Soft Comput (2018) 22:659–676
https://doi.org/10.1007/s00500-016-2365-x

METHODOLOGIES AND APPLICATION

# Evolutionary programming with a simulated-conformist mutation strategy

Han Huang[1] · Shujin Ye[1] · Zhun Fan[1] · Zhiyong Lin[1] · Liang Lv[1] · Zhifeng Hao[1]

Published online: 12 October 2016

**Abstract** Evolutionary programming has been widely implemented as a continuous optimization algorithm. Prior studies have come to a bottleneck because most of the evolutionary programming algorithms are unable to robustly solve different types of optimization problems. We argue that such a bottleneck results from the existing mutation strategies' making little use of the population information. Inspired by a psychological model which describes how a person optimizes his/her social activities by conformity behavior, this study proposes a variation vector of the mutation to simulate the conformity behavior with behavior-reference, majority-impact, and distinctive-impact factors. These factors, respectively, correspond with three types of population information for each mutated individual: heuristic information, optimal gradient, and population diversity. We use the proposed vector to design an improved evolutionary programming with a simulated-conformist mutation strategy. The results show that the population information produced by the three factors enhance the robustness of the performance of evolutionary programming in solving both uni- and multimodal functions. The finding is verified by empirical analyses of two sets of benchmark functions proposed in 1998 and 2013. The numerical results indicate that the proposed algorithm performs significantly better on average than the existing EPs and some other algorithms with similar strategies.

**Keywords** Continuous optimization · Evolutionary programming · Simulated-conformist mutation

## 1 Introduction

As one of the classical evolutionary algorithms (EAs) in evolutionary computation (Rozenberg et al. 2011), evolutionary programming (EP) has a paralleled position with the genetic algorithm (GA) and evolutionary strategy (ES) and has played a prominent role in the field of continuous optimization. It was first proposed as an approach for the finite-state machine (Fogel 2009), but its application has been expanded to the fields of engineering optimization (Duo et al. 1999; El-Sharkh et al. 2003; Hershkovitz et al. 2011). EP is often used to tackle various problems of continuous optimization and has been applied to electrical technology (Hershkovitz et al. 2011), power systems (Chung et al. 2010; El-Sharkh et al. 2003; Sinha et al. 2003), and other engineering optimization problems (Dong et al. 2009; Tan et al. 2011).

Improving EP to be more robust is one of the most important topics in EP research. Since EP is applied to solve several types of optimization problems, its optimization process is bound to tackle different features of the problems, i.e., the high-dimensional problems, the low-dimensional problems, the unimodal problems, and the multimodal problems. According to the no-free-lunch theorem (Wolpert and Macready 1997), it is difficult for EP or evolutionary algorithms to effectively solve the problems with different features. Therefore, our study of EP robustness improvement is also a challenging topic in evolutionary computation.

In this study, the robustness of EP is strengthened by improving its mutation operator. EP consists of two main operators: tournament selection and adaptive mutation. Tour-

Communicated by V. Loia.

✉ Han Huang
bssthh@163.com; hhan@scut.edu.cn

1 School of Software Engineering, South China University of Technology, Guangzhou 510006, China

nament selection is known to give non-elite individuals a better survival rate, which is different from GAs and other classical EAs. Adaptive mutation is based on stochastic strategies, and is used to generate a new individual from an individual with its renewed variation vector. Adaptive mutation strategy is a popular topic of EP research since it has had great impact on the design of other algorithms such as particle swarm optimization (PSO) (Ratnaweera et al. 2004) and differential evolution (DE) (Brest et al. 2006). Yao et al. (1999) and Lee and Yao (2004) have proposed several EP versions with mutation schemes based on different probability distributions. Later studies (Alam et al. 2011, 2012) of EP improvement are to design novel adaptive mutation operators. Our study also focuses on the improvement of adaptive mutation since it is attractive in EP research.

The population information is useful to the improvement of EP mutation operator. Despite its popular implementation in engineering optimization (Duo et al. 1999; El-Sharkh et al. 2003; Hershkovitz et al. 2011), there is little evidence of the advantages of EP over other meta-heuristic algorithms, such as PSO (Ratnaweera et al. 2004), DE (Brest et al. 2006), and memetic algorithms (Nguyen et al. 2009). Driven by its wide-ranging applications, researchers have sought to improve EP performance through hybrid selection of mutation distribution functions (Mallipeddi et al. 2010) and operators (Alam et al. 2011, 2012). Most EPs (Brest et al. 2006; Nguyen et al. 2009; Ratnaweera et al. 2004) have been based on pure probability distribution mutations, yet their performances were inferior to some heuristic mutation EPs (Alam et al. 2011, 2012) in which the population information is used for evolution.

In this paper, we attempt to improve EP by better utilizing population information. The population information refers to the information produced by the individuals in the process of searching the optimum. Considering an individual of EP population as a person of a group, we refer a psychological model of conformity behavior (Asch 1956) to improve the adaptive mutation for the individual based on useful population information. Inspired by Asch's (1956) argument, three types of population information are used to design EP mutation, including behavior-reference, majority-impact, and distinctive-impact factors. We designed a heuristic mutation strategy using the population information to make the EP solution more robust for both unimodal and multimodal functions. The proposed strategy strictly follows the framework of classical EPs (Brest et al. 2006; Nguyen et al. 2009; Ratnaweera et al. 2004), and thus, the modification focuses on the mutation operator without any further changes to the original method. It requires our improvement of EP to be flexible in its applications given its wide applications in several fields of engineering optimization.

This paper proposes an improved EP with a simulated-conformist mutation strategy (EP-SMS). A brief review of EP algorithm research is provided in Sect. 2. The concept of simulated-conformist mutation strategy, the proposed algorithm, and its analysis are presented in Sect. 3. Experimental results comparing EP-SMS and other EPs are presented in Sect. 4, and Sect. 5 offers our conclusions.

## 2 Evolutionary programming

In this section, we introduce a general framework of the EP algorithm and a brief review of EP research.

### 2.1 A brief introduction to evolutionary programming

EP is commonly used to solve minimization problems. Without loss of generality, we assume that EPs aims to solve a minimization problem in a continuous search space, defined as follows:

**Definition 1** Let $\mathbf{S} \subseteq \mathbf{R}^n$ be a finite subspace of the $n$-dimensional real domain $\mathbf{R}^n$, and let $f : \mathbf{S} \rightarrow \mathbf{R}$ be an $n$-dimensional real function. A **minimization problem**, denoted by the 2-tuple $(\mathbf{S}, f)$, is to find an $n$-dimensional vector, $\mathbf{x}_{min} \in \mathbf{S}$, such that $\forall \mathbf{x} \in \mathbf{S}$, $f(\mathbf{x}_{min}) \leq f(\mathbf{x})$.

The general process of evolutionary programming is presented in Algorithm 1. In Step 1, the generated individuals are the real vectors $\mathbf{v}_i = (\mathbf{x}_i, \boldsymbol{\sigma}_i)$, where $i = 1, 2, \ldots, k$, $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{in})$ is the *decision vector* of $n$ elements, and $\boldsymbol{\sigma}_i = (\sigma_{i1}, \sigma_{i2}, \ldots, \sigma_{in})$ is the *variation vector* of $n$ elements that affect the generation of offspring. $x_{ij}$ is the *decision variable*, and $\sigma_{ij}$ is the *variation variable*, where $j = 1, 2, \ldots, n$. We set the initial iteration number $t = 0$ and the initial $\sigma_{ij} \leq 2$ $(j = 1, 2, \ldots, n)$, as proposed in Yao et al. (1999).

**Algorithm 1:** Evolutionary Programming

```
Input: population size $k$, problem dimension $n$, and population
       $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$
Output: $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$ and their evaluation function value.
Initialization of $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$          // Step 1
Evaluate the population $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$     // Step 2
while $Evaluation\ Number < Maximum$ do
    for $i = 1 \leftarrow k$ do                                 // Step 3
        $(\overline{\mathbf{x}_i}, \overline{\boldsymbol{\sigma}_i}) = Mutation(\mathbf{x}_i, \boldsymbol{\sigma}_i)$        (1)
    end
    Evaluate the offspring                                     // Step 4
    Tournament Selection                                       // Step 5
end
```

The fitness values in Steps 2 and 4 are calculated from the objective function of the target problem. In Step 3, a single offspring $\bar{\mathbf{v}}_i = (\bar{\mathbf{x}}_i, \bar{\boldsymbol{\sigma}}_i)$ is generated for each individual $\mathbf{v}_i$ $(i = 1, 2, \ldots, k)$ by a mutation operator. The mutation is the core of EP, so we propose a novel strategy to improve it in this study.

In Step 5, for each individual in the set of all parents and offspring, $q$ different opponent individuals are uniformly randomly selected for comparison. If the fitness value of the opponent is more (for minimization problem) than the fitness value of the selected individual, the individual obtains a win. The top $k$ individuals with the most wins are selected to be the parents in the next iteration.

Most published EPs follow the framework of Algorithm 1, such as FEP (Yao et al. 1999), LEP (Lee and Yao 2004), RTEP (Alam et al. 2011), CEP (Fogel 1991), Direct EP (Hedar and Fukushima 2006), adaptive EP with reinforcement learning (Zhang and Jing 2008), EP on non-uniform mutation (Zhao et al. 2007), EP using a mixed mutation strategy (Dong et al. 2007), and DGEP (Alam et al. 2012). The difference between these EPs lies mainly within the treatment of Eq. (1) in Step 3.

## 2.2 Related work

Prior EP studies converge on three themes: the probabilistic distribution of mutation, the mutation ensemble, and the capacity of explorations and exploitations (Alam et al. 2011, 2012).

Early EP research focused on using only one probabilistic distribution as a mutation factor. Arguably, the first EP was the one with Gaussian mutation, which has been labeled as classical evolutionary programming (CEP) (Fogel 1991). This work has been intensively analyzed by Fogel (1993, 1992), Bäck and Schwefel (1993), and Schwefel (1993). Subsequently, Yao et al. (1999) proposed a fast evolutionary programming algorithm (FEP) with Cauchy mutation. Computational experiments showed FEP to be superior to CEP for multimodal and dispersed peak functions. Lee and Yao (2004) proposed an evolutionary programming based on Lévy mutation (LEP). Empirical analysis (Lee and Yao 2004) showed that LEP performs better than CEP and FEP in solving multimodal and very dispersed peak function problems (Yao et al. 1999) on average. Later, a directed EP (Hedar and Fukushima 2006) was proposed to improve the performance of FEP and LEP. An EP based on reinforcement learning (RLEP) (Zhang and Jing 2008) was proposed and was shown to outperform EPs with four basic mutation operators proposed in Fogel (1991), Hedar and Fukushima (2006), Lee and Yao (2004), Yao et al. (1999). Another EP was based on a non-uniform mutation, named as NEP (Zhao et al. 2007). NEP is faster and more robust than LEP (Lee and Yao 2004) and FEP (Yao et al. 1999) for most multimodal benchmark functions.

The second strand of research is centered on the mutation ensemble. The no-free-lunch theorem (Wolpert and Macready 1997) argues that none of the single mutation operators are efficient in solving all optimization problems. An ensemble of different mutation operators may be better than single mutation operator. A linear combination of the Gaussian distribution and the Cauchy distribution was indicated to be more efficient than either of the distributions alone (Chellapilla 1998). Mallipeddi et al. (2010) proposed an ensemble approach where each mutation operator has its associated population, which benefits from every function call. Mixed strategy (Dong et al. 2007; Yao and Liu 1998) has also been shown to perform equally well or better than the best performing operator among the Gaussian, Cauchy, Lévy, and single-point mutation operators. Hedar and Fukushima (2006) and Zhang and Jing (2008) concluded that the ensemble performed better than algorithms with single mutation strategies. Alipouri et al. (2012) improved EP by promoting its accuracy and convergence speed. To improve the performance of mutation ensemble, Hong et al. (2014) proposed a step size-based self-adaptive mutation operator for EP. For similar purpose, Anik et al. (2013) embedded a dual mutation strategy to EP.

The capacity of explorations and exploitations is another important topic of EP research. Self-adaptive parameters (Liang et al. 2001) perform well when exploiting high-quality solutions. Alam et al. (2012) proposed a diversity-based selection strategy to improve EP performance by preserving and exploiting genetic diversity (Chen et al. 2009), and this approach was shown to be effective in optimizing multimodal functions. Alam et al. (2011) proposed a novel approach for numeric optimization with a recurring two-stage EP (RTEP) based on mutation. The algorithm (Alam et al. 2011) attempts to maintain a balance between global explorations and local exploitations through its two recurring stages. Its performance is stable in solving different styles of benchmark functions (Yao et al. 1999).

To some extent, EP studies for continuous optimization have come to a bottleneck, despite there being more applications such as economic load dispatch (Sinha et al. 2003) and improvement of fuzzy clustering method (Rajan and Christober 2011). The EP mutations [e.g., FEP (Yao et al. 1999), LEP (Lee and Yao 2004), CEP (Fogel 1991), Direct EP (Hedar and Fukushima 2006), adaptive EP with reinforcement learning (Zhang and Jing 2008), and EP on non-uniform mutation (Zhao et al. 2007)] of single probabilistic distribution are only applicable to the respective optimization functions. They are not robust enough to solve different styles of optimization problems. Mutation ensemble EP (Dong et al. 2007; Mallipeddi et al. 2010; Yao and Liu 1998) performance relies on its different mutation operators. Thus, the ensemble may fail to obtain a high-quality solution when its mutation operators cannot generate a high-quality solution to the objective problem. A proper balance between global explorations and local exploitations (Alam et al. 2011, 2012) is essential to improving EP performance, but the mechanism to attain and maintain the balance is a challenging research problem.

EP can be improved to be more robust for different optimization functions when the features (e.g., the properties of separable or non-separable, unimodal or multimodal, shifted, rotated) of the functions are known. Regrettably, such features are oftentimes unavailable in real optimization. However, the feature of the optimization function can be reflected by the EP population during each iteration. Thus, we propose to improve EP by implementing a heuristic mutation based on the iterating population. Following the mutation, the population will be updated according to a simulated-conformist mutation strategy inspired by the psychological model of conformity behavior.

## 3 Evolutionary programming based on simulated-conformist mutation strategy

This section presents the proposed evolutionary programming based on a simulated-conformist mutation strategy (denoted as EP-SMS). The core of the mutation strategy is a simulated-conformist vector. The concept of such a vector arises from a psychological model of conformity behavior (Asch 1956).

### 3.1 Simulated-conformist mutation strategy

Every individual of EP represents a solution of the optimization problem and evolves via the mutation operator. Most EPs (Fogel 1991; Hedar and Fukushima 2006; Lee and Yao 2004; Yao et al. 1999) are based on the variation vector calculated from stochastic distribution functions. If the mutation simply relies on a single probabilistic distribution while ignoring the features of the optimization function, the variation vector is updated in a non-heuristic way, which may lead to inefficiencies of the EPs (Fogel 1991; Hedar and Fukushima 2006; Lee and Yao 2004; Yao et al. 1999). We modify the approach of the variation vector by extracting heuristic information from the iterating population. Our aim is to help EP evolution become more heuristic according to the population status rather than the fixed value range of a stochastic distribution function.

Our idea is inspired by a psychological model of conformity behavior proposed by Asch (1956). The model reveals how a person refers to others' behaviors in the same group to adjust his/her behavior to achieve an optimal social performance. By simulating the model of conformity behavior, we modify the variation vector of Eq. (1) in Algorithm 1 to update the individual adaptively through referring to more information of the population. There are three aspects of Asch's model, which will be discussed below. Following model, the proposed mutation simulates the situation that a person, or a group member, usually refers to others' behaviors in their group to optimize their action. In the following introduction and analysis, the person and the group of the model metaphorically represent the individual and the population of EP, respectively.

The first aspect is behavior reference. A person may refer to the behaviors and ideas of others in the group for decision-making. Practically speaking, the reference of others' performance is needed more when the person is unfamiliar with the situation. For EP algorithms, every individual needs to be renewed with other individuals' information of the population when faced with an unknown optimization problem. Therefore, the proposed updating variation vector will contain a factor generated by other individuals.

The second aspect includes the person's belief regarding others and the attraction of the majority to the individual. He/she will fully trust the group when it is high-cohesion. A high-cohesion group always has a powerful leader or other elite members who have a positive impact on other members' behaviors and ideas. For the EP algorithm, the elite individual exerts a positive influence on the updating of every individual for optimization. As a result, a factor of the elite individual's information is added for updating the variation vector.

The last aspect is the negative effects of the fear of being distinctive in the group. An elite member may feel lonely and unsupported by others when he/she is too distinctive. Even though his/her idea may be sound, they may still relinquish it if it is different from others'. Yet, blind obedience to a collective will exert a negative effect on that elite member's development; thus, the distinctive idea is necessary. In a similar way, the individual in an EP algorithm cannot be optimized when the population converges to a local optimization status. Therefore, it is necessary to modify the variation vector by adding a factor that helps the individual stands out from others when the population is premature convergence.

The simulated-conformist idea was introduced to update the variation vector, which is the key role of an EP algorithm. For example, the decision vector $\mathbf{x}_i$ in Eq. (1) is considered as the person (group member), and the variation vector $\boldsymbol{\sigma}_i$ is their decision. The $\mathbf{x}_i$ is updated by using the $\boldsymbol{\sigma}_i$, and it likes the person improves his behavior by making a new decision. The simulated-conformist idea is used to update the $\boldsymbol{\sigma}_i$. This operation is analogous to make a decision by referring to others' ideas and behaviors.

The proposed evolutionary operator is a mutation with a simulated-conformist vector. Our design focuses on Step 3 of Algorithm 1. The variation vector for each individual is updated by the simulated-conformist vector including three factors inspired from Asch' s model of conformity behavior (Asch 1956). The factors are computed with individuals in the EP population. Thus, the offsprings are generated through renewing individuals by the updated variation vector. The key point of our proposed strategy focuses on the computational indicator of the simulated-conformist vector.

## 3.2 Simulated-conformist vector

The decision variable of each individual is considered to be a basic element for computing the simulated-conformist vector. The simulated-conformist vector for the $i$th individual is $\Delta_i$, and is used for the mutation of $\mathbf{v}_i = (\mathbf{x}_i, \boldsymbol{\sigma}_i)$, where $i = 1, 2, \ldots, k$. $\Delta_i$ is calculated by three factors simulating the three aspects of the conformity behavior model (Asch 1956). The factors for $\mathbf{v}_i$ are the behavior-reference factor, $\Gamma_i^{(1)}$; the majority-impact factor, $\Gamma_i^{(2)}$; and the distinctive-impact factor $\Gamma_i^{(3)}$, as follows.

(1) *Behavior-reference factor* The behavior-reference factor simulates the aspect of behavior reference. It is a difference between the decision variable of the mutated individual and the mean of a randomly selected group. The group are generated by randomly selecting other $M_i$ individuals of the current population. The group size $M_i$ is calculated by a Poisson distribution that

$$P(M_i = y) = \frac{(\frac{k}{2})^y \cdot e^{-(\frac{k}{2})}}{y!} \tag{2}$$

Poisson distribution is usually used to estimate a number of random incidents (such as receiving calls, meeting people in the street, and other social behavior by people) in an unit time. Therefore, we use this distribution to estimate the number of persons in the aspect of behavior reference. When $M_i$ is obtained and $M_i$ individuals are selected, the behavior-reference factor is calculated by Expression (3).

$$\Gamma_{i,j}^{(1)} = x_{i,j} - \frac{1}{M_i} \sum_{m=1}^{M_i} x_{r_m, j}, \tag{3}$$

where $i = 1, 2, \ldots, k$; $j = 1, 2, \ldots, n$; $r_m$ is the random integer from 1 and $k$ such that $r_m \neq i$. Where $m = 1, 2, \ldots, M_i$. $r_1, r_2, \ldots, r_{M_i}$ represent the indexes of the selected individuals in population. They are different from each other and used to calculate the $\Gamma_{i,j}^{(1)}$. The behavior-reference factor simulates a person's decision based on the behaviors of other $M_i$ persons. $\Gamma_{i,j}^{(1)}$ is added to $\Delta_i$ to renew the mutated individual $\mathbf{v}_i$, so $\mathbf{x}_i$ is updated by $x_{r_1}, x_{r_2}, \ldots, x_{r_{M_i}}$ with the behavior-reference factor. Once the $k-1$ probabilities are calculated by expression (2), they will be normalized to sum up to 1. The $M_i$ is obtained by a roulette of $k-1$ probabilities.

The behavior-reference factor calculated from Eq. (3) only contains stochastic information from the selected individuals. However, the factor may not necessary guide the mutated individual toward the optimal solution. It is therefore important to add factors into the calculation of the simulated-conformist vector to make the mutation better suited for optimization.

(2) *Majority-impact factor* The majority-impact factor simulates the attraction of the leader or elite members to the majority of individuals. The best-so-far individual is used to simulate the leader or elite members of the group. They are considered to have most information for global optimization among the population (Liang et al. 2006; Zhang and Jing 2008).

Let $\mathbf{v}_{best} = (\mathbf{x}_{best}, \boldsymbol{\sigma}_{best})$ be the best-so-far individual. We choose the best *decision vector* $\mathbf{x}_{best}$ as a part of the majority-impact factor $\Gamma_i^{(2)}$ for $\mathbf{v}_i$. The majority-impact factor is added to the simulated-conformist vector for the $i$th individual to be mutated,

$$\Gamma_{i,j}^{(2)} = \left(1 + \frac{|S_j|}{k}\right) \cdot (x_{best,j} - x_{i,j}) \tag{4}$$

where $j = 1, 2, \ldots, n$; $S_j$ is the set of individuals that meet two properties. The individual in $S_j$ has two properties that its elements are closer to $\mathbf{x}_{best}$ than $\mathbf{x}_i$ in $j$th dimension, and that the evaluation value of its elements is better than the one of $\mathbf{x}_i$. Expression (4) indicates that the weight of the $j$th dimension will be larger if there are more individuals closer to $\mathbf{x}_{best}$ in $j$th dimension and better than $\mathbf{x}_i$. Therefore, the majority-impact factor $\Gamma_{i,j}^{(2)}$ enhances the impact of $\mathbf{x}_{best}$ on the mutated individual according to the majority of the population. Hence, the simulated-conformist vector for the $i$th individual includes a behavior-reference factor and a majority-impact factor. Every $\Delta_i$ is calculated with a shared vector $\mathbf{x}_{best}$ which is useful for fast convergence (Gämperle et al. 2002).

From Eqs. (3) and (4), the behavior-reference factor and the majority-impact factor are both produced by the past population including parents and the best-so-far individual. Besides them, the diversity of population is also very important for global optimization (Alam et al. 2011; Liang et al. 2006). However, there is no characteristic inclined to help individuals with diversity in Eqs. (3) and (4). One efficient method for increasing diversity is to make the offsprings very different from the parents. Therefore, we simulate the group member's distinctive behaviors to improve the simulated-conformist vector for population diversity.

(3) *Distinctive-impact factor* As discussed above, $\Delta_i$ will contain a factor to make the mutated individual different from others in the population. Inspired by the strategy of using a Gaussian distribution (Alam et al. 2011; Fogel 1992), we apply a distinctive-impact factor for the $i$th individual to be mutated,

$$\Gamma_{i,j}^{(3)} = N_j(0, \delta), \tag{5}$$

where $j = 1, \ldots, n$; and $N_j(0, \delta)$ is a random number by Gaussian distribution, generated for each $j$. Furthermore,

$$\delta = \left(x_j^{max} - x_j^{min}\right) / D, \tag{6}$$

where $[x_j^{min}, x_j^{max}]$ is the feasible interval for the *decision variable* in the $j$th dimension and $D$ is an adaptive parameter to control the standard deviation of $N_j(0, \delta)$.

The adaptive setting of $D$ is described as follows. $D$ is used for adjusting the standard deviation. A fixed $D$ corresponds to a special searching scope of the Gaussian distinctive-impact factor. In the initialization of the proposed algorithm EP-SMS, setting $D$ to 10 can cause a large mutation step with a high probability to make the search jumps of EP-SMS large. As $D$ increases, $\delta$ becomes smaller and the search scope of EP-SMS converges. We use a counting variable, *count*, and threshold, $\tau$, to control the increment of $D$. The variable *count* is set to record the number of times the offspring is better than the parent through the proposed mutation. $\tau$ is a threshold value of the variable *count*, $D$ will increase if $count < \tau$, where $\tau = 10\,\%$ of the population size, set empirically. $D$ is reset to 10 when $D \geq 10^{30}$. In each iteration, $D = D/10$ when $count < \tau$; $D = 10D$ when $count > \tau$. The factor of 10 acts like a longer jump adaptive setting for the parameter rather than the factor of 2, making the dynamic change of $D$ more obvious. Our approach is supported by the empirical results presented in Table 1.

The addition of the distinctive-impact factor produces a change to the previous result based on the linear combination of parents and best-so-far individual. Consequently, the distinctive-impact factor of Eq. (5) is inclined to mutate an offspring to make it more different from its parent.

The three factors are summed to be the simulated-conformist vector,

$$\Delta_{i,j} = \Gamma_{i,j}^{(1)} + \Gamma_{i,j}^{(2)} + \Gamma_{i,j}^{(3)}, \tag{7}$$

where $j = 1, 2, \ldots, n$.

The performance of the three factors and their combination are analyzed in Sect. 4.2.

### 3.3 Mutation by simulated-conformist vector

The proposed EP follows the framework of Algorithm 1. The mutation of the proposed algorithm is presented by Algorithm 2

The mutation runs like seeking differences while reserving common parts. Every dimension of the variable will be updated by the selected simulated-conformist vector with mutation probability, $P_m$. Each element of the $j$th dimension remains unchanged with probability $1 - P_m$. Thus, only some dimensions of the variation vector are able to change in the mutation operator, unlike blind searching by stochastic distributions.

Equations (8) and (9) are used to renew the $j$th dimension value of the decision variable with probability $P_m$, while the value remains unchanged through Eqs. (10) and (11) with probability $1 - P_m$.

**Algorithm 2:** Simulated-conformist Mutation (Eq. (1) in Algorithm 1)

```
Input: $i$th individual $\mathbf{v}_i$ and problem dimension $n$.
Output: $i$th mutated individual $\overline{\mathbf{v}}_i$.
for $j = 1 \leftarrow n$ do
    generate random real numbers $r_j \in [0,1]$ and $p_j \in [0,1]$
    if $r_j < P_m$ then
        //$\Delta_{ij}$ is calculated by Eq. (7)
        $\overline{\sigma}_{ij} = \Delta_{ij}$,    (8)
        $\overline{x}_{ij} = (1 - p_j) \cdot x_{ij} + p_j \cdot \overline{\sigma}_{ij}$,    (9)
    else
        $\overline{\sigma}_{ij} = \sigma_{ij}$,    (10)
        $\overline{x}_{ij} = x_{ij}$,    (11)
    end
end
```

Other steps of the proposed EP are the same as Algorithm 1.

## 4 Experimental results

We verified the effectiveness of our proposed EP-SMS by conducting three numerical experiments.

(1) We investigated the desired setting of the systematic parameters used in EP-SMS (Sect. 4.1).
(2) We mainly compared EP-SMS with different EPs obtained by using different combinations of the proposed three factors, i.e., the behavior-reference factor, the majority-impact factor, and the distinctive-impact factor (Sect. 4.2).
(3) We conducted the benchmark problems (Yao et al. 1999) to compare EP-SMS with the existing representative EPs (Alam et al. 2011, 2012; Lee and Yao 2004; Yao et al. 1999) and other algorithms (Bratton et al. 2007; Brest et al. 2006; Hansen 2006; Hansen et al. 2003) with similar strategies to evolutionary programming (Sect. 4.3).

### 4.1 Experimental setting of EP-SMS

Three parameters need to be set in advance: population size $k$, mutation probability $P_m$, and divided/multiple factor $D$. The population size of EP-SMS was set to be 100, similar to the setting for FEP (Yao et al. 1999), ALEP (Lee and Yao 2004), and RTEP (Alam et al. 2011). The mutation probability was set to $P_m = 0.1$, following the suggestion of Jung (2003). The divided/multiple factor of $D$ is set to be 10 according to the results in Table 1.

To make a fair comparison, we took the evaluation of fitness value as the basic operation of the compared algorithms. The 23 benchmark functions (Yao et al. 1999) were used in our experimental studies. Functions $F1$–$F13$ are high-dimensional functions. Functions $F1$–$F7$ are unimodal functions. Functions $F8$–$F13$ are multimodal functions. Functions $F14$–$F23$ are low-dimensional functions with many valleys and multiple minimum values. The detailed calculation of each benchmark function is given in Table 2.

For $F1$–$F13$, the maximum evaluation number was 150,000, which is consistent with the setting of DGEP (Alam et al. 2012), RTEP (Alam et al. 2011), ALEP (Lee and Yao 2004), Self-Adaptive DE (Brest et al. 2006), PSO (Ratnaweera et al. 2004), and DE (Brest et al. 2006). For example, the population size and maximum iteration number of Self-Adaptive DE (Brest et al. 2006) are 100 and 1500, respectively, i.e., the total number of evaluations does not exceed 150,000. For $F14$–$F23$, we set a uniform maximum evaluation number = 10,000. This is the same as for DGEP (Alam et al. 2012), RTEP (Alam et al. 2011), PSO (Ratnaweera et al. 2004), and DE (Brest et al. 2006). The setting also corresponds to ALEP (Lee and Yao 2004) and Self-Adaptive DE (Brest et al. 2006) except for $F16$ and $F18$. The maximum evaluation number of ALEP and Self-Adaptive DE for $F16$ and $F18$ was 3000.

Besides the constant parameters, there is an adaptive parameter, $D$ [Eq. (6)]. In this study, $D$ was set in an adaptive way rather than as a constant. In order to verify whether the adaptive way has advantage, four candidate strategies

**Table 1** Investigation of different $D$ settings to solve the benchmark functions (Yao et al. 1999) in 1998

| Function | Adaptive $D = D/10$ or $D = 10D$ | Adaptive $D = D/2$ or $D = 2D$ | $D = 1000$ | Non-$D$ |
|---|---|---|---|---|
| $F1$ | 5.19e−57 | 1.11e−56 | 6.04e−07$^\dagger$ | 1.24e−22$^\dagger$ |
| $F3$ | 3.01e−12 | 8.05e−10$^\dagger$ | 4.24e−02$^\dagger$ | 3.95e−01$^\dagger$ |
| $F10$ | 7.10e−15 | 6.68e−15 | 3.93e−04$^\dagger$ | 4.06e−06$^\dagger$ |
| $F12$ | 1.57e−32 | 1.57e−32 | 3.06e−08$^\dagger$ | 7.35e−23$^\dagger$ |
| $F13$ | 1.34e−32 | 1.34e−32 | 1.10e−08$^\dagger$ | 4.97e−21$^\dagger$ |
| $F14$ | 9.98e−01 | 9.98e−01 | 9.98e−01 | 9.98e−01 |
| $F15$ | 3.08e−04 | 3.10e−04 | 3.10e−04 | 3.08e−04 |
| $F18$ | 3.00e+00 | 3.00e+00 | 3.00e+00 | 3.00e+00 |

$^\dagger$ The $p$ value is significant ($p$ value $< 0.01$) at a 0.01 level of significance by one-way ANOVA test

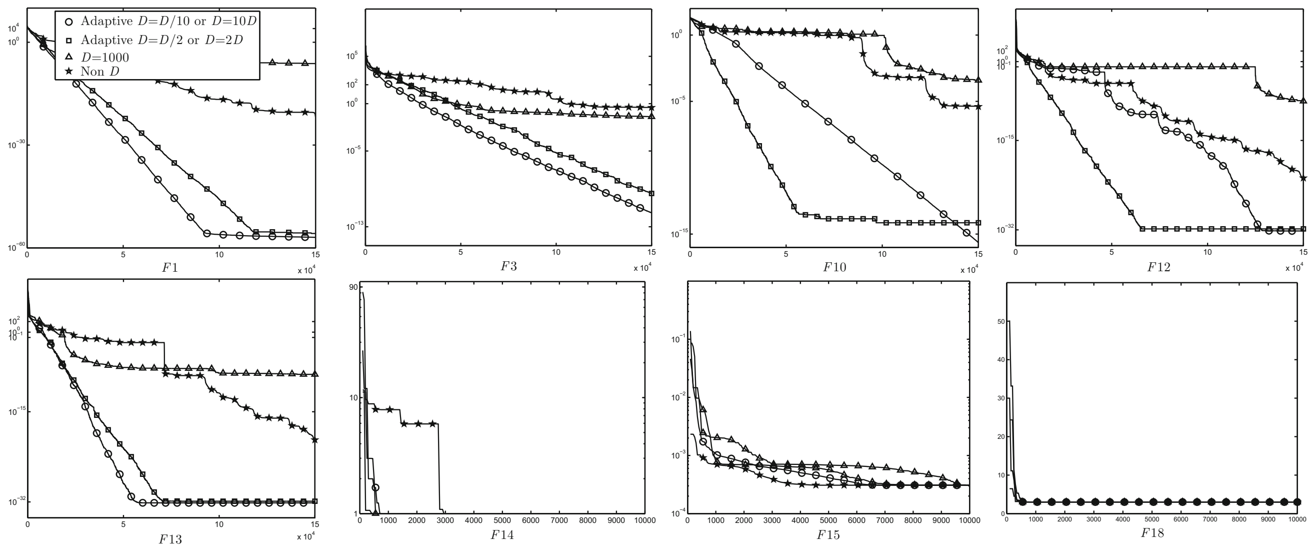

**Fig. 1** Investigated results of different $D$ settings. The *horizontal axis* is the number of generations. The *vertical axis* is the fitness value, which is the average value after running 51 times

**Table 2** Twenty-three benchmark functions used in experiments, where n is the dimension of the function. $F_{min}$ is the minimum value of the function, and $S \subseteq R^n$

| Benchmark function | $n$ | S | $F_{min}$ |
|---|---|---|---|
| $f_1(x) = \sum_{i=1}^{n} x_i^2$ | 30 | $[-100, 100]^n$ | 0 |
| $f_2(x) = \sum_{i=1}^{n} \|x_i\| + \prod_{i=1}^{n} \|x_i\|$ | 30 | $[-10, 10]^n$ | 0 |
| $f_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j)^2$ | 30 | $[-100, 100]^n$ | 0 |
| $f_4(x) = max_i \{\|x_i\|, 1 \leq i \leq n\}$ | 30 | $[-100, 100]^n$ | 0 |
| $f_5(x) = \sum_{i=1}^{n-1} [100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2]$ | 30 | $[-30, 30]^n$ | 0 |
| $f_6(x) = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$ | 30 | $[-100, 100]^n$ | 0 |
| $f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0, 1)$ | 30 | $[-1.28, 1.28]^n$ | 0 |
| $f_8(x) = \sum_{i=1}^{n} -x_i sin(\sqrt{\|x_i\|})$ | 30 | $[-500, 500]^n$ | −12,569.5 |
| $f_9(x) = \sum_{i=1}^{n} [x_i^2 - 10\cos(2\pi x_i) + 10]$ | 30 | $[-5.12, 5.12]^n$ | 0 |
| $f_{10}(x) = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp(\frac{1}{n}\sum_{i=1}^{n}\cos 2\pi x_i) + 20 + e$ | 30 | $[-32, 32]^n$ | 0 |
| $f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$ | 30 | $[-600, 600]^n$ | 0 |
| $f_{12}(x) = \frac{\pi}{n}\left\{10\sin^2(\pi y_i) + \sum_{i=1}^{n-1}(y_i - 1)^2[1 + 10\sin^2(\pi y_i + 1)] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} \upsilon(x_i, 10, 100, 4)$ <br> $y_i = 1 + \frac{1}{4}(x_i + 1)$ <br> $\upsilon(x_i, a, k, m) = \begin{cases} k(x_i - a)^m), & x_i > a \\ 0, & -a \leq x_i \leq a \\ k(-x_i - a)^m, & x_i < -a \end{cases}$ | 30 | $[-50, 50]^n$ | 0 |
| $f_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n-1}(x_i - 1)^2[1 + \sin^2(3\pi x_{i+1})] + (x_n - 1)[1 + \sin^2(2\pi x_n)]\right\} + \sum_{i=1}^{n} \upsilon(x_i, 5, 100, 4)$ | 30 | $[-50, 50]^n$ | 0 |
| $f_{14}(x) = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right]^{-1}$ | 2 | $[-65.536, 65.536]^n$ | 1 |
| $f_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$ | 4 | $[-5, 5]^n$ | 0.0003075 |
| $f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1 x_2 - 4x_2^2 + 4x_2^4$ | 2 | $[-5, 5]^n$ | −1.0316285 |
| $f_{17}(x) = (x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6)^2 + 10(1 - \frac{1}{8\pi})cosx_1 + 10$ | 2 | $[-5, 10] \times [0, 15]$ | 0.398 |
| $f_{18}(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times [30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$ | 2 | $[-2, 2]^n$ | 3 |
| $f_{19}(x) = -\sum_{i=1}^{4} c_i exp[-\sum_{j=1}^{4} a_{ij}(x_j - p_{ij})^2]$ | 4 | $[0, 1]^n$ | −3.86 |
| $f_{20}(x) = -\sum_{i=1}^{4} c_i exp[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2]$ | 6 | $[0, 1]^n$ | −3.32 |
| $f_{21}(x) = -\sum_{i=1}^{5}[(x - a_i)(x - a_i)^T + c_i]^{-1}$ | 4 | $[0, 10]^n$ | −10 |
| $f_{22}(x) = -\sum_{i=1}^{7}[(x - a_i)(x - a_i)^T + c_i]^{-1}$ | 4 | $[0, 10]^n$ | −10 |
| $f_{23}(x) = -\sum_{i=1}^{10}[(x - a_i)(x - a_i)^T + c_i]^{-1}$ | 4 | $[0, 10]^n$ | −10 |

of $D$ were investigated when solving 8 selected benchmark instances: non-$D$ [Eq. (5)] is removed from Eq. 7), constant $D = 1000$, adaptive $D = D/10$ or $D = 10D$, and $D = D/2$ or $D = 2D$. The selected instances include two unimodal functions ($F1$ and $F3$), three high-dimensional multimodal functions ($F10$, $F12$, and $F13$), and three low-dimensional multimodal functions ($F14$, $F15$, and $F18$).

Each instance was calculated 51 times using the proposed EP-SMS with the four $D$ settings, and the results are shown in Table 1 and Fig. 1. As we have discussed, $D$ is used to control the standard deviation of Gaussian distribution in Eq. (5), which adjusts the diversity of the population. A larger $D$ can cause a smaller mutation step and accelerate the convergence of EP-SMS. A smaller $D$ can cause a larger mutation step and make the search jumps of EP-SMS larger. When EP-SMS shows a good searching performance, increasing $D$ can help EP-SMS quickly converge to the optimum. When the searching performance of EP-SMS is decreased, decreasing $D$ can help EP-SMS make a larger search jump and find the better solution with the high probability. The experimental results shown in Table 1 and Fig. 1 prove that the adaptive way is better than non-$D$ and constant $D$. Besides, the adaptive $D = D/10$ or $D = 10D$ shows a significant performance improvement. Therefore, the adaptive $D = D/10$ or $D = 10D$ is a reasonable way. Empirically, we chose it to adjust the diversity of the EP population for the main comparison.

### 4.2 Performance investigation of simulated-conformist vector

In our proposed EP-SMS algorithm, the mutation is based on the simulated-conformist vector, which is constructed by combining the behavior-reference, majority-impact, and distinctive-impact factors. We investigated the effectiveness of these three factors by comparing multiple possible EP-SMS variants using different combinations of the factors. Specifically, we consider seven EP-SMS variants,

(1) EP-SMS-1: The simulated-conformist vector only includes a behavior-reference factor [Eq. (3)].
(2) EP-SMS-2: The simulated-conformist vector only includes a majority-impact factor [Eq. (4)].
(3) EP-SMS-3: The simulated-conformist vector only includes a Gaussian distinctive-impact factor [Eqs. (5), (6)].
(4) EP-SMS-1-2: The simulated-conformist vector contains behavior-reference factor and majority-impact factors [Eq. (12)].

$$\Delta_{i,j} = \Gamma^{(1)}_{i,j} + \Gamma^{(2)}_{i,j}. \tag{12}$$

(5) EP-SMS-1-3: The simulated-conformist vector contains a behavior-reference factor and a distinctive-impact factor [Eq. (13)]. So for the $i$th individual to be mutated ($j = 1, \ldots, n$),

$$\Delta_{i,j} = \Gamma^{(1)}_{i,j} + \Gamma^{(3)}_{i,j}. \tag{13}$$

(6) EP-SMS-2-3: The simulated-conformist vector contains a majority-impact factor and a distinctive-impact factor [Eq. (14)], So for the $i$th individual to be mutated ($j = 1, \ldots, n$),

$$\Delta_{i,j} = \Gamma^{(2)}_{i,j} + \Gamma^{(3)}_{i,j}. \tag{14}$$

(7) EP-SMS-1-2-3 (i.e., the proposed EP-SMS): simulated-conformist vector is made up of three factors [Eq. (7)]. EP-SMS-1-2-3 is a complete version of the proposed EP-SMS algorithm.

We compared the EP-SMS variants by testing them on the eight benchmark functions (Yao et al. 1999), i.e., $F1$, $F3$, $F10$, $F12$–$F15$, and $F18$. Each instance was calculated 51 times. These functions are representative, and they cover all possible types of the benchmark functions: high-dimensional unimodal functions, high-dimensional multimodal functions, and low-dimensional functions with many valleys and multiple minimum values.

The means and the standard deviations of the minimized evaluation functions for the different algorithms and functions are shown in Table 3. We used an ANOVA test for significance testing. The statistical result shows that EP-SMS variants using only one factor, i.e., EP-SMS-1, EP-SMS-2, and EP-SMS-3, are inferior to other EP-SMS variants. EP-SMS-1-2 shows some improvement over EP-SMS-1 and EP-SMS-2, which indicates the combination of the behavior-reference and the majority-impact factors is better than any single case. Similar results can be observed for EP-SMS-1-3 versus EP-SMS-1 or EP-SMS-3, and EP-SMS-2-3 versus EP-SMS-2 or EP-SMS-3. EP-SMS-1-2-3 is the highest performing among the EP-SMS variants, which shows the effectiveness of the proposed simulated-conformist vector (combining three factors).

The behavior-reference factor contributes heuristic information from the parents to the mutation. The performance of EP-SMS-2-3 is unstable in different instances. EP-SMS-1-2-3 is more robust than EP-SMS-2-3 owing to the heuristic information of individuals which makes the optimization capacity of mutation adaptively fit to different kinds of benchmark functions.

The optimization acceleration guarantees the mutation in a direction of global optimization. EP-SMS-1-2-3 improves on EP-SMS-1-3 in the quality of solutions, i.e, its index value is significantly closer to optimal. The superiority of EP-SMS-1-2-3 to EP-SMS-1-3 is significant for $F12$, $F13$ and $F15$ while their performance difference is not significant for the other five functions.

The distinctive-impact factor of Gaussian distribution assists the mutation in diversifying the evolutionary population. The solution of EP-SMS-1-2 was apparently trapped in local optimization. However, it makes substantial progress when the factor of Eq. (5) was added to the simulated-conformist vector, since the distinctive-impact factor improves the diversity of the population. As a result, EP-SMS-1-2-3 performs significantly better than EP-SMS-1-2 for $F1$, $F3$, $F10$, $F12$ and $F13$, while there are no significant differences among other three functions.

EP-SMS-1-2-3 is the best performing algorithm. The high performance verifies the advantage of the combination of the three factors, each of which is needed to an optimal EP mutation.

### 4.3 Comparison of EP-SMS with other EPs

To verify the effectiveness of the proposed mutation strategy with the simulated-conformist vector, EP-SMS was com-

**Table 3** Comparison of seven EP-SMS variants in the index "mean (standard deviation) of minimal evaluation function value" to solve the benchmark functions (Yao et al. 1999) in 1998

| Function | EP-SMS-1-2-3 | EP-SMS-1 | EP-SMS-2 | EP-SMS-3 | EP-SMS-1-2 | EP-SMS-1-3 | EP-SMS-2-3 |
|---|---|---|---|---|---|---|---|
| $F1$ | 5.20e−57 | 2.59e−40† | 6.47e+00† | 2.27e−37† | 4.17e−34† | 4.56e−57 | 4.39e−36† |
| | (2.44e−57) | (1.25e−40) | (2.32e+00) | (6.12e−37) | (1.94e−33) | (1.97e−57) | (1.00e−35) |
| $F3$ | 3.02e−12 | 9.24e−04† | 1.64e+04† | 5.32e+01† | 1.11e−01† | 4.93e−12 | 8.43e+01† |
| | (5.25e−12) | (1.49e−04) | (3.24e+03) | (3.65e+01) | (4.85e−01) | (9.98e−12) | (4.88e+01) |
| $F10$ | 7.11e−15 | 8.15e+00† | 1.27e+00† | 3.71e−14 | 2.34e−07† | 7.11e−15 | 3.97e−04† |
| | (1.82e−15) | (2.86e+00) | (2.80e−01) | (5.23e−15) | (1.13e−06) | (0.00e+00) | (1.94e−04) |
| $F12$ | 1.57e−32 | 7.05e+00† | 3.00e−02† | 2.00e−02† | 4.00e−02† | 1.20e−01† | 5.40e−03† |
| | (5.47e−48) | (1.23e+01) | (3.00e−02) | (4.00e−02) | (2.00e−02) | (3.00e−02) | (2.00e−02) |
| $F13$ | 1.35e−32 | 2.57e+01† | 2.80e−01† | 8.23e−04† | 5.71e−03† | 8.06e−26† | 1.48e−03† |
| | (5.58e−48) | (3.51e+01) | (7.00e−02) | (2.37e−04) | (9.38e−03) | (3.95e−25) | (3.24e−03) |
| $F14$ | 9.98e−01 | 4.65e+00† | 1.04e+00† | 1.43e+00† | 9.98e−01 | 9.98e−01 | 9.98e−01 |
| | (9.93e−17) | (5.44e+00) | (1.94e−01) | (1.08e+00) | (9.41e−11) | (3.76e−13) | (1.58e−04) |
| $F15$ | 3.08e−04 | 1.23e−03† | 1.72e−02† | 6.84e−03† | 4.00e−04 | 1.15e−03† | 3.15e−03† |
| | (3.06e−07) | (3.90e−03) | (1.87e−02) | (8.11e−03) | (2.94e−04) | (3.93e−03) | (5.96e−03) |
| $F18$ | 3.00e+00 | 3.00e+00 | 1.25e+01† | 3.00e+00 | 3.00e+00 | 3.00e+00 | 3.00e+00 |
| | (0.00e+00) | (0.00e+00) | (1.37e+01) | (0.00e+00) | (0.00e+00) | (0.00e+00) | (0.00e+00) |

The standard deviations of 0.00 is just a small number with the precision of 1e−308 because the number will be displayed to be 0.00 in MATLAB when it is smaller than 1e−308

† The $p$ value is significant($p$ value <0.01) at a 0.01 level of significance by one-way ANOVA test

pared to other EPs, CEP (Yao et al. 1999), FEP (Yao et al. 1999), ALEP (Lee and Yao 2004), RTEP (Alam et al. 2011), and DGEP (Alam et al. 2012) by calculating the 23 benchmark functions (Yao et al. 1999). The first three algorithms are classical, whereas the remainder are recent EPs. CEP, FEP, and ALEP are EPs with mutation based on stochastic distribution. They have been shown to be effective in solving benchmark functions of different minima characteristics, respectively, corresponding to their different stochastic distributions. RTEP, and DGEP are EPs with heuristic mutation. RTEP is easy to implement for its smart framework, as its mutation was based on a linear combination of a neighboring item and a stranger item. It has been shown to be robust in solving unimodal and multimodal functions (Alam et al. 2011). DGEP is a two-level mutation EP based on the diversity of population, which is proved to be more powerful in solving multimodal functions rather than unimodal functions (Alam et al. 2012).

CEP, FEP, ALEP, DGEP, RTEP, and EP-SMS were implemented and run 51 times for each instance. The five algorithms to be compared are programmed strictly following the parameter settings of (Alam et al. 2011, 2012; Lee and Yao 2004; Yao et al. 1999). Table 4 shows the means and standard deviations of the minimization values and subsequent significance tests. The items denoted by "Null" are those that corresponding results cannot be directly obtained from the original literature (Alam et al. 2011; Lee and Yao 2004; Yao et al. 1999).

In general, none of the algorithms perform best for all of the benchmark functions.

- DGEP (Alam et al. 2012) achieves the best performance for one unimodal function ($F6$), one high-dimensional multimodal function ($F10$), and six low-dimensional multimodal functions ($F16$, $F18$–$F22$).
- RTEP (Alam et al. 2011) performs best for two unimodal function ($F3$ and $F6$), one high-dimensional multimodal function ($F11$), and two low-dimensional multimodal functions ($F16$ and $F22$).
- ALEP (Lee and Yao 2004) performs best for $F7$.
- CEP (Yao et al. 1999) performs best for $F17$.
- FEP (Yao et al. 1999) performs best for $F4$.

The proposed EP-SMS algorithm performs best for sixteen instances, including five unimodal functions ($F1$–$F3$, $F5$, $F6$), four high-dimensional multimodal functions with many minima ($F8$, $F9$, $F12$, $F13$), and seven low-dimensional multimodal functions with many valleys and a few minima ($F14$–$F16$, $F18$–$F20$, $F23$). Furthermore, EP-SMS achieves the same order of magnitude outcome as the best algorithm in three low-dimensional multimodal functions ($F17$, $F21$, $F22$).

Thus, the proposed EP-SMS algorithm was effective in tackling nineteen instances (approximately 82.6 %) of the test functions.

**Table 4** Comparing EP-SMS with other EPs in terms of the mean (standard deviation) of minimal evaluation function to solve the benchmark functions (Yao et al. 1999) in 1998

| Function | EP-SMS | DGEP | RTEP | FEP | CEP | ALEP |
|---|---|---|---|---|---|---|
| $F1$ | 5.19e−57 | 5.42e−08$^{\dagger}$ | 4.56e−43$^{\dagger}$ | 7.27e−11$^{\dagger}$ | 2.07e−04$^{\dagger}$ | 3.43e−05$^{\dagger}$ |
| | (2.44e−57) | (1.48e−08) | (8.00e−43) | (5.85e−11) | (3.45e−04) | (4.72e−06) |
| $F2$ | 1.79e−29 | 6.64e−12$^{\dagger}$ | 1.26e−27$^{\dagger}$ | 7.48e−06$^{\dagger}$ | 2.43e−01$^{\dagger}$ | Null |
| | (1.35e−29) | (3.61e−12) | (1.05e−27) | (3.69e−06) | (3.37e−01) | |
| $F3$ | 3.01e−12 | 4.13e−08$^{\dagger}$ | 3.45e+01$^{\dagger}$ | 6.99e−11$^{\dagger}$ | 1.41e+03$^{\dagger}$ | 7.67e+01$^{\dagger}$ |
| | (5.25e−12) | (9.25e−09) | (1.61e+01) | (5.70e−11) | (1.07e+03) | (1.10e+02) |
| $F4$ | 4.63e−02 | 1.06$^{\dagger}$ | 7.70e−05$^{\ddagger}$ | 4.66e−06$^{\ddagger}$ | 1.85e+01$^{\dagger}$ | Null |
| | (6.28e−02) | (0.32) | (1.08e−04) | (2.77e−06) | (6.88e+00) | |
| $F5$ | 2.78e−01 | 1.28$^{\dagger}$ | 2.67e+00$^{\dagger}$ | 9.31e+01$^{\dagger}$ | 8.42e+01$^{\dagger}$ | 6.21e+01$^{\dagger}$ |
| | (2.48e−01) | (0.76) | (7.18e−01) | (1.18e+02) | (8.87e+01) | (5.62e+01) |
| $F6$ | 0.00 | 0.00 | 0.00 | 0.00 | 1.29e+02$^{\dagger}$ | Null |
| | (0.00e+00) | (0.00e+00) | (0.00e+00) | (0.00e+00) | (3.72e+02) | |
| $F7$ | 7.59e−03 | 1.94e−12$^{\ddagger}$ | Null | 3.61e−05$^{\ddagger}$ | 5.62*e*-02$^{\dagger}$ | 8.00*e*-38$^{\ddagger}$ |
| | (7.48e−04) | (8.23e−13) | | (2.31e−05) | (2.38e−02) | (0.00e+00) |
| $F8$ | −12,569.5 | −12,567.4$^{\dagger}$ | Null | −10,997.6$^{\dagger}$ | −8012.1$^{\dagger}$ | −10,368.3$^{\dagger}$ |
| | (0.00e+00) | (0.18) | | (267) | (4.00e+02) | (4.36e+02) |
| $F9$ | 0.00 | 1.56e−02$^{\dagger}$ | 1.53e+01$^{\dagger}$ | 5.72$^{\dagger}$ | 8.34e+01$^{\dagger}$ | 6.12e+00$^{\dagger}$ |
| | (0.00e+00) | (5.12e−09) | (1.07e+01) | (1.91) | (2.56e+01) | (2.40e+00) |
| $F10$ | 7.10e−15 | 2.47e−16$^{\ddagger}$ | 3.55e−15$^{\ddagger}$ | 1.56e−02$^{\dagger}$ | 1.02e+01$^{\dagger}$ | 4.47e−03$^{\dagger}$ |
| | (1.82e−15) | (6.83e−17) | (0.00e+00) | (1.16e−03) | (3.51e+00) | (3.50e−04) |
| $F11$ | 6.51e−03 | 7.52e−14$^{\ddagger}$ | 8.88e−04$^{\ddagger}$ | 2.00e−02 | 1.91e−01$^{\dagger}$ | 5.93e−02 |
| | (1.34e−02) | (2.54e−14) | (2.45e−03) | (1.98e−02) | (1.31e−01) | (7.98e−02) |
| $F12$ | 1.57e−32 | 3.32e−12$^{\dagger}$ | 1.65e−02$^{\dagger}$ | 6.49e−06$^{\dagger}$ | 1.23e+00$^{\dagger}$ | 4.87e−03$^{\dagger}$ |
| | (1.43e−32) | (1.94e−13) | (6.47e−02) | (1.74e−06) | (1.61e+00) | (2.43e−03) |
| $F13$ | 1.34e−32 | 1.74e−04$^{\dagger}$ | Null | 8.34e−05$^{\dagger}$ | 1.52e+00$^{\dagger}$ | 5.43e−05$^{\dagger}$ |
| | (6.19e−33) | (2.65e−05) | | (1.65e−05) | (1.89e+00) | (1.40e−05) |
| $F14$ | 0.998 | 1.02$^{\dagger}$ | Null | 1.27$^{\dagger}$ | 1.76e+00$^{\dagger}$ | Null |
| | (0.00e+00) | (0.04) | | (0.596) | (9.97e−01) | |
| $F15$ | 3.08e−04 | 3.17e−04 | 1.21e−03$^{\dagger}$ | 8.88e−04$^{\dagger}$ | 8.27e−04$^{\dagger}$ | Null |
| | (1.10e−04) | (8.20e−05) | (1.97e−04) | (1.83e−04) | (2.15e−04) | |
| $F16$ | −1.031 | −1.031 | −1.031 | −1.31$^{\dagger}$ | −1.031$^{\dagger}$ | −1.031$^{\dagger}$ |
| | (0.00e+00) | (0.00e+00) | (0.00e+00) | (7.57e−09) | (4.58e−08) | (5.06e−08) |
| $F17$ | 0.398 | 0.398 | Null | 0.398$^{\dagger}$ | 0.398$^{\ddagger}$ | Null |
| | (1.86e−07) | (2.40e−07) | | (3.31e−06) | (4.58e−09) | |
| $F18$ | 3.00 | 3.00 | Null | 3.00$^{\dagger}$ | 3.04$^{\dagger}$ | 3.00$^{\dagger}$ |
| | (0.00e+00) | (0.00e+00) | | (3.31e−09) | (1.89e−01) | (3.26e−07) |
| $F19$ | −3.86 | −3.86$^{\dagger}$ | −3.87$^{\dagger}$ | −3.86$^{\dagger}$ | −3.86$^{\dagger}$ | Null |
| | (2.63e−15) | (9.40e−03) | (6.28e−02) | (4.62e−07) | (2.73e−07) | |
| $F20$ | −3.32 | −3.32 | −2.98$^{\dagger}$ | −3.28$^{\dagger}$ | −3.27$^{\dagger}$ | Null |
| | (5.74e−02) | (2.50e−04) | (3.38e−01) | (5.41e−02) | (5.93e−02) | |
| $F21$ | −8.43 | −9.87 | −9.79 | −5.89$^{\dagger}$ | −7.03 | −6.10$^{\dagger}$ |
| | (2.59e+00) | (0.540) | (1.48e+00) | (2.22) | (2.65e+00) | (2.41e+00) |

**Table 4** continued

| Function | EP-SMS | DGEP | RTEP | FEP | CEP | ALEP |
|---|---|---|---|---|---|---|
| $F22$ | −9.29 | $-10.47^{\ddagger}$ | −9.75 | $-6.89^{\dagger}$ | −7.94 | $-6.56^{\dagger}$ |
| | (2.28e+00) | (0.08) | (1.83e+00) | (2.95) | (2.74e+00) | (3.18e+00) |
| $F23$ | −10.54 | −10.51 | −10.46 | $-9.23^{\dagger}$ | $-8.35^{\dagger}$ | $-8.44^{\dagger}$ |
| | (1.87e−15) | (3.80e−02) | (2.69e−01) | (2.30) | (2.91e+00) | (2.88e+00) |

The standard deviation of 0.00 is a very small number; MATLAB shows 0.00 when it is smaller than 1e−308
$^{\dagger}$, $^{\ddagger}$ The $p$ value is significant at the 0.01 level of significance by one-way ANOVA test
$^{\dagger}$ The proposed EP-SMS performs better than the corresponding algorithm
$^{\ddagger}$ The corresponding algorithm performs better than EP-SMS

**Table 5** Comparing EP-SMS with other similar-strategy algorithms in terms of the mean (standard deviation) of minimal evaluation function to solve the benchmark functions (Yao et al. 1999) in 1998

| Function | EP-SMS | Self-Adaptive DE | PSO | DE | Constricted GBest PSO | CMA-ES |
|---|---|---|---|---|---|---|
| $F1$ | 5.19e−57 | $1.28e{-}28^{\dagger}$ | $1.79e{-}48^{\dagger}$ | $2.10e{-}18^{\dagger}$ | $3.12e{-}58^{\ddagger}$ | $6.06e{-}29^{\dagger}$ |
| | (2.44e−57) | (1.69e−28) | (4.94e−48) | (2.11e−18) | (1.39e−57) | (1.15e−29) |
| $F3$ | 3.01e−12 | $5.16e{-}02^{\dagger}$ | $2.83e{-}05^{\dagger}$ | $5.13e{-}02^{\dagger}$ | $3.15e{-}05^{\dagger}$ | $1.28e{-}26^{\ddagger}$ |
| | (5.25e−12) | (4.22e−02) | (4.42e−05) | (4.99e−02) | (3.23e−05) | (2.91e−27) |
| $F5$ | 2.78e−01 | $1.35e{+}01^{\dagger}$ | $1.43e{+}01^{\dagger}$ | $5.15e{-}08^{\ddagger}$ | $1.31e{+}01^{\dagger}$ | $6.85e{-}26^{\ddagger}$ |
| | (2.48e−01) | (2.33e+01) | (2.55e+01) | (9.91e−08) | (2.73e+01) | (2.79e−26) |
| $F8$ | −12,569.5 | −12,569.5 | $-9657.3^{\dagger}$ | $-5168.2^{\dagger}$ | $2.85e{+}03^{\dagger}$ | $-7024.4^{\dagger}$ |
| | (0.00e+00) | (1.86e−12) | (6.29e+02) | (3.87e+02) | (6.70E+02) | (5.60e+02) |
| $F9$ | 0.00e+00 | 2.13e−16 | $8.65e{+}01^{\dagger}$ | $1.76e{+}02^{\dagger}$ | $9.71e{+}01^{\dagger}$ | $2.24e{+}02^{\dagger}$ |
| | (0.00e+00) | (7.81e−16) | (2.37e+01) | (1.85e+01) | (2.13e+01) | (5.45e+01) |
| $F10$ | 7.10e−15 | $7.99e{-}15^{\ddagger}$ | $1.59e{+}00^{\dagger}$ | $4.28e{-}10^{\dagger}$ | $1.11e{+}00^{\dagger}$ | $1.94e{+}01^{\dagger}$ |
| | (1.82e−15) | 7.99e−15 | (1.03e+00) | (1.71e−10) | (1.83e+00) | (1.42e−01) |
| $F11$ | 6.51e−03 | $0.00e{+}00^{\ddagger}$ | 1.98e−02 | 5.71e−03 | 2.35e−02 | $6.90e{-}04^{\ddagger}$ |
| | (1.34e−2) | (0.00e+00) | (2.72e−02) | (7.67e−03) | (3.75e−02) | (2.41e−03) |
| $F12$ | 1.57e−32 | $9.02e{-}11^{\dagger}$ | $5.84e{+}00^{\dagger}$ | $8.29e{-}03^{\dagger}$ | $8.16e{-}01^{\dagger}$ | $9.02e{-}11^{\dagger}$ |
| | (1.43e−32) | (9.53e−25) | (1.88e+01) | (2.87e−03) | (3.05e+00) | (1.14e−22) |
| $F13$ | 1.34e−32 | $7.90e{-}11^{\dagger}$ | $6.15e{-}03^{\dagger}$ | $7.90e{-}11^{\dagger}$ | $4.21e{+}01^{\dagger}$ | $7.90e{-}11^{\dagger}$ |
| | (6.19e−33) | (4.55e−25) | (1.23e−02) | (4.40e−17) | (2.06e+02) | (8.04e−23) |
| $F16$ | −1.031 | −1.031 | −1.031 | −1.031 | −1.031 | −1.031 |
| | (0.00e+00) | (5.30e−13) | (0.00e+00) | (2.02e−16) | (0.00e+00) | (1.98e−16) |
| $F18$ | 3.00 | $3.00^{\dagger}$ | $3.00^{\dagger}$ | $3.00^{\dagger}$ | 3.00 | 3.00 |
| | (0.00e+00) | (6.71e−10) | (3.53e−13) | (1.10e−13) | (0.00e+00) | (0.00e+00) |
| $F21$ | −8.43 | $-10.15^{\ddagger}$ | $-6.43^{\dagger}$ | −9.40 | $-6.44^{\dagger}$ | $-4.86^{\dagger}$ |
| | (2.59) | (2.96e−02) | (3.47e+00) | (1.98e+00) | (3.46e+00) | (3.40e+00) |
| $F22$ | −9.29 | −9.74 | −7.22 | −9.53 | $-6.81^{\dagger}$ | $-5.07^{\dagger}$ |
| | (2.28e+00) | (3.72e−04) | (3.70e+00) | (1.05e+00) | (3.58e+00) | (3.36e+00) |
| $F23$ | −10.54 | −10.54 | $-6.00^{\dagger}$ | −10.54 | $-6.92^{\dagger}$ | $-6.06^{\dagger}$ |
| | (1.87e−15) | (5.26e−04) | (3.83e+00) | (1.47e−11) | (3.70e+00) | (3.51e+00) |

The standard deviations of 0.00 is just a small number; MATLAB displays to be 0.00 when it is smaller than 1e−308
$^{\dagger}$, $^{\ddagger}$ The $p$ value is significant at a 0.01 level of significance by one-way ANOVA test. The mark $^{\dagger}$ means that the proposed EP-SMS performs better than the corresponding algorithm. $^{\ddagger}$ means that the corresponding algorithm performs better than EP-SMS

**Table 6** Comparing EP-SMS with other EPs and CMA-ES in terms of the mean (standard deviation) of minimal evaluation function to solve the benchmark functions (Liang et al. 2013) in 2013

| Function | EP-SMS | DGEP | RTEP | FEP | CEP | ALEP | CMA-ES |
|---|---|---|---|---|---|---|---|
| $F1$ | 0.00e+00 | 0.00e+00 | 0.00e+00 | 7.05e+04$^{\dagger}$ | 8.37e+04$^{\dagger}$ | 7.58e+04$^{\dagger}$ | 0.00e+00 $^{\dagger}$ |
| | (0.00e+00) | (0.00e+00) | (0.00e+00) | (1.52e+03) | (5.70e+03) | (1.29e+04) | (2.96e−13) |
| $F2$ | 1.32e+05 | 3.30e+07$^{\dagger}$ | 2.73e+07$^{\dagger}$ | 2.21e+09$^{\dagger}$ | 2.69e+09$^{\dagger}$ | 3.22e+09$^{\dagger}$ | 0.00e+00$^{\ddagger}$ |
| | (6.16e+04) | (1.92e+07) | (5.07e+06) | (6.39e+08) | (5.11e+08) | (9.40e+08) | (2.73e−13) |
| $F3$ | 8.94e+07 | 2.39e+10$^{\dagger}$ | 1.99e+09$^{\dagger}$ | 8.44e+20$^{\dagger}$ | 8.78e+17$^{\dagger}$ | 5.72e+22$^{\dagger}$ | 1.15e+05 $^{\ddagger}$ |
| | (1.24e+08) | (1.91e+10) | (8.01e+08) | (3.00e+21) | (6.39e+20) | (1.60e+23) | (5.43e+04) |
| $F4$ | 4.97e+02 | 6.93e+04$^{\dagger}$ | 3.82e+04$^{\dagger}$ | 7.38e+04$^{\dagger}$ | 7.03e+04$^{\dagger}$ | 4.34e+06$^{\dagger}$ | 0.00e+00 $^{\ddagger}$ |
| | (1.65e+03) | (9.17e+03) | (4.87e+03) | (1.97e+04) | (8.46e+04) | (1.36e+07) | (2.91e−13) |
| $F5$ | 0.00e+00 | 0.00e+00$^{\ddagger}$ | 0.00e+00$^{\ddagger}$ | 5.15e+04$^{\dagger}$ | 5.68e+04$^{\dagger}$ | 7.54e+04$^{\dagger}$ | 0.00e+00 $^{\dagger}$ |
| | (9.91e−14) | (0.00e+00) | (0.00e+00) | (1.82e+04) | (1.62e+04) | (2.49e+04) | (1.89e−09) |
| $F6$ | 1.35e+01 | 6.00e+01$^{\dagger}$ | 3.29e+01$^{\dagger}$ | 2.08e+04$^{\dagger}$ | 2.04e+04$^{\dagger}$ | 2.54e+04$^{\dagger}$ | 1.03e+00 $^{\ddagger}$ |
| | (1.49e+01) | (3.42e+01) | (8.32e+00) | (4.13e+04) | (4.21e+03) | (5.92e+03) | (5.17e+00) |
| $F7$ | 1.11e+02 | 2.16e+02$^{\dagger}$ | 8.42e+01$^{\ddagger}$ | 2.30e+07$^{\dagger}$ | 1.49e+07$^{\dagger}$ | 9.33e+07$^{\dagger}$ | 1.29e+01 $^{\ddagger}$ |
| | (2.40e+01) | (4.91e+01) | (1.14e+01) | (2.30e+07) | (4.64e+07) | (1.24e+08) | (7.48e+00) |
| $F8$ | 2.09e+01 | 2.10e+01$^{\dagger}$ | 2.10e+01$^{\dagger}$ | 2.11e+01$^{\dagger}$ | 2.12e+01$^{\dagger}$ | 2.12e+01$^{\dagger}$ | 2.14e+01$^{\dagger}$ |
| | (8.34e−02) | (6.50e−02) | (5.13e−02) | (7.20e−02) | (5.82e−02) | (6.19e−02) | (7.74e−02) |
| $F9$ | 3.11e+01 | 3.54e+01$^{\dagger}$ | 2.92e+01$^{\ddagger}$ | 4.51e+01$^{\dagger}$ | 4.88e+01$^{\dagger}$ | 5.05e+01$^{\dagger}$ | 4.38e+01 $^{\dagger}$ |
| | (4.90e+00) | (2.35e+00) | (1.57e+00) | (1.94e+00) | (1.70e+00) | (2.39e+00) | (7.34e+00) |
| $F10$ | 1.65e−01 | 2.61e+00$^{\dagger}$ | 3.40e+01$^{\dagger}$ | 2.25e+03$^{\dagger}$ | 1.05e+04$^{\dagger}$ | 1.53e+04$^{\dagger}$ | 1.56e−02 $^{\ddagger}$ |
| | (1.01e−01) | (2.37e+00) | (1.69e+02) | (1.75e+03) | (1.88e+03) | (1.91e+03) | (1.09e−02) |
| $F11$ | 0.00e+00 | 1.32e−01$^{\ddagger}$ | 2.04e+01$^{\dagger}$ | 1.18e+03$^{\dagger}$ | 1.33e+03$^{\dagger}$ | 1.28e+03$^{\dagger}$ | 9.17 + −01$^{\dagger}$ |
| | (0.00e+00) | (3.23e−01) | (2.52e+00) | (5.01e+02) | (1.17e+02) | (2.21e+02) | (2.31e+02) |
| $F12$ | 1.45e+02 | 3.63e+02$^{\dagger}$ | 1.66e+02 | 1.21e+03$^{\dagger}$ | 1.23e+03$^{\dagger}$ | 1.29e+03$^{\dagger}$ | 7.37e+02 $^{\dagger}$ |
| | (4.55e+01) | (8.64e+01) | (2.10e+01) | (8.59e+02) | (1.20e+02) | (1.46e+02) | (8.98e+02) |
| $F13$ | 1.48e+02 | 4.26e+02$^{\dagger}$ | 1.96e+02$^{\dagger}$ | 1.20e+03$^{\dagger}$ | 1.18e+03$^{\dagger}$ | 1.29e+03$^{\dagger}$ | 1.40e+03 $^{\dagger}$ |
| | (6.03e+01) | (7.55e+01) | (1.83e+01) | (9.27e+02) | (8.39e+01) | (1.89e+02) | (1.39e+03) |
| $F14$ | 7.24e−01 | 3.51e+03$^{\dagger}$ | 8.34e+02$^{\dagger}$ | 9.59e+03$^{\dagger}$ | 9.13e+03$^{\dagger}$ | 9.80e+03$^{\dagger}$ | 5.09e+03 $^{\dagger}$ |
| | (6.63e−01) | (7.52e+02) | (1.09e+02) | (2.95e+02) | (3.90e+02) | (3.23e+02) | (6.94e+02) |
| $F15$ | 4.24e+03 | 6.87e+03$^{\dagger}$ | 5.51e+03$^{\dagger}$ | 9.23e+03$^{\dagger}$ | 9.67e+03$^{\dagger}$ | 1.02e+04$^{\dagger}$ | 5.28e+03 $^{\dagger}$ |
| | (6.88e+02) | (3.73e+02) | (3.49e+02) | (6.42e+02) | (4.15e+02) | (4.15e+02) | (7.84e+02) |
| $F16$ | 1.35e+00 | 2.50e+00$^{\dagger}$ | 2.03e+00$^{\dagger}$ | 4.56e+00$^{\dagger}$ | 5.48e+00$^{\dagger}$ | 5.72e+00$^{\dagger}$ | 9.81e−02 $^{\ddagger}$ |
| | (6.33e−01) | (2.48e−01) | (2.82e−01) | (6.77e−01) | (7.67e−01) | (1.07e+00) | (8.01e−02) |
| $F17$ | 3.90e+01 | 3.45e+01$^{\ddagger}$ | 6.63e+01$^{\dagger}$ | 1.23e+03$^{\dagger}$ | 1.02e+03$^{\dagger}$ | 1.23e+03$^{\dagger}$ | 4.03e+03 $^{\dagger}$ |
| | (3.65e−01) | (1.03e+00) | (3.95e+00) | (3.77e+02) | (4.88e+02) | (3.76e+02) | (8.86e+02) |
| $F18$ | 2.02e+02 | 5.26e+02$^{\dagger}$ | 2.45e+02$^{\dagger}$ | 1.19e+03$^{\dagger}$ | 1.06e+03$^{\dagger}$ | 1.32e+03$^{\dagger}$ | 4.00e+03 $^{\dagger}$ |
| | (5.14e+01) | (7.81e+01) | (1.63e+01) | (3.01e+02) | (3.41e+02) | (3.95e+02) | (6.77e+02) |
| $F19$ | 1.66e+00 | 2.84e+00$^{\dagger}$ | 5.56e+00$^{\dagger}$ | 1.36e+06$^{\dagger}$ | 1.26e+06$^{\dagger}$ | 2.15e+06$^{\dagger}$ | 3.46e+00 $^{\dagger}$ |
| | (5.10e−01) | (1.27e+00) | (6.50e−01) | (1.29e+06) | (2.00e+06) | (2.54e+06) | (8.36e−01) |
| $F20$ | 1.39e+01 | 1.46e+01$^{\dagger}$ | 1.25e+01$^{\ddagger}$ | 1.50e+01$^{\dagger}$ | 1.50e+01$^{\dagger}$ | 1.50e+01$^{\dagger}$ | 1.40e+01 |
| | (8.85e−01) | (2.68e−01) | (3.18e−01) | (1.69e−12) | (1.38e−08) | (0.00e+00) | (1.13e+00) |
| $F21$ | 3.04e+02 | 2.77e+02 | 2.05e+02$^{\ddagger}$ | 2.86e+03$^{\dagger}$ | 2.74e+03$^{\dagger}$ | 3.19e+03$^{\dagger}$ | 2.97e+02 |
| | (8.54e+01) | (7.41e+01) | (2.05e+01) | (3.27e+02) | (6.17e+02) | (9.67e+02) | (7.67e+01) |

**Table 6** continued

| Function | EP-SMS | DGEP | RTEP | FEP | CEP | ALEP | CMA-ES |
|---|---|---|---|---|---|---|---|
| $F22$ | 1.10e+02 | 4.25e+03$^‡$ | 9.56e+02$^†$ | 1.02e+04$^†$ | 1.00e+04$^†$ | 1.05e+04$^†$ | 7.06e+03 $^†$ |
| | (6.73e+01) | (1.03e+03) | (1.53e+02) | (2.67e+02) | (4.35e+02) | (2.70e+02) | (9.76e+02) |
| $F23$ | 5.16e+03 | 7.43e+03$^†$ | 6.08e+03$^†$ | 9.93e+03$^†$ | 9.17e+03$^†$ | 1.06e+04$^†$ | 6.81e+03 $^†$ |
| | (8.86e+02) | (4.16e+02) | (3.26e+02) | (4.47e+02) | (3.68e+02) | (4.00e+02) | (6.95e+02) |
| $F24$ | 2.75e+02 | 3.13e+02$^†$ | 2.71e+02$^‡$ | 5.52e+02$^†$ | 5.64e+02$^†$ | 7.18e+02$^†$ | 8.48e+02 $^†$ |
| | (1.59e+01) | (7.88e+00) | (4.64e+00) | (1.20e+02) | (1.08e+02) | (1.85e+02) | (6.17e+02) |
| $F25$ | 3.03e+02 | 3.11e+02$^†$ | 2.99e+02$^‡$ | 4.32e+02$^†$ | 4.19e+02$^†$ | 4.34e+02$^†$ | 3.07e+02 |
| | (1.38e+01) | (3.85e+00) | (4.16e+00) | (2.40e+01) | (2.81e+01) | (5.22e+01) | (1.35e+01) |
| $F26$ | 2.85e+02 | 2.08e+02$^‡$ | 2.02e+02$^‡$ | 4.32e+02$^†$ | 4.42e+02$^†$ | 4.93e+02$^†$ | 4.74e+02$^†$ |
| | (8.88e+01) | (4.71e+00) | (4.27e−01) | (1.97e+01) | (2.04e+01) | (9.43e+01) | (4.18e+02) |
| $F27$ | 1.30e+03 | 1.29e+03$^‡$ | 1.04e+03$^‡$ | 1.78e+03$^†$ | 1.88e+03$^†$ | 2.19e+03$^†$ | 5.88e+02 $^‡$ |
| | (1.09e+02) | (5.97e+01) | (5.65e+01) | (9.57e+02) | (1.08e+02) | (2.59e+02) | (2.18e+02) |
| $F28$ | 3.00e+02 | 1.88e+03$^†$ | 3.00e+02$^†$ | 8.96e+03$^†$ | 9.00e+03$^†$ | 1.02e+04$^†$ | 1.13e+03 $^†$ |
| | (0.00e+00) | (6.53e+02) | (1.70e−02) | (7.94e+02) | (1.18e+03) | (8.91e+02) | (2.60e+03) |

$^†$, $^‡$ The $p$ value is significant at the 0.01 level of significance by one-way ANOVA test. The mark $^†$ means that the proposed EP-SMS performs better than the corresponding algorithm. $^‡$ means that the corresponding algorithm performs better than EP-SMS
The standard deviation of 0.00 is a very small number; MATLAB shows 0.00 when it is smaller than 1e−308

Comparing to the different algorithms, the performance of EP-SMS is shown by three aspects:

(1) *CEP, FEP, and ALEP* EP-SMS is superior to CEP, FEP, and ALEP in the solution of all benchmark instances except $F4$, $F7$, and $F17$. Thus, the proposed EP-SMS algorithm is more effective than the pure stochastic distribution variance for EP mutation. FEP is more effective in tackling $F4$ than EP-SMS because FEP is very fast for unimodal and separable functions. The evaluation function of $F7$ is stochastic. Thus, the mean value of the EP-SMS minimum can easily change, although EP-SMS reaches the optimum in most cases. ALEP may be more powerful to solve the problem of stochastic evaluation function such as $F7$ than the other EPs. Furthermore, Table 7 shows that EP-SMS, FEP, and ALEP are significantly different for $F5$, $F8$, $F21$–$F23$ by a Bonferroni test.

(2) *RTEP* RTEP is stable in unimodal functions and multimodal functions. EP-SMS is superior to RTEP in the solution precision through all the benchmark functions except for $F4$ and $F11$. For $F11$, RTEP performs the best among the tested EPs. EP-SMS attained the optimal solution several times in the 51 runs, but the distinctive-impact factor of Eq. (5) caused a statistical error. Therefore, the mean value and deviation of the EP-SMS minimum are weaker than RTEP for $F4$ and $F11$. Table 7 shows that EP-SMS and RTEP are significantly different for $F9$ and $F20$ by a Bonferroni test.

(3) *DGEP* DGEP is a good performer for multimodal functions, $F10$, $F11$, $F21$, and $F22$, whereas EP-SMS achieved best performance nearly two-thirds of the multimodal functions. EP-SMS is significantly better than DGEP for twelve benchmark functions ($F1$–$F5$, $F8$–$F9$, $F12$–$F14$, $F19$), and performs comparably well with DGEP for six functions ($F6$, $F15$–$F18$, $F20$, $F21$, $F23$). Of the DGEP superior cases ($F6$, $F10$, $F15$, $F22$), $F10$ is composed of exponential and cosine wave functions, and has many peaks and valleys to cause local optimal traps. The EP-SMS capacity to jump out of the trap depends on its diversity strategy [Eqs. (5), (6)]. However, the strategy independently updates each dimensions of the variation vector. Thus, EP-SMS performs worse than DGEP in solving $F10$ since there is a two-level rule to avoid local optimization in DGEP (Alam et al. 2012). Similarly for $F7$ and $F11$, the DGEP is significantly better than EP-SMS according to the properties of $F7$ and $F11$. Table 7 shows EP-SMS and DGEP are significantly different for $F3$, $F5$, $F7$, and $F8$ by a Bonferroni test.

The experimental results show that the three advantages of EP-SMS are a result of the three factors integrated in the simulated-conformist vector.

(1) EP-SMS is effective for all three types of benchmark functions on average. Its effectiveness lies in the property of behavior-reference factor [Eq. (3)]. The behavior-reference factor makes the mutation execute based on the heuristic information of the population. The population of EP-SMS can adjust itself heuristically during the solution of different problems. Thus, the adjustment assists EP-SMS in having a stable optimization capacity.
(2) EP-SMS shows improved solution quality, as shown by the precision of the mean minimal function in Table 1. The improvement is based on the majority-

**Table 7** Multiple comparison results of Bonferroni test for compared EPs to solve the benchmark functions (Yao et al. 1999) in 1998

| Function | $p$ value | EP-SMS | DGEP | RTEP | FEP | CEP | ALEP |
|---|---|---|---|---|---|---|---|
| $F1$ | 3.6562e−07 | CEP | CEP | CEP | CEP | EP-SMS FEP RTEP DGEP ALEP | CEP |
| $F2$ | 8.3868e−09 | CEP | CEP | CEP | CEP | EP-SMS FEP DGEP RTEP | – |
| $F3$ | 4.4249e−25 | DGEP CEP | EP-SMS FEP RTEP CEP ALEP | DGEP CEP | DGEP CEP | EP-SMS FEP DGEP ALEP | DGEP CEP |
| $F4$ | 1.2079e−49 | CEP | CEP | CEP | CEP | EP-SMS FEP DGEP RTEP | – |
| $F5$ | 2.5170e−06 | DGEP FEP CEP ALEP | EP-SMS RTEP | DGEP FEP CEP | EP-SMS RTEP | EP-SMS RTEP | EP-SMS |
| $F6$ | 0.0208 | * | * | * | * | * | * |
| $F7$ | 1.3092e−38 | DGEP CEP | EP-SMS FEP CEP ALEP | – | DGEP CEP ALEP | EP-SMS FEP DGEP ALEP | DGEP FEP CEP |
| $F8$ | 6.1288e−142 | DGEP FEP CEP ALEP | EP-SMS FEP CEP ALEP | – | EP-SMS CEP DGEP ALEP | EP-SMS FEP DGEP ALEP | EP-SMS FEP DGEP CEP |
| $F9$ | 1.4729e−62 | RTEP CEP | RTEP CEP | EP-SMS FEP DGEP CEP | RETP CEP | EP-SMS FEP DGEP RTEP ALEP | CEP |
| $F10$ | 2.2858e−64 | CEP | CEP | CEP | CEP | EP-SMS FEP DGEP RTEP ALEP | CEP |
| $F11$ | 2.5960e−22 | CEP | CEP | CEP | CEP | EP-SMS FEP DGEP RTEP ALEP | CEP |
| $F12$ | 2.0264e−11 | CEP | CEP | CEP | CEP | EP-SMS FEP DGEP RTEP ALEP | CEP |
| $F13$ | 1.4724e−10 | CEP | CEP | – | CEP | EP-SMS FEP DGEP ALEP | CEP |
| $F14$ | 1.6340e−5 | CEP | CEP | – | CEP | EP-SMS FEP DGEP | – |
| $F15$ | 0.3294 | * | * | * | * | * | * |
| $F16$ | 0.0322 | * | * | * | * | * | * |
| $F17$ | 1.2509e−51 | CEP | CEP | – | CEP | EP-SMS FEP DGEP | – |
| $F18$ | 0.4104 | * | * | * | * | * | * |
| $F19$ | 0.4218 | * | * | * | * | * | * |
| $F20$ | 9.3154e−12 | RTEP | RTEP | EP-SMS FEP DGEP CEP | RTEP | RTEP | – |
| $F21$ | 1.6827e−07 | FEP ALEP | RTEP | DGEP FEP | EP-SMS | RTEP | EP-SMS |

**Table 7** continued

| Function | $p$ value | EP-SMS | DGEP | RTEP | FEP | CEP | ALEP |
|---|---|---|---|---|---|---|---|
| | | | | CEP ALEP | RTEP | | RTEP |
| $F22$ | 4.9190e−09 | FEP ALEP | FEP CEP ALEP | FEP ALEP | EP-SMS DGEP RTEP | DGEP | EP-SMS DGEP RTEP |
| $F23$ | 3.2662e−06 | FEP CEP ALEP | FEP CEP ALEP | FEP CEP ALEP | EP-SMS DGEP RTEP | EP-SMS DGEP RTEP | EP-SMS DGEP RTEP |

* No algorithm is significantly different from others for solving this benchmark function

impact [Eq. (4)] and distinctive-impact factors [Eq. (5)]. The majority-impact factor helps to speed up the EP-SMS optimization process, and the distinctive-impact factor helps increase the population diversity.

(3) EP-SMS is robust enough to solve multimodal functions, which indicates that its population is continuous diversity during the solution. The adaptive strategy of the distinctive-impact factor [Eq. (6)] helps maintain the diversity of the population in different instances of multimodal functions.

EP-SMS performs weaker than DGEP in functions $F21$–$F22$, which are low-dimensional multimodal functions with many valleys and a few minima. Since the adaptive strategy always needs certain iterations to make the mutation effective, the strengthen of EP-SMS is not obvious in the case of low-dimensional multimodal functions because of too few iterations.

### 4.4 Comparison of EP-SMS and other algorithms with similar components

The EP-SMS simulated-conformist vector has some similarity with DE, Self-Adaptive DE (Brest et al. 2006), PSO and Constricted GBest PSO (Bratton et al. 2007), such as the behavior-reference factor $\Gamma_i^{(1)}$ [Eq. (3)] and majority-impact factor $\Gamma_i^{(2)}$ [Eq. (4)], where $i$ is the number of updated individual.

Table 5 shows a performance comparison between EP-SMS and other similar-strategy algorithms. Self-Adaptive DE, PSO, and DE are implemented strictly following the studies (Bratton et al. 2007; Brest et al. 2006). EP-SMS performs significantly better than Self-Adaptive DE for $F1$, $F3$, $F5$, $F12$, $F13$, and $F18$ (approximately 43 % of the functions). EP-SMS is significantly weaker than Self-Adaptive DE (Brest et al. 2006) for $F10$, $F11$, and $F21$. The difference is not significant for approximately 36 % of the tested functions. EP-SMS performs significantly better than PSO (Bratton et al. 2007) in most instances, except for $F11$ and $F16$. Moreover, EP-SMS performs slightly better than PSO for $F11$, $F16$, and $F22$. The performance difference of EP-SMS and DE is not significant for benchmark functions $F11$, $F16$, and $F21$–$F23$, whereas EP-SMS performs significantly better than DE for $F1$, $F3$, $F8$–$F10$, $F12$, $F13$, and $F18$, but weaker for $F5$. EP-SMS is significantly higher performance than constricted Gbest PSO (Bratton et al. 2007) for $F3$, $F5$, $F8$–$F10$, $F12$–$F13$, and $F21$–$F23$, but weaker for $F1$.

EP-SMS is generally superior to the other four algorithms with similar strategies. The goal of the comparison was not to show that EP-SMS was more powerful than DEs and PSOs, but to reveal a significant difference between the simulated-conformist vector and other similar-strategy algorithms. We have not considered other recently developed evolutionary algorithms such as covariance matrix adaptation evolution strategy (CMA-ES) (Hansen 2006; Hansen et al. 2003), the stud genetic algorithm (SGA) (Khatib and Fleming 1998), and self-adaptive differential evolution with multitrajectory search (SaDE) (Zhao et al. 2011), whose mutation operators are considerably different from EP-SMS.

However, the algorithmic framework of CMA-ES is most similar of all to EP-SMS, and so we choose CMA-ES to make a comparison with EP-SMS on the benchmark functions (Yao et al. 1999) proposed in 1998. Table 5 shows that EP-SMS performance is superior to CMA-ES (Hansen et al. 2003) in solving nine functions ($F1$, $F8$–$F10$, $F12$, $F13$, $F21$–$F23$), inferior in solving three ($F3$, $F5$, $F11$), and not significantly different for two ($F16$, $F19$). We also compared CMA-ES (Hansen et al. 2003) and EP-SMS on the benchmark functions of CEC 2013 (Liang et al. 2013), whose results are shown in Table 6. EP-SMS is superior to CMA-ES in solving seventeen functions ($F1$, $F5$, $F8$–$F9$, $F11$–$F15$, $F17$–$F24$, $F26$, $F28$), yet inferior in eight ($F2$–$F3$, $F6$–$F7$, $F10$, $F16$, $F27$), and not significantly different in solving three ($F20$, $F21$, $F25$) (Liang et al. 2013).

## 5 Conclusion

We proposed a simulated-conformist mutation strategy to improve the robustness of the evolutionary programming algorithm for different types of numerical optimization problems. We designed a simulated-conformist vector for updating variation vector in the mutation operator of EP. The simulated-conformist vector is constructed by combining behavior-reference, majority-impact, and distinctive-impact factors, each simulating one aspect of a psychological model of conformity behavior. We verified experimentally that the behavior-reference factor helps the proposed EP-SMS algorithm perform robustly in solving different types of functions, that the majority-impact factor accelerates the convergence of EP-SMS, and that the adaptive distinctive-impact factor averts EP-SMS's premature termination of optimization. These factors are combined as a direct summation to calculate a variation vector in the mutation of the proposed EP-SMS. Compared with classical and recently published EPs, the proposed EP-SMS performs better for most benchmark functions in terms of the means and standard deviations of minimized fitness value.

We thus conclude that the proposed EP-SMS has three strengths in the application of engineering optimization.

(1) EP-SMS performance is stable and suitable for solving different types of optimization functions whose properties are usually unknown in real-world applications.
(2) The improvement is likely to be made in the framework of classical EPs. Therefore, engineers and programmers who use CEP, FEP, and ALEP can upgrade the function of EP algorithms at their own conveniences by revising the mutation procedure.
(3) The proposed simulated-conformist vector contains three factors, respectively, impacting three properties of EP-SMS. Therefore, users of EP-SMS can adjust the weights of these factors to control the algorithmic capacity according to the specific requirements in their applications.

Future research of EP-SMS should focus on optimal adjustment of the three factors in the simulated-conformist vector. Although its high performance was verified, EP-SMS is weak in the solution of some functions, such as $F7$ and $F11$ of the 1998 benchmark functions. Direct summation of the three factors cannot be guaranteed to be the optimal combinatorial form for every problem. Different combinations of these factors need to be tested in different optimization problems. We suggest that three weight parameters should be added to the summation for three mutation factors. The parameters can be changed adaptively based on the reflection of the problem properties.

**Acknowledgements** This work is supported by National Natural Science Foundation of China (61370102), Guangdong Natural Science Funds for Distinguished Young Scholar (2014A030306050), the Fundamental Research Funds for the Central Universities, SCUT (2015PT022), Guangdong High-Level Personnel of Special Support Program (2014TQ01X664), the Guangdong High-Level University Project Green Technologies for Marine Industries, the Science and Technology Planning Project of Guangdong Province (2013B011304002), and the Project of Educational Commission of Guangdong Province, China (2015KGJHZ014). The authors thank Mr. Changjian Xu for his help with the experiment.

**Compliance with ethical standards**

**Conflict of interest** All authors declare that they have no conflict of interest.

## References

Alam MS, Islam MM, Yao X, Murase K (2011) Recurring two-stage evolutionary programming: a novel approach for numeric optimization. IEEE Tran Syst Man Cybern Part B Cybern 41(5):1352–1365

Alam MS, Islam MM, Yao X, Murase K (2012) Diversity guided evolutionary programming: a novel approach for continuous optimization. Appl Soft Comput 12(6):1693–1707

Alipouri Y, Poshtan J, Alipouri Y, Alipour MR (2012) Momentum coefficient for promoting accuracy and convergence speed of evolutionary programming. Appl Soft Comput 12(6):1765–1786

Anik M, Alam T, Ahmed S, Noman ASM, Rakibul Islam KM (2013) A dual mutation strategy embedded evolutionary programming for continuous optimization. In: World Congress on Nature and Biologically Inspired Computing (NABIC). IEEE, pp 84–91

Asch SE (1956) Studies of independence and conformity: I. A minority of one against a unanimous majority. Psychol Monogr Gen Appl 70(9):1–70

Bäck T, Schwefel H-P (1993) An overview of evolutionary algorithms for parameter optimization. Evol Comput 1(1):1–23

Bratton D, Kennedy J (2007) Defining a standard for particle swarm optimization. In: Swarm intelligence symposium. IEEE, pp 120–127

Brest J, Greiner S, Boskovic B, Mernik M, Zumer V (2006) Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems. IEEE Trans Evol Comput 10(6):646–657

Chellapilla K (1998) Combining mutation operators in evolutionary programming. IEEE Trans Evol Comput 2(3):91–96

Chen G, Low CP, Yang Z (2009) Preserving and exploiting genetic diversity in evolutionary programming algorithms. IEEE Trans Evol Comput 13(3):661–673

Chung CY, Liang CH, Wong KP, Duan XZ (2010) Hybrid algorithm of differential evolution and evolutionary programming for optimal reactive power flow. IET Gener Transm Distrib 4(1):84–93

Dong H, He J, Huang H, Hou W (2007) Evolutionary programming using a mixed mutation strategy. Inf Sci 177(1):312–327

Dong H, Dong Y, Zhou C, Yin G, Hou W (2009) A fuzzy clustering algorithm based on evolutionary programming. Expert Syst Appl 36(9):11792–11800

Duo H, Sasaki H, Nagata T, Fujita H (1999) A solution for unit commitment using lagrangian relaxation combined with evolutionary programming. Electr Power Syst Res 51(1):71–77

El-Sharkh MY, El-Keib AA, Chen H (2003) A fuzzy evolutionary programming-based solution methodology for security-constrained generation maintenance scheduling. Electr Power Syst Res 67(1):67–72

Fogel DB (1991) System identification through simulated evolution: a machine learning approach to modeling. Ginn Press, Bratislava

Fogel DB (1993) Applying evolutionary programming to selected traveling salesman problems. Cybern Syst 24(1):27–36

Fogel D (2009) Artificial intelligence through simulated evolution. Wiley-IEEE Press, London

Fogel DB (1992) Evolving artificial intelligence. PhD dissertation, La Jolla, CA, USA. UMI order no. GAX93-03240

Gämperle R, Müller SD, Koumoutsakos P (2002) A parameter study for differential evolution. Adv Intell Syst Fuzzy Syst Evol Comput 10:293–298

Hansen N (2006) The CMA evolution strategy: a comparing review. In: Lozano J, Larranaga P, Inza I, Bengoetxea E (eds) Towards a new evolutionary computation. Advances on Estimation of Distribution Algorithms. Springer, Berlin, pp 75–102

Hansen N, Müller SD, Koumoutsakos P (2003) Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol Comput 11(1):1–18

Hedar A-R, Fukushima M (2006) Directed evolutionary programming: towards an improved performance of evolutionary programming. In: IEEE Congress on Evolutionary Computation, 2006. CEC 2006. IEEE, pp 1521–1528

Hershkovitz S, Baltianski S, Tsur Y (2011) Harnessing evolutionary programming for impedance spectroscopy analysis: a case study of mixed ionic-electronic conductors. Solid State Ion 188(1):104–109

Hong L, Drake JH, Özcan E (2014) A step size based self-adaptive mutation operator for evolutionary programming. In: Proceedings of the 2014 conference companion on genetic and evolutionary computation companion. ACM, pp 1381–1388

Jung SH (2003) Queen-bee evolution for genetic algorithms. Electron Lett 39(6):575–576

Khatib W, Fleming PJ (1998) The Stud GA: a mini revolution? In: Parallel problem solving from nature-PPSN V. Springer, pp 683–691

Lee CY, Yao X (2004) Evolutionary programming using mutations based on the Lévy probability distribution. IEEE Trans Evol Comput 8(1):1–13

Liang K-H, Yao X, Newton CS (2001) Adapting self-adaptive parameters in evolutionary algorithms. Appl Intell 15(3):171–180

Liang JJ, Kai Qin A, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10(3):281–295

Liang JJ, Qu BY, Suganthan PN, Hernández-Dıaz AG (2013) Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. Zhengzhou University, Zhengzhou, China and Nanyang Technological University, Singapore, Technical report, Computational Intelligence Laboratory 201212

Mallipeddi R, Mallipeddi S, Suganthan PN (2010) Ensemble strategies with adaptive evolutionary programming. Inf Sci 180(9):1571–1581

Nguyen QH, Ong Y-S, Lim MH (2009) A probabilistic memetic framework. IEEE Trans Evol Comput 13(3):604–623

Rajan A, Christober C (2011) Hydro-thermal unit commitment problem using simulated annealing embedded evolutionary programming approach. Int J Electr Power Energy Syst 33(4):939–946

Ratnaweera A, Halgamuge SK, Watson HC (2004) Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Trans Evol Comput 8(3):240–255

Rozenberg G, Thomas B, Kok JN (2011) Handbook of natural computing. Springer, Berlin

Schwefel H-PP (1993) Evolution and optimum seeking: the sixth generation. Wiley, London

Sinha N, Chakrabarti R, Chattopadhyay PK (2003) Evolutionary programming techniques for economic load dispatch. IEEE Trans Evol Comput 7(1):83–94

Tan Q, He Q, Zhao W, Shi Z, Lee ES (2011) An improved fcmbp fuzzy clustering method based on evolutionary programming. Compu Math Appl 61(4):1129–1144

Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82

Yao X, Liu Y (1998) Scaling up evolutionary programming algorithms. Evolutionary programming VII. Proc. of the seventh annual conference on evolutionary programming (EP98), Lecture Notes in Computer Science. Springer, Berlin, pp 103–112

Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102

Zhang H, Lu J (2008) Adaptive evolutionary programming based on reinforcement learning. Inf Sci 178(4):971–984

Zhao X, Gao X-S, Ze-Chun H (2007) Evolutionary programming based on non-uniform mutation. Appl Math Comput 192(1):1–11

Zhao S-Z, Suganthan PN, Das S (2011) Self-adaptive differential evolution with multi-trajectory search for large-scale optimization. Soft Comput 15(11):2175–2185