Parameter extraction of different fuel cell models with transferred adaptive differential evolution ✩

Wenyin Gong a,b, Xuesong Yan a,b, Xiaobo Liu c, Zhihua Cai a,b

a Hubei Intelligent Geo-information Processing Key Laboratory (China University of Geosciences), Wuhan 430074, P.R. China
b School of Computer Science, China University of Geosciences, Wuhan 430074, P.R. China
c School of Automation, China University of Geosciences, Wuhan 430074, P.R. China

Abstract

To improve the design and control of fuel cell (FC) models, it is important to extract their unknown parameters. Generally, the parameter extraction problems of FC models can be transformed as nonlinear and multi-variable optimization problems. To extract the parameters of different FC models exactly and fast, in this paper, we propose a transferred adaptive differential evolution (DE) framework, in which the successful parameters of the adaptive DE solving previous problems are properly transferred to solve new optimization problems in the similar problem-domains. Based on this framework, an improved adaptive DE method (TRADE, in short) is presented as an illustration. To verify the performance of our proposal, TRADE is used to extract the unknown parameters of two types of fuel cell models, i.e., proton exchange membrane fuel cell (PEMFC) and solid oxide fuel cell (SOFC). The results of TRADE are also compared with those of other state-of-the-art evolutionary algorithms (EAs). Even though the modification is very simple, the results indicate that TRADE can extract the parameters of both PEMFC and SOFC models exactly and fast. Moreover, the V-I characteristics obtained by TRADE agree well with the simulated and experimental data in all cases for both types of fuel cell models. Also, it improves the performance of the original adaptive DE significantly in terms of both the quality of final solutions and the convergence speed in all cases. Additionally, TRADE is able to provide better results compared with other EAs.

Key words: Fuel cell, parameter extraction, differential evolution

1. Introduction

In recent years, the fuel cell (FC) technology has received considerable attention because of the low emission to environment, superior durability, good transient responses, high energy efficiency, high scalability, and so on [39]. There are several different types of fuel cells based on the nature of the electrolyte used, such as proton exchange membrane fuel cell (PEMFC), molten carbonate fuel cell (MCFC), solid oxide fuel cell (SOFC), and so on. Among them, PEMFC and SOFC have been widely studied and used in promising area for different applications[12, 61]. More details can be found in the following representative review papers [54, 41, 70, 40].

Although many PEMFC and SOFC models have been developed, their model parameters must be accurately extracted and optimized all the time to obtain high-performance system control. Generally, parameter extraction of PEMFC and SOFC models can be converted into numerical optimization problems. However, due to the nonlinear and multi-variable features, it may cause difficulties to traditional optimization techniques. Recently, the heuristic

✩ This work was partly supported by the National Natural Science Foundation of China under Grant Nos. 61203307 and 61375066, the Natural Science Foundation of Hubei Province under Grant No. 2013CFA004, the Fundamental Research Funds for the Central Universities at China University of Geosciences (Wuhan) under Grant No. CUG130413.

Corresponding author. Tel: +86-27-67883716.
Email address: wenyinong@yahoo.com; wygong@cug.edu.cn (Wenyin Gong)
Preprint submitted to Elsevier

March 23, 2015
optimization techniques have been used to solve the problems [2], for example such as genetic algorithms [49], simulated annealing [51], particle swarm optimization [78], harmony search [5], seeker optimization algorithm [17], artificial immune system [4], P systems based optimization algorithm [76], differential evolution [13] for PEMFC, and genetic algorithm [75], differential evolution [29] for SOFC, etc. However, in order to exactly and fast solve the parameter extraction problems of PEMFC and SOFC models, it is necessary to investigate more efficient optimization techniques to reduce the necessary computational efforts to achieve an optimal design [62].

In the similar problem-domains, different optimization problems always have similar features. For example, the parameter extraction problems of PEMFC and SOFC models can be converted into the optimization problems that have similar objective functions [49, 75]. Therefore, when the optimization techniques are used to find the optimal solutions for similar problems, the previous problem-solving experiences maybe benefit to solve new problems.

Differential evolution (DE), proposed by Storn and Price [65], is a simple and efficient evolutionary algorithm (EA) for global numerical optimization. It has been obtained many successful applications in diverse fields [19], such as engineering design, digital filter design [58], optimal power flow [66], simulation of solar-thermal refrigeration systems [23], hydrothermal generation scheduling [72, 79], etc. In DE, the parameter settings of the crossover rate $CR$ and the scaling factor $F$ are crucial to DE’s performance [24]. To improve its performance, different parameter adaptation techniques have been proposed in the DE literature [45, 11, 59, 80]. However, in the current work, there is an apparent impediment, i.e., the successful parameters solving previous problems do not be reused and transferred to solve new problems. For example, in JADE [80], the parameters of $\mu_{CR}$ and $\mu_{F}$ are usually initialized to be 0.5 when solving new optimization problems, but without considering their previous problem-solving experiences.

In machine learning, transfer learning [53] is a new learning framework that can transfer knowledge across different problems to improve the performance of learning. Inspired by the success of transfer learning, in this paper, a transferred adaptive DE framework is proposed, where the successful parameters of the adaptive DE solving previous problems are properly reused and transferred to solve new optimization problems in the similar problem-domains. As an example, based on the proposed framework, the parameters of $\mu_{CR}$ and $\mu_{F}$ in JADE [80] are initialized based on their previous problem-solving experiences when solving new encountered problems. The improved JADE method is referred to as TRADE, in short. TRADE is used to extract the unknown parameters of two different types of FC models, i.e., PEMFC model and SOFC model [39]. The reasons of selecting these two models are two-fold: i) The parameter extraction problems of PEMFC and SOFC models are very important to improve the performance of these two FC models, and hence, they have obtained considerable attention recently [51, 8, 75]. ii) Both of these problems can be formulated as the optimization problems, and they have similar objective functions. To the best of our knowledge, it is the first attempt to combine DE with transfer learning to solve the parameter extraction problems of both PEMFC and SOFC models simultaneously.

The main contributions of this paper are three-fold:

i) A framework of transferred adaptive differential evolution is proposed, which is simple and generic.

ii) Based on the framework, an improved JADE method, referred to as TRADE, is presented to extract the parameters of two different FC models.

iii) TRADE is highly efficient and effective to extract the parameters of both PEMFC and SOFC models, which can be an efficient alternative to other complex optimization problems of FC models.

The rest of this paper is organized as follows. Section 2 briefly introduces the parameter adaptation technique of JADE and the problem formulations of PEMFC and SOFC models used in this work. In Section 3, the related work is presented. The proposed transferred adaptive DE framework and the TRADE method are elucidated in Section 4. Followed by Section 5, the results are described and analyzed in this section. Finally, Section 6 concludes the work and points out some possible future work.

2. Preliminaries

In this section, the problem formulations of PEMFC and SOFC models and the parameter adaptation technique of JADE proposed in [80] are briefly introduced.
2.1. Problem Formulations

A fuel cell (FC) is an electrochemical device that converts chemical energy directly into electrical energy [39]. According to the nature of used electrolyte, there are several different types of FCs. However, among various types of FCs, PEMFC and SOFC have obtained considerable attention in recent years. In this section, the two FC models are briefly introduced as follows.

2.1.1. Mathematical Formulation of PEMFC Stack Model

In this work, the PEMFC stack model presented in [49] is used. For \( n_{\text{cell}} \) cells connected in series to form a stack, the terminal voltage of the stack can be calculated by [16],

\[
V_{\text{PEMFC}} = n_{\text{cell}} \cdot (E_{\text{Nernst}} - V_{\text{act}} - V_{\text{ohm}} - V_{\text{con}})
\]

where \( E_{\text{Nernst}} \) is the thermodynamic potential defined by

\[
E_{\text{Nernst}} = 1.229 - 0.846 \times 10^{-3} \cdot (T - 298.15) + 4.3085 \times 10^{-5} \cdot T \cdot \ln \left( \frac{P_{\text{H}_2}}{P_{\text{O}_2}} \right)
\]

The activation overpotential \( V_{\text{act}} \), including anode and cathode, can be expressed by the following formula [48]

\[
V_{\text{act}} = -\left( \xi_1 + \xi_2 \cdot T + \xi_3 \cdot \ln \left( \frac{P_{\text{O}_2}}{5.08 \times 10^6 \cdot \exp \left( -498/T \right)} \right) + \xi_4 \cdot T \cdot \ln (i_{\text{cell}}) \right)
\]

The ohmic voltage drop \( V_{\text{ohm}} \) can be determined by the following expression [48]

\[
V_{\text{ohm}} = (i_{\text{cell}} + i_n) \cdot \left( \frac{\ell}{A} \cdot \frac{181.6 \cdot \left( 1 + 0.03 \cdot \left( \frac{i_{\text{cell}} + i_n}{A} \right) + 0.062 \cdot \left( \frac{i_{\text{cell}} + i_n}{A} \right)^2 \right)}{\left[ \lambda - 0.634 - 3 \cdot \left( \frac{i_{\text{cell}} + i_n}{A} \right) \right] \cdot \exp \left[ 4.18 \cdot \left( \frac{T - 298}{T} \right) \right]} + R_C \right)
\]

The concentration overpotential \( V_{\text{con}} \) caused by the change in the concentration of the reactants at the surface of the electrodes as the fuel is calculated by [15]

\[
V_{\text{con}} = -B \cdot \ln \left( 1 - \frac{i_{\text{cell}} + i_n}{i_{\text{max}}} \right)
\]

In the above equations, the temperature \( T \), the hydrogen partial pressure \( P_{\text{H}_2}^* \), and the oxygen partial pressure \( P_{\text{O}_2}^* \) are measurable and dependent on the operating conditions of the system. \( i_{\text{cell}} \) is the cell current. The other 11 parameters \( \xi_1, \xi_2, \xi_3, \xi_4, \lambda, R_C, B, i_n, i_{\text{max}}, \ell, \text{ and } A \) are unknown that need to be extracted to improve the design of PEMFC model. When the optimization techniques are used to the parameter extraction problems of PEMFC model, the objective function should be defined at first. In the literature, the mean squared error (MSE) between the output voltage of the actual PEMFC stack and the model output voltage are used as the objective function [49]:

\[
\min \ f_{\text{PEMFC}}(\mathbf{x}) = \frac{1}{N} \cdot \sum_{k=1}^{N} (V_{\text{PEMFC,sa,k}} - V_{\text{PEMFC,so,k}})^2
\]

where \( \mathbf{x} = (\xi_1, \xi_2, \xi_3, \xi_4, \lambda, R_C, B, i_n, i_{\text{max}}, \ell, A) \), \( V_{\text{PEMFC,sa}} \) is the output voltage of the actual PEMFC stack, \( V_{\text{PEMFC,so}} \) is the model output voltage calculated by Equation (1), and \( N \) is the number of the data points. The search ranges of these unknown parameters are reported in Table 1. Note that, in Table 1, \( J_n = i_n \times A \) is the current density of the cell, and \( J_{\text{max}} = i_{\text{max}} \times A \).

<table>
<thead>
<tr>
<th>Parameter</th>
<th>( \xi_1 )</th>
<th>( \xi_2 )</th>
<th>( \xi_3 )</th>
<th>( \xi_4 )</th>
<th>( \lambda )</th>
<th>( R_C ) (\Omega)</th>
<th>( B ) (V)</th>
<th>( i_n ) (mA/cm(^2))</th>
<th>( J_{\text{max}} ) (mA/cm(^2))</th>
<th>( \ell ) (\mu m)</th>
<th>( A ) (cm(^2))</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lower bound</td>
<td>-3.1997</td>
<td>0.001</td>
<td>3.64E-05</td>
<td>-2.66E-04</td>
<td>0.0001</td>
<td>0.0150</td>
<td>1</td>
<td>1</td>
<td>50</td>
<td>1500</td>
<td>200</td>
</tr>
<tr>
<td>Upper bound</td>
<td>-0.8532</td>
<td>0.005</td>
<td>-9.38E-05</td>
<td>-9.54E-05</td>
<td>24</td>
<td>0.0008</td>
<td>0.5</td>
<td>30</td>
<td>1500</td>
<td>200</td>
<td>100</td>
</tr>
</tbody>
</table>
2.1.2. Mathematical Formulation of SOFC Stack Model

To implement the control of an SOFC stack for the output performance, a simple electrochemical model is presented in [39, 14, 29], which is modeled by:

\[
V_{SOFC} = N_{cell} \left[ E_0 - A_{SOFC} \sinh^{-1} \left( \frac{I}{2I_{0,a}} \right) - A_{SOFC} \sinh^{-1} \left( \frac{I}{2I_{0,c}} \right) - IR_{ohm} + B_{SOFC} \ln \left( 1 - \frac{I}{I_L} \right) \right]
\]

(3)

where \(N_{cell}\) is the number of cells in the SOFC stack, \(E_0\) is the open-circuit voltage, \(A_{SOFC}\) is the slope of Tafel line, \(R_{ohm}\) is the area-specific resistance in \(\Omega\), \(B_{SOFC}\) is a constant that depends on the fuel cell and its operating state, \(I\) is the current in mA, and \(I_L\) is the limit current in mA.

In the above SOFC electrochemical model formulated by Equation (3), there are seven unknown parameters, \(E_0, A_{SOFC}, I_{0,a}, I_{0,c}, R_{ohm}, B_{SOFC},\) and \(I_L\), which should be extracted by the optimization algorithm. When the optimization techniques are used to the parameter extraction problems of SOFC model, the mean squared error (MSE) is used as the objective function [75]:

\[
\min f_{SOFC}(x) = \frac{1}{N} \sum_{k=1}^{N} (V_{SOFC,sa,k} - V_{SOFC,so,k})^2
\]

(4)

subject to

\[I_k < I_L, \quad k = 1, \ldots, N\]

where \(x = \{E_0, A_{SOFC}, I_{0,a}, I_{0,c}, R_{ohm}, B_{SOFC}, I_L\}\), \(N\) is the number of the sample data, and \(I_k\) is the \(k\)th current in the sample data. \(V_{SOFC,sa}\) is the output voltage of the actual SOFC stack and \(V_{SOFC,so}\) is the model output voltage calculated by Equation (3). For the seven unknown parameters, the search ranges are tabulated in Table 2.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Lower bound</th>
<th>Upper bound</th>
</tr>
</thead>
<tbody>
<tr>
<td>(E_0) (V)</td>
<td>0.0</td>
<td>3.2</td>
</tr>
<tr>
<td>(A_{SOFC}) (V)</td>
<td>0.0</td>
<td>1.0</td>
</tr>
<tr>
<td>(I_{0,a}) (mA)</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>(I_{0,c}) (mA)</td>
<td>0.0</td>
<td>100.0</td>
</tr>
<tr>
<td>(R_{ohm}) (Ω)</td>
<td>0.0</td>
<td>0.2</td>
</tr>
<tr>
<td>(B_{SOFC}) (V)</td>
<td>0.0</td>
<td>0.2</td>
</tr>
<tr>
<td>(I_L) (mA)</td>
<td>0.0</td>
<td>200.0</td>
</tr>
</tbody>
</table>

2.2. Parameter Adaptation in JADE

Differential evolution (DE), which was firstly proposed by Storn and Price in 1995 [64, 65], is one of the most powerful evolutionary algorithms for global numerical optimization. The advantages of DE are its ease of use, simple structure, speed, efficacy, and robustness. In the last few years, DE has obtained many successful applications in diverse domains, such as engineering optimal design, digital filter design, image processing, data mining, multisensor fusion, and so on [58, 19]. However, the performance of DE is sensitive to its parameter settings [24]. Based on this consideration, some researchers studied the parameter adaptation techniques to improve DE’s performance, such as jDE [11], SaDE [59], JADE [80], SaJADE [28], and so on. Because the parameter adaptation technique in JADE is used in this work, it is briefly described as follows.

In JADE [80], the parameters \(CR\) and \(F\) are adaptively controlled by \(\mu_{CR}\) and \(\mu_F\), respectively. Initially, \(\mu_{CR} = 0.5\) and \(\mu_F = 0.5\) are used for new problems when there is no prior knowledge. Subsequently, they are updated based on their previous successful parameters during the evolution process.

At each generation, for each target vector, its crossover rate \(CR_i\) is independently generated as follows:

\[CR_i = \text{rnnd} (\mu_{CR}, 0.1)\]

(5)

and truncated to the interval \([0, 1]\), where \(\mu_{CR}\) is the mean value of the normal distribution used to generate \(CR_i\), with standard deviation 0.1. The \(\mu_{CR}\) is updated as follows:

\[\mu_{CR} = (1-c) \cdot \mu_{CR} + c \cdot \text{mean}_a (S_{CR})\]

(6)

where \(c\) is a constant in \([0, 1]\); \(\text{mean}_a(\cdot)\) is the usual arithmetic mean operation; and \(S_{CR}\) is the set of all successful crossover rates \(CR_i\) at generation \(g\).
In order to maintain some level of population diversity, for each target vector, the mutation factor \( F_i \) is independently calculated as:

\[
F_i = \text{rndc}(\mu_F, 0, 1)
\]

and then truncated to be 1.0 if \( F_i > 1.0 \) or regenerated if \( F_i \leq 0 \). \( \text{rndc}(\mu_F, 0, 1) \) is a random number generated according to the Cauchy distribution with location parameter \( \mu_F \) and scale parameter 0.1. The location parameter \( \mu_F \) is updated in the following manner:

\[
\mu_F = (1 - c) \cdot \mu_F + c \cdot \text{mean}_L(S_F)
\]

where \( S_F \) is the set of all successful mutation factors \( F_i \) at generation \( g \); and \( \text{mean}_L(\cdot) \) is the Lehmer mean:

\[
\text{mean}_L(S_F) = \frac{\sum_{i=1}^{\left| S_F \right|} F_i^2}{\sum_{i=1}^{\left| S_F \right|} F_i}
\]

3. Related Work

3.1. EA-based Parameter Extraction of FC Models

As shown in Equations (1) and (3), there are some unknown parameters in both PEMFC and SOFC models. For the control of high performance, these important unknown parameters need to be extracted and optimized. However, there are two challenges to extract the parameters [51, 4, 75]: i) there is no sufficient exact procedure for parameter extraction; and ii) it is a complex nonlinear and multi-variable problem. These challenges make the traditional optimization techniques difficult to solve the parameter extraction problems of FC models effectively. Therefore, the use of evolutionary algorithms for these problems has obtained considerable attention recently.

3.1.1. Parameter Extraction of PEMFC Model

In [49], Mo et al. presented a niche hybrid genetic algorithm (HGA) for parameter optimization of PEMFC model, where the niche techniques and Nelder-Mead’s simplex method are merged into GA. Outeiro et al. [51, 52] applied the simulated annealing (SA) as optimization technique to extract the parameters of PEMFC model. In [78], a particle swarm optimizer (PSO)-based parameter optimization technique of PEMFC model was presented according to the \( V-I \) characteristics. Ohenoja and Leiviskä [50] conducted comprehensively experiments to indicate how the parameter range, the validation strategy, and the selected algorithm influence on the performance of GAs in parameter optimization of PEMFC model. In [42], Li et al. firstly presented an effective informed adaptive PSO (EIA-PSO) to balance the global and local search. Then, EIA-PSO was employed for the PEMFC model parameter optimization. Askarzadeh and Rezazadeh [7] proposed a modified PSO (MPSO) to optimize the parameters of PEMFC model, where a modified method is presented for the PSO’s inertia weight in MPSO. In [6], an artificial bee swarm optimization algorithm is proposed for optimizing the parameters of a steady-state PEMFC stack model suitable for electrical engineering applications. In [5, 38, 8], the grouping-based global harmony search (HS), tournament selection based HS, and elite-based global HS were respectively presented for the PEMFC model parameter optimization. To optimize the PEMFC model parameters, Dai et al. [17] proposed a novel seeker optimization algorithm (SOA), which is based on the concept of simulating human searching behaviors. In [4], the artificial immune system (AIS)-based parameter extraction of PEMFC model was present, and its results is compared with those of GAs and PSO. Yang and Wang [76] proposed a novel bio-inspired P systems-based optimization algorithm (BPOA) to solve the PEMFC model parameter optimization problems. In [13], a DE variant, i.e., DEGL [18], was employed for the parameter optimization problem of PEMFC stack. Inspired by the mechanism of biological RNA, Zhang and Wang [81] presented an adaptive RNA GA (ARNA-GA) for estimating the PEMFC model parameters. In [9], Askarzadeh and Rezazadeh proposed a bird mating optimizer (BMO), which is inspired by the intelligent behavior of birds during mating season. In [25], the ranking-based DE [26] is employed for the parameter extraction of PEMFC model. Ensemble of different improvements of DE, Gong and Cai [27] developed the rank-MADE, where the multi-strategy adaptation and ranking-based vector selection techniques are synergized, to extract the parameters of PEMFC model. In [3], a backtracking search algorithm combined with Burger’s chaotic map was proposed to efficiently estimate the unknown parameters of the PEMFC electrochemical-based model.
3.1.2. Parameter Extraction of SOFC Model

In [73, 74], Wu et al. proposed a GA-RBF neural network method for modeling and predictive control of SOFC, where the genetic algorithm (GA) is used to optimize the parameters of RBF neural networks. Yang et al. [75] presented an improved GA (IGA) method for parameter optimization of tubular SOFC stack. In [75], the IGA method is used to optimize the parameters of a simple electrochemical model to fit the simulated data of the dynamic SOFC model. Li et al. proposed a model predictive control strategy based on GA for the SOFC control problem [43], where a support vector machine model is identified to approximate the behavior of the SOFC system and GA is used to solve the constrained predictive control problem. In [10], Bozorgmehri and Hamedi proposed an artificial neural network (ANN) and a GA-based method to model and optimize cell parameters to improve the performance of singular, intermediate-temperature SOFCs (IT-SOFCs). The ANN is used to model the SOFC performance through using experimental data, and GA is employed to optimize the SOFC parameters, i.e., anode support thickness, anode support porosity, electrolyte thickness, and functional layer cathode thickness [10]. In [63], GA is utilized to estimate the electrode microstructure distributions in NASA Bi-electrode supported SOFCs. Gong et al. [29] proposed an adaptive DE, in which the ranking-based vector selection and crossover rate repairing technique are combined to optimize the parameters of SOFC model.

3.2. EAs with Knowledge Reusage

As stated in [47], “Problems seldom exist in isolation”. If previous problem-solving knowledge can be reused to solve new similar problems, the performance of the optimization techniques may be enhanced. Recently, some researchers tried to improve the performance of EAs by reusing the knowledge. It can be briefly classified into two categories: i) EAs based on case-based reasoning (CBR), and ii) EAs based on transfer learning.

3.2.1. CBR-based EAs

In [46], Louis and Li presented a learning system that combines a genetic algorithm (GA) solver with a case-base of past problem-solving attempts to increase performance with experience when solving similar traveling salesman problems (TSPs). Borrowing ideas from CBR, Louis and McDonnel [47] proposed the case-injected genetic algorithms (CIGAR). In CIGAR, some appropriate intermediate solutions to similar previously solved problems are periodically injected into GA’s population. CIGAR is used to solve three combinational problems, i.e., combinational circuit design, asset allocation, and job shop scheduling problem (JSSP) [47]. Pérez et al. [57] presented a CBR scheme where the design patterns extracted from a GA are reused to reduce convergence times when optimizing combinational logic circuits at the gate level. In [35, 36], based on CBR an improved Bayesian optimization algorithm (BOA) is proposed for biasing BOA, where the previous solutions and the Bayesian network obtained by BOA are stored in the case, and then reused for new problem-solving. The proposed method is mainly used for solving the knapsack problem, TSP, and minimum spanning tree problem [35, 36].

3.2.2. Transfer Learning-based EAs

Besides the CBR-based EAs, very recently, transfer learning is combined with EAs to reuse previous problem-solving knowledge to improve EAs’ performance. Feng et al. developed a memetic computational paradigm for search [21, 22] that models how human solves problems. In the proposed approach [21, 22], the knowledge learned from previous problem-solving experiences can be transferred to enhance future evolutionary searches. Two challenging NP-hard routing problems, i.e., capacitated vehicle routing (CVR) and capacitated arc routing (CAR), are used to verify the performance the proposal [21, 22]. In [31, 56], to improve the performance of BOA, the probabilistic models previously obtained by BOA are reused and transferred across problems to speed up the solution of similar problems in the future. Pelikan and Hauschild [55] presented a framework to improve efficiency of model-directed optimization techniques by combining a distance metric with information mined from previous optimization runs on similar problems. In [31, 56, 55], different combinational optimization problems are used to evaluate the performance of the proposed methods. Santana et al. [60] proposed a framework for transfer learning between related optimization problems by means of structural transfer. Then, the proposed method is used for the multi-marker tagging single-nucleotide polymorphism selection problem [60]. Hacibeyoğlu et al. [30] combined transfer learning with artificial bee colony (ABC) algorithm for the numerical optimization problems. In [30], the obtained solutions of previously solved problems are transferred as some solutions in the initial population for solving similar shifted problems. In [33],
Iqbal et al. presented a genetic programming based learning classifier system, in which the useful building blocks are extracted from smaller problems and then they are reused to learn more complex, large-scale problems in the domain. The proposed method is evaluated on four different Boolean problem domains, i.e., multiplexer, majority-on, carry, and even-parity problems [33].

4. Our Proposal

In this section, we firstly introduce the motivations of this work, followed by the framework of the transfer learning based DE. Based on the proposed framework, an improved JADE method (i.e., transferred adaptive DE, TRADE, in short) is proposed to solve the parameter extraction problems of PEMFC and SOFC models.

4.1. Motivations

As reviewed in Section 3.1, in the literature, some researchers tried to reuse previous problem-solving knowledge to improve the performance of EAs when solving new similar problems. However, most of the work only considered to solve the combinational problems. Although Hacibeyoglu et al. [30] proposed the transfer learning based ABC algorithm for the numerical optimization problems, the previously obtained solutions only transferred to solve the same shifted functions. Therefore, the method proposed in [30] may not be used across different problems, especially when the decision variables of the problems have different physical meanings or different dimensions, such as the unknown parameters in PEMFC and SOFC models.

As described in Equations (2) and (4), the parameter extraction problems of PEMFC and SOFC models by the optimization techniques share the similar objective functions. In addition, for the same FC model, if the conditions of the model (such as temperature, hydrogen and oxygen partial pressures, etc.) are different, the extracted parameters are also different. However, for these problems they have the same objective functions. Thus, the algorithmic parameters and operators of the optimization techniques may be reused and transferred to solve the parameter extraction problems of PEMFC and SOFC models.

Since DE is among the most powerful optimization algorithms for numerical problems, based on the above considerations, we will try to combine transfer learning with DE to extract the unknown parameters of PEMFC and SOFC models, where the algorithmic parameters obtained from previous problem-solving experiences will be reused and transferred across different problems.

![Figure 1: The framework of adaptive DE based on transfer learning](image)

4.2. The Framework

The framework of the proposed adaptive differential evolution based on transfer learning is shown in Figure 1. Firstly, when a new problem in the problem pool will be optimized by the DE algorithm, DE inquires the parameter
database. If the database is empty, the default algorithmic parameters (such as $CR$ and $F$) given by the user will be set in DE to optimize the problem; otherwise, the $\text{Reuse()}$ procedure will be activated to reuse and transfer the parameters saved in the database as the initial parameters of DE. Then, DE optimizes the problem and updates its parameters according to a specific adaptation technique. When the stopping criterion is met, the $\text{Save()}$ procedure is activated to save the final parameters into the database.

As shown in Figure 1, the proposed framework has the following advantages:

- **Simplicity**: Compared with the original DE algorithm, only two additional procedures, i.e., $\text{Reuse()}$ and $\text{Save()}$, are used in the framework. These two procedures can be implemented by very simple techniques as shown in Section 4.3.

- **Generality**: The proposed framework is generic, it might be used for different adaptive DE variants, such as jDE [11], SaDE [59], JADE [80], and so on.

- **Interactivity**: In the frame work, the expert knowledge can be added into the parameter database. In this way, if we have prior knowledge about the problems, we can provide more accurate parameters for the problems and save them into the database to accelerate the optimization process.

### 4.3. TRADE: An Improved JADE Approach

Based on the proposed framework as shown in Figure 1, as an illustration, we propose an improved JADE method, referred to as TRADE. In TRADE, the parameter database contains the $\mu_{CR}$ and $\mu_{F}$ values obtained from previously solved problems. If the database is empty, both $\mu_{CR}$ and $\mu_{F}$ are initialized to be 0.5; otherwise, the $\text{Reuse()}$ procedure will reuse and transfer the parameters in the database to be the initial $\mu_{CR}$ and $\mu_{F}$.

#### 4.3.1. Parameter Saving: $\text{Save()}$

In TRADE, $\mu_{CR}$ and $\mu_{F}$ are saved to the parameter database. In this work, they are appended to the database when the following one of two conditions is satisfied: i) If the best solution in the current population is unchanged in the last continuous 500 generations; or ii) if the stopping criterion of the algorithm is met.

#### 4.3.2. Parameter Reusage: $\text{Reuse()}$

Suppose that there are $n$ items of $(\mu_{CR}, \mu_{F})$ in the parameter database, i.e., $|\mu_{CR}| = |\mu_{F}| = n$. In order to reuse these knowledge and transfer them to be the initial $\mu_{CR_0}$ and $\mu_{F_0}$ in TRADE to solve new encountered problems, in this work, we adopt the following simple techniques.

The initial $\mu_{CR_0}$ is calculated by

$$\mu_{CR_0} = \text{rndn}(\mu_{\overline{CR}}, \sigma_{\mu_{CR}})$$

and truncated to $[0, 1]$, where $\mu_{\overline{CR}}$ is the mean value of all $\mu_{CR}$ values in the database, $\sigma_{\mu_{CR}}$ is their standard deviation, and $\text{rndn}(\mu_{\overline{CR}}, \sigma_{\mu_{CR}})$ is a random number generator according to the normal distribution.

The initial $\mu_{F_0}$ is also calculated in the similar way:

$$\mu_{F_0} = \text{rndn}(\mu_{\overline{F}}, \sigma_{\mu_{F}})$$

and truncated to $[0, 1]$, where $\mu_{\overline{F}}$ is the mean value of all $\mu_{F}$ values in the database and $\sigma_{\mu_{F}}$ is their standard deviation.

In TRADE, when a new problem is encountered, the initial $\mu_{CR_0}$ and $\mu_{F_0}$ are obtained according to Equations (10) and (11), respectively. In this manner, the previous problem-solving knowledge can be reused to bias the values of $CR$ and $F$ in the beginning of the evolution process. Note that if $n = 1$ in the database, in this case, we set $\mu_{CR_0} = \mu_{CR}$ and $\mu_{F_0} = \mu_{F}$, where $\mu_{CR}$ and $\mu_{F}$ are the values saved in the database.

#### 4.3.3. The TRADE Method

The pseudo-code of TRADE is presented in Algorithm 1, where $Np$ is the population size, NFEs is the number of function evaluations, $\text{Max\_NFEs}$ is the maximal NFEs, $c$ and $p$ are two parameters in JADE [80]. Compared with JADE proposed in [80], the only differences in TRADE are the $\text{Reuse()}$ and $\text{Save()}$ procedures as shown in Algorithm 1. Therefore, the proposed TRADE method does not increase the overall complexity of JADE significantly. Although the modifications are minor between TRADE and JADE, TRADE can significantly improve the performance of JADE when solving the parameter extraction problems of PEMFC and SOFC models as revealed in Section 5.
Algorithm 1: The pseudo-code of the TRADE algorithm

**Input:** Control parameters: \(N_p\) and Max.\(_{NFEs}\)

**Output:** The best final solution

1. Set \(c = 0.1, p = 0.05\) as presented in [80];
2. Initialize the population randomly;
3. Calculate the objective function value of each solution in the population;
4. \(\text{NFEs} = N_p\);
5. **if** The parameter database is empty **then**
   6. Set \(\mu_{CR} = 0.5\) and \(\mu_{F} = 0.5\);
   **else**
   7. Activate the Reuse() procedure as shown in Section 4.3.2 to set the initial \(\mu_{CR}\) and \(\mu_{F}\);
8. **while** \(\text{NFEs} < \text{Max.}_{\text{NFEs}}\) **do**
   9. Sort the population from the best to the worst based on the objective function value of each solution;
   10. **for** \(i = 1\) to \(N_p\) **do** /* Calculate parameters */
       11. Generate \(CR_i\) and \(F_i\) with Equations (5) and (7), respectively;
   12. **for** \(i = 1\) to \(N_p\) **do** /* Generate the trial vector \(u_i\) */
       13. Produce the trial vector \(u_i\) with “DE/current-to-pbest/1/bin” strategy;
       14. Apply the boundary constraint-handling to the violated solution;
       15. Calculate the objective function value of the trial vector \(u_i\);
   16. **for** \(i = 1\) to \(N_p\) **do** /* Survival selection */
       17. if \(u_i\) is better than its parent \(x_i\) then
       18. \(x_i = u_i\);
       19. \(CR_i \rightarrow S_{CR}\);
       20. \(F_i \rightarrow S_{F}\);
       21. Update the \(\mu_{CR}\) and \(\mu_{F}\) with Equations (6) and (8), respectively;
   22. \(\text{NFEs} = \text{NFEs} + N_p\);
   23. Activate the Save() procedure as shown in Section 4.3.1 to save \(\mu_{CR}\) and \(\mu_{F}\) into the database;

5. Results and Analysis

In this section, to evaluate the performance of TRADE, it is applied to extract the unknown parameters of PEMFC and SOFC models as presented in Section 2.1. Totally, eighteen \(V-I\) datasets are used, containing 7 PEMFC datasets and 11 SOFC datasets, which will be described in Section 5.2. The results of TRADE are compared with those of 9 state-of-the-art EAs, including the real-coded genetic algorithm (rcGA) [32], fast evolutionary programming (FEP) [77], artificial bee colony (ABC) [37], comprehensive learning PSO (CLPSO) [44], jDE [11], SaDE [59], DEGL [18], CoDE [71], and JADE [80].

5.1. Parameter Settings

In this work, the parameter settings of the above-mentioned 10 algorithms are shown in Table 3, unless a changed is mentioned.

All algorithms are coded in standard C++. The maximal number of function evaluations (Max.\(_{NFEs}\)) are set to 30,000 and 150,000 for PEMFC and SOFC models, respectively. Since all of the ten algorithms are the stochastic algorithms, in order to make the comparison meaningful, each problem is optimized over 100 independent runs\(^1\).

\(^1\)Note that, in TRADE, for each problem only the parameters \(\mu_{CR}\) and \(\mu_{F}\) obtained in the first run are saved in the database.

9
Table 3: Parameter settings for the ten compared algorithms.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Parameter settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>rcGA [32]</td>
<td>(N_p = 100, \rho_c = 0.9, \rho_m = 0.15)</td>
</tr>
<tr>
<td>PEP [7]</td>
<td>(N_p = 100, q = 10)</td>
</tr>
<tr>
<td>ABC [37]</td>
<td>(N_p = 50, \mu = 300)</td>
</tr>
<tr>
<td>LHSQ [44]</td>
<td>(N_p = 100, m = k, c = 1.49445)</td>
</tr>
<tr>
<td>JDE [11]</td>
<td>(N_p = 50, r_1 = 0.1, r_2 = 0.1)</td>
</tr>
<tr>
<td>SaDE [59]</td>
<td>(N_p = 50, LP = 50)</td>
</tr>
<tr>
<td>DEGL [18]</td>
<td>(N_p = 10 \times D, Cr = 0.9, F = 0.8)</td>
</tr>
<tr>
<td>CoDE [71]</td>
<td>(N_p = 30)</td>
</tr>
<tr>
<td>JADE [80]</td>
<td>(N_p = 100) for PEMFC model; (N_p = 50) for SOFC model</td>
</tr>
<tr>
<td>TRADE</td>
<td>(N_p = 100) for PEMFC model; (N_p = 50) for SOFC model</td>
</tr>
</tbody>
</table>

Table 4: Parameter values and operation conditions of PEMFC and SOFC stacks.

<table>
<thead>
<tr>
<th>Item</th>
<th>Fuel Cell Name</th>
<th>(n_{cell})</th>
<th>(T) (K)</th>
<th>(P_{H_2}) (atm)</th>
<th>(P_{O_2}) (atm)</th>
<th>Number of Data Points</th>
</tr>
</thead>
<tbody>
<tr>
<td>P01</td>
<td>WNS-PEMFC [68]</td>
<td>48</td>
<td>313</td>
<td>1.5</td>
<td>1</td>
<td>144</td>
</tr>
<tr>
<td>P02</td>
<td>WNS-PEMFC [68]</td>
<td>48</td>
<td>333</td>
<td>1.5</td>
<td>1</td>
<td>147</td>
</tr>
<tr>
<td>P03</td>
<td>WNS-PEMFC [68]</td>
<td>48</td>
<td>353</td>
<td>1.5</td>
<td>1</td>
<td>148</td>
</tr>
<tr>
<td>P04</td>
<td>Ballard Mark V PEMFC [15]</td>
<td>1</td>
<td>343</td>
<td>1</td>
<td>1</td>
<td>14</td>
</tr>
<tr>
<td>P05</td>
<td>SR-12 PEM Generator [15]</td>
<td>48</td>
<td>323</td>
<td>1.74628</td>
<td>0.2095</td>
<td>37</td>
</tr>
<tr>
<td>P06</td>
<td>BCS 500-W PEMFC [15]</td>
<td>32</td>
<td>333</td>
<td>1</td>
<td>0.2095</td>
<td>7</td>
</tr>
<tr>
<td>P07</td>
<td>Temasek PEMFC [34]</td>
<td>20</td>
<td>323</td>
<td>0.5</td>
<td>0.5</td>
<td>50</td>
</tr>
<tr>
<td>P08</td>
<td>WN-SOFC [67]</td>
<td>96</td>
<td>1073</td>
<td>3</td>
<td>3</td>
<td>317</td>
</tr>
<tr>
<td>P09</td>
<td>WN-SOFC [67]</td>
<td>96</td>
<td>1173</td>
<td>3</td>
<td>3</td>
<td>317</td>
</tr>
<tr>
<td>P10</td>
<td>WN-SOFC [67]</td>
<td>96</td>
<td>1273</td>
<td>3</td>
<td>3</td>
<td>317</td>
</tr>
<tr>
<td>P11</td>
<td>ASC-SOFC [20]</td>
<td>1</td>
<td>875</td>
<td>NA</td>
<td>NA</td>
<td>9</td>
</tr>
<tr>
<td>P12</td>
<td>ASC-SOFC [20]</td>
<td>1</td>
<td>923</td>
<td>NA</td>
<td>NA</td>
<td>17</td>
</tr>
<tr>
<td>P13</td>
<td>ASC-SOFC [20]</td>
<td>1</td>
<td>923</td>
<td>NA</td>
<td>NA</td>
<td>21</td>
</tr>
<tr>
<td>P14</td>
<td>ASC-SOFC [20]</td>
<td>1</td>
<td>973</td>
<td>NA</td>
<td>NA</td>
<td>21</td>
</tr>
<tr>
<td>P15-P18</td>
<td>ASC-SOFC [20]</td>
<td>1</td>
<td>923</td>
<td>NA</td>
<td>NA</td>
<td>9</td>
</tr>
</tbody>
</table>

5.2. Data Description

To evaluate the performance of our proposal, seven \(V-I\) datasets obtained from different PEMFC models and eleven \(V-I\) datasets obtained from different SOFC models are used. The parameter values and operation conditions of PEMFC and SOFC stacks are briefly presented in Table 4. For the seven PEMFC datasets (P01 - P07), P01 - P03 are simulated data generated by the WNS-PEMFC MATLAB/SIMULINK [69], and the other 4 data are experimental data obtained from the literature. For the seven SOFC datasets (P08 - P11), there are three simulated data generated by WN-SOFC MATLAB/SIMULINK [69] (P08 - P10), and the rest 8 data are experimental data obtained from [20], which are generated by the Elcogen 10 \(\times\) 10 cm\(^2\) ASC-10B planar single cell under different operation conditions and/or parameter values.

![Figure 2: The fitting results of TRADE on the training data.](image-url)
5.3. Training of TRADE

When TRADE is used to solve the first problem, if there is no prior knowledge about the problem, the parameter database is empty. In this case, TRADE is the same as JADE [80], and \( \mu_{CR} = \mu_F = 0.5 \) is set. In this work, when TRADE is used to extract the parameters of PEMFC and SOFC models, TRADE can be trained by using the simple training data to get \( \mu_{CR} \) and \( \mu_F \). Afterwards, the saved parameters are able to be reused in the subsequent experiments. Based on this consideration, we adopt the PEMFC model shown in Equation (1) to generate the training data. The operation conditions are: \( T = 343 \text{ K}, P_{H_2} = 1 \text{ atm}, P_{O_2} = 1 \text{ atm}, \) and \( n_{cell} = 1 \). The input parameters \( x' = [0.85596174, 0.0026494826, 0.43591260] \) for running TRADE on this dataset, the extracted parameters are used to extract the parameters from the PEMFC model. The Max_NFEs is set to be 500, 000 to obtain better values of \( \mu_{CR} \) and \( \mu_F \). After running TRADE on this dataset, the extracted parameters \( x' = [0.85592364, 0.0334407883, 0.9671264] \) for TRADE fit the simple training data very well. Finally, the final \( \mu_{CR} \) and \( \mu_F \) are saved in the database. For the compared algorithms, the parameter settings are


5.4. Compared with Other EAs

In this section, TRADE is compared with other state-of-the-art EAs through the 18 datasets for the parameter extraction problems of PEMFC and SOFC models. For all of the compared algorithms, the parameter settings are shown in Table 3.

<table>
<thead>
<tr>
<th>ALGORITHM</th>
<th>P01</th>
<th>P02</th>
<th>P03</th>
<th>P04</th>
<th>P05</th>
<th>P06</th>
</tr>
</thead>
<tbody>
<tr>
<td>rCMA</td>
<td>1.38E-01 ± 7.8E-02</td>
<td>2.9E-02 ± 3.8E-02</td>
<td>2.3E-02 ± 3.9E-02</td>
<td>8.8E-05 ± 3.0E-05</td>
<td>0.37E-07 ± 1.1E-07</td>
<td>0.12E-01 ± 9.0E-04</td>
</tr>
<tr>
<td>FEP</td>
<td>1.06E-01 ± 9.7E-02</td>
<td>2.0E-01 ± 7.4E-02</td>
<td>3.8E-01 ± 5.1E-02</td>
<td>7.8E-05 ± 1.5E-05</td>
<td>0.37E-04 ± 1.6E-04</td>
<td>0.11E-02 ± 6.0E-02</td>
</tr>
<tr>
<td>CLPSO</td>
<td>2.6E-02 ± 1.9E-02</td>
<td>2.6E-02 ± 2.1E-02</td>
<td>3.2E-02 ± 2.4E-02</td>
<td>5.0E-05 ± 3.8E-05</td>
<td>0.13E-04 ± 3.1E-04</td>
<td>0.09E-02 ± 1.1E-02</td>
</tr>
<tr>
<td>JDE</td>
<td>1.08E-02 ± 5.8E-03</td>
<td>6.0E-03 ± 4.6E-03</td>
<td>7.2E-03 ± 1.2E-02</td>
<td>4.4E-05 ± 5.9E-05</td>
<td>0.10E-05 ± 6.2E-05</td>
<td>0.08E-05 ± 6.1E-05</td>
</tr>
<tr>
<td>SADE</td>
<td>5.9E-05 ± 1.2E-03</td>
<td>2.9E-05 ± 5.7E-05</td>
<td>1.5E-05 ± 3.4E-05</td>
<td>8.2E-06 ± 1.3E-04</td>
<td>0.08E-04 ± 1.3E-03</td>
<td>0.18E-05 ± 2.1E-05</td>
</tr>
<tr>
<td>DE-GL</td>
<td>1.34E-05 ± 1.3E-05</td>
<td>1.7E-05 ± 1.3E-05</td>
<td>5.8E-06 ± 4.8E-06</td>
<td>4.2E-05 ± 6.9E-05</td>
<td>0.07E-05 ± 2.0E-04</td>
<td>0.08E-05 ± 1.8E-05</td>
</tr>
<tr>
<td>CoDE</td>
<td>1.50E-04 ± 2.5E-04</td>
<td>9.1E-05 ± 1.8E-04</td>
<td>5.7E-05 ± 1.5E-04</td>
<td>4.3E-05 ± 2.4E-05</td>
<td>0.08E-06 ± 2.8E-05</td>
<td>0.08E-06 ± 6.0E-05</td>
</tr>
<tr>
<td>JADE</td>
<td>4.71E-04 ± 3.0E-03</td>
<td>4.3E-04 ± 3.6E-04</td>
<td>1.1E-04 ± 9.1E-05</td>
<td>4.2E-05 ± 2.8E-05</td>
<td>0.08E-05 ± 3.1E-05</td>
<td>0.08E-07 ± 9.0E-05</td>
</tr>
<tr>
<td>NCA</td>
<td>2.9E-04 ± 2.1E-04</td>
<td>6.4E-05 ± 8.8E-06</td>
<td>2.4E-05 ± 2.1E-11</td>
<td>8.0E-07 ± 7.0E-10</td>
<td>0.03E-06 ± 7.9E-07</td>
<td>0.01E-06 ± 5.9E-07</td>
</tr>
</tbody>
</table>

The fit-
5.4.1. On the Quality of Final Solutions

The quality of final solutions of all algorithms are reported in Table 5, where the overall best and the second best results are highlighted in **boldface** and *italic*, respectively. All results are averaged over 100 runs. In Table 5, the mean and standard deviation values are provided. To make the comparison statistically meaningful, the Wilcoxon test is used to compare TRADE with other EAs. In Table 5, “‡” indicates TRADE is significantly better than its competitor according to the Wilcoxon signed-rank test at $\alpha = 0.05$. Moreover, based on the mean values of all datasets, the Friedman test is used to calculate the average rankings of all algorithms. The results are given in Figure 3.

According to the results shown in Table 5 and Figure 3, it can be clearly observed that:

- TRADE consistently gets the best results in all problems in terms of the mean objective function values. It significantly outperforms other 9 algorithms based on the Wilcoxon test at $\alpha = 0.05$ in all problems. In addition, TRADE is the most robust algorithm according to the standard deviation values. Figure 3 also reveals that TRADE obtains the first average ranking among all algorithm by the Friedman test.

- Compared the results between TRADE and JADE, TRADE can improve the performance of JADE significantly in all problems. Even in the first dataset P01, there only contains one item of ($\mu_{ck}, \mu_F$) in the database, TRADE is still able to provide significantly better results than those of JADE. This phenomenon indicates the benefits of properly reusing previous parameters for extracting the parameters of FC models effectively.

It is worth pointing out that although the objective function values of SaDE, DEGL and JADE are very close to those of TRADE in some problems, however, since the simulated data and experimental data are generated from the different FC models, no information is available about the accurate values of the parameters of them; therefore, any reduction in the objective function value is significant because it results in improvement in the knowledge about the real values of the parameters.

5.4.2. On the Convergence Speed

In this section, the convergence curves of all algorithms are compared and plotted in Figure 4. For the sake of brevity, only some representative problems are chosen. From Figure 4, we can see that TRADE is able to consistently converge to the approximate optimal solutions in all problems. It obtains faster convergence speed than its predecessor JADE. In addition, compared with other EAs, TRADE also provides the fastest convergence speed in overall. By carefully looking at the results, we find that: i) For the PEMFC model, DEGL converges fastest in the beginning of the evolution process, however, it converges slowly after NFEs = 5,000. ii) For the SOFC model, both ABC and CLPSO converge faster than TRADE in early evolution process, however, they stagnate quickly and get the poor quality of the final solutions as shown in Table 5.

---

2The results of the Wilcoxon and Friedman tests are calculated by the KEEL software [1].
5.5. On the V-I Characteristics

In this work, TRADE is used to extract the unknown parameters of PEMFC and SOFC models. Therefore, to further evaluate the performance of TRADE, it is important to check the V-I characteristics obtained by TRADE. To achieve V-I characteristics, the optimal parameter values extracted by TRADE are fed back to the PEMFC and SOFC mathematical models as shown in Equations (1) and (3), respectively. Figure 5 provides comparisons between the data of different FC models and the data obtained by TRADE for all problems. As shown in Figure 5, it is clear that the V-I characteristics obtained by TRADE are highly coincide with the data of FC models in all cases. Thus, we can conclude that TRADE can be an effective and efficient alternative for the parameter extraction problems of PEMFC and SOFC models.

Figure 4: Comparison on the convergence speed among different EAs for the selected problems.
Figure 5: Comparisons between the data of different FC models and the data obtained by TRADE for all problems. (a): P01 - P03; (b)-(e): P04 - P07; (f): P08 - P10; (g): P11 - P14; (h): P15 - P18.
5.6. Compared with Reported Results

In the previous sections, TRADE is used to extract the unknown parameters for both PEMFC and SOFC models under different conditions. Additionally, TRADE is also compared with other advanced EAs directly. In this section, to further understand the performance of TRADE, it is compared with the reported results presented in [3], because both of them use the same PEMFC model with 11 unknown parameters. In [3], two datasets in Table 4 (i.e., P04 and P05) are used, and the extracted parameters by BMO, BSA, BSABCM-1, BSABCM-2, and BSABCM-3 are reported. The extracted parameters and their corresponding objective function values are described in Table 6 and Table 7, respectively. Note that, to make a fair comparison, the objective function values of different algorithms in [3] are re-calculated by using the parameters in Tables 6 and 7. The reason is that the data in P04 and P05 originates from the graphical diagrams in [49], this procedure may lead to some extra variability. The results in Tables 6 and 7 clearly reveal that TRADE is able to provide the best objective function values for both P04 and P05 compared with other five methods in [3].

6. Conclusions and Future Work

In general, no problems exist in isolation, which always share some similar features. Based on this consideration, in this paper, we propose a transferred adaptive differential evolution framework, where the previous problem-solving parameters of DE can be reused and transferred to solve new similar problems. According to the framework, we present an improved JADE method, in which the parameters \( \mu_{CR} \) and \( \mu_{F} \) are transferred across different problems. The modifications in TRADE are simple and minor compared with its predecessor JADE. Afterwards, the proposed TRADE method is used to solve the parameter extraction problems of two different fuel cell models, i.e., PEMFC and SOFC models. Eighteen \( V-I \) datasets generated by different PEMFC and SOFC models are used to evaluate the performance of TRADE, and the experimental results indicate that

- By transferring the previous problem-solving parameters saved in the database, TRADE yields significantly better results than JADE in terms of the quality of final solutions and the convergence speed in all problems.
- Compared with other state-of-the-art EAs, TRADE also provides very promising results. It is able to obtain the first average ranking among all compared methods.
- The V-I characteristics obtained by TRADE fit both the simulated data and experimental data pretty well in all cases. Therefore, it may be used to solve other complex optimization problems of fuel cell models.

Even with minor modifications on JADE, TRADE is capable of providing very promising results when solving similar problems. This motivates us to investigate more generic improved EAs based on transfer learning for the numerical optimization problems in the near future.

The source code of TRADE can be obtained from the first author upon request.

References


