NAEMO: Neighborhood-sensitive Archived Evolutionary Many-objective Optimization Algorithm

Raunak Sengupta[a,*], Monalisa Pal[b,*], Sriparna Saha[c], Sanghamitra Bandyopadhyay[b]

[a] *Department of Electrical Engineering, Indian Institute of Technology Patna, India*
[b] *Machine Intelligence Unit, Indian Statistical Institute, Kolkata, India*
[c] *Department of Computer Science and Engineering, Indian Institute of Technology Patna, India*

Abstract

One of the prominent strategies to address many-objective optimization problems involves using the reference direction based algorithms. However, literature severely lacks formal mathematical analysis to establish the reason behind superior performance of such methods. In this work, the neighborhood property of the many-objective optimization problems is recognized and is used to propose the neighborhood-sensitive archived evolutionary many-objective optimization (NAEMO) algorithm. In NAEMO, mating occurs within a local neighborhood and every reference direction continues to retain at least one associated candidate solution. Such preservation of candidate solutions leads to a monotonic improvement in diversity which has been theoretically and experimentally studied. Moreover, to combine the advantages of various mutation strategies, probabilistic mutation switching concept is introduced and to keep the archive size under control, periodic filtering modules are integrated with the NAEMO framework. Experimental results reveal that, in terms of inverted generational distance, hypervolume values and purity metric, NAEMO outperforms several state-of-the-art algorithms viz. NSGA-III, MOEA/D, θ-DEA, MOEA/DD, GrEA, HypE, MOPSO and dMOPSO on DTLZ1-4 test problems

*Corresponding author
Email addresses: raunaksengupta@gmail.com (Raunak Sengupta), monalisap90@gmail.com (Monalisa Pal), sriparna.saha@gmail.com (Sriparna Saha), sanghami@isical.ac.in (Sanghamitra Bandyopadhyay)

for up to 15 objectives. Further experiments show that NAEMO is competitive to M2M-based algorithms where the difficult regions of IMB problems have also been explored. These experiments make NAEMO a robust algorithm, which is additionally supported by theoretical foundations. The source code of NAEMO is available at `http://worksupplements.droppages.com/naemo`.

Key words: Evolutionary algorithms; Many-objective optimization; Pareto-optimality; Hyper-parameter adaptation; Reference vectors; Mutation switching.

List of symbols

Symbols	Meaning
M	number of objectives (dimension of objective vector)
d	number of decision variables (dimension of candidate solution)
$F(X)$	M-dimensional objective vector
X	d-dimensional candidate vector
m, m_1, m_2, m_3	slopes of different reference lines
p	parameter to define curvature of Pareto-optimal surface
d_1	magnitude of the projection of an objective vector along a given reference line
d_2	perpendicular distance from a point in the objective space to a given reference line
g	value of Penalty-based Boundary Intersection (PBI) function
θ	penalty factor of PBI function
$\mathcal{W}$	set of all reference lines
n	number of reference lines
$arch$	global archive of candidate solutions
sub_arch_i	i^{th} sub-archive i.e., subset of candidate solutions associated to the i^{th} reference line
L_{hard}	lower limit of global archive ($arch$) size ($= n$)
L_{soft}	upper limit of global archive ($arch$) size ($=$ constant $\times L_{hard}$)

List of symbols (contd.)

Symbols	Meaning
NP	population (or archive) size at a certain generation
itr	a generation, i.e., all iterations over n reference lines
tot_itr	maximum number of generations
nbr_i	set of non-empty reference lines, neighboring the i^{th} reference line
k	size of nbr_i
$parent$	a parent solution for mutation
$child$	a new candidate solution generated after mutation
β_{q_i}, η_c	parameters for Simulated Binary Crossover (SBX) based mutation
F, CR	parameters for Differential Evolution (DE) based mutation
δ_i, η_m	parameters for polynomial mutation
mut_prob	probability of performing SBX based mutation during mutation switching
$flag1, flag2$	flags to determine whether polynomial mutation follows SBX based mutation and DE based mutation, respectively
$\mu_{\eta_c}, \mu_F, \mu_{CR}$	adaptive mean of Gaussian distributions from which η_c, F and CR are sampled, respectively
S_{η_c}, S_F, S_{CR}	sets of all successful values of η_c, F and CR, respectively, over a generation (itr)
S_i^{itr}	number of members associated to the i^{th} reference line at generation (itr)
D_metric	uniformity of the spread of population in the objective space
S_{ideal}	ideal spread of population along reference lines
s	a parameter deciding number of decision variables for DTLZ problems
Z_{eff}	set of targeted points on the true Pareto-surface
$\mathcal{A}, \mathcal{A}_i, \mathcal{A}^\star$	final archive (Pareto-front) in the objective space, Pareto-front of the i^{th} approach, unified Pareto-front
d_E	Euclidean distance
r	M-dimensional reference point for Hypervolume indicator
z_{nad}	nadir point
$\mathcal{K}$	number of algorithms compared for Purity metric

List of symbols (contd.)

Symbols	Meaning
$\mathcal{Q}_i$	set of points common to $\mathcal{A}_i$ and $\mathcal{A}^\star$
$\mathcal{P}_i$	value of purity metric for the i^{th} algorithm among $\mathcal{K}$ algorithms

1. Introduction

Many-Objective Optimization algorithms (MaOO) are used in almost all application domains and hence, developments in this area have always been open to the researchers. Evolutionary algorithms are popular as these are capable of handling a population of solutions rather than a single solution in every iteration and also of finding approximate optimal solutions even for hard problems [1]. Many-objective evolutionary algorithms (MaOEAs) are used to address optimization problems with box constraints as defined in Eq. (1), where d-dimensional candidate vectors ($X = [x_1, \cdots, x_d]$) get mapped to M-dimensional objective vectors ($F(X)$). Formally, when $M \geq 4$, the particular sub-class is called many-objective optimization problem [1–4]. Applications of MaOO algorithms are found in control systems[5, 6], brain computer interfacing [7], in bioactive compound extraction [8], in box-pushing problem of robotics [9], in structural optimization of shed truss [10] and many more areas.

$$\begin{gathered} \text{Minimize } F(X) = [f_1(X), \cdots, f_M(X)] \\ \text{where, } X \in \mathbb{R}^d,\ F(X) : \mathbb{R}^d \to \mathbb{R}^M \text{ and } x_j^L \leq x_j \leq x_j^U, \forall j = 1, 2, \cdots, d \end{gathered} \tag{1}$$

Among other approaches, reference line based algorithms [11] such as non-dominated sorting genetic algorithm III (NSGA-III) [12], θ-dominance based evolutionary algorithm (θ-DEA) [13], decomposition based multi-objective evolutionary algorithm (MOEA/D) [14], dominance and decomposition based multi-objective evolutionary algorithm (MOEA/DD) [15] and several other variants and extensions of these algorithms have been developed and shown to perform well for problems with number of objectives as high as 15. MOEA/D introduced

the concept of using reference lines effectively and presented extremely promising results [1, 3]. Due to the nature of the algorithm which basically decomposes a multi-objective optimization (MOO) into several single-objective optimization problems, it has lent itself to several modified and extended algorithms as developed in [15, 16]. NSGA-III combined the concept of non-dominated sorting with a diversity maintenance operator based on reference lines unlike in NSGA-II [17] which uses a crowding distance operator. The θ-DEA algorithm introduced the concept of θ-dominated sorting which combines the penalty based boundary intersection (PBI) function value with non-dominated sorting. MOEA/DD has effectively combined the concepts of dominance as well as decomposition and presented the state-of-the-art results. The concept of decomposition in MOEA/D has also been combined with other meta-heuristics such as PSO yielding dMOPSO [18].

Recently, a new set of test problems referred to as Imbalanced problems [19] have been introduced. These problems have 'difficult' regions which create difficulty for the general algorithms to obtain the complete Pareto-front. Following this, the M2M based algorithms have been introduced [19],[20] which decomposes a multi-objective problem into several sub multi-objective optimization problems. The paper theoretically proves that the M2M strategy would yield better results.

Theoretical analyses and results are extremely necessary for understanding optimization problems and algorithms. The work in [21] shows that theoretical results pertaining to single objective optimization do not carry over to the multi-objective case. It finally develops an algorithm based on the theory which converges with probability 1 for a certain test function. The work in [22] performs theoretical analyses on the convergence of multi-objective evolutionary algorithms. The study in [23] presents theoretical analyses of decomposition based multi-objective optimization algorithms. However, much theoretical work on MaOO is still not present in the literature. The working and reasoning behind the performance of the proposed algorithms are also usually qualitative. Formal theoretical analysis will aid in finding the weaknesses of algorithms as well as

making improvements with a concrete theoretical basis. This paper has been written as an effort towards filling this gap. The motivations and contributions of this paper are enlisted below:

1. This paper presents the neighborhood property which is followed by a many-objective optimization algorithm along with its theoretical and experimental studies.
2. The Penalty-based Boundary Intersection (PBI) function [14] has gained popularity and is used very widely by MaOO algorithms, especially in ones which are an extension of MOEA/D. A theoretical analysis has been presented on the PBI function and how the shape of the actual Pareto-front affects the final solution if the PBI function is used.
3. Based on these theoretical concepts, we propose a novel many-objective optimization algorithm viz., Neighborhood-sensitive Archived Evolutionary Many-objective Optimization Algorithm (NAEMO) which has been shown to significantly outperform other state of the art algorithms in majority of the cases on the DTLZ test suite.
4. NAEMO introduces convergence-based filtering and diversity-based filtering schemes, followed by a theoretical analysis of these two operations.
5. Since the proposed algorithm guarantees diversity preservation, as proven in the later sections, this algorithm has also been evaluated on the imbalanced test problems and shown to have competitive performance with the previous state-of-the-art, multiobjective-to-multiobjective (M2M)-based algorithms [19] on the IMB test suite.
6. NAEMO shows a way of using both PBI function and Pareto-dominance simultaneously. As the proposed algorithm is also very modular, it lends itself to easy extensions and modifications.
7. Effort has been made towards making a better candidate vector generation scheme by introducing the probabilistic mutation switching concept.

In order to prove the effectiveness of NAEMO, the DTLZ [24] and IMB [19] test suites have been considered. The IMB test suite has been considered owing

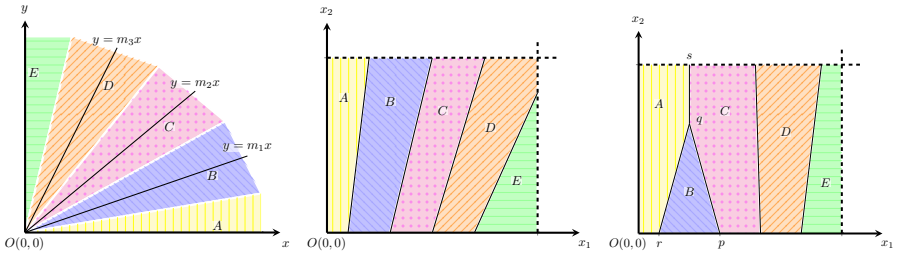
to the fact that NAEMO guarantees diversity preservation much like the M2M [19] algorithms. NAEMO has been compared to MOEA/DD [15], MOEA/D [14], θ-DEA [13], NSGA-III [12], Hypervolume estimator based evolutionary algorithm (HypE) [25] and grid based evolutionary algorithm (GrEA) [26] on the DTLZ test suites. For the IMB test suite, NAEMO has been compared to the previous state of the art in this test suite - M2M algorithms. Experimental results clearly show that NAEMO outperforms other state of the art algorithms in the DTLZ test suite and by a large margin. It is also interesting to see that NAEMO is successful in obtaining a decent Pareto-front for the imbalanced problems in spite of the difficult regions and even outperforms the M2M algorithms in some of the cases.

The remainder of the paper is structured as follows. Section 2 states the neighborhood theorem and provides a proof for it along with visualization. Section 3 presents a detailed theoretical analysis of the effect of the shape of the Pareto-front on the performance of the PBI function. Section 4 presents the new algorithm, NAEMO along with preliminary theoretical analyses. This is followed by Section 5 which presents comparison of NAEMO with other state of the art algorithms on DTLZ and IMB test suites. The paper finally ends with the conclusion section.

2. Theoretical outline of the neighborhood property

A notion of the spatial relationship between objective space and decision space, which is created by partitioning the objective space using reference vector based association of evolutionary candidate solutions, is conveyed by the following theorem based on which a new Many-Objective Optimization algorithm is proposed.

Theorem 1 (Neighborhood property)**.** *The regions corresponding to each reference line in the objective space which share a common boundary also share a common boundary in the decision space and regions which do not share a common boundary in the objective space do not share a common boundary in the*



(a) (b) (c)

Figure 1: (a) Reference lines dividing the objective space in to 5 regions (A, B, C, D, E) (b) When adjacent regions of the objective space are also adjacent in the decision space (c) When non-adjacent regions of the objective space (A and C) are adjacent in the decision space

decision space either.

Proof. The reference lines are present in the objective space as shown in Fig. 1a and their corresponding regions are marked as A, B, C, D and E. Let us first assume that regions which do not have a common boundary in the objective space can have a common boundary in the decision space. In our case, let this be true for the regions A and C. Then, a possible decision space figure might be as shown in Fig. 1c.

From this figure, it can be observed that the line sq is common to both A and C in the decision space. Hence, when transferred to the objective space, this line would correspond to a curve common to both A and C in the objective space. However, such a line is never present as can be seen from Fig. 1a. The region B always comes between A and C.

Therefore, our initial assumption that A and C can have a common region in the decision space is wrong. This proves our theorem.

The statement of the theorem can be rephrased by saying that a region corresponding to any reference line has exactly the same immediate neighbors in both the decision (Fig. 1b) and objective space (Fig. 1a). This property of multi-objective optimization problems (MOPs) when considering reference lines has been denoted as the 'neighborhood property', henceforth. □

3. Analysing the penalty factor of the penalty-based boundary intersection approach

Penalty-based Boundary Intersection (PBI) function [14] is used in several multi-objective optimization algorithms and has proven its efficacy. It helps in decomposing a multi-objective problem into several sub-problems and provides a measure of fitness for each of the sub-problems. The PBI function is a value that is calculated for a point and for a given reference line as shown in Eq. 2.

$$g = d_1 + \theta d_2 \tag{2}$$

where, d_1 is the magnitude of the projection of the given point on the given reference line in the objective space, d_2 is the perpendicular distance from the given point to the given reference line and θ is a penalty factor.

Due to its popularity in recent days, it is necessary to perform a theoretical analysis of the PBI function.

This analysis is conducted for the simple case of linear Pareto-front (PF), followed by a generic case of convex or concave PF.

3.1. Linear Pareto-front

For a problem with linear Pareto-front, the minimal objective values are attained using PBI approach with $\theta > 1$. This claim is further analyzed theoretically in the following theorem.

Theorem 2. *The optimal point (objective values) for all reference lines for a multi-objective optimization problem with a linear Pareto-front is the point of intersection of the reference line with the Pareto-front if the penalty factor has a value higher than 1, i.e.* $\theta > 1$.

Proof. We have assumed a multi-objective optimization problem (MOP) whose Pareto-front ($PF : F_1F_2$) is linear (given by Eq. (3)) and the scales for all the objective functions are same and normalized.

$$F_1F_2 : x + y = 1 \tag{3}$$

In Fig. 2a, we have labeled the ideal optimal point (point of intersection of reference line with PF) as A. Therefore, in terms of the slope of the reference line (m), the point A is given by Eq. (4).

$$A = \left(\frac{1}{m+1}, \frac{m}{m+1}\right) \tag{4}$$

Now, let us consider another random point B. For the point B, the parameters $(d_{1,B}, d_{2,B})$ of PBI-based decomposition is given by Eq. (5).

$$\begin{aligned} d_{1,B} &= \overrightarrow{OB} \cdot \widehat{OA} \\ d_{2,B} &= \left\| \overrightarrow{OB} - d_{1,B}\widehat{OA} \right\| \\ \text{where, } \widehat{OA} &= \frac{1}{\sqrt{1+m^2}}\hat{x} + \frac{m}{\sqrt{1+m^2}}\hat{y} \end{aligned} \tag{5}$$

Therefore, using PBI function, g_A is the objective function corresponding to the reference line $OA : y = mx$.

$$g_A = \frac{(x+my) + \theta\,|y - mx|}{\sqrt{1+m^2}} \tag{6}$$

Now, we find the condition on θ that satisfies Eq. (7).

$$\min\left(g_A\right) = \left(\frac{1}{m+1}, \frac{m}{m+1}\right) \tag{7}$$

First, we consider a point P which is not on the Pareto-front (F_1F_2). From Fig. 2a, we see that we can always draw a line parallel to $y = mx$ and passing through P. Say this line intersects the Pareto-front (F_1F_2) at Q. Such a point has the same value of d_2 as P since the lines are parallel i.e. $d_{2,P} = d_{2,Q}$. But the d_1 of Q is better than d_1 of P i.e. $d_{1,Q} < d_{1,P}$. Hence, g_Q $(= d_{1,Q} + \theta d_{2,Q})$ is less than g_P $(= d_{1,P} + \theta d_{2,P})$.

Hence, for every point P not on the optimal surface, there is always a point Q which is better than P when using the PBI function (Eq. (2)).

Next, we consider two points M and N at an ϵ distance from A and lying on the F_1F_2 as shown in Fig. 2b i.e. $AM = AN = \epsilon$.

Let N correspond to values $d_{1,N}$ and $d_{2,N}$ and hence, $d_{1,N} = OR$ and $d_{2,N} = RN$. We also draw AL perpendicular to $OA : y = mx$. We locate a point on

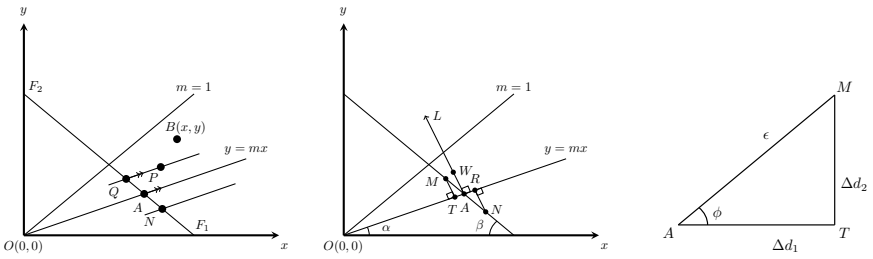


Figure 2: Given a reference line OA, (a) for every non-optimal point P, there is always an optimal point Q, (b) for every point N on the optimal surface, there is a better point W along AL such that $AW = RN = d_{2,N}$, (c) Close up of the triangle AMT to demonstrate $g_A < g_M, \forall AM = \epsilon$

AL at a distance $d_{2,N}$ from A. Let the point be W. Such a point will have $d_{2,W} = d_{2,N} = RN$ and $d_{1,W} = OA$. We see that $d_{1,N} = d_{1,W} + AR$ with $AR > 0$ and $AR \to 0$ as $\epsilon \to 0$. Thus, for every point N, there exists a better point W on AL.

Now, as we move from A to M, from the associated parameters of PBI function (Eq. (8)), we observe that the value of d_1 decreases but the value of d_2 increases.

$$\begin{aligned} g_A &= d_{1,A} + \theta d_{2,A} \\ g_M &= d_{1,M} + \theta d_{2,M} \end{aligned} \tag{8}$$

For A to be the optimal point after optimizing g_A, we must have Eq. (9)

satisfied as shown in Fig. 2b and 2c.

$$\begin{aligned} & g_M > g_A, \forall \epsilon \\ \implies & d_{1,M} + \theta d_{2,M} > d_{1,A} + \theta d_{2,A} \\ \implies & \theta > \frac{\Delta d_1}{\Delta d_2} = \frac{AT}{MT} \\ \implies & \theta > \frac{1}{\tan\phi} = \frac{1}{\tan(\alpha+\beta)} \\ \implies & \theta > \frac{1-m}{1+m} \\ \implies & \theta > 1,\ 0 \leq m \leq 1 \end{aligned} \tag{9}$$

We take $0 \leq m \leq 1$ since everything is symmetric about the reference line with $m = 1$. That is, we consider only the lower half. Therefore, $\theta > 1$ ensures that A is optimal point for all values of m in case of a linear Pareto-front. □

3.2. A more general Pareto-front

In the previous section we had assumed a linear Pareto-front and derived results for θ. However, the shape of the final Pareto-front can be concave or convex with different degrees of curvature as well. Assuming a symmetric Pareto-front about the reference line with $m = 1$, we can approximate a Pareto-front using Eq. (10).

$$x^p + y^p = 1 \tag{10}$$

From Fig. 3, we can observe that $p > 1$ corresponds to a concave shape and $p < 1$ corresponds to a convex shape. A very high or very low value of p represents a high curvature. Let us first find the ideal optimal point A in terms of m and p. The optimal point A is the intersection of the Pareto-front $x^p + y^p = 1$ and the reference line $y = mx$. Therefore, the coordinates of A can be expressed as in Eq. (11).

$$A = \left(\frac{1}{(1+m^p)^{\frac{1}{p}}}, \frac{m}{(1+m^p)^{\frac{1}{p}}} \right) \tag{11}$$

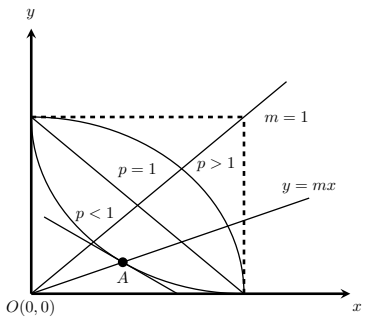


Figure 3: 2-dimensional objective space showing convex, linear and concave pareto fronts

A tangent at point A can be written as Eq. (12)

$$\begin{aligned} \frac{dy}{dx} &= -\frac{x^{p-1}}{y^{p-1}} \\ \implies \frac{dy}{dx} &= -\frac{1}{m^{p-1}} \end{aligned} \tag{12}$$

For A to be the optimal point, by observing a small region near A we again need the conditions in Eq. (13) to be satisfied.

$$\begin{aligned} \theta &> \frac{1}{\tan\phi} = \frac{1}{\tan\left(\alpha+\beta\right)} \\ \implies \theta &> \frac{m^{p-1}-1}{m^{p+1}+1} \end{aligned} \tag{13}$$

We observe that the condition $\theta > 1$ satisfies the condition for the optimality of A for all values of m and $n \geq 1$.

However, for a MOP with convex Pareto-front ($n < 1$), the condition may not be satisfied for all values of m. If we use the standard value of $\theta = 5$ and assume a convex PF which can be approximated with $p = 0.5$, the condition is satisfied only till $m = 0.03$ that is till the reference lines make 1.71 degrees with the closest axes. For reference lines which make smaller angles with the axes, the optimal point cannot be obtained. Higher the value of θ, lesser the value of the limiting angle between the reference line and axes. Thus, a higher value of θ increases the extent of the obtained PF for convex MOPs by allowing more reference lines.

It can thus be concluded that the PBI function may not always be able to give us the complete Pareto-front for MOPs. It is important to find out the effect of the value of θ on the speed of the algorithm. It might be possible that having a very high value of θ for obtaining more points might in turn reduce the speed of the algorithm, thus creating a trade-off.

4. Neighborhood-sensitive Archived Evolutionary Many-objective Optimization (NAEMO): the proposed approach

In this section we discuss in detail different steps of the proposed Neighborhood-sensitive Archived Evolutionary Many-objective Optimization (NAEMO) and the underlying features of these mechanisms.

The key concepts used in NAEMO are as follows :

1. *Using the neighborhood property:* NAEMO maintains a very organized archive to store the population. The global archive is divided into sub-archives corresponding to each reference line. Each sub-archive stores population members associated with the corresponding reference line. This helps in performing certain reference line specific operations as well as helps in easily analyzing the behavior corresponding to each reference direction.
2. *Periodic filtering of population:* In NAEMO, filtering operations are performed on the population so as to enhance either the convergence or the diversity. When the total size of the archive exceeds a specific value, diversity-based filtering is performed on the archive which aims at removing points with large PBI values from relatively crowded reference lines. The convergence-based filtering operation is performed when a newly selected candidate vector dominates other points in the archive. This operation removes such points which are dominated but also makes sure that the diversity is not hampered.
3. *Monotonic improvement in diversity:* NAEMO makes sure that once a reference line obtains an associated point, it is never lost. This feature, along

with the filtering operations, ensure that the diversity never deteriorates as proven in Section 4.8.

4. *Use of PBI function along with dominance:* Diversity-based filtering operation along with maintaining diversity, removes points which have the highest PBI value. This increases the selection pressure further in addition to the domination based selection and pushes points towards the optimal points as proven in Section 3.
5. *Improved mutation strategy:* Most of the literature on multi-/many-objective evolutionary algorithms concentrate more on the selection strategy than the other aspects of the algorithm. In this paper, NAEMO uses an improved mutation strategy and also attempts to adaptively select hyper-parameter values. This paper introduces the probabilistic mutation switching concept where different mutation operations are assigned different probability values. The frequency with which a mutation operator is used depends on the probability values.

4.1. Basic steps of NAEMO

The algorithm starts with a randomly initialized global archive of size equal to soft limit (L_{soft}). This archive consists of sub-archives, each of which stores the population associated to one of the n reference lines. The sub-archives are formed by performing the association operation (as explained in 4.3) on each of the randomly initialized points. The i^{th} sub-archive (sub_arch_i) belongs to the global archive ($arch$) as shown in Eq. (14).

$$arch = \{sub\_arch_1, sub\_arch_2, ..., sub\_arch_n\} \tag{14}$$

A single generation (itr) consists of iteration through all of the n reference lines. Let the set of all reference lines be denoted by $\mathcal{W}$, i.e., $|\mathcal{W}| = n$. We select a random point, associated to sub-archive (sub_arch_i), as the parent point ($parent$). However, if the sub-archive is empty, another random reference line is selected, from k non-empty reference lines closest to the i^{th} reference line. Let

the set of such neighboring reference lines for the reference line i be denoted by nbr_i.

The intuition behind this step is that mutation of points from neighboring regions of the empty reference line have a higher probability of generating a new point associated to the empty reference line. After obtaining the parent vector, we generate a candidate solution by applying the mutation operator (presented in 4.4) on the parent vector. However, following the neighborhood property, the mutation operation is constrained only in the k closest non-empty neighborhood i.e. the other parent vectors belong only to sub_arch_j, such that $j \in nbr_i$.

The new candidate solution (*child*) is selected for addition to the archive only if *parent* does not dominate *child*. If selected, association operation is performed on the *child* to select the sub-archive to which *child* will be added. After adding *child* to the proper sub-archive, if there exist points in the archive which are dominated by *child*, the convergence-based filtering operation is performed. If the total size of the archive exceeds a predefined value L_{soft}, diversity-based filtering is performed to reduce the number of points equal to a hard limit denoted by L_{hard}.

4.2. Initialization

The algorithm starts with the construction of reference points on a unit hyperplane of dimension same as the number of objective functions. The points can be chosen according to the requirements of the user. Reference points may be placed with a higher density in the region of high priority decided by the user. In the absence of any preference, one might place the points in a structured manner as mentioned by Das and Dennis [27]. The algorithm would attempt to retain points nearest to the lines formed by joining these reference points to the origin.

The algorithm also requires to initialize an archive ($arch$). Number of points in the archive, equal to L_{soft}, are randomly initialized. These set of points are then run through the association operation to create the structured archive which consists of sub-archives as mentioned in Section 4.1.

Algorithm 1 Framework of NAEMO

Input: tot_itr: maximum number of generations; n: number of reference lines; L_{hard} and L_{soft}: hard and soft limits on archive size; $flag1$, $flag2$, mut_prob and η_m: parameters for probabilistic mutation switching (for Algorithm 2)

Output: $\mathcal{A}$: final archive

Obtain $\mathcal{W}$ using the approach of Das and Dennis [27]
Randomly initialize archive $arch$ of size L_{soft}
Get sub_arch_i from Eq. (15) $\forall x \in arch$ and $\forall w_i \in \mathcal{W}$
for $itr = 1$ to tot_itr **do**
    $S_{\eta_c} = \phi$
    $S_F = \phi$
    $S_{CR} = \phi$
    for $j = 1$ to n **do**
        $ind = j$
        if $sub\_arch_{ind} = \phi$ **then**
            $I_{rand} \leftarrow random\,(1, |nbr_j|)$
            $ind = I_{rand}$
        end if
        $par\_ind \leftarrow random\,(1, |sub\_arch_{ind}|)$
        $parent \leftarrow sub\_arch_{ind}[par\_ind]$
        $\eta_c = Gaussian(\mu_{\eta_c}, 5)$
        $F = Gaussian(\mu_F, 0.1)$
        $CR = Gaussian(\mu_{CR}, 0.1)$
        $child =$ Mutate $parent$ using Algorithm 2
        if $parent$ does not dominate $child$ **then**
            Get $line$ where $child$ associates (Eq. (15))
            $sub\_arch_{line} = sub\_arch_{line} \cup child$
            Convergence-based filtering (Algorithm 3)
            if $|arch| > L_{soft}$ **then**

Diversity-based filtering (Algorithm 4)

end if

$S_{\eta_c} = S_{\eta_c} \cup \eta_c$

$S_F = S_F \cup F$

$S_{CR} = S_{CR} \cup CR$

end if

end for

$\mu_{\eta_c} = mean(S_{\eta_c})$

$\mu_F = mean(S_F)$

$\mu_{CR} = mean(S_{CR})$

end for

$\mathcal{A} = arch$

4.3. Association operation

The association operation associates a point with the nearest reference line. This operation helps in determining the sub-archive to which a point should belong. The formula for finding the distance (d_2) between a point vector (x) and the i^{th} reference line vector $(w_i \in \mathcal{W})$ is given by Eq. (15).

$$d_2(x, w_i) = \left\| x - \frac{x \cdot w_i}{\|w_i\|^2} \cdot w_i \right\|$$
$$x \in sub\_arch_i, \text{ when } i = \arg\min_{w_i \in \mathcal{W}} d_2(x, w_i) \tag{15}$$

For a given point, we find the value of d_2 for all reference lines and declare the reference line with the minimum value of d_2 as the associated reference line.

4.4. Mutation Strategy

A new mutation strategy referred to as the probabilistic mutation switching is used in this paper. Probabilistic mutation switching involves switching between two or more mutation strategies according to a probability assigned for each of the mutation strategies. This kind of switching between mutation

techniques often helps in combining the pros of individual mutation techniques involved. NAEMO uses a combination of Simulated Binary Crossover (SBX) based mutation [28] and a Differential Evolution (DE) based mutation [29]. The reproduction techniques used are mentioned as follows:

1. *SBX mutation:* In SBX scheme, for a parent solution ($x_i^{1,itr}$), we first choose another random solution from the archive as the second parent ($x_i^{2,itr}$). This is followed by usual SBX crossover [28] (Eq. (16)) between the two parent solutions which produces two candidate solutions ($x_i^{1,itr+1}$ and $x_i^{2,itr+1}$). The parameter β_{q_i} is sampled from the probability distribution in Eq. (17) where η_c is the SBX crossover parameter. Out of the two, only the first one ($x_i^{1,itr+1}$) is chosen as the i^{th} candidate solution and is further processed.

$$\begin{aligned} x_i^{1,itr+1} &= 0.5\left[(1+\beta_{q_i})\, x_i^{1,itr} + (1-\beta_{q_i})\, x_i^{2,itr}\right] \\ x_i^{2,itr+1} &= 0.5\left[(1-\beta_{q_i})\, x_i^{1,itr} + (1+\beta_{q_i})\, x_i^{2,itr}\right] \end{aligned} \tag{16}$$

$$P\left(\beta_i\right) = \begin{cases} 0.5\left(\eta_c+1\right)\beta_i^{\eta_c}, & \text{if } \beta_i \leq 1 \\ 0.5\left(\eta_c+1\right)\frac{1}{\beta_i^{\eta_c+2}}, & \text{otherwise} \end{cases} \tag{17}$$

2. *DE based mutation:* For every i^{th} solution, DE based mutation [30–32] involves generating a new point (x_i^{itr+1}) from three different randomly chosen points from the archive ($x_{r_1}^{itr}$, $x_{r_2}^{itr}$ and $x_{r_3}^{itr}$) using the usual Eq. (18) where the scale factor F is randomly chosen between 0 and 2. This is followed by binomial crossover.

$$x_i^{itr+1} = x_{r_1}^{itr} + F \times \left(x_{r_2}^{itr} - x_{r_3}^{itr}\right) \tag{18}$$

3. *Polynomial mutation:* For every i^{th} solution, the polynomial mutation [33] alters a d-dimensional new point (x_i^{itr+1}) within the upper and lower bounds of every j^{th} decision variable i.e. x_j^U and x_j^L respectively. It follows Eq. (19) where δ_i is the probability distribution given by Eq. (20) with η_m as the polynomial mutation parameter.

$$x_{i,j}^{itr+1} = x_{i,j}^{itr+1} + \delta_i \times \left(x_j^U - x_j^L\right), \forall j = 1, \cdots, d \tag{19}$$

$$P(\delta_i) = 0.5\,(\eta_m + 1)\,(1 - |\delta_i|)^{\eta_m} \tag{20}$$

Probabilistic mutation switching uses *mut_prob* as the probability of performing a SBX based mutation. Therefore, $(1 - mut\_prob)$ becomes the probability of performing Differential Evolution based mutation. The mutation strategy used in this paper employs two flags, $flag1$ and $flag2$, as parameters. The parameter $flag1$ determines whether the SBX based mutation is to be followed by polynomial mutation. Similarly, $flag2$ determines whether DE based mutation is to be followed by polynomial mutation. This option has been incorporated since it has been observed experimentally that in some cases not using polynomial mutation improves the convergence value. However, this also makes the algorithm prone to getting stuck at local optimal fronts. The mutation strategy can be better visualized from Algorithm 2.

Algorithm 2 Mutate

Input: *mut_prob*: mutation switching factor; $flag1$ and $flag2$: determine the use of polynomial mutation; *nbr*: set of neighboring non-empty reference lines

Output: *Child*: Newly generated point

if $rand() > mut\_prob$ **then**
  Child ← DE based mutation (Eq. (18))
  if $flag2$ is true **then**
    Child ← Polynomial mutation (Eq. (19)-(20))
  end if
else
  Child ← SBX mutation (Eq. (16)-(17))
  if $flag1$ is true **then**
    Child ← Polynomial mutation (Eq. (19)-(20))
  end if
end if

4.5. Parameter adaptation

At each generation itr , the parameter η_c of SBX based crossover is generated from a Gaussian distribution (as shown in Eq. (21)) with mean μ_{η_c} and variance 5.

$$\eta_c^{itr} = Gaussian(\mu_{\eta_c}, 5) \tag{21}$$

Let S_{η_c} denote the set of all successful values of η_c obtained at generation itr. The mean (μ_{η_c}) is initialized to a value of 30 and updated after every generation as the mean value of all the values in S_{η_c}.

Similarly, the parameters F and CR are also sampled during each iteration from Gaussian distributions as shown in Eq. (22) and (23), respectively.

$$F^{itr} = Gaussian(\mu_F, 0.1) \tag{22}$$

$$CR^{itr} = Gaussian(\mu_{CR}, 0.1) \tag{23}$$

The sampled values are then truncated to $[0, 1]$. Let S_F denote the set of all successful values of F and S_{CR} denote the set of all successful values of CR at iteration itr. The means, μ_F and μ_{CR}, are initialized to 0.5 and 0.2, respectively, and updated after every generation as the mean of S_F and S_{CR}, respectively. It has been experimentally observed that the parameter mut_prob works best for a value of 0.75 and adaptation of this parameter does not lead to any improvements. The parameters $flag1$ and $flag2$ are not adaptive in this mutation strategy.

4.6. Convergence-based filtering

This operation is performed every time a new child vector gets selected. Convergence-based filtering algorithm (Algorithm 3) looks for all the points in the archive which are dominated by the child vector and removes them. However, it does not remove a dominated point if it is the only point associated to its corresponding reference line. This ensures that no reference line is rendered

empty, thus preserving diversity. Doing this turns out to be useful for problems which have certain regions much difficult than other regions in terms of optimization.

Algorithm 3 Convergence-based filtering

Input: *arch*: unfiltered archive; sub_arch_i: i^{th} sub-archive; n: number of reference lines or number of partitions of *arch*; *child*: newly added point

Output: *arch*: filtered archive

```
for i = 1 to n do
    l ← Set of points in sub_arch_i dominated by child
    for j = 1 to |l| do
        if |sub_arch_i| > 1 then
            {sub_arch_i} ← {sub_arch_i} − {l_j}
        end if
    end for
end for
```

4.7. Diversity-based filtering

Diversity-based filtering is performed when the total population size of the archive exceeds the value L_{soft}. This is necessary since an archive with open size will require large amounts of computation. In this operation (Algorithm 4), we first find the reference line with the highest number of associated points. Then, we find the point in the associated sub-archive which has the highest PBI function value and remove it. This process continues until the total size of the archive reduces to hard limit, L_{hard}. Filtering the points using the PBI function value results into a selection pressure on the points towards the optimal point as has been proven in Section 3.

4.8. Proof of Monotonic Improvement of Diversity

In this part, we prove mathematically that NAEMO with its two filtering operations never leads to deterioration of the diversity.

Algorithm 4 Diversity-based filtering

Input: $arch$: unfiltered archive; sub_arch_i: i^{th} sub-archive; n: number of reference lines or number of partitions of $arch$; pop_arr: array of length n to store population sizes of each sub-archive; L_{hard}: minimum size of archive

Output: : $arch$: filtered archive

$l = 0$
$pop\_arr \leftarrow \left[0, \overset{n}{\cdots}, 0\right]$
for $i = 1$ to n **do**
    $sub\_arch_i \leftarrow$ Sort sub_arch_i by PBI value
    $pop\_arr[i] \leftarrow |sub\_arch_i|$
    $l = l + |sub\_arch_i|$
end for
while $l > L_{hard}$ **do**
    $ind \leftarrow$ Index of maximum value from pop_arr
    Remove last element from sub_arch_{ind}
    $pop\_arr_{ind} = pop\_arr_{ind} - 1$
    $l = l - 1$
end while

Theorem 3. *Diversity-based filtering operation (Algorithm 4) always generates a monotonic improvement of diversity measured using D_metric (Eq.* (24)*).*

Proof. We first present *D_metric* as defined in [34].

$$D\_metric^{itr} = \frac{n}{NP}\sqrt{\sum_{i=1}^{n}\left(S_i^{itr} - S_{ideal}\right)^2} \tag{24}$$

where, itr is the current generation, n is the number of reference lines, NP is population (or archive) size, S_i^{itr} is the number of members associated to the i^{th} reference line at generation itr, $S_{ideal} = \frac{NP}{n}$ and $\sum_{i=1}^{n} S_i^{itr} = NP$.

The indicator, *D_metric*, captures the uniformity of the spread of population members throughout the region covered by the reference lines. A lower value of *D_metric* would indicate a more uniform distribution and therefore, a better

diversity. The ideal value of D_metric should be 0 which occurs when all the reference lines have equal number of associated members, denoted by S_{ideal}.

In NAEMO, diversity-based filtering operation reduces the population size from L_{soft} to L_{hard}. Therefore, while performing this operation, the value of NP is constantly decreasing after each removal of point. The mechanism finds the reference line with the highest associated number of population members and removes the member with largest value of PBI function. To prove that this mechanism yields a monotonic improvement in diversity, we consider the following cases of two different reference lines $l1$ and $l2$:

1. A member was removed from some reference line $l1$. Therefore, Eq. (25) follows.

$$D\_metric_1^{itr+1} = \frac{n}{NP-1}\left(\sum_{i=1,i\neq l1,l2}^{n}\left(S_i^{itr}-\frac{NP-1}{n}\right)^2 + \left(S_{l1}^{itr}-1-\frac{NP-1}{n}\right)^2 + \left(S_{l2}^{itr}-\frac{NP-1}{n}\right)^2\right)^{0.5} \tag{25}$$

2. A member was removed from a different reference line $l2$. Therefore, Eq. (26) follows.

$$D\_metric_2^{itr+1} = \frac{n}{NP-1}\left(\sum_{i=1,i\neq l1,l2}^{n}\left(S_i^{itr}-\frac{NP-1}{n}\right)^2 + \left(S_{l2}^{itr}-1-\frac{NP-1}{n}\right)^2 + \left(S_{l1}^{itr}-\frac{NP-1}{n}\right)^2\right)^{0.5} \tag{26}$$

For maximum improvement in the diversity, the D_metric should be minimum, after removal of a point. For $D\_metric_1^{itr+1}$ to be better than $D\_metric_2^{itr+1}$,

Eq. (27) should follow:

$$\begin{aligned} & D\_metric_1^{itr+1} \leq D\_metric_2^{itr+1} \\ \implies & \left(S_{l1}^{itr} - 1 - \frac{NP-1}{n}\right)^2 + \left(S_{l2}^{itr} - \frac{NP-1}{n}\right)^2 \leq \\ & \left(S_{l2}^{itr} - 1 - \frac{NP-1}{n}\right)^2 + \left(S_{l1}^{itr} - \frac{NP-1}{n}\right)^2 \\ \implies & S_{l1}^{itr} \geq S_{l2}^{itr} \end{aligned} \tag{27}$$

Thus, for removal of a point from $l1$ to be a better decision than removal from $l2$, the sub-population size for $l1$ should be greater than that of $l2$. Since, the diversity-based filtering mechanism always finds reference lines i with highest value of S_i^{itr}, it always satisfies the inequality at the end of Eq. (27) and thus, leads to the maximum possible decrease in the value of D_metric. □

Theorem 4. *Convergence-based filtering operation (Algorithm 3) preserves diversity in the long run.*

Proof. Convergence-based filtering operation removes all the points that are dominated by the new child vector, except for those points removing which might render a reference line empty. The removal of points in this operation might lead to a decrease in D_metric since the inequality at the end of Eq. (27) might not be followed. However, our final aim is to obtain one point for each reference line. Therefore, finally $S_{ideal} = 1$ and $n = NP$, assuming $L_{hard} = n$. Therefore, for this case, D_metric (from Eq. (24)) translates to Eq. (28).

$$D\_metric^{itr} = \sqrt{\sum_{i=1}^{n} \left(S_i^{itr} - 1\right)^2} \tag{28}$$

The value of this D_metric increases (i.e. deteriorates) as more reference lines are rendered empty. However, since convergence-based filtering operation never renders a reference line empty once an associated point has been found, it can be claimed that the convergence-based filtering operator preserves diversity. □

4.9. Usage of neighborhood property

The algorithm uses the proven neighborhood property (Theorem 1) in the following two ways:

1. In case an empty reference line is encountered during any iteration, the parent vector is chosen from a neighboring non-empty reference line. This follows the intuition that mutation of a point from neighboring regions would have a higher probability of generating a point associated to the empty reference line.
2. The reproduction operation of a parent vector is constrained within the neighboring region and uses points only from its k closest neighboring reference lines. The value of k is usually between 10% to 20% of the total number of reference lines. This increases the convergence of the algorithm immensely as shown in section 5 . Let us define the Region of Improvement (ROI) of a reference line as the region in decision space where the points correspond to better solutions than the current best point. This region is obviously a sub-part of the region in decision space corresponding to the reference line. Neighboring to this ROI are the ROIs of other reference lines as proven from the neighborhood theorem 1. Performing crossover with points in the neighborhood thus has a much higher probability of producing a point in one of the ROIs than performing crossover with other random points.

4.10. Computational Complexity of NAEMO

We compute the complexity of one iteration of NAEMO considering n reference lines, M number of objectives, L_{soft} as the soft limit and L_{hard} as the hard limit. Now hard limit is chosen to be equal to the number of reference lines and the soft limit is chosen to be some constant times n. Therefore, $L_{hard} = n$ and $L_{soft} = Cn$ where C is a constant such that $C \in \mathbb{R}$ and $C > 1$.

The time taken by the algorithm depends on how frequently the if conditions in line 20 and line 24 of Algorithm 1 are satisfied apart from the values of M

and n. The frequency of the if condition in line 24 being satisfied does not affect the complexity as it contributes a constant term as shown later. The if condition in line 20 depends on how frequently the new solution dominates the parent solution and therefore also does not affect the complexity.

From Algorithm 1-line 8, we observe that there is a for loop with n iterations. Now we find the complexity for operations inside this for loop.

- In Algorithm 1 - line 21, the association operation requires $O(Mn)$.
- In Algorithm 1 - line 23, the Convergence Filtering (Algorithm 3) requires $O(ML_{soft}) = O(Mn)$
- The Diversity Filtering, in Algorithm 1 - line 24 and 25, is analysed as follows. Within the if block,
  - Maximum number of times the if condition is satisfied within n iterations is $\frac{n}{L_{soft}-L_{hard}} = \frac{1}{C-1}$. Therefore , it is constant.
  - The Diversity Filtering (Algorithm 4) requires $O(L_{soft}\log(L_{soft})) + O(n(L_{soft} - L_{hard})) = O(n\log(n)) + O(n^2) = O(n^2)$.

Therefore, total complexity over n iterations is given by $n(O(Mn)+O(n^2)) = O(Mn^2 + n^3)$.

For comparison, the computational complexities of several Many-Objective Evolutionary Algorithms (MaOEAs) are mentioned in Table 1. It can be seen that in worst case, NAEMO has intermediate time requirements. It is neither the fastest among several other competitor algorithms, nor it is the slowest one. Hence, NAEMO is designed to yield competitive performance in similar time requirements. Comparison of execution time (in seconds) has been shown in Section 5.7.4.

5. Experimental results and interpretations

In this section, we present a comparison of NAEMO with other state of the art algorithms on DTLZ1-DTLZ4 test functions from the DTLZ test suite [24]

Table 1: Worst case computational complexity for a single generation of several MaOEAs considering M as number of objectives and n as the population size (which is nearly equal to the number of reference lines).

Algorithm Name	**Computational Complexity**
NAEMO	$O(Mn^2 + n^3)$
NSGA-III [12]	$O(n^2 \log^{M-2} n + Mn^2)$
θ-DEA [13]	$O(Mn^2)$
GrEA [26]	$O(n^3)$
HypE [25]	$O(n^M + Mn \log n)$

and IMB1-IMB9 test functions from the Imbalanced problems test suite [19]. For each DTLZ test function, the number of objectives is set as $M \in \{3, 5, 8, 10, 15\}$. According to the recommendations of [24], the number of decision variables for a particular DTLZ test function is set as $d = M + s - 1$, where $s = 5$ for DTLZ1 and $s = 10$ for DTLZ2, DTLZ3 and DTLZ4. The number of decision variables (d) is 10 for all the imbalanced problems, IMB1-IMB9, as per the recommendations in [19].

This paper uses mainly two comparison metrics, IGD (Inverted Generational Distance) [35] value for comparison among reference-point based algorithms and Hypervolume (HV) value [36] for a more general comparison in case of DTLZ functions. The approach towards calculating HV is same as that in [15] and has also been elaborated later. NAEMO has been compared with MOEA/DD [15], MOEA/D [14], NSGA-III [12], θ-DEA [13], GrEA [26] and HypE [25] on the DTLZ test suite.

NAEMO has been compared to five multiobjective-to-multiobjective (M2M) based algorithms [19], viz. NSGA-II-M2M, MOEA/D-M2M, SMS-EMOA-M2M, SPEA2-M2M and GVEGA-M2M on the IMB test suite. Hypervolume (HV) in this case is calculated slightly differently than for DTLZ and is mentioned in Section 5.1.

Further analysis has also been done on mutation switching, computation time, diversity attainment, weaknesses of NAEMO as well as its performance compared to PSO based algorithms.

5.1. Comparison Metrics

Any Pareto-optimal solution, or specifically any Pareto-front, obtained from a MaOEA is assessed in terms of convergence to true Pareto-front and diversity, i.e., uniformity and spread, along the true Pareto-front. Among a large number of metrics, the most commonly used performance measures are IGD metric and HV value.

The IGD metric provides a combined information about the convergence and diversity of the obtained solutions. In reference point based algorithms, the targeted points are calculated by finding the point of intersection of the reference lines with the true Pareto surface. These targeted points, z_i, are computed and named as Z_{eff}. For any algorithm, we obtain the final archive of points in the objective space and call them the set $\mathcal{A}$. Now, we compute the IGD metric as the average Euclidean distance of points in set Z_{eff} with their nearest members of all points in set $\mathcal{A}$ as given in Eq. (29).

$$IGD\left(\mathcal{A}, Z_{eff}\right) = \frac{1}{|Z_{eff}|} \sum_{i=1}^{|Z_{eff}|} min_{j=1}^{|\mathcal{A}|} d_E\left(z_i, a_j\right) \quad \text{where } z_i \in Z_{eff} \text{ and } a_j \in \mathcal{A} \tag{29}$$

where, $d_E\left(z_i, a_j\right)$ is the Euclidean distance between z_i and a_j. A smaller value of IGD results into the conclusion that the obtained points of the archive ($\mathcal{A}$) are closer to their associated reference points and are thus better. If say a part of the obtained Pareto-front is missing, this will result into a higher value of $d_E\left(z_i, a_j\right)$ for the empty reference lines and thus will increase the value of the resultant IGD metric.

However, the IGD metric suffers from the following drawbacks:

1. For evaluating IGD by Eq. (29), defining Z_{eff} requires the knowledge of the true Pareto-front which is unavailable for practical problems.

2. The value of IGD lies in the range $[0, \infty)$. Thus, without field knowledge or unless compared with the IGD value of another Pareto-front, it becomes difficult to assert how far is the obtained Pareto-front from the true Pareto-front.
3. Algorithms based on reference lines have an unfair advantage over algorithms not based on reference lines when compared using IGD because reference line based algorithms unlike the other algorithms explicitly target the reference points at the intersection of reference lines and true Pareto-front. These reference points also constitute Z_{eff} which is used for IGD calculation. Algorithms not based on reference lines do not target any specific points on the Pareto-front and will therefore always have poorer values.

Thus, for comparing a wide range of algorithms (based on both decomposition and non-decomposition strategies) with NAEMO, the Hypervolume (HV) indicator has been used. In this paper, HV value has been used for comparing NAEMO with M2M based algorithms on the IMB test suite as well. It has been shown in [36] that the Hypervolume indicator is strictly Pareto compliant. Let $r = (r_1, r_2, \cdots, r_M)$ be a reference point in the objective space. The hypervolume is then calculated using the expression in Eq. (30).

$$HV\left(\mathcal{A}, r\right) = volume\left(\cup_{a \in \mathcal{A}} \left[a_1, r_1\right] \times \cdots \times \left[a_M, r_M\right]\right) \tag{30}$$

The HV is a measure for both convergence and diversity of a solution set simultaneously. Given a reference point r, a larger HV value means better quality.

For IMB test problems, we set r to a point little beyond the nadir point (z_{nad}) i.e. at $z_{nad} + 0.001$ as done in [37]. For DTLZ1, we set r as $\left(1.0, \overset{M}{\cdots}, 1.0\right)$ and for DTLZ2 - DTLZ4, we set r as $\left(2.0, \overset{M}{\cdots}, 2.0\right)$. The HV values obtained for the DTLZ problems are further normalized to $[0, 1]$ by dividing the total volume for the reference line, $z = \prod_{i=1}^{M} r_i$.

Although hypervolume indicator does not suffer from the drawbacks of IGD metric such as evaluation of Eq. (30) does not require the value of true Pareto-

front, yet a major concern for evaluating HV is the huge computational complexity. Nonetheless, the advantages of hypervolume indicator outweigh its drawback and has been a popular choice of performance metric in this domain.

5.2. Parameter Settings of Algorithms

The parameters for other compared algorithms are set as suggested in [13], [38], [26], [12] and [14]. The parameter settings for all the algorithms have also been categorically specified below:

1. *Parameter settings for NAEMO:* In case of NAEMO, the parameters η_c, F and CR are adaptive. The values of μ_{η_c}, μ_F and μ_{CR} are initialized to 30, 0.5 and 0.2 respectively. The value of η_m is kept constant at 20. The values of $flag1$ and $flag2$ are false in all cases except for DTLZ1 and DTLZ3 functions since they are multi-modal. The parameter $flag2$ is set as true for DTLZ1 while $flag1$ is set as true for DTLZ3. The value of θ for calculating the PBI function is set as 5.0. The neighborhood size is set as 20% of the total number of reference lines. The value of L_{hard} is equal to the number of reference lines in all cases. The value of soft limit (L_{soft}) for the DTLZ problems is mentioned in Table 2. In case of IMB test problems, the value of L_{soft} is set as 400 for 2-objective problems and 900 for 3-objective problems.
2. *Reproduction operator parameters:* The values of η_c and η_m for NSGA-III and θ-DEA are set as 30 and 20, respectively. For the other algorithms using SBX crossover and polynomial mutation, η_c and η_m are set as 20 and 20, respectively. For algorithms using SBX crossover, the crossover probability is set as 1.0 and the mutation probability is set as $1/d$ where d is the dimension of candidate solutions.
3. *Penalty parameter:* The algorithms θ-DEA [13], MOEA/D-PBI [14] and MOEA/DD [15] require a penalty parameter θ for calculating the PBI function value. For the comparisons performed, θ is set to a value of 5.
4. *Neighborhood size:* The neighborhood size for MOEA/D and MOEA/DD is set as 20.

Table 2: Population Size Settings

No. of objectives (M)	**Divisions (outer, inner) [12]**	**No. of reference lines (n)**	**Population size for NSGA-III [12]**	L_{soft} **for NAEMO**
3	12, 0	91	92	100
5	6, 0	210	212	220
8	3, 2	156	156	160
10	3, 2	275	276	280
15	2, 1	135	136	140

5. *Sampling size:* HypE approximates the hypervolume using Monte Carlo approximation. The Monte Carlo sampling size is set to 10,000 as given in [25].
6. *Grid divisions:* The grid divisions of GrEA for different test functions are according to the guidelines in [26].

5.3. Comparison on DTLZ Problems

The performance of the proposed many-objective optimization algorithm viz. NAEMO is presented in terms of IGD values in Tables 3 and 4 and in terms of HV values in Tables 5 and 6 for DTLZ problems with $M \in \{3, 5, 8, 10, 15\}$. The best, median and worst values are noted over 30 runs of NAEMO for each of the problems. For execution of NAEMO, Algorithm 1 is implemented in Python 3.4 and executed in a computer having 8GB RAM with Intel Core i7 @ 2.5GHz processor.

For establishing the efficacy of NAEMO, the IGD and HV values of other state-of-art algorithms, viz. NSGA-III, MOEA/D, θ-DEA*, MOEA/DD, GrEA and HypE are also mentioned alongside in Tables 3, 4, 5 and 6. The maximum number of generations (tot_itr) upto which the algorithms are allowed to run are mentioned in Tables 3 and 4 along with the IGD values. These values for maximum number of iterations form a standard setting as noted in [12, 15].

As can be seen from Tables 3, 4, 5 and 6, NAEMO outperforms other contemporary state-of-the-art algorithms in terms of IGD and HV values for

DTLZ1-4 problems. This indicates that NAEMO is capable of handling both multi- and many-objective, unimodal and multi-modal optimization problems and even problems with biased density of solutions.

From Tables 3 and 4, we can note that in a total of six out of 60 cases, i.e., in worst cases of DTLZ1 ($M = 3$ and $M = 8$), in the best case of DTLZ3 ($M = 10$) and in all three cases of DTLZ4 ($M = 8$), MOEA/DD performs slightly better than NAEMO. Similarly, from Tables 3 and 4, in two out of 60 cases, i.e., in median and worst cases of DTLZ1 ($M = 15$), θ-DEA* performs only slightly better than NAEMO. However, in all the remaining cases, NAEMO demonstrates improvement in IGD values, in some cases even by an order of magnitude. This large margin in improvement where the IGD metrics often reach values in the order of 10^{-5} can be attributed to the efficient use of the neighborhood property by the algorithm. The algorithm, also makes use of the PBI function value which as proven before in Section 3 creates a selection pressure on the points towards the optimal point. This selection pressure is implemented using the diversity filtering operation and adds up with the selection pressure due to the pareto dominance based selection. Tables 5 and 6, also show a similar trend in the performance of NAEMO.

The resulting final archive ($\mathcal{A}$) in the objective space, which represents the Pareto-front (PF), is visualized in Fig. 4 for DTLZ problems to further investigate the performance of NAEMO. From PF plots in Fig. 4, the following observations can be considered:

1. For 3-objective DTLZ1, PF from NAEMO is presented in Fig. 4a. As known from literature [24], DTLZ1 has a linear PF which meets the axis of objective space at 0.5 and an uniformly spread PF is obtained at the same location from NAEMO. Furthermore, no outliers are observed, hence, all local optima has been overcome successfully.
2. For 3-objective DTLZ4, PF from NAEMO is presented in Fig. 4b. As known from literature [24], DTLZ4 has a unit hyper-sphere in the first hyper-octant as the PF and an uniformly spread PF is obtained at the

same location from NAEMO. DTLZ4 also has biased density of solutions, however, the plot demonstrates that NAEMO is capable to efficiently meet such challenges. For 3-objective DTLZ2 and DTLZ3, the plots of PF, resulting from NAEMO, are identical to Fig. 4b, and hence, are not reproduced in the manuscript.

3. For higher objective DTLZ problems, the radial plot visualization method [39] is used. As seen from Fig. 4c, for 8-objective DTLZ1 problem, the PF from NAEMO lies at a radius of 0.5 in the radial plot without any outlier. For DTLZ1 problem with $M = 5, 10, 15$, the result is identical to Fig. 4c. This implies NAEMO works efficiently even for high number of objectives for problem characteristics similar to DTLZ1.
4. For 8-objective DTLZ2 problem, the PF from NAEMO lies at a radius of 1 in the radial plot (Fig. 4d) without any outlier. The result from other DTLZ3 and DTLZ4 problems with $M = 5, 8, 10, 15$, are identical Fig. 4d. These observations establishes the proficiency of NAEMO for DTLZ1-4 problems.

5.4. Comparison on IMB Problems

The efficacy of the proposed many-objective optimization algorithm viz. NAEMO in dealing with imbalanced problems is noted in terms of HV values in Table 7. The best, median and worst HV values are noted over 30 runs of NAEMO for each of the problems. For execution of NAEMO, Algorithm 1 is implemented in Python 3.4 and executed in a computer having 8GB RAM with Intel Core i7 @ 2.5GHz processor.

As the M2M-based MaOO algorithms are well-known for handling imbalanced problems [19], hence, for establishing the efficacy of NAEMO, the HV values of other state-of-art algorithms, viz. NSGA-II-M2M, MOEA/D-M2M, SMS-EMOA-M2M, SPEA2-M2M, and GVEGA-M2M are also mentioned alongside in Table 7. The maximum number of iterations (tot_itr) upto which the algorithms are allowed to run is 2000 as per the specification in [19].

Table 3: Best, median, worst IGD values for comparing MaOO approaches on M-objective multimodal (DTLZ1 and DTLZ3) problems. Best performance is highlighted in bold.

Problems(P)	M	tot_itr	**NAEMO**	**NSGA-III**	**MOEA/D**	**θ-DEA***	**MOEA/DD**	**GrEA**	**HypE**
DTLZ1	3	400	**2.725E-5**	4.880E-4	4.095E-4	3.006E-4	3.191E-4	2.759E-2	1.822E+1
			4.801E-5	1.308E-3	1.495E-3	9.511E-4	5.848E-4	3.339E-2	1.974E+1
			1.119E-3	4.880E-3	4.743E-3	2.718E-3	**6.573E-4**	1.351E-1	2.158E+1
	5	600	**3.710E-5**	5.116E-4	3.179E-4	3.612E-4	2.635E-4	7.369E-2	1.799E+1
			5.854E-5	9.799E-4	6.372E-4	4.259E-4	2.916E-4	3.363E-1	2.141E+1
			6.529E-5	1.979E-3	1.635E-3	5.797E-4	3.109E-4	4.937E-1	2.359E+1
	8	750	**4.477E-4**	2.044E-3	3.914E-3	1.869E-3	1.809E-3	1.023E-1	1.030E+1
			6.558E-4	3.979E-3	6.106E-3	2.061E-3	2.589E-3	1.195E-1	2.265E+1
			2.389E-1	8.721E-3	8.537E-3	2.337E-3	**2.996E-3**	3.849E-1	2.426E+1
	10	1000	**5.022E-4**	2.215E-3	3.872E-3	1.999E-3	1.828E-3	1.176E-1	1.427E+1
			8.536E-4	3.462E-3	5.073E-3	2.268E-3	2.225E-3	1.586E-1	1.693E+1
			1.762E-3	6.869E-3	6.130E-3	2.425E-3	2.467E-3	5.110E-1	2.034E+1
	15	1500	**1.782E-3**	2.649E-3	1.236E-2	2.884E-3	2.867E-3	8.061E-1	1.797E+1
			3.587E-3	5.063E-3	1.431E-2	**3.504E-3**	4.203E-3	2.057E+0	2.519E+1
			4.464E-3	1.123E-2	1.692E-2	**3.922E-3**	4.699E-3	6.307E+1	2.954E+1
DTLZ3	3	1000	**1.395E-4**	9.751E-4	9.773E-4	8.575E-4	5.690E-4	6.770E-2	1.653E+2
			1.682E-4	4.007E-3	3.426E-3	3.077E-3	1.892E-3	7.693E-2	1.700E+2
			2.871E-4	6.665E-3	9.113E-3	5.603E-3	6.231E-3	4.474E-1	1.757E+2
	5	1000	**4.173E-4**	3.086E-3	1.129E-3	8.738E-4	6.181E-4	5.331E-1	1.826E+2
			4.893E-4	5.960E-3	2.213E-3	1.971E-3	1.181E-3	8.295E-1	2.172E+2
			7.944E-4	1.196E-2	6.147E-3	4.340E-3	4.736E-3	1.124E+0	2.278E+2
	8	1000	**2.654E-3**	1.244E-2	6.459E-3	6.493E-3	3.411E-3	7.518E-1	2.196E+2
			3.476E-3	2.375E-2	1.948E-2	1.036E-2	8.079E-3	1.024E+0	2.700E+2
			5.102E-3	9.649E-2	1.123E+0	1.549E-2	1.826E-2	1.230E+0	2.949E+2
	10	1500	1.760E-3	8.849E-3	2.791E-3	5.074E-3	**1.689E-3**	8.656E-1	1.720E+2
			1.994E-3	1.188E-2	4.319E-3	6.121E-3	2.164E-3	1.145E+0	2.893E+2
			2.418E-3	2.082E-2	1.010E+0	7.243E-3	3.226E-3	1.265E+0	3.391E+2
	15	2000	**2.226E-3**	1.401E-2	4.360E-3	7.892E-3	5.716E-3	9.391E+1	2.358E+2
			3.017E-3	2.145E-2	1.664E-2	9.924E-3	7.461E-3	1.983E+2	2.635E+2
			3.640E-3	4.195E-2	1.260E+0	1.434E-2	1.138E-2	3.236E+2	3.451E+2

From table 7, it can be observed that the performance of NAEMO is comparable to that of the M2M algorithms in general. NAEMO outperforms all the M2M algorithms in case of IMB1, IMB2, IMB4 and IMB6 problems. It is known from [19] that imbalanced mapping difficulty occurs for IMB1-6 problems, thus showcasing the efficacy of NAEMO in tackling these types of problems. One can notice that the HV value in case of IMB3 and IMB5 are comparatively less than other algorithms even though they pose the same challenge as the other problems in IMB1-IMB6. Both of these functions have a concave Pareto-

Table 4: Best, median, worst IGD values for comparing MaOO approaches on M-objective unimodal (DTLZ2 and DTLZ4) problems. Best performance is highlighted in bold.

Problems(P)	M	tot_itr	**NAEMO**	**NSGA-III**	**MOEA/D**	**θ-DEA***	**MOEA/DD**	**GrEA**	**HypE**
DTLZ2	3	250	**2.350E-4**	1.262E-3	5.432E-4	7.567E-4	6.666E-4	6.884E-2	6.732E-2
			3.542E-4	1.357E-3	6.406E-4	9.736E-4	8.073E-4	7.179E-2	6.910E-2
			4.463E-4	2.114E-3	8.006E-4	1.130E-3	1.243E-3	7.444E-2	7.104E-2
	5	350	**4.589E-4**	4.254E-3	1.219E-3	1.863E-3	1.128E-3	1.411E-1	2.761E-1
			5.895E-4	4.982E-3	1.437E-3	2.146E-3	1.291E-3	1.474E-1	2.868E-1
			7.831E-4	5.862E-3	1.727E-3	2.288E-3	1.424E-3	1.558E-1	2.922E-1
	8	500	**1.977E-3**	1.371E-2	3.097E-3	6.120E-3	2.880E-3	3.453E-1	5.475E-1
			2.410E-3	1.571E-2	3.763E-3	6.750E-3	3.291E-3	3.731E-1	6.033E-1
			3.053E-3	1.811E-2	5.198E-3	7.781E-3	4.106E-3	4.126E-1	6.467E-1
	10	750	**1.753E-3**	1.350E-2	2.474E-3	6.111E-3	3.223E-3	4.107E-1	6.778E-1
			2.105E-3	1.528E-2	2.778E-3	6.546E-3	3.752E-3	4.514E-1	6.901E-1
			2.429E-3	1.697E-2	3.235E-3	7.069E-3	4.145E-3	5.161E-1	6.917E-1
	15	1000	**2.209E-3**	1.360E-2	5.254E-3	7.269E-3	4.557E-3	5.087E-1	6.237E-1
			2.903E-3	1.726E-2	6.005E-3	8.264E-3	5.863E-3	5.289E-1	8.643E-1
			4.019E-3	2.114E-2	9.409E-3	9.137E-3	6.929E-3	5.381E-1	3.195E+0
DTLZ4	3	600	**4.209E-5**	2.915E-4	2.929E-1	1.408E-4	1.025E-4	6.869E-2	6.657E-2
			5.963E-5	5.970E-4	4.280E-1	1.918E-4	1.429E-4	7.234E-2	7.069E-2
			1.320E-4	4.286E-1	5.234E-1	5.321E-1	1.881E-4	9.400E-1	5.270E-1
	5	1000	**3.859E-5**	9.849E-4	1.080E-1	2.780E-4	1.097E-4	1.422E-1	2.603E-1
			5.285E-5	1.255E-3	5.787E-1	3.142E-4	1.296E-4	1.462E-1	2.676E-1
			7.452E-5	1.721E-3	7.348E-1	3.586E-4	1.532E-4	1.609E-1	5.301E-1
	8	1250	6.595E-4	5.079E-3	5.298E-1	2.323E-3	**5.271E-4**	3.229E-1	4.792E-1
			7.619E-4	7.054E-3	8.816E-1	3.172E-3	**6.699E-4**	3.314E-1	4.956E-1
			1.208E-3	6.051E-1	9.723E-1	3.635E-3	**9.107E-4**	3.402E-1	5.387E-1
	10	2000	**8.560E-4**	5.694E-3	3.966E-1	2.715E-3	1.291E-3	4.191E-1	6.760E-1
			1.025E-3	6.337E-3	9.203E-1	3.216E-3	1.615E-3	4.294E-1	6.828E-1
			1.189E-3	1.076E-1	1.077E+0	3.711E-3	1.931E-3	4.410E-1	6.877E-1
	15	3000	**9.607E-4**	7.110E-3	5.890E-1	4.182E-3	1.474E-3	4.975E-1	5.986E-1
			1.496E-3	3.431E-1	1.133E+0	5.633E-3	1.881E-3	5.032E-1	6.102E-1
			2.788E-3	1.073E+0	1.249E+0	6.562E-3	3.159E-3	5.136E-1	6.126E-1

front. One might go on to conclude that NAEMO is not suitable for concave Pareto-fronts. However, this is not true as it is clear from Fig. 5c and 5e which shows a dense Pareto-front. The low values are attributed to the position of the reference point chosen for calculating the HV values. In case of the problems IMB7-IMB9, the performance of NAEMO is relatively poor. Thus NAEMO is relatively poor at tackling variable linkages (IMB7-IMB9) as compared to the other M2M algorithms. Even so, the values obtained by NAEMO are quite comparable. NAEMO efficiently explores the decision space which assists to

Table 5: Best, median, worst HV values for comparing MaOO approaches on M-objective multimodal (DTLZ1 and DTLZ3) problems. Best performance is highlighted in bold.

Problems(P)	M	**NAEMO**	**NSGA-III**	**MOEA/D**	**MOEA/DD**	**GrEA**	**HypE**
DTLZ1		**0.973668**	0.973519	0.973541	0.973597	0.967404	0.000000
	3	**0.973668**	0.973217	0.973380	0.973510	0.964059	0.000000
		0.973668	0.971931	0.972484	0.973278	0.828008	0.000000
		0.999897	0.998971	0.998978	0.998980	0.991451	0.000000
	5	**0.999897**	0.998963	0.998969	0.998975	0.844529	0.000000
		0.999897	0.998673	0.998954	0.998968	0.500179	0.000000
		0.999979	0.999975	0.999943	0.999949	0.999144	0.000000
	8	**0.999979**	0.993549	0.999866	0.999919	0.997992	0.000000
		0.994781	0.966432	0.999549	**0.999887**	0.902697	0.000000
		0.999999	0.999991	0.999983	0.999994	0.999451	0.000000
	10	**0.999999**	0.999985	0.999979	0.999990	0.998587	0.000000
		0.999978	0.999969	0.999956	0.999974	0.532348	0.000000
DTLZ3		0.926512	0.926480	0.926598	**0.926617**	0.924652	0.000000
	3	**0.926411**	0.925805	0.925855	0.926346	0.922650	0.000000
		0.925641	0.924234	0.923858	0.924901	0.621155	0.000000
		0.990532	0.990453	0.990543	**0.990558**	0.963021	0.000000
	5	**0.990532**	0.990344	0.990444	0.990515	0.808084	0.000000
		0.990428	0.989510	0.990258	0.990349	0.499908	0.000000
		0.999327	0.999300	0.999328	**0.999343**	0.953478	0.000000
	8	**0.999325**	0.924059	0.999303	0.999311	0.791184	0.000000
		0.999324	0.904182	0.508355	0.999248	0.498580	0.000000
		0.999923	0.999921	0.999922	**0.999923**	0.962168	0.000000
	10	0.999921	0.999918	0.999920	**0.999922**	0.735934	0.000000
		0.999921	0.999910	0.999915	**0.999921**	0.499676	0.000000

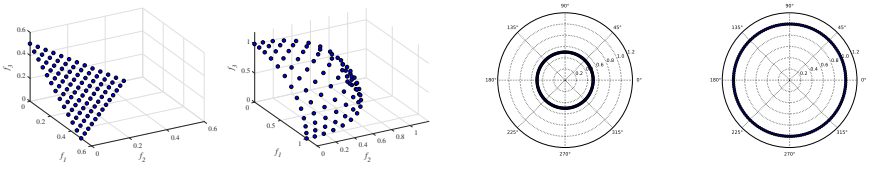
overcome the difficulties and generate a well-diverse and converged PF. This claim is supported by the plots in Fig. 5 where no part of the estimated PF is completely empty (unexplored).

5.5. Analyzing the Mutation Switching Scheme

In this experiment, the most commonly used reproduction operators viz. SBX crossover (Eq. (16)) followed by polynomial mutation (Eq. (19)) are considered for generation of new candidate solutions in the NAEMO framework

Table 6: Best, median, worst HV values for comparing MaOO approaches on M-objective unimodal (DTLZ2 and DTLZ4) problems. Best performance is highlighted in bold.

Problems(P)	M	**NAEMO**	**NSGA-III**	**MOEA/D**	**MOEA/DD**	**GrEA**	**HypE**
DTLZ2		**0.926683**	0.926626	0.926666	0.926674	0.924246	0.925691
	3	**0.926662**	0.926536	0.926639	0.926653	0.923994	0.925650
		0.926651	0.926395	0.926613	0.926596	0.923675	0.925531
		0.990535	0.990459	0.990529	**0.990535**	0.990359	0.987889
	5	**0.990535**	0.990400	0.990518	0.990527	0.990214	0.987665
		0.990521	0.990328	0.990511	0.990512	0.990064	0.987545
		0.999352	0.999320	0.999341	0.999346	0.999991	0.997401
	8	**0.999340**	0.978936	0.999329	0.999337	0.999670	0.996551
		0.999329	0.919680	0.999307	**0.999329**	0.989264	0.995761
		0.999923	0.999918	0.999922	**0.999952**	0.997636	0.998995
	10	0.999923	0.999916	0.999921	**0.999932**	0.996428	0.998934
		0.999921	0.999915	0.999919	**0.999921**	0.994729	0.998913
DTLZ4		**0.926733**	0.926659	0.926729	0.926731	0.924613	0.926351
	3	**0.926733**	0.926705	0.926725	0.926729	0.924094	0.926223
		0.926652	0.799572	0.500000	**0.926725**	0.500000	0.800459
		0.990581	**0.991102**	0.990569	0.990575	0.990514	0.988150
	5	0.990569	0.990413	0.990568	**0.990573**	0.990409	0.988009
		0.990431	0.990156	0.973811	**0.990570**	0.990221	0.987743
		0.999382	0.999363	0.999363	0.999364	0.999102	0.997994
	8	**0.999371**	0.999361	0.998497	0.999363	0.999039	0.997730
		0.999327	0.994784	0.995753	0.998360	0.998955	0.997569
		0.999921	0.999915	0.999918	**0.999921**	0.999653	0.999019
	10	**0.999921**	0.999910	0.999907	0.999920	0.999608	0.998934
		0.999921	0.999827	0.999472	0.999917	0.999547	0.998921



(a) DTLZ1 ($M = 3$) (b) DTLZ4 ($M = 3$) (c) DTLZ1 ($M = 8$) (d) DTLZ2 ($M = 8$)

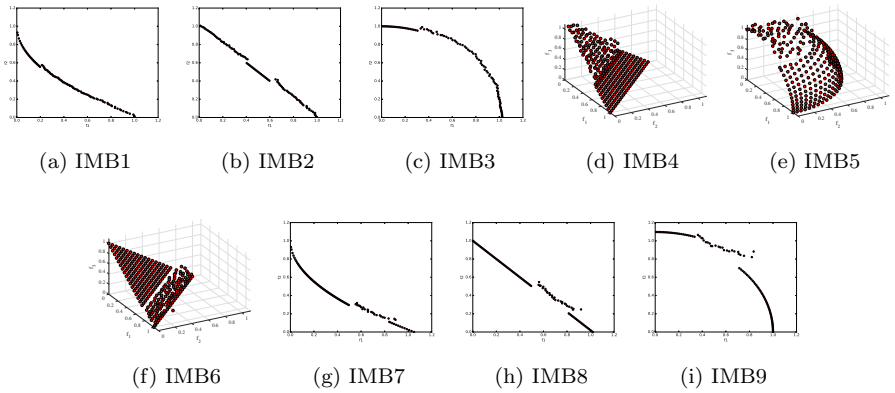
Figure 4: Demonstrating NAEMO's ability to generate Pareto-fronts, similar to true Pareto-fronts, for both low and high number of objectives of different types of DTLZ problems

Table 7: Best, median, worst HV values for comparing NAEMO with M2M-based MaOO algorithms on IMB problems. Best performance is highlighted in bold.

Problems	NAEMO	NSGA-II	MOEA/D	SMS-EMOA	SPEA2	GVEGA
IMB1	**0.6477**	0.6375	0.6387	0.6402	0.6384	0.6475
	0.6441	0.6360	0.6375	0.6386	0.6372	0.6408
	0.6399	0.6353	0.6354	0.6363	0.6351	0.5969
IMB2	0.4734	0.4605	0.4627	0.4639	0.4605	**0.4750**
	0.4710	0.4577	0.4608	0.4592	0.4564	0.4509
	0.4657	0.4537	0.4583	0.4411	0.4487	0.4224
IMB3	0.1801	0.1828	0.1851	0.1845	0.1824	**0.1964**
	0.1745	0.1815	0.1836	0.1834	0.1802	**0.1950**
	0.1639	0.1801	0.1824	0.1819	0.1783	**0.1914**
IMB4	**0.7886**	0.7445	0.7803	0.7792	0.7476	0.7798
	0.7812	0.7424	0.7795	0.7786	0.7421	0.7790
	0.7531	0.7398	**0.7785**	0.7783	0.7364	0.7784
IMB5	0.4117	0.3874	**0.4266**	0.4169	0.3973	0.4215
	0.4026	0.3842	**0.4229**	0.4140	0.3906	0.4209
	0.3974	0.3802	0.4202	0.4119	0.3832	**0.4205**
IMB6	**0.8046**	0.7700	0.7916	0.7859	0.7814	0.7837
	0.7996	0.7686	0.7909	0.7856	0.7807	0.7833
	0.7961	0.7675	0.7904	0.7853	0.7801	0.7828
IMB7	0.6501	0.6499	0.6545	**0.6559**	0.6515	0.6540
	0.6471	0.6482	0.6540	**0.6550**	0.6505	0.6537
	0.6443	0.6464	0.6534	**0.6542**	0.6494	0.6531
IMB8	0.4777	0.4798	0.4840	0.4852	0.4811	**0.4863**
	0.4703	0.4774	0.4830	0.4835	0.4795	**0.4857**
	0.4519	0.4756	0.4820	0.4820	0.4768	**0.4848**
IMB9	0.1836	0.1925	0.1975	0.1974	0.1930	**0.2011**
	0.1777	0.1912	0.1960	0.1961	0.1919	**0.2005**
	0.1719	0.1896	0.1946	0.1947	0.1911	**0.1998**

in line 19 of Algorithm 1. This gives a fair comparison as all the other competitor algorithms (Section 5.2) are also implemented with these reproduction operators. The performance of this framework (NAEMO-SBX) is noted for 10-objective DTLZ1 to DTLZ4 problems in Table 8.

The values in Table 8 show that NAEMO with the mutation switching scheme is more robust than NAEMO only with the commonly used reproduction operators. However, NAEMO-SBX framework also gives better performance compared to other competitor algorithms in most of the cases. Thus, mutation switching scheme switches among several reproduction strategies and



(a) IMB1 (b) IMB2 (c) IMB3 (d) IMB4 (e) IMB5

(f) IMB6 (g) IMB7 (h) IMB8 (i) IMB9

Figure 5: Resulting Pareto-fronts from NAEMO for IMB problems demonstrating NAEMO's capability to discover solutions even in difficult regions of the objective space.

Table 8: Best, median, worst IGD values for comparing MaOO approaches on 10-objective DTLZ problems. Best performance is highlighted in bold.

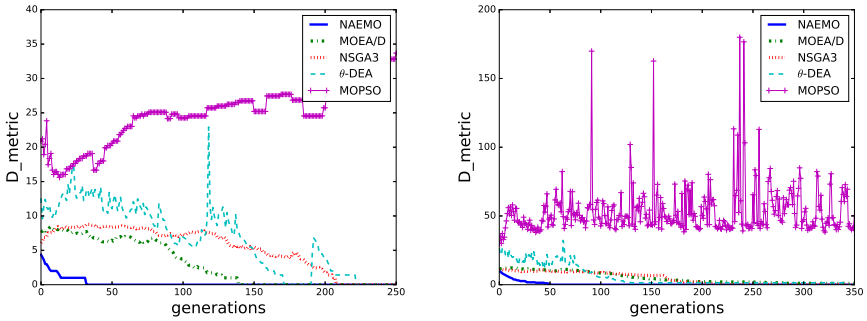
Problems(P)	**NAEMO-SBX**	**NAEMO**	**NSGA-III**	**MOEA/D**	**θ-DEA***	**MOEA/DD**	**GrEA**	**HypE**
DTLZ1	1.787E-3	**5.022E-4**	2.215E-3	3.872E-3	1.999E-3	1.828E-3	1.176E-1	1.427E+1
	2.668E-3	**8.536E-4**	3.462E-3	5.073E-3	2.268E-3	2.225E-3	1.586E-1	1.693E+1
	2.751E-3	**1.762E-3**	6.869E-3	6.130E-3	2.425E-3	2.467E-3	5.110E-1	2.034E+1
DTLZ2	1.827E-3	**1.753E-3**	1.350E-2	2.474E-3	6.111E-3	3.223E-3	4.107E-1	6.778E-1
	1.980E-3	**2.105E-3**	1.528E-2	2.778E-3	6.546E-3	3.752E-3	4.514E-1	6.901E-1
	2.369E-3	**2.429E-3**	1.697E-2	3.235E-3	7.069E-3	4.145E-3	5.161E-1	6.917E-1
DTLZ3	4.414E-3	1.760E-3	8.849E-3	2.791E-3	5.074E-3	**1.689E-3**	8.656E-1	1.720E+2
	1.842E-2	**1.994E-3**	1.188E-2	4.319E-3	6.121E-3	2.164E-3	1.145E+0	2.893E+2
	2.190E-2	**2.418E-3**	2.082E-2	1.010E+0	7.243E-3	3.226E-3	1.265E+0	3.391E+2
DTLZ4	9.227E-4	**8.560E-4**	5.694E-3	3.966E-1	2.715E-3	1.291E-1	4.191E-1	6.760E-1
	1.016E-3	**1.025E-3**	6.337E-3	9.203E-1	3.216E-3	1.615E-3	4.294E-1	6.828E-1
	1.042E-3	**1.189E-3**	1.076E-1	1.077E+0	3.711E-3	1.931E-3	4.410E-1	6.877E-1

effectively combines their advantages. It is highlighted that the proposed mutation switching scheme is a generic framework where besides the reproduction strategies considered for NAEMO, any other reproduction strategy could also be integrated and hence, enabling it to cover a more wide range of problem characteristics.

5.6. Diversity Plots

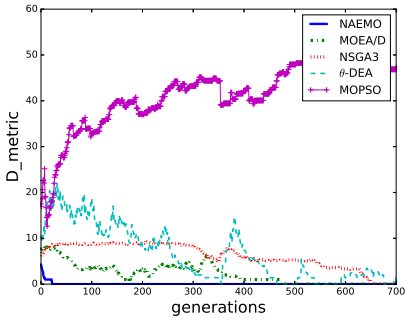
In this section, we present a comparison of the *D_metric* plots of NAEMO for the two multi-modal DTLZ functions, DTLZ1 and DTLZ3. For comparison, we also present the *D_metric* plots of several other MaOEAs such as NSGA-III, MOEA/D, θ-DEA and MOPSO for the same problems. The multi-modal functions usually cause changes in diversity at the regions of local optimum [34]. As can be seen from the plots in Fig. 6, the *D_metric* plot corresponding to NAEMO is monotonically decreasing as proven in section 4.8 and finally converges to 0. Not only does NAEMO preserve diversity, it attains the ideal diversity much faster as compared to NSGA-III, MOEA/D and θ-DEA. In case of 3-objective DTLZ1, NAEMO achieves ideal diversity by approximately 30 generations, while MOEA/D takes 140 generations, NSGA-III takes nearly 210 generations and θ-DEA takes around 220 generations. Thus, NAEMO achieves ideal diversity approximately 5 times faster than MOEA/D and 7 times faster than NSGA-III and θ-DEA. An even faster convergence can be observed for 3-objective DTLZ3 where NAEMO achieves ideal diversity after only 20 generations which is about 25 times faster than MOEA/D and 32 times faster than NSGA-III and θ-DEA. Moreover, the diversity does not monotonically improve for either of the compared algorithms. This huge difference in the *D_metric* convergence is owing to the effective utilization of the neighborhood property to our advantage in NAEMO. For comparison, the *D_metric* plot for MOPSO is also considered. In contrast with other MaOEAs, for MOPSO, in case of the MOO problems (Fig. 6a and 6c), the *D_metric* is seen to increase showing deterioration of diversity and for MaOO problems (Fig. 6b and 6d), the *D_metric* remains stagnant with changing generations. The later observation also points to the phenomenon of saturation of population by non-dominated solutions at early generations for a Pareto-dominance based MaOEA such as MOPSO.

The experiment in this section thus shows that NAEMO very successfully takes advantage of the neighborhood property to generate both very high convergence as well as diversity attainment rate.

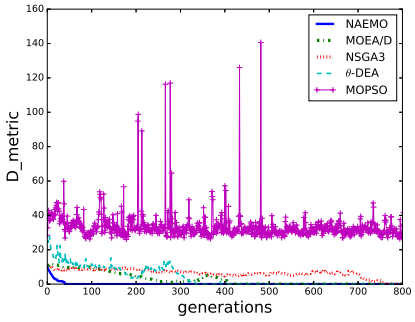


(a) DTLZ1(3)

(b) DTLZ1(8)



(c) DTLZ3(3)



(d) DTLZ3(8)

Figure 6: Resulting *D_metric* plots showing NAEMO achieves uniform diversity in the objective space much faster than several other MaOEAs.

5.7. Miscellaneous Experiments

In this sub-section, we perform some more experiments to further establish the efficacy of the proposed approach (NAEMO). These experiments are aimed at demonstrating the performance of NAEMO with respect to some variants of Multi-Objective Particle Swarm Optimization (MOPSO), studying its performance in terms of a third performance metric viz. the Purity metric [10], performing comparisons on problems with scaled and disconnected Pareto-front and finding computational time requirements of NAEMO.

Table 9: Average HV for comparing NAEMO with MOPSO variants. Best performance is highlighted in bold.

MaOEAs		Number of objectives (M)					Number of objectives (M)			
		3	5	8	10		3	5	8	10
NAEMO	DTLZ1	**1.304662**	**1.609497**	**2.143047**	**2.593741**	DTLZ2	**0.744830**	**1.308778**	**1.980806**	**2.515441**
MOPSO		0	0	0	0		0.638144	0.510065	0.060562	0.082047
dMOPSO		1.074976	1.482412	1.824428	2.317805		0.712523	1.239853	1.816420	2.428399
NAEMO	DTLZ3	**0.744840**	**1.308723**	**1.980405**	**2.515377**	DTLZ4	**0.744848**	**1.308761**	**1.980838**	**2.515418**
MOPSO		0	0	0	0		0	0	0	0
dMOPSO		0.665529	1.252229	1.428208	2.107556		0.677459	1.203429	1.829561	2.438748

5.7.1. Comparison of NAEMO with MOPSO variants

In order to compare the performance of NAEMO with a different meta-heuristic based MOO algorithm, we perform a comparison of NAEMO with two PSO based MOO algorithms viz. MOPSO[40] and dMOPSO[18]. The latter is a decomposition based algorithm and is much more recent than MOPSO. Considering the approach of [13], i.e., by setting the reference vector to 1.1 times the nadir point with respect to the true Pareto-front, followed by normalization of the obtained Pareto-front and the reference point between the ideal and nadir points, the hypervolume of this normalized Pareto-front is evaluated and the average value is noted in Table 9 for comparison. As can be observed from Table 9, NAEMO outperforms both the algorithms for all the cases in terms of the HV value. The HV of MOPSO is 0 in many of the cases. This shows that the final Pareto front, obtained from MOPSO, is completely outside the hyper-rectangle used for calculating the HV.

5.7.2. Performance of NAEMO based on Purity metric

Another metric viz. the purity metric can be used for comparison two or more Pareto-fronts [10], i.e., the results of two or more algorithms. For comparison of $\mathcal{K}$ Pareto-fronts, the first step is to obtain the unified Pareto-front ($\mathcal{A}^\star$) by performing non-dominated sorting on the union of these $\mathcal{K}$ Pareto-fronts and generating the rank-one solutions as shown by Eq. (31) where $ndset(.)$ fetches

the non-dominated set of the argument.

$$\mathcal{A}^{\star} = ndset\left(\cup_{i=1}^{\mathcal{K}} \mathcal{A}_i\right) \tag{31}$$

The next step is to obtain the set $\mathcal{Q}_i$ for every i^{th} Pareto-front using Eq. (32) which points to all the common elements of $\mathcal{A}_i$ and the unified Pareto-front, $\mathcal{A}^{\star}$.

$$\mathcal{Q}_i = \{F(X) | F(X) \in \mathcal{A}_i \wedge F(X) \in \mathcal{A}^{\star}\} \tag{32}$$

Finally, the purity metric ($\mathcal{P}_i$), for the i^{th} Pareto-front, with respect to the $\mathcal{K}$ Pareto-fronts under consideration, is evaluated by Eq. (33).

$$\mathcal{P}_i = \frac{\gamma_i^{\star}}{\gamma_i}, \text{ where } \gamma_i^{\star} = |\mathcal{Q}_i| \text{ and } \gamma_i = |\mathcal{A}_i|, \forall i = 1, \cdots, \mathcal{K} \tag{33}$$

This metric indicates the performance of the i^{th} algorithm with respect to the contribution towards an unified Pareto-front. The purity metric is bounded and can even be equal to 1 for all the Pareto-fronts, as $\sum_{i=1}^{\mathcal{K}} \mathcal{P}_i$ needs not to be 1. It should be noted that the purity metric [10] is a comparative metric, not a qualitative/quantitative metric for a single Pareto-front.

In terms of purity metric, the performance of NAEMO as compared to several other MaOEAs viz. HypE [25], MOPSO [40], NSGA-III [12] and θ-DEA [13] are noted in Table 10. In all the cases, NAEMO is observed to be superior to other MaOEAs considered in the experiment. Also, for DTLZ4, NAEMO, NSGA-III and θ-DEA have much higher purity values than HyPE and MOPSO. This shows the necessity of decomposition based MaOEAs for problems with biased density of solutions.

5.7.3. Weaknesses of NAEMO - Scaled and Disconnected Pareto-Fronts

As a preliminary experiment, we consider WFG1 and WFG2 problems which are examples of problems with scaled and disconnected Pareto-front, respectively. For implementing these problems, according to [13], with $(M-1)$ position-related variables and $(d-M+1)$ distance-related variables, the number of decision variables (d) is set to 24. Only for WFG2 problem, d is set to 23 when M is even as it requires even number of distance related variables due to

Table 10: Average value for Purity metric for comparing NAEMO with several MaOEAs. Best performance is highlighted in bold.

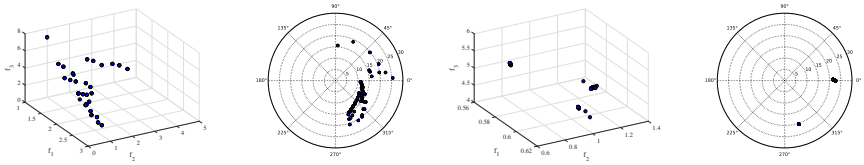
MaOEAs		Number of objectives (M)					Number of objectives (M)			
		3	5	8	10		3	5	8	10
NAEMO	DTLZ1	**1.000000**	**1.000000**	**1.000000**	**1.000000**	DTLZ2	**1.000000**	**1.000000**	**1.000000**	**1.000000**
HypE		0.000000	0.004762	0.012821	0.537879		0.318681	0.195238	0.339744	0.647273
MOPSO		0.017241	0.066667	0.326923	0.742424		0.406593	0.404762	0.551282	0.469091
NSGA-III		0.431034	0.633333	0.858974	0.946970		0.714286	0.638095	0.570513	0.512727
θ-DEA		0.965517	0.795238	0.980769	0.992424		0.637363	0.780952	0.750000	0.730909
NAEMO	DTLZ3	**1.000000**	**1.000000**	**1.000000**	**1.000000**	DTLZ4	**1.000000**	**1.000000**	**1.000000**	**1.000000**
HypE		0.092105	0.066667	0.500000	0.450909		0.844444	0.766667	0.224359	0.221818
MOPSO		0.026316	0.190476	0.282051	0.269091		0.077778	0.333333	0.538462	0.676364
NSGA-III		0.236842	0.509524	0.416667	0.505455		0.966667	0.990476	**1.000000**	**1.000000**
θ-DEA		0.421053	0.247619	0.634615	0.578182		0.977778	0.990476	0.993590	**1.000000**

Table 11: Average HV values for comparing MaOO approaches on WFG1 and WFG2 problems. Best performance is highlighted in bold.

Problems(P)	M	tot_itr	NAEMO	NSGA-III	MOEA/D	GrEA	HypE	dMOPSO
WFG1	3	400	0.387079	0.669729	0.657143	0.846287	**0.976181**	0.403170
	5	750	0.418739	0.859552	**1.349888**	1.268898	0.911020	0.461233
	8	1500	0.504653	1.424963	1.755326	**1.769013**	1.536599	0.484046
	10	2000	0.553354	2.249535	1.799394	**2.365107**	2.268813	0.536340
WFG2	3	400	0.289046	1.226956	1.111085	1.226099	**1.244737**	1.125810
	5	750	0.322285	**1.598410**	1.520168	1.570086	1.535704	1.478517
	8	1500	0.390006	**2.136525**	2.016854	2.102930	2.084336	1.971067
	10	2000	0.445272	**2.588104**	2.459026	2.570389	2.556327	2.406484

its nature of non-separable reductions. Considering the approach of [13], i.e., by setting the reference vector to 1.1 times the nadir point, followed by normalization of the obtained Pareto-front and the reference point between the ideal and nadir points, the hypervolume of this normalized Pareto-front is evaluated and the average value is noted in Table 11 for comparison.

From Table 11, it can be seen that NAEMO performs worst in all these cases among the competitor algorithms. The reason for this poor performance is that unlike other algorithms, NAEMO does not have any explicit scaling mechanism integrated with the framework. NAEMO performs poor for disconnected Pareto fronts because of its tendency to preserve diversity. The reason for its



(a) WFG1 ($M = 3$) (b) WFG1 ($M = 10$) (c) WFG2 ($M = 3$) (d) WFG2 ($M = 10$)

Figure 7: Resulting Pareto-fronts from NAEMO for WFG1 and WFG2 problems demonstrating NAEMO's optimization performance for problems with scaled and disconnected Pareto-fronts.

strength in other problems becomes a reason for weakness in case of problems with disconnected Pareto fronts. For further analysis, we also plot the final Pareto-fronts obtained by NAEMO for 3 and 10-objective WFG1 and WFG2 problems in Fig.7. Figures 7b and 7d are spherical coordinate plots [39]. For WFG1 (Fig. 7a, 7b), still a considerable part of the Pareto-front is discovered as opposed to WFG2 problems (Fig. 7c, 7d).

5.7.4. Computational Time Requirement

To assess the computational time requirements, the average execution time of NAEMO for several cases of DTLZ problems with different number of objectives are noted. NSGA-III is implemented in the same platform as per the specifications in [12] and its execution time is also noted for several test cases. These average execution times are compared in Fig. 8. It is observed from the bar graphs that NAEMO requires lesser average execution time than NSGA-III. This is primarily because in NAEMO, the main computation intensive parts get initiated only when the if condition in line 20 of Algorithm 1 gets satisfied. The general trend is increase in execution time with increase in number of objectives, yet it can be seen that MaOEAs needed lesser time for 8-objective problems than 5-objective problems. The primary reason for this is number of reference lines (n) and associatively, number of candidate solutions are lesser for 8-objective problems than 5-objective problems (Table 2) due to the use of two-layered reference lines initialization [12, 27] for 8-objective problems as opposed to single

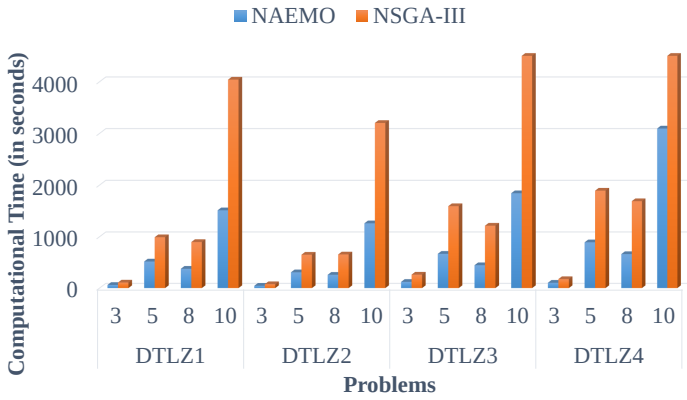


Figure 8: Illustration of computational time requirement of NAEMO as compared to NSGA-III. It should be noted that for 10-objective DTLZ3 and DTLZ4 problems, NSGA-III took more than 4500 seconds which is the maximum limit considered along y-axis for better scaling.

layer reference lines initialization for 5-objective problems.

All these experiments establishes the overall efficacy of the proposed algorithm (NAEMO) which is also supported by the theoretical foundations.

6. Conclusion and Future Research Scope

Motivated by the success of the reference direction-based evolutionary many-objective optimization algorithms, this work proposes a novel approach viz. Neighborhood-sensitive Archived Evolutionary Many-objective Optimization (NAEMO) where the neighborhood property of the many-objective optimization problems is identified and used for selecting the mating candidate solutions for generation of new candidate solutions. Moreover, NAEMO aims to preserve and monotonically improve the diversity through periodic filtering of archive where if a candidate solution ever gets associated with a reference direction, it is never lost along the evolutionary process. The robust performance of NAEMO to tackle many-objective optimization problems with several characteristics like unimodality, multi-modality, biased density of solutions, meta-variable mapping, imbalance mapping difficulty and variable linkage difficulty, has been demonstrated through experiments on problems from DTLZ and IMB test suite.

Results indicate that NAEMO outperforms several contemporary state-of-the-art algorithms on these test problems.

However, the proposed work suffers from the following limitations:

1. Experimental validation of NAEMO is presented in this manuscript through some experiments. More experiments are needed to be performed to analyze the extent of problems NAEMO can handle. More specifically, rigorous experiments on scaled problems and problems with disconnected Pareto-fronts are yet to be done. Only preliminary results are shown in this manuscript which demonstrate that these problem characteristics pose a challenge for NAEMO. In order to address such a wide range of problems, an objective scaling strategy may be introduced in future versions of this algorithm.
2. Moreover, a few parameters of NAEMO viz. η_m, $flag1$ and $flag2$ are not adaptive in the current version. Since, the characteristics of benchmark problems are known apriori, assigning values for these parameters was simple. However, it will be difficult for practical problems whose underlying characteristics are unknown.
3. Another avenue, that is yet to be explored, is reference direction specific parameter adaptation.

In future, we aim to improvise the current version of the proposed approach such that these above-mentioned issues can be addressed.

References

[1] C. A. Coello Coello, Recent results and open problems in evolutionary multiobjective optimization, in: C. Martín-Vide, R. Neruda, M. A. Vega-Rodríguez (Eds.), Theory and Practice of Natural Computing, Springer International Publishing, Cham, 2017, pp. 3–21.

[2] M. Pal, S. Bandyopadhyay, Reliability of convergence metric and hypervolume indicator for many-objective optimization, in: Control, Instrumentation, Energy & Communication (CIEC), 2016 2nd International Conference on, IEEE, 2016, pp. 511–515. doi:10.1109/CIEC.2016.7513806.

[3] A. Zhou, B. Qu, H. Li, S. Zhao, P. N. Suganthan, Q. Zhang, Multiobjective evolutionary algorithms: A survey of the state of the art, Swarm and Evolutionary Computation 1 (1) (2011) 32–49. doi:10.1016/j.swevo.2011.03.001.
URL http://dx.doi.org/10.1016/j.swevo.2011.03.001

[4] M. Pal, S. Saha, S. Bandyopadhyay, DECOR: Differential evolution using clustering based objective reduction for many-objective optimization, Information Sciences 423 (2018) 200 – 218. doi:https://doi.org/10.1016/j.ins.2017.09.051.
URL http://www.sciencedirect.com/science/article/pii/S0020025517309696

[5] P. J. Fleming, R. C. Purshouse, R. J. Lygoe, Many-objective optimization: An engineering design perspective, in: C. A. Coello Coello, A. Hernández Aguirre, E. Zitzler (Eds.), Evolutionary Multi-Criterion Optimization: Third International Conference, EMO 2005, Guanajuato, Mexico, March 9-11, 2005. Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 14–32. doi:10.1007/978-3-540-31880-4_2.
URL https://doi.org/10.1007/978-3-540-31880-4_2

[6] J. G. Herrero, A. Berlanga, J. M. M. López, Effective evolutionary algorithms for many-specifications attainment: Application to air traffic control tracking filters, IEEE Transactions on Evolutionary Computation 13 (1) (2009) 151–168. doi:10.1109/TEVC.2008.920677.
URL http://dx.doi.org/10.1109/TEVC.2008.920677

[7] M. Pal, S. Bandyopadhyay, Many-objective feature selection for motor imagery eeg signals using differential evolution and support vector machine, in: 2016 International Conference on Microelectronics, Computing and Communications (MicroCom), 2016, pp. 1–6. doi:10.1109/MicroCom.2016.7522574.

[8] Shashi, K. Deep, V. K. Katiyar, Multi objective extraction optimization of

bioactive compounds from gardenia using real coded genetic algorithm, in: C. T. Lim, J. C. H. Goh (Eds.), 6th World Congress of Biomechanics (WCB 2010). August 1-6, 2010 Singapore, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 1463–1466. doi:10.1007/978-3-642-14515-5_373.
URL https://doi.org/10.1007/978-3-642-14515-5_373

[9] P. Rakshit, A. Konar, S. Das, L. C. Jain, A. K. Nagar, Uncertainty management in differential evolution induced multiobjective optimization in presence of measurement noise, IEEE Transactions on Systems, Man, and Cybernetics: Systems 44 (7) (2014) 922–937. doi:10.1109/TSMC.2013.2282118.

[10] S. Bandyopadhyay, A. Mukherjee, An algorithm for many-objective optimization with reduced objective computations: A study in differential evolution, IEEE Transactions on Evolutionary Computation 19 (3) (2015) 400–413. doi:10.1109/TEVC.2014.2332878.

[11] R. Sengupta, S. Saha, Reference point based archived many objective simulated annealing, Information Sciencesdoi:https://doi.org/10.1016/j.ins.2018.05.013.
URL http://www.sciencedirect.com/science/article/pii/S0020025518303669

[12] K. Deb, H. Jain, An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part i: solving problems with box constraints, Evolutionary Computation, IEEE Transactions on 18 (4) (2014) 577–601.

[13] Y. Yuan, H. Xu, B. Wang, X. Yao, A new dominance relation-based evolutionary algorithm for many-objective optimization, IEEE Trans. Evolutionary Computation 20 (1) (2016) 16–37. doi:10.1109/TEVC.2015.2420112.
URL http://dx.doi.org/10.1109/TEVC.2015.2420112

[14] Q. Zhang, H. Li, MOEA/D: A multiobjective evolutionary algorithm based on decomposition, IEEE Trans. Evolutionary Computation 11 (6) (2007)

712–731. doi:10.1109/TEVC.2007.892759.
URL http://dx.doi.org/10.1109/TEVC.2007.892759

[15] K. Li, K. Deb, Q. Zhang, S. Kwong, An evolutionary many-objective optimization algorithm based on dominance and decomposition, IEEE Transactions on Evolutionary Computation 19 (5) (2015) 694–716.

[16] J. Luo, Y. Yang, X. Li, Q. Liu, M. Chen, K. Gao, A decomposition-based multi-objective evolutionary algorithm with quality indicator, Swarm and Evolutionary Computation 39 (2018) 339 – 355. doi:https://doi.org/10.1016/j.swevo.2017.11.004.
URL http://www.sciencedirect.com/science/article/pii/S2210650217303383

[17] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii, in: M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. Merelo, H.-P. Schwefel (Eds.), Parallel Problem Solving from Nature PPSN VI, Springer Berlin Heidelberg, Berlin, Heidelberg, 2000, pp. 849–858.

[18] S. Zapotecas Martínez, C. A. Coello Coello, A multi-objective particle swarm optimizer based on decomposition, in: Proceedings of the 13th annual conference on Genetic and evolutionary computation, ACM, 2011, pp. 69–76.

[19] H.-L. Liu, L. Chen, K. Deb, E. D. Goodman, Investigating the effect of imbalance between convergence and diversity in evolutionary multiobjective algorithms, IEEE Transactions on Evolutionary Computation 21 (3) (2017) 408–425.

[20] H. L. Liu, F. Gu, Q. Zhang, Decomposition of a multiobjective optimization problem into a number of simple multiobjective subproblems, IEEE Transactions on Evolutionary Computation 18 (3) (2014) 450–455. doi:10.1109/TEVC.2013.2281533.

[21] G. Rudolph, On a multi-objective evolutionary algorithm and its convergence to the pareto set, in: 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No.98TH8360), 1998, pp. 511–516. doi:10.1109/ICEC.1998.700081.

[22] T. Hanne, On the convergence of multiobjective evolutionary algorithms, European Journal of Operational Research 117 (3) (1999) 553 – 564. doi:https://doi.org/10.1016/S0377-2217(98)00262-8.
URL http://www.sciencedirect.com/science/article/pii/S0377221798002628

[23] Y. L. Li, Y. R. Zhou, Z. H. Zhan, J. Zhang, A primary theoretical study on decomposition-based multiobjective evolutionary algorithms, IEEE Transactions on Evolutionary Computation 20 (4) (2016) 563–576. doi:10.1109/TEVC.2015.2501315.

[24] K. Deb, L. Thiele, M. Laumanns, E. Zitzler, Scalable test problems for evolutionary multiobjective optimization, in: A. Abraham, L. Jain, R. Goldberg (Eds.), Evolutionary Multiobjective Optimization: Theoretical Advances and Applications, Springer London, London, 2005, pp. 105–145. doi:10.1007/1-84628-137-7_6.
URL https://doi.org/10.1007/1-84628-137-7_6

[25] J. Bader, E. Zitzler, HypE: An algorithm for fast hypervolume-based many-objective optimization, Evolutionary Computation 19 (1) (2011) 45–76. doi:10.1162/EVCO_a_00009.
URL http://dx.doi.org/10.1162/EVCO_a_00009

[26] S. Yang, M. Li, X. Liu, J. Zheng, A grid-based evolutionary algorithm for many-objective optimization, IEEE Trans. Evolutionary Computation 17 (5) (2013) 721–736. doi:10.1109/TEVC.2012.2227145.
URL http://dx.doi.org/10.1109/TEVC.2012.2227145

[27] I. Das, J. E. Dennis, Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems, SIAM Journal on Optimization 8 (3) (1998) 631–657. doi:10.1137/S1052623496307510.
URL http://dx.doi.org/10.1137/S1052623496307510

[28] K. Deb, R. B. Agrawal, Simulated binary crossover for continuous search space, Complex Systems 9 (1994) 1–34.

[29] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, Journal of global optimization 11 (4) (1997) 341–359.

[30] S. Das, S. S. Mullick, P. Suganthan, Recent advances in differential evolution - an updated survey, Swarm and Evolutionary Computation 27 (2016) 1 – 30. doi:https://doi.org/10.1016/j.swevo.2016.01.004.
URL http://www.sciencedirect.com/science/article/pii/S2210650216000146

[31] H. Sharma, J. C. Bansal, K. V. Arya, Fitness based differential evolution, Memetic Computing 4 (4) (2012) 303–316. doi:10.1007/s12293-012-0096-9.
URL https://doi.org/10.1007/s12293-012-0096-9

[32] R. P. Parouha, K. N. Das, A memory based differential evolution algorithm for unconstrained optimization, Applied Soft Computing 38 (2016) 501 – 517. doi:https://doi.org/10.1016/j.asoc.2015.10.022.
URL http://www.sciencedirect.com/science/article/pii/S1568494615006602

[33] K. Deb, Multi-objective optimization using evolutionary algorithms, Vol. 16, John Wiley & Sons, 2001.

[34] R. Sengupta, M. Pal, S. Saha, S. Bandyopadhyay, Population dynamics indicators for evolutionary many-objective optimization, in: C. R. Panigrahi,

A. K. Pujari, S. Misra, B. Pati, K.-C. Li (Eds.), Progress in Advanced Computing and Intelligent Engineering, Springer Singapore, Singapore, 2019, pp. 261–271. doi:10.1007/978-981-13-0224-4_24.
URL https://doi.org/10.1007/978-981-13-0224-4_24

[35] D. A. V. Veldhuizen, G. B. Lamont, Multiobjective evolutionary algorithm research: A history and analysis (tr-98-03) (1998).

[36] A. Auger, J. Bader, D. Brockhoff, E. Zitzler, Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications, Theor. Comput. Sci. 425 (2012) 75–103. doi:10.1016/j.tcs.2011.03.012.
URL http://dx.doi.org/10.1016/j.tcs.2011.03.012

[37] H. L. Liu, L. Chen, Q. Zhang, K. Deb, An evolutionary many-objective optimisation algorithm with adaptive region decomposition, in: 2016 IEEE Congress on Evolutionary Computation (CEC), 2016, pp. 4763–4769. doi:10.1109/CEC.2016.7744399.

[38] J. Bader, E. Zitzler, A hypervolume-based optimizer for high-dimensional objective spaces, in: D. Jones, M. Tamiz, J. Ries (Eds.), New Developments in Multiple Objective and Goal Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 35–54. doi:10.1007/978-3-642-10354-4_3.
URL http://dx.doi.org/10.1007/978-3-642-10354-4_3

[39] Z. He, G. G. Yen, Visualization and performance metric in many-objective optimization, IEEE Trans. Evolutionary Computation 20 (3) (2016) 386–402. doi:10.1109/TEVC.2015.2472283.
URL http://dx.doi.org/10.1109/TEVC.2015.2472283

[40] C. A. C. Coello, G. T. Pulido, M. S. Lechuga, Handling multiple objectives with particle swarm optimization, IEEE Transactions on evolutionary computation 8 (3) (2004) 256–279.