Honda Research Institute EU
Honda Research Institute Europe GmbH
https://www.honda-ri.de/

A Knee-based EMO Algorithm with an Efficient Method to Update Mobile Reference Points

Yu Setoguchi, Kaname Narukawa, Hisao Ishibuchi

2015

Preprint:

This is an accepted article published in The 8th International Conference on Evolutionary Multi-Criterion Optimization. The final authenticated version is available online at: https://doi.org/[DOI not available]

A Knee-based EMO Algorithm with an Efficient Method to Update Mobile Reference Points

Yu Setoguchi[1], Kaname Narukawa[2], and Hisao Ishibuchi[1]

[1] Department of Computer Science and Intelligent Systems,
Graduate School of Engineering, Osaka Prefecture University,
Sakai, Osaka, Japan
{yu.setoguchi@ci., hisaoi@}cs.osakafu-u.ac.jp
[2] Honda Research Institute Europe GmbH
63073 Offenbach am Main, Germany
kaname.narukawa@honda-ri.de

Abstract. A number of evolutionary multi-objective optimization (EMO) algorithms have been proposed to search for non-dominated solutions around reference points that are usually assumed to be given by a decision maker (DM) based on his/her preference. However, setting the reference point needs a priori knowledge that the DM sometimes does not have. In order to obtain favorable solutions without a priori knowledge, "knee points" can be used. Some algorithms have already been proposed to obtain solutions around the knee points. TKR-NSGA-II is one of them. In this algorithm, the DM is supposed to specify the number of knee points as a parameter whereas such information is usually unknown. In this paper, we propose an EMO algorithm that does not require the DM to specify the number of knee points in advance. We demonstrate that the proposed method can efficiently find solutions around knee points.

Keywords: Knee point, preference, reference point, decision maker

1 Introduction

Most of evolutionary multi-objective optimization (EMO) algorithms have been designed to approximate the entire Pareto front. However, if a decision maker (DM) faces a large number of solutions that approximate the entire Pareto front, it would be a difficult task for the DM to choose only a single solution among them. Some EMO algorithms [6, 8] have been proposed to obtain solutions around preference regions that the DM specifies according to his/her preference. Those algorithms do not try to find a large number of solutions. As a result, the DM can decrease a burden to choose a final single solution. However, setting preference regions needs a priori knowledge that the DM sometimes does not have.

As a substitute for the preference information, "knee points" have attracted much attention [1–5, 9]. If the Pareto front has a clear knee point, most DMs

may prefer solutions around the knee point. This is because a small improvement of any objective from the knee point leads to a large deterioration of at least one of the other objectives. Until now, some methods to identify knee points have been proposed [4, 5, 7, 9]. For example, Das [4] proposed the normal boundary intersection method. As shown in Fig. 1, the distance is calculated between the Pareto optimal solutions and the line connecting the extreme points. In minimization problems, the knee point has the largest distance in a convex part.

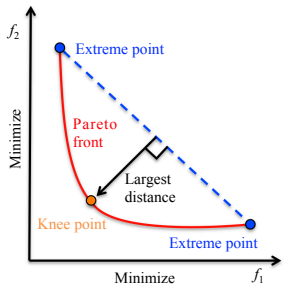


Fig. 1. Normal boundary intersection method.

Trade-off information is also used to identify knee points [9]. The trade-off information can be calculated as follows:

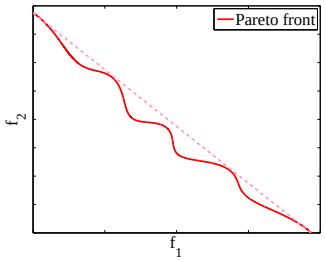
$$T(x_i, x_j) = \frac{\sum_{k=1}^{n} \max\left[0, \frac{f_k(x_j) - f_k(x_i)}{f_k^{\max} - f_k^{\min}}\right]}{\sum_{k=1}^{n} \max\left[0, \frac{f_k(x_i) - f_k(x_j)}{f_k^{\max} - f_k^{\min}}\right]}, \tag{1}$$

where n is the number of objectives, $f_k(x_i)$ indicates the k-th objective value of the solution x_i, $f_k^{\max}$ and $f_k^{\min}$ correspond to the maximum and minimum values of the k-th objective in the population, respectively. Rachmawati and Srinivasan [9] used (2) to calculate a metric in terms of the trade-off for solutions:

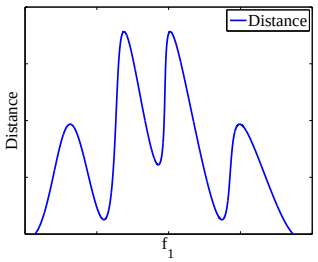
$$\mu(x_i, S) = \min_{x_j \in S, x_i \nprec x_j, x_j \nprec x_i} T(x_i, x_j). \tag{2}$$

We note that x_i and x_j are non-dominated with each other and S indicates the set of solutions x_j in the population. $\mu(x_i, S)$ denotes the least amount of improvement per unit deterioration by substituting any solution x_j from S with x_i. The knee point achieves the largest value of the trade-off metric.

In Figs. 2(a) and 3(a), the red curve shows the Pareto front with four convex regions. It should be noted that we assume a minimization problem. The dashed line in Fig. 2(a) connects the extreme points. We also show the distance from the dashed line to the Pareto front and the trade-off metric calculated by (2) in Figs. 2(b) and 3(b), respectively. The largest value of each metric appears in the regions corresponding to the convex parts.

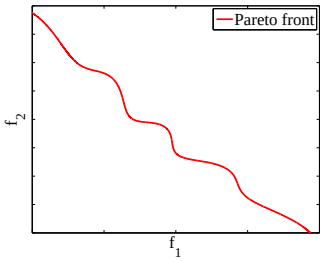


(a) Pareto front

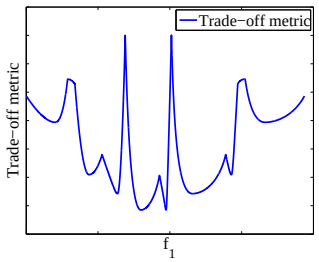


(b) Distance from the dashed line

Fig. 2. Distance metric.



(a) Pareto front



(b) Trade-off calculated by (2)

Fig. 3. Trade-off metric.

Recently, Bechikh et al. have proposed algorithms [1, 2] that are designed to obtain solutions around knee points. In their algorithms, the DM specifies the number of knee points that he/she is searching for, whereas most DMs do not have information about the number of knee points especially in the case of real world problems. As a result, the DM should examine several specifications of the number of knee points. In order to avoid such a situation, we propose an EMO algorithm that does not require the DM to specify the number of knee points. To demonstrate the performance of our proposed method, we apply it to two- and three-objective benchmark problems and a two-objective real world problem.

The rest of this paper is organized as follows. First, we explain a related study in Section 2. Then, we describe our proposed method in Section 3. Section 4 shows experimental results on benchmark problems and a real world problem, and finally we provide conclusions in Section 5. It should be noted that we consider minimization problems throughout this paper.

2 TKR-NSGA-II

Bechikh et al. have proposed TKR-NSGA-II [2], which is designed to obtain solutions around knee points. This algorithm uses the Reference point NSGA-II (R-NSGA-II) procedure [6] as the basic structure of the algorithm. R-NSGA-II is designed to obtain solutions around reference points. In R-NSGA-II, the

DM specifies reference points in advance, while in TKR-NSGA-II, an algorithm updates reference points every generation. This algorithm is called Trade-off-based Mobile Reference Points Updating Strategy (T-MRPUS), which specifies knee-like points as reference points. This algorithm is shown in Algorithm 1.

T-MRPUS uses two parameters, namely N_k and ξ. N_k denotes the number of knee points that are searched for by the DM. ξ denotes the minimum distance between reference points. In this algorithm, extreme solutions and N_k knee-like points are specified as reference points by use of the parameter ξ and the trade-off metric, which is explained in Section 1. We explain the algorithm according to each step in Algorithm 1. First, extreme solutions on the first non-dominated front (FF) are specified as reference points. Then, the trade-off metric is calculated for each solution on the FF and the solutions are sorted in descending order of the trade-off metric. After that, we examine each solution in the sorted order. A solution is chosen as a reference point if it satisfies the following condition: its shortest distance to other reference points is larger than ξ in the normalized objective space. When N_k solutions are selected, the algorithm stops choosing solutions. When the final number of chosen solutions is smaller than N_k, T-MRPUS permits to choose a solution even if its shortest distance to other reference points is not larger than ξ in the normalized objective space.

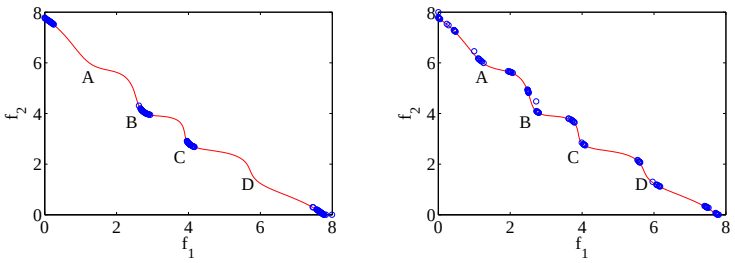
Algorithm 1 T-MRPUS

Input:
FF: the first non-dominated front, N_k: the number of knees,
n: the number of objectives, ξ: the minimum distance between reference points

Output:
MRP: the set of mobile reference points

```
 1: ES ← extreme_solutions(FF, n);
 2: MRP ← ES;
 3: for i = 1 to size(FF) do
 4:    FF(i).trade-off ← evaluate_trade-off_metric(FF(i), FF);
 5: end for
 6: sorted_FF ← sort(FF, descend);
 7: j ← 1, k ← 1;
 8: while (k <= N_k) and (j <= size(FF)) do
 9:    if (not (is_ξ_duplicate(sorted_FF(j), MRP))) then
10:       append sorted_FF(i) to MRP;
11:       k ← k + 1;
12:    end if
13:    j ← j + 1;
14: end while
15: if (size(MRP) < N_k + n) then
16:    for i = (size(MRP)+1) to N_k + n do
17:       append sorted_FF(i) to MRP;
18:    end for
19: end if
```

The DM occasionally consumes a lot of time for specifying the parameter N_k because its appropriate value should be found by trial and error. We demonstrate the difficulty of the parameter setting. As an example, we use the DEB2DK problem [3]. This problem is a two-objective minimization problem and the number of knee points can be arbitrarily specified using a parameter K. In this example, we specify the number of knee points as four (A, B, C, and D in Fig. 4). Since we assume a situation where the number of knee points is unknown, first we specify $N_k = 2$ and $\xi = 0.1$. The obtained solutions are shown in Fig. 4(a). From Fig. 4(a), we can see that the solutions are distributed around the two extreme points and the two knee points B and C. From these solutions, the DM would wonder there are other knee points. Therefore we change the value of N_k from 2 to 10. The obtained solutions are shown in Fig. 4(b). In Fig. 4(b), the solutions are distributed not only around the two extreme points and the four knee points, but also around other regions such as concave parts of the Pareto front. From these solutions, the DM would assume that $N_k = 10$ is larger than the actual number of knee points. Therefore, the next step would be decreasing N_k. Like this, the DM should adjust the parameter N_k until he/she is satisfied with an obtained result. Moreover if the number of objectives is more than three, it is difficult to visualize solutions in the objective space. As a result, it is hard for the DM to find a proper value of N_k.



(a) $N_k = 2$ (b) $N_k = 10$

Fig. 4. Obtained solutions by TKR-NSGA-II on DEB2DK problem.

3 Proposed Method

In this section, we explain our proposed method. There are two main differences between our proposed method and TKR-NSGA-II [2]. One is the basic structure of the algorithm, which corresponds to R-NSGA-II in TKR-NSGA-II. The other is a reference point selection strategy, which corresponds to T-MRPUS in TKR-NSGA-II. We explain the basic structure of the algorithm and the reference point selection strategy in our proposed method.

3.1 Basic Structure

TKR-NSGA-II uses R-NSGA-II as the basic structure of the algorithm, while our proposed method uses the Preference-based NSGA-II (P-NSGA-II) that has recently been proposed by Narukawa et al. [8]. P-NSGA-II uses a hyperplane and Gaussian functions to reflect the DM's preference. In this paper, we modify the selection of next-generation individuals in P-NSGA-II so as to realize a distribution that follows Gaussian functions. The modified P-NSGA-II is explained below.

Preference Configuration In configuring a preference, the DM specifies the center and spread values of Gaussian functions that are lying on the hyperplane. Firstly, we set the hyperplane passing through $(1, 0, ..., 0)$, $(0, 1, ..., 0)$, ..., $(0, 0, ..., 1)$ for an n-objective minimization problem. In Fig. 5, the hyperplane for a two-objective minimization problem is depicted. Next, k n-dimensional Gaussian functions $p_i(\mathbf{v})$ are used to represent the preference of the DM with respect to a vector $\mathbf{v} = (v_1, v_2, ..., v_n)$ on the hyperplane (i.e., $v_1 + v_2 + ... + v_n = 1$) as follows:

$$p(\mathbf{v}) = \max_{i=1}^{k} p_i(\mathbf{v}), \tag{3}$$

$$p_i(\mathbf{v}) = \exp\{-\sum_{l=1}^{n} \frac{(v_l - w_l^i)^2}{(s_l^i)^2}\},\ i = 1, 2, ..., k, \tag{4}$$

where k is the number of the preference regions, n is the number of objectives, $w_l^i \geq 0$ is the center (or the mean) and $s_l^i > 0$ is the spread (or the standard deviation) of the l-th element of the n-dimensional Gaussian function $p_i(\mathbf{v})$. The larger $p(\mathbf{v})$ is, the more $\mathbf{v}$ is preferred. The center vector $\mathbf{w}^i = (w_1^i, w_2^i, ..., w_n^i)$ is specified under the following condition:

$$w_1^i + w_2^i + ... + w_n^i = 1, w_l^i \geq 0 \tag{5}$$

For example, Fig. 5 depicts the preference of the DM for a two-objective optimization problem where a two-dimensional Gaussian function is specified by $\mathbf{w} = (w_1, w_2) = (0.5, 0.5)$ and $\mathbf{s} = (s_1, s_2) = (1.0, 1.0)$. As we can see in the figure, $\mathbf{v}_A$ is more preferred than $\mathbf{v}_B$ since $p(\mathbf{v}_A) > p(\mathbf{v}_B)$ holds in Fig. 5.

Preference Calculation for Solutions In order to calculate how much the DM prefers a solution with an objective vector $\mathbf{f} = (f_1, f_2, ..., f_n)$, $\mathbf{f}$ is first shifted to $\mathbf{f}' = (f_1', f_2', ..., f_n')$, where

$$f_l' = f_l - f_l^{\min},\ l = 1, 2, ..., n, \tag{6}$$

and $f_l^{\min}$ is the minimum value found so far for the l-th objective. Next, $\mathbf{f}'$ is normalized to $\mathbf{f}'' = (f_1'', f_2'', ..., f_n'')$ as follows:

$$f_l'' = \frac{f_l'}{f_l'^{\text{ext}}},\ l = 1, 2, ..., n, \tag{7}$$

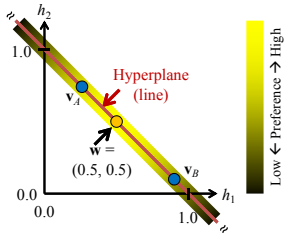


Fig. 5. Preference on a hyperplane (line) in two-objective optimization.

where f'^{ext}_l is the l-th objective value of an extreme vector $\mathbf{f}'^{\text{ext}}_l$ for the l-th objective. The extreme vector $\mathbf{f}'^{\text{ext}}_l$ is identified by finding a vector that minimizes the following achievement scalarizing function ASF with a weight vector $\mathbf{d}^l = (d^l_1, d^l_2, ..., d^l_n)$ indicating the l-th objective direction (e.g., $\mathbf{d}^1 = (d^1_1, d^1_2, ..., d^1_n) = (1, 0, ..., 0)$):

$$ASF(\mathbf{f}', \mathbf{d}^l) = \max_{i=1}^{n} f'_i / d^l_i, \tag{8}$$

where $d^l_i = 0$ is replaced with $d^l_i = 10^{-6}$ for avoiding a division by zero. Then, the hyperplane passing through n extreme vectors is generated and $\mathbf{f}''$ is mapped to $\mathbf{v} = (v_1, v_2, ..., v_n)$ on the hyperplane as follows:

$$v_i = \frac{f''_i}{f''_1 + f''_2 + ... + f''_n}, \ i = 1, 2, ..., n. \tag{9}$$

It should be noted that $v_1 + v_2 + ... + v_n = 1$ holds because $\mathbf{v}$ is on the hyperplane. These coordinate transformations are depicted in Fig. 6. Now the preference of the DM on the solution having the objective vector $\mathbf{f}$ is calculated as $p(\mathbf{v})$ in (3)-(4).

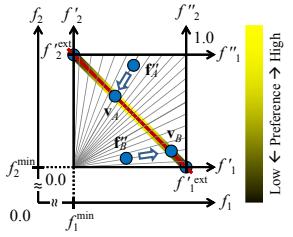


Fig. 6. Coordinate transformations of solutions.

Selection Method In the selection, first we choose extreme solutions that are needed for constructing a widespread hyperplane. Then, we pick up solutions

according to their ranks in the same manner as in NSGA-II. Let us assume S_i is the set of solutions having rank i. Next, we consider the selection among S_L. The value of L is the minimum integer that makes $|S_1 \cup S_2 \cup ... \cup S_L| \geq \mu$ hold, where μ is the parent population size. In the selection among S_L, we use an allowable radius and a shortest distance.

The allowable radius is defined for a solution having an objective vector $\mathbf{f} = (f_1, f_2, ..., f_n)$ as follows:

$$r(\mathbf{v}) = 1 - p(\mathbf{v}), \tag{10}$$

where $\mathbf{v}$ is obtained from $\mathbf{f}$ by (6)-(7), (9), and $p(\mathbf{v})$ is calculated by (3)-(4). The shortest distance $d(\mathbf{v})$ is defined as the Euclidean distance of a solution to its nearest solution on the hyperplane. The allowable radius and the shortest distance are illustrated in Fig. 7, where a dashed line represents the preference function $p(\mathbf{v})$ in (3).

In the selection among S_L, first we randomly choose a solution from S_L and calculate its allowable radius and shortest distance. If $r(\mathbf{v}) > d(\mathbf{v})$ holds, the solution is removed. Otherwise, another solution is randomly selected prohibiting the same solution(s) to be chosen, and then applied to the same operation. If any solutions are not removed, the solution with the minimum value of $d(\mathbf{v})$ is removed. We iterate this procedure until $|S_1 \cup S_2 \cup ... \cup S_L| = \mu$ holds. By removing solutions whose shortest distances are smaller than the allowable radius, it is expected that the remaining solutions follow the distribution of $p(\mathbf{v})$ in (3).

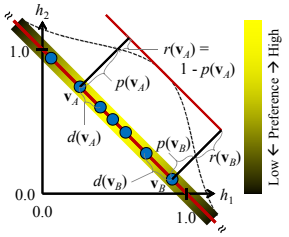
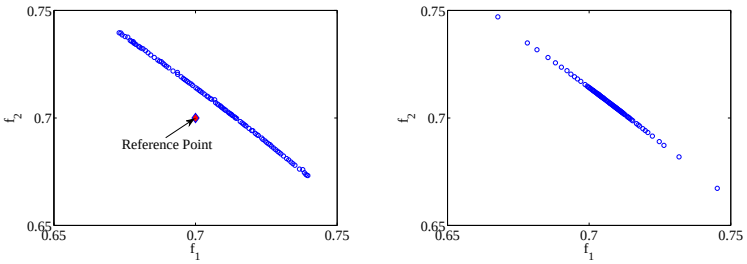


Fig. 7. Allowable radius and shortest distance for solutions $\mathbf{v}_A$ and $\mathbf{v}_B$.

We explained the procedure of P-NSGA-II. Next, we apply R-NSGA-II and P-NSGA-II to DTLZ2 problem with two objectives and compare the obtained solutions. In R-NSGA-II, a reference point is specified as $(0.7, 0.7)$. In P-NSGA-II, $\mathbf{w}$ and $\mathbf{s}$ are specified as $(0.5, 0.5)$ and $(0.5, 0.5)$, respectively. In both cases, the preference region is specified as the center of the Pareto front.In the case of R-NSGA-II, solutions are uniformly distributed around a reference point like Fig. 8(a). In the case of P-NSGA-II, the distribution of obtained solutions is similar to a Gaussian function as shown in Fig. 8(b), where they are densely distributed around the center of a preference. This distribution is useful when we search for solutions around knee points. This is because most DMs may prefer solutions that are dense around the knee point.



(a) R-NSGA-II [6] (b) P-NSGA-II

Fig. 8. Obtained solutions by R-NSGA-II and P-NSGA-II.

3.2 Reference Point Selection Strategy

For the selection of reference points, TKR-NSGA-II uses a selection strategy called T-MRPUS, which picks up solutions having a large trade-off metric as explained in Section 2, whereas our proposed method uses a selection strategy which eliminates solutions having a small distance from the hyperplane connecting extreme solutions. We call this selection strategy Distance-based Elimination MRPUS (DE-MRPUS). The algorithm of DE-MRPUS is shown in Algorithm 2.

We explain the algorithm according to each step in Algorithm 2. First, extreme solutions are identified and the distance from the hyperplane connecting the extreme solutions is calculated for each solution on the first non-dominated front (FF). After calculating the distance, the extreme solutions are removed from the FF. This is because, the extreme solutions are necessary just for calculating the distance in this algorithm. Next, the remaining solutions are sorted in ascending order of the distance. We examine each solution in the sorted order. A solution is removed if it satisfies the following condition: its shortest distance to other remaining FF solutions is not larger than ξ in the normalized objective space. ξ is a parameter which denotes the minimum distance between reference points. In Fig. 9, we show a flow of removing the solutions. Consequently, we may obtain knee-like points in each convex part as shown in Fig. 9.

After examining all solutions, we discriminate the remaining solutions based on distances that are calculated in the first step of this algorithm. For normalization, the distances of the remaining solutions are divided by the largest distance of the remaining solutions. If they have a larger normalized distance than a threshold T, they are specified as reference points. In this algorithm, the DM can control a distribution of solutions by specifying the threshold T. For example, when we use a large value as the threshold T, the DM only obtains solutions around knee points having a large distance from the hyperplane connecting extreme solutions. This result would be nice for the DM who wants to obtain solutions around knee points that have the largest distances from the hyperplane connecting extreme solutions.

In this algorithm, we can use other metrics such as the trade-off metric to identify knee-like points. In computational experiments, we examine the use of

Algorithm 2 DE-MRPUS

Input:

FF: the first non-dominated front, T: the threshold to discriminate reference points, n: the number of objectives, ξ: the minimum distance between reference points

Output:

MRP: the set of mobile reference points

```
ES ← extreme_solutions(FF, n);
for i = 1 to size(FF) do
    FF(i).distance ← normal_boundary_intersection_method(FF(i), ES);
end for
remove ES from FF;
sorted_FF ← sort(FF, ascend);
j ← 1;
while (j <= size(FF)) do
    if (is_ξ_duplicate(sorted_FF(j), sorted_FF)) then
        remove sorted_FF(j) from sorted_FF;
    else
        j ← j + 1;
    end if
end while
for i = 1 to size(sorted_FF) do
    sorted_FF(i).normalized_distance ← normalize(i, sorted_FF);
    if (T < sorted_FF(i).normalized_distance) then
        append sorted_FF(i) to MRP;
    end if
end for
```

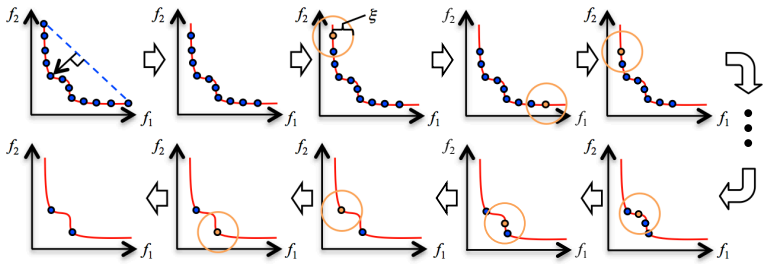


Fig. 9. Flow of removing the solutions.

Table 1. Compositions of each algorithm.

Algorithm	Basic structure	Reference point selection	Metric
TKR-NSGA-II [2]	R-NSGA-II [6]	T-MRPUS [2]	Trade-off [9]
Proposed method	P-NSGA-II	DE-MRPUS	Distance [4]
	P-NSGA-II	TE-MRPUS	Trade-off [9]

the trade-off metric in Fig. 3 as well as the distance metric in Fig. 2. When we use the trade-off metric, we call it Trade-off-based Elimination MRPUS (TE-MRPUS). Compositions of each algorithm are shown in Table 1.

4 Computational Experiment

In this section, we apply our proposed method to two- and three-objective benchmark problems and a two-objective real world problem. In order to examine the performance, we also apply TKR-NSGA-II [2] to those problems and compare the results. In all experiments, the population size is specified as 100. We use the SBX with a distribution index of 15 and the polynomial mutation with a distribution index of 20. The crossover probability is set to 0.5 and the mutation probability is set to $1/m$,where m is the number of decision variables. In this study, we use parameter values in Table 2 unless otherwise specified.

Table 2. Specification of parameters.

Algorithm	Parameter values
TKR-NSGA-II [2]	$\xi = 0.1$, $\epsilon = 0.001$, $N_k =$ actual number of knee points
Proposed method	$\xi = 0.1$, $\mathbf{s} = (0.5, ..., 0.5)$, $T = 0$

4.1 Benchmark Problem

Branke et al. [3] have proposed three knee-based test problems, namely DO2DK, DEB2DK and DEB3DK problems to evaluate the performance of knee-based EMO algorithms. The DO2DK and the DEB2DK are two-objective problems and the DEB3DK is a three-objective problem. In all problems, the number of knee points can be arbitrarily specified using a parameter K: the number of knee points is specified as K^{n-1}, where n is the number of objectives. Rachmawati and Srinivasan [9] modified those problems and proposed DO2DK-1, DEB2DK-1 and DEB3DK-1 problems that require an ability to converge towards the Pareto front. In this paper, we only show the results on DEB2DK, DEB2DK-1, and DEB3DK problems due to the paper length limitation.

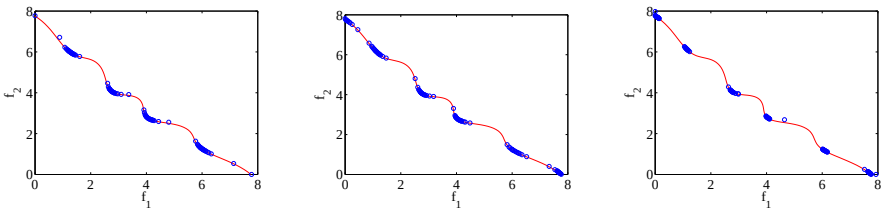
First, we consider the DEB2DK where the number of knee points is specified as four. The stopping criterion is specified as 200 generations. We show solutions obtained by Distance Elimination and Trade-off Elimination in Figs. 10(a) and 10(b), respectively. From Figs. 10(a) and 10(b), we can see that both approaches obtain solutions around knee points. Moreover we show solutions obtained by TKR-NSGA-II in Fig. 10(c). We can see that the distribution of solutions is similar to those obtained by Distance Elimination and Trade-off Elimination. However it should be noted that TKR-NSGA-II needs a trial-and-error adjustment for the parameter N_k.

Next, we change a value of the threshold T from 0 to 0.8 in our proposed method. When the threshold T is set to a large value, we only obtain solutions around knee points that have a large distance metric or trade-off metric as shown in Figs. 11(a) and 11(b). We also show solutions obtained by TKR-NSGA-II, where N_k is specified as 2 in Fig. 11(c). From Fig. 11(c), we can see that TKR-NSGA-II also obtains solutions around knee points that have a large trade-off metric. However for obtaining such a result, the DM should specify the number of knee points having a large trade-off metric, whereas the DM usually does not have such information. Moreover, we show solutions around the knee point B in Fig. 12. We can see that the distribution of obtained solutions is similar to a Gaussian function in our proposed method.

We also check an effect of the spread vector that specifies the Gaussian function. We specify the spread vector as $\mathbf{s} = (0.3, ..., 0.3)$, $\mathbf{s} = (0.5, ..., 0.5)$ and $\mathbf{s} = (0.7, ..., 0.7)$ and show the obtained solutions in Figs. 13(a), 13(b) and 13(c), respectively. In all experiments, Distance Elimination is used. We can see that when $\mathbf{s}$ is large, solutions are widely distributed around knee points, while when $\mathbf{s}$ is small, solutions are narrowly distributed around knee points.

Then, we solve the DEB2DK-1 where the number of knee points is specified as four. We specify the stopping criterion as 1000 generations. We show the obtained solutions in Fig. 14. From Fig. 14, we can see that solutions are converging towards the four knee points in our proposed method and TKR-NSGA-II.

Finally, we solve the DEB3DK where the number of knee points is specified as 1. The stopping criterion is specified as 300 generations. The obtained results are shown in Fig. 15. From Fig. 15, we can confirm that Distance Elimination and TKR-NSGA-II obtain solutions around the knee point existing in the center of the Pareto front. Trade-off Elimination obtains solutions not only around the knee point, but also around many other regions. It is not a good characteristic if the DM only wants to know information around knee points.

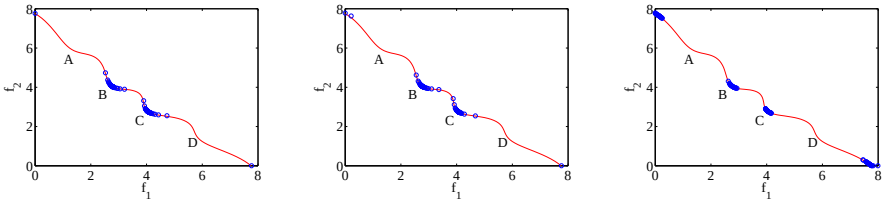


(a) Distance Elimination (b) Trade-off Elimination (c) TKR-NSGA-II [2]

Fig. 10. Obtained solutions on DEB2DK.

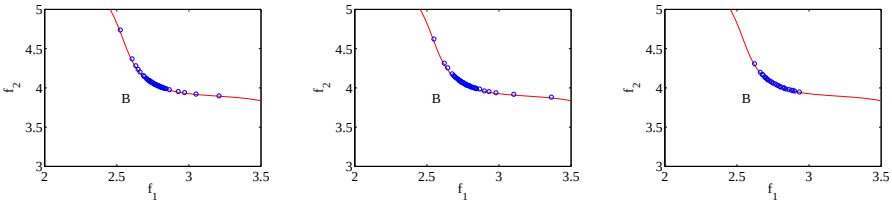
4.2 Real World Problem

As a real world problem, we use a welded beam design problem [6]. This problem has two objectives and one knee point. One of the objectives is minimizing the



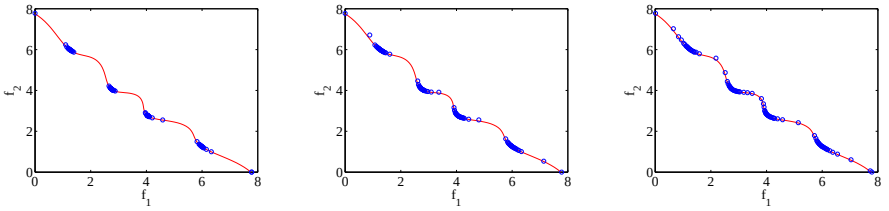
(a) Distance Elimination (b) Trade-off Elimination (c) TKR-NSGA-II [2]

Fig. 11. Change of distribution by T and N_k. $T = 0.8$ in (a) and (b). $N_k = 2$ in (c).



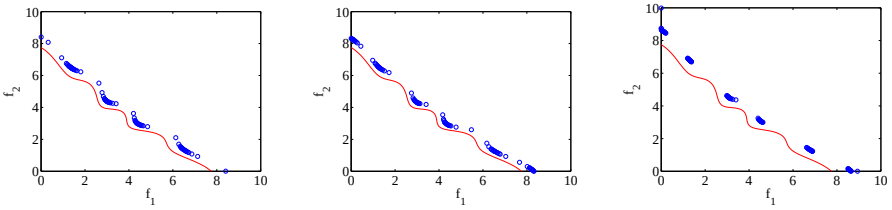
(a) Distance Elimination (b) Trade-off Elimination (c) TKR-NSGA-II [2]

Fig. 12. Solutions around the knee point B.



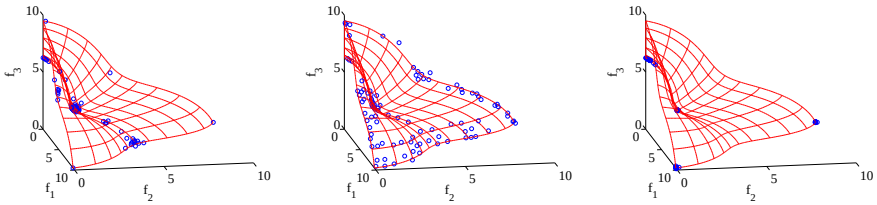
(a) $\mathbf{s} = (0.3, ..., 0.3)$ (b) $\mathbf{s} = (0.5, ..., 0.5)$ (c) $\mathbf{s} = (0.7, ..., 0.7)$

Fig. 13. Effect of $\mathbf{s}$ parameter. Distance Elimination is used in all cases.



(a) Distance Elimination (b) Trade-off Elimination (c) TKR-NSGA-II [2]

Fig. 14. Obtained solutions on DEB2DK-1.

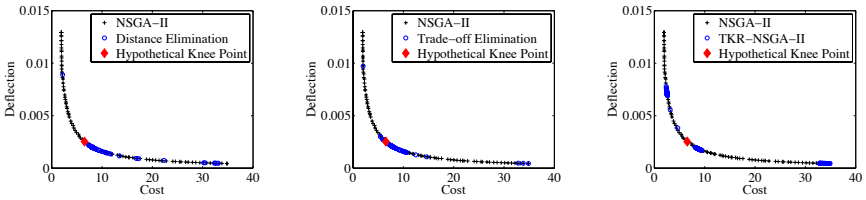


(a) Distance Elimination (b) Trade-off Elimination (c) TKR-NSGA-II [2]

Fig. 15. Obtained solutions on DEB3DK.

cost of fabrication and the other is minimizing the end of deflection of the welded beam. The stopping criterion is specified as 500 generations. We specify the threshold T as 0.5 in our proposed method for a strong convergence towards the knee point. Since there is no easy way to calculate the Pareto front of this problem, we compare the result obtained by NSGA-II in which the stopping criterion is specified as 10000 generations. Among the solutions obtained by NSGA-II, we identify the solution having the largest distance and the solution having the largest trade-off metric, respectively. (The same solution is identified in both cases.) We call the solution a hypothetical knee point.

We show the results in Fig. 16. From Fig. 16, we can see that solutions are distributed on the right side of the hypothetical knee point in all cases. This is because the left-hand extreme solution does not converge well towards the true extreme point. As a result, the place of a knee-like point is far from the place of the hypothetical knee point. It seems that the success of finding the knee point depends on finding true extreme points.



(a) Distance Elimination (b) Trade-off Elimination (c) TKR-NSGA-II [2]

Fig. 16. Obtained solutions on welded beam design problem. $T = 0.5$ in (a) and (b).

5 Conclusion

In this paper, we proposed an EMO algorithm that obtains solutions around knee points. An advantage of the algorithm is that the DM does not have to specify

the number of knee points. Moreover, thanks to P-NSGA-II, the distribution of obtained solutions is similar to a Gaussian function. In experimental results, we confirmed that our proposed method could obtain solutions around knee points on two- or three-objective benchmark problems. Throughout the experiment, we used both Distance-based Elimination and Trade-off-based Elimination. While they showed similar performances in almost all problems, Distance-based Elimination showed a better performance in the DEB3DK problem. In our proposed method, the DM can control a distribution of solutions by specifying the threshold T. In the case of a high threshold T, we only obtain solutions around knee points with a large distance or trade-off metric. Besides, we confirmed that a spread of solutions is controllable by the spread vector. In the case of a two-objective real world problem, solutions are distributed on the right side of the hypothetical knee point, because the extreme solution does not converge well.

As a future work, it would be interesting to try self-adjusting of the parameter ξ in order to decrease the burden of parameter setting. In this paper we used only two- or three-objective problems. Therefore, it would be also interesting to apply our proposed method to many-objective problems.

References

1. Bechikh, S., Said, L.B., Ghedira, K.: Searching for knee regions in multi-objecitve optimization using mobile reference points. In: Proc. of the 2010 ACM Symposium on Applied Computing. pp. 1118–1125. ACM (2010)
2. Bechikh, S., Said, L.B., Ghedira, K.: Searching for knee regions of the pareto front using mobile reference points. Soft Computing 15(9), 1807–1823 (2011)
3. Branke, J., Deb, K., Dierolf, H., Osswald, M.: Finding knees in multi-objective optimization. In: Parallel Problem Solving from Nature - PPSN VIII. Lecture Notes in Computer Science, vol. 3242, pp. 722–731. Springer (2004)
4. Das, I.: On characterizing the "knee" of the pareto curve based on normal-boundary intersection. Structural optimization 18(2-3), 107–115 (1999)
5. Deb, K., Gupta, S.: Understanding knee points in bicriteria problems and their implications as preferred solution principles. Engineering Optimization 43(11), 1175–1204 (2011)
6. Deb, K., Sundar, J., Udaya Bhaskara Rao, N., Chaudhuri, S.: Reference point based multi-objective optimization using evolutionary algorithms. International Journal of Conputational Intelligence Research 2(3), 273–286 (2006)
7. Mattson, C.A., Mullur, A.A., Messac, A.: Smart pareto filter: Obtaining a minimal representation of multiobjective design space. Engineering Optimization 36(6), 721–740 (2004)
8. Narukawa, K., Tanigaki, Y., Ishibuchi, H.: Evolutionary many-objective optimization using preference on hyperplane. In: Proc. of 2014 Conference Companion on Genetic and Evolutionary Computation Conference. pp. 91–92. ACM (2014)
9. Rachmawati, L., Srinivasan, D.: Multiobjective evolutionary algorithm with controllable focus on the knees of the pareto front. IEEE Trans. on Evolutionary Computation 13(4), 810–824 (2009)