Finding Multiple Roots of Nonlinear Equation Systems via a Repulsion-Based Adaptive Differential Evolution

Wenyin Gong, Yong Wang, *Senior Member, IEEE*, Zhihua Cai, and Ling Wang

***Abstract*—Finding multiple roots of nonlinear equation systems (NESs) in a single run is one of the most important challenges in numerical computation. We tackle this challenging task by combining the strengths of the repulsion technique, diversity preservation mechanism, and adaptive parameter control. First, the repulsion technique motivates the population to find new roots by repulsing the regions surrounding the previously found roots. However, to find as many roots as possible, algorithm designers need to address a key issue: how to maintain the diversity of the population. To this end, the diversity preservation mechanism is integrated into our approach, which consists of the neighborhood mutation and the crowding selection. In addition, we further improve the performance by incorporating the adaptive parameter control. The purpose is to enhance the search ability and remedy the trial-and-error tuning of the parameters of differential evolution (DE) for different problems. By assembling the above three aspects together, we propose a repulsion-based adaptive DE, called RADE, for finding multiple roots of NESs in a single run. To evaluate the performance of RADE, 30 NESs with diverse features are chosen from the literature as the test suite. Experimental results reveal that RADE is able to find multiple roots simultaneously in a single run on all the test problems. Moreover, RADE is capable of providing better results than the compared methods in terms of both root rate and success rate.**

***Index Terms*—Adaptive parameter control, differential evolution (DE), diversity preservation mechanism, nonlinear equation systems (NESs), repulsion technique.**

Manuscript received January 4, 2018; accepted April 3, 2018. This work was supported in part by the National Natural Science Foundation of China under Grant 61573324, Grant 61673397, and Grant 61673354, in part by the National Natural Science Fund for Distinguished Young Scholars of China under Grant 61525304, in part by the Innovation-Driven Plan in Central South University under Grant 2018CX010, in part by the Hunan Provincial Natural Science Fund for Distinguished Young Scholars under Grant 2016JJ1018, in part by the State Key Laboratory of Intelligent Control and Decision of Complex Systems, and in part by the Fundamental Research Funds for the Central Universities, China University of Geosciences, Wuhan, under Grant CUG160603. This paper was recommended by Associate Editor C. Zhang. *(Corresponding author: Yong Wang.)*

W. Gong and Z. Cai are with the School of Computer Science, China University of Geosciences, Wuhan 430074, China (e-mail: wygong@cug.edu.cn).

Y. Wang is with the School of Information Science and Engineering, Central South University, Changsha 410083, China, and also with the School of Computer Science and Electronic Engineering, University of Essex, Colchester CO4 3SQ, U.K. (e-mail: ywang@csu.edu.cn).

L. Wang is with the Department of Automation, Tsinghua University, Beijing 100084, China (e-mail: wangling@tsinghua.edu.cn).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org, provided by the author. This includes a pdf file with Tables S-I–S-R-XVI and results related to the main paper. This material is 0.272 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMC.2018.2828018

I. Introduction

NUMEROUS real-world applications can be modeled as nonlinear equation systems (NESs) [1]–[3]. In 1998, Fields Medalist Steve Smale listed 18 challenging problems for the twenty-first century including “Does P = NP?” [4]. Among them, the following three challenging problems are related to NESs.

1) *Problem 4:* Integer zeros of a polynomial.
2) *Problem 8:* Introduction of dynamics into economic theory.
3) *Problem 17:* Solving polynomial equations.

Therefore, finding the roots of NESs is not only of great significance for solving practical problems but also one of the core problems of mathematics [5].

Without loss of generality, an NES can be defined as follows:

$$\begin{cases} e_1(\mathbf{x}) = 0 \\ \quad\vdots \\ e_m(\mathbf{x}) = 0 \end{cases} \tag{1}$$

where m is the number of equations, $\mathbf{x} = (x_1, \ldots, x_n)$ is an n-dimensional decision vector, n is the number of decision variables, $\mathbf{x} \in \mathbb{S}$, and $\mathbb{S} \subseteq \mathbb{R}^n$ denotes the search space. Usually

$$\mathbb{S} = \prod_{j=1}^{n} \left[\underline{x}_j, \overline{x}_j\right] \tag{2}$$

where $j = 1, \ldots, n$, and $\underline{x}_j$ and $\overline{x}_j$ are the lower and upper bounds of x_j, respectively.

If $\forall i \in \{1, \ldots, m\}$, $e_i(\mathbf{x}^*) = 0$, then $\mathbf{x}^*$ is said to be a root (or an optimal solution) of an NES. An NES may have multiple roots, especially when $n \geq m$. For example, taking the eighth test problem (F08) in the supplementary material into account, Fig. 1 depicts its seven roots. In F08, there are two nonlinear equations [$e_1(\mathbf{x})$ and $e_2(\mathbf{x})$] and two decision variables (x_1 and x_2), i.e., $n = m = 2$. In real-world applications, each root may be equally important; therefore, the main task of solving NESs is to simultaneously find these roots in a single run, which is nondeterministic polynomial-time hard and thus poses a grand challenge in numerical computation [6].

The *repulsion* technique (also known as the *polarization* technique) [7]–[9] has great potential to guide an optimization

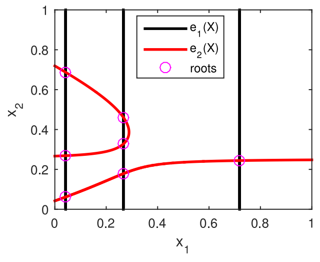

Fig. 1. Illustration of the eighth test problem (F08) in the supplementary material and its seven roots.

algorithm toward new roots by creating repulsion areas around the roots previously identified. However, to find as many roots as possible, the diversity and search ability of the optimization algorithm also play crucial roles other than the repulsion technique. On the one hand, after finding a root, if the diversity of the optimization algorithm is poor and if all the solutions of the optimization algorithm already gather in a small region, it is very hard to find new roots in the subsequent search. On the other hand, the optimization algorithm with weak search ability may not be capable of probing unexplored yet promising regions. Therefore, the repulsion technique should be equipped with an optimization algorithm with good diversity and strong search ability when solving NESs, which has not yet been systematically investigated in current research.

Based on this analysis, this paper makes the first attempt to combine the repulsion technique, diversity preservation mechanism, and adaptive parameter control in an effective way, with the purpose of developing a powerful method for finding multiple roots of NESs. We call the proposed method a repulsion-based adaptive differential evolution (RADE). In RADE, differential evolution (DE) [10] serves as the optimization algorithm, and the aim of the repulsion technique is to assist DE in finding multiple roots of NESs simultaneously in a single run. In addition, the diversity preservation mechanism is employed to diversify the population of DE, in which each individual implements mutation and crossover with its neighbors to generate a trial vector, and the trial vector is compared with the nearest individual in the parent population for survival. Moreover, the adaptive parameter control is used to automatically adapt the control parameters of DE to appropriate values, thereby enhancing the search ability for solving different NESs.

The main contributions of this paper can be summarized as follows.

1) As aforementioned, three aspects are elaborated in RADE to solve NESs, i.e., the repulsion technique—avoiding repeated search in the regions previously explored; the diversity preservation mechanism—maintaining the diversity of the population and increasing the possibility to find new roots; and the adaptive parameter control—enhancing the search ability without trial-and-error parameter settings. As a result, RADE can be considered as an effective alternative for solving NESs.
2) Thirty NESs were chosen from the literature as the test suite, which can be used as the benchmark test problems to evaluate the performance of different methods. Experimental results verify that RADE can not only consistently find multiple roots of different NESs in a single run but also yield better results than the compared methods with respect to both root rate and success rate.
3) The effectiveness of the three important aspects of RADE has been experimentally investigated. Furthermore, we have empirically discussed the influence of other repulsion techniques, other diversity preservation mechanisms, other adaptive parameter controls, other mutation operators, and different parameter settings on the performance of RADE.

The rest of this paper is organized as follows. Section II introduces the background, including the repulsion techniques and the classical DE. Section III discusses the related work and motivation. In Section IV, the proposed RADE is described in detail. The experimental studies and comprehensive discussions are given in Section V. Finally, Section VI concludes this paper.

II. Background

A. Repulsion Techniques for NESs

When solving an NES by an optimization algorithm, it is usually transformed into a *minimization* problem as follows:

$$\text{minimize} \quad f(\mathbf{x}) = \sum_{i=1}^{m} e_i^2(\mathbf{x}) \tag{3}$$

or

$$\text{minimize} \quad f(\mathbf{x}) = \sum_{i=1}^{m} |e_i(\mathbf{x})|. \tag{4}$$

Afterward, to detect the roots of an NES is equivalent to finding the global minimizers of the transformed optimization problem in (3) or (4).

In the NES literature, several repulsion techniques have been developed to find multiple roots [7]. The basic idea behind the repulsion technique is that a repulsion function is designed based on $f(\mathbf{x})$ to create repulsion areas around the roots previously found so that these roots will not be the global minimizers anymore. Therefore, the repulsion technique aims at motivating the optimization algorithm to explore new search regions and to search for new roots. Next, we briefly introduce two representative (i.e., *multiplicative* and *additive*) techniques which will be utilized in this paper.

1) *Multiplicative Repulsion Technique:* Pourjafari and Mojallali [9] presented a multiplicative repulsion function as follows:

$$\text{minimize} \quad R(\mathbf{x}) = (f(\mathbf{x}) + \epsilon) \prod_{j=1}^{K} \left|\coth\left(\alpha\delta_j\right)\right| \tag{5}$$

where

$$\delta_j = \| \mathbf{x} - \mathbf{x}_j^* \|. \tag{6}$$

K is the number of roots that have been found, $\mathbf{x}_j^*$ is the jth root, δ_j is the Euclidean distance between $\mathbf{x}$ and $\mathbf{x}_j^*$, $\alpha \geq 1$ is a density factor to adjust the radius of the repulsion regions, and ϵ is a very small positive constant

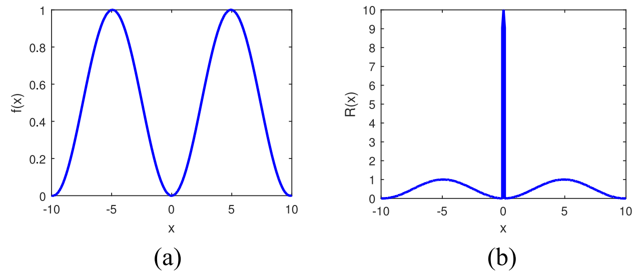

Fig. 2. Comparison between the original objective function $f(x)$ and the repulsion function $R(x)$. (a) Original objective function $f(x) = \sin^2(x/\pi), x \in [-10, 10]$. (b) Repulsion function $R(x)$. If one root $x = 0$ was obtained, then $R(x)$ creates a repulsion area around $x = 0$ to penalize the solutions in this repulsion area. As shown in Fig. 2(b), $x = 0$ is no longer a global minimizer because $R(x) \gg 0$ around $x = 0$.

or equal to 0. In this paper, $\epsilon = 1\text{E-}10$ and $\alpha = 10$ as suggested in [9]. In [11], another multiplicative repulsion function based on the error function "erf" is presented

$$\text{minimize} \quad R(\mathbf{x}) = (f(\mathbf{x}) + \epsilon) \prod_{j=1}^{K} \zeta_{\rho'}(\gamma, \delta_j) \tag{7}$$

where

$$\zeta_{\rho'}(\gamma, \delta_j) = \begin{cases} |\text{erf}(\gamma \delta_j)|^{-1}, & \text{if } \delta_j \le \rho' \\ 1, & \text{otherwise.} \end{cases} \tag{8}$$

$\gamma > 0$ scales the penalty, and ρ' adjusts the radius of the repulsion regions. In this paper, $\gamma = 0.1$ and $\rho' = 0.1 \min_{i=1}^{n}(\overline{x}_i - \underline{x}_i)$ as suggested in [11].

2) *Additive Repulsion Technique:* In [7], an additive repulsion function is formulated as follows:

$$\text{minimize} \quad R(\mathbf{x}) = f(\mathbf{x}) + \beta \sum_{j=1}^{K} \mathrm{e}^{-\delta_j} \chi_\rho(\delta_j) \tag{9}$$

where

$$\chi_\rho(\delta_j) = \begin{cases} 1, & \text{if } \delta_j \le \rho \\ 0, & \text{otherwise.} \end{cases} \tag{10}$$

$\chi_\rho(\delta_j)$ is the characteristic function, ρ is a small constant to adjust the radius of the repulsion regions, and β is a large constant to control the penalty scale. In this paper, $\rho = 0.01$ and $\beta = 1000$ as suggested in [7].

Fig. 2 gives an example to explain the principle of the repulsion technique. In this example, the repulsion technique in (9) with $\rho = 0.1$ and $\beta = 10$ is used. The original objective function is: $f(x) = \sin^2(x/\pi), x \in [-10, 10]$. It is clear from Fig. 2(a) that there are three roots with $f(x) = 0$, i.e., $x = -10$, $x = 0$, and $x = 10$. Suppose that one root (e.g., $x = 0$) was obtained. Afterward, the original objective function is modified according to the repulsion function. As a result, $x = 0$ is not a global minimizer anymore. As shown in Fig. 2(b), the solutions in the repulsion area (i.e., the area around $x = 0$) are penalized while other solutions outside the repulsion area are not affected, which signifies that the repulsion area is unpromising.

There are also other similar repulsion techniques presented in [8] and [12]–[14]. In this paper, we only use the above-mentioned three repulsion techniques; other repulsion techniques are not described to save space. Note that different repulsion techniques have different features, Section V-E1 will empirically evaluate the influence of the above-mentioned three repulsion techniques.

B. Differential Evolution

DE is a simple yet powerful evolutionary algorithm (EA) for numerical optimization [10]. Similar to other EAs, DE contains four main steps, i.e., *initialization*, *mutation*, *crossover*, and *selection*. Initially, the population is randomly generated in the search space. After initialization, DE generates the trial vectors by making use of the mutation and crossover. Then, the trial vectors are evaluated by the fitness function. Finally, in the selection, each trial vector is compared with its corresponding parent, and replaces the parent only if it has an equal or better fitness value. DE repeats the mutation, crossover, and selection until a predefined termination criterion is satisfied. More details of DE can be found in [15]. Originally, DE was developed for single-objective optimization problems. Recent advances have successfully adapted DE to other kinds of optimization problems, such as dynamic bias current control [16], constrained optimization [17], multiobjective optimization [18], and multimodal optimization [19].

III. Related Work and Motivation

A. Related Work

As reviewed in [6], [20], and [21], several different classical methods have been proposed to solve NESs, which can be classified as follows.

1) *Newton and Quasi-Newton Type Methods:* These methods may obtain super-linear convergence when the initial guess is close to one of the roots, e.g., [22] and [23].
2) *Homotopy Continuation (Embedding) Methods:* In these methods, a hard problem is first transformed into a much simpler one, and then this simpler problem gradually deforms into the original one, e.g., [24] and [25].
3) *Interval-Newton Methods:* The classical Newton-like iterative methods are applied to the interval variables in the interval-Newton methods, e.g., [26] and [27].
4) *Deterministic Branch-and-Bound Methods:* These methods transform an NES into a global optimization problem, and then the transformed optimization problem is solved by the branch-and-bound methods, e.g., [20] and [28].

Besides the classical methods, there are also some stochastic methods to find multiple roots of NESs. Generally, these methods can be classified into three categories.

1) *Clustering-Based Methods:* In the literature, the clustering techniques are used to gather similar solutions into different clusters to locate different roots of NESs. For example, a clustering method is used at the exact search phase in [9]. In addition, the clustering-based Minfider is used to find multiple roots in [29] and the fuzzy clustering means is employed in [30].
2) *Multiobjective Optimization-Based Methods*: Similar to NESs, multiobjective optimization problems also have multiple Pareto optimal solutions. Recognizing this similarity, some researchers have tried to locate multiple

roots of NESs based on multiobjective optimization. For example, an NES is transformed into an *m*-objective optimization problem in [31]. Additionally, a biobjective transformation technique called multiobjective optimization based NES (MONES) is presented in [32] and Qin *et al.* [33] presented a $(n + 1)$-objective transformation technique for NESs. Very recently, Gong *et al.* [34] presented a weighted biobjective transformation for NESs.

3) *Repulsion-Based Methods:* As mentioned in Section II-A, the repulsion techniques are able to construct the areas of repulsion around the found roots. Based on the repulsion techniques, different methods are designed to find multiple roots of NESs, such as continuous GRASP [7], two-phase root-finder using IWO [9], improved harmony search (I-HS) algorithm [11], multistart simulated annealing [12], Nelder–Mead (N–M)-based repulsion algorithm [14], and biased random-key GA [35].

B. Motivation

As pointed out in [6] and [21], the classical methods have some weaknesses. For example, the performance of the Newton and quasi-Newton type methods is highly sensitive to the initial guess; the embedding methods cannot directly deal with variable bounds and inequality constraints; the interval-Newton methods are computationally expensive; and the deterministic branch-and-bound methods need the specification of bounded intervals. More importantly, the classical methods only focus on one of the roots in a single run.

With respect to the clustering-based methods, it is difficult to determine the number of clusters if no priori information is available for the problem at hand. Moreover, the data set to be clustered and the clustering period are also problem-dependent [30].

For the multiobjective optimization-based methods, the approaches in [31] and [33] may suffer from the "curse of dimensionality" due to the fact that the number of objectives in them is related to *m* or *n*. With the drastic increase of *m* or *n*, they will transform an NES into a *many*-objective optimization problem, which is a grand challenge in the field of evolutionary computation [36].

As far as the repulsion-based methods are concerned, most of them need to restart many times so as to obtain different roots [7], [9], [11], [12], [35]. Under this condition, some useful information discovered previously may be lost in a new run. For example, some solutions in [9] and [35] may lie within the attraction basins of certain roots in the current run. However, such useful information is neglected unreasonably in a new run because of the population reinitialization, thus leading to inefficiency.

The reason why the restart is frequently used in the repulsion-based methods seems straightforward: if without restart, the diversity of the population will gradually decrease after one or several roots have been identified due to the lack of diversity preservation. Therefore, it is believed that if the diversity of the population can be maintained, we can eliminate the restart while keeping the advantage of the repulsion technique. Motivated by the above considerations, we present a repulsion-based adaptive DE in this paper, named RADE. In RADE, the repulsion technique is exploited to assist DE in locating different roots. In addition, the diversity preservation mechanism, consisting of the neighborhood mutation and the crowding selection, is used to diversify the population. In principle, after a new root has been found, the repulsion function should be modified according to the information of this root, which means that the repulsion technique will construct a sequence of minimization problems based on the sequentially modified repulsion function. Note that to locate a root of a specific minimization problem, the search ability of an optimization algorithm is vital. To this end, the adaptive parameter control is implemented to enhance the search ability of DE and to avoid the trial-and-error tuning of DE's parameters. The synergy of the repulsion technique, the diversity preservation mechanism, and the adaptive parameter control provides an effective way to find multiple roots of NESs simultaneously in a single run: when some roots have been detected, the population will continue to search new regions and find undetected roots because of the repulsion property, good diversity, and strong search ability. To the best of our knowledge, it is the first attempt to integrate these three aspects together to solve NESs.

IV. Proposed Approach

In this section, a repulsion-based adaptive DE (RADE) is proposed to find multiple roots of NESs. In RADE, DE serves as the search engine. Actually, RADE can be considered as a generic framework since it can accommodate different repulsion techniques, diversity preservation mechanisms, and adaptive parameter controls. The details of RADE are described in the following sections.

A. Repulsion in RADE

After an initial population is generated, RADE will search for the first root of an NES by minimizing $f(\mathbf{x})$ in (3) or (4). Once the first root has been located, RADE will change to optimize $R(\mathbf{x})$ in (5), (7), or (9) during the rest of evolution. Therefore, for each individual, its fitness function $F(\mathbf{x})$ can be defined as follows:

$$F(\mathbf{x}) = \begin{cases} f(\mathbf{x}), & \text{if no root has been found} \\ R(\mathbf{x}), & \text{otherwise.} \end{cases} \tag{11}$$

It can be seen from (5), (7), and (9) that when computing $R(\mathbf{x})$, the information of all the roots previously found should be used. Therefore, we store all the obtained roots into a predefined archive $\mathcal{A}$. In RADE, if a new root is identified, we will update $\mathcal{A}$ according to Algorithm 1. If this new root can be added into $\mathcal{A}$, then we will implement the following processes: 1) modifying $R(\mathbf{x})$ by incorporating the information of this new root and 2) re-evaluating all the individuals in the population based on the modified $R(\mathbf{x})$. In Algorithm 1, θ is a very small positive number to judge whether an individual is

Algorithm 1: Archive Updating

```
Input: Individual $\mathbf{x}$, archive $\mathcal{A}$, accuracy level $\theta$, distance
       radius $\tau_d$, and maximum archive size $\max_s$
Output: The updated $\mathcal{A}$ and the archive size $s$
1  if $f(\mathbf{x}) < \theta$ then                          // x is a root
2      if $s = 0$ then                                   // A is empty
3          $\mathcal{A} = \mathcal{A} \cup \mathbf{x}$;
4          $s = s + 1$;
5      else
6          Find the closest solution $\mathbf{x}' \in \mathcal{A}$ to $\mathbf{x}$ in the
           decision space;
7          if $\| \mathbf{x} - \mathbf{x}' \| < \tau_d$ then // x and x' are too close
8              if $f(\mathbf{x}) < f(\mathbf{x}')$ then
9                  $\mathbf{x}' = \mathbf{x}$;
10         else
11             if $s < \max_s$ then
12                 $\mathcal{A} = \mathcal{A} \cup \mathbf{x}$;
13                 $s = s + 1$;
14             else                                      // A is full
15                 if $f(\mathbf{x}) < f(\mathbf{x}')$ then
16                     $\mathbf{x}' = \mathbf{x}$;
```

a root or not, τ_d is a distance radius to avoid redundant roots in $\mathcal{A}$, and $\max_s$ is the maximum archive size.[1]

Over the course of evolution, $R(\mathbf{x})$ will be sequentially modified with the information of the newly found roots in $\mathcal{A}$, and any individual lying within one of the repulsion areas determined by $R(\mathbf{x})$ will be penalized as introduced in Section II-A and Fig. 2.

B. Diversity Preservation Mechanism

In evolutionary computation, the diversity of population plays a crucial role in balancing the exploitation and exploration [37], [38]. The diversity is also a major factor which affects the finding of multiple roots of NESs in a single run. In general, different diversity preservation mechanisms in evolutionary computation can be incorporated into the proposed RADE. As an illustration, we introduce a diversity preservation mechanism inspired by [39], which integrates the neighborhood mutation and the crowding selection with DE.

Suppose that the population $\mathcal{P}$ of DE consists of NP individuals: $\mathbf{x}_1, \ldots, \mathbf{x}_{NP}$. In contrast to the traditional DE mutation, the neighborhood mutation only selects several similar individuals measured by Euclidean distance in the decision space to generate a mutant vector for each individual.

1) For each individual $\mathbf{x}_i = (x_{i,1}, \ldots, x_{i,n})$ $(i = 1, \ldots, NP)$, select ℓ individuals with the smallest Euclidean distances to $\mathbf{x}_i$ in $\mathcal{P}$ to form the subpopulation $subpop_i$.

[1]Note that the parameter $\max_s$ is not mandatory. In this paper, it is used in order not to save too many solutions into $\mathcal{A}$.

2) Generate a mutation vector $\mathbf{v}_i = (v_{i,1}, \ldots, v_{i,n})$:

$$\mathbf{v}_i = \mathbf{x}_{r1} + F \cdot (\mathbf{x}_{r2} - \mathbf{x}_{r3}) \tag{12}$$

where $\mathbf{x}_{r1}$, $\mathbf{x}_{r2}$, and $\mathbf{x}_{r3}$ are randomly chosen from $subpop_i$, and F is the scaling factor of DE.

It is obvious that the neighborhood mutation produces a mutant vector within a local area, which enables $\mathcal{P}$ to distribute around the attraction basins of different roots.

After the crossover, a trial vector $\mathbf{u}_i = (u_{i,1}, \ldots, u_{i,n})$ is created:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } rand_j < CR \text{ or } j = j_{rand} \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{13}$$

where $rand_j$ is a uniformly distributed random number from $[0, 1]$, j_{rand} is a random integer uniformly generated from $\{1, \ldots, n\}$, and CR is the crossover rate of DE.

In the crowding selection, the trial vector $\mathbf{u}_i$ compares with the closest individual (denoted as $\mathbf{u}_i'$) in $\mathcal{P}$ for survival

$$\mathbf{u}_i' = \begin{cases} \mathbf{u}_i, & \text{if } F(\mathbf{u}_i) \le F(\mathbf{u}_i') \\ \mathbf{u}_i', & \text{otherwise.} \end{cases} \tag{14}$$

The crowding selection is able to facilitate multiple convergence since it is carried out between the most similar two individuals.

Overall, the neighborhood mutation and the crowding selection make $\mathcal{P}$ converge toward different roots of an NES by diversifying it.

C. Adaptive Parameter Control

DE has three parameters (i.e., the population size NP, the scaling factor F, and the crossover rate CR) that need to be set by the users. It is noteworthy that the search ability of DE is strongly dependent on the parameter settings [40]. To enhance the search ability and alleviate the tedious tasks of parameter tuning for different problems, F and CR are adaptively controlled based on their successful experience in RADE. As an example, the parameter adaptation technique presented in SHADE [41] is used, because of its superior performance on IEEE CEC2005 and IEEE CEC2013 benchmarks [41], [42]. Note that the influence of other parameter adaptation techniques will be studied in Section V-E3.

In SHADE [41], each individual $\mathbf{x}_i$ $(i = 1, \ldots, NP)$ in the population is associated with a pair of F and CR, denoted as F_i and CR_i. In addition, SHADE also maintains a historical memory with H_m entries, i.e., $M_{F,j}$ and $M_{CR,j}$ $(j = 1, \ldots, H_m)$, for F and CR. Initially, the elements of $M_{F,j}$ and $M_{CR,j}$ $(j = 1, \ldots, H_m)$ are set to 0.5.

At each generation, F_i and CR_i are generated for $\mathbf{x}_i$ as follows:

$$F_i = \text{randc}\big(M_{F,h_i}, 0.1\big) \tag{15}$$

$$CR_i = \text{randn}\big(M_{CR,h_i}, 0.1\big) \tag{16}$$

where randc$(\cdot, \cdot)$ is the Cauchy random number, randn$(\cdot, \cdot)$ is the Gaussian random number, and h_i is an integer randomly selected from $\{1, \ldots, H_m\}$.

During the evolution, if the trial vector is able to successfully replace the parent, then the corresponding F_i and CR_i

will be recorded in $\mathbf{S_F}$ and $\mathbf{S_{CR}}$, respectively. At the end of each generation, M_F and M_{CR} are updated as follows:

$$M_{F,k} = \begin{cases} \text{mean}_{\text{WL}}(\mathbf{S_F}) & \text{if } \mathbf{S_F} \neq \emptyset \\ M_{F,k} & \text{otherwise} \end{cases} \tag{17}$$

$$M_{CR,k} = \begin{cases} \text{mean}_{\text{WA}}(\mathbf{S_{CR}}) & \text{if } \mathbf{S_{CR}} \neq \emptyset \\ M_{CR,k} & \text{otherwise} \end{cases} \tag{18}$$

where $\text{mean}_{\text{WL}}(\mathbf{S_F})$ is the weighted Lehmer mean of $\mathbf{S_F}$ and $\text{mean}_{\text{WA}}(\mathbf{S_{CR}})$ is the weighted arithmetic mean of $\mathbf{S_{CR}}$. In (17) and (18), k is an index between 1 and H_m. k is set to 1 at the beginning of evolution and increases by 1 after one generation. If $k > H_m$, then it is reset to 1.

We make a simple modification to the parameter adaptation in SHADE [41]: the weighted means $\text{mean}_{\text{WL}}(\mathbf{S_F})$ and $\text{mean}_{\text{WA}}(\mathbf{S_{CR}})$ are replaced with the means $\text{mean}_{\text{L}}(\mathbf{S_F})$ and $\text{mean}_{\text{A}}(\mathbf{S_{CR}})$ presented in JADE [43],[2] respectively. The reason is that the fitness-based weights calculated in [41] are not suitable for NESs with multiple roots. For example, a parent $\mathbf{x}_i$ with $f(\mathbf{x}_i) = 0$ generates a different trial vector $\mathbf{u}_i$ with $f(\mathbf{u}_i) = 0$. Hence, $\Delta = f(\mathbf{u}_i) - f(\mathbf{x}_i) = 0$. In this way, the influence of F and CR will be ignored if the weighted means of SHADE [41] are used. However, in fact, they are successful parameters since they generate a new root $\mathbf{u}_i$.

D. Framework of RADE

By integrating the above-mentioned repulsion technique, diversity preservation mechanism, and adaptive parameter control, the framework of RADE is given in Algorithm 2.

1) In line 16, the repulsion technique is applied when the archive is not empty. Note that, since $f(\mathbf{x})$ has been computed in (3), we do not need to recomputer $f(\mathbf{x})$ in (11).
2) In line 12, the neighborhood mutation is used.
3) In lines 15 and 17–19, the crowding selection occurs.
4) In lines 3, 9, 11, and 20–22, the adaptive parameter control is implemented.

Initially, the population $\mathcal{P}$ containing NP individuals is randomly generated and all the individuals in $\mathcal{P}$ are evaluated by $f(\mathbf{x})$ in (3). At each iteration, the archive $\mathcal{A}$ is updated with the individuals in $\mathcal{P}$ by making use of Algorithm 1. Thereafter, NP trial vectors are generated via the neighborhood mutation and crossover of DE, and evaluated by $f(\mathbf{x})$ in (3). Subsequently, the crowding selection is applied to each trial vector and its closest individual in $\mathcal{P}$. If $\mathcal{A} \neq \emptyset$, the comparison between them is based on the repulsion function $R(\mathbf{x})$; otherwise $f(\mathbf{x})$. After the crowding selection, the parameters CR and F of DE will be updated adaptively. Finally, $\mathcal{A}$ will be output if the maximum number of fitness evaluations (i.e., Max_FEs) is reached.

Remark 1: From Algorithm 2, we can see that there are two major differences between RADE and neighborhood mutation-based crowding DE (NCDE) [39].

1) The repulsion technique is used in RADE to find multiple roots of NESs. It is able to avoid convergence to the previously found roots. Since the repulsion technique is employed, an external archive $\mathcal{A}$ is used to store the roots that have already been found.
2) The adaptive parameter control is implemented in RADE, which can enhance the search ability and eliminate the trial-and-error parameter tuning of DE.

Remark 2: In [11], I-HS is proposed to detect multiple roots of an NES in a single run. Unfortunately, several algorithmic parameters in [11] should be set properly. It is worth noting that RADE is similar to the work in [11], because both of them employ EAs and the repulsion technique to find multiple roots of an NES. However, there are two significant differences between them.

1) This paper combines the neighborhood mutation, the crowding selection, and the adaptive parameter control together to form an effective optimizer for NESs.
2) In [11], the population reinitialization is used to maintain the diversity, whereas in RADE the neighborhood mutation and the crowding selection are adopted to preserve

Algorithm 2: Framework of RADE

```
   Input: Control parameters: $NP$, $H_m$, and $Max\_FEs$
   Output: The final archive $\mathcal{A}$
 1 Randomly generate an initial population $\mathcal{P}$, which
   contains $NP$ individuals: $\mathbf{x}_1, \ldots, \mathbf{x}_{NP}$;
 2 Calculate $f(\mathbf{x}_i)$ $(i = 1, \ldots, NP)$ with (3);
 3 Set all values in the historical memory $M_{CR,j}$ and $M_{F,j}$
   $(j = 1, \ldots, H_m)$ to 0.5;
 4 $k = 1$;
 5 Set the archive $\mathcal{A} = \emptyset$ and $FEs = NP$;
 6 while $FEs < Max\_FEs$ do
 7     for $i = 1$ to $NP$ do
 8         Update $\mathcal{A}$ with $\mathbf{x}_i$ using Algorithm 1;
 9     Set $\mathbf{S_{CR}} = \emptyset$ and $\mathbf{S_F} = \emptyset$;
10     for $i = 1$ to $NP$ do
11         Implement (15) and (16) to produce $F_i$ and $CR_i$
           for $\mathbf{x}_i$;
12         Generate the trial vector $\mathbf{u}_i$ via the neighborhood
           mutation and crossover in (12) and (13) ;
13         Calculate $f(\mathbf{u}_i)$ with (3);
14         $FEs = FEs + 1$;
15         Find the closest solution $\mathbf{u}'_i$ in the decision space
           to $\mathbf{u}_i$ in $\mathcal{P}$;
16         Calculate $F(\mathbf{u}'_i)$ and $F(\mathbf{u}_i)$ with (11);
17         if $F(\mathbf{u}_i) \leq F(\mathbf{u}'_i)$ then
18             $\mathbf{u}'_i = \mathbf{u}_i$;
19             $f(\mathbf{u}'_i) = f(\mathbf{u}_i)$;
20             $\mathbf{S_F} \leftarrow F_i$, $\mathbf{S_{CR}} \leftarrow CR_i$;
21     if $\mathbf{S_F} \neq \emptyset$ and $\mathbf{S_{CR}} \neq \emptyset$ then
22         Update $M_{F,k}$ and $M_{CR,k}$ based on $\mathbf{S_F}$ and $\mathbf{S_{CR}}$
           (see (17) and (18));
23         $k = k + 1$;
24         if $k > H_m$ then
25             $k = 1$;
```

[2] For the sake of brevity, we omit the detailed descriptions of $\text{mean}_{\text{WL}}(\mathbf{S_F})$ and $\text{mean}_{\text{WA}}(\mathbf{S_{CR}})$ in SHADE [41], and $\text{mean}_{\text{L}}(\mathbf{S_F})$ and $\text{mean}_{\text{A}}(\mathbf{S_{CR}})$ in JADE [43]. More details can be found in the corresponding references.

TABLE I
BRIEF INFORMATION OF THE 30 NESS, WHERE *n* IS THE NUMBER OF DECISION VARIABLES, *LE* AND *NE* ARE THE NUMBER OF LINEAR AND NONLINEAR EQUATIONS, RESPECTIVELY, *NOR* IS THE NUMBER OF THE KNOWN ROOTS OF AN NES, AND *Max_FEs* IS THE MAXIMUM NUMBER OF FITNESS EVALUATIONS

Prob.	n	LE	NE	NOR	Max_FEs
F01	20	0	2	2	50,000
F02	2	1	1	11	50,000
F03	2	0	2	15	50,000
F04	2	0	2	13	50,000
F05	10	0	10	1	50,000
F06	2	1	1	8	50,000
F07	2	0	2	2	50,000
F08	2	0	2	7	50,000
F09	5	4	1	3	100,000
F10	3	0	3	2	50,000
F11	2	0	2	4	50,000
F12	2	0	2	10	50,000
F13	3	0	3	12	50,000
F14	2	0	2	9	50,000
F15	2	0	2	2	50,000
F16	2	0	2	13	50,000
F17	8	1	7	16	100,000
F18	2	0	2	6	50,000
F19	20	19	1	2	200,000
F20	3	0	3	7	50,000
F21	2	0	2	4	50,000
F22	2	0	2	6	50,000
F23	3	0	3	16	500,000
F24	3	0	3	8	100,000
F25	3	0	3	2	100,000
F26	2	0	2	2	50,000
F27	2	0	2	3	50,000
F28	2	0	2	2	50,000
F29	3	0	3	5	50,000
F30	2	0	2	4	50,000

the diversity. As we mentioned before, a possible drawback of the population reinitialization is the loss of some useful information previously obtained.

V. EXPERIMENTAL STUDIES AND DISCUSSIONS

A. Test Problems and Performance Criteria

To comprehensively evaluate the performance of different methods, we chose 30 NESs (denoted as F01-F30) from the literature. These test problems have different features, and some of them come from real-world applications, such as the interval arithmetic problem (F05) [31], multiple steady states problem (F08) [6], robot kinematics problem (F17) [44], and molecular conformation (F23) [45]. Table I briefly describes them and their detailed information has been provided in the supplementary material.

The transformed optimization problems of NESs in (3) and (4) are similar to multimodal optimization problems, because both of them are required to find multiple roots/optimal solutions. Therefore, we borrowed two performance criteria in [46] for multimodal optimization to compare the performance of different methods for NESs.

1) *Root Rate (RR):* It computes the average ratio of roots found over multiple runs

$$RR = \frac{\sum_{i=1}^{N_r} NOR_i}{NOR \cdot N_r} \tag{19}$$

where N_r is the number of runs, NOR_i is the number of roots found in the ith run, and NOR is the number of the known roots of an NES. In this paper, we made use of the archive $\mathcal{A}$ to compute NOR_i in each run based on Algorithm 1. The three parameters in Algorithm 1 were set as follows and kept unchanged.

a) *Accuracy Level:* θ = 1E-06, if $n \leq 5$; otherwise, θ = 1E-04.

b) *Distance Radius:* $\tau_d = 0.001$, if $n \leq 5$; otherwise, $\tau_d = 0.01$.

c) *Maximum Archive Size:* $\max_s = NP$.

2) *Success Rate (SR):* It measures the ratio of successful runs

$$SR = \frac{N_{sr}}{N_r} \tag{20}$$

where N_{sr} is the number of successful runs. A successful run is defined as a run where all the known roots of an NES are found.

To test the statistical differences among different methods, based on the *RR* and *SR* values, we conducted the multiple-problem Wilcoxon's test and the Friedman's test via KEEL software [47]. In the multiple-problem Wilcoxon's test, $p < 0.05$ means that there is a significant difference between the two compared methods.

B. Methods for Comparison and Their Parameter Settings

RADE is compared with five different methods.

1) *NCDE:* This method is presented in [39]. It is chosen based on two considerations: a) it obtains very promising results for multimodal optimization problems and b) similar to NCDE, the neighborhood mutation and the crowding selection are also adopted in RADE. For NCDE, (3) is considered as the objective function. Through comparing with NCDE, the influence of the repulsion technique and adaptive parameter control in RADE can be evaluated.

2) *R-JADE, R-CLPSO, and I-HS:* They are repulsion-based methods. In repulsion-based JADE (R-JADE) and repulsion-based CLPSO (R-CLPSO), JADE proposed in [43] and CLPSO proposed in [48] are considered as the optimization algorithms, respectively. The reason why the repulsion technique is integrated with JADE and CLPSO is the following: JADE and CLPSO exhibit outstanding performance among different DE variants and particle swarm optimization (PSO) variants, respectively. In [11], I-HS proposed in [49] is combined with the repulsion technique to solve NESs. By comparing with R-JADE, R-CLPSO, and I-HS, we can assess the influence of different optimization algorithms on solving NESs.

3) *MONES:* It is a recent multiobjective optimization-based method [32], which is able to locate multiple roots of NESs. Note that MONES originally uses NSGA-II [50] as the search engine. In this paper, SHADE replaces the crossover and mutation operators as well as the algorithmic parameters in NSGA-II. It is because DE operators are able to achieve better results than the original operators in NSGA-II for multiobjective optimization problems [51].

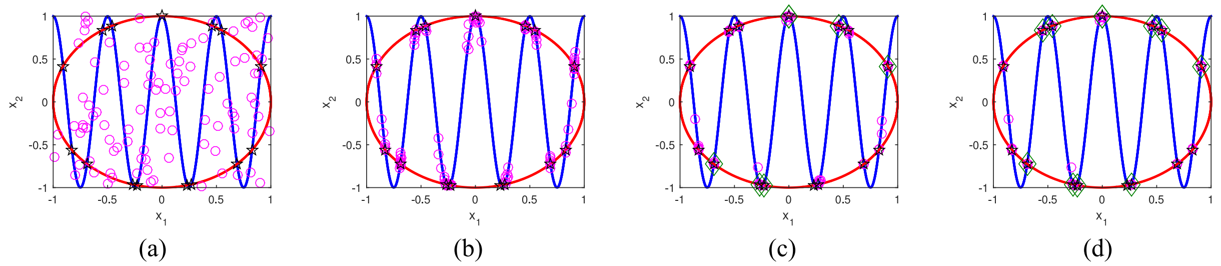

Fig. 3. Evolution of RADE-WoR over a typical run on F03. Circles denote the individuals in the population, pentagrams denote the known roots, and diamonds denote the found roots in the archive. (a) gen = 1. (b) gen = 100. (c) gen = 300. (d) gen = 500.

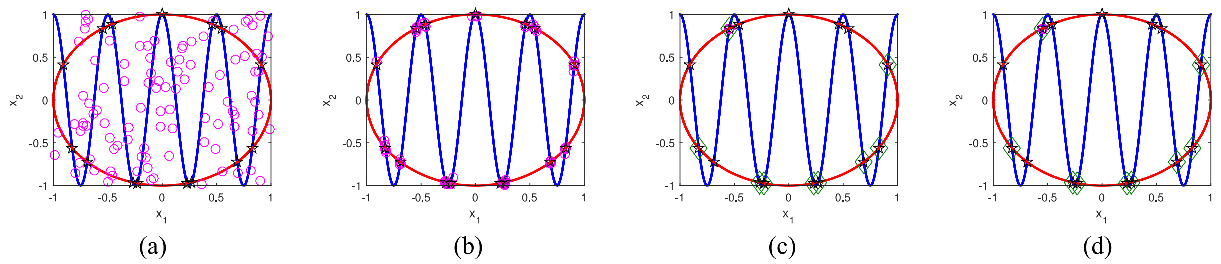

Fig. 4. Evolution of RADE-WoD over a typical run on F03. Circles denote the individuals in the population, pentagrams denote the known roots, and diamonds denote the found roots in the archive. (a) gen = 1. (b) gen = 100. (c) gen = 300. (d) gen = 500.

TABLE II
PARAMETER SETTINGS OF THE SEVEN METHODS USED IN THIS PAPER

RADE	$NP = 100, H_m = 200$
NCDE	$NP = 100, F = 0.9, CR = 0.1$
R-JADE	$NP = 100, \mu_{CR} = 0.5, \mu_F = 0.5$
R-CLPSO	$NP = 100, m = 7, c = 2.0$
I-HS	$NP = 10$, HMCR $= 0.95$, PAR$_{\min} = 0.35$, PAR$_{\max} = 0.99$, BW$_{\min} = 10^{-6}$, BW$_{\max} = 5$
MONES	$NP = 100, H_m = 200$
SaDE	$NP = 50, LP = 50$, CRm$_k = 0.5$

For the six compared methods, the parameter settings are given in Table II.[3] All the parameter settings were kept the same in our experiments unless a change was mentioned. *Max_FEs* is given in the last column of Table I. It is necessary to emphasize that *Max_FEs* was set according to the difficulties of different test problems. Each test problem was optimized over 100 independent runs. For a fair comparison, all of the compared methods started with the same initial population in each of 100 runs.

Note that the repulsion technique in (5) was used in R-JADE, R-CLPSO, I-HS, and RADE. The effect of other repulsion techniques introduced in Section II-A is discussed in Section V-E1.

C. Proof-of-Principle Results

First, we are interested in analyzing the working principle of RADE. For this purpose, we chose the third test problem F03 as an instance and considered three variants of RADE.

1) RADE-WoR, in which the repulsion technique was removed from RADE and $F(\mathbf{x}) = f(\mathbf{x})$ in (14) during the evolution.
2) RADE-WoD, where we eliminated the diversity preservation mechanism from RADE and the classic DE/rand/1/bin was utilized.
3) RADE-WoA, in which the adaptive parameter control of RADE was not used and F and CR were fixed to 0.9 and 0.1 according to the suggestion in [39], respectively.

For F03, since *Max_FEs* $= 50,000$ and *NP* $= 100$, the maximum number of generation is equal to 500. As shown in Table I, F03 contains two decision variables; thus it is easy to monitor the evolutionary status of a method. When solving F03, RADE and its three variants used the same initial population to ensure a fair comparison.

Figs. 3–6 provide the evolution of RADE and its three variants over a typical run on F03. From Figs. 3–6, we can give the following comments.

1) RADE-WoR loses some of the roots. Although some individuals already surround the attraction basins of some roots, they cannot be stored into the archive due to the low precision. Moreover, as shown in Fig. 3, if two roots are very close to each other, RADE-WoR is likely to miss one of them. The reasons of the above phenomena are twofold: a) due to the lack of the repulsion technique, if some roots have been found, the individuals near these roots will continue to search in the attraction basins of these roots, which inevitably wastes a lot of computational resource and results in low precision of other individuals, and b) if two roots are very close and if one of them has been identified, the individuals around the unfound root may be replaced with other individuals surrounding the found root.

[3]As suggested in [39], the neighborhood size ℓ was dynamically controlled as $\ell = 5 + \lfloor 5 \cdot (max\_gen - gen)/max\_gen \rfloor$ for RADE and NCDE, where *gen* and *max_gen* are the current and maximum number of generations, respectively.

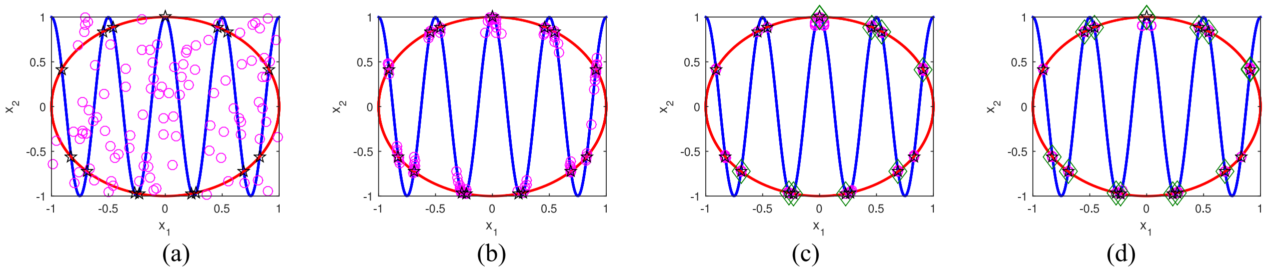

Fig. 5. Evolution of RADE-WoA over a typical run on F03. Circles denote the individuals in the population, pentagrams denote the known roots, and diamonds denote the found roots in the archive. (a) gen = 1. (b) gen = 100. (c) gen = 300. (d) gen = 500.

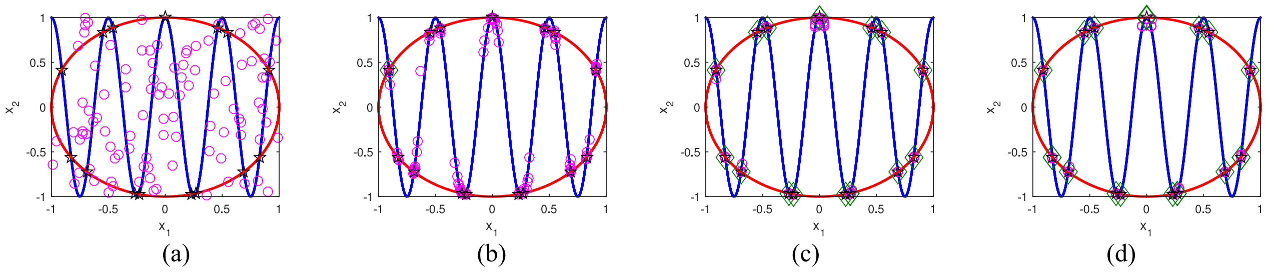

Fig. 6. Evolution of RADE over a typical run on F03. Circles denote the individuals in the population, pentagrams denote the known roots, and diamonds denote the found roots in the archive. (a) gen = 1. (b) gen = 100. (c) gen = 300. (d) gen = 500.

2) It is clear from Fig. 4 that RADE-WoD also fails to locate all the roots. Actually, we can observe that during the evolution, RADE-WoD is able to find some of the roots. It is because in the early and middle stages of evolution, the diversity of the population is good and the repulsion technique can guide the population to find some roots. However, at the end of evolution, all the individuals converge to one of the roots as shown in Fig. 4. It is not difficult to understand since the diversity of the population gradually decreases. Note that if all the individuals already get stuck in one of the roots, it is very difficult for them to jump out from the search region even though the repulsion technique is available, owing to the poor diversity.
3) As depicted in Fig. 5, again, it is a very challenging task for RADE-WoA to find all the roots. This can be explained as follows: the search ability of RADE-WoA is weak because of the unsuitable parameter settings for *F* and *CR*, which results in the low precision of some individuals.
4) From Fig. 6, RADE succeeds in finding all the roots of F03. The success of RADE can be attributed to the facts that: a) the repulsion technique has the capability to avoid searching around the roots found previously and, as a result, increases the possibility to detect more roots; b) the diversity preservation mechanism can maintain the diversity of the population, which is definitely beneficial for finding multiple roots during the evolution; and c) the adaptive parameter control is able to pursue suitable parameters, thus enhancing the performance of RADE.

To further investigate the effectiveness of the three aspects of RADE, we applied RADE and its three variants to solve the 30 NESs. Table S-R-I in the supplementary material summarizes the detailed results. Tables III and IV report the statistical test results based on the multiple-problem Wilcoxon's test and the Friedman's test, respectively. From Table III, RADE achieves higher R^+ values than R^- values in all the cases. Moreover, RADE performs significantly better than RADE-WoD and RADE-WoA because the p-values are less than 0.05. According to the Friedman's test in Table IV, RADE ranks the first in both *RR* and *SR* criteria.

Clearly, the above results verify the motivation of this paper—the repulsion technique, the diversity preservation

TABLE III
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR THE FOUR COMPARED METHODS. IN RADE-WOR, THE REPULSION TECHNIQUE WAS NOT USED; IN RADE-WOD, THE DIVERSITY PRESERVATION MECHANISM WAS NOT USED; AND IN RADE-WOA, THE ADAPTIVE PARAMETER CONTROL WAS NOT USED. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

RADE VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
RADE-WoR	303.5	161.5	1.49E-01	357.0	108.0	**9.30E-03**
RADE-WoD	433.5	31.5	**7.00E-07**	426.0	39.0	**2.84E-06**
RADE-WoA	460.0	5.0	**1.86E-08**	454.5	10.5	**9.13E-08**

TABLE IV
RANKINGS OF THE FOUR COMPARED METHODS BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
RADE	**1.5833**	**1.5833**
RADE-WoR	1.7833	1.9500
RADE-WoD	3.3833	3.2667
RADE-WoA	3.2500	3.2000

TABLE V
AVERAGE *RR* AND *SR* VALUES OF THE SIX COMPARED METHODS FOR ALL TEST PROBLEMS. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	RR	SR
RADE	**0.9491**	**0.8247**
NCDE	0.6799	0.3873
R-JADE	0.7750	0.4943
R-CLPSO	0.6865	0.4057
I-HS	0.7838	0.5163
MONES	0.6675	0.4230

TABLE VI
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR THE SIX COMPARED METHODS. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

RADE VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
NCDE	445.0	20.0	**3.73E-08**	439.5	25.5	**1.82E-07**
R-JADE	422.5	42.5	**5.11E-06**	411.5	53.5	**8.38E-05**
R-CLPSO	424.5	40.5	**3.68E-06**	423.0	42.0	**4.70E-06**
I-HS	387.5	77.5	**9.12E-04**	373.0	92.0	**1.67E-03**
MONES	427.0	38.0	**2.38E-06**	407.0	58.0	**4.70E-05**

mechanism, and the adaptive parameter control are three indispensable elements for RADE to effectively find multiple roots of NESs.

D. Comparison With NCDE, R-JADE, R-CLPSO, I-HS, and MONES

The performance of RADE is compared with that of the five methods introduced in Section V-B. The detailed results in terms of *RR* and *SR* are reported in Table S-R-II and Table S-R-III in the supplementary material, respectively. The average *RR* and *SR* values of the six compared methods are shown in Table V. From Tables S-R-II, S-R-III, and V, it can be seen that RADE provides the highest average *RR* value, i.e., 0.9491 and the highest average *SR* value, i.e., 0.8247. Furthermore, RADE successfully solves 13 out of 30 test problems over 100 runs. In contrast, NCDE, R-JADE, R-CLPSO, I-HS, and MONES successfully solve five, eight, eight, nine, and five test problems over 100 runs, respectively.

The statistical test results obtained by the multiple-problem Wilcoxon's test are reported in Table VI. Additionally, the rankings of the six compared methods derived from the Friedman's test are presented in Table VII. From Table VI, RADE consistently provides significantly better results than NCDE, R-JADE, R-CLPSO, MONES, and I-HS because the p-values are less than 0.05 in all the cases. In addition, RADE has the best ranking as shown in Table VII. Therefore, we can draw the following conclusions.

1) Comparing RADE with NCDE, we can conclude that the repulsion technique and the adaptive parameter control significantly improve the performance of NCDE.
2) Comparing RADE with R-JADE, R-CLPSO, and I-HS, we can conclude that although the repulsion technique can be exploited to find different roots of NESs, the performance of the repulsion-based methods is remarkably influenced by the optimization algorithms.

TABLE VII
RANKINGS OF THE SIX COMPARED METHODS BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
RADE	**1.9000**	**2.0333**
NCDE	4.3833	4.1500
R-JADE	3.4500	3.5500
R-CLPSO	3.7333	3.8000
I-HS	3.3500	3.4167
MONES	4.1833	4.0500

TABLE VIII
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR RADE WITH DIFFERENT REPULSION TECHNIQUES. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

RADE VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
RADE-1	266.0	199.0	≥ 0.2	270.5	194.5	≥ 0.2
RADE-2	319.5	145.5	≥ 0.2	322.5	142.5	≥ 0.2

TABLE IX
RANKINGS OF RADE WITH DIFFERENT REPULSION TECHNIQUES BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
RADE	**1.8333**	**1.8333**
RADE-1	1.9667	1.9833
RADE-2	2.2000	2.1833

The above comparison indicates that, on the whole, the cooperation of the repulsion technique, the diversity preservation mechanism, and the adaptive parameter control is really effective. As a consequence, RADE is able to yield improved performance compared with the five competitors.

E. Discussions on Different Components

As mentioned before, our proposed RADE is a generic framework. This section discusses the effectiveness of different components in RADE.

1) On Other Repulsion Techniques: In Section II-A, three representative repulsion techniques were introduced. In the previous experiments, the repulsion technique in (5) was used. To investigate the effectiveness of other repulsion techniques, we replaced (5) with (7) and (9) in RADE. As a result, two RADE variants are obtained, called RADE-1 and RADE-2.

The detailed results in terms of *RR* and *SR* are provided in Table S-R-IV in the supplementary material. The statistical test results are reported in Tables VIII and IX. From Table IX, we can see that RADE with the repulsion technique in (5) performs the best, followed by RADE-1. However, there is no significant difference between RADE and each of RADE-1 and RADE-2 according to the multiple-problem Wilcoxon's test in Table VIII since $p > 0.05$.

2) On Other Diversity Preservation Mechanisms: In RADE, the crowding selection was used as the selection operator. To study the effectiveness of other diversity preservation mechanisms, the speciation selection [39] was applied to replace the crowding selection in RADE. The resulting

TABLE X
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR THE SIX COMPARED ALGORITHMS. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

RADE-3 VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
RADE	157.0	308.0	$\geq$ 0.2	140.0	325.0	$\geq$ 0.2
NCDE	438.5	26.5	**6.56E-06**	446.5	18.5	**3.34E-05**
R-JADE	393.5	71.5	**1.51E-02**	396.0	69.0	**4.97E-02**
R-CLPSO	396.5	68.5	**1.23E-02**	392.0	73.0	6.74E-02
MONES	426.0	39.0	**2.17E-04**	401.5	63.5	**1.82E-03**

TABLE XI
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR RADE WITH DIFFERENT ADAPTIVE PARAMETER CONTROLS. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

RADE VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
RADE-4	195.0	270.0	$\geq$ 0.2	245.5	219.5	$\geq$ 0.2
RADE-5	356.0	109.0	$\geq$ 0.2	340.5	124.5	$\geq$ 0.2

TABLE XII
RANKINGS OF RADE WITH DIFFERENT ADAPTIVE PARAMETER CONTROLS BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
RADE	**1.8500**	**1.8500**
RADE-4	1.9167	2.0000
RADE-5	2.2333	2.1500

method is referred to as RADE-3. In RADE-3, all other parameter settings were kept the same with RADE. Table S-R-V, in the supplementary material, reports the *RR* and *SR* values of RADE-3. In addition, Table X compares RADE-3 with RADE, NCDE, R-JADE, and MONES according to the multiple-problem Wilcoxon's test, by integrating the experimental results in Tables S-R-II and S-R-III.

Table S-R-V indicates that RADE-3 is slightly worse than RADE in terms of the average *RR* and *SR*. However, from Table X, it can be seen that RADE-3 obtains significantly better performance than NCDE, R-JADE, and MONES. It also significantly outperforms R-CLPSO in terms of *RR* at $p = 0.05$ and in terms of *SR* at $p = 0.1$, respectively. Hence, one can conclude that other advanced diversity preservation mechanism could also be applicable to the RADE framework.

3) On Other Adaptive Parameter Controls: In the DE literature, there are also other techniques for adaptive parameter control, e.g., jDE [52] and JADE [43]. The effectiveness of the techniques in jDE and JADE was investigated by incorporating them into RADE. The resultant methods are denoted as RADE-4 (RADE with jDE's adaptive parameter control) and RADE-5 (RADE with JADE's adaptive parameter control), respectively.

The detailed results are provided in Table S-R-VI in the supplementary material. Additionally, Tables XI and XII report the statistical test results based on the multiple-problem Wilcoxon's test and the Friedman's test, respectively. It is clear from Table XII that RADE yields better ranking than RADE-4 and RADE-5. However, there are no significant differences among these three methods from Table XI.

Remark 3: Based on the above empirical discussions, it seems that there exist some other options for the repulsion techniques, diversity preservation mechanisms, and adaptive parameter controls under our RADE framework, which verifies its robustness.

4) On Different Mutation Operators: There is a variety of mutation operators in the DE community [15]. Generally, different mutation operators are suitable for different problems [53]. In RADE, "DE/rand/1" was originally employed in the neighborhood mutation as shown in (12). We replaced DE/rand/1 with other widely used mutation operators, with the aim of evaluating the effectiveness of different mutation operators for solving NESs. To this end, we tested four RADE variants: 1) RADE-6, i.e., RADE with "DE/rand-to-best/1"; 2) RADE-7, i.e., RADE with "DE/rand/2"; 3) RADE-8, i.e., RADE with "DE/current-to-best/2"; and 4) RADE-9, i.e., RADE with "DE/current-to-rand/1."[4] In addition, since the adaptation technique for mutation operators has been proved to be effective for improving the performance of DE in single-objective optimization problems [53], SaDE, a representative adaptation technique, was also assessed in this section. Note that for the four RADE variants and SaDE, the neighborhood mutation and the crowding selection were also adopted. The parameter settings were kept unchanged for the four RADE variants. For SaDE, the parameter settings were the same as in the original paper [53] and shown in Table II.

TABLE XIII
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR RADE WITH DIFFERENT MUTATION OPERATORS. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

RADE VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
RADE-6	459.5	5.5	**8.94E-08**	454.0	11.0	**1.64E-06**
RADE-7	437.5	27.5	**2.52E-03**	426.0	39.0	**4.32E-02**
RADE-8	461.0	4.0	**1.30E-08**	457.5	7.5	**1.64E-07**
RADE-9	426.5	38.5	**2.98E-04**	435.0	30.0	**9.41E-04**
SaDE	324.0	141.0	$\geq$ 0.2	322.0	143.0	$\geq$ 0.2

Tables S-R-VII and S-R-VIII in the supplementary material report the *RR* and *SR* values, respectively. The statistical test results derived from the multiple-problem Wilcoxon's and Friedman's tests are given in Tables XIII and XIV, respectively. Tables S-R-VII, S-R-VIII, XIII and XIV reveal the following.

a) Mutation operators have a significant effect on the performance of RADE. Overall, DE/rand/1 accomplishes the best results, followed by DE/rand/2 and DE/current-to-rand/1. Whereas, the mutation operators with the best solution drastically degenerate the performance of RADE.

[4]"best" in "DE/rand-to-best/1" and DE/current-to-best/2 means the best individual in the current population based on $F(\mathbf{x})$ in (14). For RADE-6, RADE-7, and RADE-8, the binomial crossover in DE was used, however for RADE-9, the crossover was not used.

TABLE XIV
RANKINGS OF RADE WITH DIFFERENT MUTATION OPERATORS BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
RADE	**1.7333**	**1.9000**
RADE-6	5.4000	4.9667
RADE-7	3.1500	3.2167
RADE-8	4.8833	4.9167
RADE-9	3.6833	3.8500
SaDE	2.1500	2.1500

TABLE XV
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR RADE WITH DIFFERENT *NP* VALUES. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

$NP = 100$ VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
$NP = 50$	323.0	142.0	5.06E-02	309.5	155.5	1.17E-01
$NP = 80$	263.5	201.5	≥ 0.2	233.5	231.5	≥ 0.2
$NP = 120$	286.5	178.5	≥ 0.2	246.0	219.0	≥ 0.2
$NP = 150$	260.5	204.5	≥ 0.2	267.5	197.5	≥ 0.2
$NP = 200$	254.5	210.5	≥ 0.2	280.5	184.5	≥ 0.2

TABLE XVI
RANKINGS OF RADE WITH DIFFERENT *NP* VALUES BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
$NP = 50$	4.3833	4.2667
$NP = 80$	3.5333	3.4167
$NP = 100$	3.3833	3.3500
$NP = 120$	**3.1500**	**3.2167**
$NP = 150$	3.2667	3.2833
$NP = 200$	3.2833	3.4667

b) SaDE provides the second best results in terms of both *RR* and *SR*, which means that mutation operator adaptation could be a good choice when solving different NESs.

F. Discussions on Different Parameter Settings

This section discusses the influence of different parameter settings of *NP*, H_m, and α on the performance of RADE.

1) Effect of Different Population Sizes: In the previous sections, the population size *NP* was fixed to 100, which has been widely adopted in the DE literature [41], [43], [52]. We empirically investigated the effect of population size by testing different *NP* values: 50, 80, 120, 150, and 200. All other parameter settings in RADE were kept the same as in Table II. Tables S-R-IX and S-R-X in the supplementary material report the detailed results of *RR* and *SR*, respectively. In addition, Tables XV and XVI show the statistical results obtained by the multiple-problem Wilcoxon's test and the Friedman's test, respectively.

From Table XVI, we can observe that $NP = 120$ is able to provide the best performance on the whole. The results in Table XV indicate that there are no significant differences among RADE with different population sizes. However, by carefully looking at the results in Tables S-R-IX and S-R-X, we find that RADE with a smaller population size provides better results on F04, F09, F16, and F19; while on F02, F03, F12, F13, F17, and F23, the better results are obtained by RADE with a larger population size. This phenomenon could motivate us to study the dynamic or adaptive control of population size to further improve the performance of RADE. However, it is out of the scope of this paper. We will leave it to our future work. In general, $NP \in [80, 120]$ can yield promising results.

TABLE XVII
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR RADE WITH DIFFERENT H_m VALUES. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

$H_m = 800$ VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
$H_m = 5$	360.0	105.0	≥ 0.2	340.5	124.5	≥ 0.2
$H_m = 10$	372.0	93.0	≥ 0.2	354.5	110.5	≥ 0.2
$H_m = 30$	358.5	106.5	≥ 0.2	322.5	142.5	≥ 0.2
$H_m = 50$	356.5	108.5	≥ 0.2	335.5	129.5	≥ 0.2
$H_m = 100$	377.5	87.5	≥ 0.2	338.5	126.5	≥ 0.2
$H_m = 200$	354.0	111.0	≥ 0.2	314.5	150.5	≥ 0.2
$H_m = 300$	355.5	109.5	≥ 0.2	343.5	121.5	≥ 0.2
$H_m = 400$	311.0	154.0	≥ 0.2	306.0	159.0	≥ 0.2
$H_m = 500$	262.0	203.0	≥ 0.2	235.5	229.5	≥ 0.2
$H_m = 1000$	322.0	143.0	≥ 0.2	317.5	147.5	≥ 0.2

TABLE XVIII
RANKINGS OF RADE WITH DIFFERENT H_m VALUES BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
$H_m = 5$	7.6333	7.2000
$H_m = 10$	7.3167	7.1500
$H_m = 30$	7.1333	6.6500
$H_m = 50$	6.6500	6.5833
$H_m = 100$	6.3833	6.0333
$H_m = 200$	5.7667	5.7500
$H_m = 300$	6.1667	6.1500
$H_m = 400$	5.0333	5.5833
$H_m = 500$	4.4833	**4.7000**
$H_m = 800$	**4.4500**	4.8167
$H_m = 1000$	4.9833	5.3833

2) Influence of Different H_m Values: In RADE, the parameter H_m controls the size of history memory for adaptively updating *F* and *CR* in Section IV-C. In the previous experiments, $H_m = 200$ was used. We tested the following H_m values: 5, 10, 30, 50, 100, 300, 400, 500, 800, and 1000. All other parameter settings were kept unchanged. The detailed results of *RR* and *SR* are reported in Tables S-R-XI and S-R-XII in the supplementary material, respectively. Furthermore, the statistical test results obtained by the multiple-problem Wilcoxon's test are shown in Table XVII and the rankings resulting from the Friedman's test are given in Table XVIII.

Tables S-R-XI and S-R-XII indicate that $H_m = 800$ provides the highest average *RR* and *SR* values. For $H_m > 200$, RADE gets better results in both *RR* and *SR* criteria compared with the default setting $H_m = 200$. However, for $H_m < 200$, the performance degradation occurs as H_m decreases. The reason might be the following: most of NESs in this paper contain many roots, when some of them are found during the previous generations, the fitness function will be altered according to the repulsion function. Subsequently, the optimizer will face a new optimization problem, and the previous suitable parameters recorded in the historical

TABLE XIX
RESULTS OBTAINED BY THE MULTIPLE-PROBLEM WILCOXON'S TEST FOR RADE WITH DIFFERENT α VALUES. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

$\alpha = 1$ VS	RR			SR		
	R^+	R^-	p-value	R^+	R^-	p-value
$\alpha = 2$	296.0	169.0	≥ 0.2	267.5	197.5	≥ 0.2
$\alpha = 5$	298.0	167.0	≥ 0.2	277.0	188.0	≥ 0.2
$\alpha = 10$	279.5	185.5	≥ 0.2	241.0	224.0	≥ 0.2
$\alpha = 20$	342.0	123.0	≥ 0.2	311.5	153.5	≥ 0.2
$\alpha = 50$	326.5	138.5	≥ 0.2	299.5	165.5	≥ 0.2
$\alpha = 100$	324.0	141.0	≥ 0.2	267.0	198.0	≥ 0.2

TABLE XX
RANKINGS OF RADE WITH DIFFERENT α VALUES BY THE FRIEDMAN'S TEST. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Algorithm	Ranking (RR)	Ranking (SR)
$\alpha = 1$	**3.2667**	**3.5833**
$\alpha = 2$	3.8167	3.7333
$\alpha = 5$	4.0667	3.9167
$\alpha = 10$	3.8833	3.7500
$\alpha = 20$	4.4333	4.4667
$\alpha = 50$	4.4000	4.3833
$\alpha = 100$	4.1333	4.1667

memory may be not suitable for the new optimization problem. Therefore, performance seems to be improved as H_m increases. Table XVIII also verifies the above observation: RADE with $H_m \geq 200$ provides better rankings than $H_m < 200$.

Table XVII shows that the performance differences among RADE with different H_m values are not significant compared with $H_m = 800$, which means that the influence of H_m is not significant in RADE. According to Tables S-R-XI, S-R-XII, and XVIII, the value of H_m can be chosen from a large range, e.g., [200, 1000].

3) Influence of Different α Values: The repulsion function in (5) has two parameters ϵ and α. The parameter α, which is used to adjust the radius of the repulsion areas, plays a more important role than ϵ.[5] Therefore, the influence of different α values on the performance of RADE was empirically studied. The detailed results of *RR* and *SR* are shown in Tables S-R-XIII and S-R-XIV in the supplementary material, respectively. Table XIX reports the statistical test results provided by the multiple-problem Wilcoxon's test for RADE with different α values. The rankings obtained by the Friedman's test are described in Table XX.

It can be seen from Tables S-R-XIII, S-R-XIV, and XX that RADE with $\alpha = 1$ achieves the highest average *RR* and *SR* values and the best ranking, followed by $\alpha = 2$. Although the default setting $\alpha = 10$ is not the best, it still obtains acceptable results. RADE with larger α values ($\alpha \geq 20$) performs worse than RADE with smaller α values ($\alpha \leq 10$). The reason can be explained as follows: as mentioned in [9], if α decreases, the radius of the repulsion regions generated by $|\coth(\alpha x)|$ will be enlarged. It means that the penalty term $\prod_{j=1}^{K} |\coth(\alpha\delta_j)|$ in (5) increases with the decrease of α. In this way, the method with smaller α values will pay more attention to search for new roots, due to the higher penalty on the solutions that are closer to the roots found previously.

[5]We also set different ϵ values (e.g., 1E-08, 1E-10, 1E-15, 1E-20, and 0) in RADE, the experimental results show that the influence of ϵ can be negligible. Interested readers can find the detailed results in Tables S-R-XV and S-R-XVI in the supplementary material.

TABLE XXI
EXPERIMENTAL RESULTS OF RADE AND N-M BASED ON 12 TEST PROBLEMS PRESENTED IN [14]. THE ORIGINAL RESULTS IN [14] WERE CONVERTED INTO THE *RR* AND *SR* VALUES BASED ON (19) AND (20), RESPECTIVELY. *RR* IS THE AVERAGE RATIO OF ROOTS FOUND OVER MULTIPLE RUNS, AND *SR* IS THE RATIO OF SUCCESSFUL RUNS

Prob.	RADE		N-M [14]	
	RR	SR	RR	SR
c01	**1.0000**	**1.00**	**1.0000**	**1.00**
c02	**1.0000**	**1.00**	**1.0000**	0.97
c03	**1.0000**	**1.00**	0.9778	0.83
c04	**1.0000**	**1.00**	0.9000	0.80
c05	**1.0000**	**1.00**	**1.0000**	**1.00**
c06	**0.9969**	**0.96**	0.9692	0.70
c07	**1.0000**	**1.00**	0.9571	0.67
c08	**1.0000**	**1.00**	**1.0000**	**1.00**
c09	**0.8733**	**0.67**	0.7333	0.50
c10	0.9300	0.86	**1.0000**	**1.00**
c11	**1.0000**	**1.00**	**1.0000**	**1.00**
c12	**1.0000**	**1.00**	0.9833	0.97
Avg.	**0.9834**	**0.9575**	0.9601	0.8700

From Table XIX, there are no significant performance differences among RADE with different α values. Suggested by the results in Tables S-R-XIII, S-R-XIV, XIX, and XX, the reasonable value of α is between 1 and 10.

G. Comparison With Other Reported Results

In the previous experiments, the superior performance of RADE has been demonstrated through the 30 NESs in Table I. To further understand the performance of RADE, it was compared with other reported results in [14]. Twelve NESs collected in [14] were used in this paper, which are referred to as c01–c12. In [14], the repulsion technique is integrated with the N–M method to find multiple roots of NESs. To make a fair comparison, RADE implemented the same number of fitness evaluations for each test problem as in [14] over 100 independent runs. In addition, the original results in [14] were converted into the *RR* and *SR* values based on (19) and (20), respectively.

The experimental results are summarized in Table XXI. On six out of 12 test problems, RADE provides better results in terms of both *RR* and *SR* criteria. However, N–M outperforms RADE on only one test problem. In addition, the number of test problems which can be successfully solved by RADE and N–M over all 100 runs is nine and six, respectively. Therefore, it can be concluded that RADE is also better than N–M for solving NESs.

VI. CONCLUSION

Solving NESs is a very important area in numerical computation. Very often, NESs contain multiple roots. However, it is a very challenging task to find multiple roots of NESs simultaneously in a single run. To address this issue, in this paper, we proposed a repulsion-based adaptive DE, called RADE, in which the repulsion technique, the diversity preservation mechanism, and the adaptive parameter control were combined to solve NESs effectively. The performance of RADE was evaluated by 30 NESs selected from the literature.

The experimental results suggested that RADE is able to find multiple roots simultaneously in a single run on all the test problems. It can also provide very competitive performance compared with other well-established methods. In addition, we carried out extensive experiments to systematically analyze the working principle of RADE, as well as the effectiveness of different components of RADE and the influence of parameter settings. According to the experiments, we demonstrated that the repulsion technique, the diversity preservation mechanism, and the adaptive parameter control are three indispensable elements of RADE, which verifies the motivation of this paper.

For the repulsion techniques introduced in Section II-A, several parameters need to be set properly. In the future, we will attempt to design dynamic or adaptive parameter controls for the repulsion techniques, such as the dynamic or adaptive radius of the repulsion regions. Additionally, we will apply RADE to deal with complex real-world NESs.

The source code of RADE can be obtained from W. Gong's homepage: http://www.escience.cn/people/wygong or Y. Wang's homepage: http://www.escience.cn/people/yongwang1.

Acknowledgment

The authors would like to thank Prof. B. Y. Qu and Prof. P. N. Suganthan for kindly providing the source code of NCDE.

References

[1] J. Pintér, "Solving nonlinear equation systems via global partition and search: Some experimental results," *Computing*, vol. 43, no. 4, pp. 309–323, 1990.

[2] D. Mehta and C. Grosan, "A collection of challenging optimization problems in science, engineering and economics," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, 2015, pp. 2697–2704.

[3] D. Guo, Z. Nie, and L. Yan, "The application of noise-tolerant ZD design formula to robots' kinematic control via time-varying nonlinear equations solving," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2017.2705160.

[4] S. Smale, "Mathematical problems for the next century," *Math. Intell.*, vol. 20, no. 2, pp. 7–15, 1998.

[5] Y. Shi and Y. Zhang, "New discrete-time models of zeroing neural network solving systems of time-variant linear and nonlinear inequalities," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2017.2751259.

[6] C. A. Floudas, "Recent advances in global optimization for process synthesis, design and control: Enclosure of all solutions," *Comput. Chem. Eng.*, vol. 23, pp. S963–S973, Jun. 1999.

[7] M. J. Hirsch, P. M. Pardalos, and M. G. C. Resende, "Solving systems of nonlinear equations with continuous GRASP," *Nonlin. Anal. Real World Appl.*, vol. 10, no. 4, pp. 2000–2006, 2009.

[8] N. Henderson, W. F. Sacco, and G. M. Platt, "Finding more than one root of nonlinear equations via a polarization technique: An application to double retrograde vaporization," *Chem. Eng. Res. Design*, vol. 88, nos. 5–6, pp. 551–561, 2010.

[9] E. Pourjafari and H. Mojallali, "Solving nonlinear equations systems with a new approach based on invasive weed optimization algorithm and clustering," *Swarm Evol. Comput.*, vol. 4, pp. 33–43, Jun. 2012.

[10] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[11] G. C. Ramadas, E. M. Fernandes, and A. A. Rocha, "Multiple roots of systems of equations by repulsion merit functions," in *Computational Science and its Applications—ICCSA 2014* (LNCS 8580), B. Murgante *et al.*, Eds. Springer, 2014, pp. 126–139.

[12] N. Henderson, L. Freitas, and G. M. Platt, "Prediction of critical points: A new methodology using global optimization," *AIChE J.*, vol. 50, no. 6, pp. 1300–1314, 2004.

[13] K. E. Parsopoulos and M. N. Vrahatis, "On the computation of all global minimizers through particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 3, pp. 211–224, Jun. 2004.

[14] G. C. V. Ramadas, A. M. A. C. Rocha, and E. M. G. P. Fernandes, "Testing Nelder–Mead based repulsion algorithms for multiple roots of nonlinear systems via a two-level factorial design of experiments," *PLoS ONE*, vol. 10, no. 4, 2015, Art. no. e0121844.

[15] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[16] S. Y. Chen and M. H. Song, "Energy-saving dynamic bias current control of active magnetic bearing positioning system using adaptive differential evolution," *IEEE Trans. Syst., Man, Cybern., Syst.*, to be published, doi: 10.1109/TSMC.2017.2691304.

[17] W. Gong, Z. Cai, and D. Liang, "Adaptive ranking mutation operator based differential evolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 716–727, Apr. 2015.

[18] P. Rakshit, A. Konar, S. Das, L. C. Jain, and A. K. Nagar, "Uncertainty management in differential evolution induced multiobjective optimization in presence of measurement noise," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 7, pp. 922–937, Jul. 2014.

[19] Y. Wang, H.-X. Li, G. G. Yen, and W. Song, "MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 830–843, Apr. 2015.

[20] C. D. Maranas and C. A. Floudas, "Finding all solutions of nonlinearly constrained systems of equations," *J. Glob. Optim.*, vol. 7, no. 2, pp. 143–182, 1995.

[21] S. K. Rahimian, F. Jalali, J. D. Seader, and R. E. White, "A new homotopy for seeking all real roots of a nonlinear equation," *Comput. Chem. Eng.*, vol. 35, no. 3, pp. 403–411, 2011.

[22] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed. New York, NY, USA: Cambridge Univ. Press, 1992.

[23] H. Ramos and M. T. T. Monteiro, "A new approach based on the Newton's method to solve systems of nonlinear equations," *J. Comput. Appl. Math.*, vol. 318, pp. 3–13, Jul. 2017.

[24] K. S. Gritton, J. D. Seader, and W.-J. Lin, "Global homotopy continuation procedures for seeking all roots of a nonlinear equation," *Comput. Chem. Eng.*, vol. 25, nos. 7–8, pp. 1003–1019, 2001.

[25] D. Mehta, "Finding all the stationary points of a potential-energy landscape via numerical polynomial-homotopy-continuation method," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 84, Aug. 2011, Art. no. 025702.

[26] H. Schwandt, "Parallel interval Newton-like Schwarz methods for almost linear parabolic problems," *J. Comput. Appl. Math.*, vol. 199, no. 2, pp. 437–444, 2007.

[27] C.-Y. Chen, "A performance comparison of the zero-finding by extended interval Newton method for Peano monosplines," *Appl. Math. Comput.*, vol. 219, no. 12, pp. 6919–6930, 2013.

[28] M. D. Stuber, V. Kumar, and P. I. Barton, "Nonsmooth exclusion test for finding all solutions of nonlinear equations," *BIT Numer. Math.*, vol. 50, no. 4, pp. 885–917, 2010.

[29] I. G. Tsoulos and A. Stavrakoudis, "On locating all roots of systems of nonlinear equations inside bounded domain using global optimization methods," *Nonlin. Anal. Real World Appl.*, vol. 11, no. 4, pp. 2465–2471, 2010.

[30] W. F. Sacco and N. Henderson, "Finding all solutions of nonlinear systems using a hybrid metaheuristic with fuzzy clustering means," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5424–5432, 2011.

[31] C. Grosan and A. Abraham, "A new approach for solving nonlinear equations systems," *IEEE Trans. Syst., Man, Cybern. A, Syst., Humans*, vol. 38, no. 3, pp. 698–714, May 2008.

[32] W. Song, Y. Wang, H.-X. Li, and Z. Cai, "Locating multiple optimal solutions of nonlinear equation systems based on multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 414–431, Jun. 2015.

[33] S. Qin, S. Zeng, W. Dong, and X. Li, "Nonlinear equation systems solved by many-objective Hype," in *Proc. IEEE Congr. Evol. Comput.*, Sendai, Japan, 2015, pp. 2691–2696.

[34] W. Gong, Y. Wang, Z. Cai, and S. Yang, "A weighted biobjective transformation technique for locating multiple optimal solutions of nonlinear equation systems," *IEEE Trans. Evol. Comput.*, vol. 21, no. 5, pp. 697–713, Oct. 2017.

[35] R. M. A. Silva, M. G. C. Resende, and P. M. Pardalos, "Finding multiple roots of a box-constrained system of nonlinear equations with a biased random-key genetic algorithm," *J. Glob. Optim.*, vol. 60, no. 2, pp. 289–306, 2013.

[36] W. Hu, G. G. Yen, and G. Luo, "Many-objective particle swarm optimization using two-stage strategy and parallel cell coordinate system," *IEEE Trans. Cybern.*, vol. 47, no. 6, pp. 1446–1459, Jun. 2017.

[37] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surveys*, vol. 45, no. 3, pp. 1–33, 2013.

[38] C. Segura, C. A. C. Coello, E. Segredo, and A. H. Aguirre, "A novel diversity-based replacement strategy for evolutionary algorithms," *IEEE Trans. Cybern.*, vol. 46, no. 12, pp. 3233–3246, Dec. 2016.

[39] B. Y. Qu, P. N. Suganthan, and J. J. Liang, "Differential evolution with neighborhood mutation for multimodal optimization," *IEEE Trans. Evol. Comput.*, vol. 16, no. 5, pp. 601–614, Oct. 2012.

[40] Q. Fan and X. Yan, "Self-adaptive differential evolution algorithm with zoning evolution of control parameters and adaptive mutation strategies," *IEEE Trans. Cybern.*, vol. 46, no. 1, pp. 219–232, Jan. 2016.

[41] R. Tanabe and A. Fukunaga, "Success-history based parameter adaptation for differential evolution," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Cancún, Mexico, 2013, pp. 71–78.

[42] R. Tanabe and A. Fukunaga, "Evaluating the performance of SHADE on CEC 2013 benchmark problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Cancún, Mexico, 2013, pp. 1952–1959.

[43] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[44] C. Wang, R. Luo, K. Wu, and B. Han, "A new filled function method for an unconstrained nonlinear equation," *J. Comput. Appl. Math.*, vol. 235, no. 6, pp. 1689–1699, 2011.

[45] I. Z. Emiris and B. Mourrain, "Computer algebra methods for studying and computing molecular conformations," *Algorithmica*, vol. 25, nos. 2–3, pp. 372–402, 1999.

[46] X. Li, A. Engelbrecht, and M. G. Epitropakis, "Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization," Evolution. Comput. Mach. Learn. Group, RMIT Univ., Melbourne, VIC, Australia, Rep., 2013.

[47] J. Alcalá-Fdez *et al.*, "KEEL: A software tool to assess evolutionary algorithms for data mining problems," *Soft Comput.*, vol. 13, no. 3, pp. 307–318, 2009.

[48] J. J. Liang, A. K. Qin, P. N. Suganthan, and S. Baskar, "Comprehensive learning particle swarm optimizer for global optimization of multimodal functions," *IEEE Trans. Evol. Comput.*, vol. 10, no. 3, pp. 281–295, Jun. 2006.

[49] M. Mahdavi, M. Fesanghary, and E. Damangir, "An improved harmony search algorithm for solving optimization problems," *Appl. Math. Comput.*, vol. 188, no. 2, pp. 1567–1579, 2007.

[50] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[51] H. Li and Q. Zhang, "Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 284–302, Apr. 2009.

[52] 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 Trans. Evol. Comput.*, vol. 10, no. 6, pp. 646–657, Dec. 2006.

[53] A. K. Qin, V. L. Huang, and P. N. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 398–417, Apr. 2009.

Wenyin Gong received the B.Eng., M.Eng., and Ph.D. degrees in computer science from the China University of Geosciences, Wuhan, China, in 2004, 2007, and 2010, respectively.

He is currently a Professor with the School of Computer Science, China University of Geosciences. He has published over 50 research papers in journals and international conferences. His current research interests include evolutionary algorithms and evolutionary optimization and their applications.

Dr. Gong served as a Referee for over 30 international journals, such as the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, IEEE TRANSACTIONS ON CYBERNETICS, *IEEE Computational Intelligence Magazine*, *ACM Transactions on Intelligent Systems and Technology*, *Information Sciences*, *European Journal of Operational Research*, *Applied Soft Computing*, and *International Journal of Hydrogen Energy*.

Yong Wang (M'08–SM'17) received the B.S. degree in automation from the Wuhan Institute of Technology, Wuhan, China, in 2003 and the M.S. degree in pattern recognition and intelligent systems and the Ph.D. degree in control science and engineering from Central South University (CSU), Changsha, China, in 2006 and 2011, respectively.

He is currently a Professor with the School of Information Science and Engineering, CSU. His current research interests include the theory, algorithm design, and interdisciplinary applications of computational intelligence.

Dr. Wang was a recipient of the Hong Kong Scholar by the Mainland-Hong Kong Joint Post-Doctoral Fellows Program, China, in 2013, the Excellent Doctoral Dissertation by Hunan Province, China, in 2013, the New Century Excellent Talents in University by the Ministry of Education, China, in 2013, the 2015 IEEE Computational Intelligence Society Outstanding Ph.D. Dissertation Award, the Hunan Provincial Natural Science Fund for Distinguished Young Scholars, in 2016, the EU Horizon 2020 Marie Sklodowska-Curie Fellowship, in 2016, and a Highly Cited Researcher Award in computer science by Clarivate Analytics, in 2017. He is currently serving as an Associate Editor for *Swarm and Evolutionary Computation*.

Zhihua Cai received the B.Sc. degree from Wuhan University, Wuhan, China, in 1986, the M.Sc. degree from the Beijing University of Technology, Beijing, China, in 1992, and the Ph.D. degree from the China University of Geosciences, Wuhan, in 2003.

He is currently a Faculty Member with the School of Computer Science, China University of Geosciences. His current research interests include data mining, machine learning, and evolutionary computation and their applications. He has published over 100 research papers.

Ling Wang received the B.Sc. degree in automation and the Ph.D. degrees in control theory and control engineering from Tsinghua University, Beijing, China, in 1995 and 1999, respectively.

Since 1999, he has been with the Department of Automation, Tsinghua University, where he became a Full Professor in 2008. He has authored five academic books and over 260 refereed papers. His current research interests include intelligent optimization and production scheduling.

Prof. Wang was a recipient of the National Natural Science Fund for Distinguished Young Scholars of China, the National Natural Science Award (Second Place) in 2014, the Science and Technology Award of Beijing City in 2008, the Natural Science Award (First Place in 2003, and Second Place in 2007) nominated by the Ministry of Education of China. He was a Reviewer for the IEEE TRANSACTIONS ON AUTOMATION SCIENCE AND ENGINEERING, *IEEE Computational Intelligence Magazine*, IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION, and IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS. He is an Associate Editor of *Swarm and Evolutionary Computation*, and an Editorial Board Member for several journals, including *Memetic Computing* and the *Journal of Optimization*.