A Self-adaptive Mixed Distribution Based Uni-variate Estimation of Distribution Algorithm for Large Scale Global Optimization

Yu Wang and Bin Li

Abstract. Large scale global optimization (LSGO), which is highly needed for many scientific and engineering applications, is a very important and very difficult task in optimization domain. Various algorithms have been proposed to tackle this challenging problem, but the use of estimation of distribution algorithms (EDAs) to it is rare. This chapter aims at investigating the behavior and performances of uni-variate EDAs mixed with different kernel probability densities via fitness landscape analysis. Based on the analysis, a self-adaptive uni-variate EDA with mixed kernels (MUEDA) is proposed. To assess the effectiveness and efficiency of MUEDA, function optimization tasks with dimension scaling from 30 to 1500 are adopted. Compared to the recently published LSGO algorithms, MUEDA shows excellent convergence speed, final solution quality and dimensional scalability.

1 Introduction

Considered as a kind of classical yet extremely difficult task, large scale global optimization (LSGO) has attracted more and more research interest in recent years [21, 31]. LSGO problems have numerous scientific and engineering applications, such as designing large scale electronic systems, scheduling problems with large number of resources, vehicle routing in large scale traffic networks, gene detection in bioinformatics, etc. Therefore, effective LSGO algorithms are in high demand.

Inherently, the nonlinear characteristics of the practical applications usually include discontinuous prohibited zones, ramp rate limits, and nonsmooth or convex

Yu Wang · Bin Li
Nature Inspired Computation and Application Laboratory (NICAL),
University of Science and Technology of China
e-mail: wyustc@mail.ustc.edu.cn, binli@ustc.edu.cn

R. Chiong (Ed.): Nature-Inspired Algorithms for Optimisation, SCI 193, pp. 171–198.
springerlink.com

cost functions. Historically, a number of algorithms, including both mathematical and evolutionary algorithms, have been proposed to handle LSGO problems [5, 10, 15, 17, 23, 26, 32, 33, 36, 37, 38, 42, 43]. Various evolutionary algorithms (EAs) have been developed, in which significant progress has been observed [20] compared to the mathematical algorithms. The major advantages of these EAs over other classical methods can be summarized as: 1) prior knowledge of the search problem is not necessary for EAs, while for mathematical approaches the highly nonlinear characteristic of the problem must be approximated beforehand; 2) they work with a population of candidate solutions and can handle LSGO problems automatically through a single run. However, almost all of these approaches inevitably suffer from the "curse of dimensionality", which means poor performances on LSGO problems.

Without loss of generality, LSGO problems considered in this chapter can be stated as follows:

$$\begin{aligned} &\text{minimize } F(\mathbf{x}) = f(x_1, x_2, ..., x_D) \\ &\text{subject to } \mathbf{x} \in X, \end{aligned} \tag{1}$$

where $X \subset R^D$ denotes the decision space with D dimensions; $\mathbf{x} = \{x_1, x_2, ..., x_D\} \in R^D$ is the decision variable vector; $f : X \rightarrow R$ stands for a real-valued continuous objective function for mapping from D dimensional space to 1 dimensional fitness value $F(\mathbf{x})$. The dimensions of LSGO problems considered in this chapter are more than 100. Hence, the purpose of the approaches is to search for the minimized solution in such a large dimensional space. If X is a closed and connected region in R^D, we call eq. (1) continuous LSGO.

In the previous works on LSGO, developing more effective operators for EAs has attracted much research attention. The successful implementations consist of self-adapting strategies for parameter setting, modification of the classical EA operators, etc. The reason of making these modifications is that the classical operators are usually developed for low-dimensional task and they lose their efficiency for high-dimensional tasks. Their performances on LSGO problems cannot be measured effectively [31]. Recently, this field has attracted increased research attention and the typical approaches include population reduction for differential evolution (DE) [5], Dynamic multi-swarm PSO [43], and estimation of distribution algorithm (EDA) with mixed sampling operator [33]. For these approaches, the LSGO problems are optimized as an entire body, which means no divide-and-conquer methods are used. Actually, the implementation of specific operators is attributed to strengthening the algorithm's capability for higher dimensional tasks.

The classical EDAs have been proven to be effective on most classical test functions with less than 100 dimensions [11]. However, EDAs also suffer from the "curse of dimensionality", which implies that their performances deteriorate quickly as the search space increases in dimensions [33]. In this chapter, the difficulties associated with EDAs in solving LSGO problems will be discussed. Then, a heavy tail distribution based sampling (also called mutation in the evolutionary computation domain) operator will be analyzed, and introduced into a Gaussian based EDA to improve its performance. Compared to classical Gaussian distribution based operators, the heavy tail distribution based operators have demonstrated better exploration and faster convergence speed on most test problems. Some typical examples include fast evolutionary programming (FEP) [40], fast evolution strategy (FES) [41], fast simulated annealing (FSA) [30], evolutionary programming using Lévy mutation (LEP) [13] and estimation of distribution algorithm with heavy tail distribution based sampling (LSEDA-gl) [33]. Due to the success of the above algorithms, the heavy tail distribution based sampling operator is regarded as a promising EA technique to tackle some difficult problems. In this chapter, the evovabilities of different sampling operators are investigated via a technique called fitness landscape portrait (FLP). Based on this analysis, a self-adaptive mixed model uni-variate EDA (MUEDA) is proposed. In order to evaluate the performance of MUEDA, it is tested on typical function optimization problems with dimensionality scaling from 30 to 1500.

The rest of this chapter is organized as follows: In section 2, the principle and a brief review of EDAs are provided. The mathematical characteristics of Gaussian and heavy tail distributions are analyzed. Then, the main contributions of this work are presented. In section 3, a general FLP analysis is carried out to investigate the evolvability of different sampling operators in the low-dimension problems. After that, a self-adaptive uni-variate EDA with mixed kernels is proposed. In section 4, discussion of the experimental studies with dimensions ranging from 30 to 1500 is presented. Following which, the scalability and efficiency of MUEDA are presented. In section 5, several general conclusions are drawn and emphasized. In section 6, the future research directions of this area are outlined.

2 Related Work

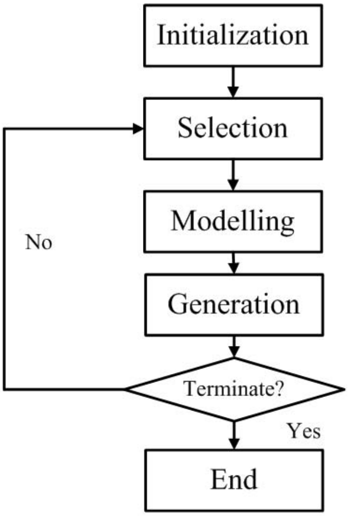
2.1 Estimation of Distribution Algorithms

The notion of modeling the search space was first proposed in the statistics and machine learning domain. Recently, many works within the Evolutionary Computation community have employed probabilistic models to describe the solution space [11]. These methods have come to be known as EDAs. It has been reported in various works that EDAs have been applied with significant success to many numerical and combination optimization problems in the past few years. Optimization by EDAs can be summarized into two major steps:

- Model the promising area of solution space of the optimization problem by learning from the superior solutions found thus far;
- Generate the population (i.e., offspring) for the next generation by sampling based on the estimated probabilistic model and then, replace the old population (i.e., parents) partially or fully.

These two steps can be regarded as a population-based version of the classical generate-and-test method [39]. As is shown in Fig. 1, there are no classical crossover or mutation operators in EDAs in contrast to traditional GA. The main operators of EDAs are as follows: selection involves selecting good solutions from the entire

Fig. 1 A general flowchart of EDA



population by truncation or league strategy; modeling involves building the probabilistic model to simulate the landscape of the problem; and generation is to sample the new population based on the probabilistic model. The evolution dynamics of EDAs depend on the distribution of the population directly. Therefore, the major advantage of EDAs is that they can explicitly extract useful structural information to efficiently generate new individuals, which results in faster convergence speed compared to GA [33].

In the EDA domain, modeling the structure of the optimization problem accurately has recently been an area of great concern. To overcome the disadvantages of limited learning ability of uni-variate EDAs, such as population-based incremental learning algorithm (PBIL) [25], stochastic hill climbing by vectors of normal distributions (SHCLVND) [24] and continuous uni-variate marginal distribution algorithm (UMDAc) [12], EDAs whose dependencies are considered in terms of pairwire or multiwire when building probabilistic model have been proposed. Some of the more successful approaches are mutual information maximizing input clustering (MIMIC) [12], estimation of Gaussian networks algorithm by edge exclusion (EGNAee) [12], estimation of Gaussian networks algorithm by BGe metric (EGNA$_{BGe}$), clustering and estimation of Gaussian distribution algorithm (CEGDA) [16], etc. The learning ability of the EDAs is always considered as the major indicator of the performance for the above-mentioned algorithms. However, the fundamental task of EDAs is to search for the global optimum of the given optimization problem, rather than to simply model the structure of the optimization accurately. Furthermore, the computational cost of constructing a complex model is too expensive for LSGO problems, whose dimensions are usually more than 100. For example, the computational cost of generating the covariance matrix for iterated density estimation algorithm (IDEA) [1, 2, 4, 7] increases exponentially. The reported studies on extending EDAs to LSGO domain are scarce so far. The emphasis of heavy tail based EDA in this work is therefore on extending EDAs to the LSGO domain.

2.2 Mathematical Characteristics of Heavy Tail Distribution

In the above reviewed EDA works, Gaussian probabilistic distribution has been widely adopted in continuous optimization. In this work, a low proportion of heavy tail stable distribution is incorporated in order to strengthen the search ability of EDAs for LSGO problems.

> **Definition:** Consider a process represented by a set Y_i of identically distributed random variables. If the sum of these random variables has the same probability distribution as individual random variables, then this process is called *stable* [13].

The Gaussian process is a typical example of a stable process with finite second moment, which would lead to a characteristic scale and the Gaussian behavior for the probability density through the central limit theorem [13]. The probability density of Gaussian distribution with mean 0 and variance σ is defined as follows:

$$N(0,\sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{x^2}{2\sigma^2}} \tag{2}$$

Different to Gaussian probability distribution whose variance can be denoted as a finite scalar, a class of probability distributions with an infinite second moment that also yields a stable process were discovered by P. Lévy in the 1930s [14]. The formal representation for such a class of probability can be expressed as follows [6, 14]:

$$\text{Ł}_{\alpha,\gamma}y = \frac{1}{\pi}\int_0^{\infty} e^{-\gamma q^{\alpha}} \cos(qy)dq \qquad y \in \mathbf{R}, \tag{3}$$

where γ is the scaling factor satisfying $\gamma > 0$, and α controls the shape of the distribution, requiring $0 < \alpha < 2$. More analytic details about the Lévy distribution are available in [6, 13, 14]. Although the analytic form of the integral is still unknown for general α, the shapes generated by Lévy distributions with different α values are known: the length of the tail is inversely proportional to the value of α. The Cauchy probability distribution adopted in FEP, is a special case of the Lévy probability distribution with $\alpha = 1$. For the limit of $\alpha = 2$, the distribution is reduced to the classical Gaussian distribution which is not included in Lévy probability distribution class. The Cauchy density with median 0 and upper quartile τ can be denoted as follows:

$$C(0,\tau) = \frac{1}{\pi}\frac{\tau^2}{x^2+\tau^2} \tag{4}$$

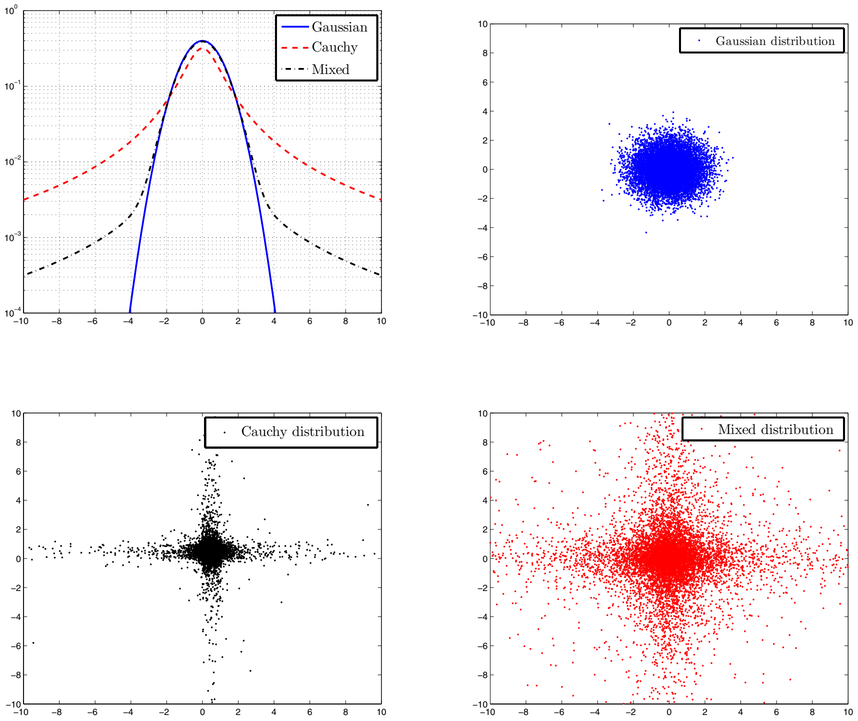


Fig. 2 Comparison among Gaussian, Cauchy and Mixed distributions in terms of the density function (up left) and the distributions of 10000 sampled points

The comparison between Gaussian and Cauchy probability density is shown in Fig. 2. It is apparent that the characteristic of infinite second moment in Cauchy probability provides a much wider distribution.

Since it is rather difficult to generate random numbers under different Lévy probability distributions except Cauchy distribution [13], several works adopt a mixed distribution, which combines the Gaussian distribution with Cauchy distribution by a suitable proportion to simulate the desired distribution. Some successful examples are [27] and [33]. These special mixed sampling density functions can be expressed as:

$$f_M = (1-\eta)\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}} + \eta\frac{1}{\pi}\frac{1}{x^2+1} \tag{5}$$

where η stands for the mixed proportion that can be tuned to any scalar between 0 and 1. In this chapter, $\eta = 0.1$ is used for analysis without statement. After investigating the mixed distribution in [13, 27, 33], it was observed that $\eta = 0.1$ favors more problems than any other value of η. For heavy distribution based sampling operator, the sampling distribution in classical uni-variate EDA ($N(\mu,\sigma) = \mu + \sigma N(0,1)$) is replaced by:

$$N_M(\mu,\sigma) = \mu + \sigma(1-\eta)N(0,1) + \sigma\eta C(0,1) \tag{6}$$

The density of 10000 sampled points by Gaussian, Cauchy and Mixed probability distribution are compared in Fig. 2. In contrast to the widest distribution sampled by Cauchy, it can be observed that the shape of a 2-D Gaussian distribution is like that of a sphere with the density increasing from the periphery towards the core. Similar observation can be made for mixed distribution but the sampling region is much wider. The mixed distribution is expected to achieve a more reasonable balance between exploration and exploitation theoretically.

2.3 Analysis of Heavy Tail Distribution in Evolutionary Computation

2.3.1 Fast Evolutionary Programming

In 1996, Yao X [40] proposed an important EP version named FEP, which replaces the Gaussian mutation in classical EP (CEP) with the Cauchy mutation. This change leads the individuals to make a much longer jump, which results in a significantly faster convergence speed when the global optima is far from the initial point. It was shown that the FEP outperforms CEP in terms of convergence speed in most test functions and the accuracy of the result in the multimodal functions [40].

Due to the excellent performance of FEP, [35] and [40] investigated the differences of expected length between Cauchy mutation and Gaussian mutation. The existing analysis methods of different probability operators focus on the properties of the operators themselves (i.e., properties related to the operators closely only). The expected length of Gaussian mutation jump with $\sigma = 1$ is calculated as follows:

$$E_G(x) = 2\int_0^{+\infty} x\frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}dx = \frac{2}{\sqrt{2\pi}} = 0.80, \tag{7}$$

and the expected length of Cauchy mutation jump is calculated as:

$$E_C(x) = 2\int_0^{+\infty} x\frac{1}{\pi}\frac{1}{x^2+1}dx = +\infty \tag{8}$$

Under the 1-D space, it is obvious that the Cauchy probability operator extends the expected sampling region to an infinite area. Therefore, a general conclusion has been made in [40]: Gaussian mutation is much more localized than Cauchy mutation. However, benefiting from the proportion of Cauchy probability incorporated into it, the mixed operator ($E_M(x) = +\infty$) also exhibits the ability of sampling population widely. There is no remarkable difference between mixed sampling operator and Cauchy sampling operator in expected sampling length theoretically. Thus, in the following section, a much more effective analysis tool FLP is adopted to distinguish the differences between different operators.

2.3.2 Heavy Tail Distribution Analysis

An important work which tries to answer the question: *when do the important tail distributions help?* is presented in [9]. The most important contributions of this work come from two hypotheses and their proofs, which can be summarized as: although the heavy tail distributions have stronger ability of maintaining diversity, they inevitably get lost in huge dimensional space, which will be proven to be worth discussing in the following section. The experimental analysis in [9] was carried out to study the behaviors of ES on Rastrigin function optimization with different distributions based mutation operators. Obviously, the viewpoint of the first hypothesis that heavy tail distributions are good at maintaining diversity is absolutely true. In our work, however, the heavy tail distributions are also proven to work in large dimension problems empirically, which is remarkably inconsistent to the second hypothesis.

2.4 Major Contribution

Based on the above review and discussion, the contributions and differences of this chapter compared to the previous work are summarized as follows:

- The tool fitness landscape portrait (FLP) [28] was proposed over over 5 years ago. Besides the random search, it has not attracted much interest in analyzing the specific EA operators, such as crossover, mutation, etc. In our work, this effective analysis tool is implemented to analyze the expected behaviors of different kernel distribution sampling operators in low dimensional spaces. Based on the analysis, some valuable suggestions on designing appropriate sampling operator are presented theoretically.
- LSGO problems are considered as a difficult task in the optimization domain. In this work, an effective algorithm called the self-adaptive MUEDA is proposed for the large scale and complex optimization problems.

3 Fitness Landscape Analysis and Self-adaptive Mixed Probability Distribution Based Uni-Variate Estimation of Distribution Algorithm

In this section, the evolvability analysis of different kernel probability distributions is presented in terms of fitness landscape analysis on low dimensional landscapes. Based on the theoretical analysis, an effective self-adaptive heavy tail based sampling operator is proposed to strengthen the search ability of uni-variate EDAs.

3.1 Fitness Landscape Analysis

In fitness landscape analysis, the optimization problems can be expressed as a set of landscapes containing one or more optima [29, 34]. Based on the number of the optima, we also classify the landscapes into smooth or rugged problems. Evolution can thus be viewed as the movement of the population, represented by a set of points (genotypes), towards lower (fitter) areas of the landscape [28]. In order to explore the evolvability of different probability based operators, we adopt partial FLP technique to test the ability of sampling population on more promising regions by different operators on two typical landscapes.

FLP that was derived from comprehensive local view for sampling is adopted as an effective tool to analyze the evolvability of given operator on special landscapes. The metric selected for describing the evolvability of different operators is defined as follows:

$$flag(x) = \begin{cases} 1, & if\ f(x) \leq f(x_g) \\ 0, & otherwise. \end{cases} \tag{9}$$

$$E_{ev}(x_g) = \frac{\int_{-\infty}^{+\infty} f_{ope}(x) \cdot flag(x)dx}{\int_{-\infty}^{+\infty} f_{ope}(x)dx}, \tag{10}$$

where evolvability $E_{ev}(x_g)$ of solution x_g with fitness $f_{ope}(x_g)$ for the EA operator ope is directly tied to the probability of solution x_g not generating offspring of lower fitness.

Since the difficulty of searching global optimum related to the structure of the fitness landscape closely is now clear, two typical fitness landscapes (i.e. sphere landscape and rugged landscape), which include most of the existing landscapes, are chosen to evaluate the evolvability of Gaussian, Cauchy and mixed probability sampling operators.

3.1.1 Smooth Landscape

The evolvability of sampling operators with $\sigma = 1$ is tested on the region intercepted by $[-10, 10]$ for the sphere landscape shown in Fig. 3 (left). The evolvability metric curve line which is generated for each mean value moving from -10 to 10 by eq.(10) is shown in Fig. 3 (right). The sphere landscape is especially adopted to evaluate the evolvability of different sampling operators on smooth problems.

It can be observed from the evolvability curve that the Gaussian sampling operator provides the best evolvability for the whole region. Accordingly, the evolvability of Cauchy sampling operator decreases sharply while the global optimum is still far away, and this delays the convergence speed significantly. This may be the reason for the unsolved question in [9, 35] that Gaussian leads to a faster convergence speed than Cauchy for Sphere problem. Benefiting from the property of Gaussian distribution, the mixed sampling operator is equipped with similar evolvability as Gaussian sampling operator, which is remarkably better than Cauchy sampling operator by itself.

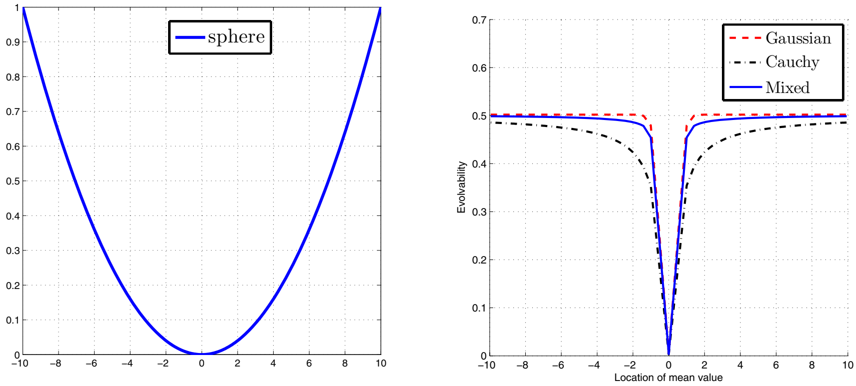


Fig. 3 Sphere landscape on the left and evolvability curve line graph on the right

3.1.2 Nonsmooth Landscape

It has been known that evolvability on rugged landscape mainly depends on the capability of escaping from the local optima. Consider the rugged landscape ($f(x) = \cos(x) - \frac{x}{20}$) shown in Fig. 4 (left). We intercept one local optimal basin $[0, 2\pi]$ for analysis. Curve lines of evolvability generated by different sampling operators with mean value moving from 0 to 2π are also generated by eq. (10).

It is apparent that the evolvability of Gaussian sampling operator worsens sharply near the local optimum, which means that its evolvability shrinks quickly towards the local optimum. By comparison, Cauchy sampling operator demonstrates the best ability of escaping from the local optima. Therefore, the incorporation of low proportional Cauchy distribution highly improves the evolvability of mixed sampling operator. Therefore, the mixed sampling operator keeps a steady high probability of escaping from the local optima and thus, maintains the evolvability effectively.

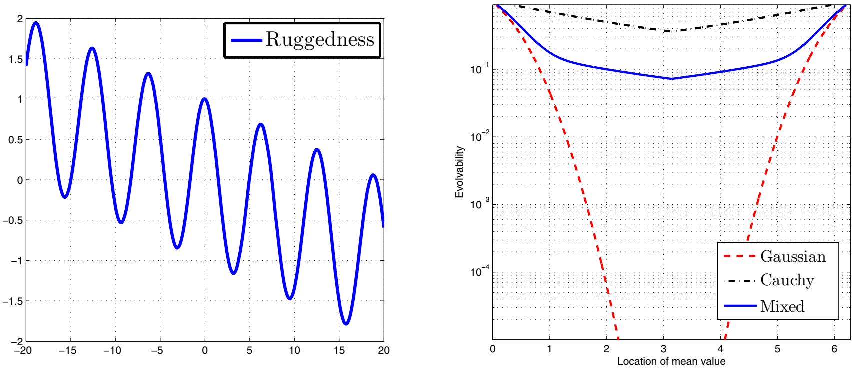


Fig. 4 Sphere landscape on the left and evolvability curve line graph on the right

Based on the above fitness landscape analysis, the mixed operator shows superior ability in both moving quickly towards the global optimum on smooth landscape and maintaining evolvability on rugged landscape globally. Compared to the classical Gaussian sampling operator, the sampling area of the mixed sampling operator exceeds the Gaussian model. In such case, the balance between learning and optimization is well handled. Therefore, it seems to be a promising way to develop more robust EDA by using mixed sampling operator.

3.2 Algorithm

Compared with the mutation operators of GA, ES and EP that mutate the individuals, the sampling operator of EDA mutates the distribution of the whole population in each generation. To strengthen the global search ability, a mixed Gaussian and Lévy distribution is adopted here, which is similar to [33].

Based on the above analysis, a new self-adaptive uni-variate EDA is proposed here. In order to reduce the complexity of conducting the probabilistic model, the uni-variate EDA whose variables are considered independently is adopted in our algorithm. Similar to UMDAc [12], the joint probabilistic distribution over a set of random variables $x = \{x_i\}$ where i = 1, 2... D for D dimensional space is defined as follows:

$$P(\mathbf{x}) = \prod_{i=1}^{D} P(x_i). \tag{11}$$

The probability distribution used to model each variable $P(x_i)$ is a single mixed Gaussian and Cauchy distribution. In contrast to iterated density estimation algorithm (IDEA) [1] developed by Bosman which requires computing all elements of covariance matrix to adapt an arbitrary Gaussian, MUEDA abandons adapting the non-diagonal elements in covariance matrix, which remarkably reduces the computational cost for LSGO problems. The updated rule for $P(x)$ is defined as follows:

$$P_{t+1}(\mathbf{x}) = (1-\theta)\cdot P_t(\mathbf{x}) + \theta \cdot P_t'(\mathbf{x}) \tag{12}$$

where $P_t'(\mathbf{x})$ is exactly the estimated joint probability distribution for the superior solutions of τ generation under the classical Gaussian distribution and θ stands for the learning intensity coefficient. Hence, the candidates for the $t+1$ generation are produced based on $P_t(\mathbf{x})$. Similar to UMDAc, the model for UMDAc is built by the current population only, which means that the learning coefficient θ is 1.

The details of mixed distribution and the strategy for generating each candidate are shown as:

$$randnum = rand;$$

$$P_m = \frac{10 \cdot randnum}{D}$$

$$N_L = \begin{cases} 0.9 \cdot N(0,1) + 0.1 \cdot C, & if\, D < 100 \\ (1-P_m)\cdot N(0,1) + P_m \cdot C, & otherwise, \end{cases} \tag{13}$$

$$X_{ji} = \overline{X}_i + \sqrt{\delta_i} \cdot N_L, \tag{14}$$

In eq.(13), $N(0,1)$ stands for the Gaussian distribution with mean 0 and standard deviation 1; C denotes the Cauchy distribution with $t = 1$; D is the dimensional size of the problem; *randnum* is generated randomly based on uniform distribution between $(0,1)$. The probability distribution for creating each offspring on different dimensional size is self-adapted as shown in eq.(13): a smaller mixed probability P_m is adopted for a larger dimensional size. In eq.(14), the ith element of jth individual is sampled under mixed Gaussian and Lévy distribution model N_L, where $\overline{X}_i$ stands for the ith element of the mean vector. Compared with UMDAc, whose sampling operator is denoted as $X_{ji} = \overline{X}_i + \sqrt{\delta_i} \cdot N(0,1)$, the standard Gaussian probability $N(0,1)$ is replaced with the mixed Gaussian and Lévy distribution N_L.

The flow of MUEDA is described as follows:

Input:

- LSGO problem;
- a termination condition;

Output: The solution with best fitness value.
Flow of MUEDA:
Step 0) Initialization:

- **Step 0.0)** Set population size $NP = (\log(D) - 3) \times 50$, selection size $N = (\log(D) - 3.5) \times 15$ and weight vector W.
- **Step 0.1)** Randomly initialize the population X_0.
- **Step 0.2)** Set $t = 0$.

Step 1) Reproduction and update:

- **Step 1.0) Reproduction:** Sample the new candidates by specific EA operator.
- **Step 1.1)** Set $t = t + 1$.
- **Step 1.2) Selection:** Select N best individuals by truncation strategy.
- **Step 1.3) Update:** Update the model with the selected individuals (eq.(6)).

Step 2) Standard Deviation Control Strategy (STDC) Then, if termination condition is met, go to step 3, else go to step 1.
Step 3) Terminate and output the GO.

It has been observed that one crucial problem that prevents uni-variate Gaussian based EDAs from biasing the search population towards a better region is that the standard deviation of some variables often shrinks to zero quickly while the global optimum is still far away [33]. In step 2, we introduce the standard deviation control strategy (STDC) to improve the exploration ability of uni-variate Gaussian based EDA. The main idea of STDC is to estimate a common threshold of standard deviations for all variables during the optimization process to control their shrinking speeds and therefore, to control the decreasing level of diversity dynamically. In more detail, the variables with lower standard deviation values than the corresponding thresholds will be forced to set their standard deviations to the corresponding thresholds. The weighted mean of the standard deviations of all variables is used as the threshold here. The details of STDC are shown as follows:

```
Pseudo code of STDC
begin STDC
   for j=1:D
      if $\delta(j) < W(j) \times \overline{\delta}$
         $\delta(j) = W(j) \times \overline{\delta}$
         /* $\overline{\delta}$ is the mean of variances */
         /* $W$ is a weight vector */
      end if
   end for
end
```

After analyzing a great deal of W values for each dimensionality setting, we calculate the value of W in an adaptive way shown as in eq. (15) for problems of different Ds:

$$W(i) = 0.55 - e^{lg(\frac{D}{10^5})} \quad (15)$$

$$\text{for} \quad i = 1, 2, ..., D$$

The metric W is determined by D only, which means a larger W is adopted for problems with lower D. It is observed that lower W value is adopted for larger D. Eq. (15) is used because the metric W is generally hard for the inexperienced users to choose.

Some existing works on standard deviation based triggering of variance scaling have been reported, such as adaptive variance scaling [8] and standard deviation ratio [3]. In these approaches, the scaling of standard deviation is determined by the performance of the latest generation. In contrast, the STDC strategy only enlarges the standard deviations of some variables under an adaptive threshold vector. Therefore, STDC is much simpler.

4 Experimental Study

4.1 Classical Function Optimization with Low Dimensionality

The purpose of this experiment is to compare mixed distribution based sampling operator of MUEDA with the Gaussian distribution based classical UMDAc. Moreover, the existing heavy tail distribution based algorithms, including FEP and FES, are adopted to provide the comparison results. The formal definitions of the test functions are summarized in Table 1. Function 1-6 are unimodal problems. Function 7-12 are multimodal problems, whose landscapes are full of local optima. In this chapter, the initialized population are randomly generated within the bounds for all experiments.

Table 1 Classical benchmark problems to be minimized

Num	Problems	Bounds	Objective function	Global location	GO
fun1	**Sphere**	[-100,100]	$f_1(x) = \sum_{i=1}^{n} x_i^2$	$x_i = 0$	0
fun2	**Schwefel**	[-10,10]	$f_2(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	$x_i = 0$	0
fun3		[-100,100]	$f_3(x) = \sum_{i=1}^{n} (\sum_{j=1}^{i} x_j^2)$	$x_i = 0$	0
fun4		[-100,100]	$f_4(x) = \max \lvert x_i \rvert$	$x_i = 0$	0
fun5	**Rochenbrock**	[-100,100]	$f_5(y) = \sum_{i=1}^{D} (100(x_i^2 - x_{i+1})^2 + (x_i - 1)^2)$	$x_i = 1$	0
fun6	**Step**	[-30,30]	$f_6(x) = \sum_{i=1}^{n} \lvert x_i + 0.5 \rvert^2$	$x_i = -0.5$	0
fun7	**Noise**	[-1.28,1.28]	$f_7(x) = \sum_{i=1}^{n} i x_i^4 + random[0,1)$	$x_i = 0$	0
fun8	**Rastrigin**	[-5.12, 5.12]	$f_8(x) = \sum_{i=1}^{n} (x_i^2) - 10cos(2\pi x_i) + 10$	$x_i = 0$	0
fun9	**Ackley**	[-32,32]	$f_9(x) = -20 \cdot exp(-0.2\sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2})$ $+e - exp(\frac{1}{n} \sum_{i=1}^{n} cos(2\pi x_i)) + 20$	$x_i = 0$	0
fun10	**Griewangk**	[-600,600]	$f_{10}(x) = \frac{1}{4000} \sum_{i=1}^{n} x_i^2 + 1 - \prod_{i=1}^{n} cos(\frac{x_i}{\sqrt{i}})$	$x_i = 0$	0
fun11	**Penalized**	[-50, 50]	see appendix	$x_i = 1$	0
fun12	**Penalized**	[-50, 50]	see appendix	$x_i = 1$	0

Table 2 Experimental results for classical function optimization

function	MUEDA	UMDAc	FES	FEP	CEP
fun1	**2.8E-160** ± **1.5E-159**	2.0E-03 ± 4.9E-03	2.5E-04 ± 6.8E-05	5.7E-04 ± 1.3E-04	2.2E-04 ± 5.9E-04
fun2	**1.3E-124** ± **3.0E-124**	2.2E-01 ± 8.9E-01	6.0E-02 ± 9.6E-03	8.1E-03 ± 7.7E-04	2.6E-03 ± 1.7E-04
fun3	**0.0E+00** ± **0.0E+00**	1.3E+01 ± 4.6E+01	1.4E-03 ± 5.3E-04	1.6E-02 ± 1.4E-02	5.0E-02 ± 6.6E-02
fun4	**4.2E-72** ± **2.9E-71**	1.3E-03 ± 3.8E-03	5.5E-03 ± 6.5E-04	3.0E-01 ± 5.0E-01	2.0E+00 ± 1.2E+00
fun5	**9.0E-05** ± **2.2E-04**	1.7E+01 ± 5.5E+00	3.3E+01 ± 4.3E+01	5.1E+00 ± 5.9E+00	6.2E+00 ± 1.4E+01
fun6	**0.0E+00** ± **0.0E+00**	0.0E+00 ± 0.0E+00	0.0E+00 ± 0.0E+00	0.0E+00 ± 0.0E+00	5.8E+02 ± 1.1E+03
fun7	**3.0E-03** ± **5.8E-03**	1.1E-02 ± 1.8E-03	1.2E-02 ± 5.8E-03	7.6E-03 ± 2.6E-03	1.8E-03 ± 6.4E-03
fun8	2.1E+01 ± 4.7E+00	1.9E+00 ± 8.7E-03	1.6E-01 ± 3.3E-01	**4.6E-02** ± **1.2E-02**	8.9E+01 ± 2.3E+01
fun9	**4.4E-15** ± **0.0E+00**	1.9E-04 ± 9.6E-03	1.2E-02 ± 1.8E-03	1.8E-02 ± 2.1E-03	9.2E+00 ± 2.8E+00
fun10	**0.0E+00** ± **0.0E+00**	4.7E-03 ± 5.9E-03	3.7E-02 ± 5.0E-02	1.6E-02 ± 2.2E-02	8.6E-02 ± 1.2E-01
fun11	**1.6E-32** ± **0.0E+00**	1.9E-06 ± 3.2E-06	2.8E-06 ± 8.1E-07	9.2E-06 ± 3.6E-06	1.8E+00 ± 2.4E+00
fun12	**1.3E-32** ± **0.0E+00**	4.0E-05 ± 7.0E-05	4.7E-05 ± 1.5E-05	1.6E-04 ± 7.3E-05	1.4E+00 ± 3.7E+00

Statistical experimental results of 50 runs

For fair comparison, we set the parameters as in [40]. The following parameters are used in this experiment: 1) population size 100 for all algorithms; 2) maximum number of generations: 1500 for function 1, function 6, function 9, function 11 and function 12; 2000 for function 2 and function 10; 3000 for function 7, 5000 for function 3, function 4 and function 8. The statistical experimental results of 50

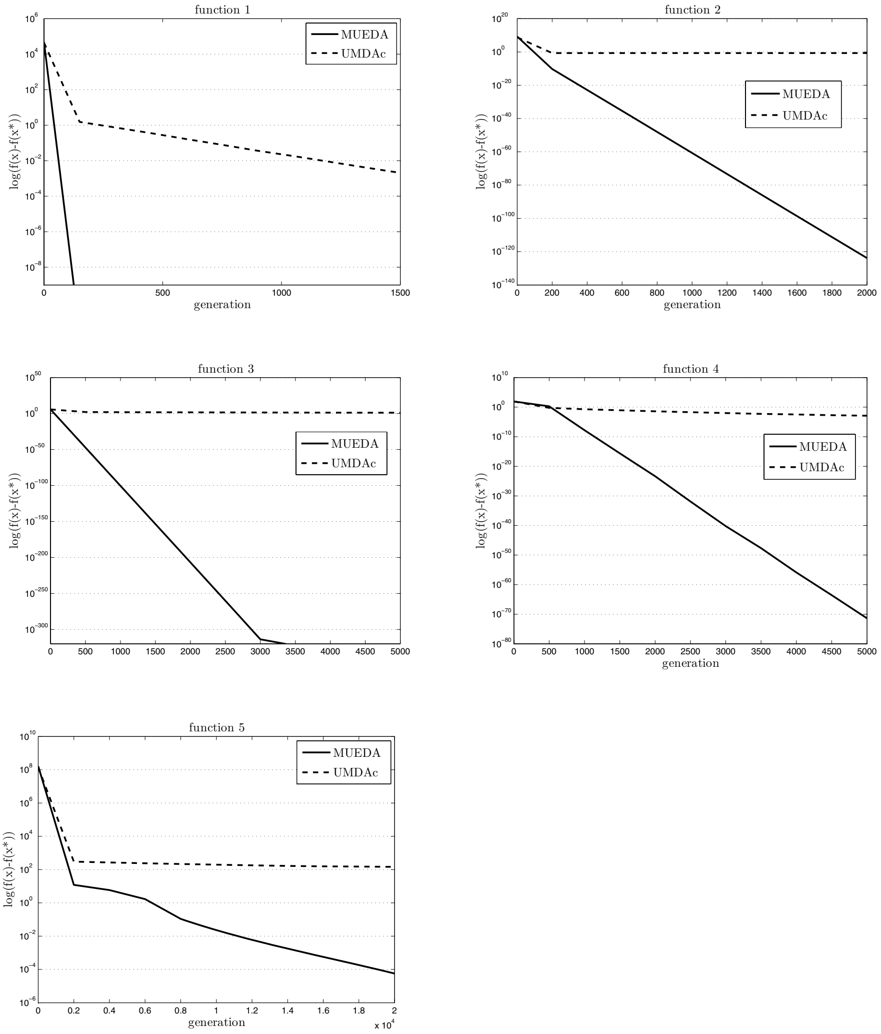


Fig. 5 Evolutionary curves of unimodal problems

independent runs are summarized in Table 2. Fig. 5 and 6 show the optimization curves for the unimodal problems and multimodal problems respectively.

Compared to UMDAc, it is apparent that MUEDA provides significantly better performance in terms of both convergence speed and accuracy of the final result for almost all of the test functions with 30 *D*. For the unimodal problems functions 1-6, MUEDA always provides the fastest convergence speed. It should be noted that, for function 5, a well-known hard test problem, the MUEDA approaches the true global optimum within the fixed number of generations, while the other algorithms are still struck at local optima after the final generation. The global search ability

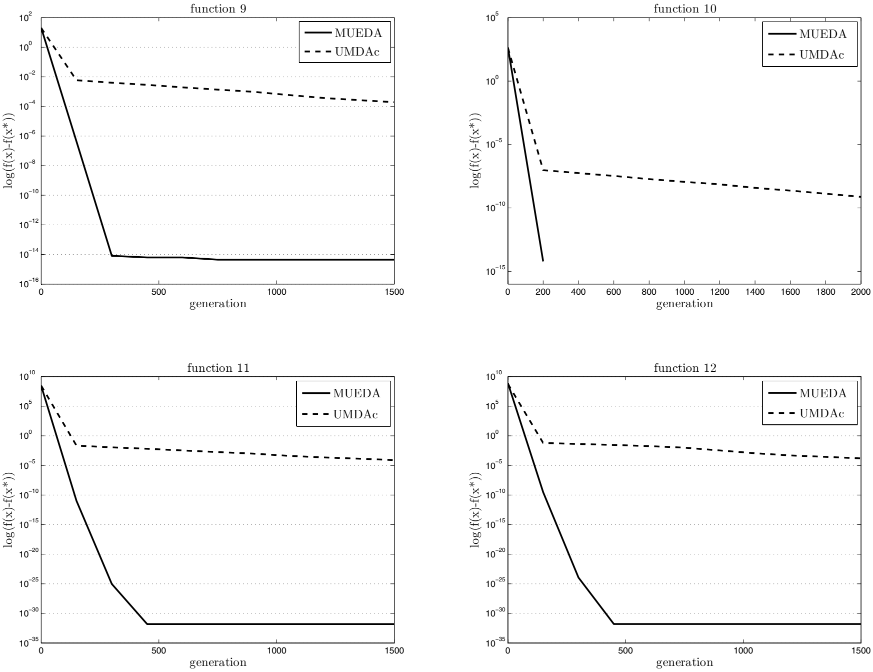


Fig. 6 Evolutionary curves of multimodal problems

of MUEDA is proven via experiments on multimodal test functions (function 7 - function 12). The accurate results (i.e. the error is lower than 10^{-8}) are achieved in all runs on function 9 - function 12. For the problem with noise incorporated function 7, MUEDA also outperforms the other approaches. For function 8, MUEDA cannot achieve satisfactory result, the reason for which will be discussed in the following section. Through this experiment, the advantages of MUEDA on both exploration and exploitation have been clearly demostrated.

4.2 Experiments on LSGO Problems

The benchmark set selected for this experiment consists of 6 test functions defined in [31]. Functions 1 - 3 are unimodal functions and Functions 4 - 6 are multimodal functions. To prevent exploitation of symmetry of the search space and of the typical zero value associated with the global optimum, the local optima of classical functions are shifted to a value different than zero and the function values of the global optima are non-zero. Without loss of generality, maximum fitness evaluation size (MFES) $500 \times D$ is adopted for function 1, 3, 5 and 6 and $5000 \times D$ for function 2 and 4. The details of standard benchmarks are defined in Table 3. The source codes for these functions are available from `http://nical.ustc.edu.cn/cec08ss.php` and `http://www3.ntu.edu.sg/home/EPNSugan/`.

Table 3 Standard benchmark problems of CEC08 to be minimized

Num	Problems	Bounds	Objective function	GO location	GO
funcec1	**Sphere**	[-100,100]	$f_1(y) = \sum_{i=1}^{D} y_i^2 + f_{bias1}$	$y_i = 0$	$f_{bias1} = -450$
funcec2	**Schwefel**	[-100,100]	$f_2(y) = \max(\lvert y_i \rvert) + f_{bias2}$	$y_i = 0$	$f_{bias2} = -450$
funcec3	**Rochenbrock**	[-100,100]	$f_3(y) = \sum_{i=1}^{D} (100(y_i^2 - y_{i+1})^2 + (y_i - 1)^2) + f_{bias3}$	$y_i = 1$	$f_{bias3} = 390$
funcec4	**Rastrigin**	[-5, 5]	$f_4(y) = \sum_{i=1}^{n} (y_i^2) - 10cos(2\pi y_i) + 10 + f_{bias4}$	$y_i = 0$	$f_{bias4} = -330$
funcec5	**Griewangk**	[-600,600]	$f_5(y) = \frac{1}{4000} \sum_{i=1}^{D} y_i^2 + 1 - \prod_{i=1}^{D} cos(\frac{y_i}{\sqrt{i}}) + f_{bias5}$	$y_i = 0$	$f_{bias5} = -180$
funcec6	**Ackley**	[-32,32]	$f_6(x) = -20 \cdot exp(-0.2\sqrt{\frac{1}{D} \sum_{i=1}^{D} y_i^2})$ $+e - exp(\frac{1}{D} \sum_{i=1}^{D} cos(2\pi y_i)) + 20 + f_{bias6}$	$y_i = 0$	$f_{bias6} = -140$

4.2.1 Comparison among Different Cauchy Proportions Based Algorithm

In order to illustrate the impacts of different heavy tail distributions, four Cauchy proportion are adopted for comparison: $\eta = 0$ (UMDAc), $\eta = 0.1$ (constant Cauchy), $\eta = 1$ (only Cauchy), and adaptive η (eq. (13) for MUEDA). For fair comparison, we choose parameters as consistently as possible. For external parameters, we set $NP = (\lvert \log(D) - 3 \rvert \times 50)$ and $N = (\lvert \log(D) - 3.5 \rvert \times 15)$. The statistical experimental results of over 50 runs are summarized in Table 4, in which the best result for each function is in boldface.

Table 4 Experimental results for 100 D cec08 function optimization

Algorithm	Metric	funcec1	funcec2	funcec3	funcec4	funcec5	funcec6
UMDAc	mean	3.81E+02	2.18E+01	2.96E+05	**2.20E+01**	3.24E+00	3.80E+00
$\eta = 0$	std	2.26E+02	3.08E+00	3.33E+05	**4.34E+00**	1.67E+00	1.14E+00
Mixed	mean	7.85E-01	1.72E+01	1.28E+07	8.03E+02	1.00E+00	3.52E+00
$\eta = 0.1$	std	5.53E-01	1.43E+00	2.01E+06	1.89E+01	6.04E-02	2.14E-01
Cauchy	mean	3.68E+05	1.52E+02	3.05E+11	1.91E+03	3.28E+03	2.13E+01
$\eta = 1$	std	2.13E+04	2.79E+00	3.75E+10	4.24E+01	1.70E+02	3.49E-02
MUEDA	mean	**6.82E-14**	**2.21E-13**	**2.89E+03**	1.04E+02	**1.28E-03**	**6.57E-12**
adaptive η	std	**2.32E-14**	**2.50E-14**	**3.69E+03**	2.49E+01	**3.63E-03**	**2.14E-11**

Statistical experimental results of 25 runs

It is apparent that Cauchy only ($\eta = 1$) distribution based algorithm fails in all problems. Therefore, excessive exploitation is not always beneficial for high dimensional problems. Generally speaking, MUEDA outperforms the other algorithms remarkably in most problems and is followed by UMDAc. For the hard task funcec 2, it is interesting to note that the result provided by MUEDA is very accurate while all of the other algorithms fail to get close to the global optima. The performances of all algorithms for Rochenbrock problem deteriorate sharply.

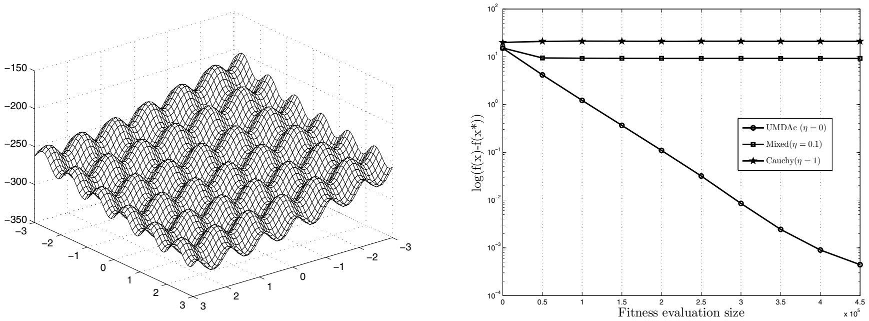


Fig. 7 Rastrigin problem: landscape on the left and convergence process comparison on the right

The 2-D landscape of Rastrigin problem and variance change process comparison are shown in Fig. 7. It is obvious that the landscape is full of local optima. The above experimental results have shown that UMDAc provides the best result for Rastrigin problem. However, the change curve of variance shows that the population of classical UMDAc just shrinks into a local optimum after a small fitness evaluation size. For this reason, the Gaussian distribution exhibits low evolvability and the final result is no longer reasonable. For the heavy tail distribution based operators, there is a high variance level throughout the search (which appears as a random search). Therefore, heavy tail distribution based operators seem more robust in these landscapes, although the final accuracy is unsatisfactory.

4.2.2 Comparison with other LSGO Evolutionary Algorithms

To benchmark MUEDA further, the comparison on larger dimensional (1000 D) problems is carried out. In some recent studies, some algorithms have reported the regular experimental results for the funcec functions. We only take the EA based algorithms into account. The algorithms are listed as follows:

- Efficient Population Utilization Strategy for Particle Swarm Optimizer (EPUS-PSO) [10]
- Unbiased Evolutionary Programming (UEP) [17]
- Self-Adaptive Differential Evolution algorithm (jDEdynNP-F) [5]
- Dynamic multi-swarm particle swarm optimizer (DMS-PSO) [43]
- Multilevel cooperative coevolution (MLCC) [38]
- Differential Evolution with Self-Adaptive cooperative co-evolution (DEwSAcc) [42]

The statistical analysis is shown in Table 5. In Table 6, the t-test results regarding $algorithm1$ vs $algorithm2$ are shown as '+', '-', 's+' and 's-' when $algorithm1$ is insignificantly better than, insignificantly worse than, significantly better than, and significantly worse than $algorithm2$ respectively. For unimodal problems, it is

Table 5 Experimental results for 1000 *D* cec08 function optimization

Algorithm	Metric	funcec1	funcec2	funcec3	funcec4	funcec5	funcec6
MLCC	mean	6.15E-01	1.09E+02	7.91E+03	**1.37E-10**	2.98E-02	2.02E-01
	std	6.58E-01	4.75E+00	1.12E+03	**3.37E-10**	2.26E-02	3.14E-01
EPUS-PSO	mean	3.46E+05	4.66E+01	3.77E+10	7.58E+03	3.09E+03	2.10E+01
	std	1.48E+04	4.00E-01	3.30E+09	1.51E+02	1.30E+02	1.62E-02
jDEdynNP-F	mean	2.92E+05	1.95E+01	7.10E+10	2.17E-04	2.53E+03	1.81E+01
	std	1.22E+04	2.25E+00	5.99E+09	4.06E-04	1.66E+02	1.81E-01
UEP	mean	7.78E+04	1.05E+02	4.43E+09	1.03E+04	7.93E+02	1.99E+01
	std	4.54E+03	7.07E+00	4.54E+08	9.94E+02	5.29E+01	1.19E-02
DEwSAcc	mean	5.68E+05	1.26E+02	2.16E+11	9.46E+03	5.19E+03	2.00E+01
	std	1.06E+05	4.48E+00	2.16E+11	7.15E+01	6.33E+02	3.33E-01
DMS-PSO	mean	**0.00E+00**	9.15E+01	2.94E+11	3.84E+03	**0.00E+00**	1.92E+01
	std	**0.00E+00**	7.14E-01	1.69E+11	1.71E+02	**0.00E+00**	4.06E-02
MUEDA	mean	3.30E-13 *II*	**9.45E-05**	**1.99E+03**	5.00E+03 *IV*	1.71E-13 *II*	**5.92E-08**
	std	2.54E-14 *II*	**9.66E-06**	**2.35E+02**	1.26E+02 *IV*	2.01E-14 *II*	**8.10E-08**

Statistical experimental results of 25 runs

Table 6 The t-test results of comparing MUEDA with the other algorithms

	funcec1	funcec2	funcec3	funcec4	funcec5	funcec6
MUEDA vs MLCC	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$
MUEDA vs EPUS-PSO	$s+$	$s+$	$s+$	$+$	$s+$	$s+$
MUEDA vs jDEdynNP-F	$s+$	$s+$	$s+$	$s-$	$s+$	$s+$
MUEDA vs UEP	$s+$	$s+$	$s+$	$s+$	$s+$	$s+$
MUEDA vs DEwSAcc	$s+$	$s+$	$s+$	$+$	$s+$	$s+$
MUEDA vs DMS-PSO	$s-$	$s+$	$s+$	$-$	$s-$	$s+$

observed that MUEDA provides accurate results. For 1000 *D* funcec 1, DMS-PSO outperforms MUEDA in terms of accuracy. For the other algorithms, high-quality results cannot be generated due to the low computational cost. Funcec 2 with high *D* is an extremely hard task because the variables are non-separable. Although the fitness landscape is very smooth, the fact that only one variable with largest absolute value contributes to fitness value makes it almost impossible to be solved by classical approaches. It is evident that only MUEDA succeeds in converging to a solution with accuracy lower than 10^0 to the true global optimum. For the other algorithms, especially cooperative coevolution based ones, the search is stuck badly, which implies that they are not effective enough to deal with the non-separate LSGO problem. These results are not surprising, because only MUEDA considers modelling the rough structure of the search space. Furthermore, the implementation of mixed distribution makes breaking the common restriction possible.

The multimodal problems become much harder as the dimensionality increases to 1000. Funcec 3, whose variables are linked to a neighboring one (i.e. Rocenbrock problem), is the hardest task. For this problem, all algorithms fail in all runs. Another difficulty of this problem is that the valley towards the global optimum is very narrow and none of the algorithms could adaptively control the shrinking speed to suit this nonseparate problem. Amongst the compared algorithms, MUEDA demonstrates the most superior performance. For funcec 4, most of the algorithms fail to obtain GO except MLCC. Funcec 5 and 6 can be solved completely by MUEDA. The dominant convergence ability of MUEDA is strongly indicated by all these results.

4.2.3 Scalability Study

In order to study the scalability of MUEDA, we compare it with cooperative coevolution based LSGO approach Cooperative coevolution differential evolution II (DECC-II) on the 1500 dimensional problems. The reasons of selecting DECC-II in this experiment are as follows: 1) the cooperative coevolution appears to be a very promising method and has becomes very popular in LSGO domain [31]; 2) Compared with the classical cooperative coevolution based algorithms FEPCC [15] and CCGA [23], DECC-II has performed better on most problems [36]. In DECC-II, the variables selected for optimization in one iteration are chosen randomly with constant size 100. The other parameters chosen for experiment are the same as [36]. The comparison of evolution process between MUEDA and DECC-II on 1500 dimensional function optimization is shown in Fig. 8, and the comparison of final error between the best solution and true global optimum is shown in Table 7.

It is apparent that for unimodal function 1 and function 2, the proposed algorithm outperforms DECC-II not only in convergence speed, but also in the accuracy of results obtained. This is especially true for function 2 whose variables are linked. Similar conclusion can be drawn for multimodal functions globally, although the results achieved by DECC-II and proposed algorithm on function 4 (i.e. shift Rastrigin) are comparable. The reasons for this result have been analyzed in the first experiment. It is observed that the proposed algorithm works well on function 5 and 6 within such low computational cost even for 1500 *D*. In summary, the proposed algorithm provides a steady and accurate performance even for problems scaling to 1500 dimensions.

4.2.4 Efficiency Study

In the traditional analysis of [18, 19], the efficiency of the search method is tested on the problems with different dimensionality setting. Efficiency is defined on number of function evaluations needed to solve the problem [18]. In order to evaluate the efficiency of MUEDA, the results for multimodal problems - Ackley function (function 9 in Table 1) and Giewangk function (function 10 in Table 1) are presented in Table 8, 9 and Fig. 9. Moreover, the effective algorithms DE and PSO are also implemented in order to provide comparison results. For fair comparison, we

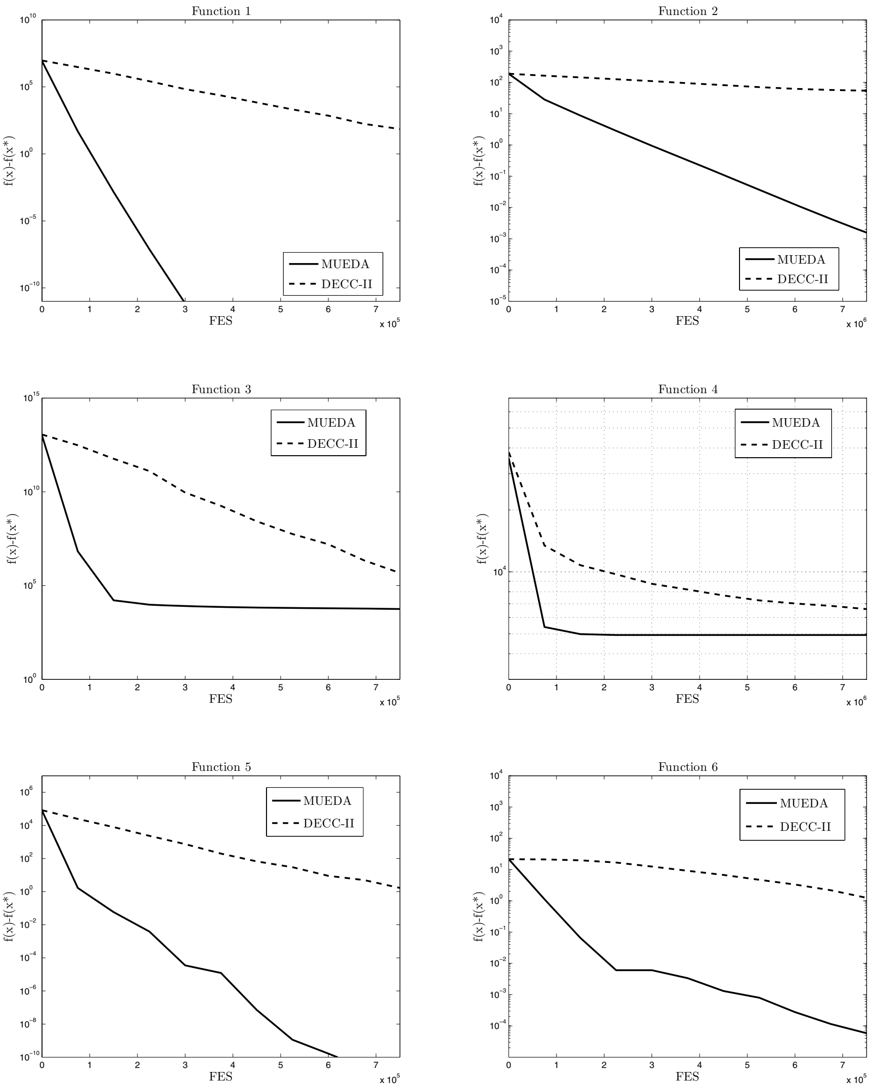


Fig. 8 Evolutionary curves of 1500 D problems

choose the same population size 200 for all algorithms and the most commonly used extensive parameters for respective algorithms.

It is apparent that the computational cost of MUEDA tends to increase linearly as the dimensionality arises. For DE and PSO, the efficiencies are much less. This is especially true when D is very large. It is evident that handling large scale optimization problems is a difficult task for evolutionary algorithms. Compared to breeder

Table 7 Experimental results for 1500 D cec08 function optimization

Algorithm	Metric	funcec1	funcec2	funcec3	funcec4	funcec5	funcec6
MUEDA	mean	1.83E-09	8.72E-05	5.83E+03	4.76E+03	1.71E-13	2.74E-05
	std	2.51E-09	2.37E-05	2.66E+03	1.41E+02	2.01E-14	2.01E-05
DECC-II	mean	2.24E+04	5.13E+01	1.44E+09	6.73E+03	7.46E+01	9.02E+00
	std	2.66E+03	3.01E+00	4.46E+08	8.60E+01	5.15E+00	3.80E-01

Statistical experimental results of 25 runs

Table 8 Efficiency test on Ackley function

Dimesionality	MUEDA	$166D+23000$	$28D\ln(D)$	DE	PSO
100	40625	39600	128945	218802	282723
200	65076	56200	148353	fail	532617
300	86008	72800	159706	fail	761572
400	100458	89400	167761	fail	fail
500	113592	106000	174009	fail	fail
600	129110	122600	179114	fail	fail
700	141982	139200	183430	fail	fail
800	159629	155800	187169	fail	fail
900	174234	172400	190467	fail	fail
1000	189410	189000	193417	fail	fail

Termination criterion is 10^{-3}.

Table 9 Efficiency test on Giewangk function

Dimesionality	MUEDA	$156D+24000$	$26D\ln(D)$	DE	PSO
100	40170	39600	119734	179030	249316
200	67432	55200	137756	fail	472276
300	92017	70800	148298	fail	fail
400	105875	86400	155778	fail	fail
500	119864	102000	161580	fail	fail
600	133825	117600	166320	fail	fail
700	142888	133200	170328	fail	fail
800	153660	148800	173800	fail	fail
900	166801	164400	176862	fail	fail
1000	180451	180000	179602	fail	fail

Termination criterion is 10^{-3}.

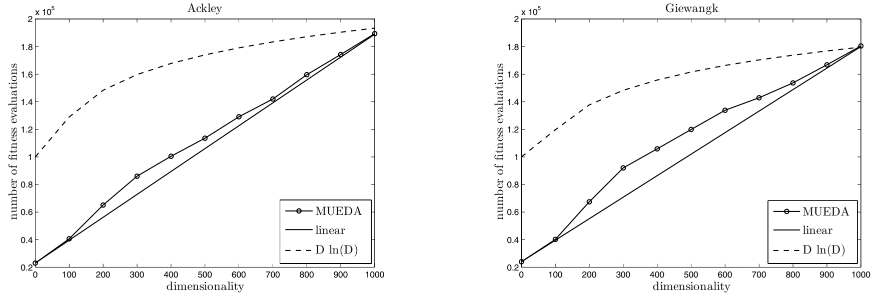


Fig. 9 Efficiency test on Ackley function on the left and Giewangk function on the right

GA scaling like $D\ln(D)$, MUEDA shows excellent efficiency, which scales linearly. Thus, the advantages of MUEDA on LSGO problems are easily observable.

5 Concluding Remarks

In this chapter, there are two major works, which can be summarized as:

- Via fitness landscape analysis, the expected behaviors and evolvability of Uni-variate EDAs with different kernel probability distributions based sampling operators are studied in low dimensional spaces. The evolvability change curve analysis reveals that mixing Gaussian with Cauchy distribution may be a promising way to strengthen the search ability.
- Based on the above analysis, a self-adaptive mixed distribution based uni-variate EDA named MUEDA is proposed for both LSGO problems. For the low dimensional problems, MUEDA provides excellent performance. Experimental evidence of large scale global function optimization is demonstrated to illustrate the merits and demerits of the proposed algorithm. Moreover, some scalability study is also carried out to evaluate MUEDA further.

In summary, this work aims at providing both expected and experimental analysis on improving the performances of uni-variate EDAs by designing a more effective sampling operator. As to the experimental results, MUEDA improves the performance of the uni-variate EDA significantly, and the proposed algorithm is good at both exploration and exploitation simultaneously. Particularly, the adaptive mixed distribution based sampling strategy could be simply incorporated into existing EDAs to accelerate the convergence speed and escape from the local optima.

6 Future Research Direction

There are still many issues that need to be urgently analyzed , of which the major ones are summarized as follows:

- Although many attempts have been carried out to analyze the characteristics of heavy tail distributions, the comprehensive mathematical properties of the heavy tail probability distributions are subject to suggestion.
- Even though the algorithm performed remarkably better than the classical algorithms, there are still some problems that cannot be completely solved by the proposed algorithm.
- In this chapter, the heavy tail distribution based sampling operator has shown good efficiency based not only on theoretical analysis, but also on the experimental analysis. More attempts are needed to prove the efficiency to the adaptive mixed strategy, such as incorporating the strategy to EP or ES.

Acknowledgements. This work was partially supported by the National Natural Science Foundation of China (Grants No. 60401015, U0835002), the Chinese Academy of Science Special Grant for Postgraduate Research, Innovation and Practice, IEEE Walter Kaplus Student Summer Research Grant, and Graduate Innovation Fund of University of Science and Technology of China.

We are deeply grateful to the reviewers for their invaluable comments which help us improve the quality of the chapter. The first author would like to thank Mr. Aimin Zhou for his warm-hearted help and the invaluable suggestions! Moreover, the authors would like to thank Mr. Xiaolei Chen and Ms. Xuexiao Lai, who helped us a lot in this chapter's preparation.

Appendix

Classical test function 11 and 12

u and y_i are defined as follows:

$$u(x_i,a,k,m)=\begin{cases} k(x_i-a)^m, & if\ x_i>a,\\ 0, & -a\le x_i\le a,\\ k(-x_i-a)^m, & if\ x_i<-a,\end{cases}$$

$$y_i=1+\frac{1}{4}(x_i+1).$$

f_{11} Generalized Penalized Functions

$$f_{11}=\frac{\pi}{30}\{10\sin^2+\sum_{i=1}^{2}9(y_i-1)^2\cdot[1+$$

$$10\sin^2(\pi y_{i+1}+(y_n-1)^2)]+\sum_{i=1}^{30}u(x_i,10,100,4)\}$$

f_{12} Generalized Penalized Functions

$$f_{12}=0.1\{\sin^2(\pi 3x_1)+\sum_{i=1}^{2}9(x_i-1)^2\cdot[1+$$

$$\sin^2(3\pi x_{i+1})]+(x_n-1)^2[1+\sin^2(2\pi x_30)]\}+\sum_{i=1}^{30}u(x_i,5,100,4)$$

A List of Terms and Definition

Heavy tail probability distribution - Different from Gaussian probability distribution whose variance can be denoted as a finite scalar, heavy tail probability distribution stands for a class of probability distributions with an infinite second moment that also yields a stable process.

Estimation of distribution algorithms (EDAs) - The methods that make use of the notion of modeling process as applied in statistics and machine learning domain to

Table 10 Abbreviations used in this chapter

abbreviation	full name
EDA	estimation of distribution algorithm
LSGO	large scale global optimization
MUEDA	mixed distribution based uni-variate EDA
EA	evolutionary algorithm
DE	differential evolution
PSO	particle swarm optimization
FEP	fast evolutionary programming
FES	fast evolution strategy
FSA	fast simulated annealing
LEP	Lévy mutation
LSEDA-gl	EDA with mixed Gaussian and Cauchy distribution for LSGO
FLP	fitness landscape portrait
GA	genetic algorithm
PBIL	population-based incremental learning algorithm
SHCLVND	Stochastic hill climbing by vectors of normal distributions
UMDAc	uni-variate marginal distribution algorithm
MIMIC	mutual information maximizing input clustering
EGNAee	estimation of Gaussian networks algorithm by edge exclusion
CEGDA	clustering and estimation of Gaussian distribution algorithm
IDEA	iterated density estimation algorithm
EP	evolutionary programming
CEP	classical EP
ES	evolution strategy
STDC	Standard Deviation Control Strategy
MFES	maximum fitness evaluation size

extract the important information to effectively build the space structure belong to EDAs.

Fitness landscape analysis - Fitness landscape analysis considers optimization problems as a set of landscapes containing one or more optima. Evolution can thus be viewed as the movement of the population, represented by a set of points (genotypes), towards lower (fitter) areas of the landscape.

Large scale global optimization - Large scale global optimization defines a suit of global optimization problems with more than 100 variables to be optimized.

Kernel probability distribution - Consider a probability with mean 0, symmetrical probability distribution and monotonously increasing; then this probability can be used as Kernel probability distribution for EDA.

References

1. Bosman, P.A.N.: Design and application of iterated density-estimation evolutionary algorithms. Ph.D. dissertation, Universiteit Utrecht, The Netherlands (2003)
2. Bosman, P.A.N., Grahl, J.: Matching inductive search bias and problem structure in continuous Estimation-of-Distribution Algorithms. Eur. J. Oper. Res. 185, 1246–1264 (2008)
3. Bosman, P.A.N., Grahl, J., Rothlauf, F.: SDR: A better trigger for adaptive variance scaling in normal EDAs. In: Proc. Genetic and Evolutionary Computation Conference (GECCO), London, United Kingdom, pp. 492–499 (2007)
4. Bosman, P.A.N., Thierens, D.: Expanding from discrete to continuous Estimation Of Distribution Algorithms: The IDEA. In: Deb, K., Rudolph, G., Lutton, E., Merelo, J.J., Schoenauer, M., Schwefel, H.-P., Yao, X. (eds.) PPSN 2000. LNCS, vol. 1917, pp. 767–776. Springer, Heidelberg (2000)
5. Brest, J., Zamuda, A., Boskovic, B., Maucec, M.S., Zumer, V.: High-dimensional real-parameter optimization using self-adaptive Differential Evolution algorithm with population size reduction. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 2032–2039 (2008)
6. Gnedenko, B., Kolmogorov, A.: Limit distribution for sums of independent random variables. Addison Wesley, Cambridge (1954)
7. Grahl, J., Bosman, P.A.N., Minner, S.: Convergence phases, variance trajectories, and runtime analysis of continuous EDAs. In: Proc. Genetic and Evolutionary Computation Conference (GECCO), pp. 516–522 (2007)
8. Grahl, J., Bosman, P.A.N., Rothlauf, F.: The correlation Ctriggered adaptive variance scaling idea. In: Proc. Genetic and Evolutionary Computation Conference (GECCO), pp. 397–404 (2006)
9. Hansen, N., Gemperle, F., Auger, A., Koumoutsakos, P.: When do heavy-tail distributions help? In: Parallel Problem Solving From Nature IX (PPSN IX), pp. 62–71. Springer, Heidelberg (2006)
10. Hsieh, S., Sun, T., Liu, C., Tsai, S.: Solving large scale global optimization using improved particle swarm optimizer. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 1777–1784 (2008)

11. Larranga, P.: A review on estimation of distribution algorithms. In: Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation, ch. 3, pp. 57–100. Springer, Heidelberg (2001)
12. Larranga, P., Etxeberria, R., Lozano, J.A., Pena, J.M.: Optimization in continuous domains by learning and simulation of Gaussian networks. In: Proc. Genetic and Evolutionary Computation Conference Workshop Program, pp. 201–204 (2000)
13. Lee, C.Y., Yao, X.: Evolutionary programming using mutations based on the Lévy Probability Distribution. IEEE Trans. Evol. Comput. 8(1), 1–13 (2004)
14. Lévy, P.: Theorie de l'Addition des Veriables Aleatoires. Paris, France, Gauthier-Villars (1937)
15. Liu, Y., Yao, X., Zhao, Q., Higuchi, T.: Scaling up fast evolutionary porgramming with cooperative coevolution. In: Proc. IEEE Congress on Evolutionary Computation (CEC), pp. 1101–1108 (2001)
16. Lu, Q., Yao, X.: Clustering and learning Gaussian Distribution for continuous optimization. IEEE Trans. Systems Man and Cybernetics 35(2), 195–204 (2005)
17. MaCnish, C., Yao, X.: Direction matters in high-dimensional optimisation. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 2377–2384 (2008)
18. Mühlenbein, H., Schlierkamp-Voosen, D.: Predictive models for breeder genetic algorithm. Evol. Comput. 1(1), 25–49 (1993)
19. Mühlenbein, H., Schomisch, M., Born, J.: The parallel genetic algorithm as function optimizer. Parallel Comput. 17, 25–49 (1991)
20. Panait, L., Luke, S., Wiegand, R.: Biasing coevolutionary search for optimal multiagent behaviors. IEEE Trans. Evol. Comput. 10(6), 629–645 (2006)
21. Pardalos, P.: Large-Scale nonlinear optimization. In: Nonconvex Optimization and Its Applications. Springer, The Netherlands (2006)
22. Potter, A.M., Jong, K.A.D.: A cooperative co-evolutionary approach to function optimization. In: Davidor, Y., Männer, R., Schwefel, H.-P. (eds.) PPSN 1994. LNCS, vol. 866, pp. 249–257. Springer, Heidelberg (1994)
23. Potter, A.M., Jong, K.A.D.: Cooperative coevolution: An architecture for evolving coadapted subcomponents. Evol. Comput. 8(1), 1–29 (2000)
24. Rudlof, S., Koppen, M.: Stochastic hill climbing with learning by vectors of normal distributions. In: Proc. First Online Workshop on Soft Computing (WSC1), Nagoya, Japan, pp. 60–70 (1996)
25. Sebag, M., Ducoulombier, A.: Extending population-based incremental learning to continuous search spaces. In: Eiben, A.E., Bäck, T., Schoenauer, M., Schwefel, H.-P. (eds.) PPSN 1998. LNCS, vol. 1498, pp. 418–427. Springer, Heidelberg (1998)
26. Shi, Y., Teng, H., Li, Z.: Cooperative co-evolutionary differential evolution for function optimization. In: Proc. First International Conference on Natural Computation, pp. 1080–1088 (2005)
27. Sinha, N., Chakrabarti, R., Chattopadhyay, P.K.: Evolutionary programming techniques for economic load dispatch. IEEE Trans. Evol. Comput. 7(1), 83–94 (2003)
28. Smith, T., Husbands, P., Layzell, P., OShea, M.: Fitness landscapes and evolvability. Evol. Comput. 10(1), 1–34 (2002)
29. Stadler, P.F.: Fitness landscapes. In: Biological Evolution and Statistical Physics, pp. 187–207. Springer, Heidelberg (2002)
30. Szu, H., Hartley, R.: Fast simulated annealing. Phys. Lett. A 122(3,4), 157–162 (1987)

31. Tang, K., Yao, X., Suganthan, P.N., MacNish, C., Chen, Y.P., Chen, C.M., Yang, Z.Y.: Benchmark functions for the CEC 2008 special session and competition on large scale global optimization. Technical Report for IEEE Congress of Evolutionary Computation (CEC) (2008) (special issue)
32. Tseng, L., Chen, C.: Multiple trajectory search for large scale global optimization. In: Proc. IEEE Congress on Evolutionary Computation (CEC), pp. 3052–3059 (2008)
33. Wang, Y., Li, B.: A restart uni-variable estimation of distribution algorithms: Sampling under mixed Gaussian and Levy Probability Distribution. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 3218–3225 (2008)
34. Wright, S.: The roles of mutation, inbreeding, crossbreeding, and selection in evolution. In: Proc. 6th Congr. Genetics, vol. 1, p. 365 (1932)
35. Yang, Z., He, J.S., Yao, X.: Making a difference to differential evolution. In: Michalewicz, Z., Siarry, P. (eds.) Advances in Metaheuristics for Hard Optimization, pp. 397–414. Springer, Heidelberg (2007)
36. Yang, Z., Tang, K., Yao, X.: Differential evolution for high-dimensional function optimization. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Singapore, pp. 3523–3530 (2007)
37. Yang, Z., Tang, K., Yao, X.: Large scale evolutionary optimization using cooperative coevolution. Inf. Sci. 178(15), 2985–2999 (2008)
38. Yang, Z., Tang, K., Yao, X.: Multilevel cooperative coevolution for large scale optimization. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 1663–1670 (2008)
39. Yao, X.: An overview of evolutionary computation. Chinese J. Adv. Software Res. 3(1), 12–29 (1996)
40. Yao, X., Liu, Y.: Fast evolutionary programming. In: Proc. Fifth Annual Conference on Evolutionary Programming (EP 1996), pp. 451–460. MIT Press, San Diego (1996)
41. Yao, X., Liu, Y.: Fast evolution strategies. Control and Cybern 26(3), 467–496 (1997)
42. Zamuda, A., Brest, J., Boskovic, B., Zumer, V.: Large scale global optimization using differential evolution with self adaptation and cooperative co-evolution. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 3719–3726 (2008)
43. Zhao, S., Liang, J.J., Suganthan, P.N., Tasgetiren, M.F.: Dynamic multi-swarm particle swarm optimizer with local search for large scale global optimization. In: Proc. IEEE Congress on Evolutionary Computation (CEC), Hong Kong, pp. 3845–3852 (2008)