Hindawi
Mathematical Problems in Engineering
Volume 2018, Article ID 4585403, 24 pages
https://doi.org/10.1155/2018/4585403

Research Article

A Phase-Based Adaptive Differential Evolution Algorithm for the Economic Load Dispatch Considering Valve-Point Effects and Transmission Losses

Xin Shen, Dexuan Zou, Xin Zhang, Qiang Zhang, and Peng Xiao

School of Electrical Engineering and Automation, Jiangsu Normal University, Xuzhou 221116, China

Correspondence should be addressed to Dexuan Zou; zoudexuan@163.com

Received 10 August 2018; Revised 22 October 2018; Accepted 29 October 2018; Published 15 November 2018

Academic Editor: Oliver Schütze

A phase-based adaptive differential evolution (PADE) algorithm is proposed to solve the economic load dispatch (ELD) considering valve-point effects (VPE) and transmission losses. To a great extent, PADE makes up for the drawbacks of the traditional differential evolution (DE) through three improvements. First, we establish an archive of storing successful individuals to improve the quality of offspring. Second, to balance the exploring and exploiting ability of the algorithm, a phase-based mutation operation is carried out. Third, two control parameters are adaptively adjusted, which is helpful for enhancing the robustness of the algorithm. In addition, two types of repair methods of constraint handling are employed for the ELD without or with transmission losses to help PADE find feasible solutions more efficiently. A performance comparison between PADE and other DE approaches from the literature was carried out on six ELD test cases which consider a set of operating constraints including the VPE and transmission losses. Results show a competitive PADE performance in all test cases regarding the compared DE approaches. Compared to methods from the literature, the costs obtained by PADE are lower in most cases while the corresponding constraint violations reach a lower level.

1. Introduction

In power systems, economic load dispatch (ELD) [1] is an important, complex optimization problem that allocates the required generation among the available generators. The main purpose of ELD is to find the optimal distribution of the generating units to make the fuel cost minimize while meeting all inequality and equality constraints in practice. Therefore, an accurate mathematical model is needed for adapting the ELD problems with different characteristics. So far, the quadratic function has been widely used as the classical mathematical model for solving the cost of the ELD problems. Moreover, addition conditions are often considered in the actual system, which constitute various versions for the ELD problems such as the valve-point effects (VPE) [2], transmission losses [3], and prohibited operating zones (POZs) [4].

Many classical methods, such as linear programming [5], quadratic programming [6], lambda iteration method [7], and gradient method [1], have been applied to solve the conventional ELD problems with the quadratic function. The above classical methods require that the generators exhibit convex and smooth characteristics. However, the input-output characteristics of generators are generally nonconvex and non-smooth in practical thermal generation plants. The reasons are caused by the VPE, POZs and so on, which may make the cost function produce a number of local optima. Therefore, these classical methods can hardly find the global optima for such types of problems. Dynamic programming [8] is different from the above classical methods. It can handle nonconvex and nonsmooth ELD problems, but it is restricted by the curse of dimensionality that inevitably leads to extra computational cost. Thus it is not also a suitable one for practical ELD problems.

In order to seek the optimal solution of ELD, more and more scholars tend to use modern intelligent algorithms. These modern intelligent algorithms have huge advantage over classical methods, especially for the ELD problems with various operation constraints. They are essentially approximation methods, which do not need the derivative

information of the ELD problems. The practical ELD problems are quite complex, so most of scholars are committed to continuously improving or hybrid modern intelligent algorithms and developing new constraint handling mechanisms at present. Qin and Cheng et al. [9] adopted an orthogonal designed method and proposed auxiliary vector generation based on multiple strategies to enhance the effectiveness of orthogonal designed operation. Based on the adaptive adjustment of acceleration coefficients, the algorithm's robustness and global search capability can be improved by employing a tent chaotic map. Furthermore, a repair method is designed to handle the practical constraints. Coelho and Mariani [10] put forward an improved harmony search (IHS) algorithm based on exponential distribution for the ELD problems. In IHS, a repair process is adopted to find feasible solutions. Al-Betar and Awadallah et al. [11] developed a modified version of harmony search algorithm that is called tournament-based harmony search (THS) algorithm, where the tournament selection process replaces the random selection process in the memory consideration operator. The tournament selection process is beneficial to improve the convergence performance of the algorithm. The introduced repair strategy can completely guarantee the feasibility of each generated solution before calculating the fuel cost. Therefore, to some extent, the THS algorithm can achieve better solutions compared to many state-of-the-art optimization approaches. Mandal and Roy et al. [12] proposed a new krill herd algorithm (KHA) to solve the ELD problems with various constraints. The crossover and mutation operations of DE are embedded into the KHA algorithm. The four versions of KHA are presented to solve different scale ELD problems. Zou and Li et al. [13] proposed a new global particle swarm optimization (NGPSO) algorithm for the economic emission dispatch with or without transmission losses. First, NGPSO devises a new position updating equation that utilizes the global best particle to guide the convergence of all particles. Second, the randomization based on the uniform distribution is used to avoid the premature convergence in the late optimization process. Simultaneously, different treatment methods are adopted for the equality and inequality constraints, respectively. For the inequality constraints, a common and simple penalty function is used. More importantly, the equality constraints are handled by three important steps, where the equality constraints are completely desirable. Different from the reported variable selection [14, 15], a variable is randomly selected to treat all variables equally in the repair process. As a result, a large number of high-quality feasible solutions can be obtained. Kumar and Chaturvedi [16] solved the ELD problems using a fuzzy particle swarm optimization (FPSO). The quadratic cost function with valve point loading and prohibited operating zone is studied in the models. The suitable value of inertia weight is yielded by a fuzzy logic controller (FLC). The FLC is composed of a knowledge base, a fuzzification interface, an inference system, and the defuzzification interface. As a result, FPSO efficiently avoids premature convergence. Mahdad and Srairi [17] adopted a hybrid method based on genetic algorithm (GA), differential evolution (DE), and pattern search (PS) [18], namely, GA-DE-PS. First, GA and DE are jointly used to find the optimal solutions. Second, the optimal solutions achieved by GA and DE are adjusted by PS. Third, the new solutions obtained by PS are employed to GA and DE for the new initial solutions of GA and DE. The interactive mechanism allows individuals to communicate with each other, which is very helpful to balance the exploiting and exploring capability of GA-DE-PS. Jeddi and Vahidinasab [19] proposed a modified harmony search algorithm (MHSA). In MHSA, a new improvising method based on wavelet mutation combined with a new memory consideration scheme based on the roulette wheel mechanism is adopted, which improves the accuracy, accelerates convergence speed, and enhances robustness of the classical HSA. For the constraint handling strategy, a common penalty function method is used to find feasible solutions. Singh and Dhillon [20] proposed an opposition-based greedy heuristic search (OGHS) to solve practical optimization problems. The initial population is composed of several randomly generated candidate individuals based on the uniform distribution, and its opposite population is formed by the opposite-based learning strategy [21]. Then the better half of the two populations is adopted to carry out subsequent optimization process, which improves the convergence speed of the algorithm. The opposition-based learning is also applied to its migration to maintain the diversity. The mutation strategy applied is based on the greediness and randomness to find the global best solution.

According to the review above, most works are focused on enhancing or hybridizing intelligent algorithms, and some of them studied the constraint handling improvement for practical ELD problems. The inequality constraints of ELD problems are relatively easy to meet. In contrast, the equality constraints of the ELD problems are hard to handle. Thus they mainly study the treatment method of the equality constraints for the ELD problems. However, their methods of treating the equality constraint with transmission losses are usually the same as those without transmission losses. In fact, the complexity of both equality constraints without transmission losses and equality constraints with transmission losses has a great difference. Therefore, the method of handling the equality constraint with transmission losses should be different from that of handling the equality constraint without transmission losses, which can differently repair the infeasible solutions. For our studies, the equality constraint refers to the power balance constraint. Meanwhile, in order to further improve the performance of the original DE algorithm and repair the infeasible solutions, the contributions of this paper contain two aspects. First, a phase-based adaptive differential evolution algorithm (PADE) is proposed to improve the performance of the original DE algorithm. Second, two repair methods are used to properly handle the equality constraints of two types of ELD problems, respectively.

The remainder of the paper is organized as follows: Section 2 describes the mathematical model of the ELD problems; Section 3 introduces the traditional differential evolution algorithm and the proposed DE variant; simulation results and the corresponding comprehensive analysis are presented in Section 4; Section 5 concludes the paper.

2. Economic Load Dispatch (ELD)

2.1. Objective Function. The objective of ELD is to minimize the total fuel cost of the entire power system under the condition of meeting various constraints. The cost of ELD can be described as follows:

$$\min f_c = \sum_{j=1}^{N_G} F_j\left(P_j\right) \tag{1}$$

where f_c represents the total fuel cost of the power generating (in \$/h). \$/h represents the total fuel cost of the power generating per hour. F_j represents the fuel cost of the generating unit j (in \$/h). The variable P_j represents the real output power of the generating unit j (in MW). MW means megawatt. N_G is the number of units in the power system.

In general, the traditional cost function of ELD is represented as the quadratic function, which can be given by Coelho and Mariani [10]:

$$F_j\left(P_j\right) = a_j {P_j}^2 + b_j P_j + c_j \tag{2}$$

where a_j, b_j, and c_j denote the cost coefficients of unit j. Under most circumstances, the valve-point effects (VPE) are contained in the practical power system. In view of this condition, the sinusoidal component is combined with the quadratic function, and the practical cost function can be amended as follows [10]:

$$F_j\left(P_j\right) = a_j {P_j}^2 + b_j P_j + c_j + \left|e_j \sin\left(f_j\left({P_j}^{\min} - P_j\right)\right)\right| \tag{3}$$

where e_j and f_j denote the cost coefficients of unit j considering VPE.

2.2. Constraints. The ELD problems are solved subject to some inequality and equality constraints, including power output limits, power balance, and prohibited operating zones (POZs).

2.2.1. Power Output Limits. The power output of each unit should be between its upper and lower bounds [10] in the power system.

$${P_j}^{\min} \le P_j \le {P_j}^{\max} \tag{4}$$

where ${P_j}^{\min}$ (in MW) and ${P_j}^{\max}$ (in MW) stand for the minimum and maximum power output of unit j, respectively.

2.2.2. Power Balance Constraints. The total generated power should be equal to the sum of the total load demand and transmission losses [10].

$$\sum_{j=1}^{N_G} P_j = P_D + P_L \tag{5}$$

where P_D denotes the total load demand of power system (in MW); P_L denotes the total transmission losses (in MW), and it can be commonly obtained by Kron's loss formula as follows [13]:

$$P_L = \sum_{j=1}^{N_G}\sum_{h=1}^{N_G} P_j B_{jh} P_h + \sum_{j=1}^{N_G} B_{0j} P_j + B_{00} \tag{6}$$

where B_{jh} denotes the transmission loss coefficient. Additionally, the simplified version of power balance constraint has been commonly studied by scholars. This version does not include the transmission loss P_L, namely, $P_L = 0$. Furthermore, it can be expressed by

$$\sum_{j=1}^{N_G} P_j = P_D \tag{7}$$

2.2.3. Prohibited Operating Zones (POZs). Under certain circumstances, a generating unit may not operate within certain ranges known as prohibited operating zones due to physical operation limitations. As a result, the feasible operation zones of the generating unit may be discontinuous. At this time, the feasible power output ranges of the generating unit can be properly described as follows [4]:

$$P_j \in \begin{cases} {P_j}^{\min} \le P_j \le P_{j,1}^{l} \\ P_{j,k-1}^{u} \le P_j \le P_{j,k}^{l} & k = 2, 3, \cdots, NPZ_j \\ P_{j,NPZ_j}^{u} \le P_j \le {P_j}^{\max} \end{cases} \tag{8}$$

where $P_{j,k}^{u}$ (in MW) and $P_{j,k}^{l}$ (in MW) denote the lower and upper bounds of prohibited operating zone k of unit j, respectively. NPZ_j is the number of prohibited operating zones of unit j.

2.3. Constraint Handling Mechanism

2.3.1. Inequality Constraint Handling. The inequality constraints (4) are easy to meet, because the optimization variables are directly set within their ranges in the initialization process, and any variable beyond their bound can be limited to the bound which it exceeds for the proposed algorithm. For the inequality constraints (8), a common penalty function approach is used to handle it. If the output power of the generating unit j lies in the POZs, the new cost function of the ELD problems with the POZs is expressed by

$$f_c = \sum_{j=1}^{N_G} F_j\left(P_j\right) + \delta \times \min\left(P_j - P_{j,k-1}^{l}, P_{j,k-1}^{u} - P_j\right) \tag{9}$$

where δ is a penalty factor. $\min(P_j - P_{j,k-1}^{l}, P_{j,k-1}^{u} - P_j)$ denotes constraint violations when the unit j operates in the POZs.

2.3.2. Equality Constraint Handling for the ELD Problems without Transmission Losses. Contrary to inequality constraints, the equality constraints are hard to satisfy. The penalty function method can be also used to help intelligent algorithms to find the optimization solutions, but the

quality of solutions cannot be ensured. Aim at the equality constraints, scholars proposed diverse repair methods for the ELD problems with or without transmission losses [9, 10, 13]. Due to the difference between the ELD problems without transmission losses and the ELD problems with transmission losses, two proposed repair methods will be, respectively, introduced. A repair method of this paper for the ELD problems without transmission losses is first introduced in this section.

To obtain more promising feasible solutions, a repair method and penalty function method are used to together cope with the equality constraint, and the equality constraint handling process is clearly shown in Algorithm 1.

Δj represents the adjustment limit of the component j. $|w|$ represents the equality constraint violation. sign is called sign function. When w is positive, the sign(w) value is equal to 1. On the contrary, when w is negative, the sign(w) value is equal to -1. The repair method will select a random component x_i^j from the individual vector x_i ($x_i = [x_i^1, x_i^2, \cdots, x_i^{N_G}], i = 1, 2, \cdots NP$), and let $J = \{1, 2, \cdots, N_G\}$. If the selected component x_i^j is equal to the upper bound $P_j^{\max}$ (or lower bound $P_j^{\min}$) when the total generating power output is smaller (larger) than demand, this component will be removed from the set J. Then a new component will be randomly selected from the set J that has excluded the previous component (s). This process will be repeated unless a component that is not equal to its upper (or lower) bound is found. The way of selecting a component can avoid some ineffective operations which do not repair those infeasible solutions. The smaller one in $|w|$ and Δj is selected to adjust the output power of the unit, which is to avoid the violation of inequality constraints after executing the repair process. Similar to the reported literature [10], a penalty method is still used to further handle equality constraint. Furthermore, a sign *flag* is set to judge whether the equality constraint has been completely met after the repair operation. If the equality constraint violations are surely equal to zero, the penalty method will not be used to further handle the equality constraint. On the contrary, if the equality constraint violations are not equal to zero, the penalty method will be used to further handle the remaining constraint violations. The inequality constraints related to the POZs are also handled by the penalty method. Because the environment of all experiments is based on MATLAB, MATLAB precision has a certain effect on the optimization results. The advantage of the above setting *flag* is that it can decrease the effects of MATLAB precision on the optimization results as much as possible. If another programming language is used to implement the proposed mechanism, the *flag* variable can be also used to eliminate the error due to the other operating environments precision. Thus, the *flag* variable cannot be ignored when other programming languages are used.

2.3.3. Equality Constraint Handling for the ELD Problems with Transmission Losses. Qin and Cheng et al. [9] presented a repair method that can reduce the equality constraint violations until these violations are within the error tolerance ε ($\varepsilon = 0.001$), which means that the repaired solutions are considered to be the feasible ones. A component j is randomly selected from the set J. x_i^j, J, P_j, $|w|$, and sign represent the concept same as the aforementioned repair method. rand represents a randomly generated number in the interval $[0, 1]$. The method also compares the $|w|$ value with the Δj value to select a smaller one that is used to update solutions. If the $|w|$ value is larger (smaller) than the Δj value, x_i^j will be subtracted to sign($\mathbf{w}$) $\times \Delta j$ (or w). Then the transmission losses and equality constraint violations are recalculated. This above procedure is repeated until the $|w|$ value is smaller than the error tolerance ε. The proposed repair method is similar to the repair method of Qin and Cheng et al. [9]. The only difference is that Δj is calculated based on the adjustment limit on the component j, and the repair method for the ELD problems with transmission losses is shown in Algorithm 2. This modification contributes to reduce more constraint violations and obtains the better solutions satisfying the tolerance error compared to the original mechanism. It will be indicated from Section 4.8 that the modification is effective.

3. Differential Evolution Algorithm and Its Improved Version

3.1. Differential Evolution Algorithm. Differential evolution (DE) algorithm, presented by Storn and Price [22], is a very competitive intelligent algorithm. It has simple structure and powerful capacity of searching solution space. Therefore, DE has been frequently applied to many fields for solving those real-word problems, such as optimal reactive power dispatch [23], parameter estimation problems in computational systems biology [24], iris recognition [25], and constraint optimization [26]. In this section, the traditional DE algorithm and its improved version will be introduced in detail.

The operation steps of DE are initialization, mutation, crossover and selection, respectively. In addition, a few parameters also play a key role on the DE's performance. These steps and parameters are explained as follows.

3.1.1. Initialization. The goal of this step is to generate NP candidate solutions consisting of a population. All candidate solutions are randomly generated within the lower and upper bounds of searching space. For example, the jth ($j = 1, 2, \cdots, N_G$) component of the ith ($i = 1, 2, \cdots, NP$) candidate solution is initialized by

$$x_i^j(0) = L^j + \text{rand} \times \left(U^j - L^j\right) \tag{10}$$

where L^j and U^j denote the lower and upper bounds of the jth component, respectively. $x_i^j(0)$ denotes the jth variable of the ith individual at the initial generation.

3.1.2. Mutation Operation. In this step, Mutant vector $v_i(t)$ is generated based on several randomly selected individuals for each parent individual vector $x_i(t)$ in the current generation t. DE/rand/1 is the most common and popular mutation strategy to generate mutant vector according to the following

Process: the equality constraint handling for the ELD problems without transmission losses
(1) Initialize a set $J = \{1, 2, \cdots, N_G\}$; Set $w = \sum_{j=1}^{N_G} P_j - P_D$;
(2) If $w < 0$
(3) Randomly select a component j from the set J;
(4) While $x_i^j = P_j^{\max}$
(5) Remove j from J, and let the new set be J'. Then randomly select a new component j' from J', and $j \longleftarrow j'$, $J \longleftarrow J'$;
(6) End While
(7) Calculate the adjustment limit of the component j, namely $\Delta j = P_j^{\max} - x_i^j$;
(8) Else If $w > 0$
(9) Randomly select a component j from the set J;
(10) While $x_i^j = P_j^{\min}$
(11) Remove j from J, and let the new set be J'. Then randomly select a new component j' from J', and $j \longleftarrow j'$, $J \longleftarrow J'$;
(12) End While
(13) Calculate the adjustment limit of the component j, namely $\Delta j = x_i^j - P_j^{\min}$;
(14) End If
(15) Set $flag = 0$;
(16) If $|w| > \Delta j$
(17) $x_i^j = x_i^j - \text{sign}(w) \times \Delta j$;
(18) $flag = 1$;
(19) Else If $|w| < \Delta j$
(20) $x_i^j = x_i^j - w$;
(21) End If
(22) If flag = 1
(23) A penalty function method is further used to handle equality constraint. The new objective function is represented as the formula $f_c = \sum_{j=1}^{N_G} F_j(P_j) + \delta \times |w|$ (or $f_c = \sum_{j=1}^{N_G} F_j(P_j) + \delta \times (|w| + \min(P_j - P_{j,k-1}^l, P_{j,k-1}^u - P_j))$);
(24) Else
(25) The equality constraint is completely met. At this time, the objective function is represented as the formula $f_c = \sum_{j=1}^{N_G} F_j(P_j)$ (or $f_c = \sum_{j=1}^{N_G} F_j(P_j) + \delta \times \min(P_j - P_{j,k-1}^l, P_{j,k-1}^u - P_j)$);
(26) End If

Algorithm 1: The pseudo code of the equality constraint handling process for the ELD problems without transmission losses.

Process: the equality constraint handling for the ELD problems with transmission losses
(1) Initialize a set $J = \{1, 2, \cdots, N_G\}$; Set $w = \sum_{j=1}^{N_G} P_j - P_D - P_L$ and the error tolerance ε;
(2) While $|w| > \varepsilon$
(3) Randomly select a component j from the set J;
(4) If $w < 0$
(5) $\Delta j = P_j^{\max} - x_i^j$;
(6) Else If $w > 0$
(7) $\Delta j = x_i^j - P_j^{\min}$;
(8) End If
(9) If $|w| > \Delta j$
(10) $x_i^j = x_i^j - \text{sign}(w) \times \Delta j$;
(11) Else If $|w| < \Delta j$
(12) $x_i^j = x_i^j - w$;
(13) End If
(14) Recalculate the transmission losses P_L and equality constraint violations $|w|$;
(15) End While;

Algorithm 2: The pseudo code of the equality constraint handling process for the ELD problems with transmission losses.

formula. More than common and popular, and it is the standard or canonical mutation operator in DE.

$$v_i(t) = x_{i3}(t) + F \times (x_{i1}(t) - x_{i2}(t)) \tag{11}$$

where $i1$, $i2$, and $i3$ are three randomly selected integers in $[1, NP]$, and $i1 \neq i2 \neq i3 \neq i$. The control parameter F represents the scale factor that is used to scale the difference vector $(x_{i1}(t) - x_{i2}(t))$. The scale factor F is a user-defined parameter ($F > 0$) [27] and usually set in the interval $(0, 1]$. Except for this mutation strategy DE/rand/1, another five popular mutation strategies are listed below [28]:

① DE/best/1

$$v_i(t) = x_{best}(t) + F \times (x_{i1}(t) - x_{i2}(t)) \quad i1 \neq i2 \neq best,\ i1 \neq i2 \neq i \tag{12}$$

② DE/rand/2

$$v_i(t) = x_{i1}(t) + F \times (x_{i2}(t) - x_{i3}(t)) + F \times (x_{i4}(t) - x_{i5}(t)) \quad i1 \neq i2 \neq i3 \neq i4 \neq i5 \neq i \tag{13}$$

③ DE/best/2

$$v_i(t) = x_{best}(t) + F \times (x_{i1}(t) - x_{i2}(t)) + F \times (x_{i3}(t) - x_{i4}(t)) \quad i1 \neq i2 \neq i3 \neq i4 \neq best,\ i1 \neq i2 \neq i3 \neq i4 \neq i \tag{14}$$

④ DE/current-to-best/1

$$v_i(t) = x_i(t) + F \times (x_{best}(t) - x_i(t)) + F \times (x_{i1}(t) - x_{i2}(t)) \quad i1 \neq i2 \neq best,\ i1 \neq i2 \neq i \tag{15}$$

⑤ DE/rand-to-best/1

$$v_i(t) = x_{i1}(t) + F \times (x_{best}(t) - x_{i1}(t)) + F \times (x_{i2}(t) - x_{i3}(t)) \quad i1 \neq i2 \neq i3 \neq best,\ i1 \neq i2 \neq i3 \neq i \tag{16}$$

where $x_{best}(t)$ represents the best individual vector at generation t.

3.1.3. Crossover Operation. Crossover has two types of strategies including binary crossover and exponential crossover. In this work, the binary crossover is adopted to obtain the trail vector $u_i(t)$ $(u_i(t) = (u_i^1(t), u_i^2(t), \cdots, u_i^{N_G}(t)))$ whose components are got from those of both parent vector $x_i(t)$ and mutant vector $v_i(t)$ based on the crossover rate. This operation is expressed by

$$u_i^j(t) = \begin{cases} v_i^j(t), & \text{If rand} < CR \text{ or } j = j_{N_G} \\ x_i^j(t), & \text{Otherwise} \end{cases} \tag{17}$$

where another important control parameter CR stands for the crossover rate which determines the probability of selecting components of mutant vector $v_i(t)$ $(i = 1, 2, \cdots, NP)$. CR is a user-defined parameter set in the interval $[0, 1]$. j_{N_G} is a randomly generated integer between 1 and N_G.

3.1.4. Selection Operation. The behavior of this operation is similar to a greedy strategy, and it only reserves better individuals into the next generation. For DE, if the updated trail vector $u_i(t)$ is better than parent vector $x_i(t)$, the former will replace latter into the next generation. Otherwise, the corresponding offspring individual vector is the same as the parent individual vector, which means that the updating of the individual is not successful. For a minimization problem, the offspring individual $x_i(t+1)$ $(i = 1, 2, \cdots, NP)$ is generated by

$$x_i(t+1) = \begin{cases} u_i(t), & \text{If } f(u_i(t)) \leq f(x_i(t)) \\ x_i(t), & \text{Otherwise} \end{cases} \tag{18}$$

where $f(\cdot)$ stands for the fitness function, which is to calculate the fitness value of an individual. In this problem, the smaller the fitness value of individual is, the better the performance of individual is.

3.2. A Phase-Based Adaptive Differential Evolution Algorithm. A phase-based adaptive differential evolution algorithm (PADE) is proposed to enhance the DE's ability in finding the global best solution. PADE and DE are different in three aspects.

3.2.1. An Archive of Storing Successful Individuals. In the DE's mutation strategy, the parent individual is usually randomly selected from the current population. This parent-selecting way is beneficial for maintaining the population diversity, but it may not jump out of local optima when stagnation is happening to DE. Based on this issue, Guo and Yang et al. [29] established an archive of recently updated individuals to improve the parent-selecting way by utilizing the successful individuals. If the current parent individual fails to update beyond the stagnation tolerance on the number of consecutive unsuccessfully updates, the parent individuals of mutation strategy will be randomly selected from the archive instead of the current population. When the archive exceeds its size, the oldest individual is removed. On the contrary, Tanabe and Fukunaga [30] adopted an external archive that stores the failed solutions at each generation to maintain the population diversity. Inspired by the study, we also set up an archive S that stores the recently updated individuals. Different from the archive of [29, 30], the randomly selected individuals from the archive are used for the base vector of mutation strategy in the current generation. Then the recently updated individuals are stored to S that has been set to empty after each generation. In other words, S only stores the recently updated individuals. As long as the recently updated individual exists, the base vector of each generation is randomly chosen from the archive in the early phase. In addition, if there are not updated individuals in a generation,

the base vector of mutation strategy will be randomly selected from the current population. The new parent-selecting way has a greater chance to generate successful individuals, and maintains the population diversity. For the initialization of S, it is first initiated to be a copy of the initial population. Additionally, in case the initial population contains infeasible solutions, they will be also used for the mutation operation in the early evolutionary phase, because those infeasible solutions are gradually evolved to feasible ones as the evolution progresses.

3.2.2. A Phase-Based Mutation Operation. In DE, mutation is an important step for obtaining the mutant individuals in the area around the base individual. In this area, the base individual locates in the center, the difference vector acts as the movement direction and step, and the scale factor scales the movement step. In fact, there are many different types of the base individuals and difference vectors, which consist of several mutation strategies of different characteristics, such as DE/rand/1, DE/rand/2, DE/best/1, DE/best/2, and DE/current-to-best/1. According to [31], DE/rand /1 and DE/rand/2 are very helpful to maintain the diversity of population but they perform poor in convergence rate and are easy to gradually approach the local optima. On the contrary, it is known from [31] that DE/best/1 and DE/best/2 are beneficial to accelerate population convergence, because the base individual is the global best individual with good information, which can guide other individuals to approach it. However, it is obvious that they are very easy to fall into the local optima. Specially, it can lead to premature convergence when they are used in the early phase of evolution. DE/current-to-best/1 [32] employs the current individual as the origin individual and the best individual as the guiding individual, which constructs a mixed base individual. DE/current-to-best/1 is called composite mutation strategies, which has better diversity than DE/best/1, but its global exploration is poor. In the entire evolutionary process, global exploration and local exploitation are two vital aspects, which should cooperate well with each other to find the promising solutions. First, the algorithm should widely search in the global area and find some local areas with promising individuals in the search space. Then, the algorithm should focus on local exploitation in these local areas. It is to enhance the ability of finding the optimal solution as fast as possible. In DE, the population individuals spread over the searching space in the early phase of evolution. They can update themselves based on other individuals through global search but it can also cause premature convergence. In the late phase of evolution, most individuals gather in a relatively small area, and they implement local exploitation instead of global exploration.

Based on the above analysis, Miranda-Varela and Mezura-Montes [33] relied on the number of feasible solutions to select one between DE/rand/1 and DE/best/1 to balance the exploration and exploitation ability. Qin and Huang et al. [31] constructed a mutation strategy candidate pool that contains several mutation strategies with diverse characteristics. In the evolution process, one strategy will be selected from the candidate pool according to a probability learned from its previous experience of generating promising solutions for each parent individual. The more successfully one strategy behaved in previous generations, the more probably it will be selected in the current generation. Different from the above studies, to match different phases of evolution, DE/srand/1 and DE/ best/1 are respectively used to execute the mutation operation in the early and late phase of evolution in PADE. For DE/srand/1, its base vector is randomly selected from the archive S, and its difference vector is the same as DE/rand/1. Therefore, DE/srand/1 is more easy to find the successful individuals than DE/rand/1. Now, the key task is how to select a generation index threshold between 1 and T to divide the entire evolution process into the early and late phase. Many scholars tend to set a probability rule based on the number of iterations [34, 35] or the fitness value of random individual [36] to determine whether the current evolution lies in the early phase or late phase of iteration. Different from their methods, Tang and Dong et al. [37] introduced the success ratio SR and an auxiliary generation index threshold G_T to help select the generation index threshold g_t. The threshold g_t is set based on the recently updated individuals, which is more heuristic for enhancing the performance of DE than the above two methods. In view of the advantage of the success ratio, this paper employs it to divide the evolution process into the early and late phase. The success ratio of each generation t is defined by:

$$SR_t = \frac{NS_t}{NP} \tag{19}$$

where NS_t is the number of the updated individuals at generation t. SR_t stands for the success ratio at generation t. In general, SR_t is a relative large value in the early phase of evolution. As the generation advances, SR_t is prone to decrease continually, so it can be used to determine whether the late phase begins. When SR_t is smaller than a threshold of success ratio SR_K for a given number of consecutive generations $N_{succ}^{\max}$, the late phase of the algorithm begins. If SR_t is larger than SR_K at generation t, the current number of consecutive generations is denoted as $N_{succ} = 0$. Otherwise, $N_{succ} = N_{succ} + 1$. If N_{succ} reaches $N_{succ}^{\max}$, N_{succ} keeps the fixed value $N_{succ}^{\max}$ until the evolution ends. That means that PADE only uses the mutation strategy DE/best/1 in the late phase. For most problems, SR_t can decrease to SR_K for consecutive generations $N_{succ}^{\max}$ in the early phase of evolution. However, it should be noted that SR_t cannot decrease to SR_K for consecutive generations $N_{succ}^{\max}$ in the early or entire evolution phase for a few problems. Therefore, under these circumstances, a threshold t_{force} is forced to set to divide the process into two phases. The threshold t' determined by success ratio is given by

$$t' = \left\{ t \mid SR_t < SR_K,\ N_{succ} = N_{succ}^{\max},\ \forall t \in [1, T] \right\} \tag{20}$$

In order to comprehensively describe the phase-based mutation operation producing the mutant vector, the following formula is given by

$$v_i(t) = \begin{cases} x_{best}(t) + F \times (x_{i1}(t) - x_{i2}(t)), & \text{if } t > t' \text{ or } t > t_{force} \\ x_{srand}(t) + F \times (x_{i1}(t) - x_{i2}(t)), & \text{Otherwise} \end{cases} \quad (21)$$

where the index of successful individual *srand* is randomly integer between 1 and the size of the archive S, and $srand \neq i1 \neq i2 \neq i$.

To visualize this phase-based mutation operation from an overall perspective, its schematic diagram is presented as follows (see Figure 1).

In Figure 1, the entire evolution phase is divided into the early and late phase. There are two situations on determining the threshold, which are represented as S1 and S2, respectively. Specifically, above the 0 to T axis, the way of dividing the entire evolution phase is represented as S1. Below the 0 to T axis, the way of dividing the entire evolution phase is represented as S2. The threshold t' can lie in $[t_{force}, T]$, or it does not exist in the $[0, T]$, where the t_{force} is used as the threshold at this time just as S2.

3.2.3. Adaptive Control Parameters. The two control parameters, scale factor F and crossover rate CR, play a significant role on the performance of DE. Their values are traditionally fixed as constants to participate in evolution. A suitable F and CR are related to a specific problem, and all individuals share their fixed control parameters in the evolution process for the original DE. The result is that the robustness of the algorithm is poor, and a lot of tedious optimization trials must be done to find a suitable F and CR for practical problems. To enhance the robustness of the algorithm and avoid tedious optimization trials, many researchers adopted different self-adaptive strategies to make control parameters self-adaptively change with individual evolution [36, 38]. Some of them randomly take a value within the range of parameters owned by the individual when the individual doesn't survive into the next generation. The other do not have the guidance rule at all, and the control parameters are changed based on a probability in the search process. In addition, there are some works [39, 40] only for this type of problems, where self-adaptive approaches have proven to be competent in different problems, for example, constraint optimization problem and biobjective tooth profile spur gear optimization problem. Qin and Huang et al. [31] proposed that the scale factor values are randomly sampled from a normal distribution and applied to each parent individual in the current population, which is helpful for the global and local of population. An adaptation scheme is used to generate the crossover rate and is based on the previous experiences of generating promising individuals. However, these self-adaptive strategies cannot well guide the population evolution for this type of ELD problems. It is clearly different that the two adaptive parameters with the learning ability are used to evolve with individuals in this paper. Because the best individuals usually own best control parameters in the improved population, control parameters of other individuals are required to learn from those of the best individuals, which is beneficial to find the global best individual. Same as other self-adaptive strategies, two control parameters are first encoded as shown in Figure 2.

Every individual vector $x_i(t)$ owns two control parameters $F_i(t)$ and $CR_i(t)$. In the initial process, $F_i(t)$ and $CR_i(t)$ are randomly generated in a given range $[F_{\min}, F_{\max}]$ and $[CR_{\min}, CR_{\max}]$, respectively. Since the population is not updated during the first generation, the two control parameters of the failed to update individuals are regenerated, which is the same as their initialization process. In the next evolution, if the current population has been improved in the tth generation (marked as $fail = 0$), two control parameters of the failed to update individuals will learn from those of the best individual. Otherwise (marked as $fail = 1$), their control parameters are randomly generated in their ranges. If the current individual is successfully improved at generation t, its corresponding parameters keep the previous value. That means that control parameters of the current individual are helpful to improve individual. Based on the above explanation, adaptive parameters with learning ability are described by

$$F_i(t+1) = \begin{cases} F_i(t) + \text{rand} \times (F_{best}(t) - F_i(t)), & \text{If } f(u_i(t)) > f(x_i(t)) \text{ and } t > 1 \text{ and fail} = 0 \\ F_{\min} + \text{rand} \times (F_{\max} - F_{\min}), & \text{If } f(u_i(t)) > f(x_i(t)) \text{ and } (t \leq 1 \text{ or fail} = 1) \\ F_i(t), & \text{Otherwise} \end{cases} \quad (22)$$

$$CR_i(t+1) = \begin{cases} CR_i(t) + \text{rand} \times (CR_{best}(t) - CR_i(t)), & \text{If } f(u_i(t)) > f(x_i(t)) \text{ and } t > 1 \text{ and fail} = 0 \\ CR_{\min} + \text{rand} \times (CR_{\max} - CR_{\min}), & \text{If } f(u_i(t)) > f(x_i(t)) \text{ and } (t \leq 1 \text{ or fail} = 1) \\ CR_i(t), & \text{Otherwise} \end{cases} \quad (23)$$

where $F_i(t)$ and $CR_i(t)$ are the scale factor and crossover rate of the current ith individual at generation t. $F_i(t+1)$ and $CR_i(t+1)$ are the scale factor and crossover rate of the ith individual at generation $t+1$. $F_{best}(t)$ and $CR_{best}(t)$ are the best control parameters owned by the best individual in the current population, $F_{\min}$ and $F_{\max}$ denote the minimum and maximum scale factor, respectively. $CR_{\min}$ and $CR_{\max}$ denote the minimum and maximum crossover rate, respectively. When control parameters of the current individual learn from the best control parameters, its learning step is a

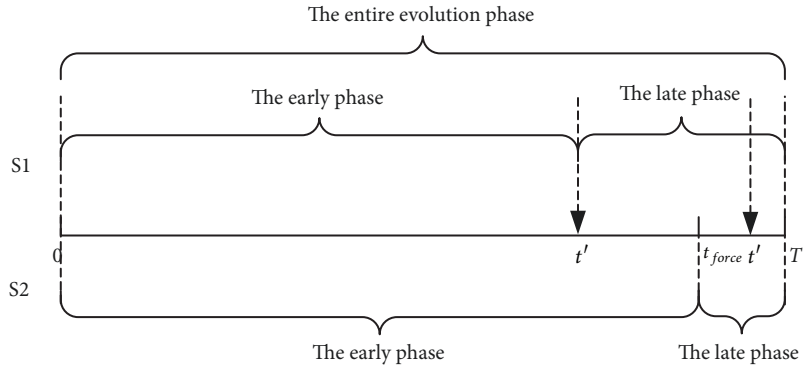


Figure 1: Schematic diagram of dividing the evolution phase.

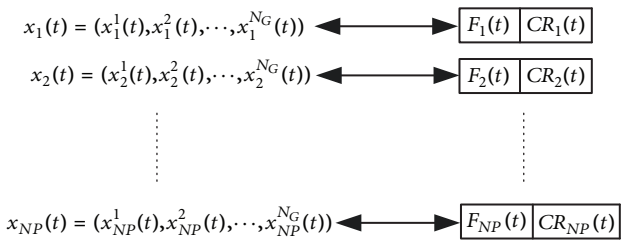


Figure 2: Description of encoding control parameters.

randomly generated number in the interval $[0, 1]$. In other words, control parameters of the current individual move to those of the current best individual by moving one step at a time.

The above way of updating control parameters imitates human learning. It solves the problems of high computing costs produced by extra optimization trails and enhances the robustness of PADE.

3.3. The Pseudo Code of the Proposed Algorithm. To fully demonstrate the computational steps of PADE, the algorithmic pseudo code is presented in Algorithm 3.

4. Simulation Results and Analysis

4.1. Experimental Cases and Parameter Settings. In this section, there are six ELD cases selected to verify the superiority of PADE compared to the other five DE variants and methods from the literature, and the parameters of experiment are set in advance. To be more specific, these cases include (1) the 6-unit system (PD = 1263MW) [41]; (2) the 6-unit system with transmission losses (PD = 1200MW) [3]; (3) the 10-unit system with VPE and transmission losses (PD = 2000MW) [14]; (4) the 14-unit system with transmission losses (PD = 950MW) [42]; (5) the 40-unit system with VPE (PD = 10500MW) [2]; (6) the 40-unit system with VPE and POZs (PD = 10500MW) [4]. Cases 2-4 consider the transmission losses of power system, and case 3 also considers VPE. Meanwhile, the cases with VPE also include the last two cases, and the last case takes the POZs into consideration. The detailed information of these cases can be obtained from the corresponding literature mentioned above [2–4, 14, 41, 42], and the prohibited operation zones are taken from Table 12 of [4].

The PADE algorithm is compared with other five state-of-the-art DE variants including differential evolution algorithm with self-adapting control parameters (jDE) [43], differential evolution algorithm with controlled search direction and self-adapting control parameters (jdDE) [44], modified differential evolution algorithm (mDE) [45], differential evolution with composite trial vector generation strategies and control parameters (CODE) [46], and improving differential evolution with a successful-parent-selecting framework (SPS-DE) [29]. The parameters of the six DE variants are set as follows: for jDE, $F_l = 0.1$, $F_u = 0.9$, and $\tau_1 = \tau_2 = 0.1$. The scale factor and crossover rate are, respectively, set to 0.6 and 0.3 in the initial generation, which cannot be obtained from [43]. For jdDE, the values of F_l, F_u, τ_1, τ_2, and the initial control parameters are the same as the above corresponding values in jDE. $CR_l = 0.1$ and $CR_u = 1.0$. For mDE, the fixed crossover rate is taken as $CR = 0.8$. For CODE, except for the maximum function evaluation number MAX_FE [47, 48], the other parameters are obtained by [46]. For SPS-DE, the stagnation tolerance Q is set to 32, and the SPS framework is incorporated with the standard DE with the strategy DE/rand/1/bin, scale factor $F = 0.6$ and crossover rate $CR = 0.3$. For the proposed algorithm, $F_{\min} = 0.1$, $F_{\max} = 0.9$, $SR_K = 0.1$, $t_{force} = 90\% \times T$, and $N_{succ}^{\max} = \lceil 3e^{N_G} \rceil$. $CR_{\min} = 0.1$ and $CR_{\max} = 0.3$ are set for the last two cases, and the crossover rate ranges between 0.3 and 0.6 for the remaining cases. It should be noted that these ELD cases can find the most promising solutions by our parameter settings. For all simulation experiments, the population size is set to 40, and they have the same MAX_FE when all six DE variants optimize the same ELD case, which make the results comparison fair. In addition, the maximum generation number T of jDE, jdDE, mDE, SPS-DE, and PADE are same for a ELD case. The maximum generation number T is equal to 1000, 300, 300, 300, 4000, and 4000 for cases 1-6, respectively. The MAX_FE is equal to 40000, 12000, 12000, 12000, 160000, and 160000 for cases 1-6, respectively. When the penalty method is used to handle constraints, the penalty factor δ is set to 10^{20}. Additionally, Matlab8.3 simulation software is used to execute all experiments under the environment of an Intel (R) Core (TM) i5-2450M CPU @

Algorithm: phase-based adaptive differential evolution

(1) Initialize a population pop_0;
(2) Set population size NP and the dimension of the individual N_G; the maximum generation number T; the maximum function evaluation number MAX_FE; the initial individual control parameters and their bounds; $N_{succ}^{\max}$; SR_K; an archive S and $S = \text{pop}_0$; $t = 0$; $N_{succ} = 0$; the cases including the 6-unit system (PD = 1263MW) [41], the 6-unit system with transmission losses (PD = 1200MW) [3], the 10-unit system with VPE and transmission losses (PD = 2000MW) [14], the 14-unit system with transmission losses (PD = 950MW) [42], the 40-unit system with VPE (PD = 10500MW) [2], the 40-unit system with VPE and POZs (PD = 10500MW) [4];
(3) While t doesn't reach T
(4) For $t = 1$ to T
(5) Set $NS_t = 0$; Obtain the global best solution $x_{best}(t)$;
(6) For $i = 1$ to NP
(7) If $N_{succ} < N_{succ}^{\max}$
(8) If $t < 90\% \times T$
(9) If S is empty
(10) Randomly generate three integers $i1$, $i2$, and $i3$ in the range $[1, NP]$, and $i1 \neq i2 \neq i3 \neq i$;
(11) $v_i(t) = x_{i3}(t) + F_i(t) \times (x_{i1}(t) - x_{i2}(t))$;
(12) Else
(13) Randomly generate two integers $i1$ and $i2$ in the range $[1, NP]$, and randomly generate a integer $srand$ between 1 and the size of the archive S, and $i1 \neq i2 \neq srand \neq i$;
(14) $v_i(t) = x_{srand}(t) + F_i(t) \times (x_{i1}(t) - x_{i2}(t))$;
(15) End If
(16) Else
(17) Randomly generate two integers $i1$ and $i2$ in the range $[1, NP]$, and $i1 \neq i2 \neq best$;
(18) $v_i(t) = x_{best}(t) + F_i(t) \times (x_{i1}(t) - x_{i2}(t))$;
(19) End If
(20) Else
(21) Randomly generate two integers $i1$ and $i2$ in the range $[1, NP]$, and $i1 \neq i2 \neq best$;
(22) $v_i(t) = x_{best}(t) + F_i(t) \times (x_{i1}(t) - x_{i2}(t))$;
(23) End If
(24) Randomly generate a integer j_{N_G} in the range $[1, N_G]$;
(25) For $j = 1$ to N_G
(26) If rand $< CR_i(t)$ or $j = j_{N_G}$
(27) $u_i^j(t) = v_i^j(t)$;
(28) Else
(29) $u_i^j(t) = x_i^j(t)$;
(30) End If
(31) End For
(32) If $f(u_i(t)) > f(x_i(t))$
(33) $x_i(t+1) = x_i(t)$;
(34) If $t > 1$
(35) If $fail = 0$
(36) $F_i(t+1) = F_i(t) + \text{rand} \times (F_{best}(t) - F_i(t))$;
(37) $CR_i(t+1) = CR_i(t) + \text{rand} \times (CR_{best}(t) - CR_i(t))$;
(38) Else
(39) $F_i(t+1) = F_{\min} + \text{rand} \times (F_{\max} - F_{\min})$;
(40) $CR_i(t+1) = CR_{\min} + \text{rand} \times (CR_{\max} - CR_{\min})$;
(41) End If
(42) Else
(43) $F_i(t+1) = F_{\min} + \text{rand} \times (F_{\max} - F_{\min})$;
(44) $CR_i(t+1) = CR_{\min} + \text{rand} \times (CR_{\max} - CR_{\min})$;
(45) End If
(46) Else
(47) $x_i(t+1) = u_i(t)$; $NS_t = NS_t + 1$;

Algorithm 3: Continued.

```
(48)                Store the updated individuals into S at generation t for the next
                    generation, and S only reserves the updated individuals in the
                    current population;
(49)              End If
(50)            End For
(51)            If N_succ < N_succ^max
(52)              If SR_t < SR_K
(53)                N_succ = N_succ + 1;
(54)              Else
(55)                N_succ = 0;
(56)              End If
(57)            End If
(58)            If the current population is improved
(59)              Update the global best solution;
(60)              fail = 0;
(61)            Else
(62)              fail = 1;
(63)            End If
(64)          End For
(65) End While
```

Algorithm 3: The pseudo code phase-based adaptive differential evolution.

2.50GHZ. The simulation results of 30 independent runs can be obtained in Table 1.

4.2. Comparison among Six DE Variants for Six ELD Problems. In order to improve the quality of solutions, an improved DE and one of the proposed repair methods are combined to find the best feasible solution for a specific case. In this section, all six DE variants are combined with the corresponding one of the proposed two repair methods that are different for the ELD problems without transmission losses and the ELD problems with transmission losses. Thus the performance of the six DEs is only compared in this section while the quality of repair methods will be discussed in Section 4.8.

In Table 1, several criteria are adopted to compare the performance of six improved DEs, including "$Cost_{\min}$", "$Cost_{\max}$", "$Cost_{\text{mean}}$", "$Cost_{\text{median}}$" and "$Cost_{\text{std}}$", which stand for the minimal, maximal, mean, median and standard deviation values of 30 test results, respectively. For case 1, all six DE variants are able to exactly find the global best solution, and all six DE variants have low $Cost_{\text{std}}$ values, indicating that their stability is strong. For case 2, the minimal cost obtained by the first three comparison algorithms are the same as that of PADE, but the performance of PADE is better than that of the three algorithms according to the $Cost_{\max}$ and $Cost_{\text{std}}$. The $Cost_{\min}$ values of the remaining two comparison algorithms are slightly larger than that of PADE. In terms of the $Cost_{\text{mean}}$, the results of PADE and jdDE are both equal to the minimal cost function value, and they are smaller than those of the other four algorithms. The $Cost_{\text{median}}$ values of PADE, jDE, and jdDE are also exactly equal to the minimal cost function value, which are slightly smaller than those of mDE, CODE, and SPS-DE. Overall, the $Cost_{\max}$, $Cost_{\text{mean}}$, and $Cost_{\text{median}}$ values of PADE are equal to its $Cost_{\min}$ value, and its $Cost_{\text{std}}$ value is close to 0. It means that PADE is easy to find the smallest objective function value in each run and has strong stability for this case. For case 3, both jdDE and PADE can find the same lowest cost compared to the other four DEs. Furthermore, the $Cost_{\max}$, $Cost_{\text{mean}}$, $Cost_{\text{median}}$, and $Cost_{\text{std}}$ values of PADE are smaller than those of jdDE. Thus PADE is superior to jdDE in stability and success rate. All the results obtained by jDE, mDE, CODE, and SPS-DE are worse than those of the other two competitors, and mDE performs the worst among all methods in terms of the $Cost_{\text{std}}$, explaining that the stability of mDE is poor. With regard to case 4, PADE can obtain the best $Cost_{\min}$, $Cost_{\text{mean}}$ and $Cost_{\text{median}}$ compared to the other approaches. The $Cost_{\min}$ value of mDE is slightly larger than that of PADE, and it is acceptable. However, unfortunately, the $Cost_{\max}$, $Cost_{\text{mean}}$, and $Cost_{\text{std}}$ values of mDE are far larger than those of another five approaches. Therefore, although mDE has the strong global search capability close to PADE, it performs poor in stability. The $Cost_{\max}$ and $Cost_{\text{std}}$ values of PADE are slightly larger than those of jDE, and they are smaller than those of the other four DEs. Therefore, the results of PADE are acceptable. jDE has the advantage over another four comparison algorithms in the average and median cost function values. On the contrary, jDE has poor performance according to the $Cost_{\min}$. In addition, CODE has the worst global search ability among all six algorithms, but its $Cost_{\text{std}}$ value is acceptable. It means that CODE has a great chance to get trapped into a local optimum over 30 runs. For case 5, PADE has absolute advantage over another five algorithms, because it is able to find the best $Cost_{\min}$, $Cost_{\max}$, $Cost_{\text{mean}}$, $Cost_{\text{median}}$, and $Cost_{\text{std}}$ values, and these values are much smaller than those of another five algorithms. It suggests that PADE performs the best among all six DEs on solving the case 5. For case 6, although system considers POZs, PADE still has great advantage over five competitors. The $Cost_{\min}$, $Cost_{\text{mean}}$, $Cost_{\text{median}}$, and $Cost_{\text{std}}$ values of PADE are better than those of another five competitors, which exhibits that the performance of PADE is very competitive. In fact, the ELD

Table 1: Comparison among six DEs for six ELD problems.

Problems	Algorithms	$Cost_{min}$	$Cost_{max}$	$Cost_{mean}$	$Cost_{median}$	$Cost_{std}$	
Case 1	jDE	15275.930392	15275.930392	15275.930392	15275.930392	5.550256e-12	=
	jdDE	15275.930392	15275.930392	15275.930392	15275.930392	7.181256e-12	=
	mDE	15275.930392	15275.930392	15275.930392	15275.930392	5.889358e-12	=
	CODE	15275.930392	15275.930392	15275.930392	15275.930392	6.098738e-12	=
	SPS-DE	15275.930392	15275.930392	15275.930392	15275.930392	5.340737e-12	=
	PADE	15275.930392	15275.930392	15275.930392	15275.930392	5.393880e-12	
Case 2	jDE	63975.710808	63975.710809	63975.710809	63975.710808	8.430184e-08	=
	jdDE	63975.710808	63975.710809	63975.710808	63975.710808	2.404592e-08	=
	mDE	63975.710808	63977.716408	63975.813336	63975.710809	0.408772	=
	CODE	63975.710809	63975.710840	63975.710817	63975.710813	8.690063e-06	-
	SPS-DE	63975.710813	63975.710871	63975.710835	63975.710834	1.629320e-05	-
	PADE	63975.710808	63975.710808	63975.710808	63975.710808	8.884254e-10	
Case 3	jDE	111497.562764	111497.563832	111497.563072	111497.563001	2.525372e-04	-
	jdDE	111497.562722	111497.563672	111497.563008	111497.562846	3.283995e-04	=
	mDE	111497.562754	111498.661457	111497.671245	111497.564231	0.273042	-
	CODE	111497.563377	111497.567630	111497.565501	111497.565494	9.863784e-04	-
	SPS-DE	111497.563050	111497.565322	111497.563878	111497.563853	5.537232e-04	-
	PADE	111497.562722	111497.563048	111497.562806	111497.562770	9.790660e-05	
Case 4	jDE	4303.508000	4303.508294	4303.508114	4303.508106	7.295542e-05	=
	jdDE	4303.508127	4303.509646	4303.508746	4303.508633	4.133750e-04	-
	mDE	4303.507990	4303.563362	4303.511529	4303.508774	0.010089	-
	CODE	4303.508406	4303.509948	4303.508994	4303.508917	3.330271e-04	-
	SPS-DE	4303.508027	4303.508544	4303.508367	4303.508367	1.199291e-04	-
	PADE	4303.507984	4303.508296	4303.508087	4303.508077	7.348985e-05	
Case 5	jDE	121424.035506	121517.821187	121466.393369	121469.287345	24.244774	-
	jdDE	121593.929338	123673.006419	122266.957996	122223.070157	436.000403	-
	mDE	121702.930425	126064.743299	124177.591269	124341.751739	1125.943225	-
	CODE	121535.443649	121649.481025	121605.792248	121608.120277	22.505147	-
	SPS-DE	121421.716683	121477.084260	121450.197647	121455.512445	21.844891	-
	PADE	121412.535537	121502.821508	121432.136531	121422.575850	20.625706	
Case 6	jDE	121420.895349	121517.821335	121464.474008	121461.674586	25.603325	-
	jdDE	121848.603334	125385.575206	123123.258233	123068.367672	696.421257	-
	mDE	123430.135957	127104.791027	125580.434851	125773.717464	1057.220836	-
	CODE	121547.806145	121669.397233	121616.312325	121623.185234	24.438200	-
	SPS-DE	121422.655205	121476.857993	121446.309372	121436.437028	21.111940	-
	PADE	121412.535602	121502.821508	121433.128567	121421.600343	20.061615	

problems with POZs are more complex than those without POZs. However, it is clearly seen that the $Cost_{min}$, $Cost_{max}$, $Cost_{mean}$, $Cost_{median}$, and $Cost_{std}$ obtained by PADE for case 6 are slightly smaller than, slightly larger than, or equal to, those of PADE for case 5. Therefore, the proposed algorithm still has strong search ability of solution space when the ELD problems become more complex. Additionally, the $Cost_{min}$, $Cost_{mean}$, and $Cost_{median}$ obtained by jDE for case 6 are even smaller than those of it for case 5. It explains that jDE also has strong search ability for the complex ELD problems. Contrary to PADE and jDE, the results of jdDE and mDE on solving the case 6 are far larger than those of them on solving the case 5 except the $Cost_{std}$ value of mDE. Thus the POZs of the ELD problems make it more difficult to find the feasible solutions for jdDE and mDE. In addition, the results of CODE and SPS-DE are worse than those of PADE except the $Cost_{max}$ value of SPS-DE for case 6. In short, jDE has good performance, but PADE performs the best among them.

In order to obtain the statistical significant difference when the proposed algorithm is compared to those comparison algorithms, Wilcoxon rank-sum test at the 0.05 significance level [51] is used in this section. The marks "+," "=," and "-" represent that the performance of PADE is better than, equal to, and worse than that of the corresponding comparison algorithms, respectively. It should be emphasized that Wilcoxon rank-sum test can only obtain whether the two algorithms have the difference. In other words, the better one cannot be still determined when the two algorithms have the

significant difference. If PADE and one of its competitors do not have significant difference, they are marked as "=." Based on the Wilcoxon rank-sum test characteristic, when PADE and any of the other five DEs have the significant difference, the criterion $Cost_{\min}$ is used to determine whether PADE is statistically better than the other algorithm. The better one is marked as "+," and the worse one is marked as "-."

The results of Wilcoxon rank-sum test statistical analysis are listed in the last column of Table 1. For case 1, all five comparison groups do not have the significant difference, suggesting that the performance of comparison algorithms is equal to that of PADE. The conclusion can be also reflected from the corresponding $Cost_{\min}$, $Cost_{\max}$, $Cost_{\text{mean}}$, $Cost_{\text{median}}$, and $Cost_{\text{std}}$ values. For case 2, there is no difference between PADE and each of jDE, jdDE and mDE. Statistically, the performance of CODE and SPS-DE is worse than that of PADE. For case 3, PADE significantly outperforms jDE, mDE, CODE, and SPS-DE. For case 4, PADE has no difference with jDE. In contrast, PADE has significant difference with jdDE, mDE, CODE, and SPS-DE, and PADE beats them in comparison on this problem. For case 5, all competitors are completely beat by PADE. For case 6, although the problem becomes complicated, PADE is still the best algorithm compared to the other DE variants. In sum, PADE is statistically no worse than the other five algorithms in all cases. In addition, PADE significantly beats all competitors for cases 5-6. Therefore, PADE can efficiently solve the six ELD problems.

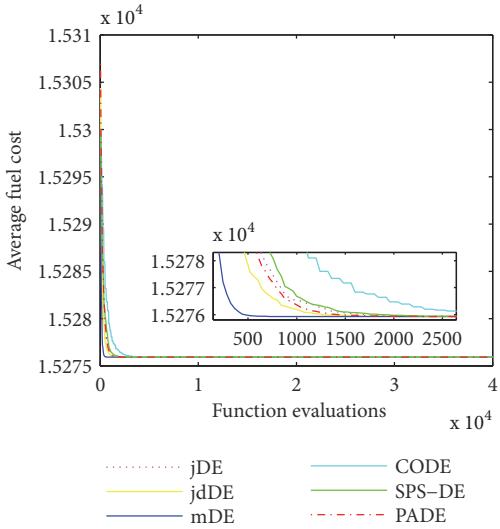
4.3. Convergence of Six DEs for The Fuel Cost Minimization. Under the condition that the proposed repair methods have been combined with six DEs to easily meet all the requirements of different constraints for the ELD problems, we now devote to study convergence of six DEs for the fuel cost minimization. In order to visualize the convergence characteristics of six DEs, in Figures 3(a)–3(f), we show the average convergence curves of six DEs for the fuel cost minimization of the selected six ELD problems over 30 runs, and these curves are based on the new objective function where the fitness function and penalty function are added for all ELD problems. The abscissa represents the function evaluations, and the ordinate represents the average fuel cost.

From Figures 3(a)–3(f), we can find that the convergence speed of PADE isn't the fastest among all six DEs for all ELD cases, but the average curves of PADE can gradually become better than, or as good as those of the other five DEs in all cases as the evolution progresses. For the first case, the convergence speed of PADE ranks third among all six DEs. In Figure 3(b), mDE, CODE and SPS-DE do not converge to the feasible zone while the other three DEs can find the feasible solutions, and the average curve of PADE is the lowest among three visible curves. In Figure 3(c), the average curve of CODE is the highest among all six DEs, because it does not lie in the feasible range. On the contrary, PADE performs the best according to the average convergence curve. In Figure 3(d), it can be clearly seen that the average curve of PADE reaches a lower level compared to those of the other five DEs. For the first four figures, the feasible solutions are only contained, because their constraint violations descend rapidly. For cases 5-6, the average curves of all six DEs descend very fast at the beginning of evolution. To be more specific, some infeasible solutions contain in the figures for the last two cases. Their constraint violations cannot descend rapidly. Thus, the new objective function values are very large at the beginning of evolution. The declining speed of constraint violations can be seen from the upper bound of ordinate. If constraint violations exist, the penalty function value combining the penalty factor (10^{20}) multiplies and the constraint violations will be very large. However, they can gradually find the feasible solutions. Furthermore, the solutions obtained by PADE are better than those obtained by the other five DEs in complex ELD cases.

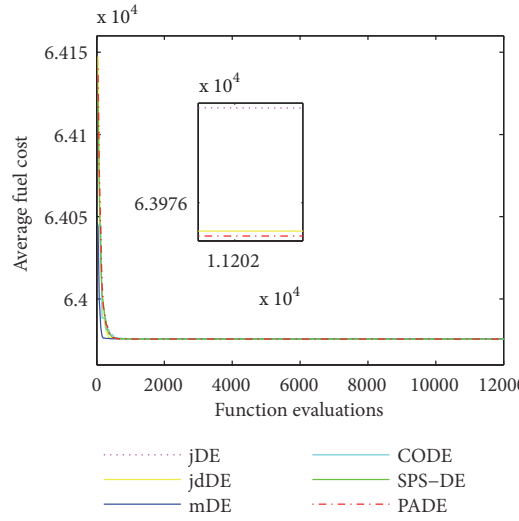
4.4. The Influence of the Introduced Archive. In this paper, the introduced archive S that stores the recently updated individuals is used to generate the promising feasible solutions with a higher probability in the early phase of evolution. To show the effect of the archive S, PADE without archive is compared with PADE with archive, where the archive component of the algorithm is eliminated to test its contributions in improving the quality of solutions. The two PADE algorithms adopt the same proposed repair methods. Furthermore, three criteria are used to compare the performance of the two PADE versions, and they are $Cost_{\text{mean}}$ and $Cost_{\text{std}}$ of 30 test results and the total constraint violation V_t of the best solution in 30 test results, respectively. The V_t value contains the violations of all the equality and inequality constraints for a solution. The experimental parameters are the same as those of Section 4.2. The comparison results are listed in Table 2.

According to Table 2, the $Cost_{\text{mean}}$ obtained by PADE with archive is the same as that of PADE without archive for cases 1-2, and their V_t values are equal to 0 for case 1. The $Cost_{\text{std}}$ value of PADE with archive is slightly smaller than that of PADE without archive for the first two cases, meaning that the archive slightly enhances the stability in PADE. For cases 3-4, the $Cost_{\text{mean}}$ and $Cost_{\text{std}}$ values of PADE with archive are slightly smaller than those of PADE without archive, which indicates that PADE with archive performs more stable than PADE without archive. With regard to the last two cases, the archive improves the quality of solutions more obvious compared to the first four cases. To be more specific, the $Cost_{\text{mean}}$ and $Cost_{\text{std}}$ values of PADE with archive are far smaller than those of PADE without archive for cases 5-6. For cases 1-4, the performance of the two PADE versions is similar in the constraint violation. PADE with archive achieves a lower level than PADE without archive for cases 5-6 in the constraint violation, which suggests that the archive motivates PADE to find a more satisfactory solution. In all, PADE with archive performs more outstanding than PADE without archive.

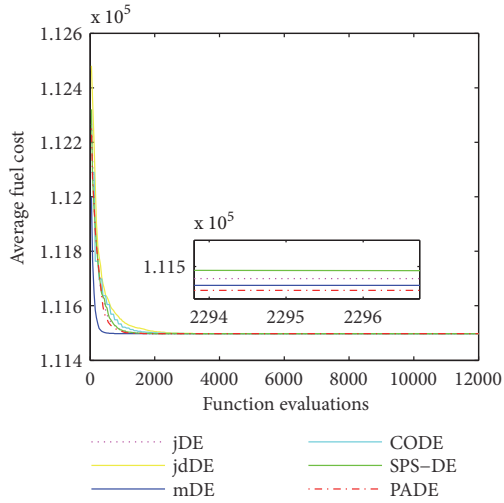
4.5. The Influence of the PADE Parameters on the Solution Quality and Stability. In order to investigate the influence of success ratio SR_K on the solution quality and stability, the SR_K value is set to 0.1, 0.3, 0.5, 0.7, and 0.9, respectively. The other experimental parameters are the same as those of Section 4.2.



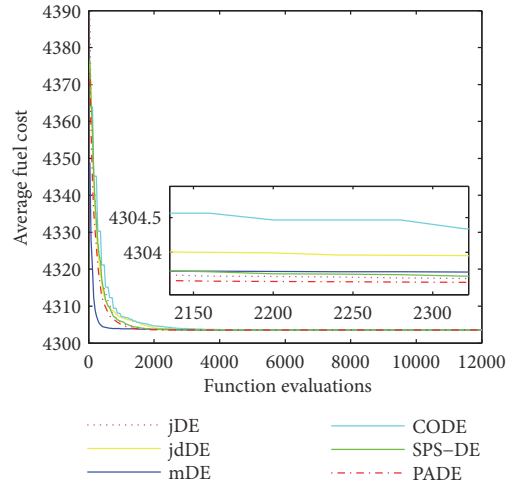
(a) The 6-unit system (PD = 1263MW)



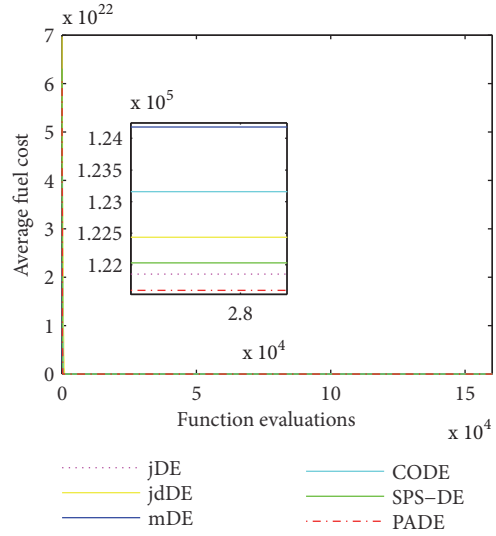
(b) The 6-unit system with transmission losses (PD = 1200MW)



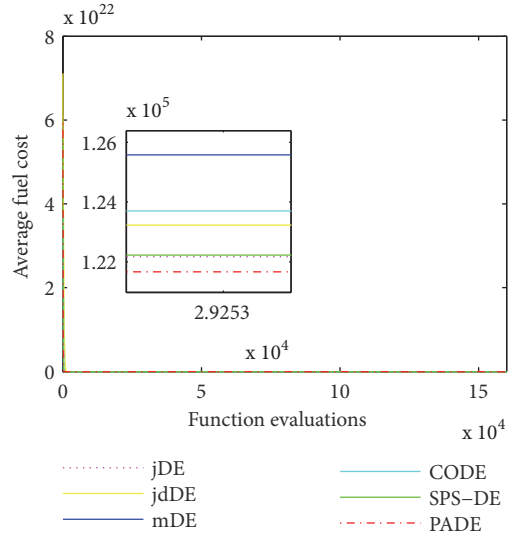
(c) The 10-unit system with VPE and transmission losses (PD = 2000MW)



(d) The 14-unit system with transmission losses (PD = 950MW)



(e) The 40-unit system with VPE (PD = 10500MW)



(f) The 40-unit system with VPE and POZs (PD = 10500MW)

Figure 3: Average convergence curves of six DEs for the fuel cost minimization of the selected six cases.

Table 2: Comparison between PADE without and with archive for six ELD problems.

Problems	Algorithms	$Cost_{\text{mean}}$	$Cost_{\text{std}}$	V_t
Case 1	PADE without archive	15275.930392	5.446504e-12	0
	PADE with archive	15275.930392	5.393880e-12	0
Case 2	PADE without archive	63975.710808	3.532562e-09	9.996231e-04
	PADE with archive	63975.710808	8.884254e-10	9.99999957e-04
Case 3	PADE without archive	111497.562836	1.110871e-04	9.999860e-04
	PADE with archive	111497.562806	9.790660e-05	9.999978e-04
Case 4	PADE without archive	4303.508088	1.010396e-04	9.977972e-04
	PADE with archive	4303.508087	7.348985e-05	9.989949e-04
Case 5	PADE without archive	121537.056249	76.502930	3.999998e-06
	PADE with archive	121432.136531	20.625706	4e-06
Case 6	PADE without archive	121567.242007	95.884391	9.999985e-07
	PADE with archive	121433.128567	20.061615	1.82e-12

Table 3: The influence of SR_K on the mean and standard deviation values of 30 results for six ELD problems.

Problems	Terms	$SR_K = 0.1$	$SR_K = 0.3$	$SR_K = 0.5$	$SR_K = 0.7$	$SR_K = 0.9$
Case 1	$Cost_{\text{mean}}$	15275.930392	15275.930392	15275.930392	15275.930392	15275.930392
	$Cost_{\text{std}}$	5.393880e-12	5.393880e-12	5.287060e-12	5.393880e-12	5.446504e-12
Case 2	$Cost_{\text{mean}}$	63975.710808	63975.710808	63975.710808	63975.710808	63975.710808
	$Cost_{\text{std}}$	8.884254e-10	1.613617e-09	2.261871e-09	1.617213e-09	3.800355e-09
Case 3	$Cost_{\text{mean}}$	111497.562806	111497.562820	111497.562812	111497.562816	111497.562839
	$Cost_{\text{std}}$	9.790660e-05	9.978463e-05	8.476343e-05	8.150888e-05	1.648294e-04
Case 4	$Cost_{\text{mean}}$	4303.508087	4303.508091	4303.508093	4303.508098	4303.508097
	$Cost_{\text{std}}$	7.348985e-05	8.618352e-05	9.250994e-05	8.149627e-05	9.892146e-05
Case 5	$Cost_{\text{mean}}$	121432.136531	121506.279026	121512.417112	121523.706952	121516.618637
	$Cost_{\text{std}}$	20.625706	65.174168	50.510727	71.082394	60.115250
Case 6	$Cost_{\text{mean}}$	121433.128567	121595.054622	121558.534137	121533.308177	121584.629751
	$Cost_{\text{std}}$	20.061615	185.677003	104.007310	75.755280	122.524991

The optimization results of PADE over 30 runs are shown in Table 3.

$Cost_{\text{mean}}$ and $Cost_{\text{std}}$ stand for the mean and standard deviation values of 30 test results, respectively. The $Cost_{\text{mean}}$ and $Cost_{\text{std}}$ values are used to measure the solution quality and stability with respect to different SR_K values, respectively. It can be seen that all the $Cost_{\text{mean}}$ values are same with respect to different SR_K values for cases 1-2, meaning that the SR_K value does not affect the solution quality. For the remaining cases, the $Cost_{\text{mean}}$ values with respect to $SR_K = 0.1$ are better than those with respect to other SR_K values, indicating that $SR_K = 0.1$ is more suitable than other values in improving the solution quality. Additionally, the $Cost_{\text{std}}$ values with respect to $SR_K = 0.1$ are the best among all the $Cost_{\text{std}}$ values with respect to different SR_K values for four out of six cases, which suggests that PADE with $SR_K = 0.1$ has more stronger stability than PADE with other SR_K values. In sum, $SR_K = 0.1$ is suitable, since it is beneficial for the solution quality and stability.

To investigate the influence of their adaptive mechanism for two control parameters on the solution quality and stability, PADE with adaptive control parameters is compared with PADE with fixed control parameters by the criteria $Cost_{\text{mean}}$ and $Cost_{\text{std}}$. The fixed F and CR values are set to 0.6 and 0.3, respectively. The other experimental parameters are the same as those of Section 4.2. The comparison results are listed in Table 4.

According to Table 4, it is clearly seen from the terms $Cost_{\text{mean}}$ and $Cost_{\text{std}}$ that PADE with adaptive control parameters is superior to PADE with fixed control parameters for all cases except case 1. For case 1, the solution quality and stability of PADE with adaptive control parameters and PADE with fixed control parameters are same. In all, the proposed adaptive control parameters have a significant effect in improving the performance of PADE for most cases.

4.6. Comparison between PADE with Phase-Based and Five Popular Mutation Operators. To test the effect of the proposed phase-based mutation compared to five popular mutation operators, PADE with phase-based mutation are compared with PADE with different mutation operators on the solution quality and stability. The other experimental parameters are the same as those of Section 4.2. The comparison results are presented in Table 5

In Table 5, the $Cost_{\text{mean}}$ value of PADE with phase-based mutation is better than, or as good as that of PADE with other

Table 4: Comparison between PADE with adaptive and fixed control parameters for six ELD problems.

Problems	Terms	PADE with fixed control parameters	PADE with adaptive control parameters
Case 1	$Cost_{\text{mean}}$	15275.930392	15275.930392
	$Cost_{\text{std}}$	5.393880e-12	5.393880e-12
Case 2	$Cost_{\text{mean}}$	63975.710817	63975.710808
	$Cost_{\text{std}}$	6.775580e-06	8.884254e-10
Case 3	$Cost_{\text{mean}}$	111497.563739	111497.562806
	$Cost_{\text{std}}$	4.724601e-04	9.790660e-05
Case 4	$Cost_{\text{mean}}$	4303.508253	4303.508087
	$Cost_{\text{std}}$	1.098271e-04	7.348985e-05
Case 5	$Cost_{\text{mean}}$	121476.565508	121432.136531
	$Cost_{\text{std}}$	22.458685	20.625706
Case 6	$Cost_{\text{mean}}$	121485.558871	121433.128567
	$Cost_{\text{std}}$	31.898932	20.061615

mutation operators for the first two cases. For cases 3-4, the $Cost_{\text{mean}}$ value of PADE with phase-based mutation is slightly larger than that of PADE with DE/best/1 and is lower than that of PADE with other mutation operators. It means that the solutions obtained by PADE with phase-based mutation are slightly worse than those of PADE with DE/best/1 but are better than those of PADE with other mutation operators. PADE with phase-based mutation outperforms PADE with another five mutation operators on the solution quality and stability for the last two complex ELD problems.

4.7. Comparison between PADE and the Other Methods Presented in the Literature. In this section, the best results of PADE are compared with those presented in the literature to evaluate the performance of different methods. Different from most literatures, we not only give the best cost function value (C), but also calculate and list the constraint violations of their best solutions listed in the paper. Here, the constraint violations are caused by power output limits, power balance constraints with or without transmission losses, or POZs for six studied ELD problems. The constraint violations of power output limits, power balance constraints with or without transmission losses and POZs are denoted as V_G, V_P and V_{POZ}. For the simplest ELD case, the best solution of case 1 can be obtained according to [41], and its corresponding cost function value is 15275.93039, which is the same as the best cost function value of PADE. Nevertheless, the values of V_G and V_P calculated are, respectively, equal to 0 and 6e-08, which makes the best solution incompletely feasible. The V_G and V_P values of PADE are exactly equal to 0 according to its best solution (446.707273, 171.257992, 264.105658, 125.216766, 172.118858, and 83.593453). Therefore, PADE performs better than that method from [41]. For the remaining five cases, most of the best solutions and corresponding costs have been presented in the literature, and all constraint violations V_G, V_P and V_{POZ} are calculated according to those best solutions. The best solutions, C, and corresponding constraint violations of inequality and equality constraints are listed in Tables 6–10. Besides, the minority of reported minimal costs exist calculation errors, and those costs are amended according to their best solutions, which is to make the comparisons fair and meaningful. To be more specific, for a reported method, its best cost is verified by bring its best distribution of the generating units in Tables 6–10 into formulas (1)-(3) to recalculate its objective function value. AC denotes the amended cost, and the term "NA" represents "not available." As we all know, the inequality constraints are easy to completely meet, so its error tolerance (ε) is set to 0. However, because the actual ELD problems usually perform nonlinear characteristics especially for the ELD problems with transmission losses, the equality constraint, that is power balance constraint, can be slightly relaxed this restriction of the error tolerance. For the ELD problems with transmission losses, ε is set to 0.001 according to our repair method. Accordingly, if $V_P \leq \varepsilon$, corresponding constraint violations are determine to be acceptable and vice versa for cases 2-4. In addition, since the proposed repair method is easy to completely repair the constraint violations for the ELD problems without transmission losses, the smaller the constraint violations are, the finer corresponding solution is. If constraint violations are equal to 0, the feasible solution can be obtained.

In Table 6, the best results produced by PADE and the other methods from the literature for case 2 are presented. The best cost obtained by PADE is lower than those of QOTLBO [3], TLBO [3], and BSA [49], and it is higher than that of OGHS [20]. OGHS [20] obtains the lowest cost among all methods, but its V_P value is equal to 2.11, which is far larger than the error tolerance ($\varepsilon = 0.001$) for this case. Thus the solution obtained by OGHS [20] is infeasible. All V_G values obtained by PADE and the other methods are equal to 0, indicating that their inequality constraints are strictly met. Furthermore, the V_P values of PADE, QOTLBO [3], TLBO [3], and BSA [49] meet the error tolerance for this case, so their solutions are considered as the feasible ones. The best cost obtained by TLBO [3] is the largest on all the best costs obtained by the other methods producing the feasible solutions. In all, PADE outperforms the other methods in achieving the minimal fuel cost while meeting the restriction of constraints.

Regarding the optimization of case 3, Table 7 presents the comparison between PADE and the other methods reported

Table 5: The influence of phase-based mutation for six ELD problems.

Problems	Operators	$Cost_{mean}$	$Cost_{std}$
Case 1	PADE with DE/rand/1	15275.930392	5.570775e-12
	PADE with DE/rand/2	15275.930392	5.550256e-12
	PADE with DE/best/1	15275.930392	5.446504e-12
	PADE with DE/best/2	15275.930392	5.550256e-12
	PADE with DE/current-to-best/1	15275.930392	5.446504e-12
	PADE with phase-based mutation	15275.930392	5.393880e-12
Case 2	PADE with DE/rand/1	63975.710809	1.398844e-07
	PADE with DE/rand/2	63975.710813	3.220744e-06
	PADE with DE/best/1	63975.710808	3.128049e-11
	PADE with DE/best/2	63975.710808	5.391868e-09
	PADE with DE/current-to-best/1	63975.710809	2.622406e-07
	PADE with phase-based mutation	63975.710808	8.884254e-10
Case 3	PADE with DE/rand/1	111497.563242	2.967826e-04
	PADE with DE/rand/2	111497.563663	4.310065e-04
	PADE with DE/best/1	111497.562751	6.472343e-05
	PADE with DE/best/2	111497.562863	8.734597e-05
	PADE with DE/current-to-best/1	111497.563057	1.742830e-04
	PADE with phase-based mutation	111497.562806	9.790660e-05
Case 4	PADE with DE/rand/1	4303.508224	1.141844 e-04
	PADE with DE/rand/2	4303.508471	1.410997e-04
	PADE with DE/best/1	4303.508002	5.642989e-05
	PADE with DE/best/2	4303.508103	8.310080e-05
	PADE with DE/current-to-best/1	4303.508264	1.028932e-04
	PADE with phase-based mutation	4303.508087	7.348985e-05
Case 5	PADE with DE/rand/1	121470.028620	27.830628
	PADE with DE/rand/2	121477.405869	26.157857
	PADE with DE/best/1	121602.556527	131.412210
	PADE with DE/best/2	121559.580775	110.238750
	PADE with DE/current-to-best/1	121456.571844	46.328972
	PADE with phase-based mutation	121432.136531	20.625706
Case 6	PADE with DE/rand/1	121470.381090	25.417602
	PADE with DE/rand/2	121475.198768	21.017932
	PADE with DE/best/1	121783.573652	378.163447
	PADE with DE/best/2	121522.816073	79.358607
	PADE with DE/current-to-best/1	121434.323771	22.678410
	PADE with phase-based mutation	121433.128567	20.061615

Table 6: Comparison among different approaches for case 2.

Unit j	QOTLBO [3]	TLBO [3]	OGHS [20]	BSA [49]	PADE
1	79.5547	80.617	83.7483	79.6762	80.754354262
2	88.8977	92.4059	92.0583	88.7507	87.6905012999
3	210	210	210	210	210
4	224.9944	225	225	225	225
5	324.9708	324.9862	315	325	325
6	324.9977	320.1625	325	324.9927	325
C	63977	64032	63953.08	63976	63975.710808
V_G	0	0	0	0	0
V_P	0.00012	2.25e-05	2.11	6.68e-05	9.99999957e-04

Table 7: Comparison among different approaches for case 3.

Unit j	QOTLBO [3]	TLBO [3]	DE [14]	BSA [49]	BSA [42]	OGHS [20]	PADE
1	55	55	55	55	55	55	55
2	79.9991	80	79.8063	80	80	80	80
3	107.9231	105.9616	106.8253	106.9395807	106.9295	106.9916	106.939697
4	98.6479	99.9321	102.8307	100.5762919	100.6028	100.5354	100.573747
5	82.018	80.6424	82.2418	81.5019997	81.499	81.445	81.503676
6	83.4878	85.7878	80.4352	83.0209509	83.0074	83.067	83.020645
7	300	300	300	300	300	299.9998	300
8	340	340	340	340	340	339.9999	340
9	469.9706	469.6979	470	470	470	470	470
10	469.9988	469.9943	469.8975	470	470	469.9999	470
C	111498	111500	111500	111497.63081	111497.6276	111490	111497.562722
AC	–	–	–	–	–	111497.612522	–
V_G	0	0	0	0	0	0	0
V_P	2.57e-05	3.86e-05	3.05e-06	1.12e-07	6.04e-05	0.00031	9.999978e-04

"–" represents that the presented C does not need to be amended.

in the literature. For OGHS [20], its best cost is reported to be 111490. However, its actual cost is equal to 111497.612522 after careful verification, which is slightly higher than that of PADE. The best cost obtained by PADE is lower than those of the other methods, and it is equal to 111497.562722. Besides, the constraint violations of all these methods reach our error tolerance set, suggesting that all solutions are the feasible ones for case 3. Therefore, PADE can achieve the most satisfactory generation dispatch of case 3.

Table 8 lists the best results obtained by the three methods for case 4. Among the three methods, the cost obtained by PADE is the lowest, and it is equal to 4303.507984. For the constraint violations of the three methods, all of their V_G values are equal to 0, and all of their V_P values are smaller than 0.001, revealing that all three methods have strong ability in finding the feasible solution. Overall, the performance of PADE is the best compared to the other two methods.

The best results yielded by PADE and the other methods from the literature are shown in Table 9 for case 5. Most methods from the literature report the corresponding best solution, but unfortunately the best solution, V_G and V_P values of IFEP [2], is not available. We get their best cost from the IFEP [2], and it is equal to 122624.35, which is larger than that of PADE. The best cost obtained by θ-MBA [50] is equal to 121412.5355, which seems similar to that of PADE. However, the value of θ-MBA [50] is incorrect after verification based on the corresponding solution, and the precise cost is equal to 121578.483653, which is much larger than that of PADE. Except for the θ-MBA [50] amended, the best cost obtained by FPSO [16] is also amended to 123860.094144. The best costs obtained by GA-DE-PS [17] and NAPSO [4] are slightly higher than that of PADE, and the best costs obtained by the remaining reported methods are far higher than that of PADE. The V_G value of FPSO [16] is nonzero and equal to 461.85; the V_G value of any of the other methods is exactly equal to 0. If the accuracy required is equal to 1e-05, the solutions obtained by θ-MBA [50], THS [11], and PADE are acceptable. However, if the accuracy required is equal to 1e-06, the solutions obtained by θ-MBA [50] and THS [11] are infeasible, and the solutions obtained by PADE are still feasible. Therefore, it depends on the accuracy required whether the solutions obtained by other methods are feasible. The V_P values of FPSO [16], NAPSO [4], GA-DE-PS [17], THS [11], and θ-MBA [50] are larger than that of PADE, indicating that PADE is better than these methods in achieving the feasible solution.

The proposed algorithm and several approaches from the literature are used to optimize case 6, and their best results are given in Table 10. The best costs obtained by NAPSO [4] and KHA-IV [12] are slightly higher than that of PADE, and the best costs obtained by the remaining reported methods are far higher than that of PADE, which suggests that PADE has stronger search ability compared to the other methods. PSO [4], FAPSO [4], and KHA-IV [12] do not report their best solutions and constraint violations. However, they are inferior to PADE in the best cost. On the obtained V_P value, the V_P value of PADE equal to 1.82e-12 is the lowest, and the V_P value of θ-MBA [50] equal to 72 is the largest, which cannot be neglected. All methods meet the power output limits and POZs completely. Based on the best cost and constraint violation, PADE is more efficient for case 6 compared to the other methods.

4.8. Comparison among Different Constraint Handling Methods. Finally, the selected six ELD cases are employed to test the efficiency of the proposed repair methods and the other reported repair methods on handling constraints, which mainly refers to the equality constraint handling methods. Extensive experiment is carried out to compare the performance of PADE with different constraint handling strategies. Due to difference of the selected problems, we separately propose two types of repair methods for the ELD problems without transmission losses and the ELD problems with transmission losses, and the two repair methods are

Table 8: Comparison among different approaches for case 4.

Unit j	BSA [49]	MHSA [19]	PADE
1	104.1756	52.066	104.115664
2	92.1099	123.913	92.211244
3	50	51.0307	50
4	50	64.1151	50.000002
5	50.0001	55.4468	50.000012
6	50	52.4076	50.000001
7	50	53.8972	50
8	50	50	50.000002
9	62.8778	56.992	62.896512
10	63.0931	75.5895	62.994068
11	62.6157	106.821	62.675866
12	177.6497	98.1268	177.626134
13	50	69.1559	50
14	50	50	50
C	4303.5111	4304.95	4303.507984
V_G	0	0	0
V_P	0.00012	0.00016	9.989949e-04

called PRM1 and PRM2, respectively. In addition, the penalty method (PM) is combined with PRM1 to further handle constraints for the ELD problems without transmission losses, which is represented as PM+PRM1. For the ELD problems with transmission losses, the obtained solution is feasible when $V_P \leq \varepsilon$, so PRM2 does not need to combine PM. The penalty method (PM), the penalty method based on the first repair method (PM+RM1) [10], and the penalty method based on the second repair method (PM+RM2) [52] are used to compare with PM+PRM1 on solving the ELD problems without transmission losses. It should be explained that PM does not need to combine with RM2 for case 1 and case 5, and PM is necessary to combine with RM2 to handle POZs for case 6. The penalty method (PM), the penalty method based on the third repair method (PM+RM3) [53], and the fourth repair method (RM4) [9] are used to compare with PRM2 on solving the ELD problems with transmission losses. RM4 also uses an approximate method to repair infeasible solutions, where its error tolerance is the same as that of PRM2, so PM is not combined with RM4. All constraint handling methods are separately incorporated into PADE, and the population size and maximum generation number are the same as the above experiments. The results of 30 independent experiments are separately listed in Tables 11-12, where the terms AOFV, OFVSTD, and TACV represent the average objective function value, standard deviation of the objective function values, and total average constraint violation, respectively.

It can be clearly seen that PADE based on PM+PRM1 performs better than PADE based on the other constraint-handling methods in AOFV and OFVSTD for the ELD cases without transmission losses, and PADE based on PRM2 also performs better than PADE based on the other constraint-handling methods in AOFV and OFVSTD for the ELD cases with transmission losses. It means that the proposed two repair methods make PADE stronger search ability and stable compared to the other constraint-handling methods. In contrast, the AOFVs and OFVSTDs of PM are the largest on all cases, indicating that the search ability and stable of PADE based on PM are poor. The TACVs of PM+PRM1are lower than that of PM for case 1, case 5, and case 6, and the TACVs of PM+PRM1 are equal to 0 for case 1. Although the TACVs of PM+PRM1 are larger than those of PM+RM1 and (PM)+RM2 for cases 5-6, its TACVs will also reach desirable accuracy if the error tolerance is set to 0.00001 for cases 5-6. For cases 2-4, the TACVs of all these methods meet the requirement of accuracy, because the error tolerance is set to 0.001 for the three cases. Therefore, to some extent, the proposed two repair methods exhibit good efficiency in finding feasible solutions for the ELD problems. In all, PM+PRM1 and PRM2 are very competitive compared to the other constraint-handling methods.

5. Conclusions

We improve the original differential evolution (DE) algorithm to solve the ELD problems. First, an archive of storing successful individuals is set to have a greater chance to obtain the updated individuals. Second, a phase-based mutation operation is executed, where DE/srand/1 and DE/best/1 are adopted in the early and late phase, respectively. To determine the threshold at the end of the early phase, a success ratio and auxiliary generation index threshold are used together to divide the entire evolution phase into the early and late phase. The mutation operation is beneficial to balance the global and local search when the algorithm solves the ELD problems. Third, the two control parameters are adaptively generated for every individual. Different from other self-adaptive control parameters, the control parameters owned by the individuals have the ability to learn from the best

Table 9: Comparison among different approaches for case 5.

Unit j	IFEP [2]	FPSO [16]	GA-DE-PS [17]	THS [11]	NAPSO [4]	θ-MBA [50]	PADE
1	NA	96.7	110.7998	110.8104	110.8018	112.246	110.799825
2	NA	185.6	110.7998	111.3803	110.8	112.3141	110.799825
3	NA	172.2	97.3999	97.4036	97.3999	97.8202	97.399913
4	NA	179.73	179.733	179.7344	179.7331	179.7331	179.7331
5	NA	169.38	87.8	87.8331	87.7997	91.7458	87.799905
6	NA	142.77	139.9999	140	140	140	140
7	NA	334.37	259.5998	259.6201	259.5996	259.6055	259.59965
8	NA	284.6	284.5996	284.6089	284.5996	284.6495	284.59965
9	NA	284.6	284.5996	284.6072	284.5996	284.6061	284.59965
10	NA	273.39	130	130	130	130	130
11	NA	240.87	94.0001	168.799	94	243.5996	94
12	NA	310.28	94	94.0014	94	168.7997	94
13	NA	394.18	214.7597	214.7608	214.7597	125	214.75979
14	NA	393.2	394.2794	394.2831	394.2793	304.5195	394.27937
15	NA	214.76	394.2792	304.5142	394.2793	394.2796	394.27937
16	NA	394.28	394.2795	394.2784	394.2793	304.5196	394.27937
17	NA	489.28	489.2793	489.2764	489.2793	489.2798	489.27937
18	NA	489.28	489.2793	489.2789	489.2793	489.2794	489.27937
19	NA	511.27	511.2793	511.2863	511.2793	511.2794	511.27937
20	NA	421.52	511.2795	511.2802	511.2793	511.2792	511.27937
21	NA	343.77	523.2797	523.2813	523.2793	523.2798	523.27937
22	NA	433.5	523.2797	523.2793	523.2806	523.2793	523.27937
23	NA	433.52	523.2793	523.2822	523.2793	523.2804	523.27937
24	NA	523.28	523.2796	523.2834	523.2793	523.2796	523.27937
25	NA	343.76	523.2796	523.2831	523.2793	523.2796	523.27937
26	NA	433.52	523.2796	523.2801	523.2793	523.2798	523.27937
27	NA	10	10.0002	10.0036	10	10	10
28	NA	9.55	10.0003	10	10	10.0001	10
29	NA	10	10	10	10	10.0002	10
30	NA	128.59	87.8003	97	87.79989	92.3796	87.799905
31	NA	259.47	189.9999	190	190	190	190
32	NA	159.74	189.9999	190	190	190	190
33	NA	259.47	189.9999	190	190	190	190
34	NA	164.79	164.8002	164.8039	164.8014	200	164.799825
35	NA	164.8	194.4059	200	194.3927	192.1066	194.397778
36	NA	239.58	199.9997	199.4622	200	200	200
37	NA	25	109.9998	110	110	109.9999	110
38	NA	7.05	110	110	110	109.9996	110
39	NA	57.05	109.9906	110	110	109.9999	110
40	NA	511.28	511.278	511.2844	511.2793	511.2794	511.27937
C	122624.35	123860.22	121412.8002	121425.15	121412.57	121412.5355	121412.535537
AC	–	123860.094144	–	–	–	121578.483653	–
V_G	NA	461.85	0	0	0	0	0
V_P	NA	0.02	0.0011	0.0002	0.0015	0.0001	4e-06

"–" represents that the presented C doesnot need to be amended.

control parameters owned by the global best individual, which further enhances PADE's robustness. In addition to the algorithm's improvement, two types of repair methods are used to meet the requirement of the equality constraint for the ELD problems without or with transmission losses, respectively. As a result, the proposed repair methods can completely eliminate the constraint violation on solving some ELD cases or reduce a lower level compared to the ones for the other ELD cases. Due to the efficient repair methods, PADE can obtain a lot of high-quality feasible solutions. Finally,

Table 10: Comparison among different approaches for case 6.

Unit j	NAPSO [4]	PSO [4]	FAPSO [4]	MHSA [19]	KHA-IV [12]	θ-MBA [50]	PADE
1	110.7997	NA	NA	112.2539	NA	110.7998	110.799829
2	110.8004	NA	NA	113.8236	NA	110.7998	110.799825
3	97.3971	NA	NA	97.8091	NA	97.39991	97.399913
4	179.7331	NA	NA	179.1497	NA	179.7331	179.7331
5	87.8009	NA	NA	95.9375	NA	87.7999	87.799905
6	139.9999	NA	NA	140	NA	140	140
7	259.5996	NA	NA	260.6374	NA	259.5997	259.59965
8	284.5961	NA	NA	284.7028	NA	284.5997	284.59965
9	284.5997	NA	NA	284.5306	NA	284.5997	284.59965
10	130	NA	NA	198.3517	NA	130	130
11	94	NA	NA	168.7328	NA	168.7998	94
12	94	NA	NA	169.5461	NA	94	94
13	214.7597	NA	NA	125	NA	125	214.75979
14	394.27937	NA	NA	394.1953	NA	400	394.27937
15	394.27937	NA	NA	305.965	NA	394.2794	394.27937
16	394.27937	NA	NA	395.1542	NA	394.2794	394.27937
17	489.27937	NA	NA	491.4043	NA	489.2794	489.27937
18	489.27937	NA	NA	488.8733	NA	489.2794	489.27937
19	511.27937	NA	NA	511.3804	NA	511.2794	511.27937
20	511.27937	NA	NA	511.2114	NA	511.2794	511.27937
21	523.2793	NA	NA	523.823	NA	511.2794	523.27937
22	523.2793	NA	NA	524.1971	NA	511.2794	523.27937
23	523.2793	NA	NA	524.4822	NA	511.2794	523.27937
24	523.2793	NA	NA	523.8066	NA	511.2794	523.27937
25	523.2793	NA	NA	522.9924	NA	511.2794	523.27937
26	523.2793	NA	NA	523.2932	NA	511.2794	523.27937
27	10	NA	NA	10.3634	NA	10	10
28	10	NA	NA	10.4596	NA	10	10
29	10	NA	NA	11.445	NA	10	10
30	87.7998	NA	NA	88.9212	NA	91.43702	87.799905
31	189.9999	NA	NA	190	NA	190	190
32	189.9999	NA	NA	190	NA	190	190
33	189.9999	NA	NA	189.8936	NA	190	190
34	164.8003	NA	NA	166.5926	NA	164.7998	164.799825
35	194.4019	NA	NA	164.8243	NA	200	194.397778
36	199.9999	NA	NA	164.8891	NA	200	200
37	109.9999	NA	NA	109.5147	NA	110	110
38	109.9999	NA	NA	110	NA	110	110
39	109.9999	NA	NA	109.7737	NA	110	110
40	511.2793	NA	NA	512.0693	NA	511.2794	511.27937
C	121412.6102	123861.45	121719.73	121690.271	121412.5991	121491.0662	121412.535602
AC	–	–	–	–	–	121522.397763	–
V_G	0	0	0	0	0	0	0
V_{POZ}	0	0	0	0	0	0	0
V_P	0.0018	NA	NA	0.0001	NA	72	1.82e-12

"–" represents that the presented C does not need to be amended.

Table 11: Comparison among different constraint handling methods for the ELD problems without transmission losses.

Constraint handling methods	Terms	Case 1	Case 5	Case 6
PM	AOFV	15322.203667	147671.117036	146645.192109
	OFVSTD	27.29	6803.57	710.872
	TACV	1.21e-11	0.0015	0.0011
PM+RM1	AOFV	15275.930392	121535.444592	121622.856479
	OFVSTD	5.45e-12	95.59	241.93
	TACV	0	0	0
(PM)+RM2	AOFV	15275.930392	121503.422274	123269.617648
	OFVSTD	7.07e-12	61.4	566.56
	TACV	0	0	0
PM+PRM1	AOFV	15275.930392	121432.136531	121433.128567
	OFVSTD	5.39e-12	20.63	20.06
	TACV	0	4e-06	1.82e-12

Table 12: Comparison among different constraint handling methods for the ELD problems with transmission losses.

Constraint handling methods	Terms	Case 2	Case 3	Case 4
PM	AOFV	65197.353919	115279.671017	4570.409455
	OFVSTD	494.82	902.82	77.76
	TACV	0	0	0
PM+RM3	AOFV	64101.996686	111851.899442	4410.372638
	OFVSTD	146.86	174.21	17.51
	TACV	0	0	0
RM4	AOFV	63975.710809	111497.563378	4303.508257
	OFVSTD	5.59e-07	0.00031	0.00011
	TACV	0	0	0
PRM2	AOFV	63975.710808	111497.562806	4303.508087
	OFVSTD	8.88e-10	9.79e-05	7.35e-05
	TACV	9.99999957e-04	9.999978e-04	9.989949e-04

PADE and the other five DE variants are used to optimize six ELD problems. The experimental results show that PADE performs better than, or as well as the other five state-of-the-art DE approaches in all cases. PADE is useful for finding better solutions in most cases while maintaining a low level for the constraint violations compared to the methods from the literature.

Data Availability

The data used to support the findings of this study will be provided by the author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

This work was supported by the National Natural Science Foundation of China [Grant no. 61403174] and the Postgraduate Research & Practice Innovation Program of Jiangsu Province [Grant no. KYCX17_1575].

References

[1] J. C. Dodu, P. Martin, A. Merlin, and J. Pouget, "Optimal formulation and solution of short- range operating problems for a power system with flow constraints," *Proceedings of the IEEE*, vol. 60, no. 1, pp. 54–63, 1972.

[2] N. Sinha, R. Chakrabarti, and P. K. Chattopadhyay, "Evolutionary programming techniques for economic load dispatch," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 1, pp. 83–94, 2003.

[3] P. K. Roy and S. Bhui, "Multi-objective quasi-oppositional teaching learning based optimization for economic emission load dispatch problem," *International Journal of Electrical Power & Energy Systems*, vol. 53, pp. 937–948, 2013.

[4] T. Niknam, H. D. Mojarrad, and H. Z. Meymand, "Non-smooth economic dispatch computation by fuzzy and self adaptive particle swarm optimization," *Applied Soft Computing*, vol. 11, no. 2, pp. 2805–2817, 2011.

[5] J. P. D. Chattopadhyay, "A multi-area linear programming approach for analysis of economic operation of the Indian power system," *IEEE Transactions on Power Systems*, vol. 11, no. 1, pp. 52–58, 1996.

[6] J. Y. Fan and L. Zhang, "Real-time economic dispatch with line flow and emission constraints using quadratic programming,"

IEEE Transactions on Power Systems, vol. 13, no. 2, pp. 320–325, 1998.

[7] P. Aravindhababu and K. R. Nayar, "Economic dispatch based on optimal lambda using radial basis function network," *International Journal of Electrical Power & Energy Systems*, vol. 24, no. 7, pp. 551–556, 2002.

[8] Z.-X. Liang and J. D. Glover, "A zoom feature for a dynamic programming solution to economic dispatch including transmission losses," *IEEE Transactions on Power Systems*, vol. 7, no. 2, pp. 544–550, 1992.

[9] Q. Qin, S. Cheng, X. Chu, X. Lei, and Y. Shi, "Solving non-convex/non-smooth economic load dispatch problems via an enhanced particle swarm optimization," *Applied Soft Computing*, vol. 59, pp. 229–242, 2017.

[10] L. D. S. Coelho and V. C. Mariani, "An improved harmony search algorithm for power economic load dispatch," *Energy Conversion and Management*, vol. 50, no. 10, pp. 2522–2526, 2009.

[11] M. A. Al-Betar, M. A. Awadallah, A. T. Khader, and A. L. Bolaji, "Tournament-based harmony search algorithm for non-convex economic load dispatch problem," *Applied Soft Computing*, vol. 47, pp. 449–459, 2016.

[12] B. Mandal, P. K. Roy, and S. Mandal, "Economic load dispatch using krill herd algorithm," *International Journal of Electrical Power & Energy Systems*, vol. 57, pp. 1–10, 2014.

[13] D. Zou, S. Li, Z. Li, and X. Kong, "A new global particle swarm optimization for the economic emission dispatch with or without transmission losses," *Energy Conversion and Management*, vol. 139, pp. 45–70, 2017.

[14] M. Basu, "Economic environmental dispatch using multi-objective differential evolution," *Applied Soft Computing*, vol. 11, no. 2, pp. 2845–2853, 2011.

[15] T. T. Nguyen and D. N. Vo, "The application of one rank cuckoo search algorithm for solving economic load dispatch problems," *Applied Soft Computing*, vol. 37, pp. 763–773, 2015.

[16] S. Kumar and D. K. Chaturvedi, "A fuzzy particle swarm optimization for solving the economic dispatch problem," *Advances in Intelligent and Soft Computing*, vol. 130, no. 1, pp. 99–110, 2012.

[17] B. Mahdad and K. Srairi, "Solving practical economic dispatch using hybrid GA-DE-PS method," *International Journal of Systems Assurance Engineering and Management*, vol. 5, no. 3, pp. 391–398, 2014.

[18] J. S. Alsumait, J. K. Sykulski, and A. K. Al-Othman, "A hybrid GA-PS-SQP method to solve power system valve-point economic dispatch problems," *Applied Energy*, vol. 87, no. 5, pp. 1773–1781, 2010.

[19] B. Jeddi and V. Vahidinasab, "A modified harmony search method for environmental/economic load dispatch of real-world power systems," *Energy Conversion and Management*, vol. 78, pp. 661–675, 2014.

[20] M. Singh and J. S. Dhillon, "Multiobjective thermal power dispatch using opposition-based greedy heuristic search," *International Journal of Electrical Power & Energy Systems*, vol. 82, pp. 339–353, 2016.

[21] R. S. Rahnamayan, H. R. Tizhoosh, and M. M. A. Salama, "Opposition-based differential evolution," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 64–79, 2008.

[22] R. Storn and K. Price, "Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[23] M. Basu, "Quasi-oppositional differential evolution for optimal reactive power dispatch," *International Journal of Electrical Power & Energy Systems*, vol. 78, pp. 29–40, 2016.

[24] D. Teijeiro, X. C. Pardo, D. R. Penas, P. González, J. R. Banga, and R. Doallo, "A cloud-based enhanced differential evolution algorithm for parameter estimation problems in computational systems biology," *Cluster Computing*, vol. 20, no. 3, pp. 1937–1950, 2017.

[25] D. X. Zou, X. Wang, and N. Duan, "Iris location algorithm based on modified differential evolution algorithm," *Control Theory & Applications*, vol. 30, no. 9, pp. 1194–1200, 2013.

[26] N. M. Hamza, D. L. Essam, and R. A. Sarker, "Constraint Consensus Mutation-Based Differential Evolution for Constrained Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 447–459, 2016.

[27] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*, Springer-Verlag, 2005.

[28] W. Yi, Y. Zhou, L. Gao, X. Li, and J. Mou, "An improved adaptive differential evolution algorithm for continuous optimization," *Expert Systems with Applications*, vol. 44, pp. 1–12, 2016.

[29] S.-M. Guo, C.-C. Yang, P.-H. Hsu, and J. S.-H. Tsai, "Improving differential evolution with a successful-parent-selecting framework," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 5, pp. 717–730, 2015.

[30] R. Tanabe and A. S. Fukunaga, "Improving the search performance of shade using linear population size reduction," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC '14)*, pp. 1658–1665, IEEE, July 2014.

[31] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398–417, 2009.

[32] J. Q. Zhang and A. C. Sanderson, "JADE: adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, 2009.

[33] M. Miranda-Varela and E. Mezura-Montes, "Constraint-handling techniques in surrogate-assisted evolutionary optimization. An empirical study," *Applied Soft Computing*, vol. 73, pp. 215–229, 2018.

[34] X. T. Li and M. H. Yin, "Modified differential evolution with self-adaptive parameters method," *Journal of Combinatorial Optimization*, vol. 31, no. 2, pp. 546–576, 2016.

[35] J. Liu, X. M. Yin, and X. S. Gu, "Differential evolution improved with adaptive control parameters and double mutation strategies," in *Proceedings of the Asian Simulation Conference/SCS Autumn Simulation Multi-Conference*, pp. 186–198, 2016.

[36] W. Yi, L. Gao, X. Li, and Y. Zhou, "A new differential evolution algorithm with a hybrid mutation operator and self-adapting control parameters for global optimization problems," *Applied Intelligence*, vol. 42, no. 4, pp. 642–660, 2015.

[37] L. X. Tang, Y. Dong, and J. Y. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Transaction on Evolution Computing*, vol. 19, no. 4, pp. 560–574, 2015.

[38] S. H. Wang, Y. Z. Li, H. Y. Yang et al., "Self-adaptive differential evolution algorithm with improved mutation strategy," *Soft Computing*, vol. 22, no. 10, pp. 3433–3447, 2018.

[39] R. D. Al-Dabbagh, F. Neri, N. Idris, and M. S. Baba, "Algorithmic design issues in adaptive differential evolution schemes: Review and taxonomy," *Swarm and Evolutionary Computation*, 2018.

[40] S. M. Elsayed, R. A. Sarker, and D. L. Essam, "Self-adaptive differential evolution incorporating a heuristic mixing of operators," *Computational optimization and applications*, vol. 54, no. 3, pp. 771–790, 2013.

[41] K. Thanushkodi, S. M. V. Pandian, R. S. D. Apragash et al., "An efficient particle swarm optimization for economic dispatch problems with non-smooth cost functions," *Wseas Transactions on Power Systems*, vol. 3, no. 4, pp. 257–266, 2008.

[42] M. Modiri-Delshad and N. A. Rahim, "Multi-objective backtracking search algorithm for economic emission dispatch problem," *Applied Soft Computing*, vol. 40, pp. 479–494, 2016.

[43] J. Brest, S. Greiner, B. Bošković, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[44] L. Jia, J. He, C. Zhang, and W. Gong, "Differential evolution with controlled search direction," *Journal of Central South University*, vol. 19, no. 12, pp. 3516–3523, 2012.

[45] D. T. Do, S. Lee, and J. Lee, "A modified differential evolution algorithm for tensegrity structures," *Composite Structures*, vol. 158, pp. 11–19, 2016.

[46] Y. Wang, Z. X. Cai, and Q. F. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, 2011.

[47] R. Mallipeddi and P. N. Suganthan, "Problem definitions and evaluation criteria for the CEC 2010 competition on constrained real-parameter optimization," Technical Report, 2010.

[48] J. J. Liang, T. P. Runarsson, E. Mezura-Montes et al., "Problem definitions and evaluation criteria for the CEC 2006 special session on constrained real-parameter optimization," Technical Report, 2006.

[49] K. Bhattacharjee, A. Bhattacharya, and S. Halder Nee Dey, "Backtracking search optimization based economic environmental power dispatch problems," *International Journal of Electrical Power & Energy Systems*, vol. 72, pp. 830–842, 2015.

[50] A. Kavousi-Fard and A. Khosravi, "An intelligent θ-modified bat algorithm to solve the non-convex economic dispatch problem considering practical constraints," *International Journal of Electrical Power & Energy Systems*, vol. 82, pp. 189–196, 2016.

[51] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 3–18, 2011.

[52] W. Sun, W. Shang, and D. Niu, "Application of improved ant colony optimization algorithm in distribution network planning," *Power System Technology*, vol. 30, no. 15, pp. 85–89, 2006.

[53] V. Hosseinnezhad and E. Babaei, "Economic load dispatch using θ-PSO," *International Journal of Electrical Power & Energy Systems*, vol. 49, pp. 160–169, 2013.