A New Differential Evolution Algorithm for Minimax Optimization in Robust Design

Xin Qiu, Jian-Xin Xu, *Fellow, IEEE*, Yinghao Xu, and Kay Chen Tan, *Fellow, IEEE*

***Abstract*—Minimax optimization, which is actively involved in numerous robust design problems, aims at pursuing the solutions with best worst-case performances. Although considerable research has been devoted to the development of minimax optimization algorithms, there still exist several fundamental limitations for existing approaches, e.g., restriction on problem types, excessively high computational cost, and low optimization efficiency. To address these issues, a minimax differential evolution algorithm is proposed in this paper. First, a novel bottom-boosting scheme enables the algorithm to identify the promising solutions in a reliable yet efficient manner. After that, a partial-regeneration strategy together with a new mutation operator contribute to an in-depth exploration over solution space. Finally, a proper integration of these newly proposed mechanisms leads to an algorithmic structure that can appropriately handle various types of problems. Empirical comparison with seven famous methods demonstrates the statistical superiority of the proposed algorithm. Successful applications in two open problems of robust design further validate the effectiveness of the new approach.**

***Index Terms*—Differential evolution (DE), evolutionary algorithm (EA), minimax optimization problem, robust design.**

I. Introduction

MINIMAX optimization problems, in which the solutions with best worst-case performances are preferred, can be utilized to model a wide variety of robust design tasks [1]–[8]. Unlike traditional optimization problems, two decision spaces are introduced simultaneously in minimax optimization, namely, solution space $\mathbb{X}$ and scenario space $\mathbb{S}$. For each solution in $\mathbb{X}$, its fitness depends on the corresponding worst-case scenario in $\mathbb{S}$. Thus, solving a minimax optimization problem involves explorations and communications among two interrelated spaces. This makes minimax optimization problems inherently challenging and difficult to address [9].

Manuscript received May 24, 2016; revised December 15, 2016 and March 2, 2017; accepted April 2, 2017. Date of publication April 24, 2017; date of current version April 13, 2018. This paper was recommended by Associate Editor F. Herrera. *(Corresponding author: Xin Qiu.)*

X. Qiu is with the NUS Graduate School for Integrative Sciences and Engineering, National University of Singapore, Singapore 119077 (e-mail: qiuxin@u.nus.edu).

J.-X. Xu and Y. Xu are with the Department of Electrical and Computer Engineering, National University of Singapore, Singapore 119077.

K. C. Tan is with the Department of Computer Science, City University of Hong Kong, Hong Kong.

This paper has supplementary downloadable multimedia material available at http://ieeexplore.ieee.org provided by the authors.

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/TCYB.2017.2692963

In the past 40 years, considerable research has been devoted to resolution of minimax optimization problems. Traditional approaches include branch-and-bound algorithms [10]–[12], mathematical programming [5], [13]–[18], and approximation methods [19]. The most significant limitation of these conventional approaches is that additional restrictions will be imposed on the objective functions [20], e.g., finite possibilities in scenario space [21], differentiability [22], and linearity [13].

Recently, a good volume of evolutionary algorithms (EAs) have been developed to solve minimax optimization problems [23]–[31] where traditional methods are not applicable. The main advantage of EAs is that ideally no prior assumption is made about the underlying objective landscape. This generality allows EAs to handle those mathematically intractable problems, which may involve nondifferentiable, nonlinear, nonconvexity or even more complex properties. Although promising performances were observed in some works [20], [26], [32], [33], there do exist several fundamental drawbacks for existing minimax optimization EAs. First, many of the existing methods can only work properly under a symmetrical condition [9], [24]–[29], [32], [33], which means the original minimax optimization problem is equivalent to a symmetrical maximin problem (see Section II-A for detailed definition). However, this condition does not hold generally, and problems not satisfying this condition are known to be extraordinarily difficult to solve [31]. Second, since most of the computational budget is assigned to the search of worst-case scenario for each solution, the overall optimization efficiency is deteriorated. While too much effort is spent on exploration in scenario space [20], [24]–[29], the optimization over solution space may be insufficient. This will lead to a bias toward reliability at the cost of quality. Third, while most existing algorithms focus on elaborating the evaluation mechanisms for scenarios, the evaluation criteria for solutions are underdeveloped. Either expensive computational cost or omission of excellent solutions may result from current evaluation approaches [9], [20], [24]–[29], [32], [33].

To circumvent the above issues, a novel minimax differential evolution (MMDE) algorithm is proposed by completely redesigning the whole algorithmic structure of conventional differential evolution (DE) [34], which has many inherent advantages compared to other EAs [35], e.g., it is simple and straightforward to implement; it exhibits good performance on a wide variety of problems and optimization competitions [35]–[46]; the number of control parameters in DE is small and the space complexity is low. In MMDE, a new bottom-boosting scheme is performed by iteratively updating the associated scenario of current best-performing individual. This scheme allows the algorithm to identify the promising solutions accurately while skipping considerable numbers of unnecessary objective function evaluations. Moreover, a partial-regeneration strategy and a new DE mutation operator are introduced to enhance both reliability and efficiency of the solution optimization. Finally, with a proper combination of these algorithmic components, MMDE is able to overcome the limitations of existing algorithms in solving asymmetrical problems. Superior performance of MMDE is observed in empirical comparisons with other famous minimax optimization EAs. In addition, two open problems in robust design area are formulated into minimax optimization tasks, and both of them are successfully solved by applying the proposed MMDE.

The remainder of this paper is organized as follows. Section II introduces the basic concepts of minimax optimization problems and reviews the existing minimax optimization EAs. A brief description of the original DE algorithm is also provided. Section III explains the technical details of the proposed algorithm and discusses the underlying rationales. Section IV studies the performance and parametric sensitivities of MMDE via experiments. Section V applies the proposed algorithm to solve two practical design problems. Section VI concludes this paper and suggests some future research directions.

II. Background

A. Minimax Optimization Problem

In general, a minimax optimization problem is represented as

$$\min_{X\in\mathbb{X}}\max_{S\in\mathbb{S}} f(X,S) \tag{1}$$

where $f(X,S)$ is the objective function, X is a solution selected from solution space $\mathbb{X}$, and S is a scenario selected from scenario space $\mathbb{S}$. For each X, its cost is measured by maximizing the objective function over scenario space $\mathbb{S}$, that is, to search for the worst-case cost of X. Subsequently, the problem aims at minimizing the worst-case cost over solution space $\mathbb{X}$. An optimal solution for this problem is the one that provides the best worst-case performance.

The problem is defined as symmetrical if

$$\min_{X\in\mathbb{X}}\max_{S\in\mathbb{S}} f(X,S) = \max_{S\in\mathbb{S}}\min_{X\in\mathbb{X}} f(X,S). \tag{2}$$

This statement is a sufficient and necessary condition [47] for the existence of $X^* \in \mathbb{X}$ and $S^* \in \mathbb{S}$ such that

$$f\big(X^*,S\big) \le f\big(X^*,S^*\big) \le f\big(X,S^*\big) \tag{3}$$

for $\forall X \in \mathbb{X}$ and $\forall S \in \mathbb{S}$. According to [20], this implies that there exists a scenario $S^* \in \mathbb{S}$ such that

$$\min_{X\in\mathbb{X}}\max_{S\in\mathbb{S}} f(X,S) = \min_{X\in\mathbb{X}} f\big(X,S^*\big). \tag{4}$$

This symmetrical condition will substantially simplify the original problem because now it is possible to reach the global optima by optimizing over solution space $\mathbb{X}$ and scenario space $\mathbb{S}$ separately. For most existing algorithms [9], [24]–[28], this symmetrical condition is necessary because they evaluate the fitness of each solution based on the same scenario set.

However, most minimax optimization problems do not satisfy condition (2), and this type of problems are defined as asymmetrical. For asymmetrical problems, no single scenario is eligible for evaluating all the solutions, thereby leading to essential difficulty for most optimization approaches.

B. Evolutionary Algorithms for Minimax Optimization

When mathematically intractable properties are involved in the minimax optimization problems, EAs are promising replacements for traditional deterministic approaches. In recent years, a good number of algorithms have been proposed by EA community for handling minimax optimization problems.

Following the consideration that two decision spaces are explored simultaneously in minimax optimization, it is natural for researchers to apply coevolutionary algorithms, in which two populations are maintained during the optimization process. Herrmann [9] proposed a two-space genetic algorithm (GA) for solving minimax optimization problems. Two populations, representing solutions and scenarios, respectively, are evolved in parallel. An individual in one population is evaluated with respect to all the individuals in the other population. More specifically, the cost of a solution X is given by

$$h(X) = \max_{S\in P_S} f(X,S) \tag{5}$$

where P_S represents the scenario population. The cost of a scenario S is calculated as

$$g(S) = \min_{X\in P_X} f(X,S) \tag{6}$$

where P_X represents the solution population. The algorithm then evolves the two populations concurrently using traditional GA operators. Similarly, Barbosa [28] proposed another coevolutionary GA using the above fitness evaluation (FE) mechanisms. The only modification is that the two populations will be optimized alternately, which means the algorithm will evolve one population for several generations while fixing the other population. In [24], [26], and [27], three algorithms were introduced by replacing the GA operators in Herrmann's framework [9] with evolution strategies, evolutionary programming, and particle swarm optimization (PSO). The most significant limitation for these methods is that they will fail on asymmetrical problems [32], which is more common than symmetrical problems in real-world applications.

In order to better handle the asymmetrical situations, various types of fitness assignment methods were developed for evaluating scenarios in coevolutionary EAs. Hur *et al.* [33] proposed the best remaining strategy wherein the fitness of a scenario is decided by the rank of corresponding solution that it performs best. Jensen [32] designed a rank-based evaluation method by assigning fitness based on the maximum rank a scenario can achieve against any solution. An evaluation scheme based on Stackelberg strategy was suggested in [20]. Four new ranking-based methods were introduced in [29], namely, worst case method, average greedy method, distance greedy method and ranking greedy method. All these approaches aim at properly addressing the asymmetrical problems by improving the scenario evaluation strategies. However, as we will analyze in the next section, due to the intrinsic limitations of coevolution-based frameworks, the existing solution evaluation mechanisms will lead to inevitable problems in face of asymmetrical conditions. Another issue with the coevolutionary algorithms is the prohibitively expensive evaluation for each solution and scenario. If the population size is N, then N times of objective function evaluations are required to determine the fitness of one single solution or scenario. This will severely impair the overall optimization performance given a limited computational budget.

Apart from coevolutionary approaches, a small number of noncoevolutionary EAs were also proposed for tackling minimax optimization problems. Laskari *et al.* [23] modified traditional PSO algorithm by evaluating each solution over all the possible scenarios. This algorithm is only applicable to discrete problems and the computational cost is too high. Zhou and Zhang [30] proposed a surrogate-assisted EA for minimax optimization. A surrogate model based on Gaussian process is built to regress the fitness function for solution space. One drawback with this method is that a large number of FEs are required to approximate the true objective function, therefore the overall computational cost does not decrease. The difficulty for building reliable models may also increase dramatically when the problems become more complex. Cramer *et al.* [20] proposed an aging sampled GA (ASGA) that can handle both symmetrical and asymmetrical problems properly. The fitness of a solution is based on the worst performance over all the scenarios it has been tested so far, and all these scenarios are sampled randomly. Except for fitness, the concept of age is introduced as a second criterion during the survival selection. Experimental results showed that ASGA could outperform most state-of-the-art EAs in solving different types of minimax optimization problems. However, since all the tested scenarios are simply sampled randomly, it may take great numbers of trials for one solution to reach its worst-case scenario, and poor solutions may replace good solutions because of "lucky" samplings. The overall optimization efficiency will then be diminished.

Based on the above literature review, there are several unsolved issues with the existing minimax optimization EAs. A new algorithm that can overcome all the aforementioned limitations is desirable.

C. Classical DE Algorithm

1) Initialization: The first step of DE is the initialization of the population of N and D over the search space, where N denotes the population size and D denotes the variable dimensions. We symbolize each individual by $X_{i,g} = (x^1_{i,g}, x^2_{i,g}, \ldots, x^D_{i,g})$, where $i = 1, 2, \ldots, N$, $g = 0, 1, \ldots, G_{\max}$ and $G_{\max}$ denotes the maximum number of generations. Furthermore, let us define the lower search bound as $X_{\min} = (x^1_{\min}, x^2_{\min}, \ldots, x^D_{\min})$ and the upper search bound as $X_{\max} = (x^1_{\max}, x^2_{\max}, \ldots, x^D_{\max})$. Finally, the initial value of the ith individual is generated as

$$x^j_{i,0} = x^j_{\min} + \text{rand}(0,1) \cdot \left(x^j_{\max} - x^j_{\min}\right), j = 1, 2, 3, \ldots, D \tag{7}$$

where rand(0, 1) is a uniform random number on the interval [0, 1] and independently generated for each i and each j.

2) Mutation: In this step, each individual will generate a new individual, called the mutant vector $V_{i,g}$. One most frequently used mutation strategy is

$$V_{i,g} = X_{r_1,g} + F \cdot \left(X_{r_2,g} - X_{r_3,g}\right) \tag{8}$$

where indices r_1, r_2, and r_3 are randomly selected integers from $\{1, 2, \ldots, N\}$ that are distinct from i and mutually different. $F \in [0, 1]$ is a real parameter, called the scaling factor.

3) Crossover: After mutation, crossover operation is employed to generate the trial vectors $U_{i,g}$. During crossover, mutant vectors are recombined with the original members of the current population, called target vectors, to form trial vectors. Traditional binomial recombination is performed on each variable and it could be outlined as

$$u^j_{i,g} = \begin{cases} v^j_{i,g}, & \text{if rand}(0,1) \le \text{Cr or } j = j_{\text{rand}} \\ x^j_{i,g}, & \text{otherwise} \end{cases} \tag{9}$$

where $j_{\text{rand}} \in \{1, 2, 3, \ldots, D\}$ is a randomly selected index to ensure that the trial vector could get at least one component from the mutant vector. Cr is called the crossover probability.

4) Selection: Selection is the last step to generate the population of next generation. The process of selection is to determine whether the target vector or the trial vector survives to the next generation according to their fitness value. The selection operation in DE is described as

$$X_{i,g+1} = \begin{cases} U_{i,g}, & \text{if } f\left(U_{i,g}\right) \le f\left(X_{i,g}\right) \\ X_{i,g}, & \text{otherwise} \end{cases} \tag{10}$$

where $f(X)$ is the fitness function to be minimized.

III. Minimax Differential Evolution

A. Overview

During the development of MMDE, three main limitations of existing minimax optimization EAs are considered and addressed.

The first issue is the incapability of most approaches, especially coevolutionary EAs, in solving asymmetrical problems. Take the following two-plane function [32] as an example:

$$f(x, s) = \min\left\{3 - \frac{2}{10}x + \frac{3}{10}s, 3 + \frac{2}{10}x - \frac{1}{10}s\right\}. \tag{11}$$

Our target is to solve the following minimax optimization problem:

$$\min_{x \in [0,10]} \max_{s \in [0,10]} f(x, s). \tag{12}$$

Fig. 1 shows the surface plot for (11). In coevolutionary EAs, the solution population and scenario population are optimized interactively. No matter what evaluation strategy is employed,

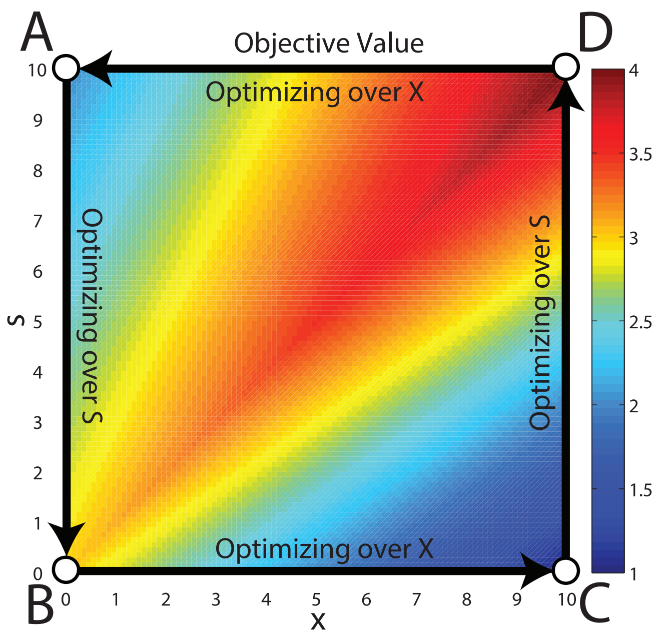


Fig. 1. Surface of objective function described by (11). The color level represents the objective value. The four arrows demonstrate the optimization cycle that will occur during existing coevolutionary algorithms.

the purpose for scenario optimization is always to find the worst-case scenarios for the individuals in solution population. Analogously, the target of solution optimization is always to obtain the best-performing solutions under the scenarios maintained in scenario population. According to Fig. 1, this objective function has four corners that may lead to optimization convergence, namely, $A(0, 10)$, $B(0, 0)$, $C(10, 0)$, and $D(10, 10)$. Suppose all the individuals in solution population and scenario population are randomly dispersed during the initialization, then in the solution optimization phase, all the individuals in solution population will have an evolutionary pressure to move to either 0 or 10. This will drive the solution-scenario combinations to distribute along the edge AB or CD. Similarly, in the scenario optimization process, the solution-scenario combinations tend to move to the diagonal BD, which can represent the worst-case situations for all the solutions. Based on the above analysis, the attractive areas overlap at points B and D. However, the solutions located in B and D are still under evolutionary pressure to move toward C and A, respectively. After that, the scenarios located in C and A will be optimized toward D and B, respectively. It can be observed that actually there is an endless optimization cycle among A, B, C, and D. The existence of this cycle is the fundamental cause for the failures of coevolutionary EAs in solving asymmetrical problems. It is notable that although some coevolutionary algorithms were claimed to successfully handle asymmetrical problems by modifying the evaluation schemes for scenarios [29], [32], [33], none of these approaches are able to avoid the above issue. The reason is that in all coevolutionary EAs, the essence of the optimization over solution space is searching for the best-performing solutions under certain scenarios rather than finding the solutions with best worst-case performance. Even though the algorithm successfully converges to the global optima, which is point $B(0, 0)$, the optimization over solution space will move the solutions away from 0, since solution $X = 10$ gives a lower cost than $X = 0$ under scenario $S = 0$. Our theoretical analysis was verified by the empirical results in [20], which show that all the coevolutionary EAs even perform worse than a purely random search in solving asymmetrical problems. To overcome this issue, MMDE introduced a totally new algorithmic structure to avoid any optimization cycle. The solution optimization process in MMDE will only focus on moving the solutions toward the regions that provide better worst-case performance.

The second issue is the excessive search for worst-case scenarios, which may lead to imbalance between solution reliability and solution quality. In existing approaches, all the solutions are treated equally and same efforts are spent on exploring their corresponding worst-case scenarios. However, a great number of FEs may be wasted in finding the worst-case scenarios for poor solutions, thereby degrading the overall optimization efficiency. In MMDE, the exploration of worst-case scenarios will be performed in a more flexible manner. A bottom-boosting scheme is proposed to skip large quantities of unnecessary scenario evaluations.

The third issue with existing algorithms is the difficulty in selecting promising solutions both inexpensively and correctly. A greedy evaluation strategy is able to avoid the miss of good solutions, but the computational cost is extremely high. A low-cost evaluation strategy can substantially reduce the numbers of objective function evaluation while it is at the risk of propagating poor solutions. In order to trigger a delicate tradeoff between efficiency and reliability, a partial-regeneration strategy and a new DE mutation operator are developed in MMDE. Together with bottom-boosting scheme, the combination of these new mechanisms enables the algorithm to efficiently identify the truly promising solutions and continue exploiting based on this information.

B. Algorithmic Description

1) General Structure: Algorithm 1 shows the general structure of MMDE. The first step is the parametric setup and initialization of population, in which each individual is represented as a pair of solution and scenario. Subsequently, at the start of each generation, a min heap [48] will be constructed based on the current population. Each individual will form one node in the min heap, and the key of each node is the objective value of the corresponding individual. The main feature of a min heap is that the node with the minimum key value will always be placed at the root, and it only takes $O(\log n)$ time to delete the current root node and $O(1)$ time to insert a new node [49], where n is the total number of nodes inside the heap. Based on the constructed min heap, a bottom-boosting scheme will be performed to update the scenarios associated with the nodes. After that, the current population will be updated by continuously extracting root nodes from the min heap. This will lead to a population wherein all the individuals are sorted according to their corresponding objective values in nondescending order. The associated solution of the first individual is recorded as the best solution found so far.

Algorithm 1 Procedure of MMDE

```
1: Parametric setup for MMDE: population size $N$, solution dimension $D_X$, scenario dimension $D_S$, maximum generation number $G_{\max}$ and control parameters for each algorithmic component
2: Set generation $g = 0$ and randomly initialize the population $P_g = \{(X_{1,g}, S_{1,g}), (X_{2,g}, S_{2,g}), \ldots, (X_{N,g}, S_{N,g})\}$, where $X_{i,g} = \{x^1_{i,g}, x^2_{i,g}, \ldots, x^{D_X}_{i,g}\}$ and $S_{i,g} = \{s^1_{i,g}, s^2_{i,g}, \ldots, s^{D_S}_{i,g}\}$, $i = 1, 2, \ldots, N$
3: for $i = 1$ to $N$ do
4:     Evaluate objective value for individual $(X_{i,g}, S_{i,g})$
5: end for
6: for $g = 1$ to $G_{\max}$ do
7:     Construct a min heap with each individual in $P_g$ as one node, the key of each node is the objective value of corresponding individual
8:     Perform Bottom-Boosting Scheme based on the constructed min heap
9:     for $i = 1$ to $N$ do
10:        $X_{i,g} = X_{\text{root}}, S_{i,g} = S_{\text{root}}$, where $(X_{\text{root}}, S_{\text{root}})$ is the individual stored in the root node of the min heap
11:        Pop the root node from the min heap
12:    end for
13:    $X_{\text{best}} = X_{1,g}$
14:    Perform Partial-Regeneration Strategy to update $P_g$
15:    $P_{g+1} = P_g$
16: end for
```

Algorithm 2 Bottom-Boosting Scheme

```
Require:
    A min heap constructed from current population $P_g$
    $K_S$: Parameter to control the total number of objective function evaluations
    $F$ and $Cr$: Control parameters for DE operators
Ensure:
    An updated min heap
1: $k = 0$
2: while $k < K_S$ do
3:     Read the root node of the min heap, suppose the individual stored inside the root node is $(X_{i,g}, S_{i,g})$
4:     generate mutant scenario as
       $SV_{i,g} = S_{r_1,g} + F \cdot (S_{r_2,g} - S_{r_3,g})$
       where indices $r_1$, $r_2$, $r_3$ are randomly selected integers from $\{1, 2, \ldots, N\}$ that are distinct from $i$ and mutually different, and $N$ is the population size
5:     Perform binomial recombination on $S_{i,g}$ and $SV_{i,g}$ to generate the trial scenario $SU_{i,g}$
6:     if $f(X_{i,g}, SU_{i,g}) > f(X_{i,g}, S_{i,g})$ then
7:         Pop the root node from the min heap
8:         $S_{i,g} = SU_{i,g}$
9:         Push the updated individual $(X_{i,g}, S_{i,g})$ into the min heap
10:    end if
11:    $k = k + 1$
12: end while
```

Finally, a partial-regeneration strategy will be conducted to further update current population, and the algorithm will then proceed to the next generation. The whole optimization process will be terminated when the maximum generation number is reached.

2) Bottom-Boosting Scheme: Algorithm 2 describes the internal procedure of the proposed bottom-boosting scheme. A new parameter K_S is introduced to control the total number of objective function evaluations incurred. Given the min heap constructed from current population, the individual stored in the root node will be extracted. Based on the scenario associated with this individual, a traditional "DE/rand/1" mutation strategy [34] and binomial recombination (same as in classical DE) will be performed to generate the trial scenario. Afterward, the original solution will be evaluated with the trial scenario to obtain a new objective value. If the new objective value is larger than the original objective value, the scenario associated with the original individual will be updated using the trial scenario. The corresponding node will also undergo a pop-update-push process to ensure the min heap is updated. The above steps will be repeated K_S times so totally K_S objective function evaluations are involved. Please note that the updating of scenarios will be performed on both current population and the min heap.

3) Partial-Regeneration Strategy: Algorithm 3 shows the basic steps of the proposed partial-regeneration strategy. A new parameter T is introduced to control the total number of regenerated individuals. Before performing partial-regeneration strategy, all the individuals have already been sorted according to their corresponding objective values in nondescending order. For the first T individuals, a new mutation strategy, namely "DE/current/1," will be performed on their associated solutions to generate T mutant solutions. Suppose $X_{i,g}$ is the associated solution for ith individual, its corresponding mutant solution will be generated via DE/current/1 as

$$V_{i,g} = X_{i,g} + F \cdot \left(X_{r_1,g} - X_{r_2,g}\right) \tag{13}$$

where indices r_1 and r_2 are randomly selected integers from $\{1, 2, \ldots, N\}$ that are distinct from i and mutually different, and N is the population size. After that, T trial solutions will be generated by performing binomial recombination (same as in classical DE) on each mutant solution and its corresponding original solution. Next, the associated solutions of the last T individuals will be updated using these trial solutions, and their associated scenarios will also be randomly reinitialized. The last step is to evaluate all these regenerated individuals and record their new objective values. Totally T objective function evaluations are carried out in the whole process.

C. Underlying Rationale

Different from traditional coevolutionary EAs, there is only one population in MMDE and each individual is represented as a pair of solution and scenario. Based on this representation, the optimization process in each generation is divided into two phases, in which the scenarios and solutions are updated, respectively. Briefly speaking, the scenario updating

Algorithm 3 Partial-Regeneration Strategy

```
Require:
    Current population $P_g$ with population size $N$, all the individuals have already been sorted according to their corresponding objective values in nondescending order
    $T$: Parameter to control the number of regenerated individuals
    $F$ and $Cr$: Control parameters for DE operators
Ensure:
    Updated population $P_g$
1: $t = 1$
2: while $t \leq T$ do
3:     $V_{i,g} = X_{i,g} + F \cdot (X_{r_1,g} - X_{r_2,g})$
       where index $i$ is a randomly selected integer from $\{1, 2, \ldots, T\}$, and indices $r_1$, $r_2$ are randomly selected integers from $\{1, 2, \ldots, N\}$ that are distinct from $i$ and mutually different
4:     Perform binomial recombination on $X_{i,g}$ and $V_{i,g}$ to generate the trial solution $U_{i,g}$
5:     $X_{N+1-t,g} = U_{i,g}$
6:     Randomly reinitialize $S_{N+1-t,g}$
7:     Evaluate the objective value of the updated individual $f(X_{N+1-t,g}, S_{N+1-t,g})$
8:     $t = t + 1$
9: end while
```

phase aims at searching for the promising solutions in terms of worst-case performance, and the solution updating phase tries to further exploit based on these solutions.

In scenario updating phase, the bottom-boosting scheme is employed to identify the promising solutions. Naturally, this can be done by searching for the worst-case scenarios for each solution. However, a large number of objective function evaluations will be wasted because our target is actually to find those solutions with good worst-case performance rather than obtain the worst-case scenarios for all the solutions. If the performance of solution X_1 under certain scenario has already been worse than the worst-case performance of X_2, it becomes pointless to further explore X_1 over scenario space. Following this consideration, the bottom-boosting scheme will focus on evolving scenarios for the best-performing individuals only. A heap data structure is utilized to minimize the computation time for finding the new best-performing individual after each updating. Compared to a normal sorting approach, the computational complexity for each best-finding operation can be reduced from $O(n \log n)$ to $O(\log n)$ by using a heap. The root node of the min heap always stores the individual with the lowest objective value. The bottom-boosting scheme will keep performing traditional DE/rand/1 mutation operation and binomial recombination to search for worse scenarios for current best-performing solution. Due to the strong explorative ability of DE/rand/1 operator, it will be very difficult for one solution to keep staying in the root node unless this solution is truly superior in terms of worst-case performance. Those solutions with poor performance under certain scenario will be eliminated from the optimization process in the early stage so that a large number of unnecessary evaluations can be skipped. In contrast, the solutions with robust performance under various scenarios will be more frequently challenged during their stay in the root node, thereby further strengthening the reliability of the final outstanding solutions.

In solution updating phase, the partial-regeneration strategy is adopted to evolve the whole population. Based on the min heap constructed in previous step, a heap sort is carried out to sort all the existing individuals based on their objective values. The worst T individuals will then be replaced by the offspring of the best T individuals. The proposed DE/current/1 mutation strategy generates the mutant solution by adding a scaled perturbation into the original solution. In this way, the offspring will keep exploiting around the promising solutions so that the overall optimization efficiency is enhanced. The associated scenarios of the regenerated solutions are randomly reinitialized because maintaining the diversity of the scenarios is beneficial to both explorative ability and reliability of the scenario updating phase. Same as the best T individuals, the remaining intermediate individuals are kept unchanged so that their robustness can be further judged in the next scenario updating phase.

With the specially designed algorithmic structure, MMDE is able to avoid the infinite optimization cycles that coevolutionary approaches will encounter in handling asymmetrical problems. The key point is that MMDE successfully avoid the behaviors of finding good solutions for any particular scenarios. MMDE only searches for bad scenarios for particular solutions. The evolution of solutions in MMDE is only based on their performance during scenario updating phase. The individuals survive to next generation are the ones providing robust performance under various scenarios instead of the ones with superior performance under particular scenario.

IV. Empirical Study

A. Comparison With Existing Algorithms

The optimization performance of the proposed MMDE is evaluated by comparing with seven famous minimax optimization EAs, namely, ASGA [20], alternating coevolutionary PSO (ACPSO) [26], alternating coevolutionary GA (ACGA) [28], parallel coevolutionary GA (PCGA) [9], best remaining coevolutionary GA (BRCGA) [33], rank-based coevolutionary GA (RBCGA) [32], and Stackelberg strategy coevolutionary GA (SSCGA) [20]. For all the existing algorithms, parametric settings suggested in [20] are utilized. For MMDE, the control parameters are set as follows: $F = 0.7$ and $\mathrm{Cr} = 0.5$ for both bottom-boosting scheme and partial-regeneration strategy, $N = 100$, $K_S = 190$, and $T = 10$.

Following the recent minimax optimization studies [20], [30], [31], the six most commonly used benchmark problems [32], [50] are tested in our experiments. Table I shows the objective functions, searching domains and global optima of the six benchmarks. $F1$, $F5$, and $F6$ are symmetrical problems, and $F2$–$F4$ are asymmetrical problems. Mean square error (MSE) [32] is used as the performance metric to

TABLE I
DESCRIPTION OF BENCHMARK PROBLEMS

F1	Objective	$f(x,s) = (x-5)^2 - (s-5)^2$
	Domain	$x \in [0,10], s \in [0,10]$
	Optimum	$(x^*, s^*) = (5,5)$
F2	Objective	$f(x,s) = \min\{3 - 0.2x + 0.3s, 3 + 0.2x - 0.1s\}$
	Domain	$x \in [0,10], s \in [0,10]$
	Optimum	$(x^*, s^*) = (0,0)$
F3	Objective	$f(x,s) = \frac{\sin(x-s)}{\sqrt{x^2+s^2}}$
	Domain	$x \in (0,10], s \in (0,10]$
	Optimum	$(x^*, s^*) = (10, 2.125683)$
F4	Objective	$f(x,s) = \frac{\cos\sqrt{x^2+s^2}}{\sqrt{x^2+s^2}+10}$
	Domain	$x \in [0,10], s \in [0,10]$
	Optimum	$(x^*, s^*) = (7.044146333751212, 10 \text{ or } 0)$
F5	Objective	$f(X,S) = 100(x_2 - x_1^2)^2 + (1-x_1)^2 - s_1(x_1 + x_2^2) - s_2(x_1^2 + x_2)$
	Domain	$X \in [-0.5, 0.5] \times [0,1], S \in [0,10]^2$
	Optimum	$(X^*, S^*) = (0.5, 0.25, 0, 0)$
F6	Objective	$f(X,S) = (x_1 - 2)^2 + (x_2 - 1)^2 + s_1(x_1^2 - x_2) + s_2(x_1 + x_2 - 2)$
	Domain	$X \in [-1,3]^2, S \in [0,10]^2$
	Optimum	$(X^*, S^*) = (1, 1, \text{any}, \text{any})$

quantitatively compare the optimization performances of algorithms. MSE is calculated according to the best solution obtained by the algorithm and the global optimal solution as

$$\mathrm{MSE}\big(X_{\text{best}}, X_{\text{opt}}\big) = \frac{1}{D_X}\sum_{j=1}^{D_X}\Big(x_{\text{best}}^j - x_{\text{opt}}^j\Big)^2 \tag{14}$$

where X_{best} is the best solution found by the algorithm, X_{opt} is the global optimal solution, D_X is dimensionality of solution space, and x_{best}^j and x_{opt}^j are the variables in dimension j. All of the simulations were done on an Intel Core i7 machine with 16-GB RAM and 3.40-GHz speed. Microsoft Visual Studio 2012 Express is used to develop the coding and run the experiments. Table II presents the mean, median and standard deviation of the MSEs for each algorithm on each benchmark problem over 100 independent runs. For all the algorithms, the maximum number of objective function evaluations or FEs is fixed as 100 000. Apart from the limited number of FEs, another termination condition is that the mean of the MSEs over 100 independent runs reaches 10^{-20} level or lower. For each problem, the total number of FEs performed by each algorithm in each run is also shown in Table II. In order to judge whether the results of the best-performing algorithm in terms of mean of MSEs differ from the results of the competitors in a statistically significant way, a nonparametric statistical test called Wilcoxons rank-sum test [51] is conducted at the 5% significance level. The P-values obtained through the rank sum test between the best algorithm and each of the remaining algorithms over all the benchmark functions are presented in Table II. NA stands for *not applicable* and occurs for the best performing algorithm itself in each case. If the P-values are less than 0.05, it indicates that the better performances achieved by the best algorithm in each case are statistically significant and have not occurred by chance [52]. In Table II, the best entries in terms of mean of MSEs, median of MSEs and number of FEs are marked in boldface.

Based on the experimental results in Table II, the proposed MMDE is able to *significantly* outperform all the other algorithms in all the benchmark problems in terms of both mean and median of MSEs over 100 independent runs. For the first three problems, MMDE can reach the global optimal solution without any errors using only 48 500, 68 500, and 2700 FEs in all the 100 independent runs. For problem $F4$ and $F5$, the number of FEs required for MMDE to reduce the mean of MSEs into 10^{-20} level or lower is only 59 900 and 27 300, respectively. For problem $F6$, MMDE also achieves considerably lower MSEs than other approaches in all the 100 independent runs. For all the coevolutionary EAs, poor performances were observed in the three asymmetrical problems ($F2$–$F4$). Figure S1 (in the supplementary material) shows the convergence graphs for the testing algorithms. One notable phenomenon is that in asymmetrical problems ($F2$–$F4$), all the coevolutionary algorithms show severe oscillations, which indicates the occurrence of infinite optimization cycles. This is in accordance with our theoretical analysis in Section III-A. Besides MMDE, ASGA is the only algorithm that solves both symmetrical and asymmetrical problems correctly. However, from the perspective of optimization efficiency, MMDE shows substantially better performance than ASGA. To summarize, MMDE is capable of solving both symmetrical and asymmetrical minimax optimization problems not only reliably but also efficiently.

In order to further validate the robustness of the proposed framework, the conventional DE optimizer in bottom-boosting scheme is replaced by simulated binary crossover (SBX) [53], which is also employed by ASGA, ACGA, PCGA, BRCGA, RBCGA, and SSCGA, together with polynomial mutation [53] ($\eta_c = \eta_m = 20$). Table S1 (in the supplementary material) compares the performance of the SBX version with other algorithms using the same experimental setups as in Table II. Based on the experimental results, the SBX version also shows superior performance compared with all the other counterparts in all the benchmark problems. Generally, the performance of SBX version is only slightly worse than the original version. This observation clearly validates the superiority of the specially designed frameworks in solving minimax optimization problems.

B. *Effectiveness of Bottom-Boosting Scheme*

In MMDE, the bottom-boosting scheme plays a very critical role in increasing the efficiency of the whole algorithm. A large number of objective function evaluations are skipped during the proposed mechanism. In order to further investigate the effectiveness of bottom-boosting scheme, the performance of the original MMDE is compared with that of a simplified version. In the simplified MMDE, the only modification is that during the scenario updating phase, instead of only optimizing the current best-performing individual, the mutation, recombination and updating operations will be conducted on each individual one by one. For a more comprehensive investigation, the number of FEs used in each scenario updating phase of the simplified version (we also call it K_S) will be set from 200 to 2000, and all these different variants will

TABLE II
SUMMARIZED RESULTS OVER 100 INDEPENDENT RUNS

Problems		MMDE	ASGA	ACPSO	ACGA	PCGA	BRCGA	RBCGA	SSCGA
F1	Mean	**0.0000E+00**	6.7677E-17	3.8636E-02	3.8149E-05	1.4495E-04	3.2412E-05	3.1323E-04	4.8554E-06
	Median	**0.0000E+00**	3.2221E-20	9.3041E-03	6.9977E-15	1.7041E-16	1.9274E-17	8.5099E-21	5.4367E-17
	Std	0.0000E+00	4.3812E-16	7.1026E-02	2.7335E-04	9.6055E-04	1.5046E-04	1.7825E-03	3.2735E-05
	P-value	NA	5.6400E-39	5.6400E-39	5.6400E-39	8.0734E-32	8.0708E-32	8.3171E-29	2.7251E-30
	FEs	**48500**	100000	100000	100000	100000	100000	100000	100000
F2	Mean	**0.0000E+00**	8.0707E-05	1.7507E+01	3.0871E+01	3.0010E+01	2.8730E+01	9.4512E+00	4.8793E+01
	Median	**0.0000E+00**	1.1160E-12	1.5118E-02	1.0128E-04	1.1536E-04	1.5118E-02	9.2164E-09	1.7764E+01
	Std	0.0000E+00	7.5651E-04	3.5838E+01	4.6208E+01	4.5080E+01	3.5838E+01	2.8471E+01	4.9080E+01
	P-value	NA	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39
	FEs	**68500**	100000	100000	100000	100000	100000	100000	100000
F3	Mean	**0.0000E+00**	1.3934E-04	2.9861E+00	2.5216E+01	2.1888E+01	2.8629E+01	1.4610E+01	2.7145E+01
	Median	**0.0000E+00**	2.3755E-09	6.4173E-01	1.4978E+01	1.2116E+01	1.0285E+01	6.2826E-02	9.0744E+00
	Std	0.0000E+00	8.0636E-04	3.5121E+00	2.7667E+01	2.7996E+01	3.6638E+01	2.8781E+01	3.6543E+01
	P-value	NA	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39	5.6400E-39
	FEs	**2700**	100000	100000	100000	100000	100000	100000	100000
F4	Mean	**1.2098E-21**	8.7356E-03	5.8835E+00	1.1798E+01	1.1210E+01	7.3399E+00	1.7349E+00	1.1879E+01
	Median	**7.0997E-30**	8.4006E-04	1.4031E+00	6.7570E+00	3.1082E+00	2.7788E+00	2.8796E-05	5.0549E+00
	Std	1.2020E-20	6.7046E-02	9.1577E+00	1.3040E+01	1.5225E+01	1.0651E+01	4.9994E+00	1.5445E+01
	P-value	NA	1.9751E-34	1.9751E-34	1.9751E-34	1.9751E-34	1.9751E-34	1.9751E-34	1.9751E-34
	FEs	**59900**	100000	100000	100000	100000	100000	100000	100000
F5	Mean	**9.4638E-20**	2.9369E-03	1.7759E-02	7.1615E-03	5.6205E-03	8.1955E-03	7.9175E-03	7.3330E-03
	Median	**2.3279E-20**	4.2936E-04	1.1057E-02	2.2320E-03	1.0877E-03	1.4470E-03	4.9840E-03	2.4920E-03
	Std	2.9861E-19	6.0315E-03	2.0919E-02	1.1399E-02	1.0152E-02	1.4941E-02	1.2295E-02	1.1715E-02
	P-value	NA	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34
	FEs	**22700**	100000	100000	100000	100000	100000	100000	100000
F6	Mean	**2.2867E-17**	4.0423E-04	6.4795E-02	6.9830E-03	4.6471E-03	1.0845E-02	1.4946E-03	2.1075E-02
	Median	**1.7417E-26**	4.2174E-05	4.7074E-02	1.8802E-03	2.2910E-04	1.3874E-04	5.5285E-05	9.4070E-04
	Std	2.2159E-16	8.3540E-04	6.2820E-02	1.1787E-02	1.1689E-02	4.8875E-02	7.7955E-03	6.2610E-02
	P-value	NA	4.2663E-34	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34	2.5621E-34
	FEs	100000	100000	100000	100000	100000	100000	100000	100000

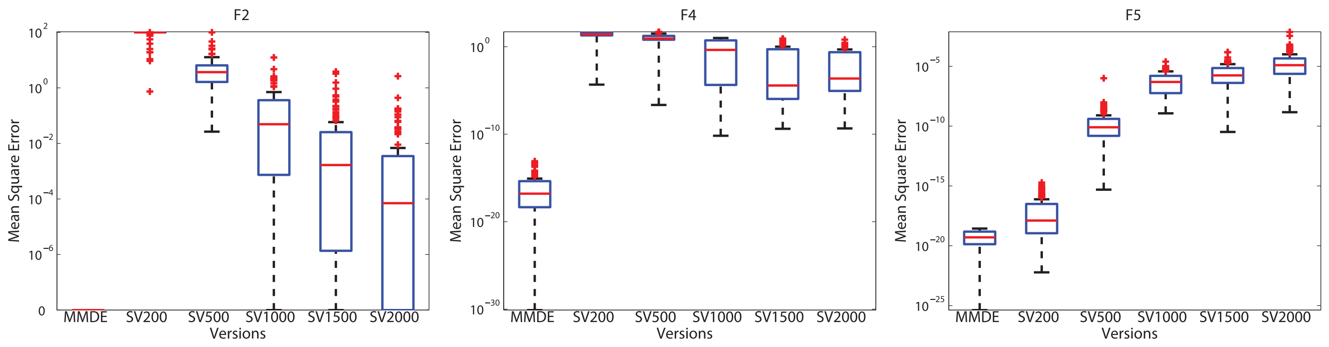


Fig. 2. Boxplots of the MSE values over 100 independent runs. "SV200" represents the simplified version with $K_S = 200$. Same rule applies to all the remaining labels. On each box, the red line is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not consider outliers, and outliers are plotted individually using red markers.

be tested for comparison. For all the other control parameters and experimental setups, same settings as described in Section IV-A are employed. Fig. 2 shows the boxplots of the MSE values obtained by each algorithm over 100 independent runs on problem $F2$, $F4$, and $F5$.

From the experimental results, the original MMDE achieves consistently better performance than all the variants of the simplified version. For problem $F2$, a K_S value less then 500 will lead to a very poor performance of the simplified version. The reason is that due to the evenly distribution of computational budget, each individual can only perform a limited number of FEs, which may not be sufficient to correctly evaluate their true quality. According to the results, a great number of FEs are required for the simplified version to provide an acceptable performance in $F2$. In comparison, the proposed bottom-boosting scheme successfully reaches the global optimal solution with no errors in all the 100 independent runs with only 190 FEs in each scenario updating phase. Similar pattern can also be observed from the experimental results on problem $F4$. This implies that the proposed mechanism successfully enhances both efficiency and reliability of the whole algorithm. For problem $F5$, a small K_S is preferred

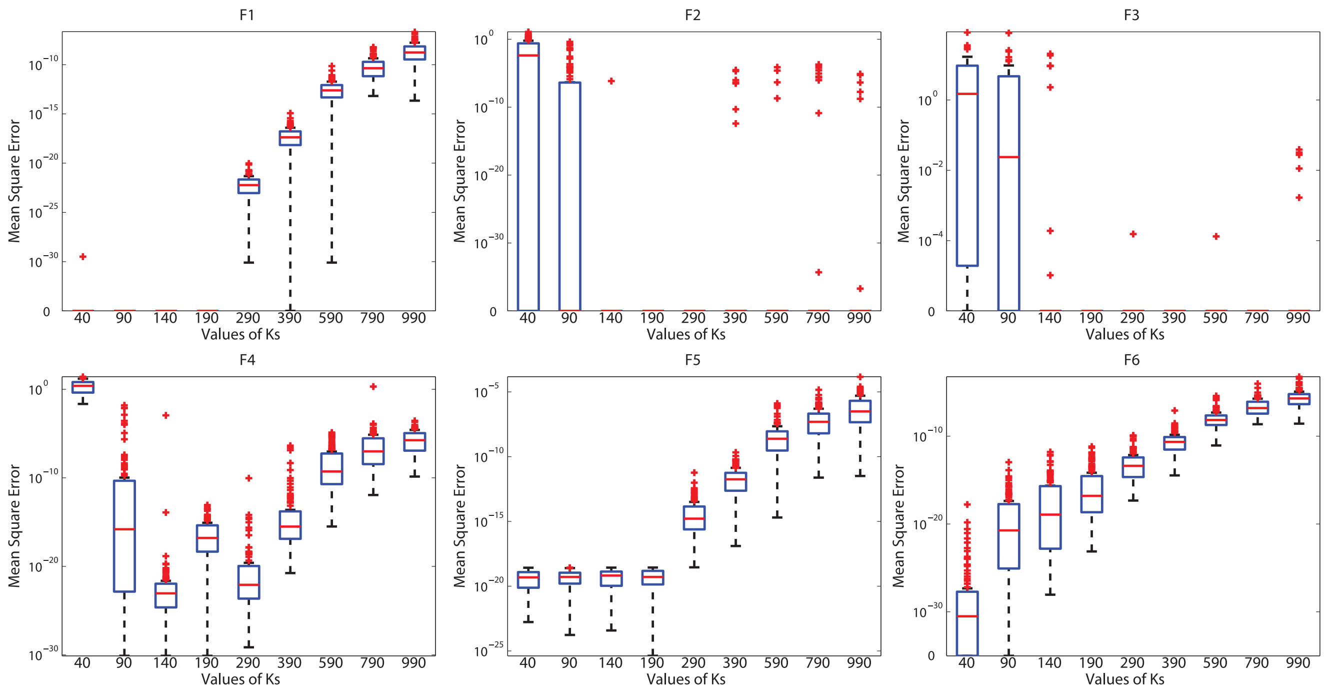


Fig. 3. Boxplots of the MSE values over 100 independent runs for different K_S settings. On each box, the red line is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not consider outliers, and outliers are plotted individually using red markers.

by the simplified version. This is because $F5$ is a symmetrical problem that satisfies (3) and (4). All the individuals will gradually evolve their associated scenarios toward the same one [e.g., S^* in (4)], thereby reducing the difficulty for scenario optimization. Since a smaller number of FEs in each scenario updating phase allows more exploration over solution space, the solution quality is further enhanced. Under this circumstance, bottom-boosting scheme still provides better overall performance than the simplified version with similar K_S values. This indicates the consistent effectiveness of the proposed scheme in solving problems with different properties.

C. *Effectiveness of Partial-Regeneration Strategy*

The main advantage of partial-regeneration strategy is the capability of avoiding infinite optimization cycles in solving asymmetrical problems. To validate the effectiveness of partial-regeneration strategy, a simplified version of MMDE is generated by replacing the partial-regeneration strategy with a normal DE optimizer, in which each solution will undergo the mutation, crossover and selection procedure as described in Section II-C. All the remaining parametric setups are identical with those in Section IV-A. Table S4 (in the supplementary material) compares the performances of the original and simplified variant on $F1$–$F6$ over 100 independent runs.

According to the experimental results, for all the three asymmetrical problems ($F2$–$F4$), the simplified version returns very poor solutions while the original version exhibits superior performance. Moreover, the original version also *significantly* outperforms the simplified variant in one symmetrical problem. In the other two symmetrical problems, the two algorithms perform equally good. The effectiveness of partial-regeneration strategy is clearly demonstrated via this observation.

D. *Sensitivity Study for K_S and T*

This section aims at studying the influences of the two new control parameters K_S and T on the overall optimization performance.

Fig. 3 shows the boxplots of MSEs over 100 independent runs for different K_S settings. All the other control parameters and experimental setups are identical with those in Section IV-A. For the three symmetrical problems $F1$, $F5$, and $F6$, relatively lower K_S values are desirable. As we discussed in Section IV-B, with the reduced difficulty in scenario optimization, the exploration over solution space becomes more important in handling symmetrical problems. A better tradeoff between the scenario optimization, which is related to solution reliability, and solution optimization, which is relevant to solution quality, will be achieved by selecting a relatively smaller K_S. For the three asymmetrical problems $F2$–$F4$, the difficulty of the scenario optimization has increased because each individual has disparate worst-case scenarios. Based on the experimental results, a K_S value less than 100 is insufficient for the scenario updating phase to correctly identify the true promising solutions, and an over large K_S value may increase the solution reliability by sacrificing the solution quality. A moderate K_S value is therefore more suitable for asymmetrical problems.

Fig. 4 presents the boxplots of MSEs over 100 independent runs for different T settings. All the other control parameters and experimental setups are identical with those in Section IV-A. For both symmetrical and asymmetrical

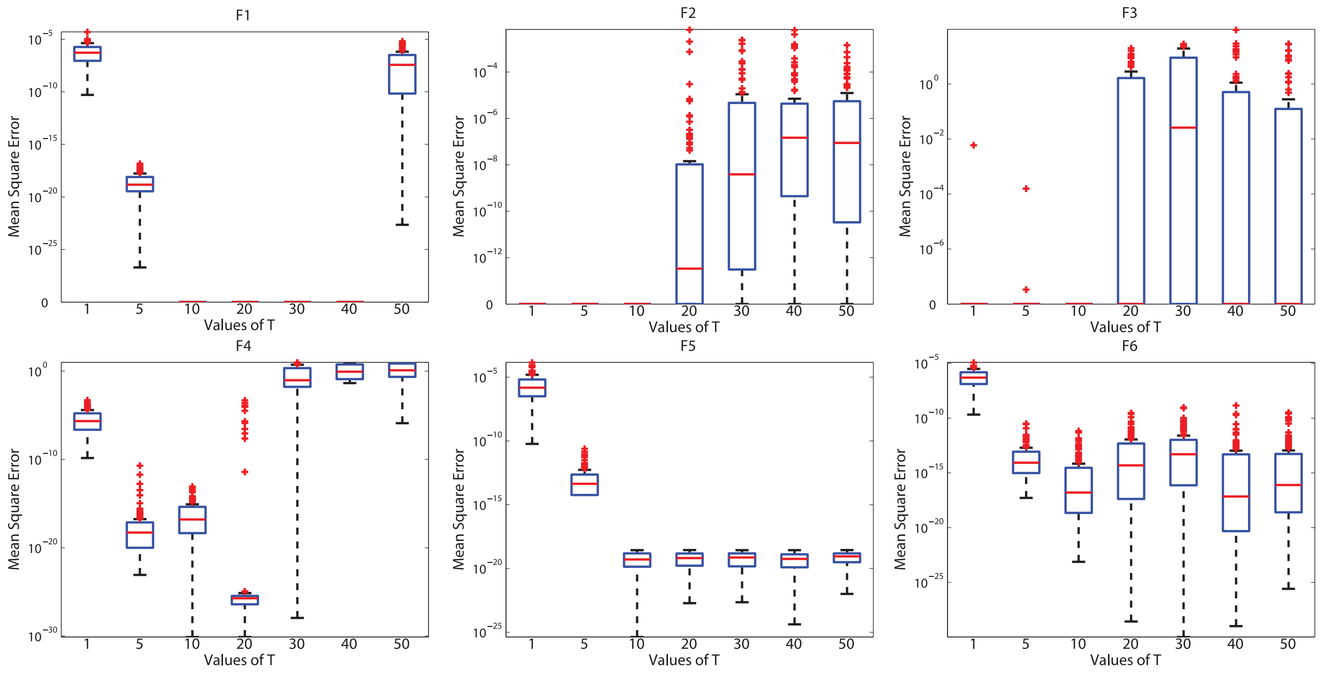


Fig. 4. Boxplots of the MSE values over 100 independent runs for different T settings. On each box, the red line is the median, the edges of the box are the 25th and 75th percentiles, the whiskers extend to the most extreme data points not consider outliers, and outliers are plotted individually using red markers.

problems, a T value of 10 provides the most robust performance. Either an over small or over large T may lead to unsatisfactory optimization performance regardless of the problem properties. This is because the value of T decides the number of regenerated individuals during each solution updating phase. If T is too small, the exploration over solution space will be conducted in a very slow manner. If T is too large, the regeneration of solutions will become excessively frequent, and the probability to discard promising solutions is increased. In order to avoid these two issues, a T value that is around 10% of the population size is recommended.

To further investigate the dependency between K_S and T, Table S3 (in the supplementary material) presents the performances of MMDE under different combinations of K_S and T values. All the remaining control parameters and experimental setups are identical with those in Section IV-A. To summarize, a larger K_S value works better with a larger T value, and vice versa. This is because a larger K_S will increase the reliability of the good-ranking individuals after bottom-boosting scheme, and a larger T allows more exploitation around these promising solutions. Conversely, a relatively small K_S requires a small T value because the conservative regeneration process can compensate for the lower reliability.

E. Scalability of MMDE

In order to investigate whether MMDE is scalable to the dimensionality of problems, benchmarks $F1$ and $F2$ are extended into higher dimensions as follows:

$$f_1(X, S) = \sum_{d=1}^{D_X}\left(x^d - 5\right)^2 - \sum_{d=1}^{D_S}\left(s^d - 5\right)^2$$

where $X \in [0, 10]^{D_X}, S \in [0, 10]^{D_S}$.

$$f_2(X, S) = \sum_{d=1}^{D}\min\left\{3 - 0.2x^d + 0.3s^d, 3 + 0.2x^d - 0.1s^d\right\}$$

where $D_X = D_S = D, X \in [0, 10]^{D_X}, S \in [0, 10]^{D_S}$.

Table S2 (in the supplementary material) shows the performances of MMDE under different settings of D_X and D_S. The maximum number of FEs is linearly increased with the dimensionality of problems. Considering the difficulty of asymmetrical problems, the value of K_S will be set as $190 \times \lfloor\sqrt{D_S}\rfloor$ while solving $F2$. All the remaining parametric and experimental setups are identical with those in Section IV-A. Based on the experimental results, MMDE exhibits excellent performances over all the different dimensionality setups for both symmetrical and asymmetrical problems. It is notable that even for the most difficult high-dimensional asymmetrical problem, MMDE is able to avoid the infinite optimization cycle and consistently locate the global optima with a very small MSE. This pattern indicates that MMDE is well scalable to the dimensionality of problems.

V. Applications

A. Robust Optimal Design of Iterative Learning Control

In this section, the proposed MMDE algorithm is applied to address an open problem in iterative learning control (ILC): how to systematically design an appropriate learning control gain matrix for nonlinear multi-input-multioutput (MIMO) systems. An appropriately chosen learning gain matrix can speed up learning convergence in the presence of the system nonlinearities and uncertainties. Targeting at time-optimal (fastest convergence) and robustness properties concurrently, we formulate the ILC design task into a minimax optimization problem.

Consider an ILC for MIMO dynamic systems

$$\begin{aligned}\dot{x}(t) &= f(x(t), u(t), t)\ x(0) = x_0\\ y(t) &= g(x(t), u(t), t)\end{aligned} \tag{15}$$

where $t \in [0, T]$, $x(t) \in \mathbb{X} \subset \mathbb{R}^n$, $y(t) \in \mathbb{Y} \subset \mathbb{R}^m$, $u(t) \in \mathbb{U} \subset \mathbb{R}^m$, $\mathbb{X}$, $\mathbb{Y}$, and $\mathbb{U}$ are compact convex subsets. Nonlinear functions $f(\cdot)$ and $g(\cdot)$ satisfy the global Lipschitz continuity condition with respect to state x and input u.

The target of ILC is to find a sequence of appropriate control inputs such that the system output $y_i(t)$ can track the reference trajectory $y_r(t)$. A typical ILC is given as

$$u_{i+1}(t) = u_i(t) + Q\Delta y_i(t) \tag{16}$$

where $Q \in \mathbb{R}^{m\times m}$ is the learning gain matrix. According to [54]–[56], The ILC convergence condition is

$$\|\Delta y_{i+1}\| \le \|I_{m\times m} - G(\xi)Q\|\|\Delta y_i\| \tag{17}$$

where $G \triangleq (\partial g/\partial u)$ is the direct feed-through matrix, ξ is a point in the compact set $\Omega = \mathbb{X} \times \mathbb{U} \times [0, T]$, and $\|\cdot\|$ is an appropriate vector norm and the induced matrix norm. Clearly, to warrant a convergent learning sequence of $\|\Delta y_i\|$, a sufficient condition is

$$\|I_{m\times m} - G(\xi)Q\| \le \gamma, \quad \forall \xi \in \Omega \tag{18}$$

where $\gamma \in (0, 1)$ is a constant. Moreover, the smaller is $\|I_{m\times m} - G(\xi)Q\|$, the faster is the learning convergence speed. Hence, the ILC design task becomes to find an appropriate gain matrix $Q \in \mathbb{R}^{m\times m}$, such that $\|I_{m\times m} - G(\xi)Q\|$ is minimal under the worst-case $G(\xi)$, $\forall \xi \in \Omega$.

Now the robust optimal design can be formulated as the following minimax optimization problem:

$$\min_{Q\in\mathbb{X}} \max_{\xi\in\Omega} \|I_{m\times m} - G(\xi)Q\| \tag{19}$$

where $\mathbb{X} = \mathbb{R}^{m\times m}$.

Due to the existence of nonlinearities, uncertainties and nonsymmetry in the system direct feed-through matrix, it is in general a very difficult task to directly solve the original minimax optimization problem with a closed-form solution. Nevertheless, the proposed MMDE does not make any prior assumption about the mathematical properties of the problem. It is practicable to apply MMDE to solve this mathematically intractable optimization problem.

To verify the effectiveness of MMDE, the proposed algorithm is applied to the robust optimal design of a two-link robotic manipulator, which is a nonlinear dynamic system described by

$$M(x)\ddot{x} + f(x, \dot{x}) = u \tag{20}$$

where

$$M(x) = \begin{bmatrix} m_{11} & m_{12}\\ m_{21} & m_{22}\end{bmatrix} \tag{21}$$

is the inertia matrix with

$$\begin{aligned} m_{11} &= m_1 l_{c1}^2 + I_1 + m_2\Big[l_1^2 + l_{c2}^2 + 2l_1 l_{c2}\cos(x_2) + I_2\Big]\\ m_{12} &= m_{21} = m_2 l_1 l_{c2}\cos(x_2) + m_2 l_{c2}^2 + I_2\\ m_{22} &= m_2 l_{c2}^2 + I_2\end{aligned} \tag{22}$$

and

$$f(x, \dot{x}) = \begin{bmatrix} f_{11}\\ f_{12}\end{bmatrix} \tag{23}$$

represents all Centrifugal, Corioli's and gravity terms with

$$\begin{aligned} f_{11} &= -2h\dot{x}_1\dot{x}_2 - h\dot{x}_2^2 + m_1 l_{c1} g\cos(x_1)\\ &\quad + m_2 g[l_{c2}\cos(x_1 + x_2) + l_1\cos(x_1)]\\ f_{12} &= h\dot{x}_1^2 + m_2 l_{c2} g\cos(x_1 + x_2)\end{aligned} \tag{24}$$

$x = [x_1, x_2]^T$ are the two joint angles, $u = [u_1, u_2]^T$ are the joint inputs.

The angular velocities are selected to be the system output, $y = \dot{x}$. The desired trajectories are

$$y_{r,1} = y_{r,2} = 20\Big(60\tau^3 - 30\tau^4 - 30\tau^2\Big) \tag{25}$$

where $y_r = [y_{r,1}, y_{r,2}]^T$ and $\tau = t/T_c$. We choose $T_c = 1$ second to be the period of learning cycle.

The system parameters are given as: link masses $m1 = 4$ kg; $m2 = 3 + \Delta m_2$ kg; link lengths $l_1 = 0.5$ m; center of gravity co-ordinates $l_{c1} = 0.3$ m; $l_{c2} = 0.25$ m; and moments of inertia $I_1 = 0.4$ Kg-m^2; $I_2 = 0.25 + \Delta I_x$ Kg-m^2. In addition, there exist parametric uncertainties in m_2 and I_2: $\pm 50\%$ deviations from their nominal values.

Since there is no direct feed-through item in the input-output mapping, a D-type ILC scheme is employed

$$u_{i+1}(t) = u_i(t) + Q\big[\dot{y}_r(t) - \dot{y}_i(t)\big] \tag{26}$$

where $Q \in \mathbb{R}^{2\times 2}$. The resulting direct feed-through matrix is $G = M^{-1}$, and G is positive definite. With the above definitions of Q and G, the robust optimal design can be formulated into a minimax optimization problem using (19), and MMDE is applied to solve it.

Conventionally, the gain matrix Q is a diagonal matrix. Therefore, in this example, Q can be represented as

$$Q = \begin{bmatrix} x_{11} & 0\\ 0 & x_{22}\end{bmatrix}. \tag{27}$$

Thus, x_{11} and x_{22} are the two variables to be optimized over solution space. In the scenario space, there are three variables, namely, $m_2 \in [1.5, 4.5]$, $I_2 \in [0.125, 0.375]$, and $x_2 \in [-\pi, \pi]$. The objective values are calculated using $\|I_{m\times m} - G(\xi)Q\|$. The control parameters are set as follows: $F = 0.7$ and $\mathrm{Cr} = 0.5$ for both bottom-boosting scheme and partial-regeneration strategy, $N = 100$, $K_S = 990$, and $T = 10$, the maximum number of FEs is 2×10^7.

Based on the experimental results, the proposed MMDE is able to find numerous solutions that satisfy condition (18). Based on the solutions provided by MMDE, we have successfully located the feasible region in solution space, in which all the solutions are valid. Fig. 5 plots the worst-case objective values of all the solutions within or around the feasible region in solution space. All the solutions with worst-case objective values less than 1 are valid solutions for this design problem. Within the feasible region, the solutions with lower worst-case objective values are preferred since they provide generally faster learning convergence speed.

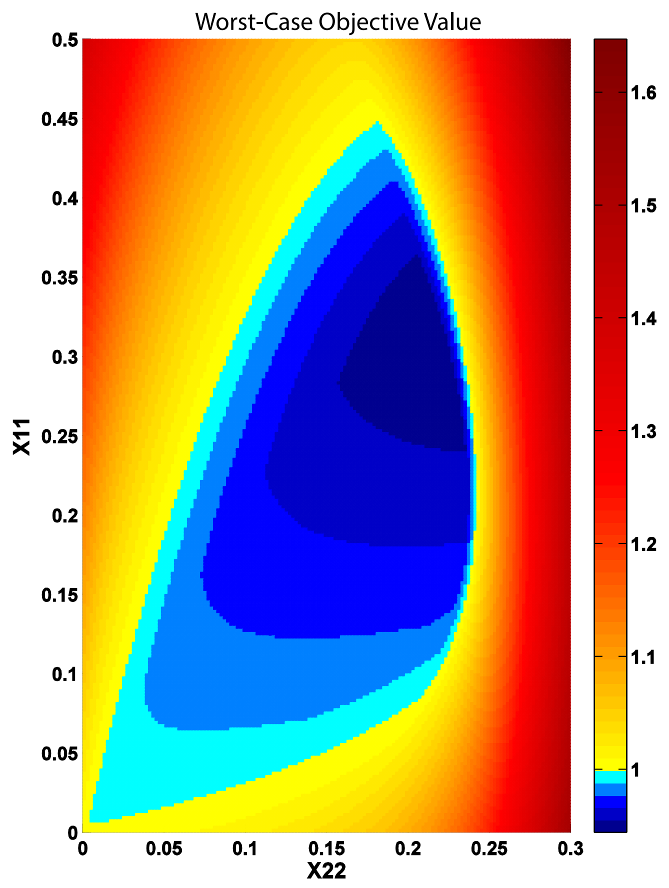


Fig. 5. Worst-case objective values of the solutions within or around the feasible region in solution space.

B. Robust Stabilization of Uncertain Time-Delay Systems

Time-delays are inevitable in many natural and industrial applications, and they often give rise to oscillations or instability of the entire systems. The robust stabilization of time-delay systems with uncertainties have been a very challenging problem in the realm of control theory [57], [58]. In [58], a linear matrix inequality (LMI) based approach was proposed to solve this robust stabilization problems, and it was claimed that the LMI approach obtained less conservative results compared to other traditional methods. In order to further examine the effectiveness of MMDE, we will formulate this robust stabilization task into a minimax optimization problem, and the results generated by MMDE will be compared with those of LMI method.

Consider an uncertain time-delay system containing nonlinear saturating actuators

$$\begin{aligned}\dot{x}(t) &= A(\delta)x(t) + A_1(\delta)x(t-d) + B(\delta)u'(t)\\ u'(t) &= \mathrm{sat}(u(t))\\ \mathrm{sat}(u(t)) &= [\mathrm{sat}(u_1(t))\mathrm{sat}(u_2(t))\cdots\mathrm{sat}(u_m(t))]\\ x(t) &= \phi(t), t\in[-\tau,0]\end{aligned} \tag{28}$$

where $x(t)\in\mathbb{R}^n$ is the state vector, $u(t)\in\mathbb{R}^m$ is the control input vector of the actuator, $u'(t)\in\mathbb{R}^m$ is the control input vector to the plant, $A(\delta) = A + \Delta A(\delta)$, $A_1(\delta) = A_1 + \Delta A_1(\delta)$, $B(\delta) = B + \Delta B(\delta)$, $A\in\mathbb{R}^{n\times n}$, $A_1\in\mathbb{R}^{n\times n}$, and $B\in\mathbb{R}^{n\times m}$ are constant matrices, $\Delta A(\cdot)$, $\Delta A_1(\cdot)$, and $\Delta B(\cdot)$ represent parameter uncertainties, $\mathrm{sat}(\cdot)$ is the nonlinear saturation function, and $\phi(t)$ is a smooth vector-valued continues initial function.

The admissible uncertainties are assumed to be of the form

$$\begin{aligned}\Delta A(\delta) &= H_1F_1(\delta)E_1\\ \Delta A_1(\delta) &= H_2F_2(\delta)E_2\\ \Delta B(\delta) &= H_3F_3(\delta)E_3\end{aligned} \tag{29}$$

where $F_i(\delta)\in\mathbb{R}^{s_i\times q_i}$, $i = 1, 2, 3$ are unknown real matrices with Lebesgue measurable elements that satisfy

$$F_i^T(\delta)F_i(\delta)\le I, i = 1, 2, 3 \tag{30}$$

and H_i, E_i, $i = 1, 2, 3$ are known real constant matrices.

The purpose of our task is to extend the upper bound τ of time-delay d such that the uncertain linear time-delay system is robustly stabilizable for any $0 < d \le \tau$. In order to convert the robust stabilization task into a minimax optimization problem, the original system described in (28) is written as

$$\dot{x}(t) = \bar{A}x(t) + \bar{A}_1x(t-d) + \bar{B}Lx(t) \tag{31}$$

where $\bar{A} = A + \Delta A(\delta)$, $\bar{A}_1 = A_1 + \Delta A_1(\delta)$, $\bar{B} = B + \Delta B(\delta)$, $Lx(t) = u'(t)$, and L is a predefined real-valued matrix. After performing Laplace transform on (31), we will have

$$sIX(s) = \bar{A}X(s) + \bar{A}_1e^{-ds}X(s) + \bar{B}LX(s). \tag{32}$$

To ensure the system stability, all poles of the system must be negative, and this is equivalent to ensuring all the roots s of equation

$$\det\left(sI - \bar{A} - \bar{A}_1e^{-ds} - \bar{B}L\right) = 0 \tag{33}$$

are negative. $\det(\cdot)$ means the determinant of a matrix. Now the target becomes to find a matrix L such that all the roots s of (33) are negative under any uncertainties resulted from $\bar{A}$, $\bar{A}_1$, and $\bar{B}$. This can be achieved by solving a minimax optimization problem that aims at minimizing the value of the largest root of (33) under the worst-case uncertainty

$$\min_{L}\max_{\bar{A},\bar{A}_1,\bar{B}}\left\{s_{\max}|\det\left(sI - \bar{A} - \bar{A}_1e^{-ds} - \bar{B}L\right) = 0\right\} \tag{34}$$

where $s_{\max}$ is the largest root of the equation. If the obtained largest root is less than 0, then it is guaranteed that all the remaining poles are also negative so that the system is successfully stabilized.

The same numerical example in [58] is tested here

$$\begin{aligned}&A = \begin{bmatrix}-2 & 0\\ 1 & -3\end{bmatrix}, A_1 = \begin{bmatrix}-1 & 0\\ -0.8 & -1\end{bmatrix}, B = \begin{bmatrix}1 & 2\\ -1 & 4\end{bmatrix}\\ &H_i = \begin{bmatrix}0.2 & 0\\ 0 & 0.2\end{bmatrix}, E_i = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}, i = 1, 2, 3\\ &F_i = \begin{bmatrix}\sin(\delta) & 0\\ 0 & \cos(\delta)\end{bmatrix}, i = 1, 2, 3.\end{aligned} \tag{35}$$

Based on the above assumptions, L is a 2 by 2 matrix, and the four elements of L will be optimized by MMDE as variables in solution space. The searching domain for all the solution variables are $[-100, 100]$. δ will be treated as the variable in scenario space, and the search domain is $[0, 2\pi]$. Other control parameters of MMDE are set as follows: $F = 0.7$ and $\mathrm{Cr} = 0.5$ for both bottom-boosting scheme and partial-regeneration strategy, $N = 100$, $K_S = 990$, and $T = 10$, the maximum number of FEs is 1×10^6. Newton–Raphson method is employed to obtain the largest roots of (33) with an extremely big initial guess. For each L found by MMDE, we will perform an exhaustive search over scenario space to ensure that the system is stabilized by L under all the possible uncertainties.

During the experiments, we will gradually increase the time-delay d if MMDE is able to find a matrix L that can robustly stabilized the system. Finally, we have increased the d value to 1.33, and validated that MMDE is able to solve this robust stabilization task for any $0 < d \leq 1.33$. In comparison, the LMI approach can only robustly stabilize this system for time-delay $d \leq 0.2961$ [58]. Using MMDE, the upper bound for feasible time-delay has increased by 349.17%.

VI. Conclusion

An MMDE algorithm is proposed in this paper to overcome the limitations of existing approaches in solving minimax optimization problems. In scenario updating phase, the bottom-boosting scheme successfully skips a large number of unnecessary objective function computations while maintaining the reliability of solution evaluations. In solution updating phase, the partial-regeneration strategy and DE/current/1 mutation operator allow an efficient solution exploration based on current population. Moreover, the overall algorithmic structure enables the proposed method to properly handle the asymmetrical problems. Experimental results show that MMDE outperforms all the other tested algorithms in both symmetrical and asymmetrical benchmarks. The effectiveness of MMDE is further validated by solving two open problems in robust design.

In future work, an online adaptation mechanism for parameter K_S is expected to be developed. The robustness and efficiency of the algorithm can be further enhanced with this new mechanism. Additionally, MMDE will be applied to solve more robust design problems in biostatistics.

Acknowledgment

The authors would like to thank Dr. A. M. Cramer from the University of Kentucky for generously providing all the raw data and source codes of his related works.

References

[1] H. Wang, C. Weng, and J. Yuan, “Multi-feature spectral clustering with minimax optimization,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Columbus, OH, USA, Jun. 2014, pp. 4106–4113.

[2] B. Chen, J. Wang, L. Wang, Y. He, and Z. Wang, “Robust optimization for transmission expansion planning: Minimax cost vs. minimax regret,” *IEEE Trans. Power Syst.*, vol. 29, no. 6, pp. 3069–3077, Nov. 2014.

[3] C. Y.-F. Ho *et al.*, “Two-channel linear phase FIR QMF bank minimax design via global nonconvex optimization programming,” *IEEE Trans. Signal Process.*, vol. 58, no. 8, pp. 4436–4441, Aug. 2010.

[4] V. Grushkovskaya and A. Zuyev, “Optimal stabilization problem with minimax cost in a critical case,” *IEEE Trans. Autom. Control*, vol. 59, no. 9, pp. 2512–2517, Sep. 2014.

[5] D. Agnew, “Improved minimax optimization for circuit design,” *IEEE Trans. Circuits Syst.*, vol. 28, no. 8, pp. 791–803, Aug. 1981.

[6] J. W. Bandler, W. Kellermann, and K. Madsen, “A superlinearly convergent minimax algorithm for microwave circuit design,” *IEEE Trans. Microw. Theory Techn.*, vol. 33, no. 12, pp. 1519–1530, Dec. 1985.

[7] A. V. Sebald and J. Schlenzig, “Minimax design of neural net controllers for highly uncertain plants,” *IEEE Trans. Neural Netw.*, vol. 5, no. 1, pp. 73–82, Jan. 1994.

[8] Y.-S. Ong, P. B. Nair, and K. Y. Lum, “Max-min surrogate-assisted evolutionary algorithm for robust design,” *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 392–404, Aug. 2006.

[9] J. W. Herrmann, “A genetic algorithm for minimax optimization problems,” in *Proc. Congr. Evol. Comput. (CEC)*, vol. 2. Washington, DC, USA, 1999, p. 1103.

[10] P. Kouvelis and G. Yu, “Robust discrete optimization and its applications,” in *Nonconvex Optimization and Its Applications*, vol. 14. New York, NY, USA: Springer, 1997.

[11] R. Montemanni, L. M. Gambardella, and A. V. Donati, “A branch and bound algorithm for the robust shortest path problem with interval data,” *Oper. Res. Lett.*, vol. 32, no. 3, pp. 225–232, 2004.

[12] R. Montemanni and L. M. Gambardella, “A branch and bound algorithm for the robust spanning tree problem with interval data,” *Eur. J. Oper. Res.*, vol. 161, no. 3, pp. 771–779, 2005.

[13] M. Inuiguchi and M. Sakawa, “Minimax regret solution to linear programming problems with an interval objective function,” *Eur. J. Oper. Res.*, vol. 86, no. 3, pp. 526–536, 1995.

[14] H. E. Mausser and M. Laguna, “A new mixed integer formulation for the maximum regret problem,” *Int. Trans. Oper. Res.*, vol. 5, no. 5, pp. 389–403, 1998.

[15] G. Yu, “Min-max optimization of several classical discrete optimization problems,” *J. Optim. Theory Appl.*, vol. 98, no. 1, pp. 221–242, 1998.

[16] B. Lu, Y. Cao, M. J. Yuan, and J. Zhou, “Reference variable methods of solving min–max optimization problems,” *J. Glob. Optim.*, vol. 42, no. 1, pp. 1–21, 2008.

[17] M. Á. Sainz, P. Herrero, J. Armengol, and J. Vehí, “Continuous minimax optimization using modal intervals,” *J. Math. Anal. Appl.*, vol. 339, no. 1, pp. 18–30, 2008.

[18] Y. Feng, L. Hongwei, Z. Shuisheng, and L. Sanyang, “A smoothing trust-region Newton-CG method for minimax problem,” *Appl. Math. Comput.*, vol. 199, no. 2, pp. 581–589, 2008.

[19] H. Aissi, C. Bazgan, and D. Vanderpooten, “Min–max and min–max regret versions of combinatorial optimization problems: A survey,” *Eur. J. Oper. Res.*, vol. 197, no. 2, pp. 427–438, 2009.

[20] A. M. Cramer, S. D. Sudhoff, and E. L. Zivi, “Evolutionary algorithms for minimax problems in robust design,” *IEEE Trans. Evol. Comput.*, vol. 13, no. 2, pp. 444–453, Apr. 2009.

[21] C. Charalambous and A. R. Conn, “An efficient method to solve the minimax problem directly,” *SIAM J. Numer. Anal.*, vol. 15, no. 1, pp. 162–187, Feb. 1978.

[22] R. A. Polyak, “Smooth optimization methods for minimax problems,” *SIAM J. Control Optim.*, vol. 26, no. 6, pp. 1274–1286, Nov. 1988.

[23] E. C. Laskari, K. E. Parsopoulos, and M. N. Vrahatis, “Particle swarm optimization for minimax problems,” in *Proc. Congr. Evol. Comput. (CEC)*, vol. 2. Honolulu, HI, USA, 2002, pp. 1576–1581.

[24] M.-J. Tahk and B.-C. Sun, “Coevolutionary augmented Lagrangian methods for constrained optimization,” *IEEE Trans. Evol. Comput.*, vol. 4, no. 2, pp. 114–124, Jul. 2000.

[25] J. Kim and M.-J. Tahk, “Co-evolutionary computation for constrained min-max problems and its applications for pursuit-evasion games,” in *Proc. Congr. Evol. Comput.*, vol. 2. Seoul, South Korea, 2001, pp. 1205–1212.

[26] Y. Shi and R. A. Krohling, “Co-evolutionary particle swarm optimization to solve min-max problems,” in *Proc. Congr. Evol. Comput. (CEC)*, vol. 2. Honolulu, HI, USA, 2002, pp. 1682–1687.

[27] R. A. Krohling, F. Hoffmann, and L. S. Coelho, “Co-evolutionary particle swarm optimization for min-max problems using Gaussian distribution,” in *Proc. Congr. Evol. Comput. (CEC)*, vol. 1. Portland, OR, USA, Jun. 2004, pp. 959–964.

[28] H. J. C. Barbosa, “A coevolutionary genetic algorithm for constrained optimization,” in *Proc. Congr. Evol. Comput. (CEC)*, vol. 3. Washington, DC, USA, 1999, pp. 1605–1611.

[29] J. Branke and J. Rosenbusch, “New approaches to coevolutionary worst-case optimization,” in *Parallel Problem Solving from Nature—PPSN X*. Heidelberg, Germany: Springer, Sep. 2008, pp. 144–153.

[30] A. Zhou and Q. Zhang, “A surrogate-assisted evolutionary algorithm for minimax optimization,” in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Barcelona, Spain, Jul. 2010, pp. 1–7.

[31] R. I. Lung and D. Dumitrescu, "A new evolutionary approach to minimax problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, New Orleans, LA, USA, Jun. 2011, pp. 1902–1905.

[32] M. T. Jensen, "A new look at solving minimax problems with coevolutionary genetic algorithms," in *Metaheuristics: Computer Decision-Making*. Boston, MA, USA: Springer, 2004, pp. 369–384.

[33] J. Hur, H. Lee, and M.-J. Tahk, "Parameter robust control design using bimatrix co-evolution algorithms," *Eng. Optim.*, vol. 35, no. 4, pp. 417–426, 2003.

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

[35] 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.

[36] R. A. Sarker, S. M. Elsayed, and T. Ray, "Differential evolution with dynamic parameters selection for optimization problems," *IEEE Trans. Evol. Comput.*, vol. 18, no. 5, pp. 689–707, Oct. 2014.

[37] S.-M. Guo and C.-C. Yang, "Enhancing differential evolution utilizing eigenvector-based crossover operator," *IEEE Trans. Evol. Comput.*, vol. 19, no. 1, pp. 31–49, Feb. 2015.

[38] S. Biswas, S. Kundu, and S. Das, "Inducing niching behavior in differential evolution through local information sharing," *IEEE Trans. Evol. Comput.*, vol. 19, no. 2, pp. 246–263, Apr. 2015.

[39] S. Bandyopadhyay and A. Mukherjee, "An algorithm for many-objective optimization with reduced objective computations: A study in differential evolution," *IEEE Trans. Evol. Comput.*, vol. 19, no. 3, pp. 400–413, Jun. 2015.

[40] L. Tang, Y. Dong, and J. Liu, "Differential evolution with an individual-dependent mechanism," *IEEE Trans. Evol. Comput.*, vol. 19, no. 4, pp. 560–574, Aug. 2015.

[41] X. Qiu, J.-X. Xu, K. C. Tan, and H. A. Abbass, "Adaptive cross-generation differential evolution operators for multiobjective optimization," *IEEE Trans. Evol. Comput.*, vol. 20, no. 2, pp. 232–244, Apr. 2016.

[42] 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.

[43] Y.-L. Li *et al.*, "Differential evolution with an evolution path: A DEEP evolutionary algorithm," *IEEE Trans. Cybern.*, vol. 45, no. 9, pp. 1798–1810, Sep. 2015.

[44] W.-F. Gao, G. G. Yen, and S.-Y. Liu, "A dual-population differential evolution with coevolution for constrained optimization," *IEEE Trans. Cybern.*, vol. 45, no. 5, pp. 1108–1121, May 2015.

[45] M. Yang, C. Li, Z. Cai, and J. Guan, "Differential evolution with auto-enhanced population diversity," *IEEE Trans. Cybern.*, vol. 45, no. 2, pp. 302–315, Feb. 2015.

[46] 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.

[47] M. T. Jensen, "Robust and flexible scheduling with evolutionary computation," Ph.D. dissertation, Dept. Comput. Sci., Univ. at Aarhus, Aarhus, Denmark, 2001.

[48] J. W. J. Williams, "Algorithm 232: Heapsort," *Commun. ACM*, vol. 7, no. 6, pp. 347–348, 1964.

[49] M. L. Fredman and R. E. Tarjan, "Fibonacci heaps and their uses in improved network optimization algorithms," *J. ACM*, vol. 34, no. 3, pp. 596–615, Jul. 1987.

[50] H. J. C. Barbosa, "A genetic algorithm for min-max problems," in *Proc. 1st Int. Conf. Evol. Comput. Appl.*, 1996, pp. 99–109.

[51] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics Bull.*, vol. 1, no. 6, pp. 80–83, Dec. 1945.

[52] 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 Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[53] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Syst.*, vol. 9, no. 2, pp. 115–148, 1995.

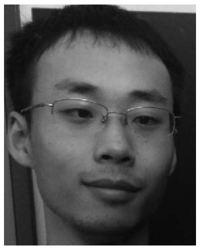
[54] S. Arimoto, S. Kawamura, and F. Miyazak, "Bettering operation of robots by learning," *J. Robot. Syst.*, vol. 1, no. 2, pp. 123–140, 1985.

[55] T.-Y. Kuc, J. S. Lee, and K. Nam, "An iterative learning control theory for a class of nonlinear dynamic systems," *Automatica*, vol. 28, no. 6, pp. 1215–1221, 1992.

[56] T.-J. Jang, C.-H. Choi, and H.-S. Ahn, "Iterative learning control in feedback systems," *Automatica*, vol. 31, no. 2, pp. 243–248, 1995.

[57] S.-I. Niculescu, J.-M. Dion, and L. Dugard, "Robust stabilization for uncertain time-delay systems containing saturating actuators," *IEEE Trans. Autom. Control*, vol. 41, no. 5, pp. 742–747, May 1996.

[58] S. Hong-Ye, H. Jian-Bo, L. James, and C. Jian, "Robust stabilization of uncertain time-delay systems containing nonlinear saturating actuators," *J. Zhejiang Univ. SCI. A*, vol. 1, no. 3, pp. 241–248, 2000.



Xin Qiu received the B.E. degree from Nanjing University, Nanjing, China, in 2012, and the Ph.D. degree from the National University of Singapore, Singapore, in 2016.

He was a member of the Computational Intelligence Research Group, National University of Singapore. His current research interests include differential evolution, applications of evolutionary algorithms, and neuroevolution.

Jian-Xin Xu (M'92–SM'98–F'11) received the Ph.D. degree from the University of Tokyo, Tokyo, Japan, in 1989.

In 1991, he joined the National University of Singapore, Singapore, where he is currently a Professor with the Department of Electrical Engineering. He has published over 200 journal papers and six books in the field of system and control, supervised 30 Ph.D. students and 15 research associates, as well as conducted near 30 research grants. His current research interests include learning theory, intelligent system and control, nonlinear and robust control, robotics, and precision motion control.

Yinghao Xu received the B.Eng. degree in electrical engineering from the National University of Singapore, Singapore, in 2016.

His current research interests include deep learning and data mining.

Mr. Xu was a recipient of the LEE KUAN YEW Gold Medal from the National University of Singapore.

Kay Chen Tan (SM'08–F'14) received the B.Eng. (First Class Hons.) degree in electronics and electrical engineering, and the Ph.D. degree from the University of Glasgow, Glasgow, U.K., in 1994 and 1997, respectively.

He is a Full Professor with the Department of Computer Science, City University of Hong Kong, Hong Kong. He has published over 200 refereed articles and five books.

Dr. Tan is the Editor-in-Chief of the IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION. He was the Editor-in-Chief of the *IEEE Computational Intelligence Magazine* from 2010 to 2013. He currently serves as the Editorial Board Member of over 20 journals. He is an elected member of the IEEE CIS AdCom for the period 2017–2019. He is an IEEE CIS Distinguished Lecturer for the period 2015–2017.