

# Accelerating Evolutionary Computation Using a Convergence Point Estimated by Weighted Moving Vectors

YU, Jun

Graduate School of Design, Kyushu University

LI, Yuhao

Graduate School of Design, Kyushu University

PEI, Yan

Computer Science Division, The University of Aizu

TAKAGI, Hideyuki

Faculty of Design, Kyushu University

<https://hdl.handle.net/2324/2235981>

---

出版情報 : Complex and Intelligent Systems, pp.1-11, 2019-05-28. Springer

バージョン :

権利関係 : This article is distributed under the terms of the Creative Commons Attribution 4.0 International License





# Accelerating evolutionary computation using a convergence point estimated by weighted moving vectors

Jun Yu<sup>1,2</sup> · Yuhao Li<sup>2</sup> · Yan Pei<sup>3</sup> · Hideyuki Takagi<sup>4</sup>

Received: 5 December 2018 / Accepted: 18 May 2019  
© The Author(s) 2019

## Abstract

We introduce weighted moving vectors to increase the accuracy of estimating a convergence point of population and evaluate its efficiency. Key point is to weight moving vectors according to their reliability when a convergence point is calculated instead of equal weighting of the original method. We propose two different methods to evaluate the reliability of moving vectors. The first approach uses the fitness gradient information between starting points and terminal points of moving vectors for their weights. When a fitness gradient is bigger, the direction of a moving vector may have more potential, and a higher weight is given to it. The second one uses the fitness of parents, i.e., starting points of moving vectors, to give weights for moving vectors. Because an individual with higher fitness may have a high probability of being close to the optimal area, it should be given a higher weight, vice versa. If the estimated point is better than the worst individual in current population, it is used as an elite individual and replace the worst one to accelerate the convergence of evolutionary algorithms. To evaluate the performance of our proposal, we employ differential evolution and particle swarm optimization as baseline algorithms in our evaluation experiments and run them on 28 benchmark functions from CEC 2013. The experimental results confirmed that introducing weights can further improve the accuracy of an estimated convergence point, which helps to make EC search faster. Finally, some open topics are given to discuss.

**Keywords** Evolutionary computation · Estimation of convergence point · Acceleration · Weight-based estimation

## Introduction

Evolutionary computation (EC) algorithms are a form of population-based optimization techniques and repeatedly simulate survival of the fittest and natural selection to find the global optima. They have been widely researched and successfully applied both in academia and industry thanks to their many outstanding features, such as robustness, intelligence, usability, parallelism, and others. As the complexity, including nonlinearity, non-convexity, and non-differentiability, of real-world problems has increased, the demand for high-performance EC algorithms is also growing rapidly.

Many metaphor-based EC algorithms but with similar optimization frameworks have been proposed, such as differential evolution (DE) [1], particle swarm optimization (PSO) [2], and bat algorithm [3]. Besides, some practitioners try to develop new efficient mechanisms to enhance the performance of existing EC algorithms. In the case of DE, Yu et al. introduced two competitive strategies to generate more potential offspring individuals and accelerate the elimina-

This work was supported in part by Grant-in-Aid for Scientific Research (17H06197, 18K11470, 19J11792).

✉ Jun Yu  
yujun@kyudai.jp  
Yuhao Li  
2DS18098N@s.kyushu-u.ac.jp  
Yan Pei  
peiyan@u-aizu.ac.jp  
Hideyuki Takagi  
takagi@design.kyushu-u.ac.jp

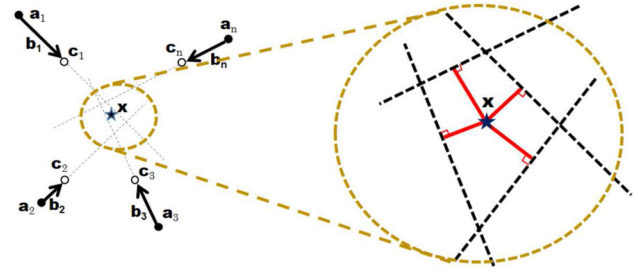
- <sup>1</sup> Japan Society for the Promotion of Science, Tokyo, Japan
- <sup>2</sup> Graduate School of Design, Kyushu University, 4-9-1, Shio-baru, Minami-ku, Fukuoka, Japan
- <sup>3</sup> Computer Science Division, The University of Aizu, Tsuruga, Ikki-machi, Aizu-Wakamatsu, Fukushima, Japan
- <sup>4</sup> Faculty of Design, Kyushu University, 4-9-1, Shiobaru, Minami-ku, Fukuoka, Japan

tion of poor individuals to accelerate its convergence [4]. As the literature [5] criticizes, they lack essential innovation and even misguide the development of EC community. We introduce a deterministic weight-based estimation method to accelerate meta-heuristic-based EC algorithms from a new perspective.

Evolutionary path contains lots of information, e.g., an evolutionary direction and local fitness landscape information, which can be fully used to guide evolution well. Murata et al. first proposed that a mathematical method could be used to calculate the global optimum using the information of two subsequent generations [6]. In this paper, we call vectors from parent individuals to their offspring in the next EC search generation (or the vector pointing from worse fitness individuals to higher fitness individuals) as *moving vectors*. We extended this basic method for estimating a convergence point to bipolar optimization tasks by applying a developed separation method for moving vectors [7]. We combined our proposed estimation method to bipolar tasks with DE, and the experimental results confirmed that it could accelerate convergence [8]. Besides, we introduced an individual pool to increase the precision of an estimated convergence point by using individual information from past generations [9]. We also applied the basic estimation method to multi-objective tasks to speed up the construction of Pareto optimality [10] and investigated the feasibility of using an estimated convergence point to accelerate interactive EC to reduce user fatigue [11]. Our previous works showed that the estimated convergence point is effective and potential to accelerate EC search, but there is still a lot of room for further research on the estimation method.

The main objective of this paper is to introduce reliable weights to moving vectors and increase the accuracy of an estimated convergence point, while the original estimation method gives the same weights to moving vectors, i.e., no weighting. The secondary objective is to propose two methods of weighting moving vectors. We show the effect of weighting visually using a two-dimensional Gaussian function and its acceleration effect of DE with the proposed method using 28 benchmark functions from CEC 2013 test suite. Finally, we summarize our works and provide some open topics for future discussions.

The remaining paper is organized as follows: we roughly summarize the basic estimation method in Sect. 4. The weight-based estimation method and two different methods of weighting are presented in detail in Sects. 3.1 and 3.2, respectively. We experimentally evaluate the effect of the proposed methods using DE and PSO in Sect. 4. Finally, we analyze the results and discuss the effects of our proposal in Sect. 5 and conclude our works in Sect. 6.



**Fig. 1** Moving vector  $b_i (= c_i - a_i)$  is calculated from a parent individual  $a_i$  and its offspring  $c_i$  in the  $d$ -dimensional searching space. The  $\star$  mark is the convergence point for these moving vectors

## Mathematical estimation of a convergence point

Most EC algorithms converge to the global optimal area gradually by updating their population. The excellent point of the basic estimation method [6] is that it can calculate the convergence point using moving information of population before the population converges.

Suppose all individuals evolve toward the optimal point correctly, the nearest point to the extensions of all their movement directions should locate near the global optimal point. Actually, the estimated point is not exactly on the global optimum due to incorrect directions or inaccurate directions of population movements. However, it is highly expected that the estimation point is close to the global optimum at least for unimodal functions.

Let us begin by defining symbols used for the original estimation method.  $a_i$  and  $c_i$  ( $a_i, c_i \in \mathbb{R}^d$ ) in Fig. 1 are the  $i$ th parent individual and its offspring individual, respectively. The  $i$ th moving vector is defined as a direction vector,  $b_i = c_i - a_i$ . The unit direction vector of the  $b_i$  is given as  $b_{0i} = b_i / \|b_i\|$ , i.e.,  $b_{0i}^T b_{0i} = 1$ .

Let  $x \in \mathbb{R}^d$  be the nearest point to the  $n$  extended directional line segments,  $a_i + t_i b_i$  ( $t_i \in \mathbb{R}$ ); the *nearest* mean that the total distances from  $x$  to the  $n$  extended directional line segments,  $J(x, \{t_i\})$  in Eq. (1), becomes the minimum.

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

$$\min \{J(x, \{t_i\})\} = \sum_{i=1}^n \|a_i + t_i b_i - x\|^2 \quad (1)$$

$$b_i^T (a_i + t_i b_i - x) = 0 \quad (\text{orthogonal condition}) \quad (2)$$

Finally, the estimated convergence point can be calculated using Eq. (3). See detail expansion of equations in the references [6].

$$\hat{x} = \left\{ \sum_{i=1}^n (I_d - b_{0i} b_{0i}^T) \right\}^{-1} \left\{ \sum_{i=1}^n (I_d - b_{0i} b_{0i}^T) a_i \right\} \quad (3)$$

## Convergence point estimation with weighted moving vectors

### Estimation method of a convergence point

Basic estimation method treats moving vectors equally to calculate a convergence point. However, their contribution to estimate a convergence point is different because of various factors; for example, some moving vectors go toward the optimal area directly, while others do not; some are closer to the optimal area, while others are far from there. Each moving vector must have a different reliability in estimating a convergence point according to its reliability. It means that higher importance should be placed on moving vectors having higher reliability, such as having a better direction to the global optimum or near the global optimum. Here, we introduce weights into the basic estimation method of a convergence point and propose two methods for calculating weights for moving vectors.

Suppose a proper weight  $w_i$  is given to the  $i$ th moving vector,  $b_i$ . All notations used in the following calculation are the same with those defined in Sect. 2. Next, we show our stepwise calculation of the weight-based estimation method.

We want to find a  $\hat{x}_w$  making the total distance,  $J_w(x, \{t_i\})$  in Eq. (4), minimal. Note that, the weight is acting on the distance between the point  $\hat{x}_w$  and moving vectors (red line segments in Fig. 1).

$$J_w(x, \{t_i\}) = \sum_{i=1}^n w_i \|a_i + t_i b_i - x\|^2 \quad (4)$$

From Eq. (2), Eq. (5) is obtained.

$$t_i = \frac{b_i^T (x - a_i)}{\|b_i\|^2} \quad (5)$$

Let us put Eq. (5) into Eq. (4) to delete  $t_i$  and obtain Eq. (6).

$$\begin{aligned} J_w(x) &= \sum_{i=1}^n w_i \left\| a_i + \frac{b_i^T (x - a_i)}{\|b_i\|^2} b_i - x \right\|^2 \\ &= \sum_{i=1}^n w_i \left\{ (x - a_i)^T (I_d - b_{0i} b_{0i}^T)^T \right. \\ &\quad \left. (I_d - b_{0i} b_{0i}^T) (x - a_i) \right\} \end{aligned} \quad (6)$$

Next,  $\hat{x}_w$  is obtained by partially differentiating each element of  $x$  and setting them equal 0. Finally, we can obtain the weight-based estimated convergence point  $\hat{x}_w$  using Eq. (7).

$$\hat{x}_w = \left\{ \sum_{i=1}^n w_i (I_d - b_{0i} b_{0i}^T) \right\}^{-1} \left\{ \sum_{i=1}^n w_i (I_d - b_{0i} b_{0i}^T) a_i \right\} \quad (7)$$

### Two methods for calculating weights

Weights have been imported into the basic estimation method successfully in Sect. 3.1. The next key problem is how to calculate weights values. Here, we propose two different perspectives to determine weights. Of course, there must be other ways of calculating the weights.

#### Fitness gradient of moving vectors

Moving vectors themselves contain a lot of information, e.g., evolutionary direction, length of moving vectors, fitness changes, and others. As the first attempt, we use a fitness gradient between the starting point and the terminal point of a moving vector to evaluate the reliability of the moving vector. Bigger fitness gradient of a moving vector means that the direction has a higher probability of approaching the global optimal area. Thus, higher weights should be given to favorable moving vectors. Conversely, lower weights should be given to moving vectors with less fitness gradients. Suppose the fitness change of the  $i$ th moving vector,  $\Delta_i$ , can be calculated as  $\Delta_i = f(c_i) - f(a_i)$ . Then, the fitness gradient information of the  $i$ th individual can be calculated as  $G_i = \frac{\Delta_i}{\|c_i - a_i\|}$ , and Eq. (8) can be used to give a weight to each moving vector.

$$w_i = \frac{G_i}{\sum_{i=1}^n G_i}, \quad (8)$$

where  $n$  represents the total number of moving vectors.

#### Fitness of parents

Generally, individuals with higher fitness may be closer to the global optimal area, while poorer individuals stay away from the area and need more iterations to converge. Poorer individuals are interfered more. For example, the directions of moving vectors in valleys among local minima are influenced by multiple hills and valleys in a fitness landscape and do not always go toward the same local minimum. It is easy to imagine that estimation errors of a convergence point become bigger. Based on this assumption, we roughly think that the closer individuals are to the optimal area, the higher given reliability, i.e., higher weight, should be. Thus, we use Eq. (9) to give a different weight to each moving vector for minimum optimization tasks.

**Algorithm 1** Weight-based estimated convergence point to accelerate EC.  $G$ : generation.

```

1: Generate an initial population.
2: Evaluate the fitness of each individual.
3: for  $G = 1$  to  $MaxGeneration$  do
4:   Obtain the next generation using an EC algorithm.
5:   Calculate moving vectors
6:   Calculate weights for all moving vectors.
7:   Obtain the estimated point using weighted moving vectors.
8:   Evaluate the fitness of the estimated convergence point.
9:   if its fitness is better than that of the worst individual in current
       population then
10:    use the estimated point as an elite individual, and replace the
        worst individual.
11:   end if
12: end for
13: return the optimum

```

$$w_i = \frac{MaxF - f(a_i)}{\sum_{i=1}^n MaxF - f(a_i)} \quad (9)$$

where  $MaxF$  is the worst fitness in the current generation and  $f()$  returns the fitness of an incoming individual.

We introduce weights successfully and explain how to determine weights in detail. The next work is to combine our proposed weight-based estimation method with EC algorithms to accelerate their convergence. Here, Algorithm 1 shows generic flowchart of the EC algorithms combined with our proposal.

## Experimental evaluations

Here, we define shortened names of three methods for comparison:

- Basic method: original estimation method of a convergence point using moving vectors without weighting [6].
- Method 1: proposed method for estimating a convergence point using moving vectors weighted by fitness gradient (see Sect. 3.2.1).
- Method 2: proposed method for estimating a convergence point using moving vectors weighted by fitness of parents (see Sect. 3.2.2).

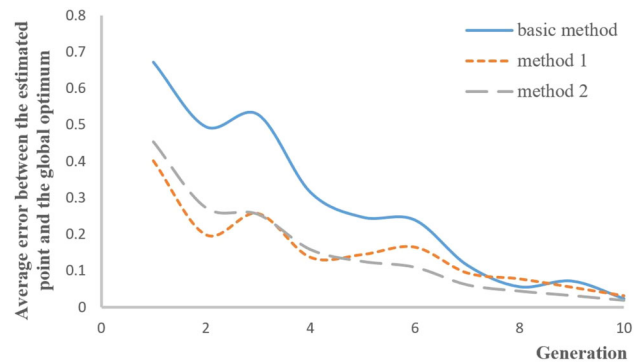
We combine each of three methods with DE and compare them:

### Experiment 1: visual evaluation of estimated convergence points

The first experiment is to visually show the effect of methods 1 and 2 by employing a two-dimensional Gaussian function as a test function. Detailed experimental settings are as follows.

**Table 1** Parameters of Eq. (10)

Dimensions	2-D
Population size	20
Search ranges	$[-6, 6]$ of all 2 variables
$a$	3.1
$\sigma_1$ and $\sigma_2$	(2.0, 2.0)
$\mu_1$ and $\mu_2$	(0.0, 0.0)



**Fig. 2** The average convergence curve of fitness error between the estimated point and the global optimum. See the definition of three method names at the top of Sect. 4

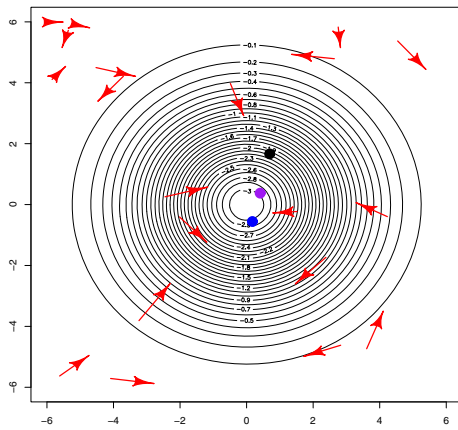
The used Gaussian function is represented by Eq. (10). Its experimental parameters are shown in Table 1.

$$f(x) = - \left\{ a \exp \left( - \sum_{i=1}^2 \frac{(x_i - \mu_i)^2}{2\sigma_i^2} \right) \right\} \quad (10)$$

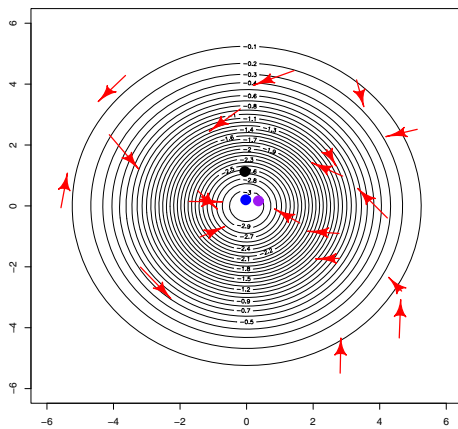
The objective of this Experiment 1 is to evaluate the accuracy of our proposal using exactly the same moving vectors. To do it, we generate a searching point randomly around each individual within a tiny area, compare fitness of the individual and the generated point, and make the poorer and better ones be the starting point and the terminal point of a moving vector, respectively. The number of moving vectors used for estimating a convergence point is the same as the population size. Then, all moving vectors aim the direction of climbing down to the local optimum or the global optimum. This method generating moving vectors without using calculated individuals but generating them in tiny areas increases the number of fitness calculations. However, it is not influenced by local hills thanks to moving vectors in tiny areas [7] and is better for exact evaluation for the Experiment 1.

These evaluations run 30 times over 10 generations. Figure 2 shows the average convergence curve of fitness error between the estimated convergence point and the global optimum. Figure 3 shows convergence points estimated by three methods and exactly the same moving vectors used for three methods.

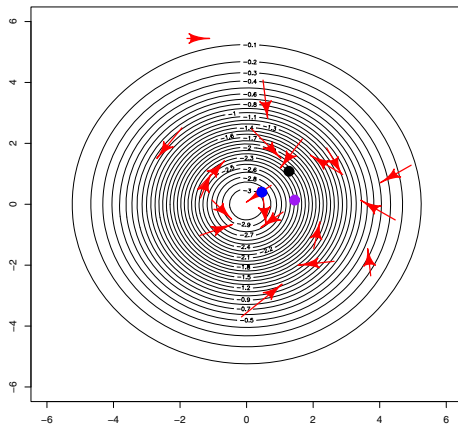




(a) First generation at first trial run



(b) Third generation at first trial run



(c) Fifth generation at first trial run

**Fig. 3** Black dot, purple dot, and blue dot represent the estimated convergence point that without use any weight method, use method 1 for handling weight, and use method 2 for handling weight, respectively

**Table 2** Statistical test results of the Friedman test and the Holm multiple comparison test for average fitness values of 30 trial runs of three methods

Generation	Basic method vs. method 1 and 2
1	Basic method $\approx$ method 2 $\approx$ method 1
2	Basic method $<$ method 2 $<$ method 1
3	Basic method $<$ method 1 $\approx$ method 2
4	Basic method $\approx$ method 2 $\approx$ method 1
5	Basic method $<$ method 1 $\approx$ method 2
6	Basic method $\approx$ method 1 $\approx$ method 2
7	Basic method $\approx$ method 1 $\approx$ method 2
8	Method 1 $\approx$ basic method $\approx$ method 2
9	Basic method $\approx$ method 1 $<$ method 2
10	Method 1 $\approx$ basic method $\approx$ method 2

$A < B$  means that  $B$  is significantly better than  $A$  with significance level 5%, and  $A \approx B$  means that there is no significant difference between  $A$  and  $B$

We apply the Friedman test and the Holm multiple comparison test at each generation to check significant difference among three methods. The results are shown in Table 2.

## Experiment 2: evaluation of EC acceleration with an estimated convergence point

The second experiment is designed to analyze the acceleration effect of proposed weighting methods for EC algorithms, where an estimated convergence point is used as an elite individual, and replace the worst individual when its fitness is better than the worst one. We use 28 benchmark functions from the CEC2013 test suite [12] in this evaluation experiment. Table 3 shows their types, characteristics, variable ranges, and optimum fitness values.

We select DE and PSO as our test baseline algorithms and combine them with our proposals with the parameter setting as described in Tables 4 and 5.

Unlike the Experiment 1, moving vectors are made using existing individuals not to increase fitness calculation cost. Since each target vector in DE generates a trial vector, we set the better one and the poor one as the terminal point and the starting point of a moving vector, respectively; a similar approach is used for PSO. It ensures that each individual can make one moving vector, i.e., the total number of moving vectors is set to the population size. These moving vectors are weighted and used to estimate a convergence point.

For fair evaluations, we evaluate convergence against the number of fitness calls rather than generations. We test each benchmark function with 51 trial runs in four different dimensional spaces. We apply the Friedman test and the Holm multiple comparison test on the fitness values at the stop condition, i.e., the maximum number of fitness calculations, to check whether there is a significant difference among all

**Table 3** Benchmark functions:  
Uni = unimodal, Multi =  
multimodal, Comp. =  
Composition

No.	Types	Characteristics	Optimum fitness
$F_1$	Uni	Sphere function	− 1400
$F_2$		Rotated high conditioned elliptic function	− 1300
$F_3$		Rotated Bent Cigar function	− 1200
$F_4$		Rotated discus function	− 1100
$F_5$	Multi	Different powers function	− 1000
$F_6$		Rotated Rosenbrock's function	− 900
$F_7$		Rotated Schaffers function	− 800
$F_8$		Rotated Ackley's function	− 700
$F_9$		Rotated Weierstrass function	− 600
$F_{10}$		Rotated Griewank's function	− 500
$F_{11}$		Rastrigin's function	− 400
$F_{12}$		Rotated Rastrigin's function	− 300
$F_{13}$		Non-continuous rotated Rastrigin's function	− 200
$F_{14}$		Schwefel's function	− 100
$F_{15}$		Rotated Schwefel's function	100
$F_{16}$		Rotated Katsuura function	200
$F_{17}$		Lunacek bi-Rastrigin function	300
$F_{18}$		Rotated Lunacek bi-Rastrigin function	400
$F_{19}$		Expanded Griewank's plus Rosenbrock's function	500
$F_{20}$	Comp.	Expanded Schaffer's $F_6$ function	600
$F_{21}$		Composition function 1 ( $n = 5$ , rotated)	700
$F_{22}$		Composition function 2 ( $n = 3$ , unrotated)	800
$F_{23}$		Composition function 3 ( $n = 3$ , rotated)	900
$F_{24}$		Composition function 4 ( $n = 3$ , rotated)	1000
$F_{25}$		Composition function 5 ( $n = 3$ , rotated)	1100
$F_{26}$		Composition function 6 ( $n = 5$ , rotated)	1200
$F_{27}$		Composition function 7 ( $n = 5$ , rotated)	1300
$F_{28}$		Composition function 8 ( $n = 5$ , rotated)	1400

**Table 4** DE algorithm  
parameter settings

Population size for 2-D, 10-D, and 30-D search	80
Scale factor $F$	0.9
Crossover rate	0.9
DE operations	DE/rand/1/bin
# of trials	51
Stop condition; max. # of fitness evaluations, $MAX_{NFC}$ , for 2-D, 10-D, and 30-D search	$1000 \times D$

**Table 5** PSO algorithm  
parameter settings

Population size for 2-D, 10-D, and 30-D search	80
Inertia factor $w$	1
Constant $c_1$ and $c_2$	1.4962 and 1.4962
Max. and min. speed $V_{\max}$ and $V_{\min}$	1 and − 1
# of trials	51
Stop condition; max. # of fitness evaluations, $MAX_{NFC}$ , for 2-D, 10-D, and 30-D search	$1000 \times D$

**Table 6** Statistical test result of the Friedman test and the Holm multiple comparison test for average fitness values of 51 trial runs of four methods

	2D	10D	30D	50D
$F_1$	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE1 > DE2 > DE0 $\approx$ DE	DE1 > DE2 > DE0 > DE	DE1 > DE2 > DE0 > DE
$F_2$	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE1 > DE2 > DE0 $\approx$ DE
$F_3$	DE2 $\approx$ DE1 $\approx$ DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE $\approx$ DE0
$F_4$	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 > DE	DE1 $\approx$ DE2 > DE $\approx$ DE0
$F_5$	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE1 > DE2 > DE $\approx$ DE0	DE1 > DE2 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 $\approx$ DE
$F_6$	DE2 $\approx$ DE $\approx$ DE0 $\approx$ DE1	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE1 > DE2 > DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE0 > DE
$F_7$	DE2 $\approx$ DE1 $\approx$ DE $\approx$ DE0	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE2 > DE1 > DE $\approx$ DE0
$F_8$	DE1 $\approx$ DE2 $\approx$ DE $\approx$ DE0	DE2 $\approx$ DE $\approx$ DE0 $\approx$ DE1	DE $\approx$ DE0 $\approx$ DE2 $\approx$ DE1	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE
$F_9$	DE0 $\approx$ DE1 $\approx$ DE2 $\approx$ DE	DE $\approx$ DE1 $\approx$ DE0 $\approx$ DE2	DE $\approx$ DE2 $\approx$ DE1 $\approx$ DE0	DE2 $\approx$ DE1 $\approx$ DE0 $\approx$ DE
$F_{10}$	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE1 > DE2 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 > DE
$F_{11}$	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 > DE
$F_{12}$	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 $\approx$ DE
$F_{13}$	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 $\approx$ DE
$F_{14}$	DE2 $\approx$ DE $\approx$ DE1 $\approx$ DE0	DE $\approx$ DE0 $\approx$ DE1 $\approx$ DE2	DE $\approx$ DE0 $\approx$ DE1 $\approx$ DE2	DE1 $\approx$ DE2 $\approx$ DE $\approx$ DE0
$F_{15}$	DE2 $\approx$ DE1 $\approx$ DE $\approx$ DE0	DE2 $\approx$ DE1 $\approx$ DE0 $\approx$ DE	DE2 $\approx$ DE1 $\approx$ DE $\approx$ DE0	DE1 $\approx$ DE2 $\approx$ DE $\approx$ DE0
$F_{16}$	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE0 $\approx$ DE1 $\approx$ DE $\approx$ DE2	DE $\approx$ DE2 $\approx$ DE0 $\approx$ DE1
$F_{17}$	DE $\approx$ DE2 $\approx$ DE0 $\approx$ DE1	DE2 $\approx$ DE1 > DE0 > DE	DE1 $\approx$ DE2 > DE0 > DE	DE2 $\approx$ DE1 > DE $\approx$ DE0
$F_{18}$	DE1 $\approx$ DE $\approx$ DE2 $\approx$ DE0	DE2 $\approx$ DE1 > DE0 > DE	DE2 $\approx$ DE1 > DE0 > DE	DE1 $\approx$ DE2 > DE0 $\approx$ DE
$F_{19}$	DE2 $\approx$ DE $\approx$ DE1 $\approx$ DE0	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 > DE	DE1 > DE2 > DE0 > DE
$F_{20}$	DE1 $\approx$ DE0 $\approx$ DE2 $\approx$ DE	DE1 $\approx$ DE2 $\approx$ DE $\approx$ DE0	DE1 $\approx$ DE2 > DE $\approx$ DE0	DE2 $\approx$ DE1 > DE $\approx$ DE0
$F_{21}$	DE2 $\approx$ DE1 $\approx$ DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE0 > DE	DE1 > DE2 > DE0 > DE	DE1 > DE2 $\approx$ DE > DE0
$F_{22}$	DE1 $\approx$ DE0 $\approx$ DE $\approx$ DE2	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE1 $\approx$ DE0 $\approx$ DE $\approx$ DE2
$F_{23}$	DE2 $\approx$ DE1 $\approx$ DE $\approx$ DE0	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE1 $\approx$ DE2 $\approx$ DE0 > DE	DE2 $\approx$ DE $\approx$ DE0 $\approx$ DE1
$F_{24}$	DE1 $\approx$ DE $\approx$ DE0 $\approx$ DE2	DE2 > DE1 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE > DE0	DE1 $\approx$ DE2 > DE $\approx$ DE0
$F_{25}$	DE2 $\approx$ DE $\approx$ DE1 $\approx$ DE0	DE2 $\approx$ DE1 > DE $\approx$ DE0	DE2 $\approx$ DE1 $\approx$ DE $\approx$ DE0	DE2 $\approx$ DE $\approx$ DE0 $\approx$ DE1
$F_{26}$	DE1 $\approx$ DE2 $\approx$ DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE2 $\approx$ DE1 > DE0 $\approx$ DE	DE1 $\approx$ DE2 > DE $\approx$ DE0
$F_{27}$	DE1 $\approx$ DE2 $\approx$ DE $\approx$ DE0	DE1 $\approx$ DE2 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 $\approx$ DE
$F_{28}$	DE1 $\approx$ DE2 $\approx$ DE $\approx$ DE0	DE1 $\approx$ DE2 > DE0 $\approx$ DE	DE1 > DE2 > DE0 > DE	DE1 > DE0 $\approx$ DE > DE2

$A > B$  means that  $A$  is significantly better than  $B$  with significant a level of 5%.  $A \approx B$  means that there is no significant difference between  $A$  and  $B$ . DE0, DE1, and DE2 mean (DE + basic method), (DE + method 1), and (DE + method 2), respectively  
 DE conventional DE without any estimated convergence point

algorithms. Tables 6 and 7 show their results of the statistical tests. Besides, we select the average convergence curves of several functions in Fig. 4 to demonstrate convergence characteristics of our proposal.

## Discussions

### Analysis of proposed estimation method using weighted moving vectors

The first discussion is the superiority of our proposed methods for estimating a convergence point. Basic estimation method gives the same weight to moving vectors. Actually, the reliability of moving vectors is different because of various factors, e.g., evolutionary direction error, and their

contribution to the estimated convergence point is also different. Thus, we use weights to enhance or weaken the influence of moving vectors to improve the accuracy of the estimated convergence point; higher weights are given to more reliable moving vectors. Note that, the proposed method adjusts weights adaptively based on the current searching situation without introducing any new control parameters or increasing fitness calculation cost. Increasing the estimation precision of a convergence point may accelerate to find the optimal solution. At least, it is better than the replaced worst individual. From the cost–performance point of view, we can say that our proposed method is a low-risk strategy and easy to use.

The second discussion is on the calculation of weights for moving vectors which affect the performance of our proposal directly. Since incorrect weights reduce the accuracy of an

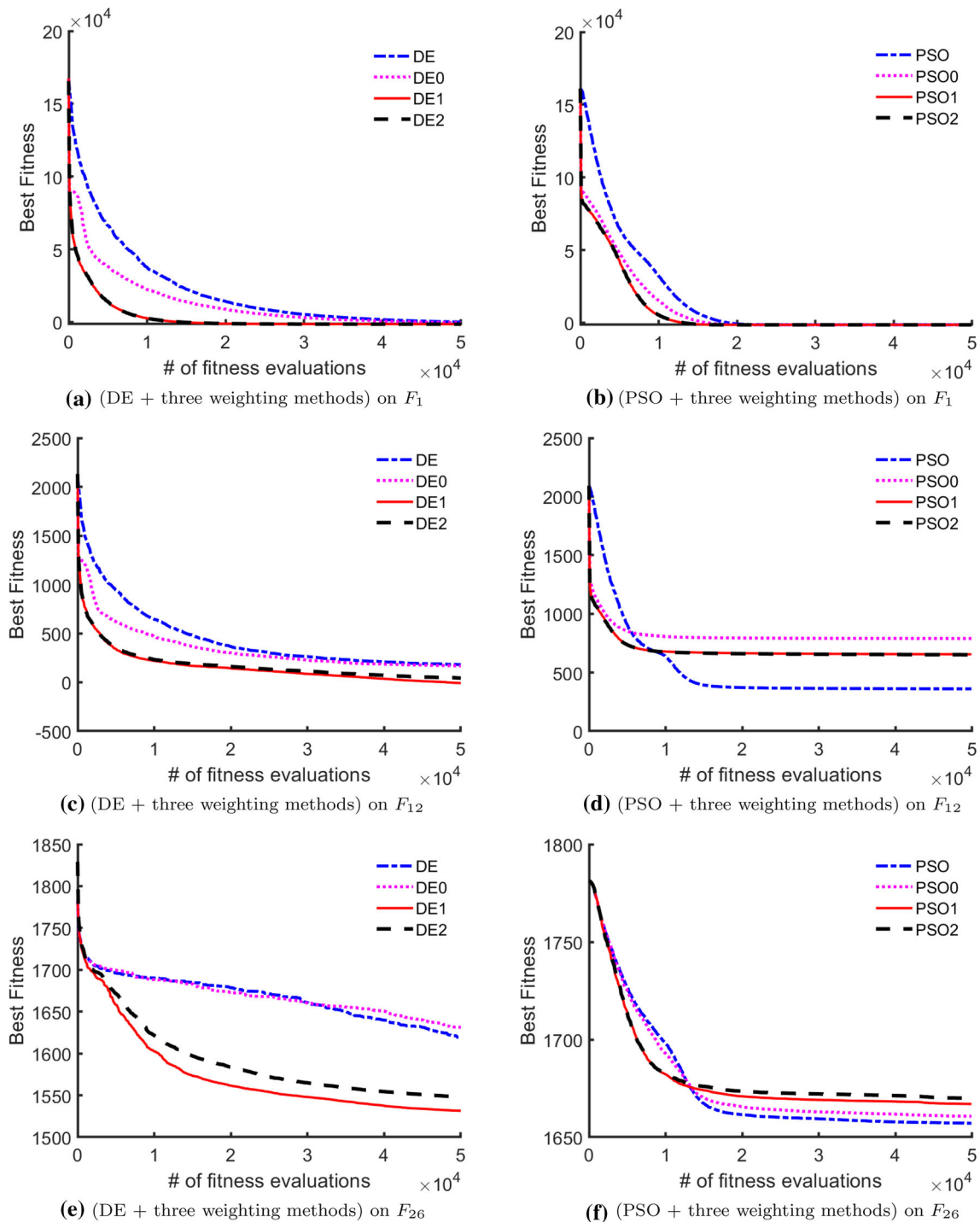


**Table 7** Statistical test result of the Friedman test and the Holm multiple comparison test for average fitness values of 51 trial runs of four methods

	2D	10D	30D	50D
$F_1$	PSO1 $\approx$ PSO2 $>$ PSO $\approx$ PSO0	PSO1 $>$ PSO2 $>$ PSO0 $>$ PSO	PSO1 $>$ PSO2 $>$ PSO0 $\approx$ PSO	PSO1 $>$ PSO2 $>$ PSO0 $\approx$ PSO
$F_2$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO1 $>$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO
$F_3$	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $>$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO
$F_4$	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO $>$ PSO1 $\approx$ PSO0 $\approx$ PSO2	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1
$F_5$	PSO2 $\approx$ PSO1 $>$ PSO0 $\approx$ PSO	PSO0 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO2	PSO1 $>$ PSO0 $>$ PSO2 $>$ PSO	PSO1 $>$ PSO2 $>$ PSO $\approx$ PSO0
$F_6$	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO0	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO0
$F_7$	PSO1 $\approx$ PSO2 $>$ PSO0 $\approx$ PSO	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO1 $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO
$F_8$	PSO2 $\approx$ PSO1 $>$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO $\approx$ PSO0 $\approx$ PSO2	PSO2 $\approx$ PSO0 $\approx$ PSO $\approx$ PSO1	PSO2 $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO
$F_9$	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO $\approx$ PSO2 $\approx$ PSO1 $\approx$ PSO0	PSO2 $\approx$ PSO $\approx$ PSO1 $\approx$ PSO0
$F_{10}$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO1 $>$ PSO0 $>$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO0 $>$ PSO
$F_{11}$	PSO2 $\approx$ PSO1 $>$ PSO $\approx$ PSO0	PSO $>$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO $>$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0
$F_{12}$	PSO2 $\approx$ PSO1 $>$ PSO $\approx$ PSO0	PSO $>$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $>$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO $>$ PSO2 $\approx$ PSO1 $>$ PSO0
$F_{13}$	PSO2 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO0	PSO $>$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO $>$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO $>$ PSO1 $\approx$ PSO0 $\approx$ PSO2
$F_{14}$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO $\approx$ PSO0	PSO $\approx$ PSO1 $\approx$ PSO2 $\approx$ PSO0	PSO1 $\approx$ PSO $\approx$ PSO0 $\approx$ PSO2
$F_{15}$	PSO0 $\approx$ PSO2 $\approx$ PSO1 $\approx$ PSO	PSO0 $\approx$ PSO2 $\approx$ PSO1 $>$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $>$ PSO $\approx$ PSO0
$F_{16}$	PSO0 $\approx$ PSO2 $\approx$ PSO $\approx$ PSO1	PSO2 $\approx$ PSO $\approx$ PSO1 $\approx$ PSO0	PSO2 $\approx$ PSO $\approx$ PSO1 $\approx$ PSO0	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1
$F_{17}$	PSO1 $\approx$ PSO2 $\approx$ PSO $\approx$ PSO0	PSO $>$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0	PSO $>$ PSO2 $\approx$ PSO1 $>$ PSO0
$F_{18}$	PSO2 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO0	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0
$F_{19}$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO2 $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO	PSO2 $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO	PSO0 $\approx$ PSO1 $\approx$ PSO2 $\approx$ PSO
$F_{20}$	PSO $\approx$ PSO2 $\approx$ PSO1 $\approx$ PSO0	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0	PSO $>$ PSO2 $\approx$ PSO1 $\approx$ PSO0	PSO $>$ PSO2 $\approx$ PSO1 $>$ PSO0
$F_{21}$	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO $>$ PSO1 $\approx$ PSO2 $>$ PSO0	PSO1 $>$ PSO $>$ PSO2 $>$ PSO0	PSO0 $>$ PSO2 $>$ PSO $>$ PSO1
$F_{22}$	PSO0 $\approx$ PSO2 $\approx$ PSO1 $\approx$ PSO	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO $\approx$ PSO1 $\approx$ PSO2 $\approx$ PSO0	PSO $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO2
$F_{23}$	PSO2 $\approx$ PSO1 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO1 $\approx$ PSO2 $\approx$ PSO $\approx$ PSO0	PSO2 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO0
$F_{24}$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $>$ PSO	PSO0 $\approx$ PSO $\approx$ PSO2 $\approx$ PSO1	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO2
$F_{25}$	PSO1 $\approx$ PSO2 $\approx$ PSO $\approx$ PSO0	PSO2 $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO	PSO $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO2	PSO2 $\approx$ PSO1 $\approx$ PSO $\approx$ PSO0
$F_{26}$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1	PSO $\approx$ PSO0 $\approx$ PSO1 $\approx$ PSO2
$F_{27}$	PSO0 $\approx$ PSO2 $\approx$ PSO1 $\approx$ PSO	PSO0 $\approx$ PSO2 $\approx$ PSO $\approx$ PSO1	PSO $\approx$ PSO0 $>$ PSO1 $\approx$ PSO2	PSO $\approx$ PSO0 $\approx$ PSO2 $\approx$ PSO1
$F_{28}$	PSO1 $\approx$ PSO2 $\approx$ PSO0 $\approx$ PSO	PSO $>$ PSO1 $\approx$ PSO0 $\approx$ PSO2	PSO $>$ PSO2 $\approx$ PSO0 $\approx$ PSO1	PSO $>$ PSO1 $\approx$ PSO2 $\approx$ PSO0

PSO0, PSO1, and PSO2 mean (PSO + basic method), (PSO + method 1), (PSO + method 1), respectively

PSO conventional PSO without any estimated convergence point. The symbols used in this table have same mean with Table 6



**Fig. 4** The average convergence curves of 50-D  $F_1$ ,  $F_{12}$  and  $F_{26}$  benchmark functions. The symbols used in this figure have same mean with Tables 6 and 7

estimated convergence point, how to give the right weight is a key issue. We proposed two methods in Sect. 3.2 to calculate weights for moving vectors from different perspectives in this paper. The method 1 considers the fitness gradient information of a moving vector, and the method 2 uses the fitness of parent generation. Both of them increase the weights of

potential moving vectors to improve the precision of an estimated convergence point.

Actually, since optimization problems have many complex features, no method is all-powerful for any problems, even for different search periods of the same problem. For example, fitness gradients on some flat places do not work

well to measure the reliability of the directions of moving vectors. Besides, when individuals converge to the optimal area, they become similar and the directions of moving vectors may be even more important. Adaptive choice of the best method among multiple methods may be the best way to calculate weights.

Finally, we discuss on the use of the estimated convergence point to accelerate EC algorithms. Generally, this is an approach of a low risk and a high return. In this paper, we used it to replace the worst individual in the current population. Since only one individual is replaced, it does not change the diversity of the population drastically. Because different EC algorithms have their own characteristics, the impact of our proposal depends on EC algorithms. For DE, an estimated convergence point replaces only the worst individual in the current population and has less impact on other individuals. However, when the estimated convergence point becomes the best individual, it affects all other particles in PSO. Figure 4 supports our hypothesis. Thus, it is accompanied by certain risks, and it may hinder convergence when its accuracy is low. From the experimental results, we observed that our proposal had a fast convergence speed in the early period, but it is easier to fall into local areas in multimodal cases when it is applied to PSO. It may be a good choice to use an estimated convergence point every several generations for PSO to reduce its impact on other individuals to avoid premature. Actually, there are many other ways to use an estimated convergence point to balance convergence and diversity. Thus, how to use the estimated convergence point reasonably and efficiently is one of our future works.

## Analysis of experimental evaluations

The Experiment 1 is designed to visually evaluate the performance of our proposal. From Fig. 3, we can see that introducing weights can increase the accuracy of an estimated convergence point obviously, which implies that appropriate weights can reduce the impact of various errors. Statistical test results show that using proposed weighting method can obtain more accurate estimated convergence point in the early stage but becomes less effective in the later stage. Anyway, using weights does not reduce the accuracy of the estimated convergence point.

There was also no significant difference between the method 1 and the method 2. Maybe individuals become similar according to the convergence, and their difference becomes less significant. We have not been able to completely analyze what fitness landscape situation and environment make our proposed methods less effective. Detail analysis is one of our future works, which may lead to a more appropriate approach to designing weights.

The Experiment 2 is designed to evaluate the acceleration effect of the convergence point estimated by our proposed

method. We designed a set of control experiments to evaluate EC vs. (EC + basic method) vs. (EC + method 1) vs. (EC + method 2). In this paper, we used the estimated convergence point as an elite individual and replace the worst one to accelerate EC convergence. The experimental results confirmed that  $EC < (EC + \text{basic method}) < (EC + \text{method 1 or 2})$ ; a convergence point estimated by weighted moving vectors can further improve acceleration performance. As the dimension increases, the acceleration effect of our proposal becomes more significant; an estimated convergence point plays a more important role for more complex problems.

As we mentioned at the end of the previous subsection, the effectiveness of our proposed method for PSO is different from that for DE. One of our next works is how to combine our method with the characteristics of EC algorithms to accelerate them effectively taking advantage of its applicability to any EC algorithms.

Clustering methods are needed to divide population into multiple subgroups according to each local optimum area of multimodal tasks automatically to further improve performance of our proposal. After that, we apply our proposal to each clustered subgroup, obtain the global or local optimum, and use multiple estimated convergence points to accelerate EC convergence. It will be also one of our future works.

## Potential and future topics

Through these above analyses, we have known that our proposal has achieved satisfactory effects and has great potential. As a new optimization approach, estimating a convergence point, there are still many aspects of improvements. Here, we list a few open topics.

### How to construct moving vectors

It is crucial to determine the acceleration performance of the estimated point. In our paper, we use parent–child relationships to make moving vectors, where one parent generates one offspring individual, and a moving vector starts from a poorer individual to the better one. However, some EC algorithms do not have a one-to-one relationship but a one-to-many or many-to-many relationship. For example, firework algorithm has a one-to-many relation, and genetic algorithm has a many-to-many relationship. Thus, how to construct moving vectors for these EC algorithm is a new topic.

We can use a distance measure to find the nearest individual regardless a parent–offspring relation to make the best moving vectors between two generations. Other methods for making moving vectors are also one of the worth topics to study.

## How to set parameters reasonably

We believe that the number of moving vectors affects the performance of our proposal. Too many or too few may reduce the accuracy of the estimated point. Thus, one of the next research directions is to investigate the relationship among the number of moving vectors, population size, and dimensions.

## How to improve the accuracy of the estimated point

It is the main topic of this paper. As described above, combining with a cluster algorithm may be a feasible approach for further improvement. Besides, there must be other potential methods to achieve this objective, e.g., other methods to calculate weights and different approaches to use an estimated point. We hope that these open topics may give some inspirations to other researchers and attract them to tackle these topics.

## Conclusion

We introduced weights into moving vectors based on their reliability to improve the accuracy of estimating a convergence point. The experimental results confirmed that weighting moving vectors can enhance the accuracy of the estimated convergence point and can further accelerate convergence of EC algorithms.

In our future works, we focus on improving the cost-performance of our proposal, investigating the relations between the accuracy of estimated convergence point and a population size or dimensions. We also try to use the estimated convergence point in various ways to accelerate convergence and develop other methods for weighting moving vectors.

**Open Access** This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

## References

1. Rainer S, Kenneth P (1997) Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. *J Glob Optim* 11(4):341–359
2. Kennedy J, Eberhart R (1995) Particle Swarm Optimization. In: IEEE international conference on neural networks. Perth, Australia, pp 1942–1948
3. Yang XS, Gandomi AH (2012) Bat algorithm: a novel approach for global engineering optimization. *Eng Comput* 29(5):464–483
4. Yu J, Pei Y, Takagi H (2018) Competitive strategies for differential evolution. In: The 2018 IEEE international conference on systems, man, and cybernetics. Miyazaki, Japan, pp 268–273
5. Sörensen Kenneth (2015) Metaheuristics: the metaphor exposed. *Int Trans Oper Res* 22(1):3–18
6. Murata N, Nishii R, Takagi H, Pei Y (2015) Analytical estimation of the convergence point of populations. In: 2015 IEEE congress on evolutionary computation. Sendai, Japan, pp 2619–2624
7. Yu J, Takagi H (2015) Clustering of moving vectors for evolutionary computation. In: 7th international conference on soft computing and pattern recognition. Fukuoka, Japan, pp 169–174
8. Yu J, Pei Y, Takagi H (2016) Accelerating evolutionary computation using estimated convergence point. In: The 2016 IEEE congress on evolutionary computation. Vancouver, Canada, pp 1438–1444
9. Yu J, Takagi H (2017) Estimation of convergence points of population using an individual pool. In: 10th international workshop on computational intelligence and applications. Hiroshima, Japan, pp 67–72
10. Pei Y, Yu J, Takagi H (2019) Search acceleration of evolutionary multi-objective optimization using an estimated convergence point. *Mathematics* 7(2):129–147
11. Yu J, Takagi H (2017) Accelerating interactive evolutionary computation using an estimated convergence point. In: The 33rd fuzzy system symposium. Yonezawa, Japan, pp 47–50 (in Japanese)
12. Liang J, Qu BY, Suganthan PN, Alfredo GHD (2013) Problem definitions and evaluation criteria for the CEC 2013 special session on real-parameter optimization. [http://al-roomi.org/multimedia/CEC\\_Database/CEC2013/RealParameterOptimization/CEC2013\\_RealParameterOptimization\\_TechnicalReport.pdf](http://al-roomi.org/multimedia/CEC_Database/CEC2013/RealParameterOptimization/CEC2013_RealParameterOptimization_TechnicalReport.pdf). Accessed Jan 2013

**Publisher's Note** Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.