Clustering of Moving Vectors for Evolutionary Computation

Yu, Jun
Graduate School of Design, Kyushu University

Takagi, Hideyuki
Faculty of Design, Kyushu University

出版情報：International Conference on Soft Computing and Pattern Recognition. 7, pp.169-174, 2015-11-13. IEEE SMC Japan Chapter
バージョン：
権利関係：

Clustering of Moving Vectors for Evolutionary Computation

Jun Yu
Graduate School of Design, Kyushu University
Fukuoka, Japan
Email: yujun901101@yahoo.co.jp

Hideyuki Takagi
Faculty of Design, Kyushu University
Fukuoka, Japan
URL: http://www.design.kyushu-u.ac.jp/~takagi/

***Abstract*—We propose a method for clustering moving vectors oriented around two different local optima and some methods for improving the clustering performance. Evolutionary computation is an optimization method for finding the global optimum iteratively using multiple individuals; we propose a method for estimating the global optimum mathematically using the moving vectors between parent individuals and their offspring. Our proposed clustering method is the first to tackle the extension of the estimation method to multimodal optimization. We describe the algorithm of the clustering method, the improvements made to the method, and the estimation performance for two local optima.**

***Keywords*-evolutionary computation, moving vectors, clustering, multi-modal functions, optimization;**

I. Introduction

A common feature of all evolutionary computation (EC) is individual-based optimization. Individuals converge to the global optimum using fitness information. Although many researchers focus on developing or improving algorithms for generating offspring from parent individuals, the direction of evolution from a parent to its offspring offers useful gradient information. Let's define a moving vector $\boldsymbol{b}_i = \boldsymbol{c}_i - \boldsymbol{a}_i$, where $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$ are the i-th parent individual and its offspring. As the moving vectors from parent generations to their offspring generations and their evolutionary paths over several EC generations includes information leading towards local optima (see Figure 1), it must be possible to to make EC improvements by focusing on the directions of these moving vectors.

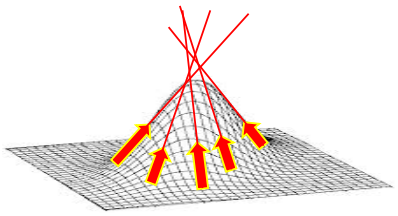


Figure 1. Moving vectors from parent individuals to their offspring individuals converge to one point near the global optimum in the d-dimensional searching space.

Using the multiple moving vectors between parent individuals and their offspring, it is possible to mathematically estimate the point of their convergence [2], [3]. The calculated convergence point is expected to be locate near the global optimum, and EC acceleration can be expected to be achieved by using the calculated convergence point as an elite individual.

However, this basic method is targeted at unimodal optimization tasks and may not work well for optimization tasks with multiple local optima. In this case, the reliability of the convergence point calculated from all moving vectors aiming at different local optima becomes low. To extend its applicability to multimodal tasks, it is necessary to cluster moving vectors with regards to their directions towards to their local optima.

The objective of this paper is to propose a clustering method for use with EC moving vectors. As the first step of this research, we start by developing a clustering method using d-dimensional bipolar tasks, i.e. optimization tasks having two local optima.

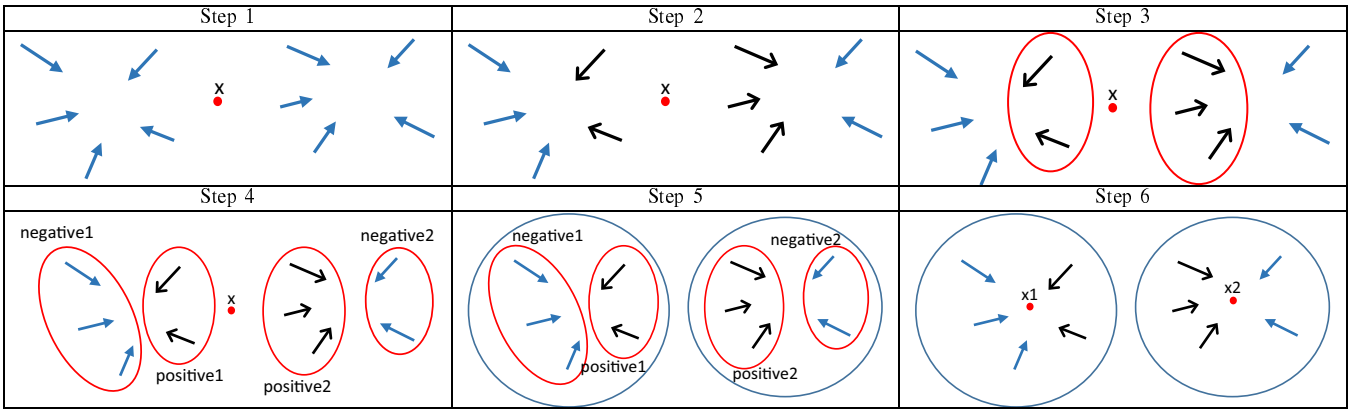
We first summarize the method by which the convergence point is calculated from moving vectors [2], [3] in Section II, then propose a basic algorithm for clustering moving vectors towards the same local optimum in Section III, and propose four methods for improving the performance of the clustering method in Section IV. We evaluate the total clustering performance using tasks of different dimensionality in Section V.

II. Method for Calculating the Convergence Point of Moving Vectors

The convergence point for moving vectors from parent individuals to their offspring in the next EC search generation can be calculated mathematically [2], [3]. This method is used in our proposed clustering method in the following section. First of all, let's begin by defining symbols. $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$ in Figure 2 are the i-th parent individual and its offspring individual, respectively ($\boldsymbol{a}_i, \boldsymbol{c}_i \in \mathbb{R}^d$). Then, the i-th moving vector is defined as a direction vector, $\boldsymbol{b}_i = \boldsymbol{c}_i - \boldsymbol{a}_i$. The unit direction vector of the $\boldsymbol{b}_i$ is given as $\boldsymbol{b}_{0i} = \boldsymbol{b}_i / ||\boldsymbol{b}_i||$, i.e. $\boldsymbol{b}_{0i}^T \boldsymbol{b}_{0i} = 1$.

Let $\boldsymbol{x} \in \mathbb{R}^d$ be the point that is the nearest to the n extended directional line segments, $\boldsymbol{a}_i + t_i \boldsymbol{b}_i$ ($t_i \in R$). Here, *nearest* means that the total distance from $\boldsymbol{x}$ to the

Table I
PROCESS OF CLUSTERING MOVING VECTORS FOR BIPOLAR OPTIMIZATION TASKS.



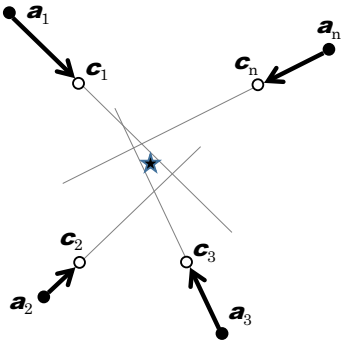


Figure 2. Moving vector $\boldsymbol{b}_i$ $(= \boldsymbol{c}_i - \boldsymbol{a}_i)$ is calculated from a parent individual $\boldsymbol{a}_i$ and its offspring $\boldsymbol{c}_i$ in the d-dimensional searching space. The $\star$ mark is the convergence point of these moving vectors.

n extended directional line segments, $\mathrm{J}(\boldsymbol{x}, \{t_i\})$ in Eq.1, becomes the minimum.

As the minimum line segment from the convergence point $\boldsymbol{x}$ to the extended directional line segments is the orthogonal projection from $\boldsymbol{x}$, we may insert an orthogonal condition, Eq. (2), into the Eq. (1) and delete t_i.

$$\mathrm{J}(\boldsymbol{x}, \{t_i\}) = \sum_{i=1}^{n} \|\boldsymbol{a}_i + t_i \boldsymbol{b}_i - \boldsymbol{x}\|^2 \tag{1}$$

$$\boldsymbol{b}_i^{\mathrm{T}}(\boldsymbol{a}_i + t_i \boldsymbol{b}_i - \boldsymbol{x}) = 0 \quad \text{(orthogonal condition)} \tag{2}$$

The $\hat{\boldsymbol{x}}$ that minimizes the total distance Eq. (1) is obtained by partially differentiating each element of $\boldsymbol{x}$ and setting them equal 0. Finally, the convergence point $\hat{\boldsymbol{x}}$ is given by the Eq. (3), where $\boldsymbol{I}_d$ is a unit matrix.

$$\hat{\boldsymbol{x}} = \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \right\}^{-1} \left\{ \sum_{i=1}^{n} \left(\boldsymbol{I}_d - \boldsymbol{b}_{0i} \boldsymbol{b}_{0i}^{\mathrm{T}} \right) \boldsymbol{a}_i \right\} \tag{3}$$

It may not be easy to solve Eq. (3) in some programming languages. An approximate solution and an iterative solution for cases where matrix operations cannot easily be performed were also presented in references [2], [3]. We use iterative methods in our experiment in Section V.

III. PROPOSAL OF A CLUSTERING METHOD FOR MOVING VECTORS TOWARDS TO LOCAL MINIMA

The method for calculating a convergence point mentioned in Section II is effective for unimodal tasks [2], [3], but it is not always valid for multimodal tasks because the moving vectors point towards different local optima. To make this method applicable to general optimization tasks, we must cluster moving vectors with regards to their directions toward local minima first and then apply the method to estimate each local minimum. This method has the potential to be a new EC niche method.

As the first step of this challenge, we start with bipolar tasks and develop a clustering method for moving vectors to estimate two local minima. Here, we explain our proposed clustering method in six steps which are illustrated in the the six images of Table I.

Step 1: Estimate the convergence point $\boldsymbol{x}$ using the method described in Section II and all moving vectors.

Step 2: Divide the moving vectors into two groups; one for the moving vectors coming towards the obtained convergence point $\boldsymbol{x}$ and the other group for those going in opposite directions from $\boldsymbol{x}$. This clustering is conducted based on the signs of the inner product of the moving vector and the vector from $\boldsymbol{x}$ to the terminal point of the moving vector. When $\boldsymbol{b}_i^{\mathrm{T}} \cdot (\boldsymbol{c}_i - \boldsymbol{x})$ is positive, the directions of two vectors are the same, i.e. the moving vector goes opposite direction from the $\boldsymbol{x}$; when it is negative, the moving vector comes towards $\boldsymbol{x}$. Let's refer to these two groups as positive class and negative class.

Step 3: Let the farthest two terminal points among all moving vectors be $\boldsymbol{c}^1$ and $\boldsymbol{c}^2$. The positive class can

be further divided into two sub-positive groups by checking which direction the moving vectors go towards, $\boldsymbol{c}^1$ or $\boldsymbol{c}^2$. We use $d1 = ||\boldsymbol{c}^1 - \boldsymbol{a}_i||/||\boldsymbol{c}^1 - \boldsymbol{c}_i||$ and $d2 = ||\boldsymbol{c}^2 - \boldsymbol{a}_i||/||\boldsymbol{c}^2 - \boldsymbol{c}_i||$ for this clustering. When $1 < d1$ and $d2 < 1$, the direction of the ith moving vector, $\boldsymbol{b}_i$, is towards $\boldsymbol{c}^1$, and in the opposite case, $\boldsymbol{b}_i$ goes towards $\boldsymbol{c}^2$. When both $d1$ and $d2$ are smaller or bigger than 1, we decide that its direction is towards the bigger one by comparing $d1$ and $d2$. Alternatively, we can ignore the moving vector for our calculations of a convergence point $\boldsymbol{x}$ as the direction of the moving vector seems unreliable. Now, we became able to divide moving vectors in a positive group into two sub-groups. Let's call them as the positive1 sub-group and the positive 2 sub-group.

Step 4: As the moving vectors in the negative class can be divide into two sub-groups that are further separated than the two positive sub-groups, we apply a conventional clustering algorithm. For example, let the farthest $\boldsymbol{c}^1$ and $\boldsymbol{c}^2$ be initial points, and k-means++ clustering method [1] is applied to divide the moving vectors in the negative class into two sub-groups. Let's refer to them as the negative1 sub-group and the negative 2 sub-group.

Step 5: We have four sub-groups after the Steps 3 and 4. We have known which moving vectors of positive1 and positive2 go towards to $\boldsymbol{c}^1$ and $\boldsymbol{c}^2$, respectively in the Step 3, and to which negative sub-group $\boldsymbol{c}^1$ or $\boldsymbol{c}^2$ belongs, respectively. From this information obtained in Steps 3 and 4, we can know which two sub-groups make a pair, i.e. pair sub-groups going towards to the same local minimum. For example, we can know that (positive1 and negative1), and (positive2 and negative2) are pairs in Step 5 of Table I.

Step 6: As the moving vectors of two sub-groups go towards to the same local minimum in Step 5, we can calculate the two convergence points, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, that the moving vectors of the two classes go toward.

IV. Improvements for the Proposed Clustering Method

We cannot always expect that the directions of moving vectors go exactly towards the local optimum. Convergence points calculated with moving vectors that do not directly go toward the local optimum may not be located near the true local optimum. Furthermore, clustering errors in the previous section increase, which reduces the estimation precision of the local optimum further.

We propose four improvements for increasing clustering correctness and the precision of estimating local minima.

1) Generating two individuals in a small area randomly and making a moving vector.
2) Clustering and calculation of the convergence point using only moving vectors where the terminal point, i.e. offspring individuals, have higher fitness values.
3) Correcting the directions of moving vectors using orthogonal vectors.
4) Correcting clustering errors by checking the directions of moving vectors from their convergence point.

Detailed explanations of these proposed improvements follow.

Improvement 1: Creating a moving vector in a tiny area.

It is common that the moving distance between a parent individual and its offspring becomes so long that the offspring may be located beyond the local minimum which is closest to the parent individual. Although the moving vector may aim towards a different local minimum by chance, it is not adequate to use it to calculate a convergence point because it does not have information regarding the local minimum near its parent individual.

A solution to create a moving vector going towards the local minimum near a parent individual is to create it in a tiny area that can be approximate with a hyper-plane. Concretely speaking, by generating $\boldsymbol{c}_i$ within $1/p$ of each searching range around $\boldsymbol{a}_i$ using a uniform random number.

$$c_i[j] = a_i[j] + rand(-1, 1) \times \frac{(\max\{v_i[j]\} - \min\{v_i[j]\})}{p} \tag{4}$$

where $a_i[j]$ and $c_i[j]$ are the j-th parameter value of $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$; $1 \leq j \leq d$, d is the dimension number.

Comparing two individuals, let the individual with a higher and lower fitness values be referred to as $\boldsymbol{c}_i$ and $\boldsymbol{a}_i$, respectively. Thus, we can determine a moving vector $\boldsymbol{b}_i$. In ordinary EC, there is no guarantee that the fitness of the offspring will be better than that of its parent individual, and about half of the vectors from a parent individual to its offspring cannot be used to calculate their convergence point. On the other hand, with this improvement that lets the better individual between two generated individuals in a tiny area be $\boldsymbol{c}_i$, we can always generate a moving vector $\boldsymbol{c}_i$ going towards a local minimum. As all moving vectors can be used to calculate their convergence point, the precision of the local minimum estimate increases.

Although creating offspring very near their parents results in a slower convergence from the EC search point of view, using a well estimated convergence point as an elite individual results in an accelerated convergence. Separating the EC search process and the process of creating moving vectors and estimating local minima is one method for approaching the problem. That is, offspring generated by EC process are not directly used to create a moving vector, but we let a randomly generated point in a tiny area be $\boldsymbol{c}_i$,

obtain a moving vector, and calculate a convergence point near the local minimum.

Improvement 2: Clustering and calculating a convergence point using only top moving vectors.

The directions of moving vectors in valleys between local minima are influenced by multiple hills and valleys in a fitness landscape and do not always go toward one local minimum; see the (a) in Table III. It is easy to imagine that estimation errors of local minima become bigger if these moving vectors are used for the estimation. Besides the calculation of a convergence point in Section II, poor directions of moving vectors influence a clustering result in Section III, too, so that the precision of estimating local minima becomes further poorer.

The second proposed improvement is to use only the moving vectors from top individuals for clustering in Section III and calculation of the convergence point. This proposal is based on the assumption that the fitness of individuals in these poorer areas is lower than those near local minima. We may assume that the directions of moving vectors of individuals locating closer to local minima go towards their nearest local minima more correctly. In our experimental evaluation in Section V, we use the average fitness as the criterion for selecting top individuals.

Improvement 3: Correcting directions of moving vectors using orthogonal vectors.

The direction of a moving vector $\boldsymbol{b}_i$ can be corrected to go towards a local minimum by using orthogonal vectors. The fitness of the terminal point of $\boldsymbol{b}_i$ obtained in the *Improvement 1* must be better than that of its initial point, but there is no guarantee that the direction of the vector is towards a local minimum. Correcting the directions of moving vectors going toward a local minimum increases the precision of the estimated local optima.

Firstly, we generate $(d-1)$ random points near $\boldsymbol{a}_i$ besides the $\boldsymbol{c}_i$ obtained in the *Improvement 1*. Let the linear independent directional vectors from $\boldsymbol{a}_i$ to these points be $\boldsymbol{b}_i^2, \boldsymbol{b}_i^3, ..., \boldsymbol{b}_i^d$, and let the $\boldsymbol{b}_i$ obtained in the *Improvement 1* be $\boldsymbol{b}_i^1$. We can obtain orthonormalized vectors, $\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_d$, by applying the Gram-Schmidt orthonormalization to $\boldsymbol{b}_i^j$ $(j = 1, 2, ..., d)$. As the lengths of these unit orthonormalized vectors is 1, sometimes they are too long to calculate the fitness in a tiny space. Thus, we adjust these lengths by multiplying by the length of $\boldsymbol{b}_i$ obtained in *Improvement 1*. Let these orthonormalized vectors with normalized lengths be $\boldsymbol{e}_1^{'}, \boldsymbol{e}_2^{'}, ..., \boldsymbol{e}_d^{'}$; $\boldsymbol{e}_1^{'} = \boldsymbol{b}_i$.

Next, we calculate a synthesized vector by weighting them by increased fitness value and let the synthesized vector ($\boldsymbol{s}$ in Figure 3) be a new $\boldsymbol{b}_i$. Increased fitness value is expressed as $\Delta_j = f(\boldsymbol{e}_j^{'}) - f(\boldsymbol{a}_i)$, where $f(\,)$ is a fitness function. The mirror point for $\boldsymbol{a}_i$ becomes $\boldsymbol{e}_j^{'}$ $(= \boldsymbol{a}_i + (\boldsymbol{a}_i - \boldsymbol{e}_j^{'}))$ in the case of $f(\boldsymbol{e}_j^{'}) < f(\boldsymbol{a}_i)$; a vector $\boldsymbol{e}$ and the vector of its opposite direction in Figure 3 show this comparison. As shown in Eq. (5), the synthesized vector becomes a new $\boldsymbol{b}_i$.

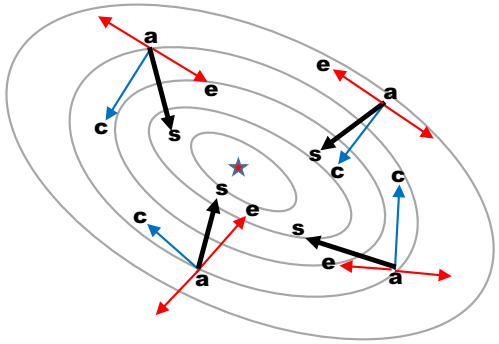


Figure 3. Visual explanation of *Improvement 3*. $\boldsymbol{a}$ and $\boldsymbol{c}$ are an initial point and an terminal point of a moving vector; $\boldsymbol{d}$ is an orthogonal vector of $(\boldsymbol{c} - \boldsymbol{a})$; $\boldsymbol{s}$ is a synthesized vector by weighting vectors $(\boldsymbol{c} - \boldsymbol{a})$ and $(\boldsymbol{d} - \boldsymbol{a})$ with increased fitness of each vector.

$$\boldsymbol{b}_i = \frac{\sum_{j=1}^{d} \Delta_j (\boldsymbol{e}_j^{'} - \boldsymbol{a}_i)}{\sum_{j=1}^{d} \Delta_j} \tag{5}$$

Improvement 4: Corrections of mis-clustering.

We apply the clustering method described in Section III to the n moving vectors obtained through *Improvements 1, 2,* and *3* and calculate the convergence point of each class. However, we cannot deny that the clustering result includes mis-clustering. *Improvement 4* is method for correcting this mis-clustering.

When a moving vector is correctly classified, the distance from the calculated convergence point to the terminal point, $\boldsymbol{c}_i$, of the moving vector must be shorter than that to its initial point $\boldsymbol{a}_i$ (in the case of $\boldsymbol{b}_1$ in Figure 4). If $||\boldsymbol{c}_i - \boldsymbol{x}_k||$ is bigger than $||\boldsymbol{a}_i - \boldsymbol{x}_k||$ such as $\boldsymbol{b}_2$ in Figure 4, we judge that the moving vector was wrongly classified, calculate the distance from the other convergence point $\boldsymbol{x}_k^{'}$ to the $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$, check if $||\boldsymbol{c}_i - \boldsymbol{x}_k^{'}|| < ||\boldsymbol{a}_i - \boldsymbol{x}_k^{'}||$, and reclassify it. After these corrections, we recalculate the convergence point for each class.

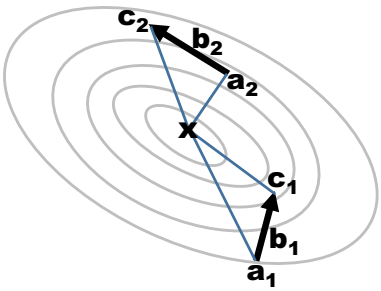


Figure 4. Detection of mis-clustering by comparing $||\boldsymbol{c}_i - \boldsymbol{x}_k||$ and $||\boldsymbol{a}_i - \boldsymbol{x}_k||$. When $||\boldsymbol{c}_i - \boldsymbol{x}_k|| > ||\boldsymbol{a}_i - \boldsymbol{x}_k||$, we doubt the clustering result.

By combing the basic clustering algorithm in Section III

Table II
PARAMETERS OF EQ. (6).

dimensions	2-D	6-D	10-D
population size	100	150	200
search ranges	[-4,4] of all 2 variables	[-4,4] of all 6 variables	[-4,4] all 10 variables
a_1 and a_2	(2.1, 3.4)	(2.1, 3.4)	(2.1, 3.4)
$\boldsymbol{\sigma}_1$ and $\boldsymbol{\sigma}_2$	{(1.5,1.5), (2.0,2.0)}	{(1.5,1.5,1.5,1.5,1.5,1.5), (2.0,2.0,2.0,2.0,2.0,2.0)}	{(1.5,1.5,1.5,1.5,1.5,1.5,1.5,1.5,1.5,1.5), (2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0)}
$\boldsymbol{\mu}_1$ and $\boldsymbol{\mu}_2$	{(-2.0,2.0), (2.0,-2.0)}	{(-2.0,2.0,-2.0,2.0,-2.0,2.0), (2.0,-2.0,2.0,-2.0,2.0,-2.0)}	{(-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0), (2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0)}

and its improvements in Section IV, we obtain the final proposed algorithm in the Algorithm 1.

Algorithm 1 Clustering algorithm of moving vectors for a bipolar task to estimate its two local minima.

1: Initialization.
2: Obtain two individuals using *Improvement 1*; one of them is a parent.
3: Correct the direction of the moving vectors using *Improvement 3*.
4: Select moving vectors for calculating a convergence point using *Improvement 2*.
5: Estimate the global convergence point x using the method described in Section II.
6: **if** More than a certain ratio of the conditions for unimodal tasks is satisfied **then**
7: The task is unimodal task.
8: goto 21
9: **end if**
10: Divide individuals into two sub-groups using the clustering method described in Section III.
11: Calculate the convergence point of each class using the method described in Section II.
12: **if** *Improvement 4* detects mis-clustered moving vectors **then**
13: Correct them using *Improvement 4*.
14: Recalculate each of two convergence points.
15: **end if**
16: **if** The conditions of bipolar tasks is satisfied **then**
17: The task has two local minima.
18: **else**
19: The task has at least three local minima.
20: **end if**
21: Output corresponding convergence point(s) as estimated local minima.
22: end of program.

V. EXPERIMENTAL EVALUATIONS

We evaluate whether our proposed method and improvements can cluster moving vectors correctly and estimate local optima correctly using the bipolar function given by Eq. (6) that consists of two d dimensional (d-D) Gaussian functions (d =2, 6, and 10).

$$f^{(d)}(x) = -\sum_{j=1}^{2}\left\{\boldsymbol{a}_j^{(d)} \exp\left(-\sum_{i=1}^{d}\frac{(x_{ij}-\mu_{ij}^{(d)})^2}{2\sigma_{ij}^{(d)}}\right)\right\} \quad (6)$$

The experimental parameters used for Eq. (6) are shown in Table II. We set p in Eq. (4) with 50.

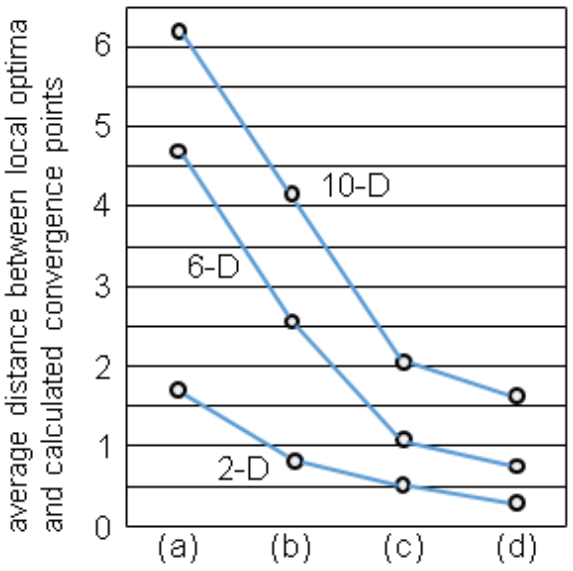
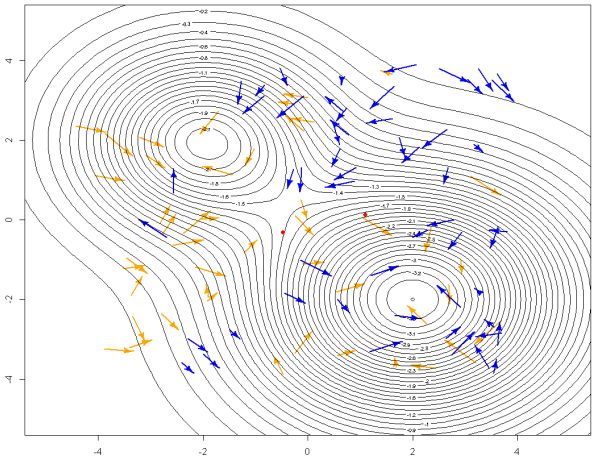


Figure 5. Distances between the calculated convergence point and its correct local minimum using 2-D, 6-D, and 10-D bipolar functions. (a) *Improvement 1*, (b) *Improvements 1+2*, (c) *Improvements 1+2+3*, (d) *Improvements 1+2+3+4*.

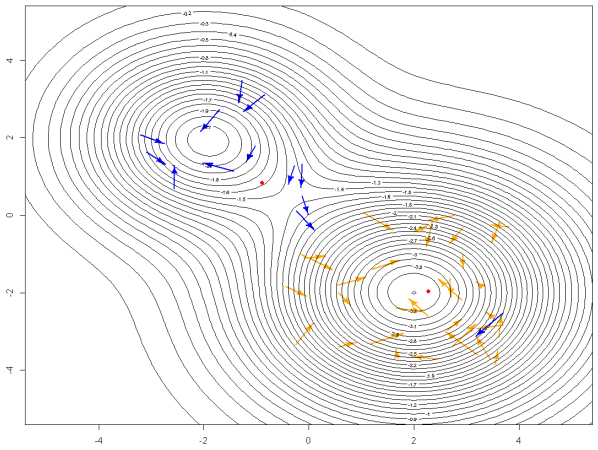
We evaluate our proposed four improvements with the Euclidian distance between calculated convergence points and their correct local minima, respectively, using 2-D, 6-D, and 10-D bipolar functions. Average distances over 10 trial runs are shown in Figure 5. Estimation of the local minima clearly became better according to an increase in the number of proposed improvements applied.

We also show the effect of the proposed improvements for clustering and estimation of local minima in Table III. Since moving vectors in tiny areas are too small for visualization, the moving vectors obtained with $p = 15$ in Eq. (4) are used in Table III(a).

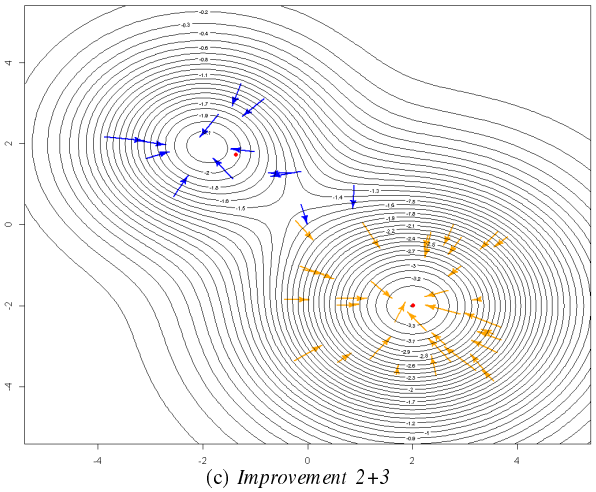
Table III
VISUALIZATION OF THE EFFECT OF THE PROPOSED IMPROVEMENTS FOR CORRECT CLUSTERING AND PRECISE ESTIMATION OF LOCAL MINIMA. THE TWO RED POINTS ARE THE CALCULATED CONVERGENCE POINTS.



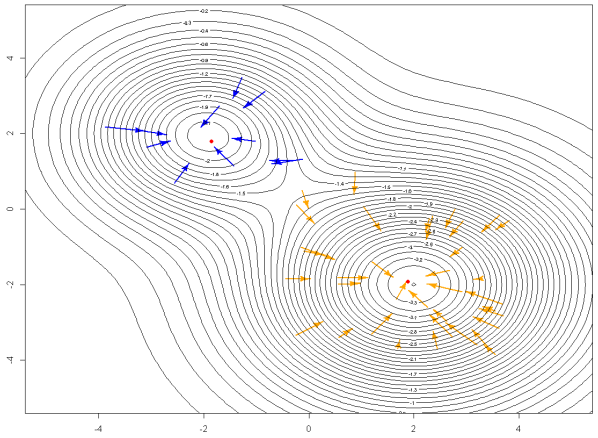
(a) Initial vectors. When the fitness of the offspring is poorer than that of its parent individual, a vector from the offspring to its parent is adopted as an initial vector.



(b) *Improvement 2*



(c) *Improvement 2+3*



(d) *Improvement 2+3+4*

VI. CONCLUSIONS

As the first step in the extension of the method for calculating the convergence point of evolutionary individuals to multimodal tasks, we clustered the directions of moving vectors for bipolar tasks consisting of two local minima. Firstly, we proposed a method for clustering moving vectors using their directions and their convergence point; secondly, we proposed four improvements to increase clustering precision; lastly, we confirmed that our proposal improved estimation performance.

We are going to extend the proposed clustering algorithm to multimodal tasks with more than two local minima and generalize our proposed method.

Acknowledgment

This work was supported in part by Grant-in-Aid for Scientific Research (15K00340 and 26540145).

REFERENCES

[1] Arthur, D. and Vassilvitskii, S., "k-means++: the advantages of careful seeding," 18th ACM-SIAM symposium on discrete algorithms (SODA2007), PA, USA, pp.1027–1035 (2007).

[2] Murata, N., Nishii, R., Takagi, H., and Y. Pei, "Estimation Methods of the Convergence Point of Moving Vectors Between Generations," Japanese Society for Evolutionary Computation Symposium 2014, Hatsukaichi, Japan, pp.210–215 (Dec., 2014). (in Japanese).

[3] Murata, N., Nishii, R., Takagi, H., and Y. Pei, " Analytical Estimation of the Convergence Point of Populations," 2015 IEEE Congress on Evolutionary Computation (CEC2015), Sendai, Japan, pp. 2619–2624 (May 25-28, 2015).