algorithms, like PSO [21], genetic algorithm (GA), [22] and so on. Compared with the first type of methods, the second one has obvious advantages in accuracy and reliability, and it has become a popular method for parameter identification. However, when GA is used in parameter identification [22–24], there are certain issues that cannot be avoided, such as the high computation time and easily falling into local optimum, i.e., local extremum. The particle swarm optimization (PSO) algorithm may have low precision and often fall into local optimum [24–29]. According to [25–27], the convergence speed and the inertia weight ω control can be improved. In [28], both GA and PSO were combined, and in [29], the double quantum PSO was adopted to identify the system parameters, which can increase the traverse ability of particles and simplify the evolution equation without using a velocity vector. However, the abovementioned method may not solve the problem in easily falling into the local optimum. In [30–32], the chaotic optimization algorithm (COA) and the improved PSO algorithm were combined to identify the parameters of the battery, load, and solar cell models. In abovementioned papers [30–32], upon accepting that the PSO algorithm has an issue of falling into local optimum, the COA is adopted to find the new searching swarm and continue the search, in order to increase global convergence and calculation precision. In [30], the difference between the covariance matrix of the particle location and the predetermined threshold has been used as the basis, while in [32] the difference between the variance of the population's fitness and the predetermined threshold is used as the basis to judge whether the solution falls into local optimum. In addition, the predetermined threshold, which has certain influence on calculation precision, is set according to the experience.

For the development of the battery model using PSO algorithm, the particles may hover around local optimal solution during the iteration process without reaching the real optimal location. For this reason, an adaptive random disturbance PSO (ARDPSO) is proposed and its performances are validated by using a classical single model and a multiplex model as target optimization functions. Also, this algorithm can be used for identifying the parameters of the battery model thus achieving higher calculation precision.

The remaining contents of this paper are arranged as follows: Section 2 introduces the standard PSO algorithm. Section 3 describes ARDPSO algorithm and validates its performance. ECM parameters identification using ARDPSO is illustrated in Section 4. The experiment results and discussion are presented in Section 5. Section 6 concludes this paper.

2. Standard PSO Algorithm

Particle swarm optimization (PSO) is an evolutionary computation technique, especially searching for optimization for continuous nonlinear, constrained and unconstrained, and nondifferentiable multimodal functions [21]. It can optimize a problem by iteratively moving particles around in the search space over the particle's position and velocity. Each particle's movement is updated as better positions in the search space [33]. Assume m particles appear in a D-dimensional search space, and the sets can be expressed as

$$\begin{aligned} X &= \{x_1, x_2, \dots, x_m\} \\ x_i &= \{x_{i1}, x_{i2}, \dots, x_{iD}\}; \quad i = 1, 2, \dots, m \\ v_i &= \{v_{i1}, v_{i2}, \dots, v_{iD}\}; \quad i = 1, 2, \dots, m \\ Pbest_i &= \{Pbest_{i1}, Pbest_{i2}, \dots, Pbest_{iD}\} \\ Gbest_i &= \{Gbest_1, Gbest_2, \dots, Gbest_D\} \end{aligned} \tag{1}$$

where X denoted by x_i represents the group containing m particles with D-dimensional vector in the search space. The term v represents the velocity of the particles. The term $Pbest_i$ represents the local optimal solution, that is, the individual best solution of particle i. The term $Gbest$ represents the global optimal solution.

In each iteration, every particle updates its position and velocity in search space according to its individual best solution $Pbest_i$ and the global optimal solution of the group $Gbest$, as follows:

$$v_i^{n+1} = v_i^n + c_1 r_1 (Pbest_i^n - x_i^n) + c_2 r_2 (Gbest^n - x_i^n) \tag{2}$$

$$x_i^{n+1} = x_i^n + v_i^{n+1} \tag{3}$$

where n is the current number of iterations; c_1 and c_2 are two positive constants called acceleration factors. r_1 and r_2 are random numbers in the range 0–1.

For a particle swarm optimization, a better global search is needed from a starting phase to help the algorithm converge to a target area quickly, and then a stronger local search is used to get a high precision value. Therefore, the modification of improved standard PSO introduces the inertia weight ω in formula (2), and its velocity is represented as follows:

$$v_i^{n+1} = \omega \cdot v_i^n + c_1 r_1 (Pbest_i^n - x_i^n) + c_2 r_2 (Gbest^n - x_i^n) \tag{4}$$

For getting a high precision solution, ω needs to be kept as a variable value, generally a decreasing value. In the improvements of the standard PSO, the linear PSO (LPSO) [25] is very representative, which uses a linearly decreasing inertia weight, given by

$$\omega = (\omega_{\max} - \omega_{\min}) \left(\frac{N - n}{N}\right) + \omega_{\min} \tag{5}$$

in which ω is current inertia weight, $\omega_{\min}$ is the minimum value (that is, the final value) of inertia weight, $\omega_{\max}$ is the maximum value (that is, the initial value) of inertia weight, n is the current iteration number, and N is the maximum number of allowable iterations.

3. ARDPSO Algorithm

3.1. The Principle of ARDPSO. Although some related PSO algorithms in convergence speed and inertia weight control have been reported in the literature, the particles may still encounter problems such as hovering around the real optimal location but being unable to locate it. It means that the local optimal solution of the particle is not updated, then resulting in the global optimal solution not to be updated. Consequently, the distance between the searched solution

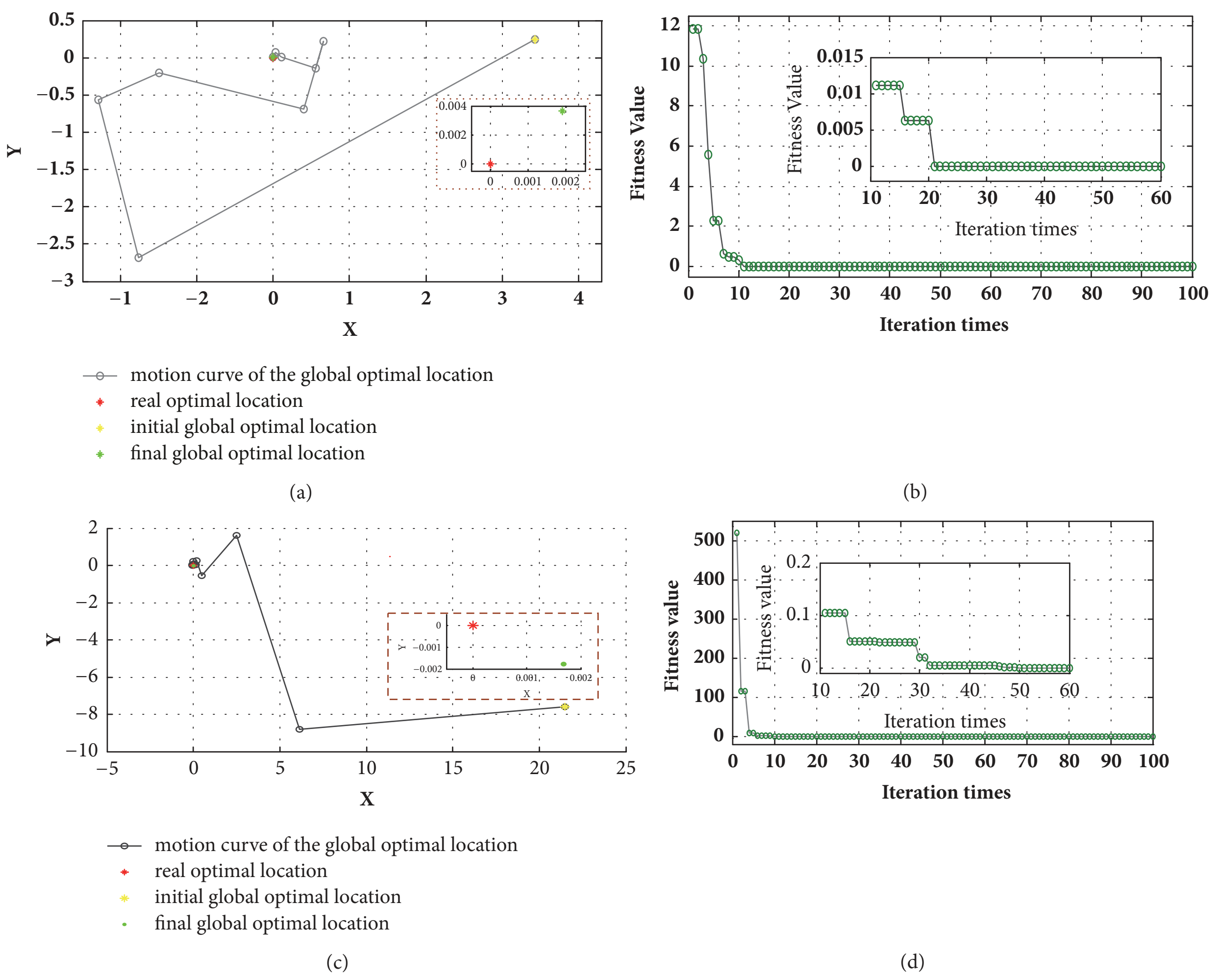


Figure 1: Optimization procedure: (a) the motion curve of the global optimal solution of the standard PSO; (b) the fitness curve of the global solution of the standard PSO; (c) the motion curve of the global solution of the LPSO; (d) the fitness curve of the standard LPSO.

and the real optimal solution will not become closer. For illustrating such a problem, a typical experiment uses the standard PSO and LPSO algorithms to find a solution that minimizes the single mode target function $f(X, Y) = X^2 + Y^2$ and obviously the real optimal solution is $(0, 0)$. Without loss of generality, the number of particles is set as 50, and the maximum number of iterations is set as 100 in the tests. Note that the performances with different initial settings such as the number of the particles, iterative times, the initial range of the particles, and velocity may produce different results.

Figures 1(a) and 1(c) show the motion curves of the global optimal solution of the standard PSO and LPSO, respectively. On the other hand, Figures 1(b) and 1(d) show the corresponding fitness for standard PSO and LPSO, respectively.

From Figure 1, the following can be seen.

(1) The global optimal solution (*Gbest*) may stop updating during a certain period of iterations. For example, as shown in Figure 1(d), from the 10th to 15th iteration, the global optimal solution stops updating, while the global optimal solution continues to update after the 16th iteration.

(2) During the iteration process, the global optimal solution of the standard PSO and LPSO gradually tends to the real optimal position $(0, 0)$. However, it is noted that the standard PSO algorithm stops updating the global optimal solution at the 24th iteration, and its fitness is 1.702324344665708e-05. On the other hand, the LPSO algorithm stops updating the global optimal solution at the 78th iteration, and its fitness is 1.6916e-12.

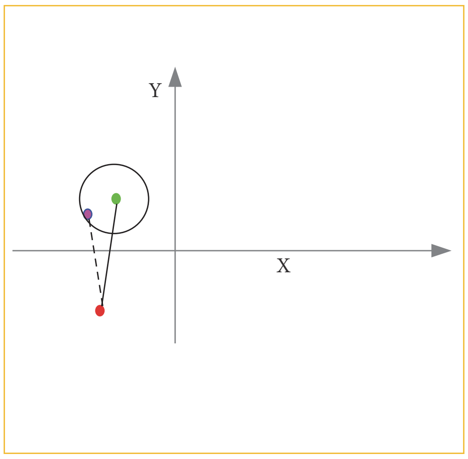
(3) The inertia weight ω shown in formula (4) in LPSO is the key factor to achieve a better fitness value than standard PSO by increasing the number of iterations to be closer to a real optimal solution.

However, both standard PSO and LPSO would encounter such problems:

(1) During the iterations, the global optimal solution may not be updated for each iteration.

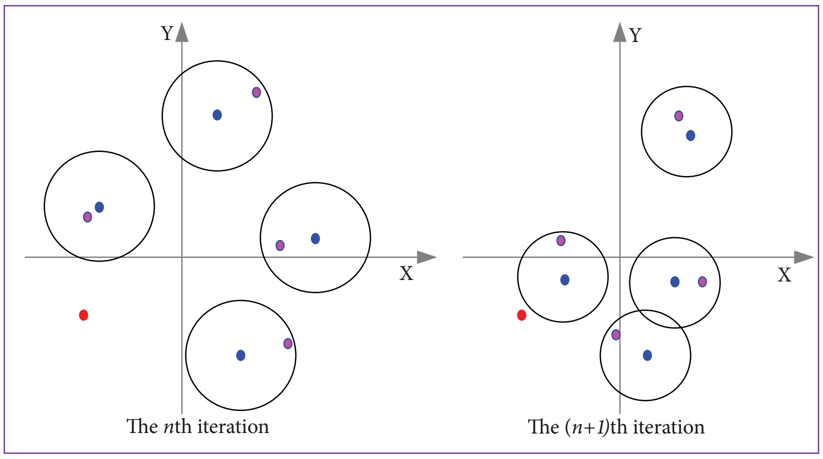
(2) The global optimal solution may stop updating before the maximum number of allowable iterations is achieved.

For this reason, the ARDPSO algorithm is proposed to give the particle more opportunities to continue to update



● Real optimal location
● Global optimal location
● New location
○ Coverage of random disturbance

(a)



● Real optimal location
● Historical local optimal location of the particle
● Location got randomly
○ Coverage of random disturbance

(b)

FIGURE 2: The principle of the ARDPSO: (a) the global optimal solution updating; (b) the convergent procedure of ARDPSO.

its local optimal solution and thus to find the global optimal more accurately. The global optimal solution updating from the ARDPSO algorithm is shown in Figure 2(a). During the iteration process, the moving of the particle will update the global optimal location if a new location has a smaller distance from the real optimal location than the global optimal location.

Figure 2(b) illustrates the procedure of the ARDPSO algorithm performance as follows.

(1) In each iteration process, for each particle, if the fitness of its current solution (that is, x_i^{n+1}) is inferior to the fitness of its local optimal solution (that is, $Pbest_i^n$), $Pbest_i^n$ will not be updated in this iteration. It is noted that the current solution x_i^{n+1} is got from formula (3), i.e., $x_i^{n+1} = x_i^n + v_i^{n+1}$; if, in the $(n)^{th}$ iteration, $Pbest_i^n$ is updated, then, $Pbest_i^n = v_i^n$. In the $(n+1)^{th}$ iteration, if $fitness(x_i^{n+1})$ is inferior to $fitness(Pbest_i^n)$, it means $fitness(x_i^{n+1})$ is inferior to $fitness(x_i^n)$; the most likely reason for this is that (a) x_i^n is the real optimal solution and (b) the moving speed is too rapid so that x_i^{n+1} exceeds the real optimal solution. If it is due to the first reason, then the algorithm has already achieved the optimal solution. In the second reason, a relatively slow moving speed needs to be set.

If $Pbest_i^n$ in the $(n)^{th}$ iteration is not updated and if $fitness(x_i^{n+1})$ in the $(n+1)^{th}$ iteration is inferior to $fitness(Pbest_i^n)$, it means that the current location of the particle x_i^{n+1} comparing with $Pbest_i^n$ has a relatively long distance from the real optimal solution. In this situation, a new solution should be got around $Pbest_i^n$.

During the evolution of the global optimal solution, the distance between the global optimal solution and the real optimal solution decreases with increasing iteration time, and the coverage of the random disturbance thus becomes less. In Figure 2(b), it can be seen that the coverage of the $(n+1)^{th}$ random disturbance is less than that of the $(n)^{th}$ random disturbance.

A new random location generation function can be defined as formula

$$x_i^{n+1}\_\mathrm{rand} = Pbest_i^n + v_i^{n+1} * \exp\left(-a * \frac{(n+1)}{N}\right) * \mathrm{rand}\,() \qquad (6)$$

If the fitness of its current solution (that is, x_i^{n+1}) is inferior to the fitness of its local optimal solution, then it will use formula (6) to get a new location. In formula (6), $\exp(-a * n/N)$ is used to scale the coverage of the random disturbance, commonly, a is a constant, n presents the current iteration number, and N presents the total iteration numbers. The function rand() *is* a random value between 0 and 1.

(2) By comparing the fitness of the randomly obtained new solution to that of the abovementioned current solution, the better solution is selected as the new current location of the particle.

(3) If the new current location of the particle is superior to its local optimal solution, thus the local optimal solution is updated.

As above, the process of the ARDPSO is concluded as follows.

At $(n+1)^{th}$ iteration, for particle i, we have the following.

Step 1. Get the moving speed: $v_i^{n+1} = \omega \cdot v_i^n + c_1 r_1 (Pbest_i^n - x_i^n) + c_2 r_2 (Gbest^n - x_i^n)$.

Step 2. Get the new location: $x_i^{n+1} = x_i^n + v_i^{n+1}$.

Table 1: Benchmarks.

No.	function	formula	minimum
f_1	Sphere Model	$f_1(x) = \sum_{i=1}^{n} x_i^2 \quad -100 \le x_i \le 100$	0
f_2	Schwefel's Problem 2.22	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert \quad -10 \le x_i \le 10$	0
f_3	Schwefel's Problem 1.2	$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j \right)^2 \quad -100 \le x_i \le 100$	0
f_4	Schwefel's Problem 2.21	$f_4(x) = \max_i \{\lvert x_i \rvert, 1 \le i \le n\} \quad -100 \le x_i \le 100$	0
f_5	Rosenbrock Function	$f_5(x) = \sum_{i=1}^{n-1} \left[100 (x_{i+1} - x_i)^2 + (x_i - 1)^2 \right] \quad -30 \le x_i \le 30$	0
f_6	Step Function	$f_6(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2 \quad -100 \le x_i \le 100$	0
f_7	Rastrigin Function	$f_7(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10] \quad -5.12 \le x_i \le 5.12$	0
f_8	Ackley Function	$f_8(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos 2\pi x_i \right) + 20 + e \quad -32 \le x_i \le 32$	0
f_9	Griewank Function	$f_9(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1 \quad -600 \le x_i \le 600$	0

Step 3. If the fitness of the current location x_i^{n+1} is inferior to the fitness of its local optimal solution, then a new location x_i^{n+1}_rand $= Pbest_i^n + v_i^{n+1} * \exp(-a * (n+1)/N) * \text{rand}()$ is obtained. Compare the fitness of x_i^{n+1} and x_i^{n+1}_rand, and then choose the best one as the current location x_i^{n+1}.

Step 4. Get $Pbest_i^{n+1}$. If the fitness of x_i^{n+1} got in Step 3 is superior to that of $Pbest_i^n$, then $Pbest_i^{n+1} = x_i^{n+1}$. Otherwise, $Pbest_i^{n+1} = Pbest_i^n$.

And then, for all of the particles, we have the following.

Step 5. Get $Gbest^{n+1}$.

3.2. Testing and Analyzing the ARDPSO. In order to verify the availability and wide applicability of the proposed ARDPSO algorithm, nine benchmarks from the BBO repository are adopted to test its performance, shown in Table 1, in which $f_1 - f_5$ are single mode functions, f_6 is a step function, and $f_7 - f_9$ are multiplex mode functions [34–36].

Without loss of generality, the parameters setting for each benchmark in Table 1 are as follows:

(i) The dimensions of these functions are all set as 2.

(i) The solution range of the benchmarks is based on Table 1.

(ii) Generate the initial position of the particles randomly within its solution range.

In addition, we have the following.

(i) For LPSO and ARDPSO, the maximum inertia weight is set as 0.9, while the minimum one is 0.5.

ii. For ARDPSO, the parameter a in the formula (5) is set as 1.

Following the above rules, in the MATLAB 2015 (b) environment, standard PSO, LPSO, and ARDPSO are used to test 9 benchmark functions in Table 1, respectively, where each benchmark function was tested 30 times. The number of local optimal solution (that is, $Pbest_i$) updating is shown in Figure 3. The abscissa values 1-9 correspond to benchmarks 1-9, and the ordinate value indicates the number of updating process. Table 2 shows the average updating times for the local optimal solution of the particle. The number of global optimal solution (that is, $Gbest$) updating is shown in Figure 4. Similarly, in Figure 4, the abscissa values 1-9 correspond to benchmarks 1-9, and the ordinate value indicates the number of updating process. Table 4 shows the minimum error, maximum error, and mean error between the fitness of optimal solution and the real optimal solution.

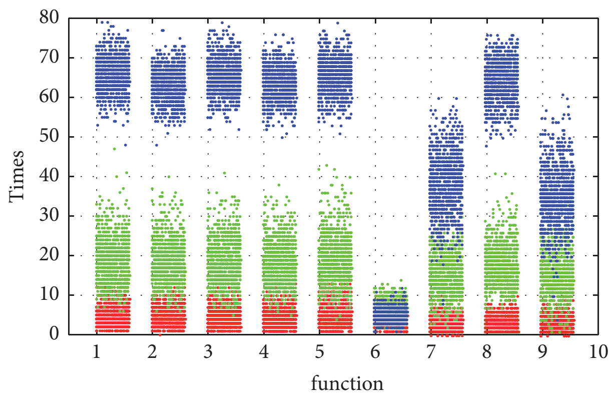


Figure 3: Statistical results diagram of historical local optimal updating (in which the red point presents the standard PSO, the green point presents the LPSO, and the blue point presents ARDPSO; the abscissa values 1-9 correspond to benchmarks 1-9, and the ordinate value indicates the number of updating times).

The performance results as above indicate the following:

(1) For local optimal solution ($Pbest$) from Figure 3 and Table 2, all benchmarks except the step function f_6 show that the average updating times by standard PSO and LPSO are

Table 2: Average updating times of the historical local optimal solution.

	Algorithm	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
Average times	PSO	4.81	4.64	4.63	4.80	5.42	4.47	3.32	3.45	3.42
	LPSO	18.95	18.70	18.77	18.52	19.22	6.92	15.69	18.06	16.01
	ARDPSO	65.74	63.08	65.47	63.69	65.52	5.06	38.49	64.20	35.28

Table 3: Average updating times of the global optimal solution.

	Algorithm	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
Average times	PSO	7.6	8.23	6.5	8.03	11.07	4.83	5.7	6.07	6.5
	LPSO	31.73	31.07	32.3	28.53	33.87	5	29.73	30.5	30.53
	ARDPSO	64.8	63.73	64.13	63.87	64.7	2.93	36.5	61.63	32.27

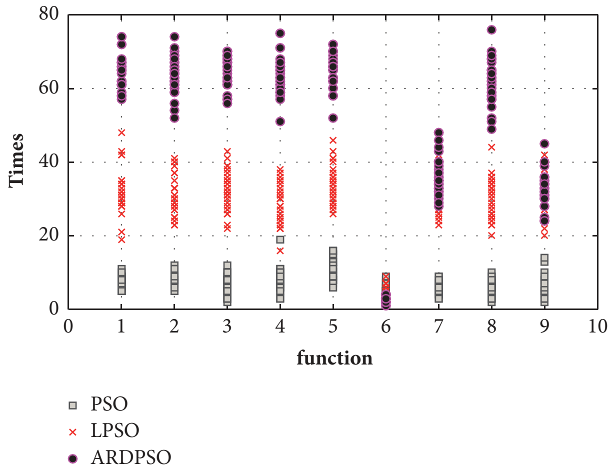


Figure 4: Statistical results diagram of global optimal updating (the abscissa values 1-9 correspond to benchmarks 1-9, and the ordinate value indicates the number of updating times).

about 5 to 18, respectively. However, the average updating times by ARDPSO are about 64 for $f_1 - f_5$ and f_8, while they are about 36 for f_7 and f_9. It is found that the benchmark f_6 may achieve relatively small updating times in standard PSO, LPSO, and ARDPSO.

(2) For global optimal solution (*Gbest*) from Figure 4 and Table 3, for all the benchmarks except the step function f_6, the average updating times by standard PSO are about 6 to 12, and they are about 28 to 32 by LPSO. The average updating times by ARDPSO are more than 60 for $f_1 - f_5$ and f_8, while they are about 33 for f_7 and f_9. Similarly, the benchmark f_6 has relatively small average updating times in standard PSO, LPSO, and ARDPSO.

(3) From the error statistics in Table 4, both LPSO and ARDPSO algorithms are superior to the standard PSO algorithm in all performance evaluations. For step function f_6, both LPSO and ARDPSO can find the real optimal result, reaching an average error=0. For $f_1 - f_5$ and f_8, APDPSO has much lower errors than LPSO. For f_7 and f_9, the relatively small updating times for local and global optimal solution imply that the global optimal solution will not be updated if the real optimal solution is found.

The following can be concluded:

(1) For step function f_6, many solutions may get the optimal fitness, and it means that there are more chances to find the real solution. In this situation, the *Pbset* and *Gbest* are relatively small for standard PSO, LPSO, and ARDPSO.

(2) From formula (2) and (4), it can be seen that LPSO adds an inertia weight ω, compared to the standard PSO. ω decreases with the increasing iterations, and this gives the particles more chances to update its *Pbest* as well as the *Gbest* for better solution.

(3) When the particle could not update its $Pbest_i$, the current solution (x_i^{n+1}) is not a better solution. A too rapid moving speed may make it happen and thus lose the real optimal solution. In the proposed ARDPSO model, when the particle could not update its $Pbest_i$, it can obtain a new solution x_i^{n+1}_rand around the $Pbest_i$. If the $Pbest_i$ is not the real optimal solution, choosing the better one from x_i^{n+1} and x_i^{n+1}_rand as the new position of the particle can encourage $Pbest_i$ to be updated. From the results shown in Tables 2–4, it can be seen that if the real optimal position is not found, the *Pbest* and *Gbest* updating times by ARDPSO may increase far higher than that of LPSO.

4. ECM Parameters Identification Based on the Proposed ARDPSO

In this section, the ARDPSO algorithm is used to identify the parameters of the ECM.

4.1. Preparation for the Experiment. The following experiments use NEWARE BTS-4008 as the power battery test system and 18650 NMC batteries (the positive material is LiCoxNiyMnzO$_2$) as the experiment object. The selected 18650 NMC battery was manufactured in Tianjin with a rated capacity of 2000 mAh and a rated voltage of 3.7 V. The rated charging and discharging cut-off voltages are 4.2 V and 2.5 V, respectively.

The open circuit voltage- (OCV-) state of charge (SOC) curve of the above tested battery is obtained according to the method described in [37].

The error criterion is a crucial factor to determine the precision of parameters identification so that the root square

Table 4: Comparison of performance results.

Algorithm	*Error*	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9
Standard PSO	Minimum Error	0.001238535	0.015541741	0.000359473	0.012069324	0.00312616	0	0.002253	0.015926	3.47E-08
	Average Error	0.07380834	0.331487833	0.061015048	0.139138918	0.362969219	1	0.846391	0.103188	0.004895
	Maximum Error	0.681954989	0.808067334	0.24427567	0.446676649	2.771664211	0.133333	2.130416	0.41696	0.010275
LPSO	Minimum Error	3.93E-14	3.03E-07	2.63E-14	8.81E-08	7.55E-14	0	1.53E-13	2.42E-08	7.77E-16
	Average Error	7.59E-12	3.48E-06	1.15E-11	2.90E-06	6.72E-10	0	0.1990	3.71E-07	0.0043
	Maximum Error	4.22E-11	1.01E-05	1.05E-10	9.85E-06	1.35E-08	0	0.994959	1.58E-06	0.0074
ARDPSO	Minimum Error	8.89E-33	1.26E-16	2.61E-32	3.08E-17	1.23E-32	0	0	8.88E-16	0
	Average Error	3.22E-29	1.98E-15	1.70E-29	2.05E-15	3.90E-27	0	0.0099	4.09E-15	0.0023
	Maximum Error	5.11E-28	7.08E-15	1.18E-28	1.78E-14	7.31E-26	0	0.994959	1.15E-14	0.0074

error is commonly used for battery models parameters identification [9]. However, many factors may influence the precision of the battery model, like the OCV-SOC relationship, the model itself, SOC initial value, and so on. For this reason, a weighted fitness function shown in formula (7) is defined as

$$\begin{gathered} fitness = \alpha * \sqrt{\frac{1}{M}\sum_{k=1}^{M}\left(V_{m,k}-V_{t,k}\right)^2} + \beta \\ * \max\left(\left|V_{m,k}-V_{t,k}\right|_{k=1:M}\right), \\ 0 \le \alpha \le 1,\ 0 \le \beta \le 1,\ \alpha+\beta=1 \end{gathered} \tag{7}$$

where M represents the total number of measurement intervals, V_m is the terminal voltage of the battery system that can be obtained by measurement, V_t is the voltage that is calculated from the battery model, and k represents the k^{th} sampling. In the following experiments, $\alpha = \beta = 0.5$.

4.2. ECM Model Used by the Experiments. At present, the simple model, the first-order RC model, and the second-order RC model are the most applied battery models [9, 38] in industry. The simple model shown in (8) has a simple structure and operating principle. The Thévenin ECM shown in (9) is known as the first-order RC model. The second-order RC model shown in (10) is an extension model from the first-order RC model.

$$V_t = V_{OCV}(SOC) - Ri_L \tag{8}$$

$$V_t = V_{OCV}(SOC) - Ri_L - U_P \tag{9}$$

$$V_t = V_{OCV}(SOC) - Ri_L - U_{P1} - U_{P2} \tag{10}$$

In above formulae, V_t represents the terminal voltage of the battery, $V_{OCV}(SOC)$ represents the value of the OCV corresponding to the value of the SOC, i_L represents the load current, and R represents the internal resistance of the battery. In (9), U_P represents the polarization voltage and its discrete form can be shown using the following formula [39].

$$U_{P,k+1} = U_{P,k}e^{-\Delta t/\tau} + R_P i_{L,k}\left(1-e^{-\Delta t/\tau}\right) \tag{11}$$

where $\tau = R_P C_P$, Δt represents the sampling interval of the system, and R_P and C_P represent the polarization resistance and the polarization capacitor, respectively. In formula (10), U_{P1}, U_{P2}, respectively, represent the electrochemical polarization voltage and the polarization voltage, and the discrete form of U_{P1}, U_{P2} can be expressed as

$$\begin{aligned} U_{P1,k+1} &= U_{P1,k}e^{-\Delta t/\tau_1} + R_{P1} i_{L,k}\left(1-e^{-\Delta t/\tau_1}\right) \\ U_{P2,k+1} &= U_{P2,k}e^{-\Delta t/\tau_2} + R_{P2} i_{L,k}\left(1-e^{-\Delta t/\tau_2}\right) \end{aligned} \tag{12}$$

where $\tau_1 = R_{P1}C_{P1}$, $\tau_2 = R_{P2}C_{P2}$, Δt represents the sampling interval of the system, R_{P1} and C_{P1} represent the electrochemical polarization resistance and the electrochemical polarization capacitor, respectively, and R_{P2} and C_{P2} represent the concentration polarization resistance and the concentration polarization capacitor, respectively. In formulae (10) and (11), k represents the k^{th} sampling.

For the three ECMs, the dynamic behavior of the battery cannot be effectively modeled by the simple model due to the lack of consideration in the battery polarization effect. For example, when the battery is at rest after it is charged/discharged for a certain time, its SOC value is a constant with 0 current. Then, the simple model will output a constant terminal voltage, but in fact its terminal voltage is variable. The first-order RC model is based on the connection with one RC network in series with the simple model, and the delay characteristic of the first-order RC network is used to simulate the polarization effect of the battery. Therefore, during the battery rest, the first-order RC model can output a variable. However, it is unsuitable for the voltage transient process of the battery. The second-order RC model that connects two RC networks in series with the simple model can simulate the electrochemical polarization and concentration polarization separately. Accordingly, it could better model the dynamic behavior of the battery. In this study, the influence from different parameter identification methods is mainly considered.

4.3. Parameter Identification Based on RDPSO Algorithm. The RDPSO algorithm evaluated with the fitness function shown in (7) is used to identify the parameters of the battery model, and the process is indicated in Figure 5.

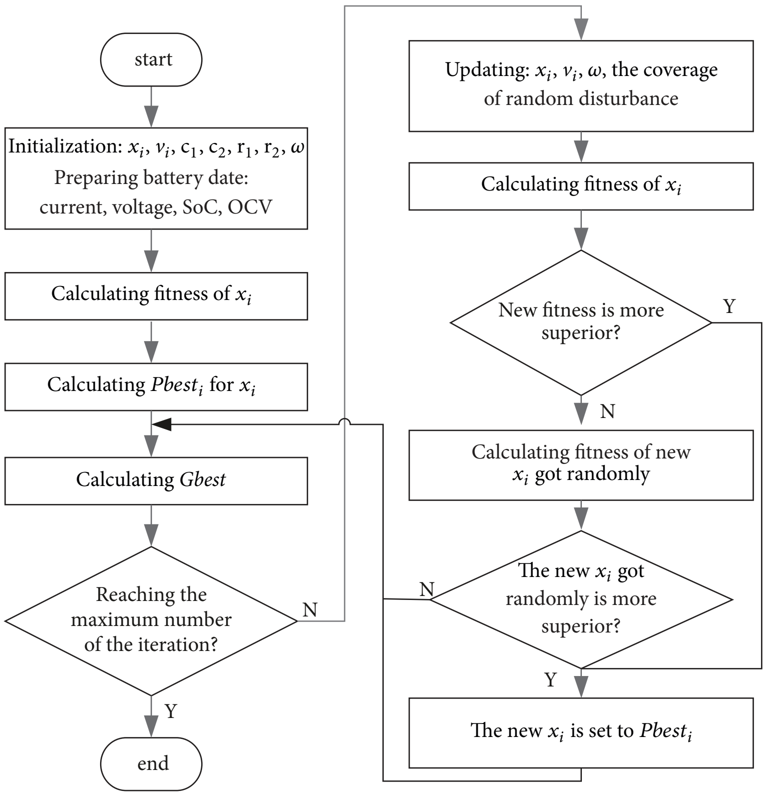

start
Initialization: x_i, v_i, c_1, c_2, r_1, r_2, ω
Preparing battery date: current, voltage, SoC, OCV
Calculating fitness of x_i
Calculating $Pbest_i$ for x_i
Calculating $Gbest$
Reaching the maximum number of the iteration?
N
Y
end
Updating: x_i, v_i, ω, the coverage of random disturbance
Calculating fitness of x_i
New fitness is more superior?
Y
N
Calculating fitness of new x_i got randomly
The new x_i got randomly is more superior?
N
Y
The new x_i is set to $Pbest_i$

FIGURE 5: Flow diagram of system identification based on ARDPSO.

5. Performance Results and Discussion

5.1. The Working Conditions. Based on the known capacity and the OCV-SOC curve of the battery, the LPSO and ARDPSO are used to identify the parameters of the abovementioned three ECM models. In this section, two different operating conditions are selected for experiments. The first one is the test operating condition shown in Figures 6(a)–6(d), and it is used to validate the effectiveness of the ARDPSO. The second one is the verification operating condition shown in Figures 7(a)–7(d), and it is used to validate the generality of the ARDPSO. In other words, the battery model parameters identified by one operating condition can suit other operating conditions. In both operating conditions, four types of battery information are provided, shown in Figures 6 and 7. In Figures 6(a)–6(d) and 7(a)–7(d), the abscissa is time and the measurement interval is 1 s; moreover, Figures 7(a)–7(d) have the same meaning with corresponding (a)–(d) shown in Figure 6.

In these figures, we have the following.

(i) Figure 6(a) shows the current for each measurement interval.

(ii) Figure 6(b) shows the battery voltage obtained from each measurement interval.

(iii) Figure 6(c) shows the value of the SOC from each measurement interval. In the test operating condition, the battery rests for two hours so that the terminal voltage of the battery can be used as initial OCV. Consequently, the initial SOC value can be obtained via the OCV-SOC relationship. Other SOC values can be obtained through Amper–Hour integral [8] using the known initial SOC value and the current shown in Figure 6(b).

(iv) Figure 6(d) shows the value of OCV corresponding to the SOC value for each measurement interval.

5.2. Effectiveness Verification. The simple, first-order RC, and second-order RC models were used to test the least square method [8], LPSO, and ARDPSO in the parameters identification under the condition shown in Figure 6. In Table 5, the performance results are concluded according to the root mean square (RMS) error, maximum error (Max error), and the accumulative error (Acc error), where each error formula is expressed as

$$\begin{aligned}
&\text{RMS error: } \sqrt{\frac{1}{M}\sum_{k=1}^{M}\left(V_{m,k}-V_{t,k}\right)^2}\\
&\text{Max error: } \max\left(\left|V_{m,k}-V_{t,k}\right|_{k=1:M}\right) \qquad (13)\\
&\text{Max error: } \sum_{k=1}^{M}\left|V_{m,k}-V_{t,k}\right|
\end{aligned}$$

Note that each variable in formula (13) has the same definition as that of formula (7).

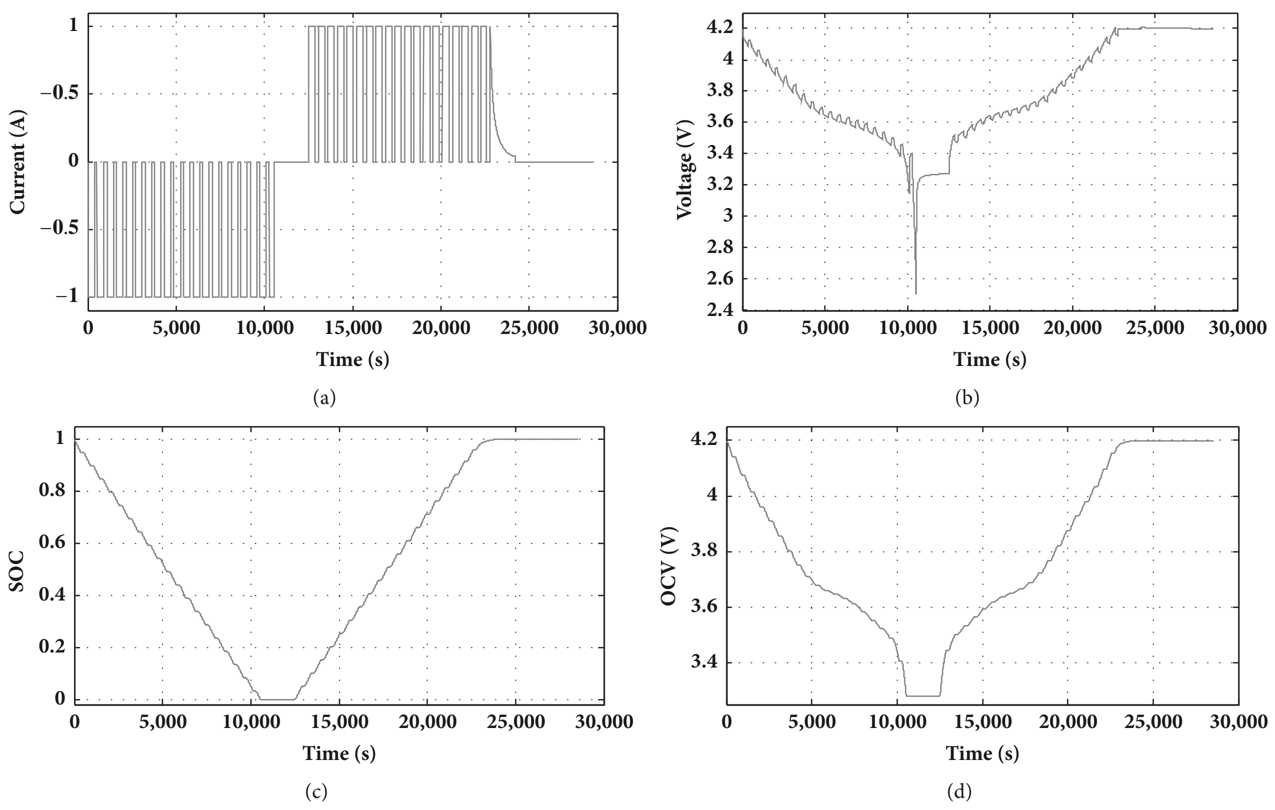


Figure 6: Test operating condition: (a) battery current; (b) measurement voltage; (c) SOC; (d) OCV.

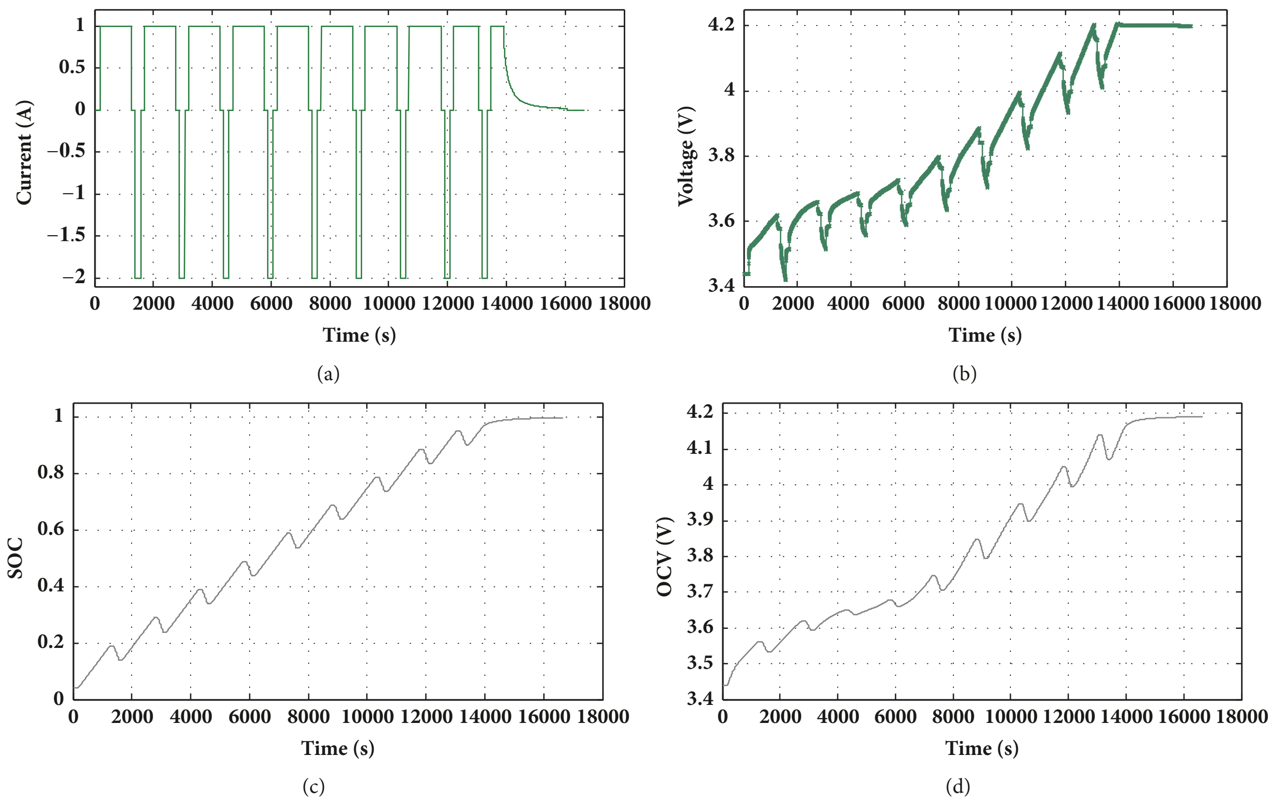


Figure 7: Verification operating condition: (a) battery current; (b) measurement voltage; (c) SOC; (d) OCV.

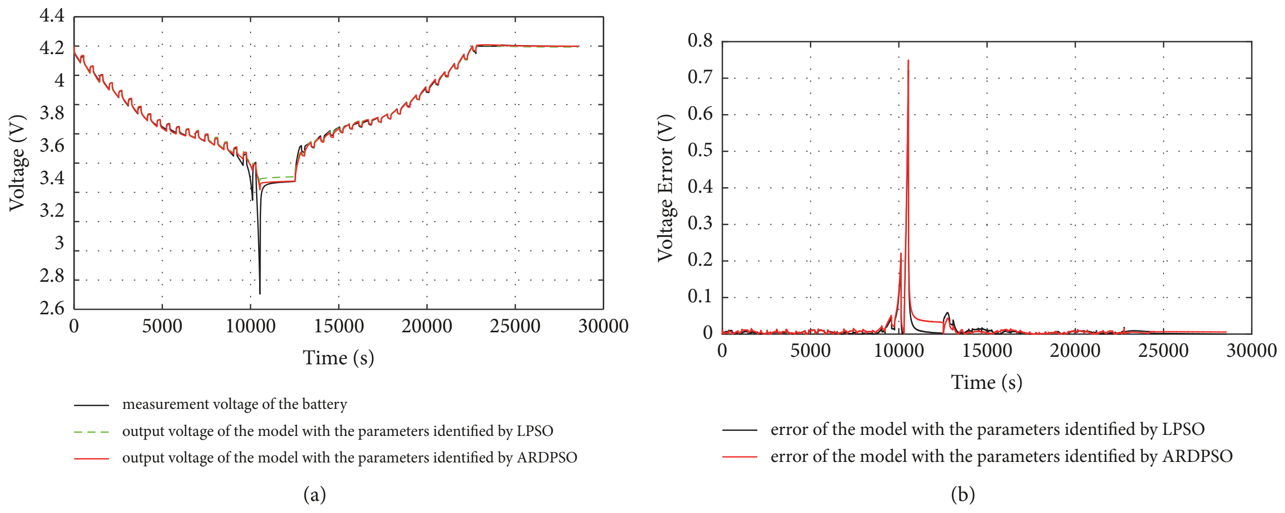


FIGURE 8: Testing results of the second-order RC model: (a) the measured voltage of the battery and the simulation results of the battery model; (b) simulation errors of the battery model.

TABLE 5: Error result of least square, LPSO, and ARDPSO in test operating conditions.

Algorithm	Error	Simple model	First-order RC	Second-order RC
Least Square	RMS Error	0.0472	0.0451	0.0426
	Max Error	0.7836	0.7045	0.7264
	Acc Error	578.7310	413.8880	383. 9528
LPSO	RMS Error	0.0472	0.0443	0.0434
	Max Error	0.7836	0.7196	0.7494
	Acc Error	578.5120	386.2990	347.1315
ARDPSO	RMS Error	0.0461	0.0421	0.0397
	Max Error	0.7836	0.7139	0.7217
	Acc Error	544.5102	323.9048	313.6077

The above three models based on the same parameters have similar output voltages and error distributions, and only the results of the second-order RC model based on LPSO and ARDPSO are shown in Figure 8. The measured voltage of the battery and the simulation result are shown in Figure 8(a). The simulation errors of the battery model are shown in Figure 8(b).

From the experiment results, the following can be concluded:

(1) When the three abovementioned models are identified under the same operating conditions, the global optimal location is further optimized by the ARDPSO with the added random disturbance function, as compared to the LPSO. As a result, the output voltage obtained from the ARDPSO is much closer to the true battery voltage than the LPSO.

(2) Under the same operating conditions, the same identification method is used to obtain the parameters of the three models. The second-order RC model that uses two RC networks to simulate the electrochemical polarization and concentration polarization has more precision than the first-order RC model with one RC network. However, the simple model does not consider the polarization characteristics; comparing with the simple model, the first-order RC model is more precise.

(3) From Figure 8(a), it is shown that, during the battery discharging process, when the value of SOC is small, the voltage drops relatively fast for the same current due to the battery inherent characteristic. It would result in a large error in this case. Consequently, most of the literatures use only the 20%–80% of SOC information [9, 40]. It can be seen from Figure 8(b) that LPSO and ARDPSO have relatively small errors as the value of SOC is in between 20% and 80%.

(4) Table 5 reveals that the RMS error, the max error, and the min error of ARDPSO are smaller than those of the LPSO; therefore, it can be seen that the novel method combination with the adaptive random disturbance can produce a better performance.

(5) From Table 5, comparing with the least square algorithm, LPSO and ARDPSO could get lower Acc Error. However, the result of least square algorithm is very sensitive to the initial value setting, and an improper initial setting may cause a large error.

5.3. Generality Verification. To validate the generality of the ARDPSO for each of ECM models, the parameters identified under the test operating condition are used as the input to obtain the output voltage of ECM model. Figure 9 shows the output voltage error distribution in different ECM models.

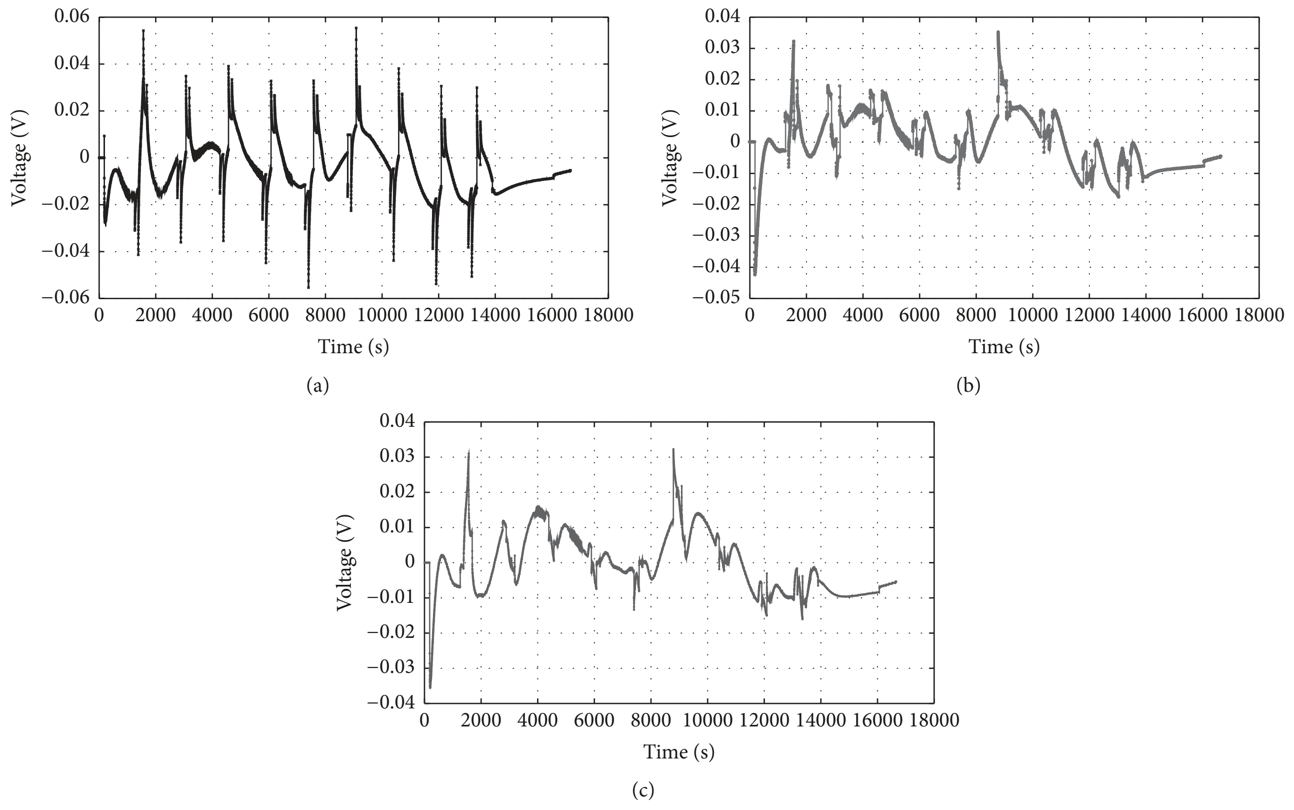


Figure 9: Error distribution of the battery models with the parameters identified by ARDPSO: (a) simple model; (b) first-order RC model; (c) second-order RC model.

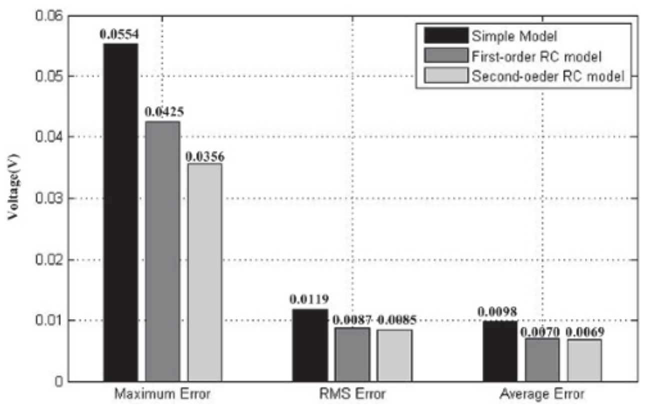


Figure 10: Error results of the battery model parameters by ARDPSO.

Figure 10 concludes the statistical results of the maximum error, RMS error, and the average error.

The following can be seen from Figures 9 and 10:

(1) For the three abovementioned models, the results of the verification operating condition (as shown in Figure 7) using the parameters identified under test operating condition (as shown in Figure 6) indicate that the output voltage of the ECMs almost coincides with the real battery voltage, shown in Figure 7(b). As a result, throughout the whole simulation process, the maximum error is low enough within a small range, as shown in Figure 9.

(2) Among the three models, the second-order model can better simulate the static and dynamic behavior of the battery. Thus the second-order RC model has smaller maximum error, RMS error, and average error than simple model and the first-order RC model.

(3) The parameters identified by the proposed ARDPSO under one operating condition are suitable for other operating conditions for ECM models, verifying the effectiveness and generality of the ARDPSO. That is, the battery model can be used by BMS to predict the states of the battery.

6. Conclusions

The PSO has been widely used in many applications like identification of ECM model. It and its extended algorithm such as LPSO could update both local and global optimal solutions by moving particles to achieve the target. However, it is found that the solution either local or global optimum may not keep updating for a period of time during the particles movement. The ARDPSO algorithm is proposed to continue to update the optimal solutions. Test results from multiple benchmark functions have verified that the ARDPSO can improve the updating process for both local and global optimal solutions. Accordingly, the ARDPSO can reach higher solution precision than the standard PSO and LPSO.

As ECM model parameters can affect the static and dynamic behaviors of the battery model, the ARDPSO therefore introduces a new weighted fitness function to identify

ECM parameters. Based on the evaluation tool using the maximum error, RMS error, and the average error, it is obvious that the parameters of ECM model have been identified accurately under the test operating condition. Besides, it indicates the ARDPSO promises a better performance than the LPSO. For future work, the black box algorithms such as neural network and support vector machine will be used to model the battery and further compared with the ARDPSO algorithm in the state of charge (SOC) and state of health (SOH).

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research work was partly supported by the National Science and Technology Ministry [no. 2015BAA09B01], the National Natural Science Foundation of China [Project no. 51377044], the Natural Science Foundation of Hebei Province [Project no. E2017202284], and the Youth Foundation of Hebei Education Department [Project nos. QN2017314, QN2017316].

References

[1] J. Unger, C. Hametner, S. Jakubek, and M. Quasthoff, "A novel methodology for non-linear system identification of battery cells used in non-road hybrid electric vehicles," *Journal of Power Sources*, vol. 269, pp. 883–897, 2014.

[2] A. P. Schmidt, M. Bitzer, Á. W. Imre, and L. Guzzella, "Experiment-driven electrochemical modeling and systematic parameterization for a lithium-ion battery cell," *Journal of Power Sources*, vol. 195, no. 15, pp. 5071–5080, 2010.

[3] M. A. Rahman, S. Anwar, and A. Izadian, "Electrochemical model parameter identification of a lithium-ion battery using particle swarm optimization method," *Journal of Power Sources*, vol. 307, pp. 86–97, 2016.

[4] X. Dang, L. Yan, K. Xu, X. Wu, H. Jiang, and H. Sun, "Open-Circuit Voltage-Based State of Charge Estimation of Lithium-ion Battery Using Dual Neural Network Fusion Battery Model," *Electrochimica Acta*, vol. 188, pp. 356–366, 2016.

[5] W. He, N. Williard, C. Chen, and M. Pecht, "State of charge estimation for Li-ion batteries using neural network modeling and unscented Kalman filter-based error cancellation," *International Journal of Electrical Power & Energy Systems*, vol. 62, pp. 783–791, 2014.

[6] C. Zhang, K. Li, S. McLoone, and Z. Yang, "Battery modelling methods for electric vehicles - A review," in *Proceedings of the 13th European Control Conference, ECC 2014*, pp. 2673–2678, fra, June 2014.

[7] K. M. Tsang, L. Sun, and W. L. Chan, "Identification and modelling of Lithium ion battery," *Energy Conversion and Management*, vol. 51, no. 12, pp. 2857–2862, 2010.

[8] G. L. Plett, "Extended Kalman filtering for battery management systems of LiPB-based HEV battery packs—part 2. Modeling and identification," *Journal of Power Sources*, vol. 134, no. 2, pp. 262–276, 2004.

[9] X. S. Hu, S. B. Li, and H. Peng, "A comparative study of equivalent circuit models for Li-ion batteries," *Journal of Power Sources*, vol. 198, pp. 359–367, 2012.

[10] C. Zhang, C. Zhang, J. Liu, and S. M. Sharkh, "Identification of dynamic model parameters for lithium-ion batteries used in hybrid electric vehicles," *High Technology Letters*, vol. 16, no. 1, pp. 6–12, 2010.

[11] M. A. Roscher and D. U. Sauer, "Dynamic electric behavior and open-circuit-voltage modeling of $LiFePO_4$-based lithium ion secondary batteries," *Journal of Power Sources*, vol. 196, no. 1, pp. 331–336, 2011.

[12] A. Seaman, T.-S. Dao, and J. McPhee, "A survey of mathematics-based equivalent-circuit and electrochemical battery models for hybrid and electric vehicle simulation," *Journal of Power Sources*, vol. 256, pp. 410–423, 2014.

[13] J. Sabatier, J. M. Francisco, F. Guillemard, L. Lavigne, M. Moze, and M. Merveillaut, "Lithium-ion batteries modeling: A simple fractional differentiation based model and its associated parameters estimation method," *Signal Processing*, vol. 107, pp. 290–301, 2015.

[14] X. Zhang, Y. Wang, D. Yang, and Z. Chen, "An on-line estimation of battery pack parameters and state-of-charge using dual filters based on pack model," *Energy*, vol. 115, pp. 219–229, 2016.

[15] Y. Li, P. Chattopadhyay, A. Ray, and C. D. Rahn, "Identification of the battery state-of-health parameter from input-output pairs of time series data," *Journal of Power Sources*, vol. 285, pp. 235–246, 2015.

[16] Y. Hu and S. Yurkovich, "Linear parameter varying battery model identification using subspace methods," *Journal of Power Sources*, vol. 196, no. 5, pp. 2913–2923, 2011.

[17] G. L. Plett, "Sigma-point Kalman filtering for battery management systems of LiPB-based HEV battery packs. Part 2: Simultaneous state and parameter estimation," *Journal of Power Sources*, vol. 161, no. 2, pp. 1369–1384, 2006.

[18] H. Zhu, Y.-F. Liu, and C. Zhao, "Parameter identification and SOC estimation of lithium ion battery," *Hunan Daxue Xuebao/Journal of Hunan University Natural Sciences*, vol. 41, no. 3, pp. 37–42, 2014.

[19] S. Nejad, D. T. Gladwin, and D. A. Stone, "A systematic review of lumped-parameter equivalent circuit models for real-time estimation of lithium-ion battery states," *Journal of Power Sources*, vol. 316, pp. 183–196, 2016.

[20] F. Baronti, W. Zamboni, N. Femia et al., "Parameter identification of Li-Po batteries in electric vehicles: A comparative study," in *Proceedings of the 2013 IEEE 22nd International Symposium on Industrial Electronics, ISIE 2013*, twn, May 2013.

[21] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia, December 1995.

[22] Y. Shang, Q. Zhang, N. Cui, and C. Zhang, "Research on variable-order RC equivalent circuit model for lithium-ion battery based on the AIC criterion," *Diangong Jishu Xuebao/Transactions of China Electrotechnical Society*, vol. 30, no. 17, pp. 55–62, 2015.

[23] T. Kim, W. Qiao, and L. Qu, "Real-time state of charge and electrical impedance estimation for lithium-ion batteries based on a hybrid battery model," in *Proceedings of the 28th Annual IEEE Applied Power Electronics Conference and Exposition (APEC '13)*, pp. 563–568, Long Beach, Calif, USA, March 2013.

[24] N. Moldovan, R. Picos, and E. Garcia-Moreno, "Parameter extraction of a solar cell compact model usign genetic algorithms," in *Proceedings of the Spanish Conference on Electron Devices (CDE '09)*, pp. 379–382, Santiago de Compostela, Spain, February 2009.

[25] Y. Shi and R. Eberhart, “A modified particle swarm optimizer,” in *Proceedings of the IEEE International Conference on Evolutionary Computation and IEEE World Congress on Computational Intelligence*, (Cat. No.98TH8360), pp. 69–73, Anchorage, Alaska, USA, May 1998.

[26] S.-F. Cheng, X.-H. Cheng, and L. Yang, “Application of wavelet neural network with improved particle swarm optimization algorithm in power transformer fault diagnosis,” *Dianli Xitong Baohu yu Kongzhi/Power System Protection and Control*, vol. 42, no. 19, pp. 37–42, 2014.

[27] H.-C. Tsai, Y.-Y. Tyan, Y.-W. Wu, and Y.-H. Lin, “Gravitational particle swarm,” *Applied Mathematics and Computation*, vol. 219, no. 17, pp. 9106–9117, 2013.

[28] L. Wang, J. Zhao, D. Liu, J. Wang, G. Chen, and W. Liu, “Governor actuator piecewise linear model and parameter identification based on genetic algorithm-particle swarm optimization,” *Diangong Jishu Xuebao/Transactions of China Electrotechnical Society*, vol. 31, no. 12, pp. 204–210, 2016.

[29] P. Han and S. Yuan, “Multivariable system identification based on double quantum particle swarm optimization and big data,” *Zhongguo Dianji Gongcheng Xuebao/Proceedings of the Chinese Society of Electrical Engineering*, vol. 34, no. 32, pp. 5779–5787, 2014.

[30] Y. Xiang, X.-J. Ma, C.-G. Liu, R.-S. Ke, and Z.-X. Zhao, “Estimation of model parameters and SOC of lithium batteries based on IPSO-EKF,” *Binggong Xuebao/Acta Armamentarii*, vol. 35, no. 10, pp. 1659–1666, 2014.

[31] Z. Wang, S. Bian, X. Liu, K. Yu, and Y. Shi, “Research on load model parameter identification based on the CQDPSO algorithm,” *Transactions of China Electrotechnical Society*, vol. 29, no. 12, pp. 211–217, 2014.

[32] Z. Cheng, M. Dong, T. Yang, and L. Han, “Extraction of solar cell model parameters based on self-adaptive chaos particle swarm optimization algorithm,” *Diangong Jishu Xuebao/Transactions of China Electrotechnical Society*, vol. 29, no. 9, pp. 245–252, 2014.

[33] A. Khare and S. Rangnekar, “A review of particle swarm optimization and its applications in Solar Photovoltaic system,” *Applied Soft Computing*, vol. 13, no. 5, pp. 2997–3006, 2013.

[34] Feng. Siling, “Biogeography-based optimization algorithm and application on biological sequence motif discovery [D],” *School of Computer Science*, 2014.

[35] X. Yao, Y. Liu, and G. Lin, “Evolutionary programming made faster,” *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 2, pp. 82–102, 1999.

[36] Feng. Quanxi, *Research on biogeography-based optimization and its application[D]*, Xidian University, China, 2014.

[37] Li. Jianwei, *Performance-Driven Behavioral Battery Modeling for Large Format Batteries*, Mississippi State University, Mississippi, USA, 2012.

[38] H. W. He, R. Xiong, H. Q. Guo, and S. C. Li, “Comparison study on the battery models used for the energy management of batteries in electric vehicles,” *Energy Conversion and Management*, vol. 64, pp. 113–121, 2012.

[39] H. Dai, X. Wei, Z. Sun, J. Wang, and W. Gu, “Online cell SOC estimation of Li-ion battery packs using a dual time-scale Kalman filtering for EV applications,” *Applied Energy*, vol. 95, pp. 227–237, 2012.

[40] F. Zheng, Y. Xing, J. Jiang, B. Sun, J. Kim, and M. Pecht, “Influence of different open circuit voltage tests on state of charge online estimation for lithium-ion batteries,” *Applied Energy*, vol. 183, pp. 513–525, 2016.