Accelerating Evolutionary Computation Using Estimated Convergence Points

Jun Yu*, Yan Pei†, and Hideyuki Takagi‡
*Graduate School of Design, Kyushu University, Fukuoka, Japan 815-8540
Email: yujun@kyudai.jp
†Computer Science Division, the Universally of Aizu, Aizu-wakamatsu, Japan 965-8580
Email: peiyan@u-aizu.ac.jp
‡Faculty of Design, Kyushu University, Fukuoka, Japan 815-8540
URL: http://www.design.kyushu-u.ac.jp/~takagi/

***Abstract*—We use the convergence points estimated by our proposed method as elite individuals for evolutionary computation and evaluate the acceleration effect and analyze the effect and computational cost. The worst individuals in population are replaced with the convergence points estimated from the moving vectors between parent individuals and their offspring; i.e. these convergence points are used as elite individuals. Differential evolution (DE) and 14 benchmark functions are used in our evaluation experiments. The experimental results show that use of the estimated convergence points as elite can accelerate DE search in spite of the calculation cost of the convergence points. We finally analyze the components of the proposed estimation method to improve cost-performance.**

***Keywords*—evolutionary computation, estimation of convergence points, acceleration, multi-modal optimization**

I. Introduction

Evolutionary Computation (EC) algorithms are a form of individual-based optimization. EC uses fitness information iteratively to find the optimal solution. One of the primary research directions in EC is to improve EC convergence and find the global optimum, or local optima in some cases, more quickly. There have been several attempts to achieve this objective, e.g. by developing new EC algorithms [2], [3], [6], improving EC operations [4], [5], approximating the fitness landscape for a rough but quick search [4], [5], and others. We also know that evolutionary paths can also offer useful information. In this paper, we call the vector which follows this evolutionary path from a parent to its offspring a moving vector.

It was shown mathematically that the convergence point of EC individuals can be estimated using multiple moving vectors [7], [8]. Since the estimated convergence point is expected to be located near the global optimum, the EC algorithm can accelerate its convergence speed by using this point as an elite individual. In our preliminary work, we showed that the rank of the elite was higher among population, which means that it has higher potential to accelerate EC convergence speed. We also extended the method for estimating the convergence point to multimodal tasks to increase the estimation performance [10].

The main objective of this paper is to evaluate, using benchmark functions, how well the combination of the convergence point estimation method with differential evolution (DE) accelerates EC convergence compared to pure DE. The second objective is to analyze the computational costs and convergence of components of the proposed method so that we can improve the cost-performance of our method for the next stage of our research in the near future.

We summarize our basic method for estimating the convergence point of moving vectors [7], [8] in Section II-A and an extension of the method for bipolar tasks [10] in Sections II-B and II-C. We then introduce our experimental evaluations in Section III and conduct the evaluations using DE with 14 benchmark functions in Section IV. Introducing the estimation method described in II is not the objective of this paper; rather we seek to explore how DE + the proposed extended estimation method converges faster than DE alone. Finally, we discuss the effect of the proposed method, analyze the effect of its components and conclude in the remained sections.

II. Convergence Point Estimation

A. Basic Estimation Algorithm for Unimodal Tasks

The convergence point for the moving vectors between parent individuals and their offspring in the next EC search generation can be calculated mathematically [7], [8]. This method is used in our proposed clustering method in the following section. Let us begin by defining symbols. $\boldsymbol{a}_i$ and $\boldsymbol{c}_i$ in the Figure 1 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 the 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, $\boldsymbol{a}_i + t_i\boldsymbol{b}_i$ ($t_i \in R$). By *nearest*, we mean that the total distance from $\boldsymbol{x}$ to the 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 thus remove t_i.

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

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

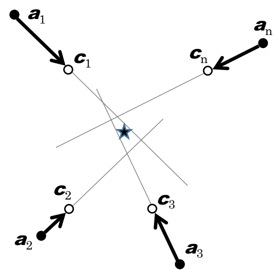


Fig. 1: Moving vector $\boldsymbol{b}_i$ $(= \boldsymbol{c}_i - \boldsymbol{a}_i)$ is calculated from a parent individual $\boldsymbol{a}_i$ and its offspring $\boldsymbol{c}_i$ in the d-dimensional searching space. The $\star$ mark is the convergence point for these moving vectors.

The $\hat{\boldsymbol{x}}$ that minimizes the total distance in 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 Eq. (3), where $\boldsymbol{I}_d$ is a unit matrix.

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

B. Clustering for Bipolar Task

The method for calculating a convergence point discussed in the previous section is clearly effective for unimodal tasks as show in [7], [8], but it is not always valid for multi-modal tasks because moving vectors go towards different local optima. To make this method applicable to general optimization tasks, we must extend the basic estimation method to multi-modal tasks.

As the first step of this challenge, we start with bipolar task and develop a clustering method of moving vectors to estimate two local minima [10]. Here, we explain our proposed clustering method in six steps.

Step 1: Estimate the convergence point $\boldsymbol{x}$ using the method described in the previous section.

Step 2: According to the signs of the inner product of a moving vector and a vector from the $\boldsymbol{x}$ to the terminal point of the moving vector, we divide moving vectors into a group of moving vectors coming towards $\boldsymbol{x}$, called the negative group, and the group of those going in the opposite directions of $\boldsymbol{x}$, called the positive group.

Step 3: Let the farthest two terminal points among all moving vectors be $\boldsymbol{c}^1$ and $\boldsymbol{c}^2$. The positive group can be further divided into two positive sub-groups by checking which direction the moving vectors go towards, $\boldsymbol{c}^1$ or $\boldsymbol{c}^2$.

Step 4: For the moving vectors in the negative groups, we apply a conventional clustering algorithm. For example, let the farthest $\boldsymbol{c}^1$ and $\boldsymbol{c}^2$ be initial points. Then we apply the k-means++ clustering method [1] to divide the moving vectors in the negative group into two sub-groups.

Step 5: We now know which moving vectors in the positive and negative groups go towards $\boldsymbol{c}^1$ and $\boldsymbol{c}^2$, respectively. From the information obtained in Steps 3 and 4, we can know which two sub-groups make a pair.

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

Note that this extended method includes the basic estimation method of Section II-A. When all the signs in the Step 2 are the same, this method estimates only one convergence point $\boldsymbol{c}$ and stops.

C. Four Improvements for the Clustering Method

We have proposed four improvements for increasing clustering correctness and the precision of the estimated local minima. A convergence point which is calculated from moving vectors that do not directly go toward the local optimum may not be locate near the true local optimum. Additionally, clustering errors can further reduce the estimation precision of the local optimum.

Improvement 1: Creating a moving vector in a tiny area

In ordinary EC, there is no guarantee that the fitness of the offspring will be better than that of its parent individual, and the moving distance between a parent individual and its offspring sometimes becomes so far that the offspring locates beyond the local minimum near the parent individual. For these situations, the solution such that the moving vector goes towards the local minimum is to create a moving vector in a tiny area that can be approximated with a hyper-plane. Alternatively, let an individual with higher fitness value and another be offspring and parent, respectively.

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

The second proposed improvement is to use only the moving vectors of the top individuals for clustering and the calculation of the convergence point. This strategy is based on the fact that the fitness of individuals in poorer areas is lower than those near local minima; we may assume that the moving vector directions of individuals located closer to local minima go towards their nearest local minima more correctly. In our experimental evaluation, 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. First we generate $(d-1)$ random points near $\boldsymbol{a}_i$ besides the $\boldsymbol{c}_i$ obtained in the *Improvement 1*. Let 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 the 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 it is too long to calculate fitness in a tiny space. Thus, we adjust these lengths by multiplying the length by $\boldsymbol{b}_i$

obtained in the *Improvement 1*. Let these orthonormalized vectors which lengths are normalized 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 with increased fitness values and let the synthesized vector be a new $\boldsymbol{b}_i$.

Improvement 4: Corrections of Mis-clustering.

We apply a clustering method to n moving vectors obtained through *Improvements 1, 2,* and *3* and calculate a convergence point for each class. However, we cannot deny that the clustering result includes mis-clustering. *Improvement 4* is a method for correcting for this mis-clustring. 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 $\boldsymbol{a}_i$. After these corrections, we calculate the convergence point for each class.

III. Effect of Estimated Convergence Point for Accelerating Evolutionary Computation

The estimated convergence point can be an excellent elite individual. Since the peak of an approximated fitness landscape is expected to be located near the global optimum, it was used as an elite individual and its acceleration effect for EC search was demonstrated in [9]. The estimated convergence point is expected to also demonstrate the same effect.

The objective of this paper is to use estimated convergence points as elite individuals and evaluate the acceleration effect for EC algorithms. We obtain the estimated point(s) using our estimation method outlined in the Sections II-B and II-C and replace the worst individual(s) with the estimated point(s). The estimation method obtains two convergence points, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ in the Step 6 in the Section II-B, when it is applied to multi-modal tasks. Whether both $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ are used as the elite or only the better one is used depends on the chosen elitism strategy. Algorithm 1 shows the flowchart of the EC algorithm combined with this elitism strategy.

For multimodal tasks, using two convergence points, $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, should have higher potential to accelerate the EC search than using only one convergence point, $\boldsymbol{x}$ from the Step 1. However, we do not know which is better; whether it is better to (a) replace the two worst individuals with these $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$ or (b) replace only the worst individual with the better of $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$. To answer this question is the secondary objective of this paper.

A drawback of the estimation method outlined in Section II-B and II-C is computational cost, especially the improvements in Section II-C [10]. Achieving a balance between the acceleration effect on EC convergence and the computational cost is important when considering practical use. There are four improvements, with especially *Improvements 1* and *3* involving extra computational cost. Assume that the EC searches for the global optimum with p individuals in a d-dimensional space. Since *Improvement 1* generates p offspring individuals around the parent individuals in tiny areas, it requires the fitness calculation be performed p extra times. *Improvement 3* needs to to perform the fitness calculations d times to obtain one d-dimensional orthogonal vector. As a consequence, the

Algorithm 1 Estimated convergence point to accelerate EC. G: generation.

```
1: Generate an initial population.
2: Evaluate the fitness for each individual.
3: for $G = 1$ to $MaxGeneration$ do
4:     Obtain the next generation using EC algorithm.
5:     Obtain the estimated points and evaluate their fitness.
6:     if their fitness values are better than those of the worst
       individuals then
7:         use the estimated point(s) as elite individual(s), re-
           place the worst individuals with the estimated points
8:     end if
9: end for
10: return the optimum
```

number of fitness calculations for *Improvement 3* becomes $p \times d$ in total.

IV. Experimental Evaluations

We use 14 benchmark functions from the CEC2005 benchmark test suite [11] in our evaluations. Table I shows their types, characteristics, variable ranges, and optimum fitness values. These landscape characteristics include shifted, rotated, global on bounds, unimodal and multi-modal. We test them with three dimensional settings, $D = 2$ (2-D), 10 (10-D), and 30 (30-D).

TABLE I: Benchmark functions (Uni=Uni-modal, Multi=Multi-modal, Sh=Shifted, Rt=Rotated, GB=Global on Bounds, NM=Number Matrix)

No.	Types	Characteristics	Ranges	Optimum fitness
f_1		Sh Sphere		-450
f_2		Sh Schwefel 1.2		-450
f_3	Uni	Sh Rt Elliptic	$[-100, 100]$	-450
f_4		f_2 with Noise		-450
f_5		Schwefel 2.6 GB		-310
f_6		Sh Rosenbrock	$[-100, 100]$	390
f_7		Sh Rt Griewank	$[0, 600]$	-180
f_8		Sh Rt Ackley GB	$[-32, 32]$	-140
f_9		Sh Rastrigin	$[-5, 5]$	-330
f_{10}	Multi	Sh Rt Rastrigin	$[-5, 5]$	-330
f_{11}		Sh Rt Weierstrass	$[-0.5, 0.5]$	90
f_{12}		Schwefel 2.13	$[-\pi, \pi]$	-460
f_{13}		Sh Expanded F8F2	$[-5, 5]$	-130
f_{14}		Sh Rt Scaffer F6	$[-100, 100]$	-300

The DE experimental parameters are set as in Table II. Evaluations are conducted under a hard search condition: we search with only 20, 50, and 100 individuals for 2-D, 10-D, and 30-D functions, respectively. The stop conditions are MAX_{NFC} in Table II.

Our proposed method, i.e. DE + the estimated convergence point + the improvements in Section II, needs more fitness calculations than simple DE, especially for *Improvements 1* and *3*. For fair evaluations, we evaluate DE convergence along the number of fitness calls instead of generations. We test each benchmark function with 30 trial runs in 3 different dimensional spaces and apply the Wilcoxon signed-rank test on the fitness values at the maximal number of fitness calculations,

TABLE III: Mean values of 2-D, 10-D, and 30-D benchmark functions at the stop conditions in Table II. "Proposal with 4 improvements" means DE + estimated convergence point + four improvements in the Section II. The symbols, $+$, $-$, and $=$, mean that the proposed method is better, worse, and equal than/to the canonical DE significantly ($p < 0.05$) according to the Wilcoxon signed rank tests.

Func.	2-D			10-D			30-D		
	DE	proposal with 4 improvements	p-value mark	DE	proposal with 4 improvements	p-value mark	DE	proposal with 4 improvements	p-value mark
f_1	-450.000	-450.000	=	-449.998	-450.000	+	-398.286	-449.359	+
f_2	-449.999	-450.000	+	-449.917	-449.837	-	8158.109	7355.029	=
f_3	-434.151	-419.431	-	339303.984	42683.940	+	31108698.580	14062029.680	+
f_4	-449.999	-450.000	+	449.420	-449.617	=	15696.170	12782.250	+
f_5	-310.000	-309.937	-	-309.995	-302.167	-	3545.480	4162.050	-
f_6	414.755	401.1079	=	409.077	398.927	=	209329.800	3427.617	+
f_7	-140.847	-140.847	-	1087.046	1090.753	-	4533.662	5401.898	-
f_8	-122.920	-125.036	=	-119.443	-119.482	=	-119.037	-118.926	-
f_9	-329.996	-329.900	=	-328.147	-315.874	-	-309.509	-150.881	-
f_{10}	-329.858	-329.894	=	-297.886	-311.386	+	-135.596	-136.837	=
f_{11}	90.184	90.043	+	96.231	96.893	=	120.933	132.486	-
f_{12}	-459.195	-459.995	+	-252.113	-432.709	+	49584.309	61050.128	-
f_{13}	-129.955	-129.986	+	-128.811	-128.036	-	-104.396	-113.195	=
f_{14}	-299.935	-299.966	+	-296.119	-296.320	+	-286.574	-286.124	-

TABLE II: DE algorithm parameter setting.

population size for 2-D, 10-D, and 30-D search	20, 50, and 100
scale factor F	1
crossover rate	0.9
DE operations	DE/rand/1/bin
stop condition; max. # of fitness evaluations, MAX_{NFC}, for for 2-D, 10-D, and 30-D search	2,000, 50,000, 200,000
dimensions of benchmark functions, D	2, 10, and 30
# of trial runs	30

MAX_{NFC}, to check the significance of the proposal and make an algorithm ranking.

Tables III shows mean fitness values at the stop conditions of two methods, canonical DE and the proposed method of full improvements, i.e. "DE + the estimated convergence point + four improvements," and Wilcoxon signed-rank test results between two methods. The convergence curves of the 30-D functions are shown in Fig. 2. These figures include two other additional convergence curves for the proposed method with partial improvements, and we use them to analyze our proposed method in the next section.

V. Discussions

The first discussion is the superiority of our proposed method, especially full version of four *Improvements*, despite its increased computational cost. Comparative results between canonical DE and the full version of our proposed method, are shown in Table III. From the numbers of $+$, $-$, and $=$ in the p-value marks rows, the cost-performance of the full version of our proposed method does not seem significantly improved. It is true that correcting directions of moving vectors toward to their local optimum is quite effective to estimate their convergence point [10]. However, the *Improvement 3* requests d times of calculating fitness values for d orthogonal vectors in a d dimensional space, which is a reason why the cost-performance of the proposed full version is not effective.

The second discussion point is how to improve the cost-performance. Here, we may be able to improve the cost-performance of our proposed method. The search performance of the proposed method must decrease by eliminating *Improvement 3*, but its cost-performance may increase because of its high computational cost. We should find a balance between the computational cost and the convergence performance, taking into account the complexity of the tasks and the computational cost of the fitness calculations. Some methods for controlling the balance include: using only some of the four *Improvements*, applying our proposed method at every k-th generation, applying it after screening better individuals by run only *Improvement 2*, and others.

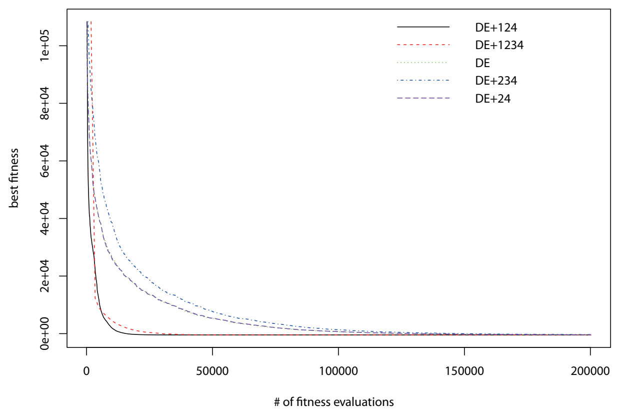
To confirm the effect of "using only some of the four *Improvements*" mentioned previously, we perform experimental evaluations and analyze the results using four variations on our proposed method. Let's denote full version of our proposed method that uses the four *Improvements* as DE+1234, and denote other partial use of the *Improvements* with used *Improvements* such as DE+124, DE+234, and DE+24. These three additional experiments as well as DE+1234 and canonical DE are shown in the Fig. 2.

We apply the Friedman test and the Holm's multiple comparison test among canonical DE and variations of these proposed methods and show their results in the Table IV. A simplified version of our proposed method without *Improvement 3*, i.e. DE+124 in the table, outperformed DE convergence significantly for 10 benchmark functions ($p < 0.01$), is almost same with DE for f_8, f_9 and F_{14}, and is slower than DE for f_{11}. From these results, we can say that a partial combination of our four proposed *Improvements*, the DE+124, has the best acceleration performance.

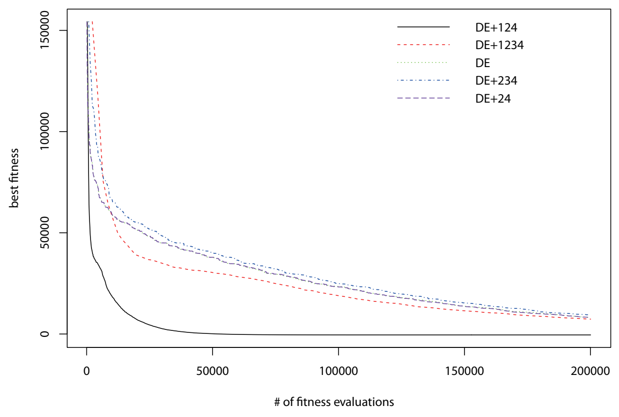
VI. Conclusion

In 2015, we proposed a method for estimating the convergence points in a population-based search [10] that was an extension version of the basic algorithm [7], [8]. In this paper, we tried to use the estimated convergence points as elite individuals to accelerate the EC algorithm. We evaluated its acceleration effect by combining it with DE and comparing with canonical DE.

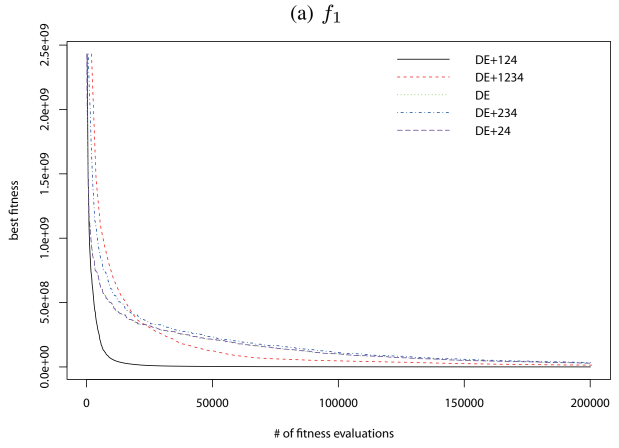
The proposed method, especially the method using *Improvements 1, 2, and 4*, accelerated DE convergence for all



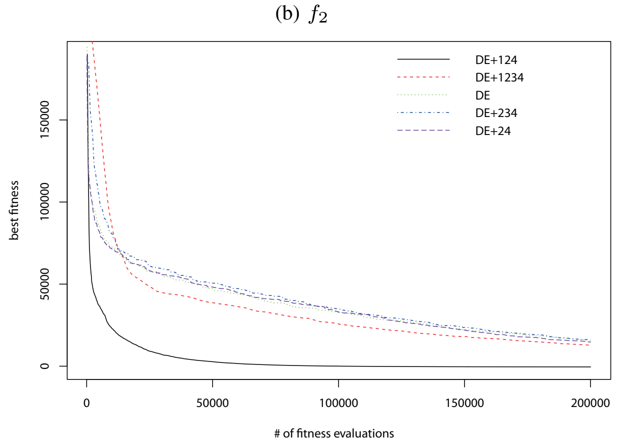
(a) f_1



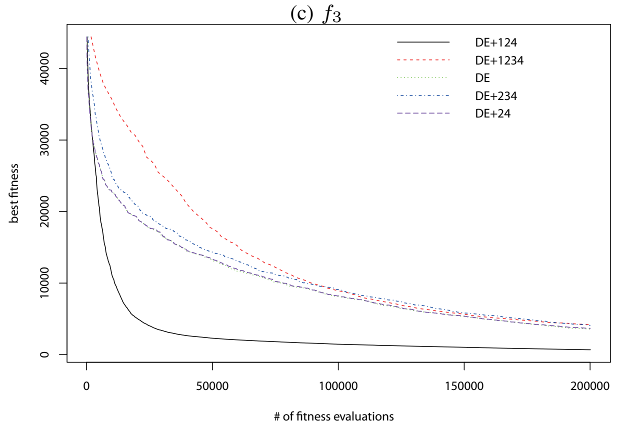
(b) f_2



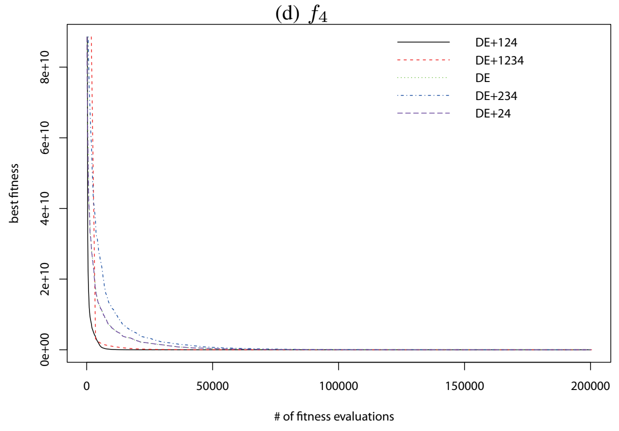
(c) f_3



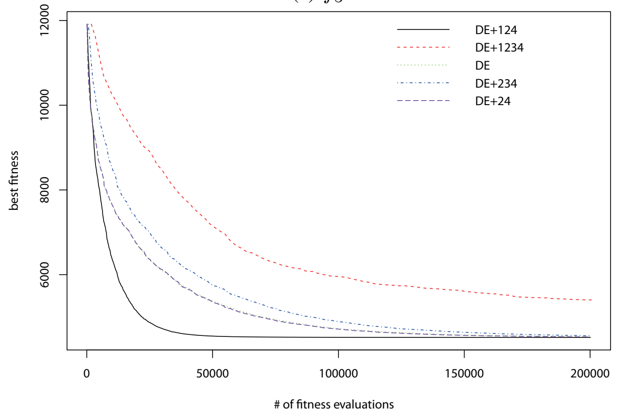
(d) f_4



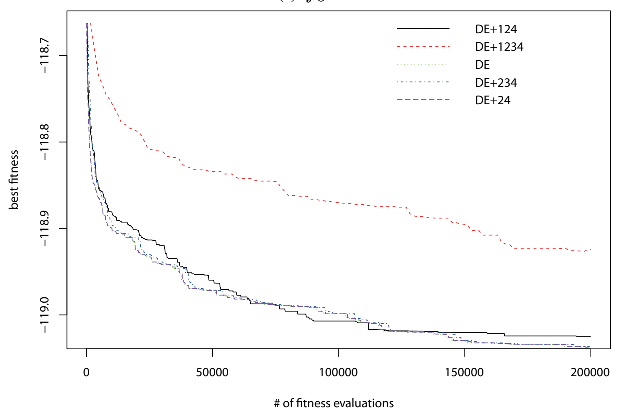
(e) f_5



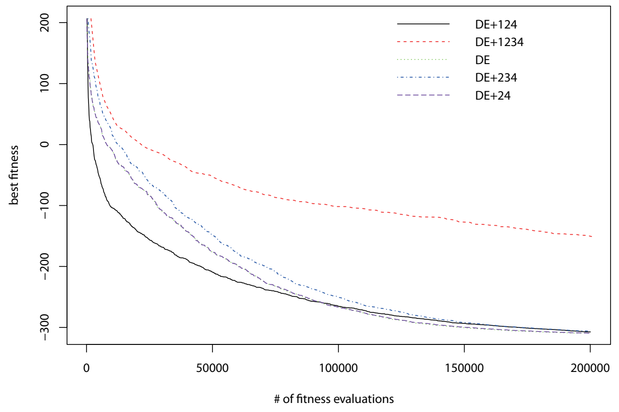
(f) f_6



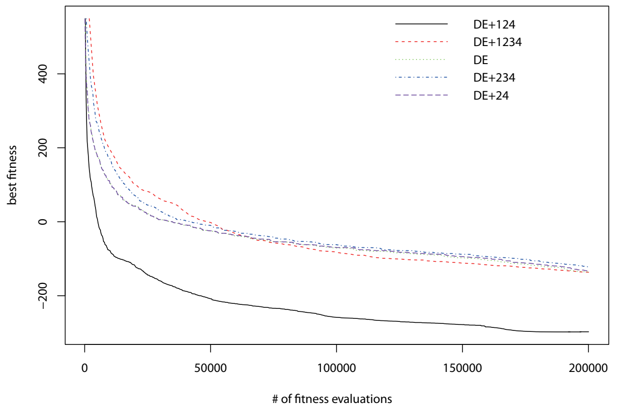
(g) f_7



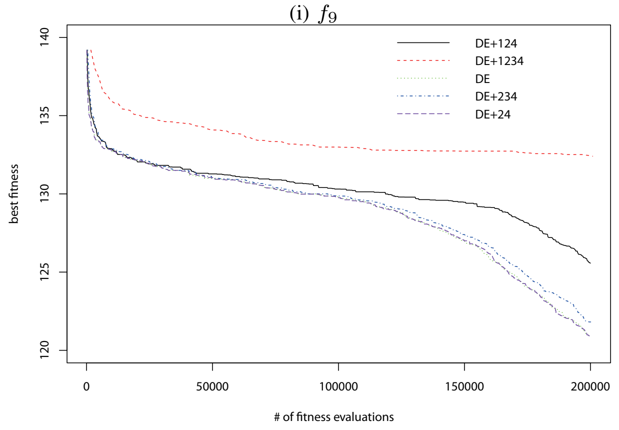
(h) f_8



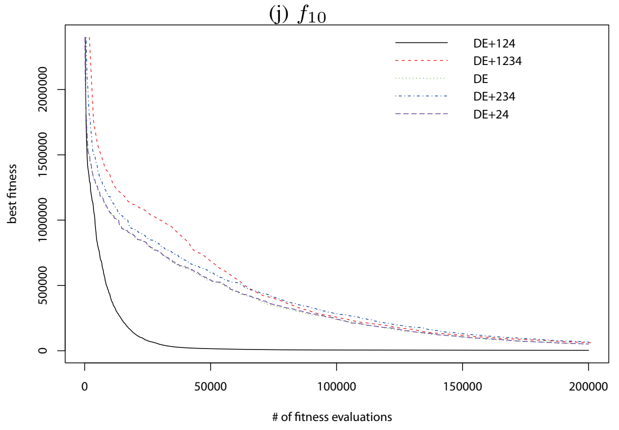
(i) f_9



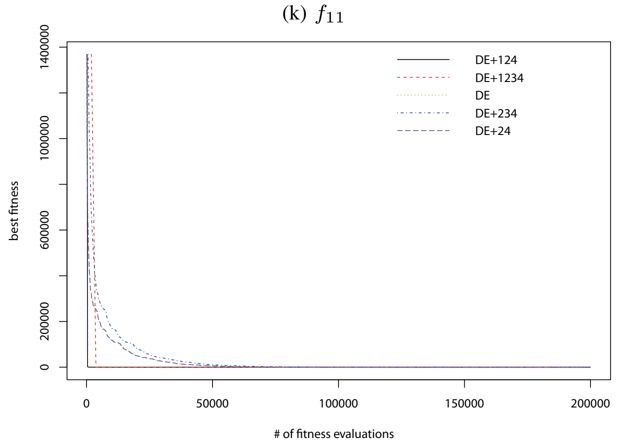
(j) f_{10}



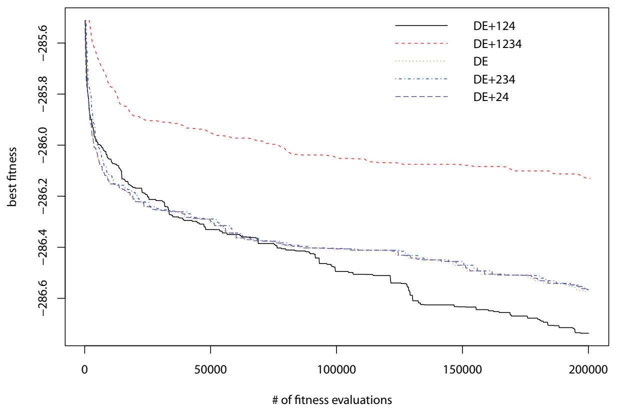
(k) f_{11}



(l) f_{12}



(m) f_{13}



(n) f_{14}

Fig. 2: Convergence curves of 30-D f_1–f_{14}. DE+1234, DE+124, DE+234, and DE+24 refer to DE + the convergence point(s) + the full or partial *Improvements* outlined in Section II-B, where the numbers correspond to which*Improvements* are used.

benchmark functions and showed the effectiveness of the use of an estimated EC convergence point(s).

Theoretically, when the convergence point is estimated well, the point must be close to the global optimum and therefore the point becomes a strong searching point. We were concerned about the additional computational cost required. However, experimental evaluations along with the number of fitness calls showed that using the estimated convergence point is still beneficial for accelerating EC convergence from the cost-performance point of view.

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] Back, T., Hammel, U., and Schwefel, H.-P., “Evolutionary computation: Comments on the history and current state,” IEEE Trans. on Evolutionary Computation, vol.1, no.1, pp.3–17 (1997).

TABLE IV: Friedman test and Holm's multiple comparison test results for average fitness values of 5 methods for 30 trial runs for 14 30-D functions. $\ll$, $<$, and $\approx$ mean that there are significant difference with significant levels 1%, 5%, and no significance, respectively. DE+1234 means "DE + estimated convergence point + four *Improvements* (1+2+3+4)". Other numbers mean the combination of *Improvement* No. in the Section II, too.

f_1	DE+124 $\ll$ DE+1234 $\ll$ DE $\approx$ DE+24 $\ll$ DE+234
f_2	DE+124 $\ll$ DE+1234 $\approx$ DE $\approx$ DE+24 $\approx$ DE+234
f_3	DE+124 $\ll$ DE+1234 $\ll$ DE+24 $\approx$ DE $\approx$ DE+234
f_4	DE+124 $\ll$ DE+1234 $\approx$ DE+24 $\approx$ DE $\approx$ DE+234
f_5	DE+124 $\ll$ DE $\approx$ DE+24 $\approx$ DE+234 $\approx$ DE+1234
f_6	DE+124 $<$ DE+1234 $\ll$ DE $\approx$ DE+24 $\ll$ DE+234
f_7	DE+124 $\ll$ DE+24 $\approx$ DE $\ll$ DE+234 $\ll$ DE+1234
f_8	DE $\approx$ DE+234 $\approx$ DE+24 $\approx$ DE+124 $\ll$ DE+1234
f_9	DE $\approx$ DE+24 $\approx$ DE+124 $\approx$ DE+234 $\ll$ DE+1234
f_{10}	DE+124 $\ll$ DE+1234 $\approx$ DE $\approx$ DE+24 $\approx$ DE+234
f_{11}	DE+24 $\approx$ DE $\approx$ DE+234 $\ll$ DE+124 $\ll$ DE+1234
f_{12}	DE+124 $\ll$ DE $\approx$ DE+24 $\approx$ DE+1234 $\approx$ DE+234
f_{13}	DE+124 $\ll$ DE+1234 $\approx$ DE $\approx$ DE+24 $\ll$ DE+234
f_{14}	DE+124 $\approx$ DE $\approx$ DE+234 $\approx$ DE+24 $\ll$ DE+1234

[3] Coello Coello, C.A. "Evolutionary multi-objective optimization: A historical view of the field," IEEE Computational Intelligence Magazine (2006).

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

[5] Eiben, Á.E., Hinterding, R., and Michalewicz, Z., "Parameter control in evolutionary algorithms," IEEE Trans. on Evolutionary Computation, vol.3, no.2, pp.124–141 (1999).

[6] Mullen, R.J., Monekosso, D., Barman, S., and Remagnino, P., "A review of ant algorithms," Expert Systems with Applications, vol.36, no.6, pp.9608–9617 (2009).

[7] 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).

[8] 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).

[9] Takagi, H. Ingu, T. and Ohnishi, K., "Accelerating a GA Convergence by Fitting a Single-Peak Function," J. of Japan Society for Fuzzy Theory and Intelligent Informatics, vol.15, no.2, pp.219-229 (2003). (in Japanese).

[10] Yu J. and Takagi H., "Clustering of Moving Vectors for Evolutionary Computation," 7th Int. Conf. on Soft Computing and Pattern Recognition (SoCPaR2015), pp.169-174, Fukuoka, Japan (13-15, Oct. 2015).

[11] P. N. Suganthan, N. Hansen, J. J. Liang, K. Deb, Y.-P. Chen, A. Auger, and S. Tiwari. "Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization," https://www.lri.fr/ hansen/Tech-Report-May-30-05.pdf.

[12] J. H. Zar. *Biostatistical analysis*. Pearson Education India, 1999.