Applied Soft Computing 11 (2011) 1679–1696

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Differential evolution algorithm with ensemble of parameters and mutation strategies

R. Mallipeddi [a], P.N. Suganthan [a,*], Q.K. Pan [b], M.F. Tasgetiren [c]

[a] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore
[b] College of Computer Science, Liaocheng University, Liaocheng 252059, PR China
[c] Department of Industrial Engineering, Yasar University, Bornova, Izmir, Turkey

ARTICLE INFO

Article history:
Received 30 July 2009
Received in revised form 4 March 2010
Accepted 30 April 2010
Available online 10 May 2010

Keywords:
Differential evolution
Global optimization
Parameter adaptation
Ensemble
Mutation strategy adaptation

ABSTRACT

Differential evolution (DE) has attracted much attention recently as an effective approach for solving numerical optimization problems. However, the performance of DE is sensitive to the choice of the mutation strategy and associated control parameters. Thus, to obtain optimal performance, time-consuming parameter tuning is necessary. Different mutation strategies with different parameter settings can be appropriate during different stages of the evolution. In this paper, we propose to employ an ensemble of mutation strategies and control parameters with the DE (EPSDE). In EPSDE, a pool of distinct mutation strategies along with a pool of values for each control parameter coexists throughout the evolution process and competes to produce offspring. The performance of EPSDE is evaluated on a set of bound-constrained problems and is compared with conventional DE and several state-of-the-art parameter adaptive DE variants.

© 2010 Elsevier B.V. All rights reserved.

1. Introduction

Evolutionary algorithms (EAs), inspired by Darwinian theory of evolution, are well known for their ability to deal with non-linear and complex optimization problems. The primary advantage of EAs over other numerical methods is that they just require the objective function values, while properties such as differentiability and continuity are not necessary. However the performance of an EA depends on the encoding schemes, evolutionary operators and parameter settings such as probability of mutation and population size. Choosing suitable parameter values is a problem dependant task and generally requires time-consuming trial-and-error parameter tuning process. This approach is not appropriate if the global optimization is required in an automated environment or if the user has no experience in the fine art of the control parameter tuning. To overcome this, different parameter adaptation schemes have been presented [1–6]. The parameter adaptation techniques can be categorized as [5]: deterministic, adaptive and self-adaptive. Deterministic rules modify the parameters according to certain predefined rationales without utilizing any feedback from the search process. Adaptive rules incorporate feedback from the search process to guide the parameter adaptation. Self-adaptive rules directly encode parameters into the individuals and evolve them together with the encoded solutions. The most appropriate parameter values produce better offspring that are more likely to survive and propagate the better parameter values [7]. Generally speaking, self-adaptive rules refer to those that mainly utilize the feedback from the search process to guide the updating of parameters. In this paper, we propose an ensemble approach, where each parameter has a pool of values competing to produce future offspring based on their success in the past generations. Although the ensemble concept is general and can be applied with any evolutionary algorithm framework, we demonstrate it using the differential evolution (DE) algorithm.

DE proposed by Storn and Price [8] is a fast and simple technique which performs well on a wide variety of problems. DE is a population based stochastic search technique, which is inherently parallel. DE has been successfully applied in diverse fields such as mechanical engineering [9,10], communication [11], optics [12], pattern recognition [13–15], signal processing [16] and power systems [17]. The performance [18] of the DE algorithm is sensitive to the mutation strategy and respective control parameters such as the population size (*NP*), crossover rate (*CR*) and the scale factor (*F*). The best settings for the control parameters can be different for different optimization problems and the same functions with different requirements for consumption time and accuracy. Therefore, to successfully solve a specific optimization problem, it is generally necessary to perform a time-consuming trial-and-error search for the most appropriate strategy and to tune its associated parameter

* Corresponding author.
E-mail addresses: mall0004@ntu.edu.sg (R. Mallipeddi), epnsugan@ntu.edu.sg (P.N. Suganthan).

1568-4946/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2010.04.024

values. However, such a trial-and-error search process suffers from high computational costs. The population of DE may evolve through different regions in the search space, within which different strategies [19] with different parameter settings may be more effective than others. Although different partial adaptation schemes have been proposed [7,19–22] to overcome the time-consuming trial-and-error procedure, we demonstrate the superior performance of the proposed ensemble strategy with adaptation of all parameters and mutation strategies.

The reminder of this paper is organized as follows: Section 2 presents the classical DE algorithm, while Section 3 presents a literature survey on different mutation strategies and parameter settings used in DE. Section 4 presents the proposed ensemble of mutation strategies and parameters in DE (EPSDE) algorithm. Section 5 presents the experimental results and discussions while Section 6 concludes the paper.

2. The DE aLGORITHM

Differential evolution (DE) is a parallel direct search method which utilizes *NP* *D*-dimensional parameter vectors, so-called individuals, which encode the candidate solutions, *i.e.* $X_{i,G} = \{x^1_{i,G}, \ldots, x^D_{i,G}\}$, $i = 1, \ldots, NP$. The initial population should cover the entire search space as much as possible by uniformly randomizing the initial individuals within the search space constrained by the prescribed minimum and maximum parameter bounds $X_{\min} = \{x^1_{\min}, \ldots, x^D_{\min}\}$ and $X_{\max} = \{x^1_{\max}, \ldots, x^D_{\max}\}$. For example, the initial value of the jth parameter of the ith individual at generation $G = 0$ is generated by:

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

where $rand(0, 1)$ represents a uniformly distributed random variable within the range [0, 1].

2.1. Mutation operation

After initialization, DE employs the mutation operation to produce a mutant vector $\mathbf{V}_{i,G}$ with respect to each individual $\mathbf{X}_{i,G}$, so-called target vector, in the current population. For each target vector $\mathbf{X}_{i,G}$ at the generation G, its associated mutant vector $V_{i,G} = \{v^1_{i,G}, v^2_{i,G}, \ldots, v^D_{i,G}\}$ can be generated via certain mutation strategy. The most frequently used mutation strategies are

"DE/best/1" [11]: $$V_{i,G} = X_{best,G} + F \cdot (X_{r^j_1,G} - X_{r^i_2,G}) \tag{2}$$

"DE/best/2" [11]: $$V_{i,G} = X_{best,G} + F \cdot (X_{r_1{}^i,G} - X_{r^i_2,G}) + F.(X_{r_3{}^i,G} - X_{r^i_4,G}) \tag{3}$$

"DE/rand/1" [11]: $$V_{i,G} = X_{r_1{}^i,G} + F \cdot (X_{r_2{}^i,G} - X_{r^i_3,G}) \tag{4}$$

"DE/rand/2" [19]: $$V_{i,G} = X_{r_1{}^i,G} + F \cdot (X_{r_2{}^i,G} - X_{r^i_3,G}) + F \cdot (X_{r^i_4,G} - X_{r^i_5,G}) \tag{5}$$

"DE/rand-to-best/1" [11] or "DE/target-to-best/1" [23]

$$V_{i,G} = X_{i,G} + K \cdot (X_{best,G} - X_{i,G}) + F \cdot (X_{r_1{}^i,G} - X_{r^i_{2,G}}) \tag{6}$$

"DE/rand-to-best/2" [19] or "'DE/target-to-best/2"

$$V_{i,G} = X_{i,G} + K \cdot (X_{best,G} - X_{i,G}) + F \cdot (X_{r^i_1,G} - X_{r^i_2,G} + X_{r^i_3,G} - X_{r^i_4,G}) \tag{7}$$

"DE/current-to-rand/1" [24]

$$U_{i,G} = X_{i,G} + K \cdot (X_{r^i_1,G} - X_{i,G}) + F \cdot (X_{r^i_2,G} - X_{r^i_3,G}) \tag{8}$$

The indices r^i_1, r^i_2, r^i_3, r^i_4, r^i_5 are mutually exclusive integers randomly generated within the range [1, *NP*], which are also different from the index i. These indices are randomly generated once for each mutant vector. The scale factor F is a positive control parameter for scaling the difference vector. $\mathbf{X}_{best,G}$ is the best individual vector with the best fitness value in the population at generation G. K is randomly chosen within the range [0, 1].

2.2. Crossover operation

After the mutation phase, crossover operation is applied to each pair of the target vector $\mathbf{X}_{i,G}$ and its corresponding mutant vector $\mathbf{V}_{i,G}$ to generate a trial vector: $U_{i,G} = \{u^1_{i,G}, u^2_{i,G}, \ldots, u^D_{i,G}\}$. In the basic version, DE employs the binomial (uniform) crossover defined as follows [25]:

$$u^j_{i,G} = \begin{cases} v^j_{i,G} & \text{if } (rand_j[0,1] \le CR) \text{ or } (j = j_{rand}) \\ x^j_{i,G} & \text{otherwise} \end{cases} \quad j = 1, 2, \ldots, D \tag{9}$$

In Eq. (9), the crossover rate CR is a user-specified constant within the range [0, 1], which controls the fraction of parameter values copied from the mutant vector. j_{rand} is a randomly chosen integer in the range [1, D]. The binomial crossover operator copies the jth parameter of the mutant vector $\mathbf{V}_{i,G}$ to the corresponding element in the trial vector $\mathbf{U}_{i,G}$ if $rand_j[0, 1]$ or $j = j_{rand}$. Otherwise, it is copied from the corresponding target vector $\mathbf{X}_{i,G}$. There exists another exponential crossover operator, in which the parameters of trial vector $\mathbf{U}_{i,G}$ are inherited from the corresponding mutant vector $\mathbf{V}_{i,G}$ starting from a randomly chosen parameter index till the first time $rand_j[0, 1]$. The remaining parameters of the trial vector $\mathbf{U}_{i,G}$ are copied from the corresponding target vector $X_{i,G}$. The condition $j = j_{rand}$ is introduced to ensure that the trial vector $\mathbf{U}_{i,G}$ will differ from its corresponding target vector $\mathbf{X}_{i,G}$ by at least one parameter. DE's exponential crossover operator is functionally equivalent to the circular two-point crossover operator [19].

2.3. Selection operation

If the values of some parameters of a newly generated trial vector exceed the corresponding upper and lower bounds, we randomly and uniformly reinitialize them within the pre-specified range. Then the objective function values of all trial vectors are evaluated. After that, a selection operation is performed. The objective function value of each trial vector $f(\mathbf{U}_{i,G})$ is compared to that of its corresponding target vector $f(\mathbf{X}_{i,G})$ in the current population. If the trial vector has less or equal objective function value (in a minimization problem) than the corresponding target vector, the trial vector will replace the target vector and enter the population of the next generation. Otherwise, the target vector will remain in the population for the next generation. The selection operation can be expressed as follows:

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

The above three steps are repeated generation after generation until a termination criterion (reaching the maximum number of function evaluations set) is satisfied. The algorithmic description of the DE is summarized in Table 1.

3. Literature review

Differential evolution (DE) algorithm is a floating-point encoded evolutionary algorithm for global optimization over continuous spaces [25]. Although the DE has attracted much attention recently, the performance of the conventional DE algorithm depends on the chosen mutation strategy and the associated control parameters. The performance of DE becomes more sensitive to the strategy and the associated parameter values when the problem is complex [26]. Inappropriate choice of mutation strategy and parameters may lead to premature convergence, stagnation or wastage of computational resources [21,23,26–28]. Initially it was thought that [25,29] the control parameters of DE are not difficult to choose. But due to the complex interaction of control parameters with the DE's performance on hard optimization problems [7], choosing an appropriate

Table 1
The standard DE algorithm.

Step 1 Set the generation number $G = 0$, and randomly initialize a population of NP individuals $P_G = \{X_{1,G},...,X_{NP,G}\}$ with $X_{i,G} = \{x^1_{i,G},...,x^D_{i,G}\}$ $i = 1,...,NP$ uniformly distributed in the range $[X_{min}, X_{max}]$, where $X_{min} = \{x^1_{min},...,x^D_{min}\}$ and $X_{max} = \{x^1_{max},...,x^D_{max}\}$

Step 2 WHILE stopping criterion is not satisfied DO

Step 2.1 ***Mutation step***

/*Generate a mutated vector $V_{i,G} = \{v^1_{i,G}, v^2_{i,G},..., v^D_{i,G}\}$ for each target vector $\mathbf{X}_{i,G}$ */

FOR $i = 1$ to NP

Generate a mutated vector $V_{i,G} = \{v^1_{i,G}, v^2_{i,G},..., v^D_{i,G}\}$ corresponding to the target vector $\mathbf{X}_{i,G}$ via one of the equations (2)-(6).

END FOR

Step 2.2 ***Crossover step***

/*Generate a trial vector $U_{i,G} = \{u^1_{i,G}, u^2_{i,G},..., u^D_{i,G}\}$ for each target vector $\mathbf{X}_{i,G}$ */

/*Binomial crossover*/

FOR $i = 1$ to NP

$j_{rand} = [rand[0,1].D]$

FOR $j = 1$ to D

$$u^j_{i,G} = \begin{cases} v^j_{i,G} & \text{if } (rand_j[0,1] \le CR) \text{ or } (j = j_{rand}) \\ x^j_{i,G} & \text{otherwise} \end{cases} \quad j = 1, \ 2, \ \dots, D$$

END FOR

END FOR

Step 2.3 ***Selection step***

/*Selection*/

FOR $i = 1$ to NP

/* Evaluate the trial vector $U_{i,G}$ */

IF $f(U_{i,G}) \le f(X_{i,G})$, THEN $X_{i,G+1} = U_{i,G}$, $f(X_{i,G+1}) \le f(U_{i,G})$

IF $f(U_{i,G}) < f(X_{best,G})$, THEN $X_{best,G} = U_{i,G}$, $f(X_{best,G}) \le f(U_{i,G})$

/* $X_{best,G}$ is the individual with the best fitness value in generation G */

ELSE $X_{i,G+1} = X_{i,G}$

END IF

END IF

END FOR

Step 2.4 ***Increment the generation count*** $G = G + 1$

Step 3 **END WHILE**

mutation strategy and control parameters require some expertise. Since DE was proposed various empirical guidelines were suggested for choosing a mutation strategy and its associated control parameter settings.

3.1. Population size (NP)

Initially $NP = 10D$ was considered as a good choice [11] for DE to find a global optimum. However, to balance the speed (number

of function evaluations) and reliability (ability to find global optimum) different ranges of NP values such as $5D$ to $10D$ [25], $3D$ to $8D$ [26] and $2D$ to $40D$ [30] were suggested.

The larger the population size, the higher the probability of finding a global optimum for multi-modal problems is. But, a larger population implies a slower convergence rate requiring a larger number of function evaluations. Therefore, separable and uni-modal functions require smaller population sizes to speed up the convergence, while parameter-linked multi-modal functions require larger populations to avoid premature convergence. In [23] it is stated that $NP = 5D \times CR$ is usually a good low-end default setting, while for highly multi-modal, parameter dependant functions, NP may need to be 10D or higher. However, NP should be larger than a critical value depending on the mutation strategy used [26]. For example, NP must be at least 4 for DE/rand/1/bin and 5 for DE/best/2/bin, respectively to ensure that DE will have enough mutually different vectors.

3.2. Mutation strategies

The classical DE proposed by Price and Storn uses DE/rand/1/bin, which is most widely used. Later, different DE mutation strategies were proposed with different mutations strategies [11,31]. In [25,26] it was stated that a 2 difference vector strategies such as DE/rand/2/bin and DE/best/2/bin are better than DE/rand/1/bin and DE/best/1/bin due to their ability to improve diversity by producing more trial vectors [28]. DE/best/1/bin and DE/rand-to-best/1/bin are faster on easier optimization problems, but become unreliable when solving highly multi-modal problems. To balance the exploration and exploitation abilities of DE, DE/target-to-best/1/bin scheme with the concept of neighborhood of each population member was proposed [32]. Even increasing the population size (NP) had a little impact on the convergence probabilities. Classical DE was slower, but more robust than the other strategies that rely on the best-so-far vector. DE/current-to-rand/1/bin being a rotation-invariant strategy can solve rotated problems better than other strategies [24].

3.3. Crossover rate (CR)

The crossover rate controls which and how many components are mutated in each element of the current population [33]. The crossover rate CR is a probability $0 \leq CR \leq 1$ of mixing between trial and target vectors. A large CR speeds up convergence [11,25,26]. In [25], it is said that $CR = 0.1$ is a good initial choice while $CR = 0.9$ or 1.0 can be tried to increase the convergence speed. In [26], a good choice for CR is said to be between 0.3 and 0.9. For separable problems, CR from the range (0, 0.2) is the best while for multi-modal, parameter dependant problems CR in the range (0.9, 1.0) is best [23,30]. Based on these observations, CR with non-continuous ranges are also used [34]. When $CR = 1.0$ is chosen, the number of trial solutions will be reduced dramatically which may lead to stagnation [28,30]. Therefore, $CR = 0.9$ or 0.99 can be used instead.

3.4. Scaling factor (F)

Scaling factor, F is usually chosen in [0.5, 1] [11]. The upper limit of $F = 1.2$ is empirically determined and optimization problems requiring F greater than 1.2 to solve efficiently have not been encountered. In [25], it is said that values of F smaller than 0.4 and greater than 1.0 are occasionally effective. The scale factor F, is strictly greater than zero. A larger F increases the probability of escaping from a local optimum [26,30]. While $F > 1$ can solve many problems the convergence speed decreases making it difficult to converge as the perturbation is larger than the distance between two members. $F \leq 1$ is usually both faster and reliable. F must be above a certain critical value to avoid premature convergence to a sub-optimal solution [26,30], but if F becomes too large, the number of function evaluations to find the optimum grows very quickly. In [25,26], it is said that $F = 0.6$ or 0.5 would be a good initial choice while in [30] it is said that $F = 0.9$ would be a good initial choice. Typical values of F are 0.4–0.95 according to [30].

Zaharie [35] theoretically proved that DE could converge to global optimum in the long time limit if F can be transformed into a Gaussian random variable. Despite the theoretical proof, it [36] was proved that transforming F into a Gaussian random variable did not significantly enhance DE's performance. In addition, Price et al. [23] demonstrated that unless the variance of the randomizing distribution is very small, DE will suffer a significant performance loss on highly conditioned non-separable functions [30]. Similar to CR, $F = 1.0$ is not recommended since it reduces the number of potential trial solutions and may lead to stagnation.

3.5. Overall parameter tuning methods

Even though the above guidelines are useful for choosing the individual parameters of DE to some extent, the performance of DE is more sensitive to the combination of the mutation strategy and its associated parameters. For a mutation strategy [7], a particular value of CR makes the parameter F sensitive while some other values of CR make the same F robust. It was recommended in [31] to use the mutation strategy DE/current-to-rand/1/bin and parameter setting $NP = 20D$, $K = 0.5$, $F = 0.8$. If the DE converges prematurely, one should increase the value of NP or F, or randomly choose K within the range [0, 1]. If none of the above configuration works, one may try the strategy DE/rand/1/bin with a small CR value.

From the above, it can be observed that various conflicting conclusions had been drawn regarding the manual parameter tuning of DE, which lack sufficient justifications. To avoid the tuning of parameters by trial-and-error procedure, various techniques have been developed. The scale factor F can be linearly reduced from a maximum to a minimum value with increasing generation count [27], or randomly varied in the range (0.5, 1) or a uniform distribution between 0.5 and 1.5 (with mean of 1) can be employed [37]. Recently, several researchers [21,38–40] focused on the adaptation of the control parameters F and CR. FADE adapts the control parameters F and CR based on fuzzy logic controllers whose inputs are the relative function values and individuals of successive generations [39]. FADE outperformed the conventional DE on higher dimensional problems. A parameter adaptation of DE (ADE) based on controlling the population diversity, and multi-population approach [21] was proposed, which was later extended to form an adaptive Pareto DE algorithm for multi-objective optimization [40]. Abbas [38] self-adapted the crossover rate of DE for multi-objective optimization problems, by encoding the crossover rater into each individual, to simultaneously evolve with other parameters. The scale factor F was generated using a Gaussian distribution $N(0, 1)$ for each individual.

Qin et al. [19] proposed a self-adaptive DE algorithm (SaDE) in which the mutation strategies and the respective control parameter are self-adapted based on their previous experiences of generating promising solutions. The scale factor, F was randomly generated with a mean and standard deviation of 0.5 and 0.3, respectively. Omran et al. [20] introduced a self-adaptation scheme (SDE) in which CR is generated randomly for each individual using a normal distribution $N(0.5, 0.15)$, while scale factor F is adapted analogous to the adaptation of crossover rate CR in [38]. In addition to control parameters F and CR, Teo proposed differential evolution with self-adapting populations (DESAP) [41], based on Abbas's self-adaptive Pareto DE. Brest et al. [7] proposed a self-adaptation scheme (JDE), in which control parameters F and CR are encoded into the individuals and are adjusted by introducing two new parameters τ_1 and

τ_2. Initially, F and CR values of each individual were assigned to 0.5 and 0.9, respectively. Then, a random number *rand* is uniformly generated in the range of [0, 1]. If $rand < \tau_1$, the F value was reinitialized to a new random value in the range [0.1, 1.0]. Otherwise, it was kept unchanged. Similarly, if $rand < \tau_2$, then CR was reinitialized in the range [0, 1]. Otherwise, it is kept unchanged.

4. Ensemble of mutation strategies and parameters in DE (EPSDE)

The effectiveness of conventional DE in solving a numerical optimization problem depends on the selected mutation strategy and its associated parameter values. However, different optimization problems require different mutation strategies with different parameter values depending on the nature of problem (uni-modal and multi-modal) and available computation resources. In addition, to solve a specific problem, different mutation strategies with different parameter settings may be better during different stages of the evolution than a single mutation strategy with unique parameter settings as in the conventional DE. Motivated by these observations, we propose an ensemble of mutation strategies and parameter values for DE (EPSDE) in which a pool of mutation strategies, along with a pool of values corresponding to each associated parameter competes to produce successful offspring population. The candidate pool of mutation strategies and parameters should be restrictive to avoid the unfavorable influences of less effective mutation strategies and parameters [19]. The mutation strategies or the parameters present in a pool should have diverse characteristics, so that they can exhibit distinct performance characteristics during different stages of the evolution, when dealing with a particular problem. The EPSDE is described in detail below.

4.1. Selection of pool of mutation strategies

The different mutation strategies in DE literature are presented in Section 2.1. The different mutation strategies can be classified as:

1. Strategy without crossover (Eq. (8)).
2. Strategies with crossover.
   2.1. Individuals forming the mutant vector are randomly selected (Eqs. (4) and (5)).
   2.2. Strategies relying on the best solution found so far (Eqs. (2), (3), (6) and (7)).

From the different mutation strategies available, we select a few of them to form a pool with diverse characteristics as follows:

- DE/best/1/bin, DE/best/2/bin, DE/rand-to-best/1/bin and DE/rand-to-best/2/bin rely on the best solution found so far. DE/best/1/bin and DE/best/2/bin are degenerate cases of DE/rand-to-best/1/bin and DE/rand-to-best/2/bin with $K = 1$. However, a strategy with two difference vectors is more robust than a strategy with one difference vector. Therefore, to make the strategy pool diverse, we select DE/best/2/bin out the four mutation strategies which use the best solution so far found.
- DE/rand/1/bin and DE/rand/2/bin are two mutation strategies which bear stronger exploration capabilities. Out of the two strategies we include DE/rand/1/bin into pool, since it is faster, robust and is one of the most widely used mutation strategy in the DE literature.
- DE/current-to-rand/1/bin being a rotation-invariant strategy without crossover is different from the other strategies. Hence, it is included in the pool.

4.2. Selection of parameter ranges

The crucial parameters that affect the performance of DE are the population size (NP), crossover rate (CR) and the scale factor (F). In the proposed EPSDE, the population size ($NP = 50$) is maintained constant throughout the evolution process.

Separable uni-modal problems require low CR values while parameter-linked mulimodal problems require higher CR values. In order to balance the speed and efficiency while solving problems with different characteristics, the pool of CR values is taken in the range 0.1–0.9 in steps of 0.1. Based on the literature presented in Section 3.4, the pool of F values is taken in the range 0.4–0.9 in steps of 0.1.

4.3. Overall implementation

EPSDE consists of a pool of mutation strategies with diverse characteristics and a pool of values for each of the associated control parameters. Each member in the initial population is randomly assigned with a mutation strategy and associated parameter values taken from the respective pools. The population members (target vectors) produce offspring (trial vectors) using the assigned mutation strategy and parameter values. If the generated trial vector produced is better than the target vector, the mutation strategy and parameter values are retained with trial vector which becomes the parent (target vector) in the next generation. The combination of the mutation strategy and the parameter values that produced a better offspring than the parent are stored. If the target vector is better than the trial vector, then the target vector is randomly reinitialized with a new mutation strategy and associated parameter values from the respective pools or from the successful combinations stored with equal probability. This leads to an increased probability of production of offspring by the better combination of mutation strategy and the associated control parameters in the future generations (Table 2).

4.4. Comparison of EPSDE with SaDE, JDE, ADE, JADE and SDE

Algorithm	Comparison
SDE	CR is generated randomly for each individual using normal distribution $N(0.5, 0.15)$, while F is adapted. The mutation strategy is not adapted.
ADE	Parameters F and CR are adapted such the population diversity is preserved in every generation. The lower and upper bounds for CR are 0.1 and 1.0, respectively. The lower and upper bounds for F are $1/\sqrt{NP}$ and 2.0, respectively. There is no adaptation of the mutation strategy.
JDE	The mutation strategy is not adapted. The ranges of F and CR values are [0.1, 1.0] and [0, 1], respectively. The F and CR values are adapted based on two constants τ_1 and τ_2, which are set to 0.1.
SaDE	The F values are randomly generated with a mean and standard deviation of 0.5 and 0.3, respectively. The mutation strategy and the parameter CR are self-adapted based on their previous performance.
JADE	JADE implements a new mutation strategy "DE/current-to-$p$best" with optional external archive. In every generation, the CR value corresponding to each individual is randomly initialized with a normal distribution of mean μ_{CR} and standard deviation 0.1, while the F value is randomly initialized with a Cauchy distribution with a location parameter μ_F and standard deviation 0.1. After every generation, the μ_{CR} is updated based on the arithmetic mean of the successful CR values, while μ_F is updated based on the Lehar mean of the successful F values.
EPSDE	Each member in the initial population is assigned a mutation strategy and parameter values randomly selected from the respective pools. The mutation strategy and parameter values producing better offspring survive while those fail to produce better offspring are reinitialized.

5. Numerical experiments and results

5.1. Test functions

In the following, 14 benchmark problems are listed, among which functions f_1–f_4 are uni-modal and functions f_5–f_{14} are multi-modal. These 14 test functions (f_1–f_{14}) are dimension-wise scalable [19,42].

(1) Shifted sphere function

$$f_1(x) = \sum_{i=1}^{D} z_i^2, \quad \mathbf{z} = \mathbf{x} - \mathbf{o}, \quad \mathbf{o} = [o_1, o_2, \ldots, o_D]:$$

the shifted global optimum

Table 2
Ensemble of mutation strategies and parameters in DE (EPSDE).

Step 1 Set the generation number $G = 0$, and randomly initialize a population of NP individuals $P_G = \{X_{i,G}, \ldots, X_{NP,G}\}$ with $X_{i,G} = \{x_{i,G}^1, \ldots, x_{i,G}^D\}$ $i = 1, \ldots, NP$ uniformly distributed in the range $[X_{min}, X_{max}]$, where $X_{min} = \{x_{min}^1, \ldots, x_{min}^D\}$ and $X_{max} = \{x_{max}^1, \ldots, x_{max}^D\}$

Step 2 Select a pool of mutation strategies and a pool of values for each associated parameters corresponding to each mutation strategy.

Step 3 Each population member is randomly assigned with one of the mutation strategy from the pool and the associated parameter values are chosen randomly from the corresponding pool of values.

Step 4 WHILE stopping criterion is not satisfied
DO

Step 4.1 ***Mutation step***

/*Generate a mutated vector $V_{i,G} = \{v_{i,G}^1, v_{i,G}^2, \ldots, v_{i,G}^D\}$ for each target vector $\mathbf{X}_{i,G}$ */

FOR i = 1 to NP

Generate a mutated vector $V_{i,G} = \{v_{i,G}^1, v_{i,G}^2, \ldots, v_{i,G}^D\}$ corresponding to the target vector $\mathbf{X}_{i,G}$ using the mutation strategy and parameters associated with the target vector.

END FOR

Step 4.2 ***Crossover step***

/*Generate a trial vector $U_{i,G} = \{u_{i,G}^1, u_{i,G}^2, \ldots, u_{i,G}^D\}$ for each target vector $\mathbf{X}_{i,G}$ */

/*Binomial crossover*/

FOR i = 1 to NP

j_{rand}= [$rand$[0,1).D]

FOR j = 1 to D

$$u_{i,G}^j = \begin{cases} v_{i,G}^j & \text{if } (rand_j[0,1] \le CR) \text{ or } (j = j_{rand}) \\ x_{i,G}^j & \text{otherwise} \end{cases} \quad j = 1, \ 2, \ \ldots, D$$

END FOR

END FOR

Step 4.3 ***Selection step***

/* Selection by competition between target (parent) and trial (offspring) vectors */

FOR i =1 to NP

/* Evaluate the trial vector $U_{i,G}$ */

IF $f(U_{i,G}) \le f(X_{i,G})$, THEN $X_{i,G+1} = U_{i,G}$, $f(X_{i,G+1}) \le f(U_{i,G})$

IF $f(U_{i,G}) < f(X_{best,G})$, THEN $X_{best,G} = U_{i,G}$, $f(X_{best,G}) \le f(U_{i,G})$

Table 2 (*Continued*)

/* $X_{best,G}$ is the best individual in generation G */

ELSE $X_{i,G+1} = X_{i,G}$

END IF

END IF

END FOR

Step 4.4 ***Updating Step***

FOR i =1 to NP

IF $f(U_{i,G} >) f(X_{i,G})$ THEN

Randomly select a new mutation strategy and parameter values from the pools or from the stored successful combinations.

END IF

END FOR

Step 4.5 ***Increment the generation count*** $G = G + 1$

Step 5 **END WHILE**

(2) Shifted Schwefel's Problem 1.2

$$f_2(x) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2, \quad \mathbf{z} = \mathbf{x} - \mathbf{o}, \quad \mathbf{o} = [o_1, o_2, \ldots, o_D]: \text{ the shifted global optimum}$$

(3) Rosenbrock's function

$$f_3(x) = \sum_{i=1}^{D-1} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$$

(4) Shifted Schwefel's Problem 1.2 with noise in fitness

$$f_4(x) = \left(\sum_{i=1}^{D} \left(\sum_{j=1}^{i} z_j \right)^2 \right) (1 + 0.4|N(0,1)|), \quad \mathbf{z} = \mathbf{x} - \mathbf{o}, \quad \mathbf{o} = [o_1, o_2, \ldots, o_D]: \text{ the shifted global optimum}$$

(5) Shifted Ackley's function

$$f_5(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} z_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi z_i) \right) + 20 + e, \quad z = x - o,$$

$\mathbf{o} = [o_1, o_2, \ldots, o_D]$: the shifted global optimum

(6) Shifted rotated Ackley's function

$$f_6(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} z_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi z_i) \right) + 20 + e, \quad z = \mathbf{M}(\mathbf{x} - \mathbf{o}),$$

$cond^{b}(\mathbf{M}) = 1$

$\mathbf{o} = [o_1, o_2, \ldots, o_D]$: the shifted global optimum

[b] *Cond*(**M**) means the condition number of rotation matrix **M**.

(7) Shifted Griewank's function

$$f_7(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{z_i}{\sqrt{i}}) + 1, \quad \mathbf{z} = \mathbf{x} - \mathbf{o}, \quad \mathbf{o} = [o_1, o_2, \ldots, o_D]: \text{ the shifted global optimum}$$

(8) Shifted rotated Griewank's function

$$f_8(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{z_i}{\sqrt{i}}\right) + 1, \quad \mathbf{z} = \mathbf{M}(\mathbf{x} - \mathbf{o}), \quad cond(\mathbf{M}) = 3$$

$\mathbf{o} = [o_1, o_2, \ldots o_D]$: the shifted global optimum

(9) Shifted Rastrigin's function

$$f_9(x) = \sum_{i=1}^{D} (z_i^2 - 10\cos(2\pi z_i) + 10), \quad \mathbf{z} = \mathbf{x} - \mathbf{o}, \quad \mathbf{o} = [o_1, o_2, \ldots, o_D]: \text{ the shifted global optimum}$$

(10) Shifted rotated Rastrigin's function

$$f_{10}(x) = \sum_{i=1}^{D} (z_i^2 - 10\cos(2\pi z_i) + 10), \quad \mathbf{z} = \mathbf{M}(\mathbf{x} - \mathbf{o}), \quad cond(\mathbf{M}) = 2$$

$\mathbf{o} = [o_1, o_2, \ldots, o_D]$: the shifted global optimum

(11) Shifted non-continuous Rastrigin's function

$$f_{11}(x) = \sum_{i=1}^{D} (y_i^2 - 10\cos(2\pi y_i) + 10), \quad y_i = \begin{cases} z_i & |z_i| < 1/2 \\ round(2z_i)/2 & |z_i| \geq 1/2 \end{cases} \quad \text{for } i = 1, 2, \ldots, D,$$

$\mathbf{z} = \mathbf{x} - \mathbf{o}$, $\mathbf{o} = [o_1, o_2, \ldots, o_D]$: the shifted global optimum

(12) Schwefel's function

$$f_{12}(x) = 418.9829 \times D - \sum_{i=1}^{D} x_i \sin(|x_i|^{1/2})$$

(13) Composition function 1 (CF1) in [42]

The function f_{13} (CF1) is composed by using 10 sphere functions. The global optimum is easy to find once the global basin is found. The details of constructing such functions are presented in [42].

(14) Composition function 6 (CF6) in [42]

The function f_{14} (CF6) is composed by using 10 different benchmark functions, *i.e.* 2 rotated Rastrigin's functions, 2 rotated Weierstrass functions, 2 rotated Griewank's functions, 2 rotated Ackley's functions and 2 rotated Sphere functions (Table 3).

5.2. Algorithms for comparison

Experiment 1. First, we form an ensemble with a single mutation strategy (DE/rand/1/bin), which is referred to as ensemble of parameters in DE (EPDE). The performance of EPDE is compared with classical DE with different fixed parameter settings.

Experiment 2. EPDE is formed with different mutation strategies DE/best/2/bin (EPDE1), DE/rand/1/bin (EPDE2) and DE/current-to-rand/1/bin (EPDE3) and the performance is compared with EPSDE.

Experiment 3. Finally, we compare the performance of EPSDE with various state-of-the-art self-adaptive methods such as SaDE [19], JDE [7], ADE [21], JADE [43] and SDE [20].

In all the above experiments, the maximum number of function evaluations is set to 100,000 for 10D problems and 300,000 for 30D problems. All experiments were run 30 times, independently on each problem. The performance of different algorithms is compared with ensemble by statistical t-test with a significance level of 0.05. Numerical values $-1, 0, 1$ represent that the ensemble is inferior to, equal to and superior to the algorithm with which it is compared, respectively.

5.3. Experimental results and discussions

Experiment 1. The results (mean, standard deviation values and t-test results) of experiment 1 are presented in Tables 4 and 5 for 10D and 30D problems, respectively. The t-test comparison is between the classical DE with different parameter settings and EPDE.

In Tables 4 and 5, to compare the performance, the statistically significant result is highlighted. To compare the performance of classical DE with different parameter settings the significantly best results obtained by classical DE are underlined. From the underlined results it can be observed that the significantly best results are well scattered. Thus no single parameter setting is apt for all the problems.

For 10D problems from the t-test results present in Table 4, it can be observed that EPDE is inferior to, equal to and superior to classical DE with $F = 0.9$ and $CR = 0.1$ in 1, 8 and 5 problems, respectively. EPDE is inferior to, equal to and superior to classical DE with $F = 0.9$ and $CR = 0.9$ in 1, 0 and 13 problems, respectively. EPDE is inferior to, equal to and superior to classical DE with $F = 0.5$ and $CR = 0.3$ in 0, 8 and 6 problems, respectively.

Table 3
Global optimum, search ranges and initialization ranges of the test functions.

f	Dimension	Global optimum x^*	$f(x^*)$	Search range	Initialization range
f_1	10 and 30	$\mathbf{o}$	0	$[-100, 100]^D$	$[-100, 100]^D$
f_2		$\mathbf{o}$	0	$[-100, 100]^D$	$[-100, 100]^D$
f_3		$(1, 1, \ldots, 1)$	0	$[-100, 100]^D$	$[-100, 100]^D$
f_4		$\mathbf{o}$	0	$[-100, 100]^D$	$[-100, 100]^D$
f_5		$\mathbf{o}$	0	$[-32, 32]^D$	$[-32, 32]^D$
f_6		$\mathbf{o}$	0	$[-32, 32]^D$	$[-32, 32]^D$
f_7		$\mathbf{o}$	0	$\Re$	$[0, 600]^D$
f_8		$\mathbf{o}$	0	$\Re$	$[0, 600]^D$
f_9		$\mathbf{o}$	0	$[-5, 5]^D$	$[-5, 5]^D$
f_{10}		$\mathbf{o}$	0	$[-5, 5]^D$	$[-5, 5]^D$
f_{11}		$(420.96, \ldots, 420.96)$	0	$[-500, 500]^D$	$[-5, 5]^D$
f_{12}		$(420.96, \ldots, 420.96)$	0	$[-500, 500]^D$	$[-500, 500]^D$
f_{13}		$\mathbf{o_1}$	0	$[-5, 5]^D$	$[-5, 5]^D$
f_{14}		$\mathbf{o_1}$	0	$[-5, 5]^D$	$[-5, 5]^D$

$\mathbf{o}$ is the shifted vector. $\mathbf{o_1}$ is the shifted vector for the first basic function in the composition function.

For 30D problems from the *t*-test results present in Table 5, it can be observed that EPDE is inferior to, equal to and superior to classical DE with $F = 0.9$ and $CR = 0.1$ in 1, 7 and 6 problems, respectively. EPDE is inferior to, equal to and superior to classical DE with $F = 0.9$ and $CR = 0.9$ in 0, 0 and 14 problems, respectively. EPDE is inferior to, equal to and superior to classical DE with $F = 0.5$ and $CR = 0.3$ in 1, 7 and 6 problems, respectively.

Experiment 2. The results (mean, standard deviation values and *t*-test results) of experiment 2 are presented in Tables 6 and 7 for 10D and 30D problems, respectively.

In Tables 6 and 7, to compare the performance, the statistically significant result is highlighted. To compare the performance of EPDE with different mutation strategies the significantly best results obtained are underlined. From the underlined results it can be observed that the underlined results are well scattered and thus no single mutation strategy is apt for all the problems.

For 10D problems from the *t*-test results present in Table 6, it can be observed that EPSDE is inferior to, equal to and superior to EPDE1 in 0, 7 and 7 problems, respectively. EPSDE is inferior to, equal to and superior to EPDE2 in 0, 10 and 14 problems, respectively. EPSDE is inferior to, equal to and superior to EPDE3 in 0, 5 and 9 problems, respectively.

For 30D problems from the *t*-test results present in Table 7, it can be observed that EPSDE is inferior to, equal to and superior to EPDE1 in 0, 3 and 11 problems, respectively. EPSDE is inferior to, equal to and superior to EPDE2 in 0, 9 and 5 problems, respectively. EPSDE is inferior to, equal to and superior to EPDE3 in 0, 0 and 14 problems, respectively.

Experiment 3. The results (mean, standard deviation values and *t*-test results) of experiment 3 are presented in Tables 8–10 for 10D, 30D and 50D problems, respectively.

In Tables 8–10, to compare the performance of different algorithms the significantly best results obtained for a problem are highlighted. Out of the five algorithms used for comparison, if ESPDE is the statistically better than all the algorithms, the second best result is underlined to enable better comparison. From the results, it can be observed that EPSDE can perform better than other algorithms in most of the cases. Among the algorithms used for comparison, SDE is weaker in performance. ADE performs better on multi-modal problems f_5–f_{14} than on uni-modal problems f_1–f_4. JDE, JADE and SaDE perform equally well on both uni-modal and multi-modal problems, but SaDE is slightly better on some of the problems. The clear difference in performance can be observed as the dimensionality of the problems increase from 10D to 50D.

For 10D problems from the *t*-test results present in Table 8, it can be observed that EPSDE is inferior to, equal to and superior to SaDE in 1, 12 and 1 problems, respectively. EPSDE is inferior to, equal to and superior to JDE in 1, 9 and 4 problems, respectively. EPSDE is inferior to, equal to and superior to ADE in 0, 9 and 5 problems, respectively. EPSDE is inferior to, equal to and superior to SDE in 0, 6 and 8 problems, respectively. EPSDE is inferior to, equal to and superior to JADE in 1, 9 and 7 problems, respectively.

For 30D problems from the *t*-test results present in Table 9, it can be observed that EPSDE is inferior to, equal to and superior to SaDE in 1, 6 and 7 problems, respectively. EPSDE is inferior to, equal to and superior to JDE in 1, 6 and 7 problems, respectively. EPSDE is inferior to, equal to and superior to ADE in 0, 8 and 6 problems, respectively. EPSDE is inferior to, equal to and superior to SDE in 1, 0 and 13 problems, respectively. EPSDE is inferior to, equal to and superior to JADE in 1, 6 and 7 problems, respectively.

For 50D problems from the *t*-test results present in Table 10, it can be observed that EPSDE is inferior to, equal to and superior to SaDE in 1, 3 and 10 problems, respectively. EPSDE is inferior to, equal to and superior to JDE in 1, 4 and 9 problems, respectively. EPSDE is inferior to, equal to and superior to ADE in 0, 9 and 5 problems, respectively. EPSDE is inferior to, equal to and superior to SDE in 1, 1 and 12 problems, respectively. EPSDE is inferior to, equal to and superior to JADE in 2, 5 and 7 problems, respectively.

5.4. Analysis of EPSDE

When solving an optimization problem using DE, different mutation strategies combined with different parameter settings perform better during different stages of the evolution. In EPSDE, initially each target vector is assigned a mutation strategy and parameter values chosen randomly from the respective pool. As the evolution progresses the inefficient combination of mutation strategies and parameters are replaced with a new mutation strategy and parameters randomly selected from the respective pools. Initially all the mutation strategies and parameter values in respective pools have equal probability of producing an offspring. But the process of replacing the ineffective mutation strategy and associated parameters, leads to an increase in the probability of producing an offspring by the best suited mutation strategy and parameters, which is indicated in the graphs in Figs. 1 and 2.

In Figs. 1 and 2, *X*-axis represents the generation count in the evolution process, while *Y*-axis represents the probability of an offspring being produced by a particular mutation strategy or a particular parameter value. For better representation, the probability values corresponding to a mutation vector or parameter value, represented on the *Y*-axis are averaged over every 50 generations. Figs. 1 and 2 correspond to 10D problems f_3 and f_7, respectively.

Table 4
Comparison between classical DE and EPDE on 10D problems.

Fcn	Rand/1/bin ($F = 0.9$, $CR = 0.1$)		Rand/1/bin ($F = 0.9$, $CR = 0.9$)		Rand/1/bin ($F = 0.5$, $CR = 0.3$)		EPDE	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
F1	**0** 0	**0**	4.95E−13 1	5.27E−13	**0** 0	**0**	**0** –	**0**
F2	8.89E−01 1	4.96E−01	1.44E−05 1	1.13E−05	9.63E−09 1	5.99E−09	**5.85E−19** –	**3.18E−18**
F3	9.01E−01 −1	7.94E−01	**7.11E−03** −1	**2.74E−02**	1.76E+00 0	1.54E+00	1.55E+00 –	1.74E+00
F4	2.41E+01 1	1.28E+01	2.42E−04 1	1.38E−04	5.42E−06 1	4.44E−06	**1.37E−09** –	**6.41E−09**
F5	**0** 0	**0**	4.59E−07 1	2.41E−07	**0** 0	**0**	**0** –	**0**
F6	3.8E1−05 1	1.30E−04	6.86E−07 1	3.89E−07	3.32E−15 1	9.01E−16	**0** –	**0**
F7	**0** 0	**0**	3.05E−01 1	2.02E−01	**0** 0	**0**	**0** –	**0**
F8	**1.22E−01** 0	**2.77E−02**	2.41E−01 1	2.00E−01	**1.60E−01** 0	**3.75E−02**	**1.57E−01** –	**5.76E−02**
F9	**0** 0	**0**	8.71E+00 1	5.53E+00	**0** 0	**0**	**0** –	**0**
F10	**1.33**E+**01** 0	**3.00**E+**00**	1.63E+01 1	1.10E+01	1.65E+01 1	2.99E+00	**1.22**E+**01** –	**3.69**E+**00**
F11	**0** 0	**0**	8.20E+00 1	3.37E+00	**0** 0	**0**	**0** –	**0**
F12	**0** 0	**0**	2.82E+00 1	1.41E+01	**0** 0	**0**	**0** –	**0**
F13	3.41E−02 1	5.98E−02	4.27E−13 1	3.74E−13	6.50E−01 1	3.56E+00	**0** –	**0**
F14	6.62E+00 1	1.33E+00	2.17E−01 1	5.64E−01	9.48E+00 1	2.49E+01	**0** –	**0**

Table 5
Comparison between classical DE and EPDE on 30D problems.

Fcn	Rand/1/bin ($F=0.9$, $CR=0.1$)		Rand/1/bin ($F=0.9$, $CR=0.9$)		Rand/1/bin ($F=0.5$, $CR=0.3$)		EPDE	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
F1	**0** 0	**0**	4.50E−02 1	6.15E−02	**0** 0	**0**	**0** –	**0**
F2	3.62E+03 1	8.22E+03	1.73E+03 1	1.40E+03	1.38E+03 1	2.53E+02	**6.51E+01** –	**2.47E+02**
F3	**3.39E+01** −1	**1.51E+01**	1.04E+02 1	6.25E+01	**2.14E+01** −1	**1.98E+00**	4.54E+01 –	2.73E+01
F4	1.24E+04 1	2.15E+03	8.98E+03 1	5.74E+03	5.19E+03 1	1.24E+03	**4.22E+03** –	**6.89E+03**
F5	**0** 0	**0**	3.86E−02 1	2.18E−02	4.03E−15 0	1.23E−15	6.99E−15 –	6.49E−16
F6	3.81E−05 1	1.3E−04	7.64E−02 1	5.11E−02	**3.67E−15** 0	**6.49E−16**	**6.76E−15** –	**1.08E−15**
F7	**0** 0	**0**	1.52E−01 1	1.15E−01	**0** 0	**0**	**0** –	**0**
F8	9.12E−02 1	3.08E−02	9.01E−01 1	1.40E−01	**2.24E−05** 0	**1.19E−04**	**2.47E−04** –	**1.40E−03**
F9	**0** 0	**0**	8.54E+01 1	3.30E+01	3.10E+01 1	3.24E+00	**0** –	**0**
F10	**1.68E+02** 0	**1.43E+01**	2.45E+02 1	2.20E+01	**1.87E+02** 0	**1.09E+01**	**1.39E+02** –	**1.78E+01**
F11	**0** 0	**0**	6.93E+01 1	2.37E+01	2.88E+01 1	1.94E+00	**0** –	**0**
F12	**0** 0	**0**	4.98E+03 1	1.75E+03	**0** 0	**0**	**0** –	**0**
F13	3.78E−03 1	1.15E−02	5.06E−03 1	5.54E−03	4.49E−06 1	4.62E−06	**0** –	**0**
F14	1.33E+01 1	3.08E+00	1.56E+01 1	5.46E+00	1.15E+01 1	1.54E+00	**3.46E+00** –	**1.63E+00**

Table 6
Comparison of EPDE with different mutation strategies with EPSDE on 10D problems.

Fcn	EPDE1		EPDE2		EPDE3		EPSDE	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
F1	**0** 0	**0**	**0** 0	**0**	**0** 0	**0**	**0** –	**0**
F2	4.71E−29 0	1.61E−28	5.85E−19 0	3.18E−18	8.42E−31 0	4.61E−30	**0** –	**0**
F3	5.32E−01 1	1.38E+00	1.55E+00 1	1.74E+00	5.84E+00 1	1.23E+00	**7.13E−10** –	**3.90E−09**
F4	6.66E−27 0	3.64E−26	1.37E−09 1	6.41E−09	2.82E−24 0	1.25E−23	**0** –	**0**
F5	**0** 0	**0**	**0** 0	**0**	**0** 0	**0**	**0** –	**0**
F6	**0** 0	**0**	**0** 0	**0**	**0** 0	**0**	**0** –	**0**
F7	3.90E−03 1	8.70E−03	**0** 0	**0**	2.25E−01 1	1.21E−01	**0** –	**0**
F8	2.43E−01 1	8.03E−02	1.57E−01 1	5.76E−02	3.83E−01 1	9.83E−02	**8.91E−02** –	**4.27E−02**
F9	**0** 0	**0**	**0** 0	**0**	2.30E−01 1	3.58E+00	**0** –	**0**
F10	1.18E+01 1	2.86E+00	1.22E+01 1	3.69E+00	2.40E+01 1	3.42E+00	**7.33E+00** –	**1.39E+00**
F11	**0** 0	**0**	**0** 0	**0**	1.19E+01 1	2.85E+00	**0** –	**0**
F12	3.95E+00 1	2.16E+01	**0** 0	**0**	1.69E+03 1	2.13E+02	**0** –	**0**
F13	3.33E+01 1	6.61E+01	**0** 0	**0**	1.67E+01 1	3.59E+01	**0** –	**0**
F14	1.67E+01 1	3.80E+01	**0** 0	**0**	6.32E+03 1	3.30E+02	**0** –	**0**

Table 7
Comparison of EPDE with different mutation strategies with EPSDE on 30D problems.

Fcn	EPDE1		EPDE2		EPDE3		EPSDE	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
F1	8.41E−30 0	3.77E−29	**0** 0	**0**	2.58E−01 1	1.39E+00	**0** –	**0**
F2	1.14E+02 1	6.28E+02	6.51E+01 1	2.47E+02	8.38E−06 1	4.57E−05	**2.78E−12** –	**1.26E−11**
F3	9.47E+00 1	1.98E+01	4.54E+01 1	2.73E+01	2.00E+02 1	5.49E+02	**2.69E−22** –	**9.92E−22**
F4	5.62E+02 1	2.39E+03	4.22E+03 1	6.89E+03	4.92E−02 1	2.13E−01	**1.32E−05** –	**5.41E−05**
F5	7.43E−12 1	4.06E−11	6.99E−15 0	6.49E−16	4.62E−02 1	1.47E−01	**3.55E−15** –	**0.00E+00**
F6	**6.99E−15** 0	**6.49E−16**	**6.76E−15** 0	**1.08E−15**	1.28E−03 1	4.07E−03	**3.55E−15** –	**0.00E+00**
F7	4.11E−04 1	2.20E−03	**0** 0	**0**	3.73E−01 1	1.69E+00	**0** –	**0**
F8	3.00E−03 1	5.40E−03	2.47E−04 1	1.40E−03	5.07E−01 1	1.45E+00	**3.97E−17** –	**8.51E−17**
F9	5.60E−01 1	3.07E+00	**0** 0	**0**	1.62E+02 1	8.31E+00	**0** –	**0**
F10	1.29E+02 1	2.07E+02	1.39E+02 1	1.78E+01	1.65E+02 1	1.04E+01	**7.81E+01** –	**1.33E+01**
F11	**0** 0	**0**	**0** 0	**0**	1.32E+02 1	1.03E+01	**0** –	**0**
F12	1.22E−13 1	4.62E−13	**0** 0	**0**	8.40E+03 1	2.73E+02	**0** –	**0**
F13	3.33E+00 1	1.80E+01	**0** 0	**0**	7.62E+03 1	4.71E+02	**0** –	**0**
F14	7.63E+00 1	4.32E+00	**3.46E+00** 0	**1.63E+00**	8.35E+03 1	6.25E+02	**4.94E+00** –	**7.19E−01**

Table 8
Comparison between EPSDE and various state-of-the-art methods on 10D problems.

Fcn	SaDE		jDE		ADE		SDE		JADE		EPSDE	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
f_1	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** –	**0** –
f_2	**0** 0	**0** 0	**0** 0	**0** 0	7.50E−03 1	6.60E−03 1	**0** 0	**0** 0	**0** 0	**0** 0	**0** –	**0** –
f_3	**0** 0	**0** 0	2.66E−01 1	1.01E+00 1	7.59E−01 1	7.38E−01 1	2.21E+00 1	1.77E+00 1	5.30E−01 1	1.40E+00 1	7.13E−10 –	3.90E−09 –
f_4	**0** 0	**0** 0	**0** 0	**0** 0	1.02E+00 1	5.93E−01 1	1.83E−09 1	1.00E−08 1	**0** 0	**0** 0	**0** –	**0** –
f_5	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** –	**0** –
f_6	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	2.04E+01 1	8.48E−01 1	**0** –	**0** –
f_7	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	4.20E−03 1	7.90E−03 1	9.68E−12 0	1.77E−11 0	**0** –	**0** –
f_8	**1.37E−02** 0	**1.18E−02** 0	**2.26E−02** 0	**1.77E−02** 0	**7.93E−02** 0	**4.24E−02** 0	**3.81E−02** 0	**3.06E−02** 0	**2.19E−02** 0	**8.70E−3** 0	**8.91E−02** –	**4.27E−02** –
f_9	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	1.33E−02 1	1.26E−02 1	**0** 0	**0** 0	**0** –	**0** –
f_{10}	**2.82E+00** −1	**1.28E+00** −1	4.41E+00 −1	1.14E+00 −1	5.46E+00 0	1.37E+00 0	3.98E+00 0	2.06E+00 0	4.22E+00 −1	1.30E+00 −1	7.33E+00 –	1.39E+00 –
f_{11}	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	1.40E+00 1	1.19E+00 1	**0** 0	**0** 0	**0** –	**0** –
f_{12}	**0** 0	**0** 0	1.18E+01 1	3.61E+01 1	**0** 0	**0** 0	7.90E+00 1	3.01E+01 1	**0** 0	**0** 0	**0** –	**0** –
f_{13}	**0** 0	**0** 0	6.67E+00 1	2.54E+01 1	1.20E−03 1	3.00E−03 1	1.67E+01 1	3.79E+01 1	2.33E+01 1	4.30E+01 1	**0** –	**0** –
f_{14}	2.54E−01 1	5.21E−01 1	1.27E+00 1	3.20E+00 1	5.87E+00 1	4.85E+00 1	8.26E+00 1	1.82E+01 1	3.33E+00 1	1.83E+01 1	**0** –	**0** –

Table 9
Comparison between EPSDE and various state-of-the-art methods on 30D problems.

Fcn	SaDE		jDE		ADE		SDE		JADE		EPSDE	
	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD	MEAN	STD
f_1	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	**0** 0	3.14E−01 1	1.62E+00 1	**0** 0	**0** 0	**0** –	**0** –
f_2	**0** 0	**0** 0	**0** 0	**0** 0	9.29E+02 1	2.55E+02 1	3.77E−01 1	7.09E−01 1	**0** 0	**0** 0	2.78E−12 –	1.26E−11 –
f_3	1.33E−01 1	7.28E−01 1	5.98E−01 1	1.44E+00 1	3.18E+01 1	1.57E+01 1	5.31E+02 1	1.59E+03 1	4.02E+00 1	1.60E+01 1	**2.69E−22** –	**9.9E−22** –
f_4	1.01E+01 1	3.01E+01 1	2.24E+00 1	1.01E+01 1	7.51E+03 1	1.54E+03 1	3.33E+02 1	3.22E+02 1	3.60E−03 1	6.70E−03 1	**1.32E−05** –	**5.41E−05** –
f_5	**0** 0	**0** 0	**0** 0	**0** 0	4.03E−15 0	1.23E−15 0	1.67E−01 1	3.37E−01 1	5.32E−15 0	1.81E−15 0	**0** –	**0** –
f_6	**0** 0	**0** 0	**0** 0	**0** 0	3.91E−15 0	1.08E−15 0	2.11E−01 1	5.16E−01 1	1.65E−01 1	5.21E−01 1	**0** –	**0** –
f_7	4.10E−03 1	8.80E−03 1	**0** 0	**0** 0	**0** 0	**0** 0	1.54E+00 1	4.58E+00 1	**0** 0	**0** 0	**0** –	**0** –
f_8	2.20E−03 1	4.70E−03 1	**0** 0	**0** 0	1.36E−07 1	6.27E−07 1	1.08E+00 1	1.87E+00 1	1.49E−02 1	1.34E−02 1	**0** –	**0** –
f_9	1.33E−01 1	4.32E−01 1	3.32E−02 1	1.82E−01 1	**0** 0	**0** 0	1.00E+01 1	3.49E+00 1	**0** 0	**0** 0	**0** –	**0** –
f_{10}	**2.38E+00** −1	**4.97E+00** −1	2.36E+01 −1	3.23E+00 −1	4.22E+01 0	4.10E+00 0	2.32E+01 −1	6.98E+00 −1	2.57E+01 −1	4.47E+00 −1	7.81E+01 –	1.33E+01 –
f_{11}	2.00E−01 1	4.07E−01 1	3.33E−02 1	1.83E−01 1	**0** 0	**0** 0	1.46E+01 1	4.23E+00 1	**0** 0	**0** 0	**0** –	**0** –
f_{12}	**0** 0	**0** 0	3.55E+01 1	6.34E+01 1	**0** 0	**0** 0	3.76E+02 1	1.89E+02 1	7.90E+00 1	3.00E+01 1	**0** –	**0** –
f_{13}	**0** 0	**0** 0	1.00E+01 1	4.03E+01 1	1.67E−03 1	9.13E−03 1	6.76E+00 1	3.65E+01 1	1.33E+01 1	5.07E+01 1	**0** –	**0** –
f_{14}	3.51E+01 1	9.38E−01 1	9.22E+00 1	2.56E+01 1	4.18E+00 1	1.41E+00 1	1.16E+01 1	1.86E+01 1	8.84E+00 1	2.55E+01 1	**4.94E+00** –	**7.19E−01** –

Table 10
Comparison between EPSDE and various state-of-the-art methods on 50D problems.

Fcn	SaDE			jDE			ADE			SDE			JADE			EPSDE		
	MEAN		STD	MEAN		STD	MEAN		STD	MEAN		STD	MEAN		STD	MEAN		STD
f_1	**0**	0	**0**	**2.57E−29**	0	**6.61E−29**	**0**	0	**0**	2.17E+01	1	3.85E+01	**0**	0	**0**	**0**	–	**0**
f_2	1.47E−09	0	5.93E−09	2.11E−04	1	2.72E−04	1.70E+04	1	2.72E+03	3.26E+02	1	3.57E+02	**5.20E−23**	−1	**1.13E−22**	4.47E−09	–	1.75E−08
f_3	1.42E+00	1	2.44E+00	7.25E+00	1	1.77E+01	4.55E+01	1	1.08E+00	5.54E+06	1	1.36E+07	1.20E+00	1	1.93E+00	**0**	–	**0**
f_4	3.05E+03	1	1.98E+03	1.26E+03	1	1.39E+03	4.15E+04	1	4.98E+03	9.46E+03	1	3.38E+03	1.42E+03	1	1.55E+03	**5.47E+02**	–	**9.47E+02**
f_5	6.45E−01	1	7.09E−01	**4.62E−15**	0	**1.63E−15**	7.11E−11	0	8.45E−12	1.17E+00	1	7.35E−01	**7.11E−15**	0	**1.52E−16**	**8.05E−15**	–	**2.46E−15**
f_6	1.05E+00	1	6.58E−01	**4.38E−15**	0	**1.50E−15**	**6.87E−15**	0	**8.86E−16**	9.90E−01	1	7.95E−01	1.08E+00	1	7.78E−01	**7.11E−15**	–	**2.46E−15**
f_7	6.40E−03	1	1.15E−02	2.00E−04	1	1.30E−03	**0**	0	**0**	1.40E+01	1	1.20E+01	5.70E−03	1	1.09E−01	**0**	–	**0**
f_8	5.00E−03	1	1.26E−02	7.00E−04	1	2.90E−03	6.49E−12	0	2.56E−11	1.14E+01	1	1.40E+01	3.20E−03	1	5.30E−03	**0**	–	**0**
f_9	2.09E+00	1	1.34E+00	1.99E−01	1	3.99E−01	**0**	0	**0**	2.39E+01	1	5.54E+00	**0**	0	**0**	**0**	–	**0**
f_{10}	7.33E+01	−1	1.65E+01	**4.09E+01**	−1	**5.98E+00**	1.27E+02	0	9.79E+01	5.94E+01	−1	1.23E+01	7.75E+01	−1	1.96E+01	2.00E+02	–	2.9E+01
f_{11}	1.57E+00	1	1.28E+00	3.33E−01	1	4.71E−01	2.00E−01	1	4.00E−01	3.46E+01	1	5.71E+00	**0**	0	**0**	**0**	–	**0**
f_{12}	3.95E+00	1	2.16E+01	1.10E+02	1	1.36E+02	1.82E−11	0	3.45E−12	1.07E+03	1	3.29E+02	5.92E+01	1	1.15E+02	**0**	–	**0**
f_{13}	1.67E+01	1	4.61E+01	2.33E+01	1	4.96E+01	1.19E−08	1	6.52E−08	2.24E+01	1	5.44E+01	1.00E+01	1	3.16E+01	**0**	–	**0**
f_{14}	**5.01E+01**	0	**1.75E+02**	**2.41E+01**	0	**4.02E+01**	**1.26E+01**	0	**2.16E+00**	**2.14E+01**	0	**2.55E+01**	**1.52E+01**	0	**3.11E+01**	**3.15E+01**	–	**4.13E+01**

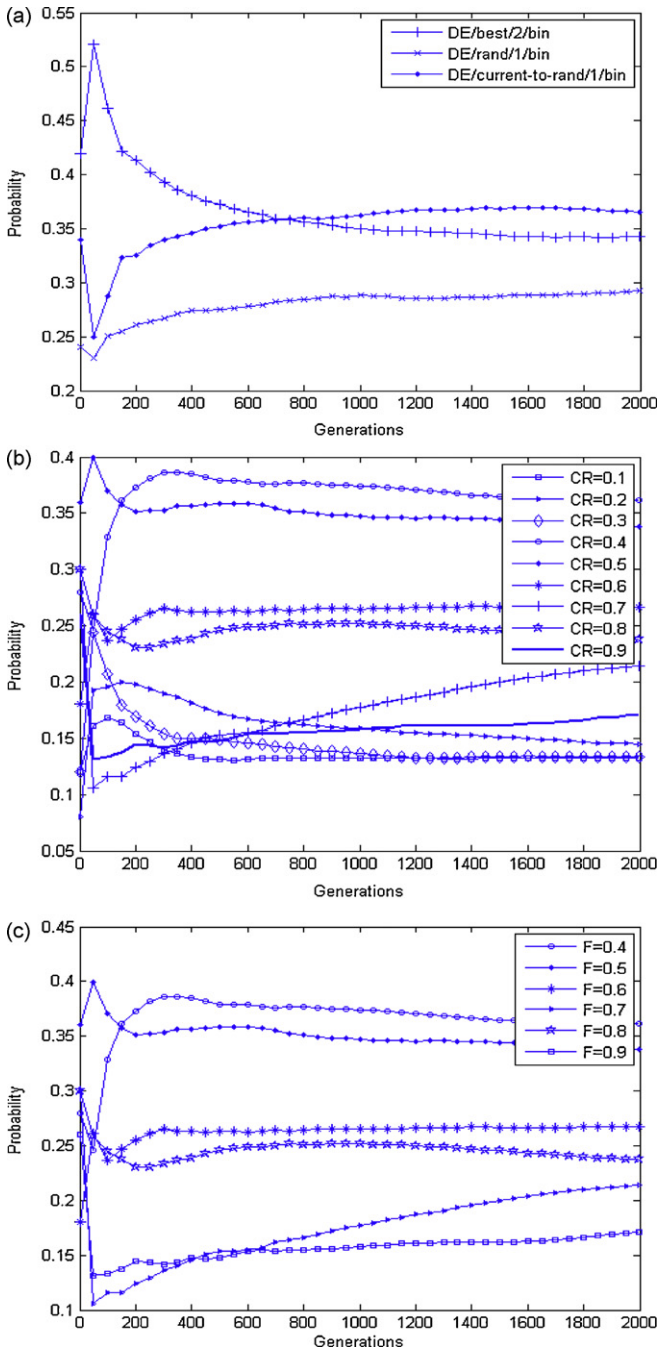

Fig. 1. (a) Probability of an offspring production by a mutation strategies versus generations. (b) Probability of an offspring production by a particular CR value versus generations. (c) Probability of an offspring production by a particular *F* value versus Generations.

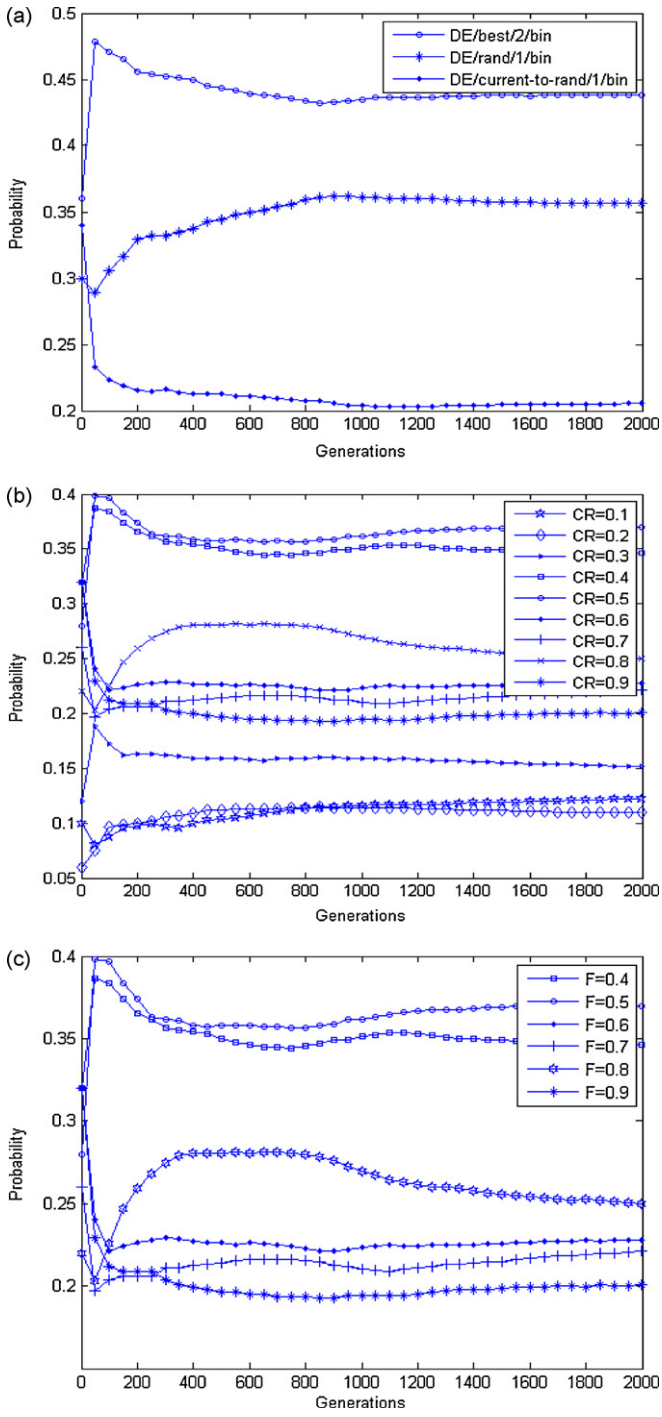

Fig. 2. (a) Probability of an offspring production by a mutation strategies versus generations. (b) Probability of an offspring production by a particular CR value versus generations. (c) Probability of an offspring production by a particular *F* value versus generations.

6. Conclusions

The performance of DE depends on the selected mutation strategy and its associated parameter values. Different optimization problems require different mutation strategies with different parameter values depending on the nature of problem and available computation resources. For a problem at hand different mutation strategies with different parameter settings may be more effective during different stages of the evolution than a single mutation strategy with unique parameter settings as in the conventional DE. Based on these observations, we propose an ensemble of mutation strategies and parameter values in which a pool of mutation strategies, along with a pool of values corresponding to each associated parameter compete to produce offspring population. The performance of EPSDE is evaluated on set of benchmark problems and is favorably compared with the state-of-the-art DE methods in the literature.

Acknowledgment

Authors acknowledge the financial support offered by the A*Star (Agency for Science, Technology and Research, Singapore) under the grant #052 101 0020.

References

[1] P.J. Angeline, Adaptive and self-adaptive evolutionary computation, in: M. Palaniswami, Y. Attikiouzel, R.J. Marks, D.B. Fogel, T. Fukuda (Eds.), Computational Intelligence: A Dynamic System Perspective, IEEE Press, 1995, pp. 152–161.
[2] A.E. Eiben, R. Hinterding, Z. Michalewicz, Parameter control in evolutionary algorithms, IEEE Transactions on Evolutionary Computation 3 (1999) 124–141.
[3] J. Gomez, D. Dasgupta, F. Gonazalez, Using adaptive operators in genetic search, in: Proceedings of the Genetic and Evolutionary Computation Conference 2003 (GECCO'03), Chicago, Illinois, USA, 2003, pp. 1580–1581.
[4] B.R. Julstrom, What have you done for me lately? Adapting operator probabilities in a steady-state genetic algorithm, in: Proceedings of the 6th International Conference on Genetic Algorithms, Pittsburgh, PA, USA, 1995, pp. 81–87.
[5] J.E. Smith, T.C. Fogarty, Operator and parameter adaptation in genetic algorithms, Soft Computing 1 (June) (1997) 81–87.
[6] A. Tuson, P. Ross, Adapting operator settings in genetic algorithms, Evolutionary Computation 6 (1998) 161–184.
[7] J. Brest, S. Greiner, B. Boscovic, M. Mernik, V. Zumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, IEEE Transactions on Evolutionary Computation 10 (December) (2006) 646–657.
[8] R. Storn, K. Price, Differential Evolution – A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces, Technical Report TR-95-012, ICSI, http://http.icsi.berkeley.edu/~storn/litera.html, 1995.
[9] R. Joshi, A.C. Sanderson, Minimal representation multisensor fusion using differential evolution, IEEE Transactions on Systems, Man, and Cybernetics Part A: Systems and Humans 29 (1999) 63–76.
[10] T. Rogalsky, R.W. Derksen, S. Kocabiyik, Differential evolution in aerodynamic optimization, in: Proceeding of 46th Annual Conference of Canadian Aeronautics and Space Institute, Montreal, Quebec, 1999, pp. 29–36.
[11] R. Storn, On the usage of differential evolution for function optimization, in: Biennial Conference of the North American Fuzzy Information Processing Society (NAFIPS), IEEE, Berkeley, 1996, pp. 519–523.
[12] M.K. Venu, R. Mallipeddi, P.N. Suganthan, Fiber Bragg grating sensor array interrogation using differential evolution, Optoelectronics and Advanced Materials - Rapid Communications 2 (2008) 682–685.
[13] J. Ilonen, J.K. Kamarainen, J. Lampinen, Differential evolution training algorithm for feed-forward neural networks, Neural Processing Letters 17 (2003) 93–105.
[14] S. Das, A. Konar, Automatic image pixel clustering with an improved differential evolution, Applied Soft Computing 9 (January) (2009) 226–236.
[15] U. Maulik, I. Saha, Modified differential evolution based fuzzy clustering for pixel classification in remote sensing imagery, Pattern Recognition 42 (2009) 2135–2149.
[16] R. Storn, Differential evolution design of an IIR-filter, in: IEEE International Conference on Evolutionary Computation, IEEE, 1996, pp. 268–273.
[17] M. Varadarajan, K.S. Swarup, Differential evolution approach for optimal reactive power dispatch, Applied Soft Computing 8 (September) (2008) 1549–1561.
[18] J. Liu, J. Lampinen, On setting the control parameter of the differential evolution method, in: Proceedings of the 8th International Conference on Soft Computing (MENDEL 2002), 2002, pp. 11–18.
[19] A.K. Qin, V.L. Huang, P.N. Suganthan, Differential evolution algorithm with strategy adaptation for global numerical optimization, IEEE Transactions on Evolutionary Computation 13 (April) (2009) 398–417.
[20] M.G.H. Omran, A. Salman, A.P. Engelbrecht, Self-adaptive differential evolution, in: Computational Intelligence and Security, PT 1, Proceedings Lecture Notes in Artificial Intelligence, 2005, pp. 192–199.
[21] D. Zaharie, Control of population diversity and adaptation in differential evolution algorithms, in: Proceedings of the 9th International Conference on Soft Computing, Brno, 2003, pp. 41–46.
[22] J. Tvrdik, Adaptation in differential evolution: a numerical comparison, Applied Soft Computing 9 (June) (2009) 1149–1155.
[23] K.V. Price, R.M. Storn, J.A. Lampinen, Differential evolution: a practical approach to global optimization, in: Natural Computing Series, Springer, Berlin, 2005.
[24] A. Iorio, X. Li, Solving rotated multi-objective optimization problems using differential evolution, in: Australian Conference on Artificial Intelligence, Cairns, Australia, 2004, pp. 861–872.
[25] R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, Journal of Global Optimization 11 (1997) 341–359.
[26] R. Gämperle, S.D. Müller, P. Koumoutsakos, A parameter study for differential evolution, in: Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation, WSEAS Press, Interlaken, Switzerland, 2002, pp. 293–298.
[27] S. Das, A. Konar, U.K. Chakraborty, Two improved differential evolution schemes for faster global search, in: Proceedings of the 2005 Conference on Genetic and Evolutionary Computation, 2005, pp. 991–998.
[28] J. Lampinen, I. Zelinka, On stagnation of the differential evolution algorithm, in: Proceedings of MENDEL 2000, 6th International Mendel Conference on Soft Computing, 2000, pp. 76–83.
[29] R. Storn, K. Price, Differential evolution: a simple evolution strategy for fast optimization, Dr. Dobb's Journal 22 (April) (1997) 18–24.
[30] J. Rönkkönen, S. Kukkonen, K.V. Price, Real-parameter optimization with differential evolution, in: IEEE Congress on Evolutionary Computation, 2005, pp. 506–513.
[31] K.V. Price, An introduction to differential evolution, in: D. Corne, M. Dorgio, F. Glover (Eds.), New Ideas in Optimization, McGraw-Hill, London, UK, 1999, pp. 79–108.
[32] S. Das, A. Abraham, U.K. Chakraborthy, Differential evolution using a neighborhood-based mutation operator, IEEE Transactions on Evolutionary Computation 13 (June) (2009) 526–553.
[33] D. Zaharie, Influence of crossover on the behavior of differential evolution algorithms, Applied Soft Computing 9 (June) (2009) 1126–1138.
[34] E. Mezura-Montes, J. Velazquez-Reyes, C.A. Coello Coello, Modified differential evolution for constrained optimization, in: IEEE Congress on Evolutionary Computation, 2006, pp. 25–32.
[35] D. Zaharie, Critical values for the control parameters of differential evolution, in: Proceedings of MENDEL 2002, Eighth International Conference on Soft Computing, Brno, Czech Republic, 2002, pp. 62–67.
[36] J. Rönkkönen, J. Lampinen, On using normally distributed mutation step length for the differential evolution algorithm, in: Proceedings of MENDEL 2003,Ninth International MENDEL Confernce on Soft Computing, Brno, Czech Republic, 2003, pp. 11–18.
[37] U.K. Chakraborthy, S. Das, A. Konar, Differential evolution with local neighborhood, in: Proceedings of Congress on Evolutionary Computation, Vancouver, BC, Canada, 2006, pp. 2042–2049.
[38] H.A. Abbass, The self-adaptive Pareto differential evolution algorithm, in: Proceedings of the IEEE Congress on Evolutionary Computation, vol. 1, 2002, pp. 831–836.
[39] J. Liu, J. Lampinen, A fuzzy adaptive differential evolution algorithm, Soft Computing 9 (June) (2005) 448–462.
[40] D. Zaharie, D. Petcu, Adaptive Pareto differential evolution and its parallelization, in: Proceedings of 5th International Conference on Parallel Processing and Applied Mathematics, Czestochowa, Poland, 2003, pp. 261–268.
[41] J. Teo, Exploring dynamic self-adaptive populations in differential evolution, Soft Computing 10 (2006) 673–686.
[42] J.J. Liang, P.N. Suganthan, K. Deb, Novel composition test functions for numerical global optimization, in: IEEE Swarm Intelligence Symposium 2005, Pasadena, California, 2005, pp. 68–75.
[43] J.Q. Zhang, A.C. Sanderson, JADE: adaptive differential evolution with optional external archive, IEEE Transactions on Evolutionary Computation 13 (October) (2009) 945–958.