Adaptive Directional Mutation for an Adaptive Differential Evolution Algorithm

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

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

***Abstract*—Differential Evolution (DE) has been successfully applied to various optimization problems. The performance of DE is affected by algorithm parameters, mutation strategies and so on. One of the most successful studies on controlling the parameters is JADE. In this study, we focus on mutation strategies and propose an adaptive mutation strategy to improve JADE. Some directional mutation strategies, which utilize directional difference vectors from bad individuals to good individuals, have been proposed to improve search speed. However, in order to avoid local solutions, the vectors in opposite direction may be necessary. The proper frequency of such vectors depends on the problem and the search process. We propose an adaptive mutation strategy that controls the frequency of directional and opposite directional vectors adaptively to realizes efficient and stable search. The advantage of JADE with the proposed method is shown by solving thirteen benchmark problems.**

***Index Terms*—evolutionary algorithm, differential evolution, directional mutation, adaptive directional mutation**

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) such as genetic algorithms and differential evolution (DE). DE was proposed by Storn and Price [1] and has been successfully applied to various optimization problems [2]–[4]. It has been shown that DE is a very fast and robust algorithm.

The performance of DE is affected by algorithm parameters such as a scaling factor F, a crossover rate CR and population size, and by mutation strategies. Many studies have been done to control the parameters and the strategies. One of the most successful studies on controlling the parameters is JADE(adaptive DE with optional external archive) [5].

In this study, we focus on the mutation strategies and investigate a method to improve the search efficiency of JADE. In DE, a child is generated by crossing a parent with a mutant vector generated by a mutation operation. The mutant vector is obtained by perturbing a base vector with one or more difference vectors. The difference vectors are usually obtained by taking the difference between randomly selected individuals and the direction of the difference vectors is not considered. On the other hand, mutation strategies using directional difference vectors from bad individuals to good individuals have been proposed [6]–[8]. When the population is converging to the optimal solution, whether the directional difference vectors are effective or not depends on the position of the base vector. When the population is moving towards the optimal solution, the directional difference vectors are often effective moving directions as in Fig. 1, and the search efficiency will be improved. However, if only the directional difference vectors are used, there is a risk that the population moves in same direction repeatedly and the population will be trapped in a local solution.

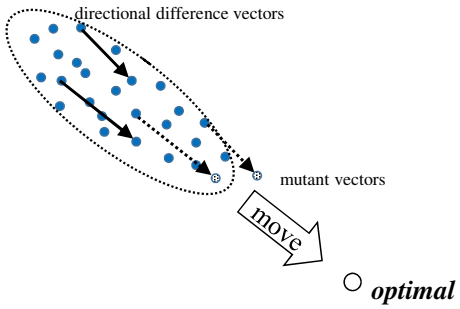

Fig. 1. Directional difference vectors when the population is moving towards the optimal solution.

In order to realize stable search, it is necessary to properly adjust the frequency of directional difference vectors and opposite directional difference vectors. It is thought that the proper frequency depends on the problem and the search process. In this study, the population is sorted by the objective function value to obtain the ranks of individuals. The adaptive directional mutation is realized by adaptively determining the range of ranks for starting individuals and ending individuals of the difference vectors. The advantage of JADE with the adaptive directional mutation is shown by solving thirteen benchmark problems.

In Section 2, optimization problems and DE are briefly explained. Related works including JADE are described in Section 3. In Section 4, JADE with adaptive directional mutation is proposed. The experimental results are shown in Section 5. Finally, conclusions are described in Section 6.

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.

$$\begin{array}{ll} \text{minimize} & f(\boldsymbol{x}) \\ \text{subject to} & l_j \leq x_j \leq u_j,\ j = 1, \ldots, D, \end{array} \tag{1}$$

where $\boldsymbol{x} = (x_1, x_2, \cdots, x_D)$ is a D dimensional vector and $f(\boldsymbol{x})$ is an objective function. The function f is a nonlinear real-valued function. Values l_j and u_j are the lower bound and the upper bound of x_j, respectively.

B. Differential Evolution

In DE, an initial population is formed by generating initial individuals randomly within given search space. Each individual $\boldsymbol{x}_i, i = 1, 2, \cdots, N$ contains D genes as decision variables, where N is the population size. At each generation, all individuals are selected as parents. Each parent $\boldsymbol{x}_i$ is processed as follows:

A mutation operation is performed by choosing several individuals from the population except for the parent $\boldsymbol{x}_i$. 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. For example, in case of the rand mutation strategy with one difference vector, the mutation operation is defined as follows:

$$\boldsymbol{m}_i = \boldsymbol{x}_{r1} + F(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}) \tag{2}$$

where three individuals $\boldsymbol{x}_{r1}$, $\boldsymbol{x}_{r2}$ and $\boldsymbol{x}_{r3}$ are chosen randomly from the population without overlapping $\boldsymbol{x}_i$ and each other, and $\boldsymbol{x}_{r1}$ is 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. In case of the binomial crossover, a child $\boldsymbol{v}_i$ is created as follows:

$$v_{ij} = \begin{cases} m_{ij} & \text{if } j = j_{rand} \text{ or } u(0,1) < CR \\ x_{ij} & otherwise \end{cases} \tag{3}$$

where j_{rand} is a randomly selected integer in $[1, D]$ and $u(0, 1)$ is a uniform random number in $[0, 1]$.

This crossover operation produces a child, or 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.

III. Related Works

The performance of DE is affected by algorithm parameters such as F, CR and N, and by mutation strategies.

A. Studies on Algorithm Parameters

The methods of controlling algorithm parameters can be classified into some categories as follows:

(1) selection-based control: Strategies and parameter values are selected regardless of current search state. CoDE [9] generates three trial vectors using three strategies with randomly selected parameter values from parameter candidate sets and the best trial vector is used for 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 [10] observes the movement of search points and the change of function values between successive generations, and controls F and CR. DESFC [11] adopts fuzzy clustering, observes partition entropy of search points, and controls CR and the mutation strategies between the rand and the species-best strategy. LMDE [12], [13] detects the landscape modality using the change of the objective values at sampling points along a line. Greedy parameter settings for local search are selected in unimodal landscape, and parameter settings for global search are selected in multimodal landscape.

(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. 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 [14] controls F, CR and N self-adaptively. SaDE [15] 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 [16] controls F and CR self-adaptively. JADE [5] and MDE_pBX [17] control the mean or power mean values of F and CR according to the mean values in success cases. CADE [18] introduces the correlation of F and CR to JADE.

B. Studies on Difference Vectors

There are some studies on difference vectors where some directional information is considered.

In [19], a trigonometric mutation operation (TDE) is proposed as follows:

$$\begin{aligned} \boldsymbol{m} &= \frac{1}{3}(\boldsymbol{x}_{r1} + \boldsymbol{x}_{r2} + \boldsymbol{x}_{r3}) \\ &\quad +(p_2 - p_1)(\boldsymbol{x}_{r1} - \boldsymbol{x}_{r2}) + (p_3 - p_2)(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}) \\ &\quad +(p_1 - p_3)(\boldsymbol{x}_{r3} - \boldsymbol{x}_{r1}) \end{aligned} \tag{4}$$

$$p_k = \frac{|f(\boldsymbol{x}_{rk})|}{|f(\boldsymbol{x}_{r1})| + |f(\boldsymbol{x}_{r2})| + |f(\boldsymbol{x}_{r3})|},\ k = 1, 2, 3 \tag{5}$$

where $\boldsymbol{m}$ is the mutant vector and integers $r1$, $r2$, $r3$ are randomly selected from $[1, N]$ without overlapping i and each other. If $\boldsymbol{x}_{r1}$ is better than $\boldsymbol{x}_{r2}$, the value of $(p_2 - p_1)$ becomes positive and the difference vector from a bad individual $\boldsymbol{x}_{r2}$ to a good individual $\boldsymbol{x}_{r1}$ is used. It is thought that this operation does not work well if $f(\cdot)$ is negative.

In [6], RAND/DIR and RAND/BEST/DIR strategies are proposed as follows: Randomly selected individuals $\{\boldsymbol{x}_i\}$ are divided into two classes C_+ and C_- where $\forall \boldsymbol{x}_i \in C_+$ and $\forall \boldsymbol{x}_j \in C_-$, $f(\boldsymbol{x}_i) \leq f(\boldsymbol{x}_j)$.

$$\begin{aligned} V_s^{\pm} &= \lambda(\boldsymbol{x}_{C_\pm}^{\min} - \boldsymbol{x}_{C_\pm}^{\max}),\ \lambda = \frac{1}{2}, \\ &\quad \boldsymbol{x}_{C_\pm}^{\min} = \arg\min_{\boldsymbol{x} \in C_\pm} f(\boldsymbol{x}),\ \boldsymbol{x}_{C_\pm}^{\max} = \arg\max_{\boldsymbol{x} \in C_\pm} f(\boldsymbol{x}) \end{aligned} \tag{6}$$

$$V_s = \frac{1}{2}(V_s^+ + V_s^-) \tag{7}$$

where $V_s^{\pm}$ is the shifts inside of each class, λ is influence constant and V_s is the average shift. The mutant vector is generated as follows:

$$\boldsymbol{m} = V_{C_+} + F(V_{C_+} - V_{C_-} + V_s), \text{ RAND/DIR} \quad (8)$$

$$\boldsymbol{m} = \boldsymbol{x}_{\text{best}} + F(V_{C_+} - V_{C_-} + V_s), \text{ RAND/BEST/DIR} \quad (9)$$

where V_{C_+} and V_{C_-} are barycenters of the C_+ and C_-, respectively.

In [7], SRDE (Stochastic ranking based Differential Evolution) is proposed to solve constrained optimization problems using stochastic ranking [20] as follows: Individuals in a population is sorted using the stochastic ranking and the sorted population is divided into two sets, upper individuals C_+ and lower individuals C_- according to the population partitioning factor which specify the rate of C_+ in the population. The mutant vector is generated as follows:

$$\boldsymbol{m} = \boldsymbol{x}_{r1} + F(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3}) \quad (10)$$

where $\boldsymbol{x}_{r1}$ is randomly selected from the population, $\boldsymbol{x}_{r2}$ is randomly selected from C_+ and $\boldsymbol{x}_{r3}$ is randomly selected from C_-.

In [21], we enhanced the idea of SRDE and proposed a mutation operation, which allows overlap between C_+ and C_- in order to use not only the directional difference vectors but also the difference vectors in the opposite direction. $\boldsymbol{x}_{r2}$ is selected from the first rank to the $\lfloor R_+ N \rfloor$-th rank individuals and $\boldsymbol{x}_{r3}$ is selected from the $\lfloor R_+(1-R_o)N+1 \rfloor$-th rank to the N-th rank individuals, where R_+ is the population partitioning rate, R_o is the overlapping rate and $\lfloor \cdot \rfloor$ is the round down to the nearest integer.

In this study, we rearrange the idea of [21] and propose adaptive control of difference vectors.

C. JADE

In JADE, the mean value of the scaling factor μ_F and the mean value of the crossover rate μ_{CR} are learned to define two probability density functions (PDFs), where initial values are μ_F=μ_{CR}=0.5. The scaling factor F_i and the crossover rate CR_i for each individual $\boldsymbol{x}_i$ are independently generated according to the two PDFs as follows:

$$F_i \sim C(\mu_F, \sigma_F) \quad (11)$$

$$CR_i \sim N(\mu_{CR}, \sigma_{CR}^2) \quad (12)$$

where F_i is a random variable according to a Cauchy distribution $C(\mu_F, \sigma_F)$ with a location parameter μ_F and a scale parameter σ_F=0.1. CR_i is a random variable according to a normal distribution $N(\mu_{CR}, \sigma_{CR}^2)$ of a mean μ_{CR} and a standard deviation σ_{CR}=0.1. CR_i is truncated to $[0, 1]$ and F_i is truncated to be 1 if $F_i > 1$ or regenerated if $F_i \leq 0$. The location μ_F and the mean μ_{CR} are updated as follows:

$$\mu_F = (1-c)\mu_F + cS_{F^2}/S_F \quad (13)$$

$$\mu_{CR} = (1-c)\mu_{CR} + cS_{CR}/S_N \quad (14)$$

where S_N is the number of success cases, S_F, S_{F^2} and S_{CR} are the sum of F, F^2 and CR in success cases, respectively. A constant c is a weight of update in (0,1] and the recommended value is 0.1.

JADE adopts a strategy called "current-to-pbest" as follows: A mutation vector is generated as follows:

$$\boldsymbol{m}_i = \boldsymbol{x}_i + F_i(\boldsymbol{x}_{pbest} - \boldsymbol{x}_i) + F_i(\boldsymbol{x}_{r2} - \widetilde{\boldsymbol{x}}_{r3}) \quad (15)$$

where $\boldsymbol{x}_{pbest}$ is a randomly selected individual from the top $100p\%$ individuals. $\widetilde{\boldsymbol{x}}_{r3}$ is selected randomly from the current population in case of "without an archive", and is selected randomly from the union of the current population and an archive in case of "with an archive". The archive is initialized to be empty. Defeated parents by the children are added to the archive. If the number of archived individuals exceeds the archive size, randomly selected individuals are removed to keep the archive size.

In order to satisfy bound constraints, a child that is outside of the search space is moved into the inside of the search space. In JADE, each outside element of the child is set to be the middle between the corresponding boundary and the element of the parent as follows:

$$v_{ij} = \begin{cases} \frac{1}{2}(l_j + x_{ij}) & \text{if } v_{ij} < l_j \\ \frac{1}{2}(u_j + x_{ij}) & \text{if } v_{ij} > u_j \end{cases} \quad (16)$$

This operation is applied after a new point is generated by JADE operations.

IV. Proposed method

A. A Mutation Strategy with Omnidirectional to Directional Difference Vectors

In this study, a new mutation strategy, which is the modified version of [21], is proposed as follows:

1) The indexes I of the individuals in the population P, $I = \{1, 2, \cdots, N\}$ is sorted according to the objective values f.
2) Let I_2 be the indexes of individuals whose rank according to f is 1 to $r_2^{\max}$, which is the set of the first element to the $r_2^{\max}$-th element of I. When $r_2^{\max}$=N, I_2 contains all indexes. When $r_2^{\max}$ is small, I_2 includes only high-ranking indexes.
3) Let I_3 be the indexes of individuals whose rank is $r_3^{\min}$ to N, which is the set of the $r_3^{\min}$-th element to the N-th element of I. When $r_3^{\min}$=1, I_3 contains all indexes. When $r_3^{\min}$ is large, I_3 includes only low-ranking individuals.
4) $\boldsymbol{m}_i = \boldsymbol{x}_i + F_i(\boldsymbol{x}_{pbest} - \boldsymbol{x}_i) + F_i(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3})$ where $r2$ and $r3$ are randomly selected from I_2 and I_3, respectively.

Instead of using the ranks in $[1, N]$, the rates of the ranks in $[0, 1]$ can be used. The $r_2^{\max}$ can be specified by R_2 which is the rate of the $r_2^{\max}$. The $r_3^{\min}$ can be specified by R_3 which is the rate of the $r_3^{\min}$. If $R_2 = 1$ ($r_2^{\max} = N$) and $R_3 = 0$ ($r_3^{\min} = 1$), I_2 and I_3 are the set of all indexes, $\boldsymbol{x}_{r2}$ and $\boldsymbol{x}_{r3}$ are randomly selected from the population, and the difference vector is omnidirectional which is equivalent to that of original JADE. If $R_2 = 0.5$ and $R_3 = 0.5$, the difference vector is directional which is almost equivalent to that of SRDE except that $(\frac{N}{2} + 1)$-th element is included in both of I_2 and I_3.

Fig. 2 shows the range of ranks for $\boldsymbol{x}_{r2}$ and $\boldsymbol{x}_{r3}$ when (R_2, R_3) is in $\{(1,0),(0.6,0.4),(0.5,0.5)\}$. The mutation is omnidirectional in case of (1,0), is directional in case of (0.5,0.5) and is partly directional with overlapping in case of (0.6,0.4). In this study, an overlapping rate is defined as $R_2 - R_3$. The overlapping rate is 1 in the omnidirectional case (1,0), is zero or negative in directional cases such as (0.5,0.5). is positive in the partly directional cases such as (0.6,0.4).

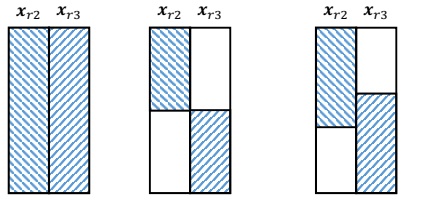

Fig. 2. The range of ranks for 3 cases.

In order to maintain the diversity for $r2$ and $r3$, the minimum size of I_2 and I_3 is set to $\min_R$=3 and the ranges of indexes in I for I_2 and I_3 are modified accordingly. The detail is explained in the next subsection.

B. Adaptive Directional Mutation

In adaptive directional mutation (ADM), the mean values of the two parameters $\mu_{R_2}, \mu_{R_3} \in [0,1]$ are learned, where initial values are $\mu_{R_2} = 1$ and $\mu_{R_3} = 0$. The values $R_{2i}, R_{3i} \in [0,1]$ for each individuals $\boldsymbol{x}_i$ are independently generated according to μ_{R_2} and μ_{R_3} as follows:

$$R_{2,i} \sim N(\mu_{R_2}, \sigma_R^2) \text{ truncated to } [\min_R/N, 1] \quad (17)$$
$$R_{3,i} \sim N(\mu_{R_3}, \sigma_R^2) \text{ truncated to } [0, 1 - \min_R/N] \quad (18)$$

where σ_R is the standard deviation.

$r_2^{\max}$ and $r_3^{\min}$ are determined as follows:

$$r_{2,i}^{\max} = \lfloor R_{2,i} N + 1 \rfloor \text{ truncated to } N \quad (19)$$
$$r_{3,i}^{\min} = \lfloor R_{3,i} N + 1 \rfloor \text{ truncated to } N \quad (20)$$

$r2$ and $r3$ are randomly selected integers from the first to the $r_{2,i}^{\max}$-th and the $r_{3,i}^{\min}$-th to the N-th elements of I, respectively.

The mean values are updated as follows:

$$\mu_{R_2} = (1-c)\mu_{R_2} + cS_{R_2}/S_N \quad (21)$$
$$\mu_{R_3} = (1-c)\mu_{R_3} + cS_{R_3}/S_N \quad (22)$$

where S_{R_2} and S_{R_3} are the sum of $R_{2,i}$ and $R_{3,i}$ in success cases, respectively.

C. Algorithm

The algorithm of the proposed method JADEadm (JADE with adaptive directional mutation) can be described as follows:

Step0 Parameter setup. The size of the archive N_A is specified as zero (without the archive) or N (with the archive). The initial mean values μ_{R_2}=1 and μ_{R_3}=0, $\min_R$=3 and the standard deviation σ_R is specified.

Step1 Initialization of the individuals. $P = \{\boldsymbol{x}_i | i = 1, 2, \cdots, N\}$ are generated randomly in the search space and form an initial population. The archive A is made empty.

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

Step3 Initialization for each generation. The list of success cases S is made empty. The set of indexes $I = \{1, 2, \cdots, N\}$ of the individuals in P is sorted according to the objective values.

Step4 JADE operation with adaptive directional mutation. For each individual $\boldsymbol{x}_i$, F_i is generated according to Eq.(11) and CR_i is generated according to Eq.(12). $r_{2,i}^{\max}$ and $r_{3,i}^{\min}$ are generated according to Eqs.(17), (19) and Eqs.(18), (20), and I_2 and I_3 are determined, respectively. $r2$ and $r3$ are randomly selected from I_2 and I_3, respectively. JADE operation is executed and a new child is generated.

Step5 Survivor selection. If the child is better than the parent, or in a success case, the child becomes a survivor. The successful combination of parameter values $(F_i, CR_i, R_{2,i}, R_{3,i})$ is added to S. Defeated parent is added to A. Otherwise, the parent $\boldsymbol{x}_i$ becomes a survivor. Go back to Step 4 until all individuals are processed.

Step6 Resizing the archive. If the size of the archive exceeds N_A, randomly selected elements in A are deleted until the size of A becomes N_A.

Step7 Learning parameters. The means of the scaling factor μ_F and the means of crossover rate μ_{CR} are updated using S according to Eqs. (13) and (14). The means for adaptive directional mutation μ_{R_2} and μ_{R_3} are updated using S according to Eqs. (21) and (22). Go back to Step2.

Figure 3 shows the pseudo-code of JADEadm. Lines starting with '+' shows the modified lines from original JADE.

V. Numerical Experiments

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

A. Test Problems

The 13 scalable benchmark functions [5] are solved. Every function has an optimal objective value 0. 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. 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.

Experimental conditions are same as JADE as follows: Population size $N = 100$, initial mean for scaling factor μ_F = 0.5 and initial mean for crossover rate μ_{CR} = 0.5, the pbest parameter p = 0.05, and the learning parameter c = 0.1. The

```
JADEadm()
{
  μ_F=μ_CR=0.5; σ_F=σ_CR=0.1; N_A=0 or N; // archive size
+ μ_R2=1; μ_R3=0; min_R=3; σ_R is specified;
// Initialize a population
  P=N individuals generated randomly in the search space;
  FE=N;
  A=∅;
  for(t=1; FE < FE_max; t++) {
// JADE with adaptive directional mutation
    S=∅;
    Indexes I is sorted according to objective values of P;
    for(i=1; i ≤ N; i++) {
      CR_i = μ_CR + N(0, σ²_CR);
      truncate CR_i to [0, 1];
      do {
        F_i=μ_F + C(0, σ_F);
      } while(F_i ≤ 0);
      if(F_i > 1) F_i = 1;
+     R_2,i=μ_R2 + N(0, σ²_R) truncated to [min_R /N, 1];
+     R_3,i=μ_R3 + N(0, σ²_R) truncated to [0, 1 − min_R /N];
+     r^max_2,i=⌊R_2,i N + 1⌋ truncated to N;
+     r^min_3,i=⌊R_3,i N + 1⌋ truncated to N;
+     I_2=the first to the r^max_2,i-th elements of I;
+     I_3=the r^min_3,i-th to the N-th elements of I;
      pbest=Randomly selected from top 100p% in I;
+     r2=Randomly selected from I_2 (r2 ∉ {i});
+     r3=Randomly selected from I_3 ∪ Â (r3 ∉ {i, r2});
      m_i=x_i+F_i(x_pbest − x_i)+F_i(x_r2 − x_r3);
      v_i=generated from x_i and m_i by crossover;
      FE=FE+1;
// Survivor selection
      if(f(v_i) < f(x_i)) {
        z_i=v_i;
        S=S ∪ {(F_i, CR_i, R_2,i, R_3,i)}; // add a success case
        if(N_A>0) A=A ∪ {x_i};
      }
      else z_i = x_i;
    }
    P = {z_i};
// Resizing the archive
    while(|A|>N_A)
      remove a randomly selected element from A;
// Learning parameters
    if(|S| > 0) {
      μ_F = (1 − c)μ_F + c Σ_{F_i∈S} F_i² / Σ_{F_i∈S} F_i;
      μ_CR = (1 − c)μ_CR + c Σ_{CR_i∈S} CR_i/|S|;
+     μ_R2 = (1 − c)μ_R2 + c Σ_{R_2,i∈S} R_2,i/|S|;
+     μ_R3 = (1 − c)μ_R3 + c Σ_{R_3,i∈S} R_3,i/|S|;
    }
  }
}
```

Fig. 3. The pseudo-code of the proposed method JADEadm where $\hat{A}$ is indexes of A.

archive size N_A is selected from $\{0, N\}$, where N_A=0 means that the algorithm is executed without the archive. As for JADEadm, initial means μ_{R_2}=1 and μ_{R_3}=0, and σ_R is selected from $\{0.1, 0.2\}$.

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

B. Experimental Results

Table I show the experimental results on JADE and JADEadm (σ_R=0.1 and 0.2) with and without the archive. The mean value, the standard deviation, and the median value of best objective values in 50 runs are shown for each function. The median value is shown under the mean value. The maximum number of function evaluations is selected for each function and is shown in column labeled $FE_{\max}$. Since the variability of each trial is not small and the reliability of the mean value is not high, the best median value among algorithms is highlighted. Also, Wilcoxon signed rank test is performed and the result for each function is shown on the right side of the median value in parentheses. Symbols '+', '−' and '=' are shown when each algorithm is significantly better than JADEadm (σ_R=0.2 without the archive), is significantly worse than the JADEadm, and is not significantly different from the JADEadm, respectively. Symbols '++' and '−−' show that the significance level is 1% and '+' and '−' show that the significance level is 5%.

From Table I, JADEadm (σ_R=0.2 without the archive) attained the best median results in 7 functions f_1, f_2, f_6 and f_{10}–f_{13} out of 13 functions. JADEadm (σ_R=0.2 with the archive) attained the best median results in 5 functions f_3–f_5, f_8 and f_9. JADEadm (σ_R=0.1 without the archive) attained the best median results in 2 functions f_6 and f_7. JADE without the archive attained the best median result in f_6.

JADEadm (σ_R=0.2 without the archive) attained significantly better results than JADE without the archive in 10 functions except for f_6, f_7 and f_9 and attained no significantly worse result than the JADE. The JADEadm attained significantly better results than JADE with the archive in 12 functions except for f_7 and attained no significantly worse result. The JADEadm attained significantly better results than JADEadm (σ_R=0.1 without the archive) in 8 functions and no significantly worse result. The JADEadm attained significantly better results than JADEadm (σ_R=0.1 with the archive) in 7 functions and attained 3 significantly worse results. The JADEadm attained significantly better results than JADEadm (σ_R=0.2 with the archive) in 7 functions and attained 5 significantly worse results. Thus, it is thought that JADEadm (σ_R=0.2 without the archive) is the best method followed by JADEadm (σ_R=0.2 with the archive). It is shown that the idea of adaptive directional mutation is very effective and improve the performance of JADE.

Figure 4 shows the change of μ_{R_2} and μ_{R_3} over the number of function evaluations in case of JADEadm (σ_R=0.2 and 0.1 without the archive) for a unimodal function f_1, a function with ridge structure f_5, a discrete function f_6 and a multimodal function f_9. In all graphs, the value of μ_{R_2} gradually decreases from 1 and the value of μ_{R_3} gradually increases from 0. Also, the overlapping rate, $\mu_{R_2} - \mu_{R_3}$ in this case, gradually decreases from 1 to a smaller value.

Compared JADEadm (σ_R=0.1 without the archive) with JADEadm (σ_R=0.2 without the archive), the changing speed

TABLE I
EXPERIMENTAL RESULTS ON 13 FUNCTIONS

Func	FEmax	JADEadm 0.2 w/o A	JADE w/o A	JADE w A	JADEadm 0.1 w/o A	JADEadm 0.1 w A	JADEadm 0.2 w A
f_1	150000	1.45e-64 ± 9.8e-64	9.38e-59 ± 6.5e-58	6.50e-58 ± 4.5e-57	6.25e-65 ± 3.2e-64	8.76e-61 ± 5.9e-60	1.33e-62 ± 5.7e-62
		1.98e-72	4.71e-66 (−−)	1.13e-63 (−−)	2.56e-70 (=)	3.57e-67 (−−)	2.27e-67 (−−)
f_2	200000	3.61e-29 ± 2.5e-28	4.19e-31 ± 2.4e-30	2.21e-24 ± 1.2e-23	5.01e-29 ± 2.6e-28	1.13e-24 ± 4.4e-24	1.58e-24 ± 1.1e-23
		3.29e-41	1.96e-37 (−−)	6.73e-33 (−−)	5.73e-41 (=)	1.56e-35 (−−)	1.02e-38 (−−)
f_3	500000	3.69e-88 ± 1.2e-87	8.17e-62 ± 3.0e-61	2.29e-83 ± 1.1e-82	2.73e-83 ± 1.1e-82	7.91e-95 ± 5.4e-94	6.77e-97 ± 4.0e-96
		4.47e-91	2.30e-63 (−−)	7.25e-86 (−−)	5.64e-86 (−−)	3.53e-99 (++)	**4.07e-101** (++)
f_4	500000	4.17e-63 ± 2.0e-62	2.01e-23 ± 9.8e-23	1.58e-62 ± 4.3e-62	3.47e-53 ± 1.3e-52	1.40e-75 ± 5.8e-75	2.66e-77 ± 9.4e-77
		5.51e-65	9.27e-26 (−−)	1.15e-63 (−−)	8.87e-55 (−−)	3.73e-77 (++)	**1.40e-78** (++)
f_5	150000	2.39e-01 ± 9.5e-01	5.83e-01 ± 3.6e+00	2.39e-01 ± 9.5e-01	1.59e-01 ± 7.8e-01	7.97e-02 ± 5.6e-01	7.97e-02 ± 5.6e-01
		9.24e-26	3.04e-09 (−−)	1.82e-19 (−−)	1.61e-18 (−−)	4.27e-27 (++)	**4.52e-30** (++)
f_6	10000	2.92e+00 ± 1.2e+00	3.02e+00 ± 1.3e+00	4.92e+00 ± 1.4e+00	2.88e+00 ± 1.2e+00	3.94e+00 ± 1.5e+00	3.98e+00 ± 1.4e+00
		3.00e+00	**3.00e+00** (=)	5.00e+00 (−−)	**3.00e+00** (=)	4.00e+00 (−−)	4.00e+00 (−−)
f_7	300000	5.41e-04 ± 1.7e-04	6.04e-04 ± 2.4e-04	6.24e-04 ± 2.5e-04	5.40e-04 ± 2.5e-04	6.10e-04 ± 2.5e-04	5.50e-04 ± 1.8e-04
		5.28e-04	5.78e-04 (=)	5.65e-04 (=)	**5.01e-04** (=)	5.79e-04 (=)	5.20e-04 (=)
f_8	100000	7.11e+00 ± 2.8e+01	2.37e+00 ± 1.7e+01	7.11e+00 ± 2.8e+01	9.48e+00 ± 4.0e+01	7.11e+00 ± 2.8e+01	9.48e+00 ± 3.2e+01
		1.22e-05	2.87e-05 (−−)	3.70e-05 (−−)	2.08e-05 (−−)	1.52e-05 (=)	**3.13e-06** (++)
f_9	100000	1.04e-04 ± 5.8e-05	1.01e-04 ± 3.9e-05	1.34e-04 ± 7.2e-05	1.43e-04 ± 6.6e-05	9.55e-05 ± 4.7e-05	4.17e-05 ± 2.2e-05
		8.82e-05	9.19e-05 (=)	1.18e-04 (−)	1.40e-04 (−−)	8.28e-05 (=)	**3.50e-05** (++)
f_{10}	50000	3.16e-10 ± 3.0e-10	9.20e-10 ± 6.4e-10	2.87e-09 ± 4.8e-09	4.22e-10 ± 3.0e-10	1.44e-09 ± 1.3e-09	9.18e-10 ± 7.4e-10
		1.94e-10	7.15e-10 (−−)	1.88e-09 (−−)	3.42e-10 (−)	9.66e-10 (−−)	6.75e-10 (−−)
f_{11}	40000	1.13e-11 ± 5.6e-11	3.17e-07 ± 1.6e-06	1.71e-07 ± 1.2e-06	3.34e-09 ± 2.3e-08	2.56e-07 ± 1.8e-06	1.03e-11 ± 3.7e-11
		2.98e-13	2.55e-12 (−−)	4.68e-12 (−−)	7.65e-13 (=)	1.42e-12 (−−)	1.46e-12 (−)
f_{12}	50000	1.22e-18 ± 2.2e-18	2.40e-16 ± 1.6e-15	3.20e-16 ± 1.1e-15	2.07e-03 ± 1.5e-02	4.39e-17 ± 1.7e-16	1.72e-17 ± 5.6e-17
		2.11e-19	2.27e-18 (−−)	1.43e-17 (−−)	5.89e-19 (−)	3.28e-18 (−−)	2.13e-18 (−−)
f_{13}	50000	1.15e-17 ± 2.9e-17	1.15e-16 ± 2.2e-16	7.98e-16 ± 1.4e-15	3.45e-17 ± 1.3e-16	1.27e-16 ± 2.5e-16	9.03e-17 ± 2.6e-16
		3.03e-18	2.69e-17 (−−)	3.03e-16 (−−)	5.52e-18 (−)	3.13e-17 (−−)	1.70e-17 (−−)
+		—	0	0	0	3	5
=		—	3	1	5	3	1
−		—	10	12	8	7	7

of μ_{R_2} and μ_{R_3} is much faster at σ_R=0.2 than at σ_R=0.1. It is thought that the reason why JADEadm with σ_R=0.2 was better than JADEadm with σ_R=0.1 is that generating the values of $R_{2,i}$ and $R_{3,i}$ in a wide range could find the appropriate values of μ_{R_2} and μ_{R_3} quickly.

As for the unimodal function f_1, in case of σ_R=0.2, the overlapping rate at 100,000 evaluations was about 0.044. It is thought that the mutation is changed from omnidirectional to almost directional and the search efficiency was improved.

As for the function with ridge structure f_5, in case of σ_R=0.2, the overlapping rate at 100,000 evaluations was about -0.025. It is thought that the mutation is changed from omnidirectional to directional completely. In the ridge structure, the individuals need to move through a thin path and approach the optimal solution. The directional mutation supported this move and the search efficiency was improved.

As for the discrete function f_6, the function value is a discrete value, and many individuals will take the same function value as the search progresses. When all individuals are sorted according to the function value, even the same function value will be ranked differently, and the ranking will be inaccurate. Since the exact direction from good individual to bad individual is not known, the overlapping rate remains high, where the overlapping rate at 100,000 evaluations was about 0.425 in case of σ_R=0.2. The improvement of search efficiency is not large.

As for the multimodal function f_9, in case of σ_R=0.2, the overlapping rate at 100,000 evaluations was about 0.069. The rates in f_1 and f_9 were similar, but the rate in the multimodal function f_9 is slightly higher than the rate in the unimodal function f_1. In multimodal functions, the individuals need higher diversity than in unimodal problems in order to avoid local solutions. Therefore, the overlapping rate is larger than in unimodal functions.

VI. CONCLUSION

In this study, a new mutation strategy including from omnidirectional to directional mutation and adaptive control of the strategy was proposed to improve the performance of JADE. It has been shown that using a directional difference vector from a bad individual to a good individual is effective to enforce convergence speed. However, there is a risk of premature convergence. Therefore, we enhanced the idea of SRDE, where difference vectors from lower rank individuals to higher rank individuals were used, so as to permit overlap between the higher and the lower rank individuals. The two adaptive parameters μ_{R_2} and μ_{R_3} were introduced to generate the parameters $r_2^{\max}$ and $r_3^{\min}$, which realize various mutation from omnidirectional to directional, based on the values $R_{2,i}$ and $R_{3,i}$ in success cases. The proposed method was applied the optimization of various 13 functions including unimodal functions, ridge functions and multimodal functions. It was shown that the proposed method JADEadm was very efficient compared with JADE.

In the future, we will investigate the effect of algorithm parameters σ_R in detail. Also, we will introduce the ε constrained method into JADEadm to solve constrained optimization problems.

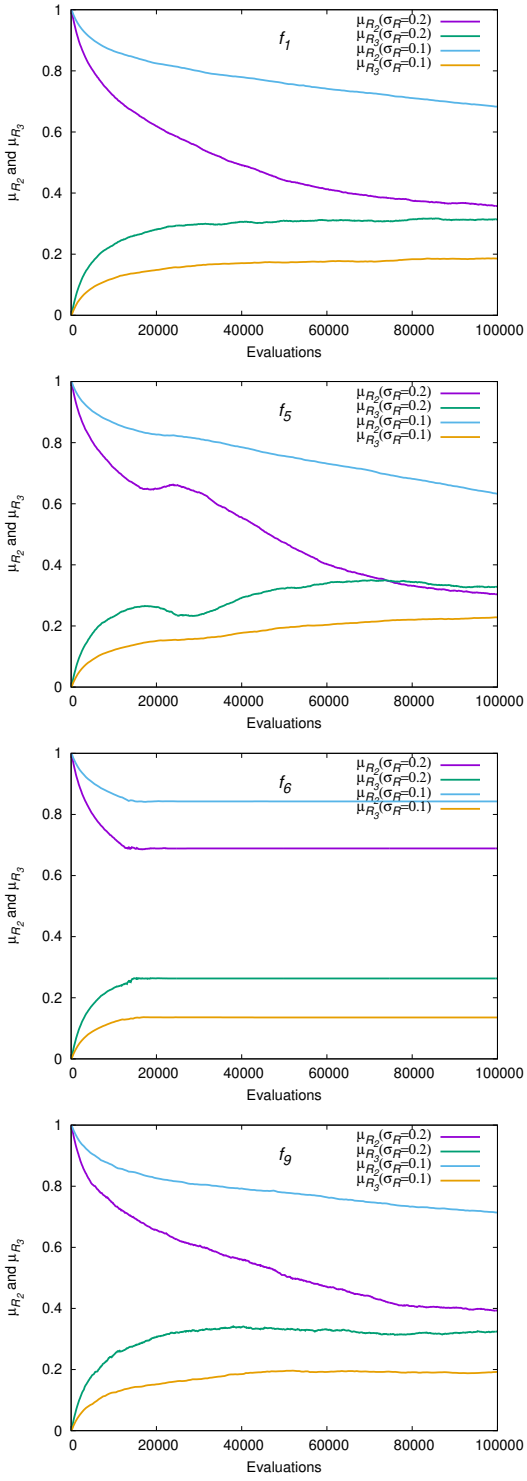

Fig. 4. The graphs of μ_{R_2} and μ_{R_3} over the number of function evaluations.

Acknowledgment

This study is supported by JSPS KAKENHI Grant Numbers 17K00311 and 19K04916.

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. Price, R. 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] 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.

[6] V. Feoktistov and S. Janaqi, "Generalization of the strategies in differential evolution," in *Proceedings of the 18th International Parallel and Distributed Processing Symposium*, April 2004, pp. 165–170.

[7] J. Liu, Z. Fan, and E. D. Goodman, "SRDE: an improved differential evolution based on stochastic ranking," in *GEC Summit*, 2009.

[8] Y. Cai and J. Wang, "Differential evolution with neighborhood and direction information for numerical optimization," *IEEE Transactions on Cybernetics*, vol. 43, no. 6, pp. 2202–2215, Dec 2013.

[9] 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.

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

[11] 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.

[12] T. Takahama and S. Sakai, "Differential evolution with dynamic strategy and parameter selection by detecting landscape modality," in *Proc. of the 2012 IEEE Congress on Evolutionary Computation*, Jun. 2012, pp. 2114–2121.

[13] T. Takahama and S. Sakai, "Large scale optimization by differential evolution with landscape modality detection and a diversity archive," in *Proc. of the 2012 IEEE Congress on Evolutionary Computation*, Jun. 2012, pp. 2842–2849.

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

[15] 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, Apr. 2009.

[16] 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, Dec. 2006.

[17] 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, Apr. 2012.

[18] T. Takahama and S. Sakai, "An adaptive differential evolution considering correlation of two algorithm parameters," in *Proc. of the Joint 7th International Conference on Soft Computing and Intelligent Systems and 15th International Symposium on Advanced Intelligent Systems (SCIS&ISIS2014)*, Dec. 2014, pp. 618–623.

[19] H.-Y. Fan and J. Lampinen, "A trigonometric mutation operation to differential evolution," *Journal of Global Optimization*, vol. 27, no. 1, pp. 105–129, Sep 2003.

[20] T. P. Runarsson and X. Yao, "Stochastic ranking for constrained evolutionary optimization," *IEEE Trans. on Evolutionary Computation*, vol. 4, no. 3, pp. 284–294, Sep. 2000.

[21] S. Sakai and T. Takahama, "A study on a directional mutation operation for an adaptive differential evolution," in *Current Researches for Applied Economics, Information Systems, Mathematics and OR*, A.Kadoya and J.Maeda, Eds. Kyushu University Press, Feb. 2020, pp. 59–71.