Adaptive Constrained Differential Evolutionary Algorithm Based on Interval Probability Mechanism

WEI Yanting, FENG Quanxi, and YUAN Sainan

***ABSTRACT*—This paper aims to improve premature convergence and low precision of the differential evolution algorithm. An adaptive constrained differential evolution algorithm based on a probability interval update mechanism (PIMDE) is proposed. First, the interval probability update mechanism is developed to adaptively generate the mutation probability value, and the hybrid mutation strategy is formulated by effectively selecting complementary mutation strategies based on the mutation probability. Subsequently, the adaptive mechanism is employed to dynamically adjust the mutation strategy and parameters; as a result, the global detection and local search ability of the algorithm can be balanced, and the search efficiency of the algorithm can be enhanced. To verify determine the performance of the proposed algorithm, ten standard constrained optimization problems and five engineering optimization problems are analyzed. As revealed from experimental results, PIMDE can effectively solve COPs with high precision and robustness.**

***Index Terms*—Constrained optimization, differential evolution algorithm, parameter self-adaptive, ε-constraint processing technique**

I. Introduction

THE models of constrained optimization problems (COPs) are derived from scientific research[1] and engineering applications[2,3]. Without loss of generality, the COPs (minimization problems) are defined below:

$$\begin{aligned} &\min \quad f(X) \\ &s.t. \quad g_i(X) \le 0, i = 1,2,...,m; \\ &\qquad h_j(X) = 0,\ j = m+1, m+2,...,n; \end{aligned} \tag{1}$$

where $X = [x_1, x_2, \mathrm{K}\ , x_n] \in \Omega$ denotes the decision variable. Ω represents the feasible region that meets the following boundary constraints:

$$Low_d \le x_d \le Upper_d,\ d = 1,2,...,D, \tag{2}$$

where $g_i(X)(i = 1,2,\mathrm{K}\ ,m)$ and $h_j(X)(j = m+1,\mathrm{K}\ ,n)$ denote the i^{th} inequality constraint and the n-j^{th} equality constraint, respectively. The D-dimensional real number space $S = \prod_{i=1}^{D}[Low_d, Upper_d]$ consisting of Low_d and $Upper_d$ (respectively the lower and upper boundaries of x_d) is termed as the decision space, and $\Omega \in S$.

To effectively solve COPs, Runarsson and Yao [4] proposed a stochastic sorting based differential evolution algorithm (SRDE) in 2000. In 2006, Tessema [5] developed an adaptive penalty based differential evolution algorithm (SPDE). In 2010, An ensemble of constraint handling techniques (ECHT) was presented by Malleepeddi *et al.* [6] to comprehensively exploit various existing constraint processing techniques.

Differential evolution (DE) refers to an efficient and robust heuristic optimization algorithm proposed by Storn and Price [7] in 1995. In [8], Brest proposed an approach termed as jDE, capable of controlling parameters *F* and *Cr* by an adaptive scheme. Mohamed [9] presented a novel constrained differential evolution (NDE) algorithm based on triangular mutation. To solve high-dimensional optimization problems in a continuous space, Mohamed *et al.* [10] integrated the triangular mutation with DE/rand/1/bin and then proposed the differential evolution algorithm (ANDE). In [11], Wang presented a modified CoDE-based approach termed as CCoDE, capable of solving constrained optimization problems.

The setting of control parameters and the selection of evolution strategy are critical to determine the performance of constrained DE (CDE). Overall, when an adaptive CDE algorithm complies with multiple mutation strategies, it usually adopts a fixed mutation probability and overlooks the variation of population. To effectively select the mutation strategy, this study proposes an adaptive constrained differential evolution algorithm based on the probability interval mechanism (PIMDE). First, the algorithm splits the given continuous range into intervals and gives the probability. According to the two initial values of the interval, two consecutive values are generated as the mutation probability. A hybrid mutation strategy is presented by exploiting the mutation probability to adaptively select mutation strategies and balance the global search and local search. Subsequently, the novel population is taken with the ε-constraint processing technique, and the re-initialization scheme is employed to jump out of the local optimum and enhance the population diversity. Next, to promote the algorithm to be more robust and adaptable, the adaptive

Manuscript received Nov. 22, 2019; revised Jun. 28, 2021. This paper was supported in part by the National Natural Science Foundation of China (61763008, 11661030, 11401357, 11861027), Natural Science Foundation of Guangxi (2018GXNSFAA138095), Fangchenggang Scientific research and technology development plan 2014 No. 42, and Project supported by Program to Sponsor Teams for Innovation in the Construction of Talent Highlands in Guangxi Institutions of Higher Learning ([2011]47) and Doctoral Research Fund of Guilin University of Technology.

Wei Yanting is Master Degree Candidate of School of Science, Guilin University of Technology, Guilin, 541006, PR China, e-mail: 1243566132@qq.com.

Feng Quanxi (corresponding author) is a professor of School of Science, Guilin University of Technology, Guilin, 541004, PR China, TEL: +86-15977437575; e-mail: fqx9904@163.com.

Yuan Sainan is Master Degree Candidate of School of Science, Guilin University of Technology, Guilin, 541006, PR China, e-mail: 1157464726@qq.com.

parameter control mechanism is introduced again.

II. Adaptive Constrained Differential Evolution Algorithm

To remedy the defects of DE premature convergence and low optimization precision, an adaptive constrained differential evolution algorithm proposed in this study is primarily improved from according to the selection of mutation strategies. First, the algorithm presents a hybrid mutation strategy that exploits mutation probabilities to take mutation strategies (e.g., DE\rand\1, triangle mutation strategy, and DE\best\2). Subsequently, the algorithm introduces an interval probability update mechanism to reward and update the mutation probability, as an attempt to ensure that an appropriate mutation strategy is adopted. The scaling factor *F* of the mutation strategy is adaptive. Next, the algorithm adopts a binomial crossover strategy, with the adaptive crossover probability factor *Cr*. Afterwards, the algorithm adopts the constraint handling method to update population. Lastly, the algorithm substitutes the poor individuals with a restart plan and re-initializes the mutation for each of the worst individuals to maintain the diversity of populations.

A. Hybrid Mutation Strategy

1. Introduction to Mutation Strategies

The triangular mutation was first presented in 2003 by Fan and Lampinen [14]. Subsequently, Mohamed [9] improved triangular mutation to enhance the local search performance of the DE algorithm, balance the global exploration capability and local development trend, and expedite the convergence. Set $\boldsymbol{V}$ as the mutation vector, and the triangle mutation (expressed as DE\rand\ triangular) is expressed as follows:

$$\begin{aligned} V = {} & X_c + F_1 * (X_{best} - X_{better}) \\ & + F_2 * (X_{best} - X_{worst}) \\ & + F_3 * (X_{better} - X_{worst}). \end{aligned} \tag{3}$$

The basic vector X_c is calculated by:

$$X_c = w_1 * X_{best} + w_2 * X_{better} + w_3 * X_{worst} \tag{4}$$

where X_{r1} , X_{r2} , and X_{r3} ($r1 \neq r2 \neq r3 \neq i$) denote three individuals that are randomly selected from the parent population *Pop* . X_{best} , X_{better} , and X_{worst} are obtained by classifying them from good to bad in accordance with the objective function value and the degree of constraint violation with the ε -constraint handling method [15]. F_1 , F_2 , and F_3 indicate mutation coefficients generated from uniform distribution in [0, 1]. $w_i, i = 1,2,3$ represents the weight satisfying $w_i > 0$ and $\sum_{i=1}^{3} w_i = 1$,which is calculated by:

$$w_i = p_i / \sum_{i=1}^{3} p_i, \quad i = 1,2,3, \tag{5}$$

Further, p_i holds that:

$$p_1 = 1, p_2 = rand(0.75,1), p_3 = rand(0.5, p_2) \tag{6}$$

where $rand(a,b)$ denotes a uniform random sampling function between *a* and *b*. Besides the triangular mutation operator, this study also adopts other common mutation operators (e.g., DE\rand\1 and DE\best\2) as follows:

DE\rand\1:

$$V = X_{r1} + F * (X_{r2} - X_{r3}); \tag{7}$$

DE\best\2:

$$V = X_{best} + F * (X_{r1} - X_{r2}) + F * (X_{r3} - X_{r4}), \tag{8}$$

where *F* denotes the scaling factor between [0, 1]; r_1, r_2, r_3, r_4, r_5 are randomly selected in the range [1, *NP*], and they differ from each other. The X_{best} indicates the individual exhibiting the optimal fitness in the current population.

2. Interval Probability Update Mechanism

Different mutation strategies in DE exhibit different performance. For hybrid strategies, how to select different mutation strategies based on the actual problems is considered a difficult issue. An interval probability update mechanism is developed in the present section to dynamically adjust the probability values of the probability points W_1 and W_2 . The initial values of W_1 and W_2 are generated randomly with the roulette method; subsequently, they are updated in line with the performance of the mutation strategies.

① Initial Value Selection

Most hybrid strategies generally exploit a fixed probability value, while overlooking the middle probability value, thereby adversely affecting the performance of the mutation strategy. In contrast, the present section designs an interval probability selection mechanism in which the values from a given probability interval are randomly taken as the mutation probability; as a result, more mutation probability values can be selected. Meanwhile, the probability interval is adjusted regulated dynamically based on the performance of the mutation strategy. To select the mutation probability, interval [0, 1] is first split into multiple sub-intervals, from which two sub-intervals are randomly selected. From the mentioned two sub-intervals, W_1 and W_2 are respectively generated. Assume that $W_1 < W_2$, the continuous value produced is expressed as:

$$W_i = \underline{S} + rand * (\overline{S} - \underline{S}), i = 1,2 \tag{9}$$

where $\underline{S}$ and $\overline{S}$ respectively denote the front and rear endpoints of the subinterval; *rand* represents a uniform random number between [0, 1]. The pseudocode for this mechanism is shown in Algorithm 1.

Algorithm 1: Pseudocode of the initial probability value selection mechanism.

1: $r1_{set} \leftarrow r1_1, r1_2, \mathrm{K}, r1_{nr1}; r2_{set} \leftarrow r2_1, r2_2, \mathrm{K}, r2_{nr2}$; *w1p* and *w2q* are the probabilities of the intervals in $r1_{set}$ and $r2_{set}$, respectively;
2: $Q1(1) = w1p(1); Q2(1) = w2q(1)$;
3: **for** *k*=2: length(*w1p*) **do**
4: Calculate the cumulative probability $Q1$ and $Q2$ for *w1p* and *w2q*, respectively, to prepare for roulette;
5: **end for**
6: Randomly generate two numbers between 0 and 1, perform roulette selection, and set them as the Lower boundary of the intervals $r1_{set}$ and $r2_{set}$;
7: **if** $r1_{set} \geq r2_{set}$ **then**;
8: $r2_{set}$ should be regenerated to ensure $r1_{set} < r2_{set}$;
9: **end if**
10: generate W_1 and W_2 such that $W_i = \underline{S} + rand * (\overline{S} - \underline{S}), i = 1,2$.

② Probability Interval Update

W_1 and W_2 ($W_1 < W_2$) are updated by the probability interval;

they do not pertain to the identical interval. The two mutation probabilities W_1 and W_2 take values in (0, 0.9) and (0.1, 1), respectively, and both interval widths reach 0.1. Assume that K intervals $A=\{a_1,\mathrm{L}\ ,a_k\}$ are adopted, and $q_a(t)$ is set as the known empirical evaluation of interval a, which is updated as:

$$q_a(t+1)=q_a(t)+\alpha*[r_a(t)-q_a(t)] \tag{10}$$

where $\alpha\in(0,1]$ denotes the adaptation rate; the initial value of $q_a(t)$ is zero; $r_a(t)$ represents the reward that interval a obtains when running at time t is as:

$$r_a(t)=\frac{\sum_{i=1}^{|S_a|}S_a(i)}{|S_a|} \tag{11}$$

where S_a denotes the probability of successful parent individual replacement in each interval of W_1 and W_2 complying with the mutation strategy.

Based on the empirical evaluation value, the interval probability $W_i(t)(i=1,2)$ is updated as follows:

$$W_i(t+1)=W_{i,\min}+(1-K*W_{i,\min})\frac{q_a(t+1)}{\sum_{i=1}^{K}q_i(t+1)} \tag{12}$$

where $W_{i,\min}\in(0,1)$ denotes the smallest probability value of $W_i(t)(i=1,2)$, which is employed to ensure that $W_i(t)$ is not 0. In 2010, Gong *et al.* [13] proposed to select the appropriate mutation strategy based on a probability matching technique and update the strategy by complying with the fitness variation of the applied strategy. In this study, the developed interval probability update mechanism generates mutation probability based on the probability interval, and then the adaptive mutation strategy is selected.

3. Hybrid Mutation Strategies

An ideal constrained optimization search algorithm strikes a balance between diversity and convergence, as well as a balance between constraints and objective functions. Thus, three different mutation strategies, i.e., DE\rand\1, DE\best\2, and DE\rand\ triangular, are involved in the search algorithm designing as a hybrid mutation strategy. The hybrid mutation strategy operates as follow. If the random number is in $(0, W_1]$, the DE\rand\1 strategy is adopted for the global search; if it falls in $(W_1, W_2]$, the triangle mutation strategy is adopted to facilitate local search; otherwise, it falls in $(W_2, 1)$, and the DE\best\2 strategy is employed to expedite the local search convergence speed. The hybrid mutation strategy is expressed as:

$$mutation\ strategy=\begin{cases}DE\setminus rand\setminus 1, & rand_i\in(0,W_1];\\ DE\setminus rand\setminus triangular, & rand_i\in(W_1,W_2];\\ DE\setminus best\setminus 2, & rand_i\in(W_2,1).\end{cases} \tag{13}$$

B. Parameter Adaptation

In 2017, Zhou *et al.* [12] proposed a parameter adaptive control method. Assuming that F_i and Cr_i are generated independently based on the successful parameters of the previous generation, the formula of Cr_i is written as:

$$Cr_i=randn_i(\mu_{Cr},0.1) \tag{14}$$

where $randn_i(\mu_{Cr},0.1)$ denotes the generation of a uniformly distributed random value; 0.1 represents the standard deviation; μ_{Cr} represents the mean value, which is calculated by:

$$\mu_{Cr}=(1-c)*\mu_{Cr}+c*mean_A(S_{Cr}) \tag{15}$$

where $mean_A(x)$ denotes the commonly used arithmetic mean; S_{Cr} refers to a set of successful crossover probabilities of the previous generation; c is a constant between 0 and 1, which is set to 0.001 in this study after repeated trials and comparison. The adaption process of F_i is similar to that of Cr_i:

$$\begin{cases}F_i=randc_i(\mu_F,0.1);\\ \mu_F=(1-c)*\mu_F+c*mean_L(S_F);\\ mean_L(S_F)=\sum_{S_F}F^2/\sum_{S_F}F.\end{cases} \tag{16}$$

where $randc_i(\mu_F,0.1)$ denotes a random value with Cauchy distribution based on parameter value μ_F and 0.1; $mean_L(S_F)$ represents the Lamer mean [12]; S_F refers to the set of mutation factors promoting the offspring to successfully replace with the parent.

C. ε -constraint Handling Method

The ε-constraint handling method [15] is employed to solve constraints in this study. For solutions x_i and x_j, x_i is preference to x_j if both of them meet the following conditions:

$$\begin{cases}f(x_i)<f(x_j), & if\ G(x_j)\le\varepsilon\wedge G(x_j)\le\varepsilon;\\ f(x_i)<f(x_j), & if\ G(x_i)=G(x_j);\\ G(x_i)<G(x_j), & otherwise.\end{cases} \tag{17}$$

where ε decreases with the growth of its iterative generation; its calculation formula is as follows:

$$\varepsilon=\begin{cases}\varepsilon_0(1-\frac{t}{T})^{cp} & ,if\ \frac{t}{T}\le p;\\ 0 & ,otherwise.\end{cases} \tag{18}$$

$$cp=-\frac{\log\varepsilon_0+\lambda}{\log(1-p)}, \tag{19}$$

where ε is initialized as $\varepsilon_0=V(X_{0.2*PS})$; $X_{0.2*PS}$ denotes the top $0.2\times PS^{\text{th}}$ individual according to violation degree, and λ is 6. $T\in[0.1*T_{\max},0.8*T_{\max}]$ are the control parameter ranges, and $cp\in[2,10]$.

D.Re-initialization

For some nonlinear programming problems coupled with extremely complicated constraints, the infeasible regions have highly nonlinear and multi-modal properties. In this scenario, the population is easily stagnant in the infeasible area, causing population diversity to decline. To solve this problem, this study adopts a restart mechanism, similar to [10], when the standard deviation of the overall constraint violation degree or the standard deviation of the overall objective function value is less than a predefined threshold (overall set to $1.0\times e^{-8}$). if the population is not feasible, the restart plan is triggered, and all individuals are randomly generated in the decision space. Meanwhile, the worst individual is substituted with randomly generated one.

E. Algorithm Flow

In this study, the flowchart of the adaptive constrained differential evolutionary algorithm based on the interval

probability mechanism (PIMDE) is illustrated in Fig 1. The detailed steps are presented below.

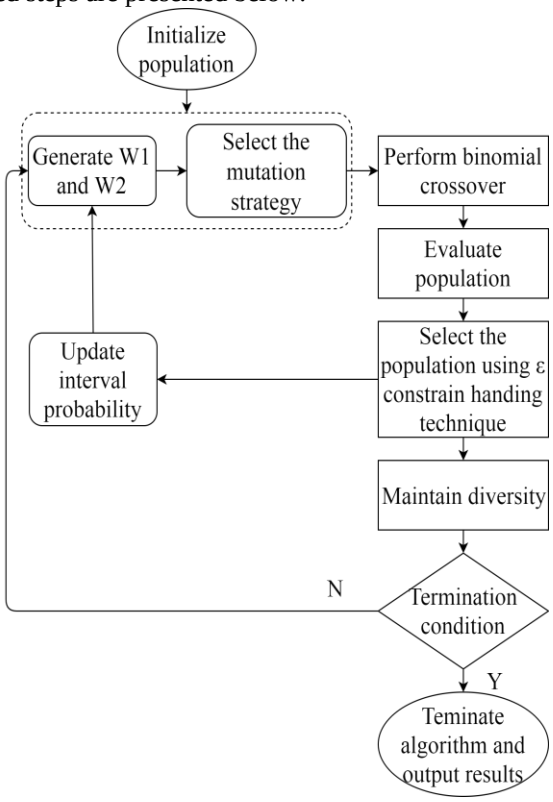


Fig. 1. Flow chart of the PIMDE algorithm

Step 1: Initialize the population.

Step 2: Set the initial probability value of intervals *w1p* (0, 0.9) and *w2q* (0.1, 1). Subsequently, mutation strategies DE\rand\1, DE\rand\triangular, and DE\current2best\1 are selected. In addition, *goodCR*=*goodF*=0.5. The initial value of ε is $\varepsilon_0 = V(X_{0.2*PS})$.

Step 3: For each individual, the values of *F* and *Cr* are generated by exploiting uniform distribution and Cauchy distribution, respectively.

Step 4: The adaptive interval probability update mechanism is employed to generate W_1 and W_2, *NP* random numbers $\text{rand}_i(i = 1,\text{L}\ ,NP)$ are generated, and the test vector is yielded by adaptively adopting a strategy.

Step 5: Perform a binomial crossover to obtain a test population.

Step 6: Select the population *pop* with the ε-constraint handling technique. Meantime, the novel interval probabilities *w1p* and *w2q* are generated by the probability update mechanism.

Step 7: Restart the worst individual [9] to yield the next generation of *Pop*.

III. Algorithm Testing and Analysis

To verify the effectiveness of the proposed algorithm, PIMDE is compared with other algorithms in terms of ten standard Constrained Optimization Problems, abbreviate as COPs, and five engineering COPs. The five engineering COPs consist of tension-pressure spring design, welded beam design, pressure vessel design, hydrodynamic thrust bearing design, as well as butterfly spring design. The algorithms are operating independently for 30 iterations. The optimal value (*MinBest*), an average value (*MinMean*), and standard deviation (*Std*) of the 30-iterations results act as the performance assessment indexes. A statistical analysis is conducted according to the obtained results. Three statistical analysis methods (i.e., the Mann-Whitney rank-sum test, Iman-Davenport test, and Wilcoxon signed-rank test) are adopted for statistical analysis.

A. Result Analysis of Standard COPs

The maximum number of fitness function assessment, abbreviate as *MaxFEs*, for ten standard test problems are set to 500,000. The population sizes of the PIMDE, CCoDE [11], ECHTEP [6], NDE [9], SPDE [4], SRDE [4], and jDE [8] are all set to 100. The initial values of μ_F and μ_{CR} are set to 0.5, and the other parameters are identical to those of the original papers. The test results are listed in Table I, which are represented with four fonts. The skewed and bold font represents the results of the PIMDE algorithm. As revealed by the regular font, the results of the comparative algorithm are worse than those of PIMDE. The bold font suggests that the results of PIMDE are similar to those of the other algorithms. The bold font and gray shading cell indicate that the numerical results of the algorithms outperform those of the PIMDE algorithm. The "*NAN*" means that the corresponding feasible solution is not identified.

To compare the degree of difference between different algorithms for the identical problem, the Mann-Whitney rank-sum test results with a significance level of 0.05 are listed in Table II. In Table II, where '+', '≈', and '−' denote that the results of PIMDE is superior to, similar to, or slightly worse than those of the other algorithms, respectively.

As presented in Table I, the PIMDE, CCoDE, ECHTEP, NDE, and jDE algorithms can achieve feasible solutions for all the problems, except for SPDE and SRDE algorithms. The numerical results of g01, g04, g05, g06, g11, g12, and g13 solved by PIMDE are superior to or similar to those by algorithms (e.g., CCoDE, ECHTEP, NDE, SPDE, SRDE, and jDE). For function g02, the numerical results of PIMDE are better than those of ECHTEP, SPDE, and SRDE, and they are slightly inferior to those of CCoDE and NDE. For function g08, the results of the proposed algorithm outperform those of SPDE, SRDE, and jDE, while those are slightly worse than those of ECHTEP. In terms of function g09, the numerical results of the proposed algorithm are better than those of ECHTEP and SRDE, while they are slightly worse than those of CCoDE, NDE, SPDE, and jDE.

As revealed from the comparison of the Mann-Whitney rank-sum test results in Table II, the results of PIMDE outperformed those of other algorithms on 1, 6, 3, 8, 9, and 3 of 10 functions. Furthermore, the results of PIMDE tied with those of the other algorithms for six functions 7, 3, 5, 1, 1, and 5. Finally, the results of the PIMDE are worse than those of the other algorithms for functions 2, 1, 2, 1, 0, and 2.

The Iman-Davenport test is performed on the mean (*MinMean*) and the best value (*MinBest*) to compare the differences of all the algorithms in terms of different test problems.

According to the Iman-Davenport test in Table II, CCoDE ranks first for the average value, and the ranks of PIMDE and

TABLE I
EXPERIMENTAL RESULTS OF STANDARD COPS

Algorithm	Statistics	g01	g02	g04	g05	g06
PIMDE	***MinBest***	***-1.50E+01***	***-8.02E-01***	***-3.07E+04***	***5.13E+03***	***-6.96E+03***
	MinMean	***-1.50E+01***	***-8.00E-01***	***-3.07E+04***	***5.13E+03***	***-6.96E+03***
	MinStd	***0.00E+00***	***4.21E-03***	***1.09E-11***	***9.09E-13***	***1.82E-12***
CCoDE	*MinBest*	**-1.50E+01**	**-8.04E-01**	**-3.07E+04**	**5.13E+03**	**-6.96E+03**
	MinMean	**-1.50E+01**	**-8.03E-01**	**-3.07E+04**	**5.13E+03**	**-6.96E+03**
	MinStd	**0.00E+00**	**2.87E-03**	**1.09E-11**	**9.09E-13**	**1.82E-12**
ECHTEP	*MinBest*	-1.50E+01	-8.04E-01	**-3.07E+04**	5.13E+03	**-6.96E+03**
	MinMean	-1.50E+01	-7.91E-01	**-3.07E+04**	5.13E+03	**-6.96E+03**
	MinStd	1.85E-10	1.02E-02	**1.09E-11**	9.21E-02	**1.82E-12**
NDE	*MinBest*	-1.50E+01	**-8.02E-01**	**-3.07E+04**	5.13E+03	**-6.96E+03**
	MinMean	-1.50E+01	**-8.00E-01**	**-3.07E+04**	5.13E+03	**-6.96E+03**
	MinStd	8.25E-11	**1.92E-03**	**1.09E-11**	2.86E+01	**1.82E-12**
SPDE	*MinBest*	-1.50E+01	-8.04E-01	-3.39E+04	6.55E+04	-6.96E+03
	MinMean	-1.36E+01	-7.94E-01	-3.38E+04	6.55E+04	-6.96E+03
	MinStd	2.24E+00	1.24E-02	7.04E+01	NAN	1.47E+00
SRDE	*MinBest*	-1.50E+01	-3.03E-01	-3.39E+04	6.55E+04	-6.56E+03
	MinMean	-1.28E+01	-2.35E-01	-3.39E+04	6.55E+04	6.55E+04
	MinStd	1.11E+00	2.74E-02	5.09E+00	NAN	NAN
jDE	*MinBest*	**-1.50E+01**	**-8.04E-01**	**-3.07E+04**	5.13E+03	**-6.96E+03**
	MinMean	**-1.50E+01**	**-8.02E-01**	**-3.07E+04**	5.15E+03	**-6.96E+03**
	MinStd	**0.00E+00**	**3.74E-03**	**1.09E-11**	4.02E+01	**1.82E-12**

TABLE I (CONT.)
EXPERIMENTAL RESULTS OF STANDARD COPS

Algorithm	Statistics	g08	g09	g11	g12	g13
PIMDE	***MinBest***	***-9.58E-02***	***6.81E+02***	***7.50E-01***	***-1.00E+00***	***5.39E-02***
	MinMean	***-9.58E-02***	***6.81E+02***	***7.50E-01***	***-1.00E+00***	***5.39E-02***
	MinStd	***2.78E-17***	***5.26E-13***	***1.11E-16***	***0.00E+00***	***2.48E-17***
CCoDE	*MinBest*	**-9.58E-02**	**6.81E+02**	**7.50E-01**	**-1.00E+00**	5.39E-02
	MinMean	**-9.58E-02**	**6.81E+02**	**7.50E-01**	**-1.00E+00**	5.39E-02
	MinStd	**2.78E-17**	**3.98E-13**	**1.11E-16**	**0.00E+00**	3.42E-17
ECHTEP	*MinBest*	**-9.58E-02**	6.81E+02	7.50E-01	**-1.00E+00**	5.39E-02
	MinMean	**-9.58E-02**	6.81E+02	7.50E-01	**-1.00E+00**	5.39E-02
	MinStd	**1.70E-17**	1.05E-05	2.57E-05	**0.00E+00**	2.19E-07
NDE	*MinBest*	**-9.58E-02**	**6.81E+02**	**7.50E-01**	**-1.00E+00**	5.39E-02
	MinMean	**-9.58E-02**	**6.81E+02**	**7.50E-01**	**-1.00E+00**	5.39E-02
	MinStd	**2.78E-17**	**3.91E-13**	**1.11E-16**	**0.00E+00**	3.42E-17
SPDE	*MinBest*	-9.58E-02	**6.81E+02**	1.00E+00	**-1.00E+00**	6.55E+04
	MinMean	-9.58E-02	**6.81E+02**	6.55E+04	**-1.00E+00**	6.55E+04
	MinStd	2.83E-17	**4.31E-13**	*NAN*	**0.00E+00**	*NAN*
SRDE	*MinBest*	-9.58E-02	6.81E+02	7.50E-01	**-1.00E+00**	6.55E+04
	MinMean	-9.58E-02	6.81E+02	8.86E-01	**-1.00E+00**	6.55E+04
	MinStd	2.94E-17	3.53E-10	1.14E-01	**0.00E+00**	*NAN*
jDE	*MinBest*	-9.58E-02	**6.81E+02**	**7.50E-01**	**-1.00E+00**	4.55E-01
	MinMean	-9.58E-02	**6.81E+02**	**7.50E-01**	**-1.00E+00**	8.86E-01
	MinStd	2.83E-17	**4.27E-13**	**1.11E-16**	**0.00E+00**	9.15E-02

CCoDE are similar and rank second. For the analysis of the best value, the performance of the PIMDE algorithm is relatively flat with CCoDE and jDE and outperforms the other four algorithms.

B. Comparisons of Engineering COPs

To more specifically verify the effectiveness of the PIMDE algorithm in practical problems, this study compares the PIMDE with six algorithms (i.e., CCoDE, ECHTEP, NDE, SPDE, SRDE and jDE) in five engineering COPs. For

TABLE II
STATISTICAL TEST RESULTS OF STANDARD COPS

Statistic Test	Symbol	PIMDE vs	CCoDE	ECHTEP	NDE	SPDE	SRDE	jDE
Mann-Whitney	Win	+	1	6	3	8	9	3
	Tied	≈	7	3	5	1	1	5
	Lose	−	2	1	2	1	0	2
Iman-Davenport	MinMean's Rank	**3.25**	**2.7**	4.6	3.5	4.9	5.5	3.55
	MinBest's Rank	**3.65**	**3.1**	4.65	4.05	4.1	5.2	**3.25**

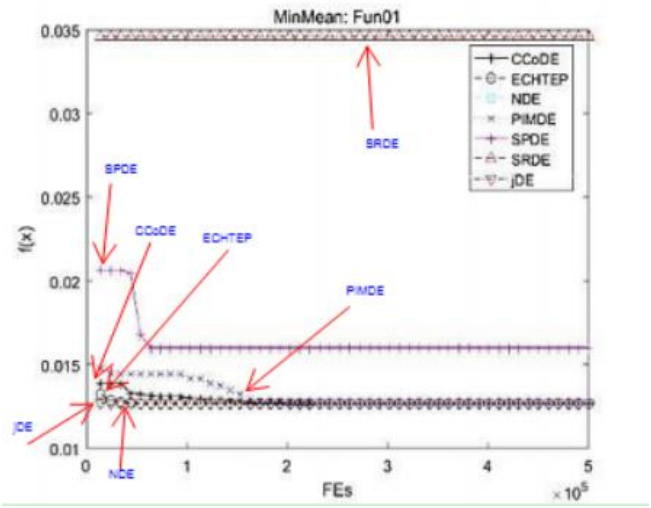
TABLE III
EXPERIMENTAL RESULTS OF ENGINEERING COPS

Algorithm	Statistics	Tension-pressure spring design	Welded beam design	Pressure vessel design	Hydrodynamic thrust bearing design	Butterfly spring design
PIMDE	***MinBest***	**1.27E-02**	**1.72E+00**	**5.89E+03**	**1.63E+03**	**1.98E+00**
	MinMean	**1.27E-02**	**1.72E+00**	**5.89E+03**	**1.63E+03**	**1.98E+00**
	MinStd	**5.98E-18**	**1.11E-15**	**9.09E-13**	**4.55E-13**	**1.41E-15**
CCoDE	*MinBest*	1.27E-02	**1.72E+00**	**5.89E+03**	1.63E+03	**1.98E+00**
	MinMean	1.27E-02	**1.72E+00**	**5.89E+03**	1.63E+03	**1.98E+00**
	MinStd	6.50E-18	**1.11E-15**	**9.09E-13**	7.26E-13	**1.33E-15**
ECHTEP	*MinBest*	1.27E-02	**1.72E+00**	**5.89E+03**	1.63E+03	1.98E+00
	MinMean	1.27E-02	**1.72E+00**	**5.89E+03**	1.63E+03	2.04E+00
	MinStd	2.20E-08	**2.71E-11**	**9.09E-13**	4.95E-07	6.22E-02
NDE	*MinBest*	1.27E-02	**1.72E+00**	**5.89E+03**	**1.63E+03**	1.98E+00
	MinMean	1.27E-02	**1.72E+00**	**5.89E+03**	**1.63E+03**	2.00E+00
	MinStd	6.17E-18	**1.11E-15**	**9.09E-13**	**4.55E-13**	4.64E-02
SPDE	*MinBest*	1.27E-02	**1.72E+00**	5.89E+03	1.65E+03	2.12E+00
	MinMean	1.60E-02	**3.31E+00**	5.91E+03	1.68E+03	2.13E+00
	MinStd	4.75E-03	**1.23E+00**	7.97E+00	1.61E+01	9.30E-03
SRDE	*MinBest*	1.33E-02	**2.42E+00**	**5.89E+03**	3.58E+03	6.55E+04
	MinMean	6.55E+04	**6.55E+04**	**5.89E+03**	6.55E+04	6.55E+04
	MinStd	*NAN*	***NAN***	**9.09E-13**	*NAN*	*NAN*
jDE	*MinBest*	1.27E-02	**1.72E+00**	**5.89E+03**	**1.63E+03**	**1.98E+00**
	MinMean	1.27E-02	**1.72E+00**	**5.89E+03**	**1.63E+03**	**1.98E+00**
	MinStd	6.04E-13	**1.11E-15**	**9.09E-13**	**4.55E-13**	**1.13E-15**

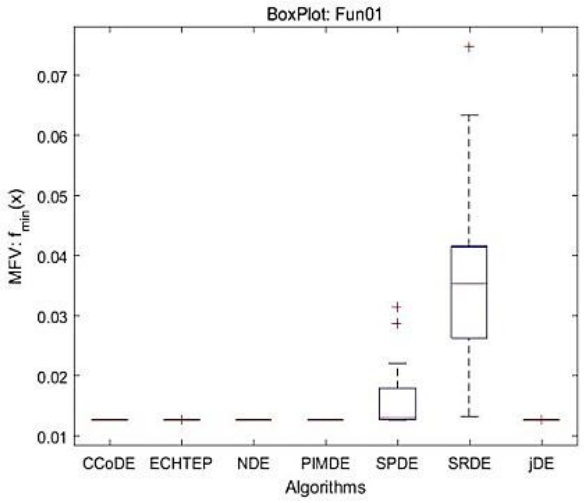
simplified comparison, the population size is 100, and the maximum number of fitness function evaluations, i.e., *MaxFEs*, is 500,000. The other parameter settings comply with those of the original papers. The test results are listed in Table III, where the data are labeled likewise as in Section 3.1. The setting of the Mann-Whitney test and Iman-Davenport test used in the present section is identical to with that in Section 3.1. Besides test the degree between PIMDE and the other algorithms, the Wilcoxon symbol rank test is performed in the present section. Table III lists the experimental results of the engineering COPs. The test results of three statistical analysis methods are compared, as listed in Table IV. Fig. 2 present the convergence curves of the test results and the box plots for the tension-pressure spring design (F01), the welded beam design (F02), as well as the pressure vessel design (F03).

Table III suggests that PIMDE, CCoDE, ECHTEP, NDE, SPDE, and jDE, except for the SRDE algorithm, can achieve a feasible solution for all the engineering problems. The numerical results of PIMDE for the design of tension spring, welded beam design, pressure vessel design, and hydrodynamic thrust bearing design are close to or superior to those of the CCoDE, ECHTEP, NDE, SPDE, SRDE, and jDE algorithms. The numerical results of the PIMDE solution for the butterfly spring design are slightly inferior to those of CCoDE and jDE, but better than the other 4 algorithms.

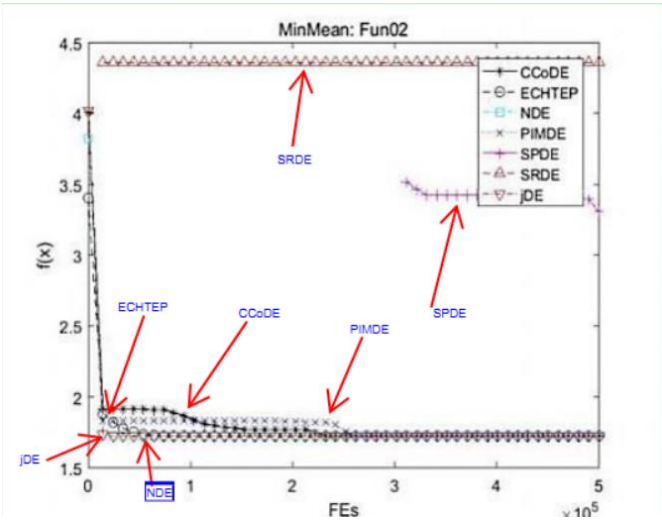
As revealed from the comparison of Mann-Whitney rank-sum test results, the results of PIMDE algorithm outperformed than those of CCoDE, ECHTEP, NDE, SPDE, SRDE, and jDE on 2, 4, 2, 5, 4, and 1 of 5 engineering COPs. The results of PIMDE algorithm are similar to those of the others on 2, 1, 3, 0, 1 and 3, lost with the others on 1, 0, 0, 0, 0 and 1, respectively. The results of 's test indicate that for *MinMean*, the rank Iman-Davenport value of the PIMDE algorithm is 2.6, which ranks second in 7 algorithms. For *MinBest*, the rank values of the PIMDE and NDE are 2.9, better than the others. From these two comparisons, it is concluded that the performance of the PIMDE algorithm is similar with that of CCoDE; both algorithms can successfully solve engineering constrained optimization problems, and



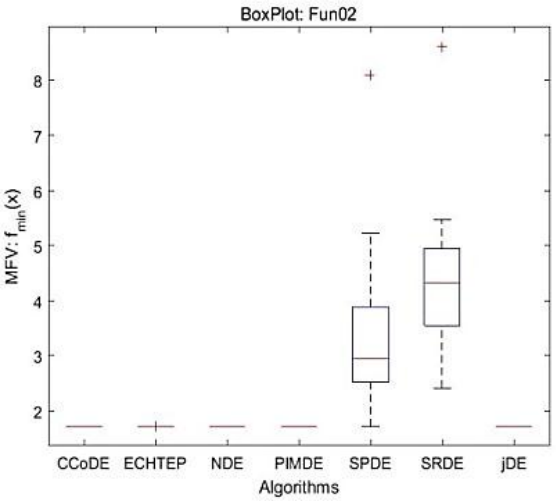
(a) the tension-pressure spring design



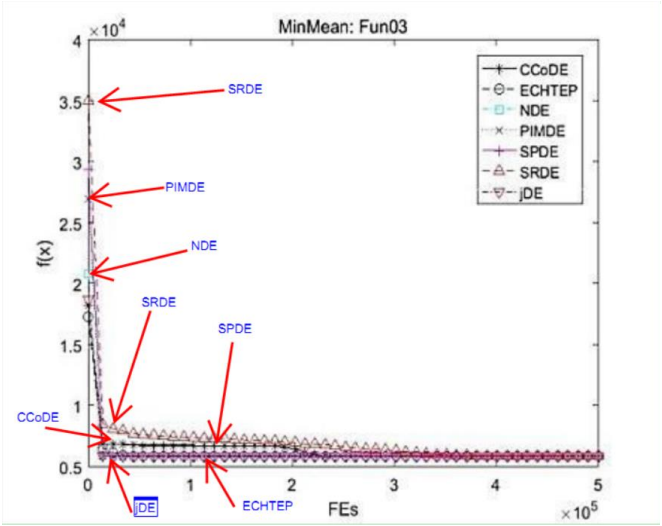
(b) the tension-pressure spring design



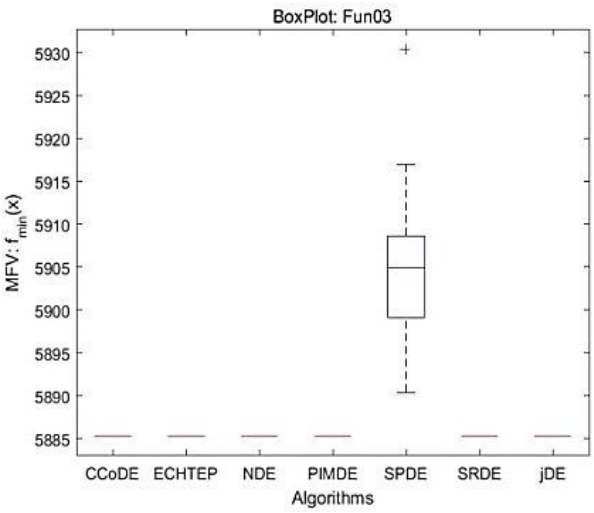
(c) the welded beam design



(d) the welded beam design



(e) the pressure vessel design



(f) the pressure vessel design

Fig. 2. Convergence curves (a, c, e) and the box plots (b, d, f) for engineering COPs

TABLE IV
STATISTICAL TEST RESULTS OF ENGINEERING COPS

Statistic Test	Symbol	PIMDE vs	CCoDE	ECHTEP	NDE	SPDE	SRDE	jDE
Mann-Whitney	Win	+	2	4	2	5	4	1
	Tied	≈	2	1	3	0	1	3
	Lose	−	1	0	0	0	0	1
Iman-Davenport	MinMean's Rank	**2.6**	**2.3**	4.7	2.9	6.2	6.3	3
	MinBest's Rank	**2.9**	3.3	3.8	2.9	5.5	6.3	3.3
Wilcoxon	Signed Rank	R^-	7.5	2.5	7	0	2.5	7
		R^+	7.5	12.5	8	15	12.5	8

both algorithms outperform the other comparison algorithms. In a comparison of the results of the Wilcoxon symbol rank test, the symbol rank test R^- are overall smaller than R^+, demonstrating that PIMDE algorithm performs the best.

IV. CONCLUSION

In this study, an adaptive constrained differential evolution algorithm largely based on interval probability update mechanism is proposed. In the proposed algorithm, a hybrid mutation strategy is presented by designing the interval probability update mechanism to adaptively generate the mutation probability value, which can be effectively employed to select the complementary mutation strategy. Such mutation strategy is capable of achieving a good balance the exploration ability and exploitation ability, while enhancing the optimization accuracy and convergence speed of the algorithm. Furthermore, the parameter adaptive mechanism is employed to adjust the *F* and *CR* values according to different test problems to avoid manual parameter adjustment. Moreover, the restart plan is introduced into PIMDE to enable the population to jump out of the stranded area during the evolution. As revealed from the simulation results of 10 test functions and 5 engineering constrained optimization problems, the PIMDE algorithm exerts better solution effect than the ECHTEP, NDE, SPDE, SRDE and jDE algorithms, and it exhibits similar performance to the CCoDE algorithm.

In the subsequent study, the robustness of the algorithm will be enhanced continuously, so the algorithm is capable of solving more constrained optimization problems.

REFERENCES

[1] Jing Wang, "A Novel Firefly Algorithm for Portfolio Optimization Problem," IAENG International Journal of Applied Mathematics, vol. 49, no.1, pp45-50, 2019.

[2] Gonggui Chen, Xingting Yi, Zhizhong Zhang, and Shiyuan Qiu, "Solving Optimal Power Flow Using Cuckoo Search Algorithm with Feedback Control and Local Search Mechanism," IAENG International Journal of Computer Science, vol. 46, no.2, pp321-331, 2019.

[3] Seyyid Ahmed Medjahed, Tamazouzt Ait Saadi, Abdelkader Benyettou, and Mohammed Ouali, "Binary Cuckoo Search Algorithm for Band Selection in Hyperspectral Image Classification," IAENG International Journal of Computer Science, vol. 42, no.3, pp183-191, 2015.

[4] Runarsson Thomas P., Xin Yao, "Stochastic Ranking for Constrained Evolutionary Optimization," IEEE Transactions on Evolutionary Computation, vol. 4, no. 3, pp284-294, 2000.

[5] Tessema Biruk, Gary G. Yen, "A Self-adaptive Penalty Function based Algorithm for Constrained Optimization," In 2006 IEEE International Conference on Evolutionary Computation, pp246-253, 2006.

[6] Mallipeddi Rammohan, Ponnuthurai N Suganthan, "Ensemble of Constraint Handling Techniques," IEEE Transactions on Evolutionary Computation, vol. 14, no. 4, pp561-579, 2010.

[7] Storn, Rainer, Kenneth Price, "Differential Evolution–a Simple and Efficient Heuristic for Global Optimization over Continuous Spaces," Journal of Global Optimization, vol.11, no.4, pp341-359, 1997.

[8] Brest Janez, Sao Greiner, Borko Boskovic, Marjan Mernik, and Viljem Zumer, "Self-adapting Control Parameters in Differential Evolution: A Comparative Study on Numerical Benchmark Problems," IEEE Transactions on Evolutionary Computation, vol. 10, no. 6, pp646-657, 2006.

[9] Mohamed Ali Wagdy, "An Improved Differential Evolution Algorithm with Triangular Mutation for Global Numerical Optimization," Computers & Industrial Engineering, vol. 85, pp359-375, 2015.

[10] Mohamed Ali Wagdy, Abdulaziz S Almazyad, "Differential Evolution with Novel Mutation and Adaptive Crossover Strategies for Solving Large Scale Global Optimization Problems," Applied Computational Intelligence and Soft Computing, pp1-18, 2017.

[11] Wang Bing-Chuan, Han-Xiong Li, Jia-Peng Li, and Yong Wang, "Composite Differential Evolution for Constrained Evolutionary Optimization," IEEE Transactions on Systems, Man, and Cybernetics: Systems, vol. 49, no. 7, pp1482-1495, 2018.

[12] Zhou Yin-Zhi, Wen-Chao Yi, Liang Gao, and Xin-Yu Li, "Adaptive Differential Evolution with Sorting Crossover Rate for Continuous Optimization Problems," IEEE Transactions on Cybernetics, vol. 47, no. 9, pp2742-2753, 2017.

[13] Gong Wenyin, Álvaro Fialho, and Zhihua Cai, "Adaptive strategy selection in differential evolution," In Proceedings of the 12th annual conference on Genetic and evolutionary computation, pp. 409-416, 2010.

[14] Fan Hui-Yuan, Jouni Lampinen, "A Trigonometric Mutation Operation to Differential Evolution," Journal of Global Optimization, vol. 27, no. 1, pp105-129, 2003.

[15] Takahama Tetsuyuki, Setsuko Sakai, "Constrained optimization by the constrained differential evolution with gradient-based mutation and feasible elites," In 2006 IEEE International Conference on Evolutionary Computation, pp1-8, 2006.