Differential Evolution with Dynamic Strategy and Parameter Selection by Detecting Landscape Modality

Tetsuyuki Takahama
Department of Intelligent Systems
Hiroshima City University
Asaminami-ku, Hiroshima, 731-3194 Japan
Email: takahama@info.hiroshima-cu.ac.jp

Setsuko Sakai
Faculty of Commercial Sciences
Hiroshima Shudo University
Asaminami-ku, Hiroshima, 731-3195 Japan
Email: setuko@shudo-u.ac.jp

***Abstract*— Differential Evolution (DE) is an evolutionary algorithm. DE has been successfully applied to optimization problems including non-linear, non-differentiable, non-convex and multimodal functions. There are several mutation strategies such as the best and the rand strategy in DE. It is known that the best strategy is suitable for unimodal problems and the rand strategy is suitable for multimodal problems. However, the landscape of a problem to be optimized is often unknown and the landscape is changing dynamically while the search process proceeds. In this study, we propose a new and simple method that detects the modality of landscape being searched: unimodal or not unimodal. In the method, some points on the line connecting the centroid of search points and the best search point are sampled. When the objective values of the sampled points are changed decreasingly and then increasingly, it is thought that one valley exists. If there exists only one valley, the landscape is unimodal and a greedy strategy like the best strategy is adopted. Otherwise, the rand strategy is adopted. Also, the sampled points realize global search in the region spanned by all search points and realize local search near the best search point. The effect of the proposed method is shown by solving some benchmark problems.**

***Keywords*-differential evolution; landscape modality; parameter control; strategy selection**

I. Introduction

Optimization problems, especially nonlinear optimization problems, are very important and frequently appear in the real world. There exist many studies on solving optimization problems using evolutionary algorithms (EAs). Differential evolution (DE) is an EA by Storn and Price [1]. DE has been successfully applied to optimization problems including non-linear, non-differentiable, non-convex and multimodal functions [2]–[4]. It has been shown that DE is a very fast and robust algorithm.

There are several mutation strategies such as the best strategy and the rand strategy in DE. It is known that the best strategy can solve unimodal problems efficiently but the strategy cannot solve multimodal problems stably and the search by the strategy is sometimes trapped at a local optimal solution. On the contrary, it is known that the rand strategy can solve multimodal problems stably but the strategy cannot solve unimodal problems efficiently. However, the landscape of a problem to be optimized is often unknown and the landscape is changing dynamically while the search process proceeds. Thus, it is difficult to select a proper strategy.

In this study, we propose a new and simple method that detects the modality of landscape being searched: unimodal or not unimodal. In the method, some points on the line connecting the centroid of search points and the best search point are sampled. When the objective values of the sampled points are changed decreasingly and then increasingly, it is thought that one valley exists. If there exists only one valley, the landscape is unimodal and a greedy strategy is adopted. In the strategy, a base vector is selected from among top-ranked search points. Thus, it is expected that the strategy improves the efficiency of the search. If the number of valley is not one, the rand strategy is adopted. In the strategy, a base vector is selected randomly from whole search points. Thus, it is expected that the strategy improves the divergence of the search and prevents premature convergence. Also, the sampled points realize global search in the region spanned by all search points and realize local search near the best search point. The effect of the proposed method is shown by solving 13 benchmark problems including unimodal problems and multimodal problems.

In Section II, optimization problems and DE are explained. Related works are briefly reviewed in Section III. DE with detecting landscape modality is proposed in Section IV. In Section V, experimental results on some problems are shown. Finally, conclusions are described in Section VI.

II. Optimization by Differential Evolution

A. Optimization Problems

In this study, the following optimization problem with lower bound and upper bound constraints will be discussed.

$$\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & l_i \le x_i \le u_i,\ i = 1, \ldots, D, \end{array} \tag{1}$$

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_D)$ is an D dimensional vector and $f(\boldsymbol{x})$ is an objective function. The function f is a nonlinear real-valued function. Values l_i and u_i are the lower bound and the upper bound of x_i, respectively. Let the search space in which every point satisfies the lower and upper bound constraints be denoted by $\mathcal{S}$.

B. Differential Evolution

DE is a stochastic direct search method using a population or multiple search points.

In DE, initial individuals are randomly generated within given search space and form an initial population. Each individual contains D genes as decision variables. At each generation or iteration, all individuals are selected as parents. Each parent is processed as follows: The mutation operation begins by choosing several individuals from the population except for the parent in the processing. The first individual is a base vector. All subsequent individuals are paired to create difference vectors. The difference vectors are scaled by a scaling factor F and added to the base vector. The resulting vector, or a mutant vector, is then recombined with the parent. The probability of recombination at an element is controlled by a crossover rate CR. This crossover operation produces a trial vector. Finally, for survivor selection, the trial vector is accepted for the next generation if the trial vector is better than the parent.

There are some variants of DE that have been proposed. The variants are classified using the notation DE/$base$/num/$cross$ such as DE/rand/1/bin and DE/rand/1/exp.

"$base$" specifies a way of selecting an individual that will form the base vector. For example, DE/rand selects an individual for the base vector at random from the population. DE/best selects the best individual in the population.

"num" specifies the number of difference vectors used to perturb the base vector. In case of DE/rand/1, for example, for each parent $\boldsymbol{x}^i$, three individuals $\boldsymbol{x}^{p1}$, $\boldsymbol{x}^{p2}$ and $\boldsymbol{x}^{p3}$ are chosen randomly from the population without overlapping $\boldsymbol{x}^i$ and each other. A new vector, or a mutant vector $\boldsymbol{x}'$ is generated by the base vector $\boldsymbol{x}^{p1}$ and the difference vector $\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}$, where F is the scaling factor.

$$\boldsymbol{x}' = \boldsymbol{x}^{p1} + F(\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3}) \tag{2}$$

"$cross$" specifies the type of crossover that is used to create a child. For example, 'bin' indicates that the crossover is controlled by the binomial crossover using a constant crossover rate, and 'exp' indicates that the crossover is controlled by a kind of two-point crossover using exponentially decreasing the crossover rate. Fig. 1 shows the binomial and exponential crossover. A new child $\boldsymbol{x}^{\text{child}}$ is generated from the parent $\boldsymbol{x}^i$ and the mutant vector $\boldsymbol{x}'$, where CR is a crossover rate.

C. The Algorithm of Differential Evolution

The algorithm of DE is as follows:

Step1 Initialization of a population. Initial N individuals $P = \{\boldsymbol{x}^i, i = 1, 2, \cdots, N\}$ are generated randomly in search space and form an initial population.

Step2 Termination condition. If the number of function evaluations exceeds the maximum number of evaluation $FE_{\max}$, the algorithm is terminated.

Step3 DE operations. Each individual $\boldsymbol{x}^i$ is selected as a parent. If all individuals are selected, go to Step4. A mutant vector $\boldsymbol{x}'$ is generated according to Eq. (2). A trial vector (child) is generated from the parent $\boldsymbol{x}^i$ and the mutant vector $\boldsymbol{x}'$ using a crossover operation shown in Fig. 1. If the child is better than or equal to the parent, or the DE operation is succeeded, the child survives. Otherwise the parent survives. Go back to Step3 and the next individual is selected as a parent.

Step4 Survivor selection (generation change). The population is organized by the survivors. Go back to Step2.

Fig. 2 shows a pseudo-code of DE/rand/1.

```
binomial crossover DE/·/·/bin
  $j_{rand}$=randint(1,$D$);
  for($k$=1; $k \le D$; $k$++) {
    if($k == j_{rand}$ || $u(0,1) < CR$) $x_k^{\text{child}}$=$x'_k$;
    else $x_k^{\text{child}}$=$x_k^i$;
  }
exponential crossover DE/·/·/exp
  $k$=1; $j$=randint(1,$D$);
  do {
      $x_j^{\text{child}}$=$x'_j$;
      $k$=$k$+1; $j$=$(j+1)\%D$;
  } while($k \le D$ && $u(0,1) < CR$);
  while($k \le D$) {
      $x_j^{\text{child}}$=$x_j^i$;
      $k$=$k$+1; $j$=$(j+1)\%D$;
  }
```

Fig. 1. Binomial and exponential crossover operation, where randint(1,D) generates an integer randomly from $[1, D]$ and $u(l, r)$ is a uniform random number generator in $[l, r]$.

```
DE/rand/1()
{
// Initialize an population
 $P$=$N$ individuals generated randomly in $\mathcal{S}$;
 for($t$=1; $FE \le FE_{max}$; $t$++) {
  for($i$=1; $i \le N$; $i$++) {
// DE operation
     $\boldsymbol{x}^{p1}$=Randomly selected from $P$($p1 \ne i$);
     $\boldsymbol{x}^{p2}$=Randomly selected from $P$($p2 \notin \{i, p1\}$);
     $\boldsymbol{x}^{p3}$=Randomly selected from $P$($p3 \notin \{i, p1, p2\}$);
     $\boldsymbol{x}'$=$\boldsymbol{x}^{p1}$+$F(\boldsymbol{x}^{p2} - \boldsymbol{x}^{p3})$;
     $\boldsymbol{x}^{\text{child}}$=trial vector is generated from
           $\boldsymbol{x}^i$ and $\boldsymbol{x}'$ by the crossover operation;
// Survivor selection
     if$\big(f(\boldsymbol{x}^{\text{child}}) \le f(\boldsymbol{x}^i)\big)$ $\boldsymbol{z}^i$=$\boldsymbol{x}^{\text{child}}$;
     else                  $\boldsymbol{z}^i$=$\boldsymbol{x}^i$;
     $FE$=$FE$+1;
  }
  $P$=$\{\boldsymbol{z}^i,\ i = 1, 2, \cdots, N\}$;
 }
}
```

Fig. 2. The pseudo-code of DE, FE is the number of function evaluations.

III. Related Works

The performance of DE is affected by control parameters such as the scaling factor F, the crossover rate CR and the population size N, and by mutation strategies such as the rand strategy and the best strategy. Many researchers have been studying on controlling the parameters and the strategies. The methods of the control can be classified into some categories as follows:

(1) selection-based control: Strategies and parameter values are selected regardless of current search state. CoDE(composite DE) [5] generates three trial vectors using three strategies with randomly selected parameter values from parameter candidate sets and the best trial vector will head to the survivor selection.

(2) observation-based control: The current search state is observed, proper parameter values are inferred according to the observation, and parameters and/or strategies are dynamically controlled. FADE(Fuzzy Adaptive DE) [6] observes the movement of search points and the change of function values between successive generations, and controls F and CR. DESFC(DE with Speciation and Fuzzy Clustering) [7] adopts fuzzy clustering, observes partition entropy of search points, and controls CR and the mutation strategies between the rand and the species-best strategy.

(3) success-based control: It is recognized as a success case when a better search point than the parent is generated. The parameters and/or strategies are adjusted so that the values in the success cases are frequently used. It is thought that the self-adaptation, where parameters are contained in individuals and are evolved by applying evolutionary operators to the parameters, is included in this category. DESAP(Differential Evolution with Self-Adapting Populations) [8] controls F, CR and N self-adaptively. SaDE(Self-adaptive DE) [9] controls the selection probability of the mutation strategies according to the success rates and controls the mean value of CR for each strategy according to the mean value in success case. jDE(self-adaptive DE algorithm) [10] controls F and CR self-adaptively. JADE(adaptive DE with optional external archive) [11] and MDE_pBX(modified DE with p-best crossover) [12] control the mean and power mean values of F and CR according to the mean values in success cases.

In the category (1), useful knowledge to improve the search efficiency is ignored. In the category (2), it is difficult to select proper type of observation which is independent of the optimization problem and its scale. In the category (3), when a new good search point is found near the parent, parameters are adjusted to the direction of convergence. In problems with ridge landscape or multimodal landscape, where good search points exist in small region, parameters are tuned for small success and big success will be missed. Thus, search process would be trapped at a local optimal solution.

In this study, we propose new observation-based control in the category 1). In the control, F and CR are tuned and a strategy is selected between a greedy strategy and the rand strategy according to the landscape modality which is inferred by a kind of line search.

IV. Differential Evolution with Detecting Landscape Modality

In this section, LMDE (DE with detecting Landscape Modality) is proposed.

A. Detecting Landscape Modality

In this study, whether the search points are in unimodal landscape or not is detected using the current search points $P = \{\boldsymbol{x}^i | i = 1, 2, \cdots, N\}$. The objective values are examined along the following line, which connects the centroid of search points $\boldsymbol{x}^g$ and the best search point $\boldsymbol{x}^b$.

$$\boldsymbol{x} = \boldsymbol{x}^g + \lambda(\boldsymbol{x}^b - \boldsymbol{x}^g) \tag{3}$$

$$\boldsymbol{x}^g = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{x}^i \tag{4}$$

$$\boldsymbol{x}^b = \arg\min_i f(\boldsymbol{x}^i) \tag{5}$$

where λ is a parameter for deciding the position of a point on the line. The range of the search points $[\boldsymbol{x}^{\min}, \boldsymbol{x}^{\max}]$ can be given as follows:

$$x_j^{\min} = \min_i x_j^i \tag{6}$$

$$x_j^{\max} = \max_i x_j^i \tag{7}$$

The range of the λ, $[\lambda^{\min}, \lambda^{\max}]$ satisfies the following condition:

$$x_j^{\min} \le x_j^g + \lambda(x_j^b - x_j^g) \le x_j^{\max} \tag{8}$$

Thus, if $(x_j^b - x_j^g)$ is positive, the range of the λ is given by:

$$\lambda^{\min} = \max_j \frac{x_j^{\min} - x_j^g}{x_j^b - x_j^g} \tag{9}$$

$$\lambda^{\max} = \min_j \frac{x_j^{\max} - x_j^g}{x_j^b - x_j^g} \tag{10}$$

If $(x_j^b - x_j^g)$ is negative, $x_j^{\min}$ and $x_j^{\max}$ in the equations are exchanged.

In order to decide M sampling points $\{\boldsymbol{x}_k | k = 1, 2, \cdots, M\}$, λ_k is given as follows:

$$\lambda_k = \lambda^{\min} + \frac{\lambda^{\max} - \lambda^{\min}}{M - 1}(k - 1) \tag{11}$$

$$\boldsymbol{x}_k = \boldsymbol{x}^g + \lambda_k(\boldsymbol{x}^b - \boldsymbol{x}^g) \tag{12}$$

Figure 3 shows an example of the sampling, where search points are shown by black circles, the centroid is shown by a white circle, sampling points are shown by triangles in case of $M = 6$.

In the sequence $\{f(\boldsymbol{x}_k) | k = 1, 2, \cdots, M\}$, hill-valley relation is examined. For each point, the function dir is introduced in order to judge whether the change is increasing or decreasing:

$$dir(\boldsymbol{x}_k) = \begin{cases} 1 & (f(\boldsymbol{x}_{k+1}) > f(\boldsymbol{x}_k)) \\ -1 & (f(\boldsymbol{x}_{k+1}) < f(\boldsymbol{x}_k)) \\ dir(\boldsymbol{x}_{k-1}) & (\text{otherwise}) \end{cases} \tag{13}$$

If the value of dir changed from -1 to 1 only once, it is thought that one valley exists and the landscape is unimodal. Otherwise, the landscape is not unimodal. Figure 4 shows an example of detecting unimodal landscape, where the objective values are shown by the function of λ.

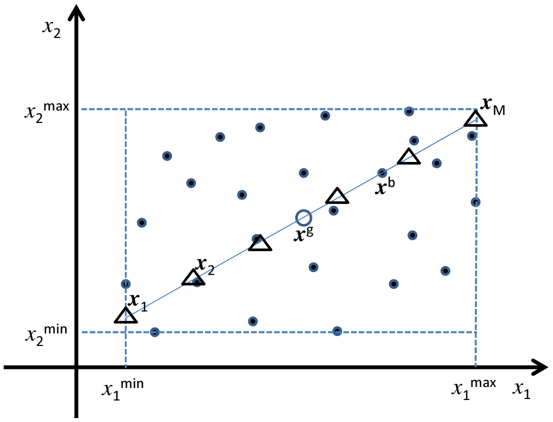

Fig. 3. An example of sampling for detecting landscape modality

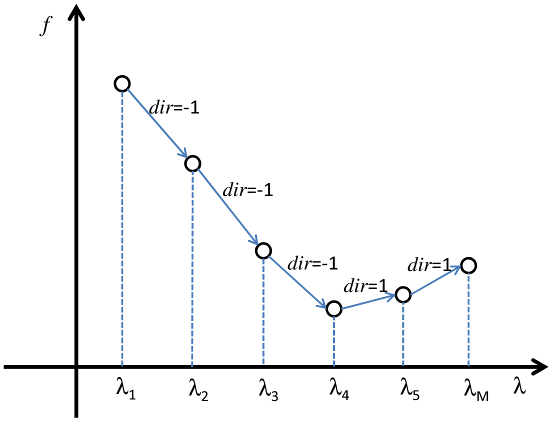

Fig. 4. An example of detecting unimodal landscape

If the best value of $f(\boldsymbol{x}_k)$ is better than $f(\boldsymbol{x}^b)$, $\boldsymbol{x}^b$ is replaced by the $\boldsymbol{x}_k$.

$$\boldsymbol{x}^b = \begin{cases} \arg\min_k f(\boldsymbol{x}_k) & (\min_k f(\boldsymbol{x}_k) < f(\boldsymbol{x}^b)) \\ \boldsymbol{x}^b & (\text{otherwise}) \end{cases} \quad (14)$$

It is thought that the sampling performs global search in the region spanned by all search points and also perform local search near the best search point.

B. Algorithm of LMDE

Fig. 5 shows the pseudo-code of LMDE. Some modifications to standard DE are applied for proposed method:

1) Dynamic strategy selection and parameter selection are performed according to landscape modality. If current landscape is unimodal, a greedy strategy, which is called as the pbest strategy, is adopted. In the strategy, a base vector is selected from top p ranking individuals, where the rank of the base vector is in $[1, pN]$. Also, the initial scaling factor F_0 is decreased by 0.1 in order to converge search points to a valley fast. If current landscape is not unimodal, the rand strategy is adopted. In order to generate a crossover rate CR, the base rate CR_0 is perturbed by less than a small value 0.1 randomly.

```
LMDE/{pbest,rand}/1/exp()
{
  isGreedy=0; $F$=$F_0$; $FE$=0;
// Initialize a population
 $P$=$N$ individuals generated randomly in $\mathcal{S}$;
 $FE$=$FE$+$N$;
 for($t$=1; $FE \le FE_{max}$; $t$++) {
  if($t$%$T_d$==$T_d - 1$) {
    Detecting landscape modality;
    $FE$=$FE$+$M$;
    if(landscape is unimodal) {
      isGreedy=1; $F$=$F_0$-0.1;
    }
    else {
      isGreedy=0; $F$=$F_0$;
    }
  }
  $CR$=($CR_0 - 0.05$)+0.1$u(0,1)$;
  for($i$=1; $i \le N$; $i$++) {
// DE/rand/1/exp operation
   if(isGreedy)
     $\boldsymbol{x}^{p_1}$=Randomly selected from top $p$ ranking in $P$;
   else
     $\boldsymbol{x}^{p_1}$=Randomly selected from $P$($p_1 \neq i$);
   $\boldsymbol{x}^{p_2}$=Randomly selected from $P$($p2 \notin \{i, p_1\}$);
   $\boldsymbol{x}^{p_3}$=Randomly selected from $P$($p3 \notin \{i, p_1, p_2\}$);
   $\boldsymbol{x}'$=$\boldsymbol{x}^{p_1}$+$F$($\boldsymbol{x}^{p_2} - \boldsymbol{x}^{p_3}$);
   $\boldsymbol{x}^c$=trial vector is generated from
            $\boldsymbol{x}^i$ and $\boldsymbol{x}'$ by exponential crossover;
   $FE$=$FE$+1;
// Survivor selection
   if$\left(f(\boldsymbol{x}^c) \le f(\boldsymbol{x}^i)\right)$ $\boldsymbol{x}^i$=$\boldsymbol{x}^c$;
  }
 }
}
```

Fig. 5. The pseudo-code of LMDE where T_d is the term of detecting landscape modality and p is the ratio of top-ranking individuals.

2) Continuous generation model [13] is adopted. Usually discrete generation model is adopted in DE and when the child is better than the parent, the child survives in the next generation. In this study, when the child is better than the parent, the parent is immediately replaced by the child. It is thought that the continuous generation model improves efficiency because the model can use newer information than the discrete model.

3) Reflecting back out-of-bound solutions [14] is adopted. In order to keep bound constraints, an operation to move a point outside of the search space $\mathcal{S}$ into the inside of $\mathcal{S}$ is required. There are some ways to realize the movement: generating solutions again, cutting off the solutions on the boundary, and reflecting points back to the inside of the boundary [15]. In this study, reflecting back is used:

$$x_{ij} = \begin{cases} l_i + (l_i - x_{ij}) - \left\lfloor \frac{l_i - x_{ij}}{u_i - l_i} \right\rfloor (u_i - l_i) & (x_{ij} < l_i) \\ u_i - (x_{ij} - u_i) + \left\lfloor \frac{x_{ij} - u_i}{u_i - l_i} \right\rfloor (u_i - l_i) & (x_{ij} > u_i) \\ x_{ij} & (\text{otherwise}) \end{cases} \quad (15)$$

where $\lfloor z \rfloor$ is the maximum integer smaller than or equal to z. This operation is applied when a new point is generated by DE operations.

V. Solving Optimization Problems

In this paper, well-known thirteen benchmark problems are solved.

A. Test Problems and Experimental Conditions

The 13 scalable benchmark functions are shown in Table I [11]. All functions have an optimal value 0. Some characteristics are briefly summarized as follows: Functions f_1 to f_4 are continuous unimodal functions. The function f_5 is Rosenbrock function which is unimodal for 2- and 3-dimensions but may have multiple minima in high dimension cases [16]. The function f_6 is a discontinuous step function, and f_7 is a noisy quartic function. Functions f_8 to f_{13} are multimodal functions and the number of their local minima increases exponentially with the problem dimension [17].

TABLE I
Test functions of dimension D. These are sphere, Schwefel 2.22, Schwefel 1.2, Schwefel 2.21, Rosenbrock, step, noisy quartic, Schwefel 2.26, Rastrigin, Ackley, Griewank, and two penalized functions, respectively [18]

Test functions	Bound constraints
$f_1(\boldsymbol{x}) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$
$f_2(\boldsymbol{x}) = \sum_{i=1}^{D} \lvert x_i\rvert + \prod_{i=1}^{D} \lvert x_i\rvert$	$[-10, 10]^D$
$f_3(\boldsymbol{x}) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j\right)^2$	$[-100, 100]^D$
$f_4(\boldsymbol{x}) = \max_i\{\lvert x_i\rvert\}$	$[-100, 100]^D$
$f_5(\boldsymbol{x}) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2\right]$	$[-30, 30]^D$
$f_6(\boldsymbol{x}) = \sum_{i=1}^{D} \lfloor x_i + 0.5\rfloor^2$	$[-100, 100]^D$
$f_7(\boldsymbol{x}) = \sum_{i=1}^{D} ix_i^4 + rand[0, 1)$	$[-1.28, 1.28]^D$
$f_8(\boldsymbol{x}) = \sum_{i=1}^{D} -x_i \sin\sqrt{\lvert x_i\rvert}$ $+ D \cdot 418.98288727243369$	$[-500, 500]^D$
$f_9(\boldsymbol{x}) = \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-5.12, 5.12]^D$
$f_{10}(\boldsymbol{x}) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)$ $- \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + 20 + e$	$[-32, 32]^D$
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^D$
$f_{12}(x) = \frac{\pi}{D}[10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2$ $\{1 + 10\sin^2(\pi y_{i+1})\} + (y_D - 1)^2]$ $+\sum_{i=1}^{D} u(x_i, 10, 100, 4)$ where $y_i = 1 + \frac{1}{4}(x_i + 1)$ and $u(x_i, a, k, m) =$ $\begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50, 50]^D$
$f_{13}(x) = 0.1[\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2$ $\{1 + \sin^2(3\pi x_{i+1})\} + (x_D - 1)^2$ $\{1 + \sin^2(2\pi x_D)\}] + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$

Independent 50 runs are performed for 13 problems. The dimension of problems is 30 (D=30). Each run stops when the number of function evaluations (FEs) exceeds the maximum number of evaluations $FE_{\max}$.

B. Experimental Results on the Proposed Method

The control parameters for LMDE are as follows: The population size N=50, the initial scaling factor F_0=0.7, the base crossover rate CR_0=0.9, the term of detecting landscape modality T_d=20, the number of sampling points is same as the population size or M=N, and the ratio of top-ranking ratio p=0.2. The parameter for standard DEs are as follows: The population size N=50, the scaling factor F=0.7 and the crossover rate CR=0.9. These settings showed very good and stable performance in constrained optimization [19].

Table II shows the experimental results on LMDE and standard DEs, or DE/rand/1/exp and DE/rand/1/bin. The mean value and the standard deviation of best objective value in each run are shown for each function. The best result among algorithms is highlighted using bold face fonts. Apparently, LMDE found better solutions than standard DEs did in all problems. In standard DEs, DE/rand/1/exp found better solutions than DE/rand/1/bin did in all problems except for f_7.

Figures 6 to 16 show the change of best objective value found and the number of changes in dir values of Eq. (13) over the number of FEs within 200,000 evaluations. Apparently, proposed method can find better objective values faster than the standard DEs in all problems. Also, the number of dir changes is almost 1 in unimodal functions f_1 to f_4.

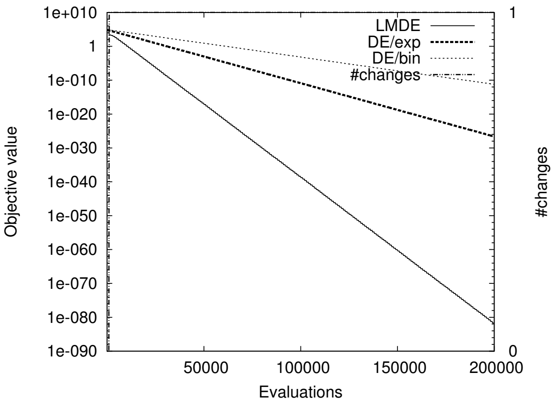

Fig. 6. The graph of f_1

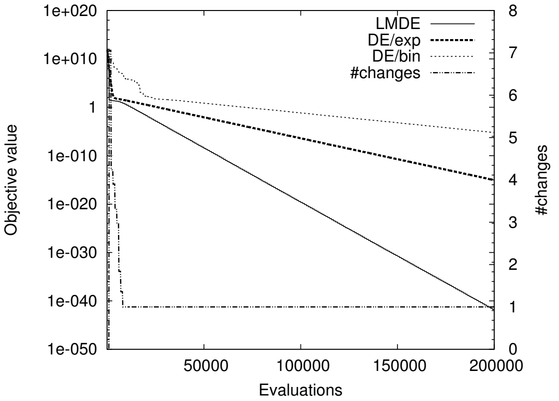

Fig. 7. The graph of f_2

TABLE II

EXPERIMENTAL RESULTS ON LMDE AND STANDARD DES. MEAN VALUE ± STANDARD DEVIATION IN 50 RUNS ARE SHOWN

	FE_{max}	LMDE	DE/rand/1/exp	DE/rand/1/bin
f_1	150,000	**3.79591e-61** ± 4.837e-61	1.90493e-19 ± 1.198e-19	5.83076e-08 ± 5.088e-08
f_2	200,000	**1.09424e-42** ± 9.846e-43	8.86812e-16 ± 3.194e-16	5.86097e-06 ± 3.192e-06
f_3	500,000	**1.48640e-70** ± 6.230e-70	6.82357e-08 ± 3.063e-08	5.00996e-07 ± 6.499e-07
f_4	500,000	**1.57039e-34** ± 5.342e-34	2.53499e-07 ± 6.265e-08	1.58056e-02 ± 2.626e-02
f_5	300,000	**0.00000e+00** ± 0.000e+00	2.45220e-14 ± 2.978e-14	1.00177e+00 ± 7.345e-01
	2,000,000	0.00000e+00 ± 0.000e+00	0.00000e+00 ± 0.000e+00	0.00000e+00 ± 0.000e+00
f_6	10,000	**7.64000e+00** ± 3.974e+00	1.92612e+03 ± 3.381e+02	1.25022e+04 ± 3.504e+03
	150,000	0.00000e+00 ± 0.000e+00	0.00000e+00 ± 0.000e+00	0.00000e+00 ± 0.000e+00
f_7	300,000	**4.63046e-04** ± 2.336e-04	9.68155e-03 ± 2.274e-03	9.53495e-03 ± 2.927e-03
f_8	100,000	**0.00000e+00** ± 0.000e+00	2.39661e-09 ± 2.649e-09	3.62890e+03 ± 1.598e+03
	900,000	**0.00000e+00** ± 0.000e+00	**0.00000e+00** ± 0.000e+00	1.05016e+02 ± 1.561e+02
f_9	100,000	**1.43683e+01** ± 3.363e+00	1.45655e+01 ± 3.767e+00	1.64190e+02 ± 2.584e+01
	500,000	**0.00000e+00** ± 0.000e+00	1.98992e-02 ± 1.393e-01	1.82250e+01 ± 5.894e+00
f_{10}	50,000	**6.52192e-09** ± 6.471e-09	1.04405e-02 ± 2.011e-03	2.08414e+00 ± 4.564e-01
	200,000	**0.00000e+00** ± 0.000e+00	2.27729e-14 ± 3.734e-15	6.78799e-07 ± 3.623e-07
f_{11}	50,000	**2.62380e-10** ± 1.830e-09	1.87902e-02 ± 1.659e-02	1.07438e+00 ± 3.875e-02
	300,000	**0.00000e+00** ± 0.000e+00	**0.00000e+00** ± 0.000e+00	3.45066e-04 ± 1.708e-03
f_{12}	50,000	**3.72164e-17** ± 7.188e-17	7.04676e-05 ± 2.548e-05	3.59688e+00 ± 1.940e+00
	150,000	**1.57054e-32** ± 0.000e+00	1.45567e-20 ± 9.382e-21	4.19561e-08 ± 8.577e-08
f_{13}	50,000	**3.16981e-14** ± 1.098e-13	3.59667e-04 ± 1.194e-04	7.34430e+00 ± 4.543e+00
	150,000	**1.34978e-32** ± 0.000e+00	7.04252e-20 ± 4.546e-20	1.28632e-07 ± 1.248e-07

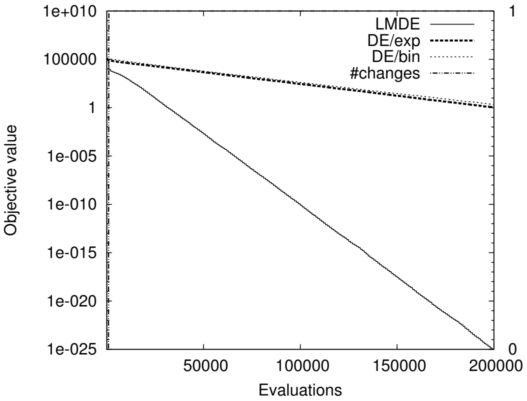

Fig. 8. The graph of f_3

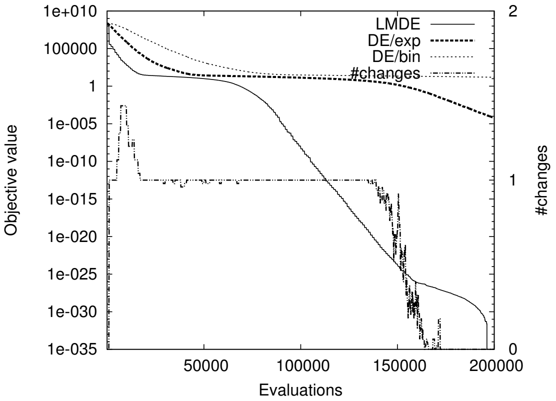

Fig. 10. The graph of f_5

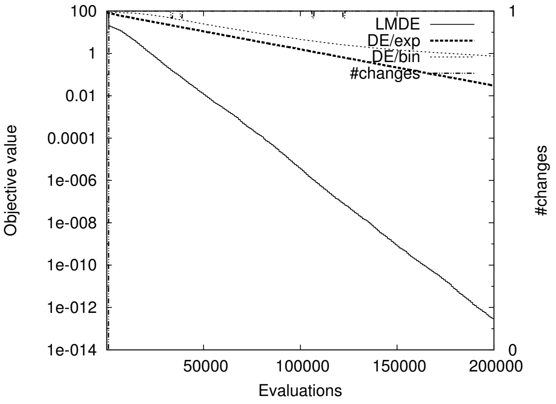

Fig. 9. The graph of f_4

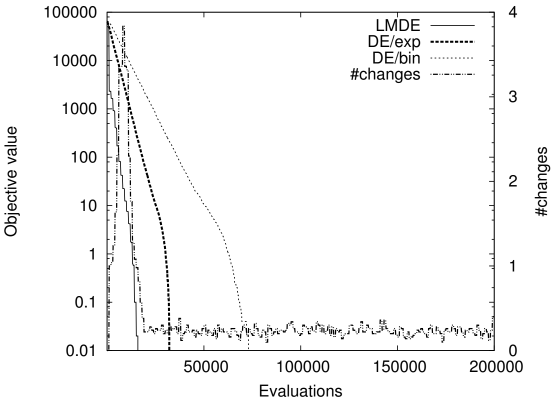

Fig. 11. The graph of f_6

C. Comparison with Other Methods

Table III compares LMDE with other methods including JADE, jDE, SaDE, DE/rand/1/bin and PSO. Results except for LMDE are taken from [11]. The control parameters are as follows: The population size N=100 for DEs, and F=0.5 and CR=0.9 for DE/rand/1/bin.

LMDE attained the best results among all methods in 7 problems f_1, f_2, f_5, f_7, f_8, f_{11} and f_{12}. Also, LMDE attained

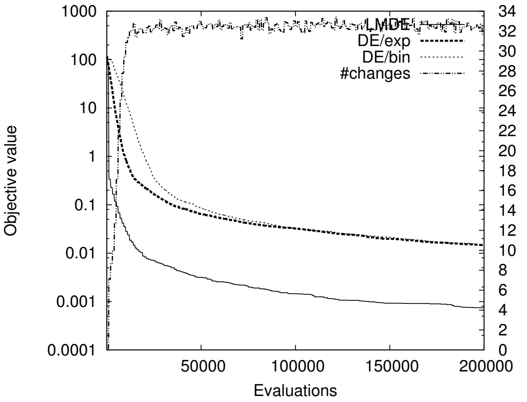

Fig. 12. The graph of f_7

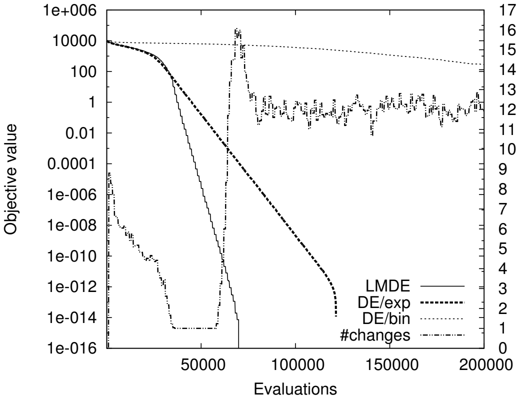

Fig. 13. The graph of f_8

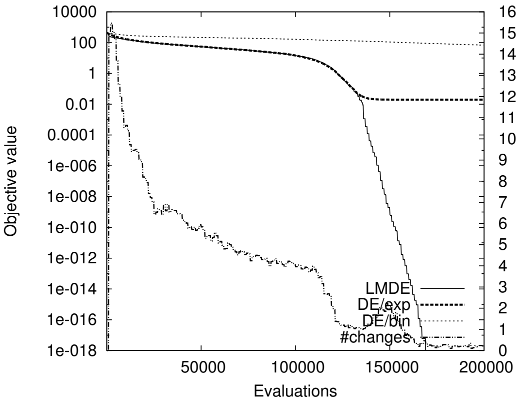

Fig. 14. The graph of f_9

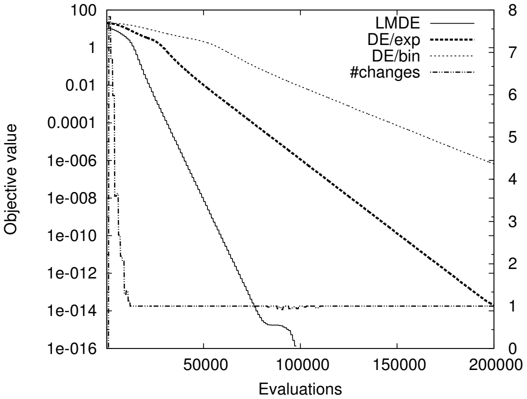

Fig. 15. The graph of f_{10}

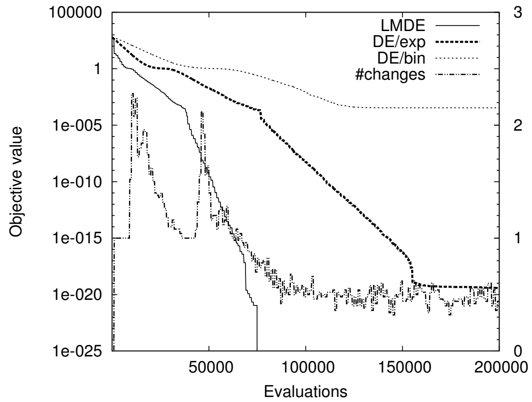

Fig. 16. The graph of f_{11}

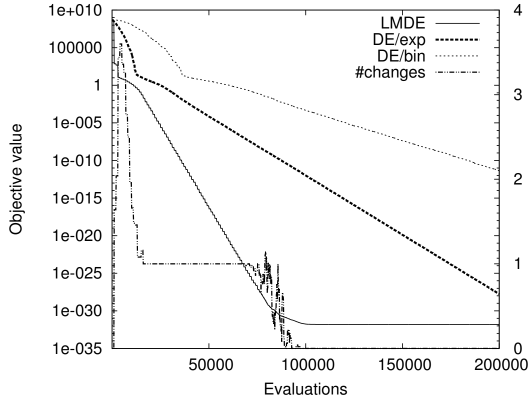

Fig. 17. The graph of f_{12}

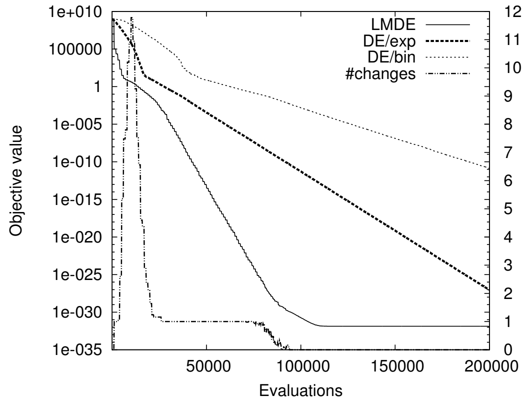

Fig. 18. The graph of f_{13}

the best final results in 2 problems f_{10} and f_{13}. LMDE outperformed JADE without archive in 9 problems and in 2 problems for final results. LMDE outperformed JADE with archive in 7 problems and in 2 problems for final results. LMDE outperformed jDE, SaDE, DE/rand/1/bin and PSO in all problems. Thus, it is thought that LMDE is effective to various problems.

VI. Conclusion

Differential evolution is known as a simple, efficient and robust search algorithm that can solve nonlinear optimization problems. In this study, we proposed the landscape modality detection and LMDE algorithm to select a proper strategy and

TABLE III
EXPERIMENTAL RESULTS ON LMDE AND OTHER DES. MEAN VALUES AND STANDARD DEVIATIONS IN 50 RUNS ARE SHOWN

	FE_{max}	LMDE	JADE w/o archive	JADE with archive	jDE	SaDE	DE/rand/1/bin	PSO
f_1	150,000	**3.8e-61 (4.8e-61)**	1.8e-60 (8.4e-60)	1.3e-54 (9.2e-54)	2.5e-28 (3.5e-28)	4.5e-20 (6.9e-20)	9.8e-14 (8.4e-14)	9.6e-42 (2.7e-41)
f_2	200,000	**1.1e-42 (9.8e-43)**	1.8e-25 (8.8e-25)	3.9e-22 (2.7e-21)	1.5e-23 (1.0e-23)	1.9e-14 (1.05e-14)	1.6e-09 (1.1e-09)	9.3e-21 (6.3e-20)
f_3	500,000	1.5e-70 (6.2e-70)	5.7e-61 (2.7e-60)	**6.0e-87 (1.9e-86)**	5.2e-14 (1.1e-13)	9.0e-37 (5.43e-36)	6.6e-11 (8.8e-11)	2.5e-19 (3.9e-19)
f_4	500,000	1.5e-34 (5.3e-34)	8.2e-24 (4.0e-23)	**4.3e-66 (1.2e-65)**	1.4e-15 (1.0e-15)	7.4e-11 (1.82e-10)	4.2e-01 (1.1e+00)	4.4e-14 (9.3e-14)
f_5	300,000	**0.0e+00 (0.0e+00)**	8.0e-02 (5.6e-01)	3.2e-01 (1.1e+00)	1.3e+01 (1.4e+01)	2.1e+01 (7.8e+00)	2.1e+00 (1.5e+00)	2.5e+01 (3.2e+01)
	2,000,000	**0.0e+00 (0.0e+00)**	8.0e-02 (5.6e-01)	3.2e-01 (1.1e+00)	8.0e-02 (5.6e-01)	1.8e+01 (6.7e+00)	8.0e-02 (5.6e-01)	1.7e+01 (2.3e+01)
f_6	10,000	7.6e+00 (4.0e+00)	**2.9e+00 (1.2e+00)**	5.6e+00 (1.6e+00)	1.0e+03 (2.2e+02)	9.3e+02 (1.8e+02)	4.7e+03 (1.1e+03)	4.5e+01 (2.4e+01)
	150,000	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	8.0e-02 (2.7e-01)
f_7	300,000	**4.6e-04 (2.3e-04)**	6.4e-04 (2.5e-04)	6.8e-04 (2.5e-04)	3.3e-03 (8.5e-04)	4.8e-03 (1.2e-03)	4.7e-03 (1.2e-03)	2.5e-03 (1.4e-03)
f_8	100,000	**0.0e+00 (0.0e+00)**	3.3e-05 (2.3e-05)	7.1e+00 (2.8e+01)	7.9e-11 (1.3e-10)	4.7e+00 (3.3e+01)	5.9e+03 (1.1e+03)	2.4e+03 (6.7e+02)
	900,000	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	7.1e+00 (2.8e+01)	0.0e+00 (0.0e+00)	4.7e+00 (3.3e+01)	5.7e+01 (7.6e+01)	2.4e+03 (6.7e+02)
f_9	100,000	1.4e+01 (3.4e+00)	**1.0e-04 (6.0e-05)**	1.4e-04 (6.5e-05)	1.5e-04 (2.0e-04)	1.2e-03 (6.5e-04)	1.8e+02 (1.3e+01)	5.2e+01 (1.6e+01)
	500,000	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	7.1e+01 (2.1e+01)	5.2e+01 (1.6e+01)
f_{10}	50,000	6.5e-09 (6.5e-09)	**8.2e-10 (6.9e-10)**	3.0e-09 (2.2e-09)	3.5e-04 (1.0e-04)	2.7e-03 (5.1e-04)	1.1e-01 (3.9e-02)	4.6e-01 (6.6e-01)
	200,000	**0.0e+00 (0.0e+00)**	4.4e-15 (0.0e+00)	4.4e-15 (0.0e+00)	4.7e-15 (9.6e-16)	4.3e-14 (2.6e-14)	9.7e-11 (5.0e-11)	4.6e-01 (6.6e-01)
f_{11}	50,000	**2.6e-10 (1.8e-09)**	9.9e-08 (6.0e-07)	2.0e-04 (1.4e-03)	1.9e-05 (5.8e-05)	7.8e-04 (1.2e-03)	2.0e-01 (1.1e-01)	1.3e-02 (1.7e-02)
	300,000	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	2.0e-04 (1.4e-03)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	0.0e+00 (0.0e+00)	1.1e-02 (1.6e-02)
f_{12}	50,000	**3.7e-17 (7.2e-17)**	4.6e-17 (1.9e-16)	3.8e-16 (8.3e-16)	1.6e-07 (1.5e-07)	1.9e-05 (9.2e-06)	1.2e-02 (1.0e-02)	1.9e-01 (3.9e-01)
	150,000	1.6e-32 (0.0e+00)	1.6e-32 (5.5e-48)	1.6e-32 (5.5e-48)	2.6e-29 (7.5e-29)	1.2e-19 (2.0e-19)	1.1e-14 (1.0e-14)	1.9e-01 (3.9e-01)
f_{13}	50,000	3.2e-14 (1.1e-13)	**2.0e-16 (6.5e-16)**	1.2e-15 (2.8e-15)	1.5e-06 (9.8e-07)	6.1e-05 (2.0e-05)	7.5e-02 (3.8e-02)	2.9e-03 (4.8e-03)
	150,000	**1.3e-32 (0.0e+00)**	1.4e-32 (1.1e-47)	1.4e-32 (1.1e-47)	1.9e-28 (2.2e-28)	1.7e-19 (2.4e-19)	7.5e-14 (4.8e-14)	2.9e-03 (4.8e-03)

to adjust control parameters. It was shown that LMDE can improve the search efficiency compared with standard DEs in all 13 problems. Also, it was shown that LMDE outperformed jDE, SaDE and PSO in all 13 problems, and JADE in 7 problems and 2 problems for final results. Thus, it is thought that LMDE is a very efficient optimization algorithm compared with other methods.

In the future, we will design more dynamic control of parameter values for LMDE.

ACKNOWLEDGMENT

This research is supported in part by Grant-in-Aid for Scientific Research (C) (No. 20500138,22510166) of Japan society for the promotion of science and Hiroshima City University Grant for Special Academic Research (General Studies).

REFERENCES

[1] R. Storn and K. Price, "Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, pp. 341–359, 1997.

[2] K. V. Price, R. M. Storn, and J. A. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Springer, 2005.

[3] U. K. Chakraborty, Ed., *Advances in Differential Evolution*. Springer, 2008.

[4] S. Das and P. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[5] Y. Wang, Z. Cai, and Q. Zhang, "Differential evolution with composite trial vector generation strategies and control parameters," *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 55–66, Feb. 2011.

[6] J. Liu and J. Lampinen, "A fuzzy adaptive differential evolution algorithm," *Soft Comput.*, vol. 9, no. 6, pp. 448–462, 2005.

[7] T. Takahama and S. Sakai, "Fuzzy c-means clustering and partition entropy for species-best strategy and search mode selection in nonlinear optimization by differential evolution," in *Proc. of the 2011 IEEE International Conference on Fuzzy Systems*, Jun. 2011, pp. 290–297.

[8] J. Teo, "Exploring dynamic self-adaptive populations in differential evolution," *Soft Comput.*, vol. 10, no. 8, pp. 673–686, 2006.

[9] A. Qin, V. Huang, and P. Suganthan, "Differential evolution algorithm with strategy adaptation for global numerical optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 398 –417, april 2009.

[10] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems," *IEEE Transaction on Evolutionary Computation*, vol. 10, no. 6, pp. 646–657, 2006.

[11] J. Zhang and A. C. Sanderson, "JADE: Adaptive differential evolution with optional external archive," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 5, pp. 945–958, Oct. 2009.

[12] S. M. Islam, S. Das, S. Ghosh, S. Roy, and P. N. Suganthan, "An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 42, no. 2, pp. 482–500, april 2012.

[13] T. Takahama and S. Sakai, "Constrained optimization by the ε constrained differential evolution with gradient-based mutation and feasible elites," in *Proc. of the 2006 IEEE Congress on Evolutionary Computation*, Jul. 2006, pp. 308–315.

[14] S. Kukkonen and J. Lampinen, "Constrained real-parameter optimization with generalized differential evolution," in *Proc. of the 2006 IEEE Congress on Evolutionary Computation*. Vancouver, BC, Canada: IEEE Press, 16-21 July 2006, pp. 207–214.

[15] T. Takahama and S. Sakai, "Solving difficult constrained optimization problems by the ε constrained differential evolution with gradient-based mutation," in *Constraint-Handling in Evolutionary Optimization*, E. Mezura-Montes, Ed. Springer-Verlag, 2009, pp. 51–72.

[16] Y.-W. Shang and Y.-H. Qiu, "A note on the extended rosenbrock function," *Evolutionary Computation*, vol. 14, no. 1, pp. 119–126, 2006.

[17] X. Yao, Y. Liu, , and G. Lin, "Evolutionary programming made faster," *IEEE Transactions on Evolutionary Computation*, vol. 3, pp. 82–102, 1999.

[18] X. Yao, Y. Liu, K.-H. Liang, and G. Lin, "Fast evolutionary algorithms," in *Advances in evolutionary computing: theory and applications*, A. Ghosh and S. Tsutsui, Eds. New York, NY, USA: Springer-Verlag New York, Inc., 2003, pp. 45–94.

[19] T. Takahama and S. Sakai, "Fast and stable constrained optimization by the ε constrained differential evolution," *Pacific Journal of Optimization*, vol. 5, no. 2, pp. 261–282, May 2009.