Memetic Comp. (2016) 8:45–61
DOI 10.1007/s12293-015-0175-9

REGULAR RESEARCH PAPER

A multi-objective memetic algorithm based on decomposition for big optimization problems

Yutong Zhang[1] · Jing Liu[1] · Mingxing Zhou[1] · Zhongzhou Jiang[1]

Received: 29 July 2015 / Accepted: 7 December 2015 / Published online: 1 February 2016

Abstract When solving multi-objective optimization problems (MOPs) with big data, traditional multi-objective evolutionary algorithms (MOEAs) meet challenges because they demand high computational costs that cannot satisfy the demands of online data processing involving optimization. The gradient heuristic optimization methods show great potential in solving large scale numerical optimization problems with acceptable computational costs. However, some intrinsic limitations make them unsuitable for searching for the Pareto fronts. It is believed that the combination of these two types of methods can deal with big MOPs with less computational cost. The main contribution of this paper is that a multi-objective memetic algorithm based on decomposition for big optimization problems (MOMA/D-BigOpt) is proposed and a gradient-based local search operator is embedded in MOMA/D-BigOpt. In the experiments, MOMA/D-BigOpt is tested on the multi-objective big optimization problems with thousands of variables. We also combine the local search operator with other widely used MOEAs to verify its effectiveness. The experimental results show that the proposed algorithm outperforms MOEAs without the gradient heuristic local search operator.

Keywords Big optimization problems · Decomposition · Evolutionary multi-objective optimization · Gradient methods · Memetic algorithms

✉ Jing Liu
neouma@163.com

Yutong Zhang
495412022@qq.com

1 Key Laboratory of Intelligent Perception and Image Understanding of Ministry of Education, Xidian University, Xi'an 710071, China

1 Introduction

In the era of Big Data, there is an urge of fast data processing methods with optimization. Tackling complex optimization problems, more and more researchers turn to computational intelligent methods, especially to evolutionary algorithms (EAs). However, traditional EAs are facing great challenges for the intolerable costs of thousands even millions of evaluations that blocks the real time application [1]. In this paper, we solve a multi-objective optimization problem (MOP) with thousands of variables which is a big optimization introduced by Abbass et al. [1].

A faster convergence rate is an essential property of any methods solving big optimization problems. Compared with EAs, traditional optimization methods using gradient information show potentials in numerical optimization because their computational cost is low. Decomposition approaches [2,3] transforming an MOP into a set of single objective optimization problems make the gradient-based optimization methods be capable for solving MOPs. It is known that classical numerical optimization methods meet great troubles in approximating entire Pareto fronts (PFs), because only one solution is found. For this issue, the advantages of EAs are obvious since a population is used. Literature [4] and [5] show this property with numbers of applications and instances.

The scheme that employs a local search operator directly improving the individual in EAs is also named as "memetic algorithm" [6,7]. Memetic algorithms have been revealed to be of high-efficiency for discrete optimization in both single and multi-objective optimization [8–10]. For the continuous optimization, the strategy of randomly searching around needs to be altered. The gradient is an immediate heuristic knowledge guiding search without meaningless exploration [11]. It is significant for large scale numerical optimization, and the experiments in this paper will demonstrate this property. Moreover, the population based gradient searching method can overcome the defect of being sensitive to initial points and can approximate the PF simultaneously.

The main contribution of this paper is a new algorithm, named as multi-objective memetic algorithm based on decomposition for big optimization problems (MOMA/D-BigOpt), is proposed. We employ gradient information and the decomposition method to design the local search operator. The designed local search operator improves the robustness and generalization of algorithm. It can deal with more complicated landscapes. Nevertheless, we are not going to design a state-of-the-art new MOEA. We want to demonstrate and analyze the effectiveness and limitation of gradient-based local search operators and gradient information in solving big optimization problems. So the next contribution is that, experimentally, we discuss the issues on the quick convergence of MOEAs in solving multi-objective big optimization problems.

Multi-objective evolutionary algorithm based on decomposition (MOEA/D) is a well-known MOEA with its good performance in convergence [3]. Compared with the differential evolutionary (DE) version of MOEA/D [12], the experimental results show that MOMA/D-BigOpt has better convergence with similar computational cost. When we integrate the specially designed gradient-based operator with some widely used MOEAs, they also get the satisfactory results that reveal the generalization of this operator. In addition, the experiments also show that the gradient directions of the decomposition function used in the local search operator can also guide the search of the algorithms with other decomposition approaches which exceeds our expectations.

The remaining parts of this paper are organized as follows. In Sect. 2, a brief review of hybrid MOEAs especially the employment of gradient information in MOPs is provided. MOMA/D-BigOpt is described in detail in Sect. 3, and the effectiveness of local search operator is discussed in Sect. 4. In Sect. 5, a big data MOP is introduced. Section 6 reports the experimental results on 6 datasets, and Sect. 7 summarizes the work in this paper.

2 Related work

The combination of gradient related methods with MOPs has appeared for many years. Filege and Svaiter did the first analytic description and investigation on the generation of gradient in multi-criteria optimization in 2000 [13]. In 2007, Emmerich et al. [14] put forward the gradient of the 1-D indicator in MOPs concerning the geometric of a set-based Pareto front. Moreover, in 2014, Newton's method was employed to optimize this indicator in [15]. With the similar idea, Bosman [11], in 2011, proposed another set-based multi-objective gradient direction, named as non-dominated direction. A solution can be moved in this direction so that objective function values can be improved or remain the same.

However, the complexity of computing gradient involving set of individuals is unacceptable in dealing with multi-objective big optimization problems. So a straightforward knowledge we would like to use is related to the objective function. Sindhya et al. [16] employed the scalarizing function and gradient-based sequential quadratic programming to implement local search, but, unfortunately, this is also intolerable because the number of evaluations in local search is closely related to the dimensions of decision space. In this paper, the derivations of objective function with the same amount of evaluations and the gradient directed local search with small numbers of evaluations save the computational costs, which is simple but satisfies the demands of big optimization.

Even so, the hybrid strategy in the above literature is an appropriate structure for the integration of other optimization methods and MOEAs. Goh et al. [17] initiatively utilized the evolutionary method to improve the gradient search direction. Tang and Wang [18] referenced the idea from particle swarm optimization and hybridized the personal and global best individual's information to individuals. In the problem we are solving, since a wide distribution of individuals cannot be guaranteed because of the quantity of variables, we did not use these two methods.

Memetic strategies have attracted high attentions in some recent literatures. In [19], a greedy-based local search method was employed in the artificial bee colony algorithm which accelerates the convergence to the best local optimal solutions. Liang et al. [20] emphasized the domain-specific knowledge employed in solving complex optimization problems. Additionally, Liang et al. [21] also proposed a transfer-learning-based evolutionary algorithm that makes full use of knowledge generated by other previous optimizations to promote the distribution of initial population. The method proposed in this paper is also inspired by these ideas and make individuals explore its preference with strong heuristics knowledge, gradient.

For a direct combination of MOEA and gradient-based search, decomposition method becomes our first choice. Decomposition approaches have attracted increasing attentions in the multi-objective, even many-objective [22], optimization society since first comprehensively investigated by Zhang [3]. Some modified versions of MOEA/D improved the original one, like the adaptive operator selection, proposed by Li et al. [23] and a sub-problem decomposition mechanism used by Liu et al. [24].

3 Algorithm description

In MOMA/D-BigOpt, the gradient of objective functions is used as a strong heuristic knowledge guiding the local search operator. The basic framework is inspired by MOEA/D [3], which decomposes the multi-objective problem into a series of single objective problems. Concerning the single objective optimization, we find that in [25,26], the algorithms have good performance in solving large-scale numerical optimization problems and also have potential in solving MOPs when supported by decomposition approach.

3.1 Initialization

During the initialization procedure, each individual is represented by a vector of decision variables, i.e. $\boldsymbol{x} = \{x_1, x_2, \ldots, x_{N_{dim}}\}$ where x_i is a real number and N_{dim} is the dimension of decision space. Besides, a random value evenly distributed in the feasible interval of each dimension is the initial value of x_i.

A set of direction vectors are assigned to each individual that will be used in decomposition function and local search operators, i.e. $\boldsymbol{d}^{(i)} = \{d^{(i)}(1), d^{(i)}(2), \ldots, d^{(i)}(N_{obj})\}$ is the reference direction of individual i. For this vector, $d^{(i)}(j)$, $j = 1, 2, \ldots, N_{obj}$, is a real number ranges from 0 to 1, and N_{obj} is the number of objectives. $\boldsymbol{d}^{(i)}$ is generated uniformly distributed in the objective space by the same means in MOEA/D [3].

Inspired by the lattice structure, proposed in [25], where individuals live in, we create neighborhood lists for individuals that each list contains at least one individual with the direction vector adjacent and at least two individuals with the direction vector not adjacent. The size of each list is set to be 4 which is used in [25].

3.2 Evolutionary operators

The **fitness function** determines which individual is better and should be kept into the next generation. An indicator to evaluate individuals is designed as follows,

$$fit_i = \text{Func}\left(\boldsymbol{f}^{(i)}, \boldsymbol{d}^{(i)}, \boldsymbol{ref}\right) \tag{1}$$

where fit_i, a real number indicator of individual i, corresponds to the fitness value of traditional genetic algorithms, and Func$(\cdot)$ is a decomposition function with independent variables $\boldsymbol{f}^{(i)}$ and $\boldsymbol{d}^{(i)}$, which are the vector of objective function values and direction vector of individual i. $\boldsymbol{ref}$ is the reference information like reference points which is the same for all individuals. More specifically, weighted sum function [2,3] is with the form:

$$fit_i = \sum_{j=1}^{N_{obj}} d^{(i)}(j) \times f^{(i)}(j) \tag{2}$$

where $f^{(i)}(j)$ is the j-th objective function value of individual i.

Other decomposition functions employed by MOEA/D like Tchebycheff approach [2,3], which needs the reference points z^* have weaker differentiability than the weighted sum function. So the weighted sum function becomes our first choice.

Crossover operator is designed with the similar idea in [25]. For individual i, comparisons will be made with its neighbors using following function:

$$cp_{i,j} = fit_i - \text{Func}\left(\boldsymbol{f}^{(j)}, \boldsymbol{d}^{(i)}, \boldsymbol{ref}\right) \quad j \in neig(i) \tag{3}$$

where $cp_{i,j}$ indicates the difference between individual i and its neighbor individual $j$in the direction $\boldsymbol{d}^{(i)}$ and $neig(i)$ is the list of neighbors of individual i assigned before. If $cp_{i,j} > 0$ for all j, individual $i$will not change. Otherwise, with crossover probability P_c, the crossover operation will be applied on individual $i$adding some information from j_{min}, i.e. the neighbor individual generates the minimum $cp_{i,j}$. Here, the strategy employed in [26] is used.

$$e_{(m)}^{(i)} = \alpha_m x_{(m)}^{(i)} + (1 - \alpha_m) x_{(m)}^{(j_{\min})} \tag{4}$$

where $e_{(m)}^{(i)}$ and $x_{(m)}^{(i)}$ are the m-th elements of new and original individual i. Coefficient $\alpha_m = U(0, 1)$ is a random number uniformly distributed in the range of 0 and 1. It is not necessary to assign the same weight on different elements of chromosome. This operator makes use of the information of two individuals and can guarantee the new individual does not beyond the feasible region if the region is convex.

In the **mutation operator**, following the same comparison procedure conducted with (3), all the individuals can be modified with mutation probability P_m who have one or more neighbor individuals with better performance. For example, considering individual i, if there is a neighbor$j$ that $cp_{i,j} < 0$, i will be modified with probability P_m. It means that this individual is not optimal along its direction vector when compared with its neighbors. The same mutation method used in [25] is applied. The operator follows the function:

$$e_{(m)}^{(i)} = \begin{cases} x_{(m)}^{(i)} & U(0,1) < \frac{1}{N_{dim}} \\ bound\left(x_{(m)}^{(i)} + Gaus(0, \frac{1}{n_{gen}})\right) & otherwise \end{cases} \tag{5}$$

where $bound(\cdot)$ is used to make sure the new element stays in the feasible region. If one or more elements of $\boldsymbol{e}^{(i)}$ exceed the bound, we set them to be the boundary values. $Gaus(0, 1/n_{gen})$ is a random number with normal distribution with the mean being 0 and variance being $1/n_{gen}$ where n_{gen} is the number of generations.

The **occupation operator** is motivated by the idea that an individual i will copy another individual around if the neighbor's performance in individual i's direction is better. The comparison function (3) is used again. Individual i, for example, will copy individual j_{min} who has the minimum and negative $cp_{i,j}$; otherwise, if all $cp_{i,j} > 0$, individual i will not change. This operator has the similar idea of related operators employed in [3] and has the identical effect of selection operator in traditional genetic algorithms.

In order to improve the convergence, the **local search operator** makes full use of gradient information, as a strong prior knowledge [11,26] of problems. In MOMA/D-BigOpt, the search direction vector $\boldsymbol{s}^{(i)}$ of individual i, if available, is generated by the following function:

$$s_{(m)}^{(i)} = \frac{\partial \text{ Func}\left(\boldsymbol{f}^{(i)}, \boldsymbol{d}^{(i)}, \boldsymbol{ref}\right)}{\partial x_{(m)}} \tag{6}$$

where $s_{(m)}^{(i)}$ is the m-th element of $\boldsymbol{s}^{(i)}$. This function's structure is similar to the method in [26], but the decomposition functions are used and the direction information and the reference global information, if necessary, are added. If $\boldsymbol{s}^{(i)}$ is unavailable, it will be replaced by a random vector that means the individual will randomly search around.

We use the term "sub-gradient", as is used in [26], to call the search direction vector because, in some cases, there may exists some points with no gradient mathematically. We will discuss this scenario there-in-after.

A naive but sufficient search strategy, Steepest Descent method proposed by Cauchy in 1847, is employed in this operator. The update function is given as follows:

$$e_{(m)}^{(i)} = x_{(m)}^{(i)} + \hat{\theta}\frac{s_{(m)}^{(i)}}{\|\boldsymbol{s}^{(i)}\|} \tag{7}$$

where $\hat{\theta}$ is the optimizing step length along this search direction. $||\boldsymbol{s}^{(i)}||$ is the 2- norm of $\boldsymbol{s}^{(i)}$ generated before.

To reduce the computational cost, this operator is only applied once on each individual in each generation. For the same purpose, we choose a simple liner search method, random sampling which stems from [26], to search for $\hat{\theta}$. The new individual with as small variation as possible and as high fit_i as possible is preferred in the algorithm.

At the very beginning of this liner search method, we need to generate a number of random values of θ_k that have the exponential distribution,

$$\theta_k = \begin{cases} Exp\left(\frac{c \times n_{gen} \times p}{\|\boldsymbol{s}^{(i)}\|}\right) & k = 1, 2, \ldots, N_{try};\ p \geq 1 \\ 0 & k = 0 \end{cases} \tag{8}$$

where positive constant c is the factor need to be specified that controls the length of trial step and should satisfy the scale of searching space, and p is the punishment factor whose function and variation will be introduced later. $c \times n_{gen} \times p/\|\boldsymbol{s}^{(i)}\|$ turns into the parameter of random number generator, $Exp(\lambda)$. N_{try} is the number of times each individual can try in its sub-gradient direction.[1] This is also a parameter need to be specified, and will influence the velocity of convergence. After that, we will evaluate each feasible temporary individual $i(k)$ corresponding to each θ_k. Greedily, we will choose the one with the highest $fit_{i(k)}$ to replace individual i or keep it unchanged if none of $fit_{i(k)} > fit_{i(0)}$, $k > 0$.

We can use some information generated in local search operator as an effective terminal criterion. When an individual reaches the point whose $||\boldsymbol{s}^{(i)}||$ is smaller than a threshold ε, this operator is not conducted. If the individual has chances to be modified in other operators, the local search operator will be conducted on it again; otherwise, the individual will skip local search operation within this generation. The reason why this criterion is necessary is that, in this situation, the local search operator has little chance to improve this individual. When all individuals stop to improve themselves with identical condition, the algorithm stops before the maximum generation is reached. For a specific individual, in the case of θ_0 being chosen as the best step length in the previous iteration, we will enlarge the parameter of random number generator in (8) to shorten step length because the steps tried are identified as longer than the proper length. To achieve this goal, p is set to the square of times of iterations that the individual has not changed continually. When an individual has not changed during the number of iterations exceeding a threshold N_{error}, we believe that this individual has reached the ideal point and we stop to conduct local search operation on it. Algorithm 1 provides the details of MOMA/D-BigOpt, and the local search operator is described in Algorithm 2.

[1] This is the case of maximization problem. For the minimization problem, minus sub-gradient direction will be used.

Algorithm 1: MOMA/D-BigOpt

Input:

Parameters:

N_{indiv}—the population size; P_c—the probability of crossover; P_m—the probability of mutation; N_{try}—the local search trial times; N_{error}—the threshold of the number of iterations that local search is conducted with an oversize step length; c—trial step length controller; ε—the threshold of the 2-norm of sub-gradient; MAX_{gener}—the maximum number of generations.

Output:

***S*—solutions in the final generation**;

```
generation ← 1;
Initialization (S);
for i = 1 to $N_{indiv}$ do
    error[i] ← 0; finish[i] ← 0;
end for;
while generation < $MAX_{gener}$ and not all finish[i] = 1 do
    $S^c$ ← crossover-operator(S, $P_c$);
    for each s ∈ $S^c$ do
        if s changed then
            Calculate $fit_s$;
            error[s] ← 0;
        end if;
    end for;
    $S^m$ ← mutation-operator($S^c$, $P_m$);
    for each s ∈ $S^m$ do
        if s changed then
            Calculate $fit_s$;
            error[s] ← 0;
        end if;
    end for;
    $S^o$ ← occupation-operator($S^m$);
    for each s ∈ $S^o$ do
        if s changed then
            Calculate $fit_s$;
            error[s] ← 0;
        end if;
    end for;
    {S, error, finish} ← local-search-operator($S^o$, error, finish);
    generation ← generation+1;
end while;
return S;
```

Algorithm 2: Local-search-operator in MOMA/D-BigOpt

Input:

S^o—population need to be optimized; *error*[]—a list of integers which record the times of each individual that does not improve itself in the local search operator; *finish*[]—a list of binary variables which are the stop mark for each individual that local search operator will not be conducted on them.

Output:

S^l—optimized population; *error*[]—updated error list; *finish*[]—updated finish list;

```
for $i = 1$ to $N_{indiv}$ do
    if $finish[i] \neq 1$ then
        Generate $\boldsymbol{s}^{(i)}$ for individual $s(i) \in S^o$;
        if $\boldsymbol{s}^{(i)}$ is available and $\|\boldsymbol{s}^{(i)}\| > \varepsilon$ then
            Randomly sampling in direction $\boldsymbol{s}^{(i)}$ with $N_{try}$ times and controlled by $error[i]$;
            Select the best $\theta_k$ and replace $s(i)$;
        else if $\boldsymbol{s}^{(i)}$ is available and $\|\boldsymbol{s}^{(i)}\| < \varepsilon$ then
            $finish[i] \leftarrow 1$;
        else if $\boldsymbol{s}^{(i)}$ is unavailable then
            Generate a random direction $\boldsymbol{s}_{\mathrm{r}}^{(i)}$ and randomly sampling in this direction with $N_{try}$ times and controlled by $error[i]$;
            Select the best $\theta_k$ and replace $s(i)$;
        end if;
        if $\theta_0$ is selected then
            $error[i] \leftarrow error[i] + 1$;
        else if ($\theta_k$, $k > 0$) is selected then
            Calculate $fit_i$;
            $error[i] \leftarrow 0$;
        end if;
        if $error[i] > N_{error}$ then
            $finish[i] \leftarrow 1$;
        end if;
    end if;
end for;
$S^l \leftarrow S^o$;
return $S^l$, $error[]$, $finish[]$;
```

The motivation that we employ the strategies mentioned before is the simplicity and generalization pursued in local search operator. With low in complexity of calculation and small in external data storage, sub-gradient is effective local information of majority of problem and can guide individual to be improved. Finding the optimal solution near an individual is more practical since the gradient of a point in decision space has good performance only in its neighborhood. Low-order derivation is not suitable for approximating landscape globally.

4 Discussion of the local search operator

In the gradient-based memetic algorithm, local search operator is the most powerful technique that improves the convergence with reducing iterations and evaluation times which is meaningful for optimization problems with thousands of variables. The landscape of searching space can be extremely irregular with increasing dimensions. Hence, we will illustrate the generalization of our local search operator in various example functions. Let us observe the following two simple problems:

$$\begin{aligned} &\min \ f_1(x_1, x_2) = \left(\sqrt{x_1^2 + x_2^2} + 1\right)^2 \\ &\text{s.t.} \quad -1 \le x_i \le 1, \quad i = 1, 2 \end{aligned} \tag{9}$$

$$\begin{aligned} &\min \ f_2(x_1, x_2) = -\frac{1}{\sqrt{x_1^2+x_2^2+1}} \\ &\text{s.t.} \quad -1 \le x_i \le 1, \quad i = 1, 2 \end{aligned} \tag{10}$$

Each one is a single-objective problem with box constrains. Since the decomposition function converts the multi-objective problem into several single-objective problems for each individual with different directions, the single-objective problems are proper examples to illustrate some properties of optimization problems and methods. f_1 and f_2 are the continuous optimization problems that objective functions are connective everywhere and derivable everywhere except point (0, 0). The objective function f_1 is convex while the objective function f_2 is not.

Newton's method does not perform well in both two problems while the descent method can deal with these problems. In the first problem, the search direction generated by Newton's method is workable but unchanging step length is misleading and liner search is necessary. However, the second problem cannot be solved by Newton's method for non-positive definition of Hessian matrix. In contrast, sub-gradient can provide a useful search direction in both two problems, and is not sensitive to convexity and the initial point. In addition, sub-gradient is easier to be obtained and stored. In a word, the gradient-based method is more adaptive for complex optimization with the framework of evolutionary algorithm.

The liner search method we employed is suitable for various kinds of extreme value points. How to find extreme value point of objective functions is the core to solve single and multi-objective optimization problems. The theoretical optimal solution of these two problems is $x_1 = 0$ and $x_2 = 0$ obviously. Meanwhile, this is the point without gradient. Calculus knowledge guide us that extreme value points exist in the points with the norm of its gradient being 0, or without gradient, or the boundary of searching area. Since the analytics methods are not sufficient for complexity of real world problems, we need to employ optimization methods even evolutionary algorithms. The challenge is that how could we search for the theoretical optimal solution with the framework of genetic or memetic algorithms. The points with gradient = 0 or without gradient is the most promising points in our algorithm. Owing the fact that the ideal point we find is non-differentiable in the examples and its neighborhood is differentiable without tending to be 0, the norm of gradient being smaller than a threshold and other gradient-based measuring methods are invalid terminal criteria in this condition. Iterative sequence convergence is a widely used method to overcome the challenge above and liner search method used in local search operator, random sampling with adaptive parameter, have the same function.

Additionally, facing the challenge of multimodal scenarios, this local search operator still own the potential to explore local landscape and discover the unexpected region with possibility. Sampling techniques are practical approaches in dealing with the multimodal scenarios, but traditional sampling techniques, which are controlled by the sampling interval and density, are sensitive to the fast changes in landscape. In contrast, the random sampling partly avoids this defect and has probability to discovery the nearest local optimal solution as illustrated in Fig 1. Figure 1a is the part of curve of the following function

$$f(x) = \frac{2}{x + 1.2} \times \sin\left(\frac{10\pi}{x + 0.6}\right) \tag{11}$$

where x corresponds to step length θ mentioned above, and we limit $x \in [0, 5]$. The uniform sampling may lose the nearest minimum while the random sampling with exponential distribution finds it with probability according to the comparison of Fig. 1b, c ($\lambda = 1$), 1d ($\lambda = 2$). Each figure is with sampling in 50 points.

5 Big optimization problems

The test problem stems from the Optimization of Big Data 2015 Competition [1], which has also been used in [26] with

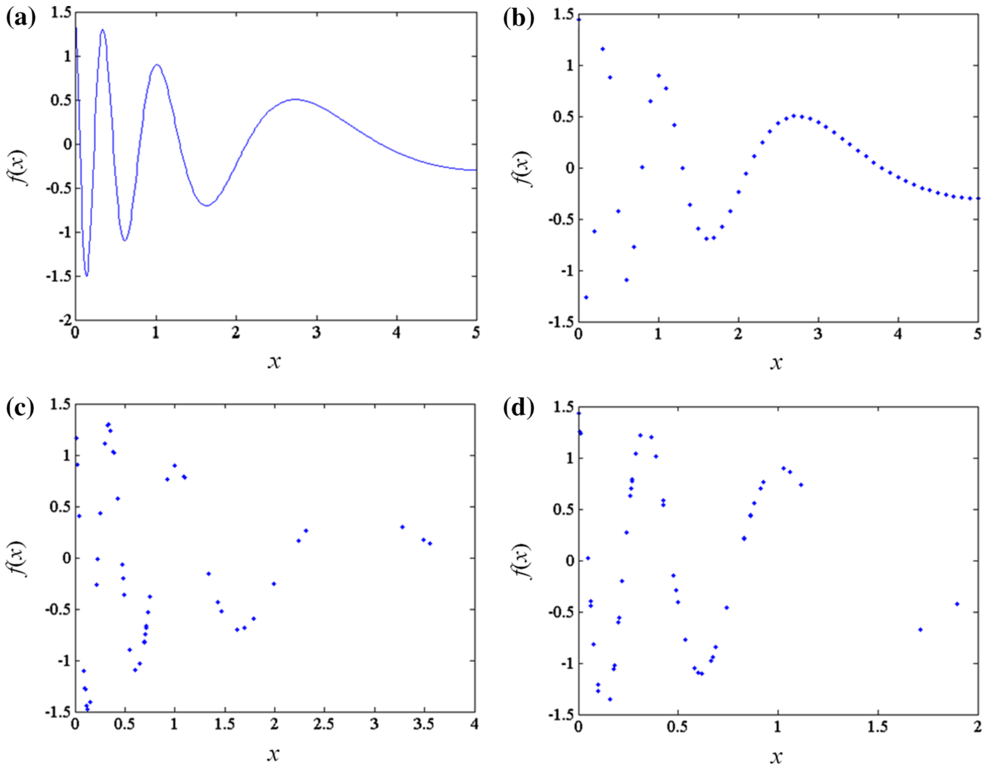


Fig. 1 Comparison of the uniform sampling and the random sampling in a simple multimodal example. **a** The curve of example function, **b** uniform sampling with 50 points from 0 to 5, **c** random sampling with 50 points ($\lambda = 1$), and **d** random sampling with 50 points ($\lambda = 2$)

a single objective optimization problem. In this problem, the model is encompassed with thousands of variables and the number of fitness function evaluations (NFES) is the indicator that we must concern.

The background of this problem is electroencephalographic (EEG) signals processing with independent component analysis (ICA) [27–29]. For the convenience of further usage and analysis, the data of EEG signal should be filtered in real time when it is obtained from sensors.

The mathematic model of this problem with deterministic functions is abstracted as shown below. Two large matrixes, **S** and **X** which are of dimensions $N \times M$, and a relatively small matrix, **A** with dimension being $N \times N$, are the building blocks of the functions. According to the datasets used in the experiments, N ranges in 4, 12, and 19; $M = 256$ is with a constant value. The relationship of these three matrixes is

$$\mathbf{X} = \mathbf{A} \times \mathbf{S} \tag{12}$$

The goal is separating **S** into two matrixes with same dimension that $\mathbf{S} = \mathbf{S}^{(1)} + \mathbf{S}^{(2)}$. Assume that **C** is the matrix of Pearson correlation coefficient

$$C_{i,j} = \frac{covar\left(\mathbf{x}_i, (\mathbf{A} \times \mathbf{S}^{(1)})_j\right)}{\sigma\left(\mathbf{x}_i\right) \times \sigma\left((\mathbf{A} \times \mathbf{S}^{(1)})_j\right)} \tag{13}$$

where $covar(\cdot)$ is the covariance of two vectors, and $\sigma(\cdot)$ is the standard deviation of a vector. The element of **C** in row i and column j is generated by i-th row of matrix **X** and j-th row of matrix $\mathbf{A} \times \mathbf{S}^{(1)}$.

The first objective function is used to generate **C** with maximal diagonal elements and minimal non-diagonal elements. The second objective function is used to minimize the difference between **S** and $\mathbf{S}^{(1)}$. Naturally, the elements of $\mathbf{S}^{(1)}$ are the decision variables. There is a box constrain that each element of $\mathbf{S}^{(1)}$ varies from -8 to 8.

$$\min f_1 = \frac{1}{N^2 - N} \sum_{i,j \neq i} C_{ij}^2 + \frac{1}{N} \sum_i (1 - C_{ii})^2 \tag{14}$$

$$\min f_2 = \frac{1}{N \times M} \sum_{i,j} \left(S_{ij} - S_{ij}^{(1)}\right)^2 \tag{15}$$

Aiming at the objective functions, we apply the aforementioned weighted sum decomposition function, because primal experiments and observations show that the Pareto front tends to be convex in objective space. The sub-gradient for each direction is easy to be generated according to [26].

Assume $\mathbf{T} = \mathbf{A} \times \mathbf{S}^{(1)}$, and $T_{qk} = \sum_l A_{ql} S_{lk}^{(1)}$. The mean and standard deviation of T_q, the q-th row of matrix **T**, is obtained firstly

Table 1 The parameters of MOMA/D-BigOpt used in experiments

Parameters	N_{indiv}	c	N_{error}	N_{try}	P_c	P_m	ε
Value	49	5×10^{-4}	10	30	0.8	0.1	1×10^{-4}

$$\mu(T_q) = \frac{1}{M}\sum_k T_{qk} \tag{16}$$

$$\sigma\left(T_q\right) = \left(\frac{1}{M-1}\sum_k \left(T_{qk} - \mu\left(T_q\right)\right)^2\right)^{\frac{1}{2}} \tag{17}$$

Then the derivative of $\mu(T_q)$ and $\sigma(T_q)$ can be generated by

$$\frac{\partial \mu\left(T_q\right)}{\partial S_{ij}^{(1)}} = \frac{A_{qi}}{M} \tag{18}$$

$$\frac{\partial \sigma\left(T_q\right)}{\partial S_{ij}^{(1)}} = \frac{T_{qk} - \mu\left(T_q\right) A_{qi}}{(M-1)\,\sigma\left(T_q\right)} \tag{19}$$

The covariance of X_p and T_q can be calculated by

$$cover\left(X_p, T_{\mathrm{q}}\right) = \frac{1}{M}\sum_k \left(X_{pk} - \mu\left(X_p\right)\right)\left(T_{qk} - \mu\left(T_q\right)\right) \tag{20}$$

Hence, the derivative of covariance can be obtained by the equation

$$\frac{\partial cover\left(X_p, T_{\mathrm{q}}\right)}{\partial S_{ij}^{(1)}} = \frac{1}{M}\left(X_{pj} - \mu\left(X_p\right)\right) A_{qi} \tag{21}$$

If $\sigma\left(T_q\right) \neq 0$, we can get the derivation of C_{pq} that

$$\frac{\partial C_{pq}}{\partial S_{ij}^{(1)}} = \frac{\left(X_{pj} - \mu\left(X_p\right)\right) A_{qi}}{M \cdot \sigma\left(X_p\right)\sigma\left(T_q\right)} - \frac{\left(X_{pj} - \mu\left(X_p\right)\right) A_{qi} \cdot C_{pq}}{(M-1)\,\sigma\left(T_q\right)^2} \tag{22}$$

Now, the sub-gradient of two objective functions can be obtained.

$$\begin{cases} \frac{\partial f_1}{\partial S_{ij}^{(1)}} = \frac{2}{N^2-N}\sum_{p,q\neq p}\frac{\partial C_{pq}\cdot C_{pq}}{\partial S_{ij}^{(1)}} + \frac{2}{N}\frac{\partial C_{pp}\left(C_{pp}-1\right)}{\partial S_{ij}^{(1)}} \\ \frac{\partial f_2}{\partial S_{ij}^{(1)}} = \frac{2}{N\times M}\left(S_{ij}^{(1)} - S_{ij}\right) \end{cases} \tag{23}$$

In our algorithm, $\boldsymbol{x} = \{x_m\} = \{S_{11}^{(1)}, S_{12}^{(1)}, \ldots, S_{1M}^{(1)}, S_{21}^{(1)}, S_{22}^{(1)}, \ldots, S_{2M}^{(1)}, \ldots, S_{N1}^{(1)}, \ldots, S_{NM}^{(1)}\}$, and $N_{dim} = N \times M$. According to (2), searching directions in local search operator are generated by the following equation,

$$s_{(m)}^{(i)} = -d^{(i)}(1) \times \frac{\partial f_1}{\partial x_m^{(i)}} - d^{(i)}(2) \times \frac{\partial f_2}{\partial x_m^{(i)}} \tag{24}$$

6 Experiments and results

MOMA/D-BigOpt is tested on 6 datasets, namely D4, D4N, D12, D12N, D19, and D19N, which are available online.[2] The "N" in the name of datasets means noisy that each original dataset is added noisy component at the level of 0.1. More details about the noise added can be referenced in [1,30]. With the same population size and parameters of operators, the proposed algorithm outperforms baseline and the DE version of MOEA/D, proposed in [12].The parameters used in the experiments are given in Table 1.

Hypervolume (HV) is one of the most popular indicators that can measure both the convergence and the diversity of the approximation of the PF simultaneously, which is introduced by Zitzler [31]. Because of the unknown of optimal PFs, we cannot apply any indicators that measure the difference between the ideal PF and its approximation, so the hypervolume is the major numerical indicator used in the following experiments.

Firstly, we conduct MOEA/D and improved non-dominated sorting genetic algorithm (NSGA-II) [32] on 6 datasets to construct a baseline in objective space. The number of individuals is set to be 49 in all experiments below. The maximum generation are specified to be 2040 and 40,800, corresponding to 100 thousands (99,960) and 2 millions (1,999,200) of NFES approximately. The maximum number of generations is set to be 50 for the proposed algorithm corresponding to about 80 thousands NFES. The two maximum generations set above can be the reference results of the algorithm without local search. Corresponding to the dataset D4, D12, and D19, Figs. 2, 3, and 4 preliminarily illustrate the insufficiency of MOEA/D without local search operator in solving this problem, and the insufficiency enlarges with the increasing of the number of variables. Given the limitation of space available, the PFs generated by NSGA-II are not plotted in the figures because they are dominated by PFs obtained by MOEA/D, and the distance is too far to be combined in one or two figures. Numerically, we compare the performance of proposed algorithm, MOEA/D with 100k NFES, and NSGA-II with 100k NFES in Table 2 in terms of hypervolume. The reference point is set to be (1.01, 22.34) which is the mean value of two objectives steam from the solutions randomly generated. A great number of experiments show that this reference point is suitable for all 6 datasets.

The above experiments demonstrate the necessity of a gradient-based local search mechanism for solving the big

2 http://www.husseinabbass.net/BigOpt.html.

optimization problem used in this paper. In order to verify the generalization of local search operator, we design the following experiments to transplant this operator to the other MOEAs and the aforementioned 6 datasets are used to test the performance of these combinations with the same parameter setting.

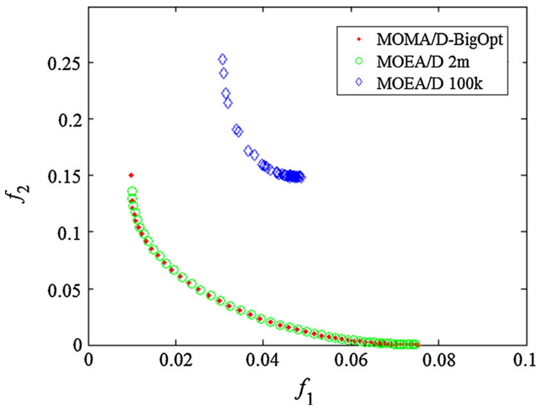


Fig. 2 3 PFs obtained by two methods on D4

Based on the previous experiments, MOEA/D is selected to be added this operator. Since the reference direction has already been used and each direction was assigned to individual, the implementation of this local search operator in the structure of MOEA/D is not hard. At the end of each generation, individual searches in the decision space with the guidance of sub-gradient. The decomposition function employed in MOEA/D is still weighted sum. This modified algorithm is called MOEA/D-local.

Additionally, we choose NSGA-III [33] as another basic framework to be integrated with gradient-based local search operator, because the idea of decomposition is appeared in this basic algorithm and the good performance in traditional test problems shows the potential in dealing with big optimization problems. However, the reference points are not

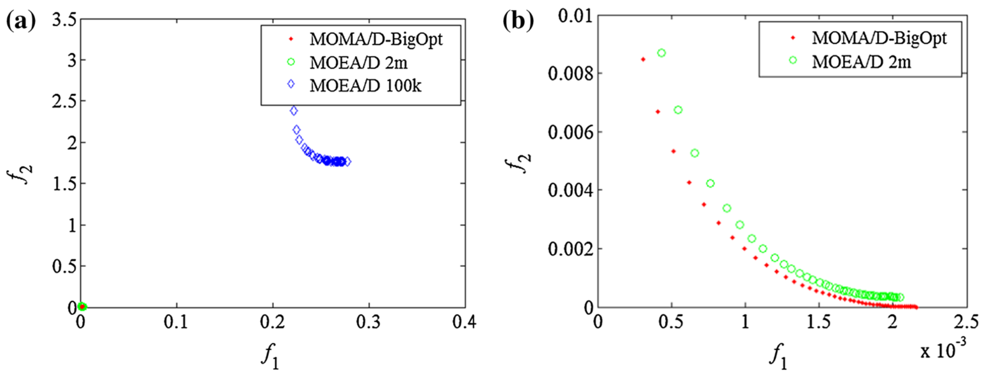


Fig. 3 PFs obtained by two methods on D12. **a** Global view of 3 PFs and **b** partial enlarged drawing of the *bottom left corner* of the global view

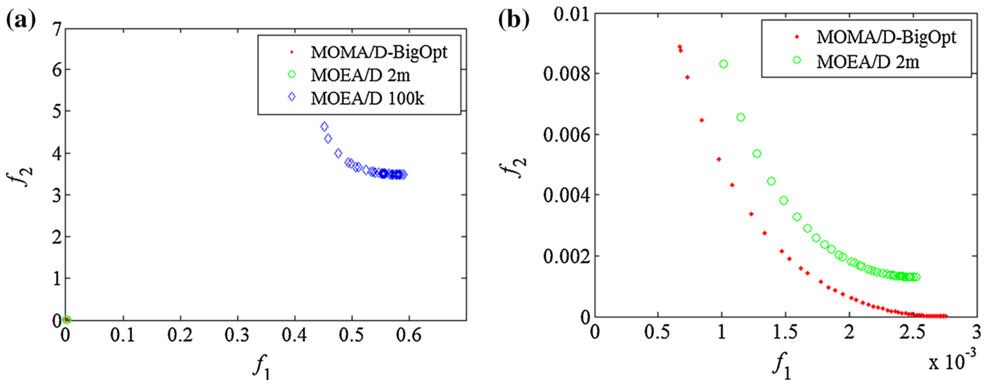


Fig. 4 PFs obtained by two methods on D19. **a** Global view of 3 PFs and **b** partial enlarged drawing of the *bottom left corner* of the global view

Table 2 The hypervolume generated by three algorithms on 6 datasets

Dataset	D4	D4N	D12	D12N	D19	D19N
MOME/D-BigOpt	22.3402	22.3522	22.5586	22.5589	22.5522	22.5514
MOEA/D 100k	21.8301	21.8668	16.6269	16.9594	12.6812	12.1832
NSGA-II 100k	16.3265	16.7924	8.6611	9.2461	7.2814	7.1086

assigned fixedly to individuals, so we employed the association operation as is used in NSGA-III to assign the nearest reference points to individuals in each generation. This modified algorithm is called NSGAIII-local.

What needs to explain is that since the difference of the other original operators in the above comparison algorithms, the stop criteria and amplifying factor p of (8) in gradient-based local search operator employed have to be removed, but the core function of this operator is kept. Therefore, in

Table 3 Hypervolume, standard derivation, and NFES of three algorithms tested on 6 datasets

	D4	D4N	D12	D12N	D19	D19N
MOMA/D-BigOpt						
HV	22.34013	22.35215	22.55864	22.55898	22.55202	22.55121
SD	3.05×10^{-5}	2.08×10^{-5}	1.89×10^{-4}	8.54×10^{-5}	5.43×10^{-4}	5.80×10^{-4}
NFES[a]	39423	39716	78682	78712	77364	77901
MOEA/D-local						
HV	22.34015	22.35216	22.55882	22.55907	22.55306	22.55258
SD	5.60×10^{-6}	7.43×10^{-6}	6.36×10^{-5}	3.95×10^{-5}	1.23×10^{-4}	2.09×10^{-4}
NFES	39249	39249	78449	78449	78449	78449
NSGAIII-local						
HV	22.28587	22.28587	22.53942	22.53918	22.53035	22.52914
SD	0.007593	0.005438	0.003282	0.00269	0.004147	0.004886
NFES	41652	41652	83252	83252	83252	83252

[a] For the uncertainty of proposed algorithm, NFES cannot be fixed for independent running. So it is the average value and is rounded to the nearest integer. The variation of it is less than 5 % of total NFES

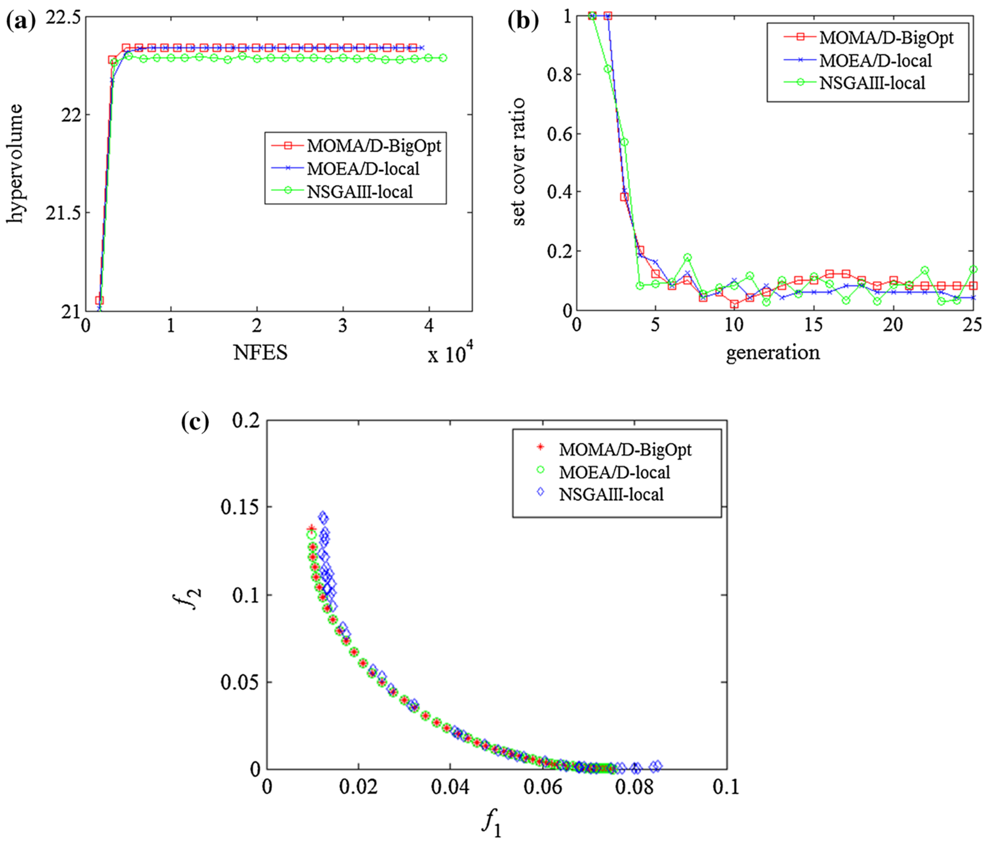


Fig. 5 Performance of three algorithms tested on D4 dataset. **a** Hypervolume vs. NFES, **b** set cover ratio vs. generations, and **c** distribution of population after the last generation in the objective space

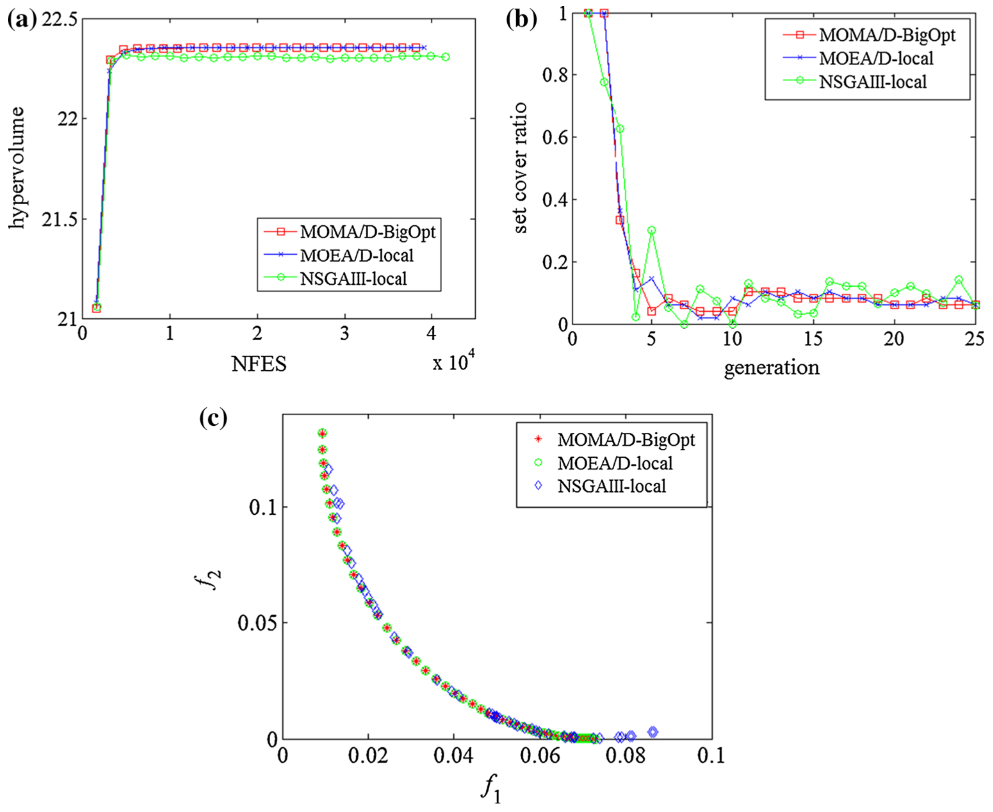


Fig. 6 Performance of three algorithms tested on D4N dataset. **a** Hypervolume vs. NFES, **b** set cover ratio vs. generations, and **c** distribution of population after the last generation in the objective space

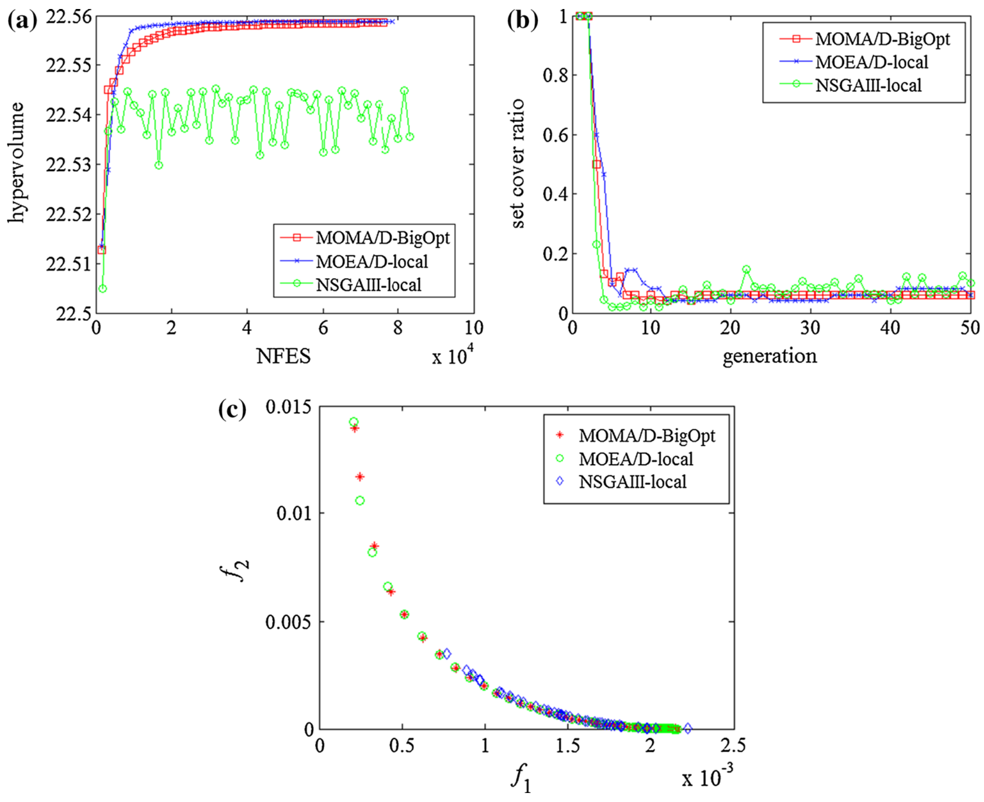


Fig. 7 Performance of three algorithms tested on D12 dataset. **a** Hypervolume vs. NFES, **b** set cover ratio vs. generations, and **c** distribution of population after the last generation in the objective space

each generation, each individual has chance to be applied local search operator constantly. So the NFES in this two modified algorithms are constant but not the same.

The maximum generation of both three tested algorithms are set to be 25, 50, and 50 corresponding to datasets D4 (and D4N), D12 (and D12N), and D19 (and D19N). The reason why we choose the same maximum generation for the large datasets and simply double it compared with the smaller datasets is that the local search operator employed is not sensitive to the quantity of variables. To guarantee the convergence in the lager dataset tests, we simply double the maximum generation.

To analyze the convergence rate of three algorithms, we record the variation of hypervolume with the increasing of NFES, and, on the other hand, the variation of set cover ratio (SCR) over the number of generation [34,35]. SCR is the proportion of individuals in the population of previous generation $(t-1)$ dominated by individuals in the population of current generation (t).

Each algorithm is tested on each dataset with 20 independent runs. The average hypervolume and its standard deviation (SD) obtained by them is given in Table 3. Additionally, the NFES for each algorithm is also shown in the table. Due to the difference of basic structure of algorithms, the number of evaluations is different. In Figs. 5, 6, 7, 8, 9 and 10, the performances of three different algorithms on 6 datasets are compared.

Observing the variation of hypervolume and set cover ratio, we can draw a brief conclusion that all the three algorithms can rapidly converge. The influence of quantity of variables is not obvious. Statistic data show the stability of MOMA/D-BigOpt and MOEA/D-local, while the performance of NSGAIII-local relatively shows the instability. From the distribution of final population, it can be seen that NSGAIII-local dose not converge to the PF evenly, while the indicator variation indicates the convergence of algorithm.

From the PFs generated by the three algorithms, it can be discovered that the distribution of finial population is dense in the optimal value of the second objective and sparse in that of the other objective. Because the decomposition function we employed has the inherent limitation which is sensitive to the shape of PF approximated, especially the scale of different objectives, all the algorithms show an unbalance convergence to these two objectives. More reference directions (or points) concentrate on the objective with wider range in PF, which confirms the results in Table 3. The ratio of range of two objectives in PFs is 0.5 (for D4 and D4N), 0.14 (for D12 and

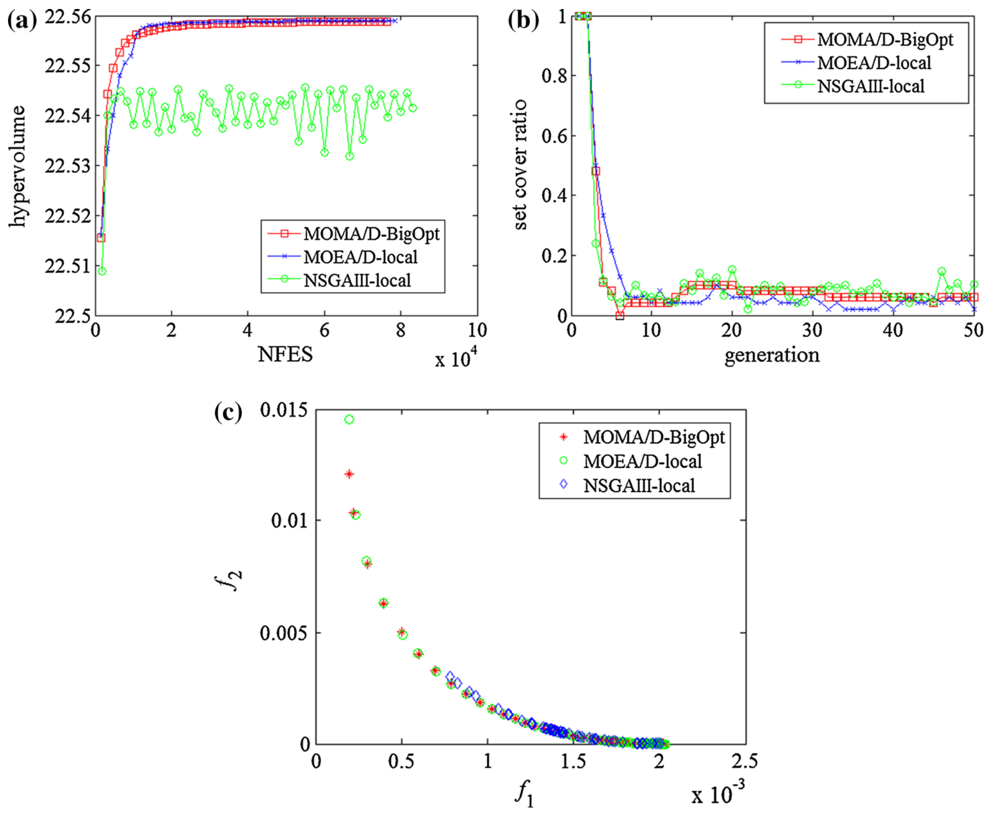


Fig. 8 Performance of three algorithms on D12N dataset. **a** Hypervolume vs. NFES, **b** set cover ratio vs. generations, and **c** distribution of population after the last generation in the objective space

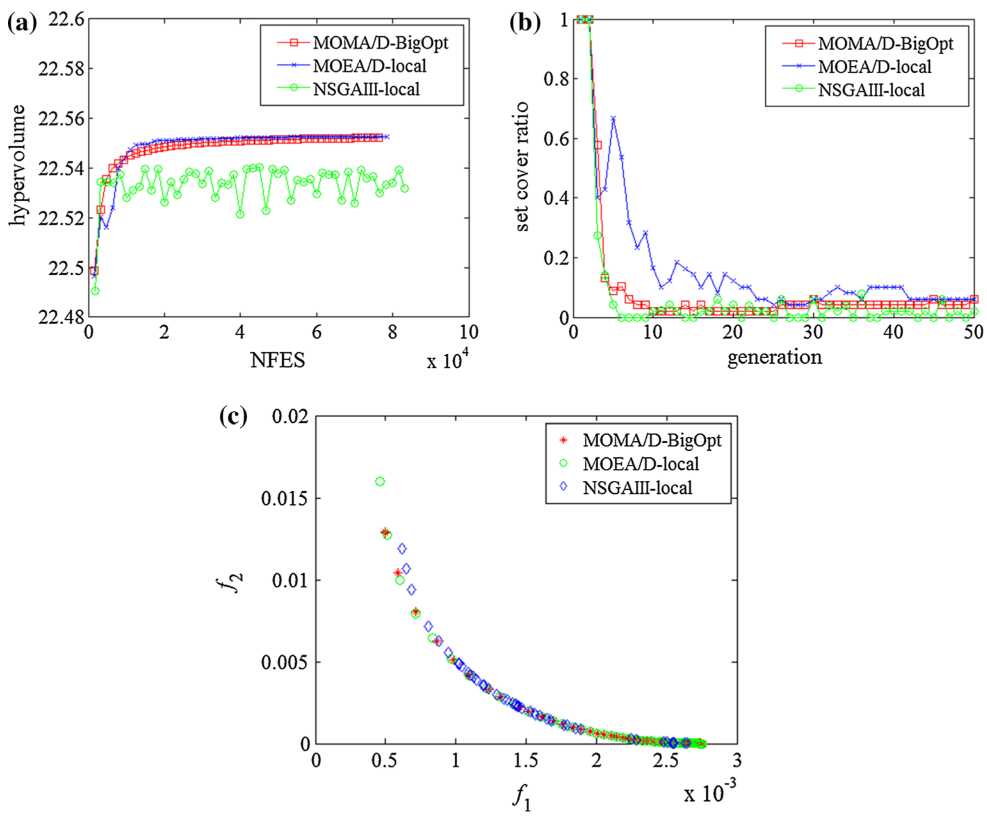


Fig. 9 Performance of three algorithms tested on D19 dataset. **a** Hypervolume vs. NFES, **b** set cover ratio vs. generations, and **c** distribution of population after the last generation in the objective space

D12N), and 0.17(for D19 and D19N) approximately according to our experiments.

Diversity loss is another issue that affects the distribution of Pareto sets. It is more evident in NSGAIII-local because the individuals are assigned with the similar reference points if they are distributing closely before the implementation of local search operator. This unfixed assignment has much worse effects if the diversity of population lost before. It is much easy that individuals aggregate to one or more clusters, and this phenomenon appears in two terminals of PF in D4 and central of PF in D19. Compared with NSGAIII-local, two other algorithms show less diversity losing, but they still need mechanism to maintain or improve it.

In a word, three algorithms can rapidly converge to the uniform PF with the guidance of gradient-based local search operator. The distribution of population after the final generation is different because of the reference direction assignment. Defects of the decomposition function and the unbalance of different objectives influence the density of individual distribution. It can be concluded that a higher convergence velocity, although necessary in big optimization, sacrifices diversity and the current diversity maintain and recovery mechanism does not work very well with small NFES.

In order to generate PFs evenly distributed, we change the decomposition method. However, gradients or sub-gradients of other decomposition methods are not easy to be generated. Therefore, local search direction generation method in the following experiment does not change, but the calculation of fitness function value uses different methods.

The MOEA/D-local framework is selected to conduct this experiment because it has the best performance in the previous experiments. Tchebycheff approach, normal-boundary intersection method (NBI) [3,35], and penalty-based boundary intersection method (PBI) [3] with $\theta = 2$ are employed corresponding to Fig. 11a–c. We also give the result of weighted sum method, which has been shown above, as a reference in Fig. 11d. This experiment is conducted on D4 dataset with the same parameter setting as used before. Moreover, the same experiments are also conducted on D19 dataset because the unbalance of two objectives is much larger and the quantity of variables increases. The population's distributions with different decomposition functions in objective space are plotted in Fig. 12.

Some unexpected PFs obtained by this experiment reveal that the convergence is kept and the distribution is improved. The features of population distribution, born from decomposition methods, are also kept. PBI method shows a uniformly

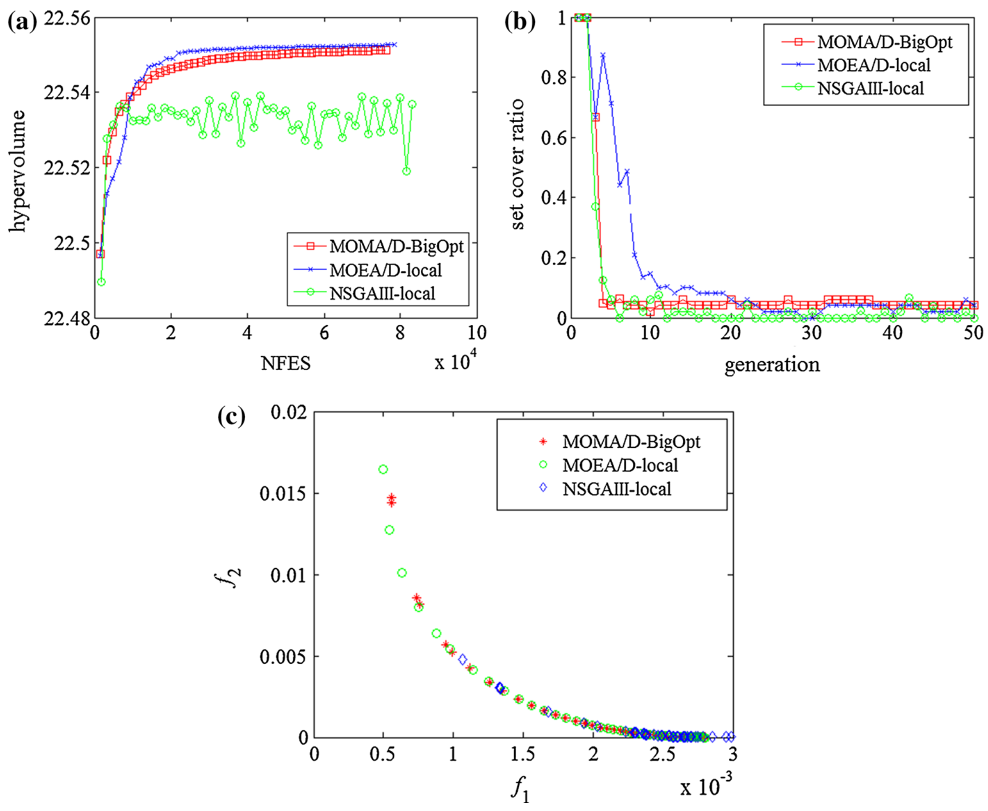
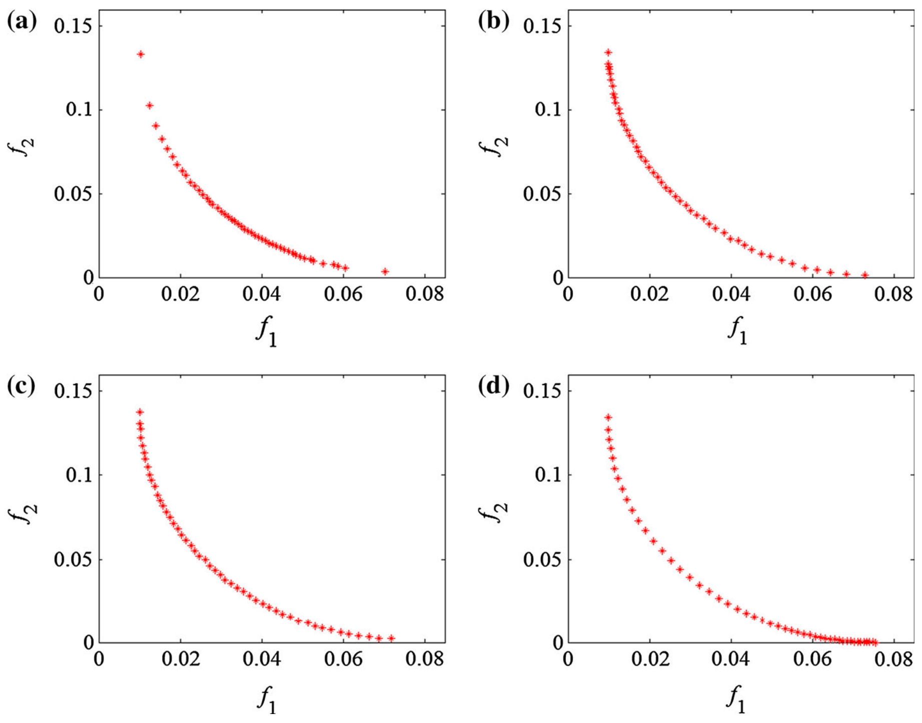


Fig. 10 Performance of three algorithms tested on D19N dataset. **a** Hypervolume vs. NFES, **b** set cover ratio vs. generations, and **c** distribution of population after the last generation in the objective space

Fig. 11 PFs generated by MOEA/D-local with different decomposition functions on D4 dataset. **a** Tchebycheff approach, **b** NBI method, **c** PBI method, and **d** weighted sum method



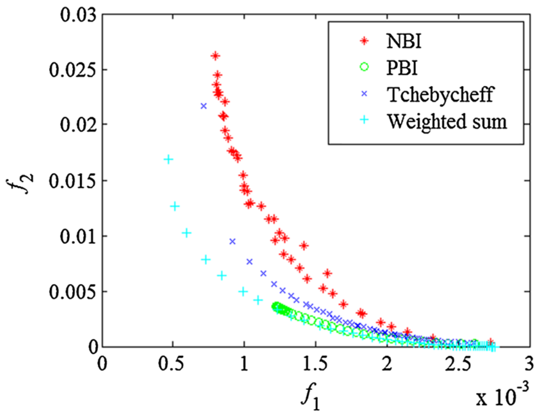


Fig. 12 The distribution of population after the last generation in the objective space obtained by MOEA/D-local with 4 different decomposition functions tested on D19 dataset

distributed PF which is better than other methods on D4 dataset. Although the convergence decreases on D19 dataset, the result is better than our anticipation yet. Compared with other methods, NBI decomposition method gets the best distribution evenness and does not sensitive to the scale of different objectives. The results beyond our expectation because the local search directions are not used in the three other decomposition functions directly. We can find that the guidance from weighed sum function is still workable for some other decomposition functions but the theoretical basis remains to be investigated.

7 Conclusions and futurework

In this paper, we propose a new decomposition-based multi-objective memetic algorithm using sub-gradient information to guide local search operator. Analyses and experimental tests on a big MOP, i.e. EEG data processing, illustrate the effectiveness and efficiency of MOMA/D-BigOpt, and its local search operator. The analysis on more widely used MOEAs with designed local search operator reveals the diversity of population is an important issue for solving big optimization problems.

In the future, more works of us will focus on the research that how to overcome defects of quick convergence in MOEAs. Besides, a more universal and easy-to-be-obtained gradient-based searching direction is interesting to us.

Previous experiments inspire us that the diversity keeping mechanism in MOEAs is worth to be studied. The normalization methods show great improvement in solving optimization problems whose objectives are with unbalanced scale, but the generation of gradient or sub-gradient of this converting function is still a problem. In addition, other decomposition methods only get some preliminary investigation and the analysis does not reach the end.

Acknowledgments This work is partially supported by the Outstanding Young Scholar Program of National Natural Science Foundation of China (NSFC) under Grant 61522311, the General Program of NSFC under Grant 61271301, the Overseas, Hong Kong & Macao Scholars Collaborated Research Program of NSFC under Grant 61528205, the Research Fund for the Doctoral Program of Higher Education of China under Grant 20130203110010, and the Fundamental Research Funds for the Central Universities under Grant K5051202052.

References

1. Goh SK, Abbass HA, Tan KC, Mamun AA (2015) Evolutionary big optimization (BigOpt) of signals. In: Proc. IEEE Congr. Evol. Comput. Sendai, Japan, pp 3332–3339
2. Miettinen K (1999) Nonlinear Multiobjective Optimization. Kluwer, Norwell, MA
3. Zhang Q, Li H (2007) MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. 11(6):712–731
4. Knowles J, Corne D, Deb K (2008) Multiobjective Problem Solving from Nature. Springer-Verlag, Berlin, Germany
5. Coello CAC, Lamont GB, Veldhuizen DAV (2007) Evolutionary Algorithms for Solving Multiobjective Problems. Springer-Verlag, Berlin, Germany
6. Krasnogor N, Hart W, Smith J (2004) Recent Advances in Memetic Algorithms and Related Search Technologies. Springer-Verlag, Berlin, Germany
7. Chen XS, Ong XS, Lim MH, Tan KC (2011) A multi-facet survey on memetic computation. IEEE Trans. Evol. Comput. 15(5):591–667
8. Merz P, Freisleben B (2000) Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. IEEE Trans. Evol. Comput. 4(4):337–352
9. Knowles JD, Corne D (2000) M-PAES: A memetic algorithm for multiobjective optimization. In: Proc. IEEE Congr. Evol. Comput, California, USA, pp 325–332
10. Islam MK, Chetty M (2013) Clustered memetic algorithm with local heuristics for ab initio protein structure prediction. IEEE Trans. Evol. Comput. 17(4):558–576
11. Bosman PAN (2012) On gradients and hybrid evolutionary algorithms for real-valued multiobjective optimization. IEEE Trans. Evol. Comput. 16(1):51–69
12. Li H, Zhang Q (2009) Multiobjective optimization problems with complicated Pareto sets, MOEA/D and NSGA-II. IEEE Trans. Evol. Comput. 15(2):284–302
13. Fliege J, Svaiter BF (2000) Steepest descent methods for multicriteria optimization. Math. Methods Oper. Res. 51(3):479–494
14. Emmerich M, Deutz A, Beume N (2007) Gradient-based/evolutionary relay hybrid for computing Pareto front approximations maximizing the S-metric. In: Hybrid Metaheuristics. Lecture Notes in Computer Science, vol. 4771, pp 140–156
15. Hernández VAS, Schütze O, Emmerich M (2014) Hypervolume Maximization via Set Based Newton's Method. In: EVOLVE-A Bridge between Probability. Set Oriented Numerics, and Evolutionary Computation V, vol. 288, pp 15–28
16. Sindhya K, Miettinen K, Deb K (2013) A hybrid framework for evolutionary multi-objective optimization. IEEE Trans. Evol. Comput. 17(4):495–511
17. Goh CK, Ong YS, Tan KC (2008) An investigation on evolutionary gradient search for multiobjective optimization. In: Proc. IEEE Congr. Evol. Comput. Hong Kong, China, pp 3741–3746

18. Tang L, Wang X (2013) A hybrid multiobjective evolutionary algorithm for multiobjective optimization problems. IEEE Trans. Evol. Comput. 17(1):20–46
19. Jadon SS, Bansal JC, Tiwari R, Sharma H (2015) Accelerating artificial bee colony algorithm with adaptive local search. Memetic Comput. 7(3):215–230
20. Feng L, Ong Y, Lim MH, Tsang IW (2015) Memetic search with interdomain learning: a realization between CVRP and CARP. IEEE Trans. Evol. Comput. 19(5):644–658
21. Feng L, Ong Y, Tan AH, Tsang IW (2015) Memes as building blocks: a case study on evolutionary optimization + transfer learning for routing problems. Memetic Comput. 7(3):159–180
22. Asafuddoula M, Ray T, Sarker R (2015) A decomposition-based evolutionary algorithm for many objective optimization. IEEE Trans. Evol. Comput. 19(3):445–460
23. Li K, Fialho A, Kwong S, Zhang Q (2014) Adaptive operator selection with bandits for a multiobjective evolutionary algorithm based on decomposition. IEEE Trans. Evol. Comput. 18(1):114–130
24. Liu H, Gu F, Zhang Q (2014) Decomposition of a multiobjective optimization problem into a number of simple multiobjectivesubproblems. IEEE Trans. Evol. Comput. 18(3):450–455
25. Zhong W, Liu J, Xue M, Jiao L (2004) A multiagent genetic algorithm for global numerical optimization. IEEE Trans. on Syst., Man, and Cybern., Part B 34(2):1128–1141
26. Zhang Y, Zhou M, Jiang Z, Liu J (2015) A multi-agent genetic algorithm for big optimization problems. In: Proc. IEEE Congr. Evol. Comput, Sendai, Japan, pp 703–707
27. Goh SK, Abbass HA, Tan KC, Al-Mamun A (2015) Decompositional independent component analysis using multi-objective optimization. Soft Computing, pp 1–16
28. Abbass HA (2014) Calibrating independent component analysis with laplacian reference for real-time EEG artifact removal. In: Neural Information Processing. Springer, vol. 8836, pp 68–75
29. Goh SK, Abbass HA, Tan KC, Al Mamun A (2014) Artifact removal from EEG using a multi-objective independent component analysis model. In: Neural Information Processing. Springer, vol. 8834, pp 570–577
30. Goh CK, Tan KC (2007) An investigation on noisy environments in evolutionary multiobjective optimization. IEEE Trans. Evol. Comput. 11(3):354–381
31. Zitzler E, Thiele L (1998) Multiobjective optimization using evolutionary algorithms – a comparative case study. In: Proc. 5th Int. Conf. Parallel Problem Solving from Nature. Springer-Verlag, Berlin, Germany, pp 292-301
32. Deb K, Pratap A, Agarwal S, Meyarivan T (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. 6(2):182–197
33. Deb K, Jain H (2014) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints. IEEE Trans. Evol. Comput. 18(4):577–601
34. Deb K (2001) Multi-objective optimization using evolutionary algorithms. John Wiley & Sons, New York
35. Xiong J, Liu J, Chen Y, Abbass HA (2014) A knowledge-based evolutionary multiobjective approach for stochastic extended resource investment project scheduling problems. IEEE Trans. Evol. Comput. 18(5):742–763
36. Das I, Dennis JE (1998) Normal-boundary intersection: A new method for generating Pareto optimal points in multicriteria optimization problems. SIAM J. Optim. 8(3):613–657