COMPREHENSIVE MULTI-OBJECTIVE OPTIMIZATION AND DECISION MAKING

By

Anirudh Suresh

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Mechanical Engineering—Doctor of Philosophy

2025

ABSTRACT

The typical aim of a multi-objective evolutionary algorithm (MOEA) is to identify a set of well-converged and uniformly distributed Pareto optimal (PO) solutions. This is followed by a multi-criterion decision making (MCDM) step where the decision-maker (DM) has to select a desired solution for further consideration. A convenient and effective MCDM process can be performed when a well-distributed set of Pareto optimal (PO) solutions is provided and the DM is able to visualize, analyze and interpret the solutions. These PO solutions can be associated with unique identifiers - vectors of the same dimension as the Pareto surface. While PO solutions can form arbitrarily complex shapes based on non-domination properties, these identifiers can have simpler properties such as lying on the unit simplex or unit sphere. Since these identifiers inherently capture the properties of the PO frontier, they can be readily used during multi-objective optimization (MOO) to achieve a good distribution of solutions in the corresponding identifier spaces. While most algorithms focus on achieving a good distribution of solutions in the objective space, a good distribution of solutions in the decision-making space can be immensely helpful to the DM. We present and compare several identifier spaces with respect to their properties, and advantages and disadvantages in optimization, visualization and decision-making and propose methods to achieve a superior distribution of solutions in these identifier spaces. We also demonstrate that a combination of these identifiers can be used during optimization to achieve desired distributions for subsequent successful decision-making.

The solutions achieved by an MOEA and provided to the DM must provide a comprehensive representation of the PO solutions. These solutions should span the whole PO front while being well-converged and uniformly distributed. However, this cannot be guaranteed in practical scenarios due to the stochasticity of the algorithm and the difficulties associated with the problem itself. A seemingly incomplete Pareto front (PF) is highly problematic during decision-making as the DM might desire more solutions in sparse regions of the current non-dominated (ND) front. Unexpected gaps and discontinuities in the PO front can lead to a lack of trust in the solutions obtained. In this study, we propose a convenient machine learning (ML) assisted multi-criterion decision-making

framework that can alleviate some of these issues. We propose to train ML models to map from pseudo-weights to decision variables of PO solutions. These trained models can be used to create solution vectors for any new pseudo-weight vector. We demonstrate that this process can be used to cater to the typical needs of decision-makers, like quickly populating the PF, filling gaps, or extending the PF without further optimization. To deal with constrained problems, we propose an archiving strategy that can be used along with any non-dominated sorting-based MOEA. This archive provides an augmented training dataset that can be used to train ML models to predict the feasibility of newly created solutions before evaluating them, thereby avoiding wasteful evaluations of infeasible solutions.

While the previously discussed ML-based MCDM methods can be quick and convenient, they still require iterative solution evaluations in order to confirm the existence of newly created solutions. These MCDM attempts might not be fruitful if the desired solutions do not actually exist. We propose integrating the ML-assisted MCDM concepts into MOEAs as operators to induce confidence in the achieved PF by demystifying the flaws in the PO front. This operator can attempt to answer MCDM concerns, try and fix inherent flaws in the ongoing optimization process, and collect information regarding flaws that could not be fixed. This auxiliary information can be provided to the decision-maker to induce trust in the achieved solutions and can eliminate the need for post-optimal analysis.

ACKNOWLEDGEMENTS

I would like to thank my advisor, Prof. Kalyanmoy Deb, for his invaluable guidance, support and patience throughout my time at MSU. I would like to thank my committee members - Prof. Vishnu Boddeti, Prof. Erik Goodman, and Prof. Firas Khasawneh, for their guidance and valuable comments. I also would like to thank my lab-mates, Ritam Guha, Santoshkumar Balija, Abhiroop Ghosh, and Ahmer Khan, for their company, suggestions, and numerous discussions in the lab.

I would like to express my deepest gratitude to my family and friends for their unwavering support and encouragement throughout this journey. Their belief in me has been instrumental in making this achievement possible.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

ANN Artificial Neural Network

DM Decision Maker

EM(a)O Evolutionary Multi- or Many-objective Optimization

EMO Evolutionary Multi-objective Optimization

GPR Gaussian Process Regression

MCDM Multi Criterion Decision-making

M(a)OEA Multi- or Many-Objective Evolutionary Algorithm

MOEA Multi-objective Evolutionary Algorithm

MOO Multi-objective Optimization

NSGA Non dominated Sorting Genetic Algorithm

CHAPTER 1

INTRODUCTION AND MOTIVATION

The aim of a typical multi- or many-objective evolutionary optimization algorithm (MaOEA) is to find a dense and uniformly distributed set of non-dominated solutions that approximates the Pareto-optimal (PO) front well. This is then followed by a multi-criterion decision-making (MCDM) step where the decision-maker (DM) can choose a solution for further consideration. The DM must visualize, analyze and interpret the provided solutions to be able to select a desired solution. This step might also involve multiple decision-makers in which case the advantages and disadvantages of the obtained Pareto solutions must be discussed. Successful execution of both of these steps is necessary for the overall process to be successful. In this dissertation, we take a comprehensive approach to multi-objective optimization and decision-making and develop methods for convenient execution of these two steps.

1.1 Motivation for developing unique identifiers

Decades of research has led to a number of algorithms [1, 2, 3, 4, 5, 6, 7] that achieve the first step of finding a set of PO solutions reasonably well. Similarly, a number of successful MCDM methods [8, 9] as well as advanced visualization techniques [10, 11] have been developed to aid in the decision-making step. Decision-making preferences are usually independently utilized a priori, a posteriori, or an interactive manner [12, 13] with a multi-objective optimization algorithm. In some studies, RVs are arranged according to certain pre-defined monotonic decision-making preferences [14, 15, 16, 17, 18, 19], which are challenging to define. While the DM can visualize the solutions in the objective space, this might not always be fruitful as the PF can be of arbitrary shape due to non-domination properties and complexity of the objective functions and constraints. From a decision-making point of view, DMs look for the relative importance of objectives, in terms of their trade-offs, direct relevance to preferred objectives, etc., that each Pareto solution may offer. In this sense, a representation of the Pareto solutions in the objective space, which is typically done, may not always be most informative to DMs. On the other hand, EM(a)O algorithms already use a set of supplied reference vectors (RVs) to drive their search to locate a single optimal solution

for each RV. In a sense, each RV is truly employed as an *identifier* for a unique and specific Pareto optimal solution. An interesting thought here is the possible visualization of PO solutions in the identifier space, rather than in the objective space. The decision-making process can be performed conveniently if a suitable identifier space is used.

In the EM(a)O literature, it has been well demonstrated that the commonly used ideal-point-based RVs are not suitable to handle different shapes of PO fronts (for example, an inverted simplex arising in the Inverted DTLZ1 problem) – as many RVs do not contribute to any PO solution at the end, causing a waste of computations achieved by using all supplied RVs [20]. In such cases, there is a need to look for other identifiers which may be more appropriate for these problems. Several ideas in this direction have been explored in the literature [21, 22, 23, 24]. Similarly, pseudo-weights, introduced in [25], have been regularly used for the purpose of decision-making. This is usually done after optimization by computing pseudo-weights of all solutions and choosing the solution closest to the DM's requirements. This step might not be meaningful if no solutions are available near the provided pseudo-weight vector. This concept of finding a good distribution of solutions in the objective space and then hoping to find a desirable solution during decision-making might not always work as these identifiers possess different characteristics. A mismatch in compatibility between the optimization identifier and the decision-making identifier might cause issues for the DM. The above discussion indicates the merits of exploring different identifier spaces for representing PO solutions for achieving both tasks – optimization and decision-making. In this study, we discuss six such identifier spaces and highlight their advantages and disadvantages for optimization and decision-making. Also, we provide mathematical relationships between them and also with the objective space so that an easy transition from one identifier space to another in search of a better or a hybrid representation of PO solutions can be achieved. If decision-making must be done in a particular identifier space, it is prudent to provide the DM with a good distribution of solutions in this given space. We present modifications to NSGA-III that can help achieve a good distribution of solutions in any of these spaces. We also propose a method to obtain a good distribution of solutions in multiple identifiers if desired by the decision-maker.

1.2 Motivation for post-optimal machine learning

The solutions obtained by an MOEA should span the entire PO frontier and should convey all necessary information to the decision-maker (DM) in order to allow choice of the desired solution. The ideal scenario is to run the optimization algorithm once, collect all solutions, and select a solution from the computed Pareto set. However, in practice, this is seldom achieved due to inherent difficulties in multi-objective optimization. As envisioned via the No-Free-Lunch (NFL) theorem, even after continuous improvement in the optimization algorithms, we might end up with a set of ND-solutions that contains gaps, sparse regions and irregular distributions and might not have an algorithm that can solve all problems up to our expectations. It is also possible that the ND-set found by the MOEA does not span the entire PO frontier and that the algorithm has missed the actual extreme solutions. These situations arise for several reasons, such as the complexity of the problem, the complexity of the PO front, local attractors in the decision variable space, etc. Apart from all these unavoidable difficulties, multi-objective optimization is usually performed with a limited computational budget, such that only a finite number of solution evaluations is allowed. These issues can have a huge impact during the decision-making process, where a flawed approximation of the PO set can lead to a number of concerns and issues. For example, if there are gaps in the PO frontier, the DM might wonder if these are *real* gaps in the objective space or if they are failures of the MOEA. If the DM is interested in a sparse region of the PO front, it might be difficult to understand the PO front or choose a suitable solution. Similarly, the DM might be curious about the end points of the PO front and might want to gain confidence that these end points are in fact the extreme solutions of the PO front.

These concerns and issues are usually addressed by utilizing more compute and solution evaluations. For example, it is a common practice to run the MOEA several times and aggregate the solutions from these runs. If there are sparse regions in the ND-front, reference-direction based methods like R-NSGA-II [26] and R-NSGA-III [27] can be used to try to fill the deficiencies of the ND-set. Similarly, if there are gaps in the PO front, gap-finding methods [28] can be used to verify the existence of the gaps and potentially fill the gaps with new solutions. However, these methods

require more solution evaluations and can quickly become computationally prohibitive, causing decision-making to be iterative and slow. It might not be possible to perform re-optimization whenever the DM finds a flaw in the PO front. Hence, it is pragmatic to develop methods that can fix the flaws of the MOEA without re-optimization.

Theoretical studies [29] have revealed that, owing to optimality conditions of multi-objective problems, the PO-set might lie on a lower-dimensional manifold in the decision-variable space. This property can be exploited to find more PO solutions and has been a prominent research direction in the recent past [30, 31]. In this study, we present one such method where we exploit the regularity property of the PO front by learning the mapping between pseudo-weights [25] and a PO-set using machine-learning. We use various machine-learning techniques in order to learn the mapping between pseudo-weights and PO-set and show that these convenient unique identifiers can readily map to new PO solutions and can drastically speed up the decision-making process.

1.3 Motivation for finding identifier based solutions within MOO

While post-optimal MCDM and analysis is a valid option, we realize that these MCDM requirements are also valid concerns *during* the optimization itself. Instead of obtaining a PF and then performing analysis, we can streamline the process and attempt to fix some of the flaws of the PF before the optimization is completed. While most innovization studies explore the idea of using data to accelerate optimization and convergence [32, 33, 34, 35], we propose an ML based operator that can attempt to perform some MCDM tasks such as filling gaps and populating sparse regions during optimization. If these tasks are unsuccessful, this data can be provided to the DM along with the obtained PF as auxiliary data to answer subsequent questions regarding the achieved results. This could eliminate the need for post-optimal analysis.

1.4 Organization of the dissertation

This report is organized as follows: We explain some preliminary concepts in Chapter 2. In Chapter 3, we introduce different unique identifiers for representing Pareto-optimal points and discuss their advantages and disadvantages for optimization and decision-making. We also propose decision-making aware optimization methods here. In Chapter 4, we discuss results

from our proposed ML-assisted MCDM method. In Chapter 5, we propose and present results from integrated ML-assisted optimization. In Chapter 6, we summarize our findings and propose extensions and future directions.

CHAPTER 2

BACKGROUND

In this chapter, we present some necessary background concepts, such as multi-objective optimization and multi-criterion decision-making.

2.1 Multi-objective optimization (MOO)

Real-world optimization problems often consist of more than one conflicting objective that must be optimized simultaneously. Since the objectives are conflicting with each other, the aim of the optimization process is not just to find one *optimal* solution but rather a set of trade-off solutions. This prohibits us from using typical single-objective optimization algorithms and makes the optimization problem much more complex. Problems with three or more objectives are often referred to as many objective problems and are considered more difficult than their multi-objective counterparts.

A multi-objective optimization problem is defined as

$$\begin{aligned} &\text{Minimize } f_i(\mathbf{x}) && \text{where, } i \in (1, ..., M), \\ &\text{subject to } g_j(\mathbf{x}) \leq 0, && \text{where, } j \in (1, ..., J), \\ &\qquad h_k(\mathbf{x}) = 0, && \text{where, } k \in (1, ..., K), \\ &\mathbf{x}_d^L \leq \mathbf{x}_d \leq \mathbf{x}_d^U, && \text{where, } d \in (1, ..., D) \end{aligned}$$

A feasible solution is a solution that does not violate its constraints and variable bounds. That is, $\mathbf{x} : g_j(\mathbf{x}) \leq 0$, where, $j \in (1, ..., J)$ and $h_k(\mathbf{x}) = 0$, where, $k \in (1, ..., K)$.

Intuitively, owing to the multiple objectives, two solutions cannot be easily compared as they might be *better* in some objectives and *worse* in others. Hence, we use the concept of Pareto-domination (or domination) to ascertain the relationship between the two solutions.

Definition 1 (Domination) *A feasible solution* $\boldsymbol{a}$ *dominates another feasible solution* $\boldsymbol{b}$ *if* $f_i(a) \leq f_i(b)$ *holds* $\forall i \in (1, ..., M)$ *and* $\exists j \in (1, ..., M)$ *such that* $f_j(a) < f_j(b)$.

Given that we have the means to compare two solutions, we can also define optimality conditions based on this comparison operator, called Pareto optimality.

Definition 2 (Pareto-optimal) *A feasible solution* $\boldsymbol{a}$ *is Pareto optimal if there exists no feasible solution* $\boldsymbol{b}$ *that dominates* $\boldsymbol{a}$.

The aim of a multi-objective optimization algorithm is to find a set of Pareto-optimal solutions. The decision variables of the Pareto-optimal solutions are called the Pareto-set (PS) and the objective vectors are called the Pareto-frontier (PF).

2.1.1 Multi-objective Evolutionary Algorithms

Evolutionary algorithms are especially successful in MOO due to the need for finding multiple Pareto-optimal solutions. These population-based methods are tasked with achieving Pareto-optimal fronts that are both *converged* and *diverse*. Decades of research in multi-objective evolutionary algorithms (MOEA) have given rise to a number of successful algorithms, such as NSGA-II [4] (for multi-objective problems), and NSGA-III [2, 36] and MOEA/D [7] (for many objective problems) that perform reasonably well on a variety of multi- and many-objective problems. These algorithms use different mechanisms to maintain diversity amongst the non-dominated population. NSGA-II [4] is considered the state of the art for two-objective problems and uses a crowding distance based metric to allow survival of a diverse set of candidates for the next generation. NSGA-III [2, 36] uses reference directions to allow survival of diverse candidates, and MOEA/D [7] uses reference-direction based decomposition.

2.2 Multi-criterion decision-making (MCDM)

MOOs focus on achieving a converged and diverse set of ND solutions. However, given a set of non-dominated solutions that approximate the PO frontier, it is equally important to be able to identify a desirable solution. This process of choosing a solution from the PO front is dubbed multi-criterion decision-making and is a crucial step in the MOO process. This decision-making step becomes difficult and non-trivial when:

- there are more than 3 objectives to consider as visualization becomes difficult

- there are gaps in the PO front
- the PO front is in the form of a complicated shape

MCDM has been an important direction of research in the past. Some existing methods in MCDM are briefly discussed here.

2.2.1 Compromise programming

When objective priorities are known by the user, we use these priorities as scalarization or decomposition weights and try to find the closest solution from the available PO front. Commonly, scalarization methods like Achievement Scalarization Function (ASF) [37] are employed to retrieve the most relevant solution.

2.2.2 Pseudo-weights

The concept of pseudo-weights, defined in Equation 2.1, was proposed in [25] to provide representatives for non-dominated solutions. They are computed as the normalized distance to the worst solution in each objective.

$$w_i^{(k)} = \frac{\left(f_i^{\max} - f_i^{(k)}\right) \Big/ \left(f_i^{\max} - f_i^{\min}\right)}{\sum_{j=1}^{M} \left(f_j^{\max} - f_j^{(k)}\right) \Big/ \left(f_j^{\max} - f_j^{\min}\right)}, \tag{2.1}$$

These vectors are conveniently normalized, $w_i \in [0, 1]$, and act as *priorities* of the decision maker. For example, for a two-objective case, pseudo-weights of (0,1) would be the best solution for f_2 and worst solution for f_1. Similarly, pseudo-weights of (0.5,0.5) would mean equal priorities for both objectives and would yield solutions from the middle of the PO front. Usually, once ND-solutions are identified, we compute pseudo-weights for all solutions and select the one closest to the user's expectations.

2.3 Software implementations

Pymoo [38], developed at the Computational Optimization and Innovation Laboratory (COIN) has been used for all implementations of evolutionary computation presented in this dissertation. Apart from this, libraries such as Pytorch [39], GPytorch [40], Botorch [41], Optuna [42] and EZModel [43] have been used for machine learning implementations.

CHAPTER 3

UNIQUE IDENTIFIERS FOR OPTIMIZATION, VISUALIZATION AND DECISION-MAKING

In this chapter, we define *unique identifiers* and discuss in detail about six possible unique identifiers. We discuss the impact of the choice of these identifiers for optimization, visualization, and decision-making. We describe the identifiers used in this study along with the procedure to compute them and present guidelines on choosing identifiers based on optimization and decision-making requirements. We experimentally demonstrate the advantages and disadvantages of the identifiers on a few test problems. Some parts of this chapter are based on [44].

3.1 Literature survey

Several identifiers have been studied in the literature, mostly from the perspective of achieving a good distribution of solutions in the objective space. A modified variant of MOEA/D, called MOEA/D-IPBI [22], has been developed that computes the distances from the nadir point rather than the ideal point. This change allows the algorithm to capture a better distribution of solutions for inverted PF problems, like the Inverted DTLZ1. The authors argue that the effectiveness of ideal point RVs, evidenced by the success of RV based algorithms, is partly owed to the shapes of the PO fronts of the commonly used test problems. These PO fronts span the whole unit simplex and hence lead to a dense and uniform set of solutions when associated with ideal point RVs. If the projected PO front does not span the whole unit simplex, the ideal point RV based MOEAs will only associate a few RVs to the PO front, leading to a sparse set of objective vectors. For these problems, nadir point RVs can be highly effective as the whole set of RVs can capture the inverted PO front leading to a better distribution of solutions. Other diversity preservation methods, such as grids in polar coordinates [21, 45], projection-based identifiers (also called parallel distance projections) [24], radial projection [46], etc., have been considered for optimization. Some works have also proposed the use of multiple identifiers at these same time [23]. These alternative identifier based methods have also been adopted in other algorithms owing to their inherent advantages [47]. Even though these studies have incorporated these identifiers into MOEAs, they have mostly focused on

maintaining diversity in the objective space by catering to requirements of irregularly shaped PFs.

Recently, several methods have been proposed where the reference vectors adaptively conform to the shape of the PO front to capture solutions in irregular PFs [48, 49, 50]. These methods can achieve a great distribution of solutions for problems with irregular PF shapes that typical (uniformly initialized) identifiers fail to do. However, this also does not focus on the impact of identifiers on decision-making if the DM is to take place in the identifier space.

3.2 Unique identifiers for PO solutions

In this section, we describe the unique identifiers used in the study along with their existing applications in MOEAs. First, we define the requirements for *unique identifiers* for PO solutions. Every identifier for a PO solution needs to have the following properties:

1. **Unique mapping**: Every point on the identifier space should come from exactly one Pareto solution.

2. **Known bounds**: These identifier spaces should have known and interpretable bounds to ensure a proper understanding of the extent of the spread of Pareto solutions.

3. **Interpretable tradeoff**: The decision maker should be able to understand tradeoffs and the relationship between objectives from a relatively easier visualization of the distribution of points in the respective identifier space. This may provide greater flexibility to the DM to visualize and make decisions, which may not have been possible by directly investigating the objective vectors.

4. **Identify gaps and discontinuities**: Identifier spaces are intricately linked with objective space. Hence, any apparent gap or discontinuities on the obtained Pareto surface should also reflect the same way on the identifier space.

5. **Interpretable extremes**: The DM should understand which solutions in the identifier space correspond to the extreme solutions such that the overall tradeoffs between objectives can be

understood. Relationships derived in this study should enable a clear understanding of the linking of objective and identifier spaces.

In this study, we consider ideal point RVs (**r**), used in [2, 7], nadir point RVs (**R**) used in [22, 23], projection RVs (**P**), pseudo-weights (**w**) introduced in [25], and an angle based identifier (θ) used in [21, 45]. We also discuss the possibility of features from low-dimensional visualization techniques, like RadViz [51] used in [46], being used as plausible identifiers. First, we describe each of them and establish relationships between them.

3.2.1 Ideal point RV

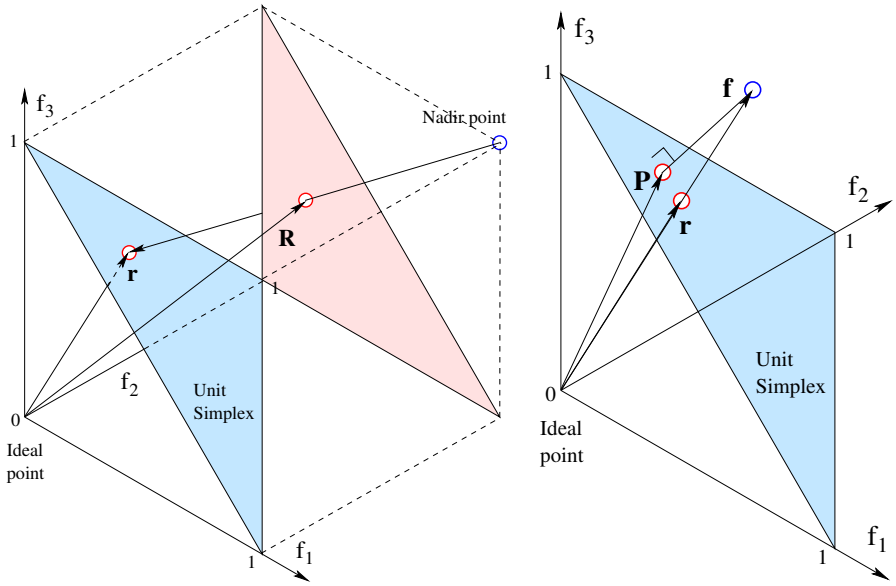
Reference vectors (RVs) originating from the ideal point and covering the entire positive orthant are commonly used in reference direction based MOEAs like NSGA-III [2], MOEA/D [7], RVEA [1], etc. For any **r**, $\sum_{i=1}^{M} r_i = 1$, makes each RV vector lie on a *unit simplex* forcing $r_i \in [0, 1]$. When a set of non-dominated (ND) or Pareto front (PF) solutions (**f**) are normalized so that normalized $\hat{f}_i \in [0, 1]$, they can be directly compared with **r** vectors. Since **f** can occur with arbitrary values depending on the underlying problem, a unique identifier (an appropriate **r**) can be used to represent an optimal objective vector:

$$r_i = \frac{\hat{f}_i}{\sum_{j=1}^{M} \hat{f}_j}, \quad \forall i. \tag{3.1}$$

Note that reverse mapping from **r** to a suitable $\hat{\mathbf{f}}$ is not possible unless the Pareto surface is known.

Figure 3.1b illustrates the identification of a Pareto solution ($\mathbf{f}^*$) with a RV (**r**). Notice that since the unit simplex covers the entire positive orthant, any point in it can be represented by a suitable RV having $r_i \in [0, 1]$ with $\sum_{i=1}^{M} r_i = 1$.

Thus, for an EMaO algorithm or for multi-criterion decision analysis (MCDA), **r** identifier vectors can be investigated for (i) evaluating the distribution ability of the algorithm, (ii) finding gaps and lean density of Pareto solutions on the PF, or (iii) directly relating decision-making preferences with **r** vectors. For a certain specific number of points, the Das and Dennis method [52] provides a perfectly uniform set of points on the unit simplex. Recently proposed Riesz Energy methods are used to find an arbitrary number of RVs on the unit simplex for a space of dimension



(a) Computation of **R** from **r**. (b) Projection RV (**P**) from **f**.

Figure 3.1 Ideal and nadir point RVs.

[53].

3.2.2 Nadir point RV

RVs originating from the nadir point have also been proposed as an alternative to ideal point RVs [22] where PBI from MOEA/D is reformulated to compute distance from the nadir point and is referred to as IPBI [22]. As discussed in Ishibuchi et al. [20], RVs from the nadir point provide advantages in capturing a good distribution of solutions when the PO front is inverted, like in the Inverted DTLZ1, or for deeply convex PFs.

As for ideal point-based RVs, any identifier must be able to represent any non-negative objective vector. A little thought will reveal that to achieve this, a nadir point RV (**R**) must satisfy $\sum_{i=1}^{M} R_i = (M-1)$. Thus, **R** vectors also fall on a linear hyperplane on the M-dimensional objective space, except that it does not intercept the objective axis at one, rather it intersects at $R_i = (M-1)$ for all i. But, there is a limit to this hyperplane that is enough to cover all non-negative objective vectors. As shown in Figure 3.1a, it is an inverted triangle for three-objective problems. For a normalized $\hat{\mathbf{f}}$ having nadir point at the vector of ones, the respective **R** is given as follows:

$$R_i = 1 - \frac{1 - \hat{f}_i}{M - \sum_{j=1}^{M} \hat{f}_j}, \quad \forall i. \tag{3.2}$$

After arriving at $\mathbf{r}$ from $\mathbf{f}$ using Equation 3.1, a transition between $\mathbf{r}$ and $\mathbf{R}$ can be obtained as follows (derived from Figure 3.1a):

$$R_i = \frac{M - 2 + r_i}{M - 1}, \quad \forall i, \tag{3.3}$$

in which $r_i \in [0, 1]$, causing $R_i \in [(M-2)/(M-1), 1]$. Interestingly, R_i are always non-negative for every RV from the unit simplex. Notice that for $M = 2$, both $\mathbf{r}$ and $\mathbf{R}$ are identical identifiers. Similarly, every $\mathbf{R}$ can also be represented by a specific $\mathbf{r}$:

$$r_i = (M - 1)R_i - (M - 2), \quad \forall i, \tag{3.4}$$

in which $R_i \in [0, 1]$, causing $r_i \in [-(M-2), 1]$. Interestingly, certain $\mathbf{R}$ vectors will transform to ideal point based RVs taking negative r_i values. The geometry, shown in Figure 3.1a, can be used to derive the above relationships.

Nadir point based RVs may be of importance for certain problems, as described above, and a modification of EMaO algorithms and decision-making approaches using $\mathbf{R}$ may be beneficial in those cases. The above equations can allow an easy way to update the existing algorithms for optimization and decision-making to freely investigate the points in both identifier spaces.

3.2.3 Projection RV

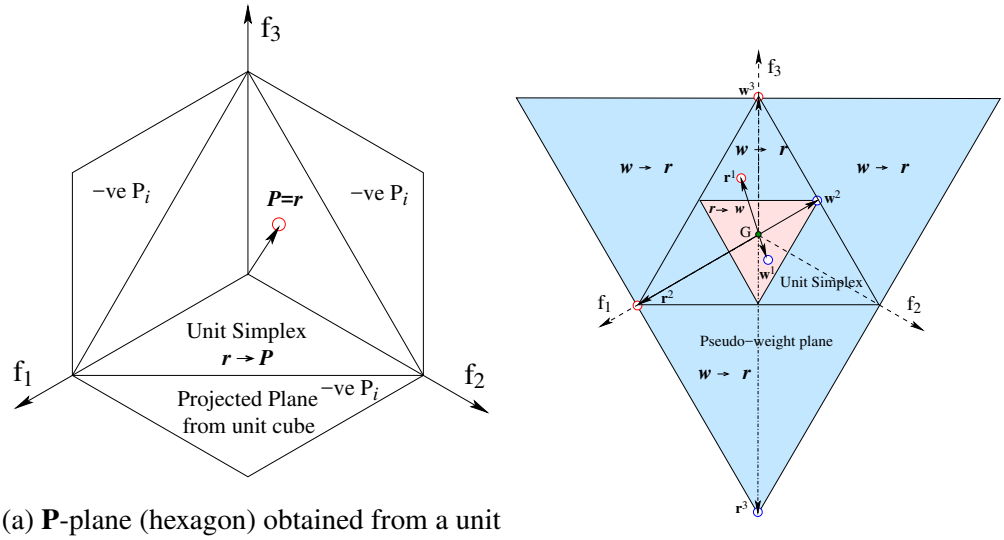
Another possible identifier is a vector on the extended unit simplex obtained by the orthogonal projection of objective vectors onto the unit simplex. This is illustrated in Figure 3.1b. A normalized $\hat{\mathbf{f}}$ point can be represented by a projection RV as follows:

$$P_i = \frac{1 + M\hat{f}_i - \sum_{j=1}^{M} \hat{f}_j}{M}, \quad \forall i. \tag{3.5}$$

The projection RVs ($\mathbf{P}$) can act as a middle ground compared to ideal and nadir point RVs. However, this also has the disadvantage at three or more dimensions that some regions in the unit hypercube can lead to negative RV values.

Some geometric considerations reveal that an $\mathbf{r}$ vector can be transformed into a $\mathbf{P}$, as follows:

$$P_i = r_i, \quad \forall i, \tag{3.6}$$



(a) **P**-plane (hexagon) obtained from a unit cube in 3D space.

(b) Relationship between **r** and **w**.

Figure 3.2 Projection RVs and pseudo-weights.

in which $r_i \in [0, 1]$, causing $P_i \in [0, 1]$ as well. Figure 3.2a shows that for three-dimensional problems, all non-negative points in the unit hypercube $\hat{f}_i \in [0, 1]$ cause the respective **P** to lie on the hexagonal extension of the unit simplex, making some P_i's negative, whereas r_i for all points in the unit cube is always non-negative.

Projection RVs have been studied briefly in the EMO literature[24, 54]. For both the EMO search process and MCDM activities leading to a better-distributed set of points on an orthogonally projected hyperplane of the PF hyper-surface, this RV may be used. The above conversion relationships may be useful for an easier update of an existing EMaO code or for identifying diversity and gaps on a projected plane on the objective space for decision-making purposes.

3.2.4 Pseudo-weight Vector

Pseudo-weights were introduced in [25] as a decision-making concept. These values are computed as the normalized distances of Pareto points from the worst objective values, as given in Equation 3.7 for a normalized **f** vector:

$$w_i = \frac{1 - \hat{f}_i}{M - \sum_{j=1}^{M} \hat{f}_j}, \quad \forall i. \tag{3.7}$$

Pseudo-weights are normalized to lie on the unit simplex plane ($\sum_i^M \mathbf{w}_i = 1$). The pseudo-weight w_i can be considered as an indicator of *importance* of the i-th objective. This is because an

objective value close to the worst value of the objective function will be assigned a zero pseudo-weight, indicating that the respective solution produces the least importance among all objectives. On the other hand, a solution having the minimum objective value for the i-th objective will make a large w_i value indicating that this specific solution indicates a high importance to the i-th objective. Similarly, a pseudo-weight value of (0.33, 0.33, 0.34) for a three-objective problem would mean equal priorities to all objectives and will correspond to a solution in the center of the three-objective PF.

Pseudo-weights can provide a number of advantages from the perspective of decision-making. Owing to the indicated relative importance of objectives, they can be directly related to DM's objective priorities and may be suitable for decision-making in cases where the geometry of the PF is not known or understood. This can also be helpful in cases where the DM does not care about the objective values themselves but only about the relative importances. In some sense, ideal point based RVs ($\mathbf{r}$) and pseudo-weights ($\mathbf{w}$) have contradictory meanings. Ideal point based RVs can be converted to a pseudo-weight vector, as follows:

$$w_i = \frac{1 - r_i}{M - 1}, \quad \forall i, \tag{3.8}$$

in which $r_i \in [0, 1]$ making $w_i \in [0, 1/(M-1)]$. Thus, pseudo-weights are always non-negative. Interestingly, both $\mathbf{r}$ points on the entire unit simplex map to the smaller inverted triangle as $\mathbf{w}$ points. The mapping of two $\mathbf{r}$ points ($\mathbf{r}^1$ and $\mathbf{r}^2$) to respective $\mathbf{w}$ points ($\mathbf{w}^1$ and $\mathbf{w}^2$) are shown in the figure.

The mapping of $\mathbf{w}$ to $\mathbf{r}$ is as follows:

$$r_i = 1 - (M - 1)w_i, \quad \forall i, \tag{3.9}$$

in which $w_i \in [0, 1]$ making $r_i \in [-(M-2), 1]$. For some $\mathbf{w}$ vectors ($w_i > 1/(M-1)$), certain r_i values can take negative value for $M > 2$, thereby making them fall outside the boundary of the unit simplex. Figure 3.2b shows the mapping of $\mathbf{w}$ points on the entire unit simplex goes to the bigger inverted triangle on $\mathbf{r}$ space with some points extending to the negative orthants ($r_i \in [-1, 1]$). For example, the point $\mathbf{w}^3$ maps to $\mathbf{r}^3$ point on the ideal point based RV space.

An interesting observation about the mapping is that the pair **r** and **w** lies on the opposite side of the centroid point G on the simplex, making G fall exactly in a ratio $(M-1):1$ from **r** and **w** points on the extended unit simplex. For a three-objective space, **w** is half the distance away from G as **r** is on the other side of G, and for bi-objective problems, they are equidistant from the centroid on either side. Thus, except the point G, every other point from one identifier space shifts to a different point in the other identifier space.

3.2.5 Angle based identifier

Our next identifier comes from a representation of an objective vector with angle vectors, rather than Cartesian vectors. A point in an M-dimensional space can be represented by polar coordinates – one distance value and $(M-1)$ remaining angles suitably measured from objective axes. This representation has been used for optimization in [21, 45]. If we want to represent points on a hyper-sphere (having a unit distance from the origin), then $(M-1)$ angles are sufficient. Conveniently, the non-domination principle will ensure this projection has only one solution at a point. Figure 3.3 shows the angles (θ) for a point (**f**) on the unit sphere in the 3D space.

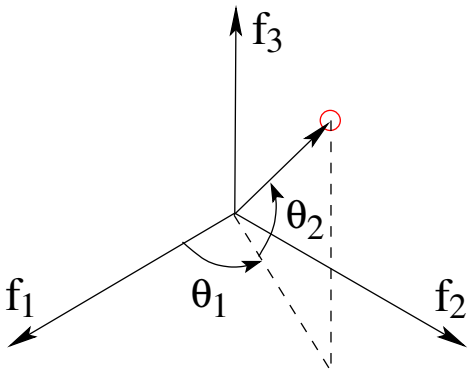


Figure 3.3 Angle vector for representing **f**.

For any **f** vector or a normalized $\hat{\mathbf{f}}$ vector, the respective θ identifier is given as follows:

$$q_i = \frac{\hat{f}_i}{\sqrt{\sum_{j=1}^{M} \hat{f}_j^2}}, \quad \theta_i = \tan^{-1} \frac{q_{i+1}}{\sqrt{\sum_{j=1}^{i} q_j^2}}, \quad \forall i. \tag{3.10}$$

The respective **r** vector on the unit simplex that will represent the same **f** vector is also marked in

the figure. Using geometry, the following conversion of $\mathbf{r}$ to an equivalent θ can be derived:

$$\theta_i = \tan^{-1} \frac{r_{i+1}}{\sqrt{\sum_{j=1}^{i} r_j^2}}, \quad \forall i, \tag{3.11}$$

in which $r_i \in [0, 1]$, causing $\theta_i \in [0, \pi/2]$. Similarly, the converse can be derived:

$$r_i = \begin{cases} \Pi_{j=1}^{M-1} \cos\theta_j, & \text{for } i = 1, \\ \sin\theta_{i-1} \Pi_{j=i}^{M-1} \cos\theta_j, & \text{for } i > 1, \end{cases} \tag{3.12}$$

in which $\theta_i \in [0, \pi/2]$, causing $r_i \in [0, 1]$.

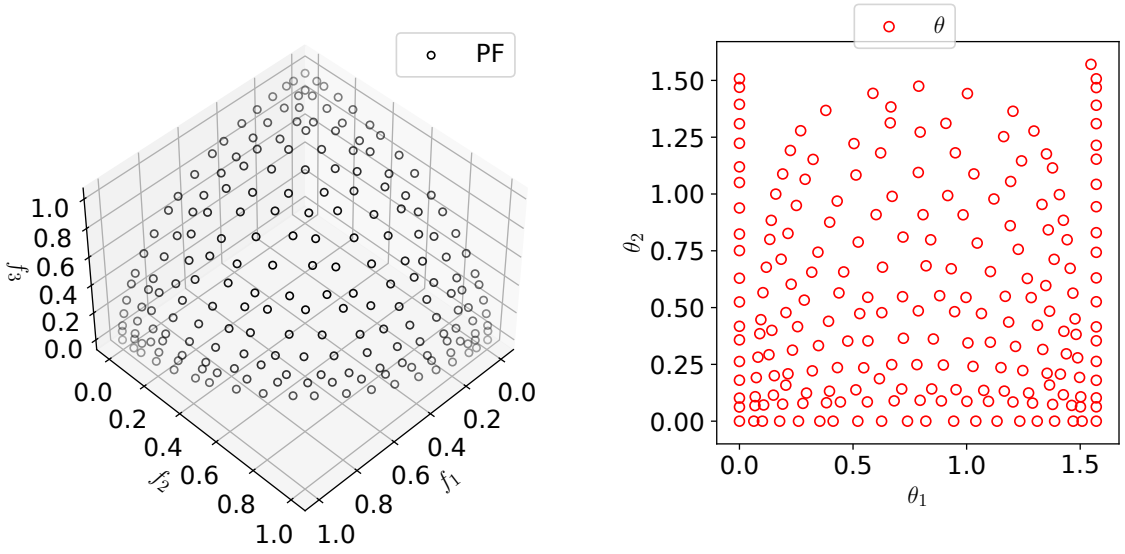
An example of visualizing tradeoffs with spherical coordinates is shown in Figure 3.4. Here, Pareto optimal solutions from three-objective DTLZ2 problem, as shown in Figure 3.4a, is transformed to spherical coordinates. Since the normalized Pareto front is already on a spherical surface, this PF is ideal to be represented by θ vectors. As can be seen from Figure 3.4b, the distribution in the θ space is almost uniform. The apparent gaps near θ_2 close to $\pi/2$ happen due to the existence of fewer solutions near to the $f_3 \approx 1$ part of the PF.

The angles can be useful from a convenient decision-making point of view. According to our angle convention, a small θ_i means points have minimized f_{i+1} well (evident from Equation 3.11). Thus, if f_{i+1} is the most important objective of all, then identifying Pareto solutions with small θ_i would be the direct and easiest way to analyze the PF. Also, good values of f_1 occur for large values of θ_1.

3.2.6 Low-dimensional Identifiers - RadViz coordinates

There exist many low-dimensional (namely, two-dimensional) visualization techniques, such as, radial basis visualization [51] (RadViz), star-coordinate visualization, and others. Although a specific objective vector $\mathbf{f}$ can be mapped to an respective unique low-dimensional ($< (M-1)$) coordinate system (say, $\mathbf{L}$) by using the above-mentioned techniques, the reverse (a unique low-dimensional $\mathbf{L}$ mapped to a unique M-dimensional $\mathbf{f}$) must also exist for it to be a usable identifier space. Let us discuss this further with respect to the popular radial visualization (RadViz) technique.

In RadViz ($RViz$), any $\mathbf{f}$-vector is mapped to a 2D point (x, y) inside a regular simplex formed with M anchor points lying on a 2D circle (having center at origin and radius one). For minimization



(a) Pareto optimal front. (b) Angle vectors.

Figure 3.4 Pareto-optimal front and angle identifiers of DTLZ2.

problems, objective values with smaller value should be given more importance, hence normalized objective vector $\hat{\mathbf{f}}$ should be first converted into $\hat{\mathbf{f}}' = 1 - \hat{\mathbf{f}}$ to compute RadViz coordinates for a more convenient decision-making purpose. With the 2D coordinate system (x-y) having origin at the center of the circle, the location of j-th anchor point on the circle can be specified as $(\cos(\alpha_j), \sin(\alpha_j))^T$ making an angle $\alpha_j = (j-1)2\pi/M$ radians from the positive x-axis. Thus, the anchor point 1 (representing f_1) always lies on the intersection of positive x-axis with the circle and the angle between two consecutive anchor points (objectives) is $2\pi/M$ radians. The exact unique location of (x, y) on the Radviz space for a given $\hat{\mathbf{r}}$ is given below for any M:

$$x = \frac{\sum_{i=1}^{M} \hat{f}_i' \cos(\alpha_i)}{\sum_{j=1}^{M} \hat{f}_j}, \qquad y = \frac{\sum_{i=1}^{M} \hat{f}_i' \sin(\alpha_i)}{\sum_{j=1}^{M} \hat{f}_j}. \tag{3.13}$$

By denoting $\hat{r}_i = \hat{f}_i / \sum_{j=1}^{M} \hat{f}_j$, it is realized that each $\hat{\mathbf{r}}$ vector falls on the unit simplex and above equations can be written as:

$$x = \sum_{i=1}^{M} \hat{r}_i \cos(\alpha_i) = \mathbf{c}^T \hat{\mathbf{r}}, \qquad y = \sum_{i=1}^{M} \hat{f}_i \sin(\alpha_i) = \mathbf{s}^T \hat{\mathbf{r}}, \tag{3.14}$$

where $\mathbf{c} = (\cos(\alpha_1), \cos(\alpha_2), \ldots, \cos(\alpha_M))^T$ and $\mathbf{s} = (\sin(\alpha_1), \sin(\alpha_2), \ldots, \sin(\alpha_M))^T$.

It is clear that since α_i values are fixed, any $\hat{\mathbf{r}}$ vector on the unit simplex is mapped to a (x, y) vector in the RadViz space. As a special case, note that for $M = 2$, $c = (1, -1)^T$ and $s = (0, 0)^T$,

making $y = 0$. A two-dimensional $\hat{\mathbf{r}}$-vector maps to $x = \hat{r}_1 - \hat{r}_2$ in the Radviz space. Also, $\hat{\mathbf{r}}$-vector can be obtained from a x value as $\hat{r}_1 = (1 + x)/2$ and $\hat{r}_2 = (1 - x)/2$. However for $M \geq 3$, the following is true:

Theorem 1 *A specific (x, y) point in the Radviz space results from a unique $\hat{\mathbf{r}}$ vector for $M = 3$, whereas for $M > 3$, it can result from different $\hat{\mathbf{r}}$ vectors on the unit simplex.*

Proof 1 *In addition to Equations 3.14, there is a third equation that every $\hat{\mathbf{r}}$ must satisfy: $\sum_{i=1}^{M} \hat{r}_i = 1$. These three equations can be written for a given (x, y) point, as follows:*

$$\begin{bmatrix} \mathbf{c}^T \\ \mathbf{s}^T \\ (1, 1, \ldots, 1) \end{bmatrix} \hat{\mathbf{r}} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}, \tag{3.15}$$

$$A\hat{\mathbf{r}} = b. \tag{3.16}$$

Note that for $M = 3$, the above equation can be uniquely solved: $\hat{\mathbf{r}} = A^{-1}b$, revealing

$$\begin{aligned} \hat{r}_1 &= (2x + 1)/3, \\ \hat{r}_2 &= (-x + \sqrt{3}y + 1)/3, \\ \hat{r}_3 &= (-x - \sqrt{3}y + 1)/3. \end{aligned} \tag{3.17}$$

But for $M > 3$, the above linear system of equations is under-determined. There are only three equations, but M unknowns ($\hat{r}_i$). For different $(M - 3)$ independent $\hat{r}_i$ values, three dependent $\hat{r}_i$ values can be obtained satisfying above three equations. Hence, there may exist multiple $\hat{\mathbf{r}}$-vectors for a specific (x, y) point for $M > 3$.

For the above reason, despite the unique mapping from $\mathbf{f}$ to low-dimensional space, Radviz identifier and any other low-dimensional identifiers cannot be used as a viable identifier space for more than three objectives.

3.2.7 Advantages and Disadvantages of Identifiers

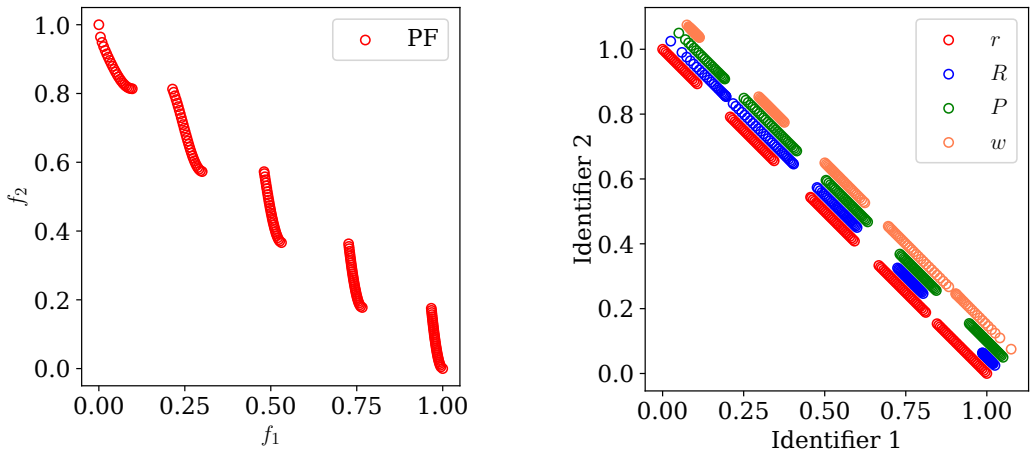
We evaluate and compare the identifiers based on the above properties. For this purpose, we analyze and visualize PO solutions of several test problems. Typically, these test problems are solved by ideal point based RVs. Here, we show their representation on alternate identifiers and discuss their suitability for easier visualization and decision-making purposes. RV-based methods typically struggle in dealing with inverted problems, hence they are considered here.

Lastly, disjointed PO fronts are also included to investigate whether any specific identifier is more suitable to identify gaps and discontinuities in their PO fronts. For each problem, we collect up to 210 PO solutions (based on ideal point based reference direction survival, closest to RVs) from NSGA-III runs for the purpose of our analysis. We independently compute the identifiers from $\mathbf{f}$ vectors.

3.2.7.1 Effect of Shape of PO Front

2D example Here, we discuss each identifier's properties from the context of a specific two-objective problem – ZDT3. All identifiers are arbitrarily shifted from each other in order to show them clearly in Figure 3.5b, while the normalized PO front is shown in Figure 3.5a. In this plot, we clearly see the effect of different transformations of the objective vectors. In two-dimensions, both ideal point RVs and nadir point RVs intersect on the same hyperplane (unit simplex line here). However, in higher dimensions, ideal point RVs and nadir point RVs will not lie on the same hyperplane. While the extreme solutions correspond to the same solutions, these solutions are shifted based on the perspective of the ideal or nadir point. Hence, the amount of gaps between the islands are reversed for these identifiers. On the other hand, projection RVs have similar amounts of gaps between the islands as they are projected perpendicularly to the unit simplex hyperplane. Contrary to RV based approaches, pseudo-weights provide a different perspective and have the islands reversed compared to ideal point RVs. Here, the solutions at the maximum value of w_1 correspond to the best value of f_1 at the top left corner of Figure 3.5a.

Typical Multi-objective Test Problems Typical test problems are problems where using ideal point RVs lead to optimal distribution in the objective space. We can easily observe this to be true



(a) PF

(b) Identifier space

Figure 3.5 Normalized PF and identifiers for ZDT3 problem. Identifiers lie on the equi-angled line from both axes.

for the DTLZ1 problem, as shown in Figure 3.6a. Here, the normalized PO solutions are present on the unit simplex and they map to a uniformly filled unit simplex hyperplane in the ideal point RV (**r**) identifier. A similar observation can be made about the projection RV (**P**) identifier as well. However, this is not the case with nadir point RV (**R**), pseudo-weight (**w**), and angle (θ) identifiers. Nadir point RVs do not span the entire hyperplane, as discussed in subsection 3.2.2 and forms a smaller triangle in the middle of the hyperplane. The same issue is also present with the **w** identifiers, as they also do not span the whole unit simplex and instead form a smaller inverted triangle inside the unit simplex. The inverting aspect is further explained by selecting an inverted triangle of solutions for visualization, shown as a gray surface on Figure 3.6a. These points are marked in blue color in other identifiers (Figures 3.6b-3.6e). Contrary to the situation with other identifiers, **w** vectors have inverted the blue triangle of solutions.

Importantly, if any of these identifier spaces are considered for visualization and decison-making, the user would be expecting points on the entire bounded space marked by the triangular boundary (for 3D). Except for **r** and **P**, other identifier spaces show gaps insider the triangular bounded space. This can cause an apparent concern to the decision-maker. Without a proper understanding of the properties of the various identifier spaces and their bounds, the DM might

suspect that this could be an indication that all possible PO solutions are not found by the EMO algorithm, while clearly for the DTLZ1 problem this is not the case. Any shape that spans the whole unit simplex with respect to the ideal point RVs cannot cover the area with nadir point RVs and pseudo-weights. However, if an EMO-created points show gaps inside the respective triangular regions for any of the four identifier spaces, it is expected that the Pareto front has gaps or that EMO was unable to find a well distributed set of points.

As discussed in Section 3.2.5, uniformly distributed objective vectors will lead to some sparse regions and gaps in the angle vector space. This is seen in Figure 3.4b. This sparsity and these gaps are pathologies of the coordinate transformation and must be kept in mind by the decision maker.

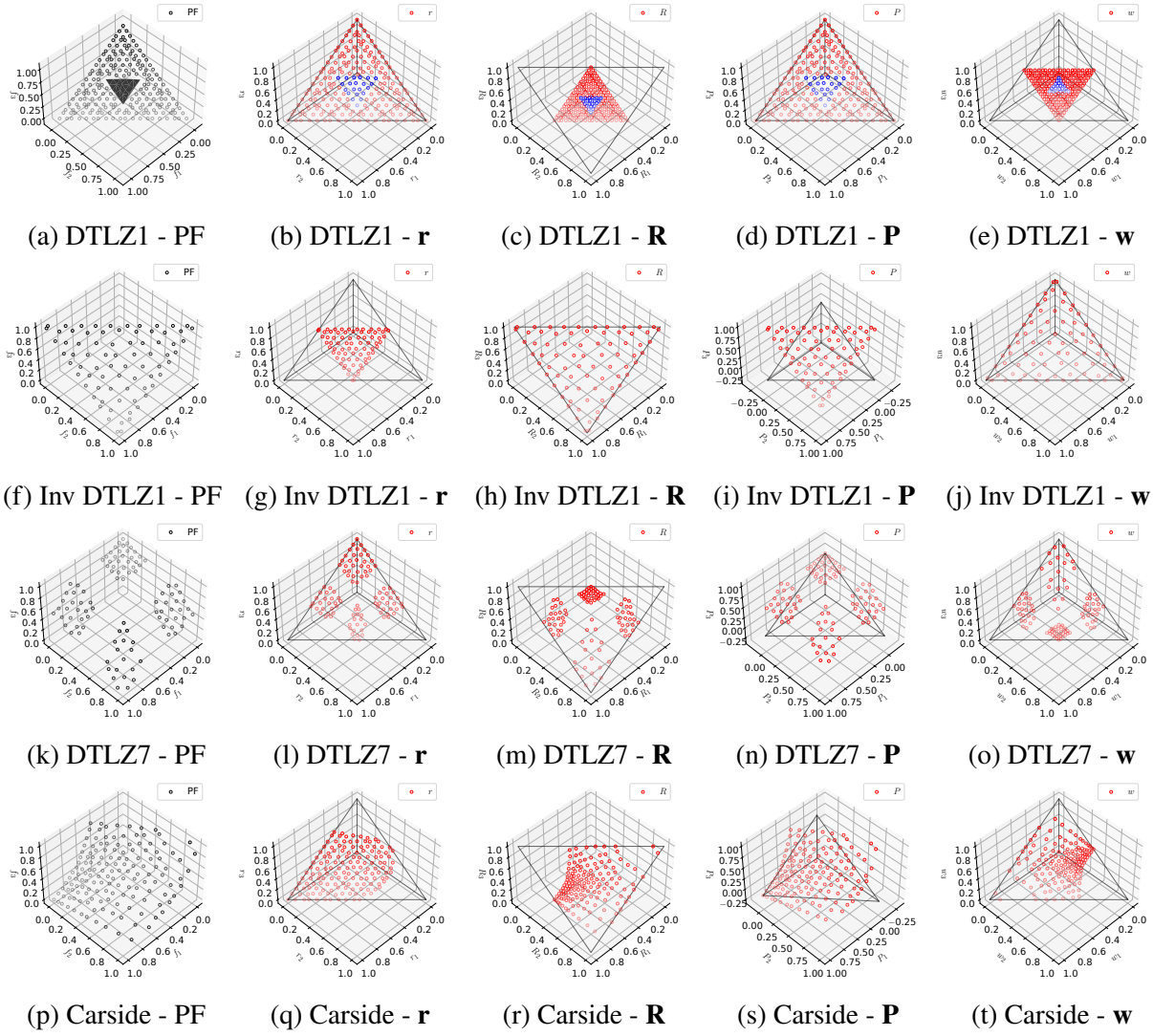
Inverted PF problems Inverted PO front problems, like Inverted DTLZ1, show contrasting behavior in different identifier spaces. The PO front and the corresponding identifier values for the inverted DTLZ1 are shown in Figures 3.6f and 3.7b, respectively. Here, ideal point RVs only form a small inverted triangle on the unit simplex hyperplane. This property is reflected in the fact that ideal point RV based algorithms do not find sufficient solutions in the PO front of the inverted problems, as discussed in [20]. A similar effect is seen for projected RV space, as they also form an inverted triangle but are bigger and are spilling out of the first orthant with negative values. On the other hand, nadir point RVs cover the hyperplane completely. This can be vital from a decision-making perspective as an incomplete unit simplex hyperplane, as in the case of ideal point RVs ($\mathbf{r}$), can be misleading regarding the completeness of the PO front. On the other hand, the nadir point RVs show a much clearer picture regarding the fact that the PO front, as in the case of inverted DTLZ1, covers the whole hyperplane.

A remarkable feature of the pseudo-weight identifier, as seen in Figure 3.6j, is that the pseudo-weights of the inverted DTLZ1 PO front completely cover the unit simplex hyperplane. Since pseudo-weights are normalized by definition, the extreme solutions correspond to extremes of the unit simplex hyperplane. The mirroring effect of pseudo-weights conveniently fills the hyperplane here and can be of advantage during decision-making.

Angle based identifier spaces for DTLZ7 and inverted DTLZ1 are shown in Figure 3.7. While

four disjointed parts of the PO front are clear from the angle identifier space for DTLZ7, the inverted nature of the inverted DTLZ1 is also somewhat clear from the figure.

These properties of different identifiers reveal an interesting fact – a specific shape of a PO front can be dealt with better by a specific identifier. Thus, if some idea of the shape of a PO front is known beforehand, EMO algorithms and ensuing decision-making can be performed using a specific identifier well suited to that PO front shape.



(a) DTLZ1 - PF (b) DTLZ1 - **r** (c) DTLZ1 - **R** (d) DTLZ1 - **P** (e) DTLZ1 - **w**

(f) Inv DTLZ1 - PF (g) Inv DTLZ1 - **r** (h) Inv DTLZ1 - **R** (i) Inv DTLZ1 - **P** (j) Inv DTLZ1 - **w**

(k) DTLZ7 - PF (l) DTLZ7 - **r** (m) DTLZ7 - **R** (n) DTLZ7 - **P** (o) DTLZ7 - **w**

(p) Carside - PF (q) Carside - **r** (r) Carside - **R** (s) Carside - **P** (t) Carside - **w**

Figure 3.6 Identifiers for typical, inverted and disconnected PO front problems.

Disconnected PFs An important purpose of visualization in the identifier space is to analyze apparent gaps and understand flaws, if any, with the EMO algorithm or in the PO front. The

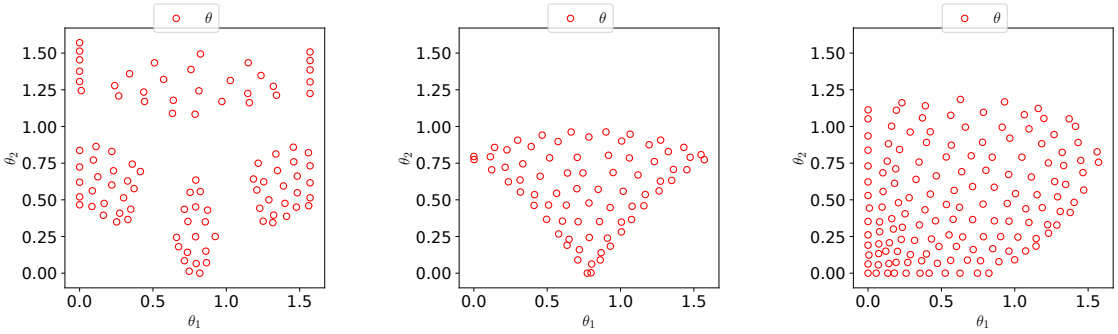
disconnected PO front along with the identifiers of the DTLZ7 problem are shown in Figures 3.6k-3.6o. The normalized PO front (Figure 3.6k) has four parts with different sizes. All the identifiers form four yet differently-sized islands. For example, the island close to the maxima of f_1 and f_2 forms the smallest island with ideal point RVs and yet forms the largest island with nadir point RVs. Projection RVs forms relatively bigger and well-distrubuted islands that spill over to negative values. The nadir based RV plot skews the distribution, providing a higher density of points near the top apex of the triangular PO front.

Pseudo-weights form differently sized islands conveying different trade-offs for the islands. This demonstrates that the rate at which the DM can navigate through the solutions for changing priorities is different for different islands. This could be important information to be conveyed to the DM. Similarly, angle based identifiers, shown in Figure 3.7a also demonstrate different sizes and densities, showing different trade-offs as we navigate the $\theta_1 - \theta_2$ space.

Gaps and islands in the objective space always correspond to gaps and islands in the identifier space. However, the relationship between solutions in objective space and identifier space might not be the same. The identifiers can have different densities of solutions, changing the interpretation of the trade-off between solutions. Solutions in apparent gaps should be sought carefully while considering the intricacies of each identifier space. Expecting solutions everywhere on the identifier hyperplane might not be feasible as many PO fronts cannot have solutions everywhere. For example, with angle-based identifiers, as shown in Figure 3.7a on DTLZ7, one of the four disconnected islands seems to come from a wide variety of θ_1, but $\mathbf{r}$ or $\mathbf{P}$ identifiers find fairly uniform distributions of points within each island. For the inverted DTLZ1 problem, the distribution of points in the θ space is more or less uniform. Still, the points appear in a bounded region, indicating that not the entire θ space is covered by PO solutions.

3.2.8 Decision-making in higher dimensions

Identifiers such as ideal point RVs, nadir point RVs and projection RVs retain the geometric properties of the PO front and these identifiers do not reduce the dimension of the identifier space. This allows the DM to choose desirable solutions, identify gaps and interpret them spatially. If the



(a) DTLZ7 (b) Inv DTLZ1 (c) Carside

Figure 3.7 Angle Vectors.

DM knows the approximate location of the desired solution or can understand the geometry of the whole PO front, RV-based identifiers can be directly used. If the DM does not know the geometric location of the desired solutions and can only navigate the PO front using priorities and trade-offs, pseudo-weights or angle based identifiers are better suited. However, visualizing solutions for decision-making is difficult with more than 3-objectives as the *geometry* of the PO front cannot be taken into account anymore. Though visualization techniques like parallel coordinate plots (PCP), RadViz [51], RadViz3d [10], etc., are available, these methods do not convey the spatial information regarding the PO fronts. While gap-finding methods [28] have been developed to perform at higher dimensions, it can be difficult to interpret the meaning of these gaps. However, identifiers such as pseudo-weights and angle-based identifiers are agnostic to the geometry and capture the essential trade-off information necessary to choose a solution.

Pseudo-weights always sum to one, and the values convey information regarding the priorities of the objectives. Even if the whole PO front cannot be visualized due to the dimensionality, merely providing priority values can help the DM in finding a desirable solution or region of interest. PCP plots of pseudo-weights can aid in DM at any number of dimensions. Angle based identifiers can provide similar advantages as these values convey specific trade-off information. Increasing or decreasing certain θ_i values can help the DM navigate the PO front without understanding the shape of the PO front. Hence, when we are dealing with higher numbers of objectives, pseudo-weights or angle-based identifiers can be a better choice for decision-making.

3.3 Choice of identifier space based on decision-making preference

In the following, we highlight the relevance of choosing a specific identifier space during optimization to achieve a certain decision-making purpose.

1. **Ideal point RV:** This identifier is desirable to use to to identify knee-like solutions with more dense points, or if the DM is most interested in compromise (middle part) in a convex-like PF, or if the DM is interested in obtaining a uniform distribution for a concave-like PF.

2. **Nadir point RV:** This identifier is ideal to identify the middle part of a concave-like PF, or if the DM is interested in a uniform distribution on a convex-like PF.

3. **Projection RV:** This identifier is ideal to achieve a uniform distribution on a linear-like PF, or if the DM is interested in points on the entire PF with well spread-out points, perhaps to fit a surrogate Pareto surface.

4. **Pseudo-weight vectors:** This identifier allows the DM to obtain points corresponding to relative importance factors of objectives (for example, a weight vector (0.75, 0.25) indicating that f_1 is three times more important than f_2).

5. **Angle vectors:** This identifier is capable of providing solutions close to a specific objective axis (say f_i), meaning solutions having better values of all objectives except f_i.

These guidelines can be used by decision-makers to choose the right identifier for their use case.

3.4 Decision-making aware optimization

The varying properties of these identifiers take us to some important questions:(i) Can different identifiers described here be used as reference points (or directions) for EMaO algorithms to produce different DM-preferred distributions, (ii) Which MoEAs would allow different identifiers to be implemented without much change in their existing procedures? (iii) Can we use a combination of identifiers for optimization?

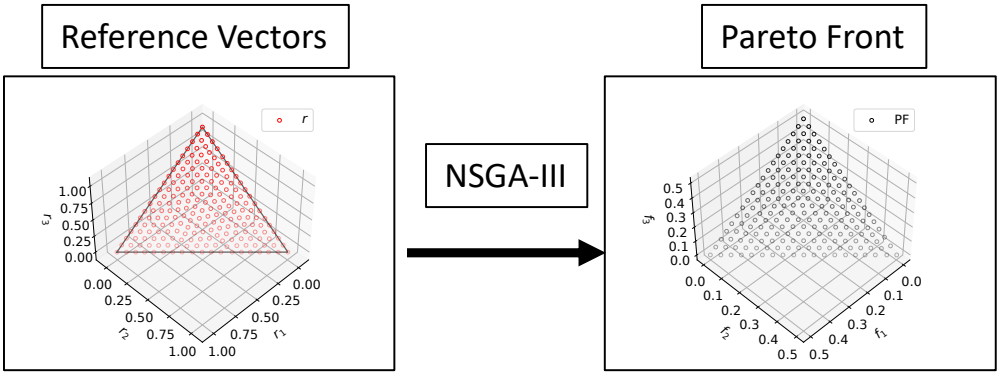
In this study, we consider three such alternative approaches: Customized NSGA-III where RVs are replaced with alternative identifiers; Converted RV NSGA-III where uniformly distributed

identifiers are converted to ideal point RVs and are then used in NSGA-III; and a third alternative in which a combination of identifiers is used to create a set of reference identifiers that can then be used during optimization. The three procedures are pictorially demonstrated in Figure 3.8 and are explained in detail in this section.

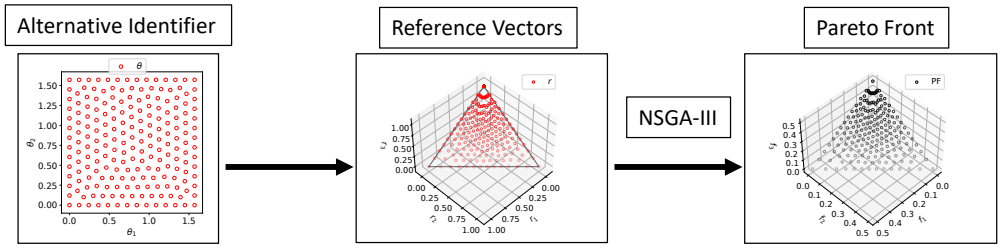
We run all versions of the proposed algorithms on ten 3-objective problems, namely - DTLZ1 [55], DTLZ2, DTLZ7, Inverted DTLZ1, Inverted DTLZ2, WFG2 [56], MaF3 [57], C2DTLZ2, Carside impact [58] and Crashworthiness [59]. Every algorithm is run for 200,000 solution evaluations with a population size of 200 and is repeated 31 times with different random seeds. We consider three objective problems in this study to discuss the implications during visualization but many of these ideas can be extended to more objectives. We quantify the effectiveness of different methods by computing hypervolume. We also present an IGD [60] metric computed in the identifier space, which we call as IGDI. We use a set of reference identifiers, generated using Riesz Energy method, to compute IGD in the identifier space. We perform Wilcoxon rank-sum tests to verify statistically significant differences in the metrics.

3.4.1 Alternative identifier spaces for optimization

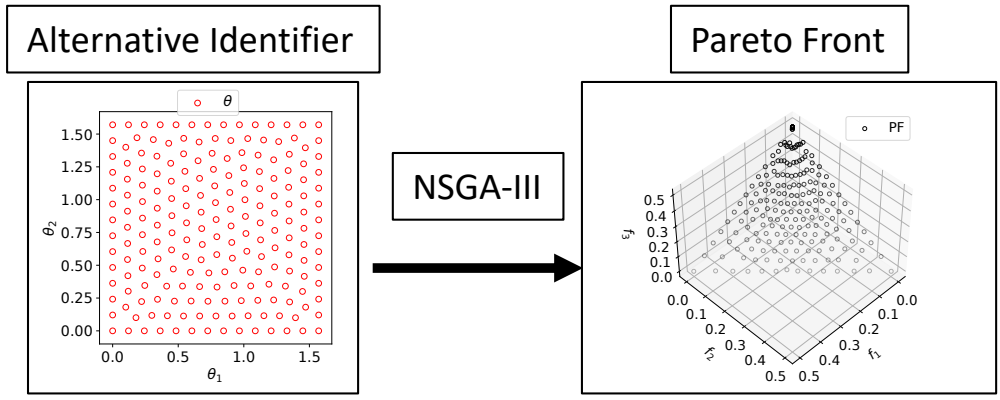
As demonstrated in the above sections, different identifiers produce different distributions in their respective spaces. Moreover, some identifier spaces can be directly associated with decision-making priorities to the liking of the DM. It is possible that the DM is comfortable with or prefers one or more of these identifiers. Depending on the DMs preferences regarding geometric or trade-off interpretation, prior knowledge of shape of PO front, etc., the DM might want to visualize and analyze PO solutions using a certain identifier. PO solutions identified using ideal point RVs might not give a good distribution in every identifier's space. Hence, it is prudent to provide the DM with a *good* distribution of solutions in the identifier where decision-making takes place. A simple yet effective solution to this issue is to use these identifiers within an EMaO algorithm, instead of using the standard ideal point or nadir point based RVs. This can lead to a different distributions of solutions in the objective space and yet may produce a better distribution of solutions in the space of identifiers.



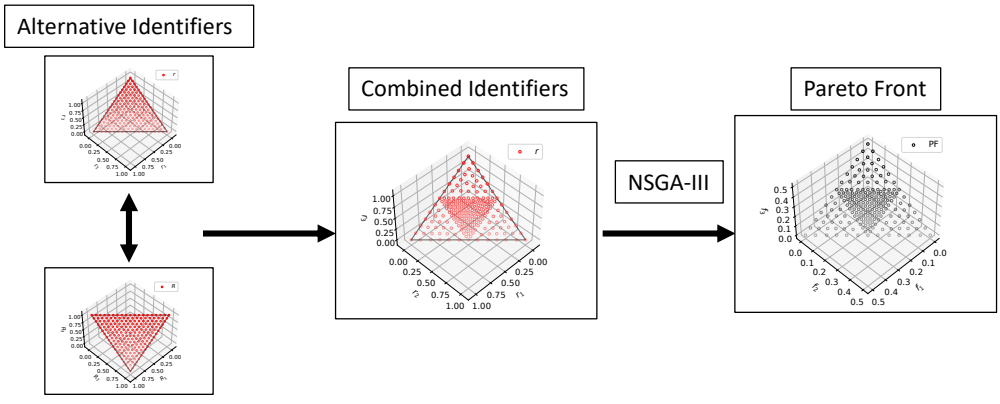
(a) Baseline



(b) Converted RV NSGA-III



(c) Customized NSGA-III



(d) Combined RVs NSGA-III

Figure 3.8 Optimization using alternative identifiers.

The next relevant question is whether an MoEA would allow an easier way to replace its niche-creating methods with an alternate identifier discussed in this study. As can be realized that most existing MOEAs use the **r** (RV from the ideal point) identifier. A few studies have used **R**[22] and some others with both **r** and **R** [23]. MoEA/D uses two distance values, d_1 and d_2, values for any **f** for a specific **r** vector. This can be readily swapped for **R** by computing the distance from the nadir point instead of the ideal point, as demonstrated in [22]. Similarly, variants of MOEA/D with angular distance [21, 45] have been proposed where diversity preservation is performed in the angle space. Some works with projection based RVs have been proposed where specialized methods are used to preserve solutions in the projected space [24]. Radial projection (RadViz) based MOEAs [46] have been proposed in which a grid-based diversity preservation method has been used in radially projected space.

In this study, we modify NSGA-III's survival and niche-preservation mechanism to work in the projected space instead of the objective space, and call this Customized-NSGA-III, as shown in Figure 3.8c. This minor modification allows us to use different projection functions that map to different identifiers and ensure a good distribution of solutions in the respective identifier spaces. In NSGA-III, the *niching* operation involves the association of normalized objective vectors to reference vectors. This is done by computing the perpendicular distance of the objective vector from all RVs. In our proposed Customized NSGA-III, we project the points to the identifier space and compute distances in the lower dimensional space. For example, for the nadir RV (**R**) case, the objective vectors are projected to the (M-1) hyperplane and then *associated* with RVs. Conveniently, NSGA-III performs non-dominated sorting that itself eliminates one of the dimensions of the objective space. This helps us in replacing the association operator's Euclidean distance with a projected distance metric. As long as we have a well-distributed set of *reference identifiers* in the projected space, they can be used to drive optimization and achieve a distribution as good as that of the reference identifiers.

We consider **R**, **P**, **w**, θ and $RadViz$ ($RViz$) as alternative identifiers and run Customized NSGA-III on several test and real-world problems. We use the Reisz Energy method [53, 61]

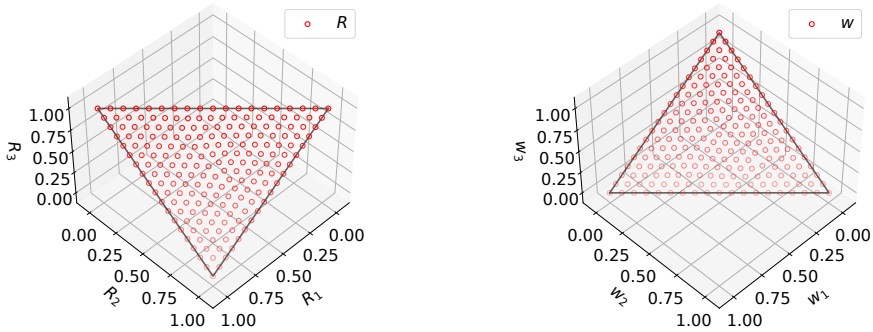
to create well-distributed reference identifiers in their respective spaces. Examples of reference identifiers created in different spaces are shown in Figure 3.9. For each of these methods, boundary points are generated and fixed and then internal points are generated and optimized to be well distributed in each space. These methods can be readily extended to more objectives as well. We can observe uniformly distributed *reference identifiers* filling boundaries of different shapes and ranges.

Pareto fronts achieved using different identifiers for several problems are shown in Figure 3.11. The median hypervolumes for these runs are shown in Table 3.1. PFs shown in Figure 3.11 show similar observations as mentioned in the previous sections - with the difference that these are aimed at achieving a good distribution of solutions in their respective identifier spaces, hence leading to consequent differences in objective values. It is important to understand that the HV values reflect the distribution of the solutions in the objective space and hence are directly influenced by the compatibility between the problem and the identifier used for optimization. Figure 3.10 shows PFs and corresponding identifiers of Carside impact problem solved using a set of different identifiers. The key observation here is that since each was solved using the identifier itself, all identifier plots (Figures 3.10f-3.10j) show a well distributed set of identifiers as compared with Figure 3.6. This comes at the cost of ill-spaced solutions in the objective space but is not of concern if the DM's preferred choice of MCDM domain is the identifier-space instead of the objective values.

Table 3.1 Median hypervolumes for different versions of NSGA-III with alternative identifiers.

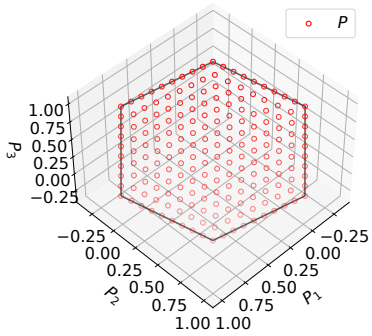
	Problem	M	Customized RV NSGA-III					Converted RV NSGA-III		Combined RV NSGA-III		NSGA-III
			R	θ	*RViz*	**w**	**P**	**R**	θ	**r-R**	**P-r**	
1	DTLZ1	3	9.314E-01	9.579E-01	9.108E-01	9.309E-01	9.490E-01	8.324E-01	9.582E-01	9.579E-01	9.522E-01	**9.629E-01**
2	DTLZ2	3	5.702E-01	5.894E-01	5.658E-01	5.705E-01	5.847E-01	4.653E-01	5.906E-01	5.912E-01	5.845E-01	**5.925E-01**
3	C2DTLZ2	3	5.034E-01	5.176E-01	5.030E-01	5.030E-01	5.149E-01	3.514E-01	5.210E-01	**5.260E-01**	5.221E-01	5.222E-01
4	InvertedDTLZ1	3	**2.190E-01**	1.967E-01	2.189E-01	**2.190E-01**	2.089E-01	2.192E-01	1.970E-01	2.094E-01	2.097E-01	1.941E-01
5	InvertedDTLZ2	3	**7.079E-02**	5.651E-02	7.064E-02	7.048E-02	6.341E-02	5.805E-02	5.297E-02	5.841E-02	6.444E-02	5.350E-02
6	WFG2	3	1.052E+00	1.071E+00	1.051E+00	1.052E+00	1.070E+00	9.437E-01	1.071E+00	1.071E+00	1.075E+00	**1.076E+00**
7	MaF3	3	1.090E+00	1.107E+00	1.079E+00	1.086E+00	1.105E+00	1.044E+00	1.107E+00	1.107E+00	1.108E+00	**1.111E+00**
8	DTLZ7	3	4.138E-01	**4.220E-01**	4.127E-01	4.119E-01	**4.207E-01**	3.382E-01	4.169E-01	4.180E-01	**4.196E-01**	4.188E-01
9	Carside	3	9.044E-01	9.062E-01	8.996E-01	9.043E-01	9.113E-01	8.541E-01	9.117E-01	9.206E-01	**9.243E-01**	9.098E-01
10	Crashworthiness	3	8.937E-01	**9.080E-01**	8.958E-01	8.956E-01	9.036E-01	7.710E-01	9.023E-01	8.997E-01	9.054E-01	**9.083E-01**

As established earlier, nadir point RVs and pseudo-weights cover the entire bounds of the identifier space for inverted PO front problems. When these problems are optimized with either **R** or **w**, as shown in Figure 3.11a and Figure 3.11c respectively, they achieve a much denser distribution

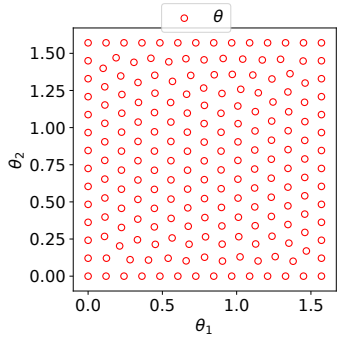


(a) **R**

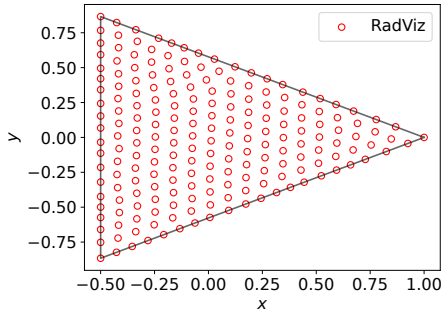
(b) **w**



(c) **P**



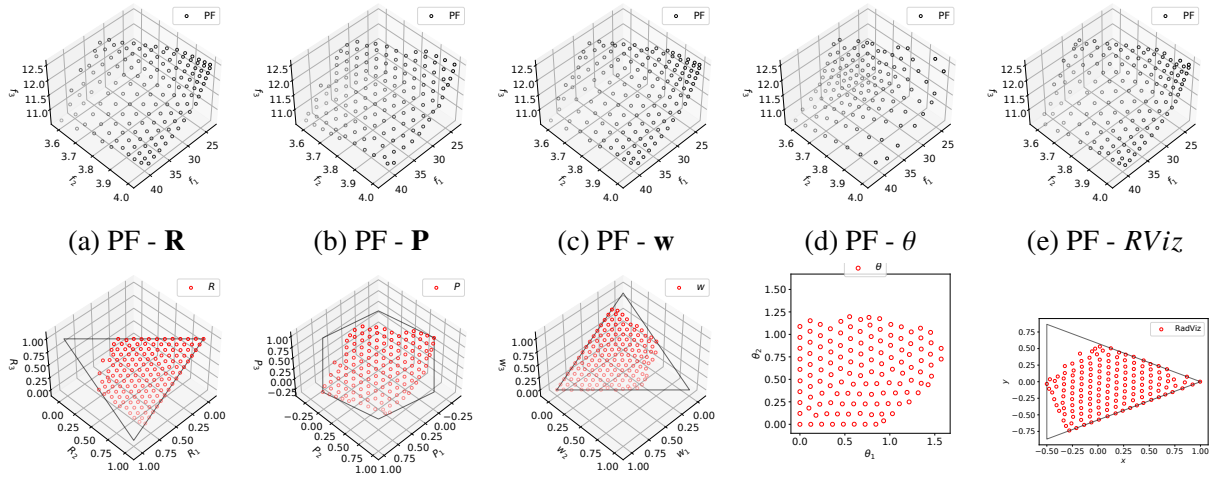
(d) θ



(e) *RViz*

Figure 3.9 Uniformly distributed set of alternative identifier vectors.

of solutions in the objective space as well. This, as expected, comes at the cost of sparse solutions for problems such as MaF3 and WFG2, as shown in Figures 3.11k and Figures 3.11f, respectively. An interesting observation is that even though pseudo-weights are *adaptive* in nature, they produce a similar set of solutions compared to the nadir point RV case. Median HV values for Customized RV NSGA-III with **R** and **w** are shown in Table 3.2 and statistically similar HV values are italicized. We observe that for most problems, both **R** and **w** based Customized NSGA-III have statistically



(a) PF - **R** (b) PF - **P** (c) PF - **w** (d) PF - θ (e) PF - *RViz*

(f) Identifiers - **R** (g) Identifiers - **P** (h) Identifiers - **w** (i) Identifiers - θ (j) Identifiers - *RViz*

Figure 3.10 PF and identifiers for Customized NSGA-III with alternative identifiers for Carside impact problem.

similar performances with the **R** based method performing best compared to baseline for inverted PF problems.

Table 3.2 Median hypervolumes of Customized NSGA-III for nadir point RV, pseudo-weights and Baseline NSGA-III.

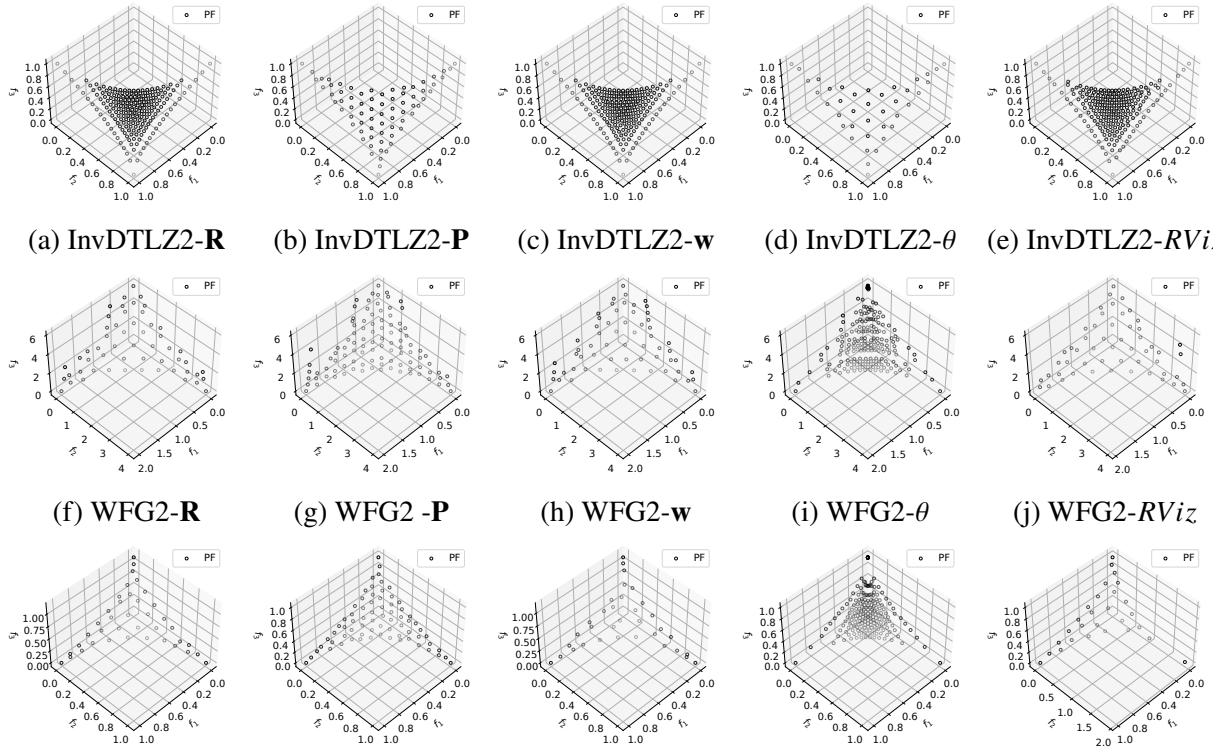
	Problem	M	Customized NSGA-III		NSGA-III
			R	**w**	
1	DTLZ1	3	*9.314E-01*	*9.309E-01*	**9.629E-01**
2	DTLZ2	3	*5.702E-01*	*5.705E-01*	**5.925E-01**
3	C2DTLZ2	3	*5.034E-01*	*5.030E-01*	**5.222E-01**
4	InvertedDTLZ1	3	**2.190E-01**	**2.190E-01**	1.941E-01
5	InvertedDTLZ2	3	**7.079E-02**	7.048E-02	5.350E-02
6	WFG2	3	*1.052E+00*	*1.052E+00*	**1.076E+00**
7	MaF3	3	*1.090E+00*	*1.086E+00*	**1.111E+00**
8	DTLZ7	3	*4.138E-01*	*4.119E-01*	**4.188E-01**
9	Carside	3	*9.044E-01*	*9.043E-01*	**9.098E-01**
10	Crashworthiness	3	*8.937E-01*	*8.956E-01*	**9.083E-01**

Angle based identifiers showed gaps and sparsities when optimized with ideal point RVs (Figure 3.4b). However, when we find PO solutions corresponding to an optimal distribution of θ-vectors, we end up filling the gaps in the θ-space but end up with an oddly dense set of solutions near the minimum values of f_1 and f_2 and maximum values of f_3. This can be observed in Figure 3.11i and Figure 3.11n. This could be of immense value in cases where the DM wants

to use θ-space as their preferred choice of identifiers for decision-making making it convenient to navigate the PF. On the other hand, this does not matter for problems where no PO solutions exist in that region, like the Carside impact problem, as seen in Figure 3.10d.

Projection based RVs present a *middle ground* between the ideal and nadir point RV case and show a *good* distribution of solutions in the objective space irrespective of the shape of the PF. We can observe the difference in Figure 3.10b, Figure 3.11b and Figure 3.11l.

RadViz based optimization leads to a dense yet *messy* distribution of solutions in the PO front, as evidenced by the HV values in Table 3.1. This could be due to the complexity of the projection function and its effect on the higher dimensional objective space.



(a) InvDTLZ2-**R** (b) InvDTLZ2-**P** (c) InvDTLZ2-**w** (d) InvDTLZ2-θ (e) InvDTLZ2-*RViz*

(f) WFG2-**R** (g) WFG2 -**P** (h) WFG2-**w** (i) WFG2-θ (j) WFG2-*RViz*

(k) MaF3-**R** (l) MaF3-**P** (m) MaF3-**w** (n) MaF3-θ (o) MaF3-*RViz*

Figure 3.11 PFs achieved with alternative identifiers in NSGA-III.

3.4.2 Converted identifier space

A simpler approach to achieve a good distribution of solutions in certain identifiers could be to convert the identifiers to ideal point based RVs (**r**) and to run baseline NSGA-III using the *converted*

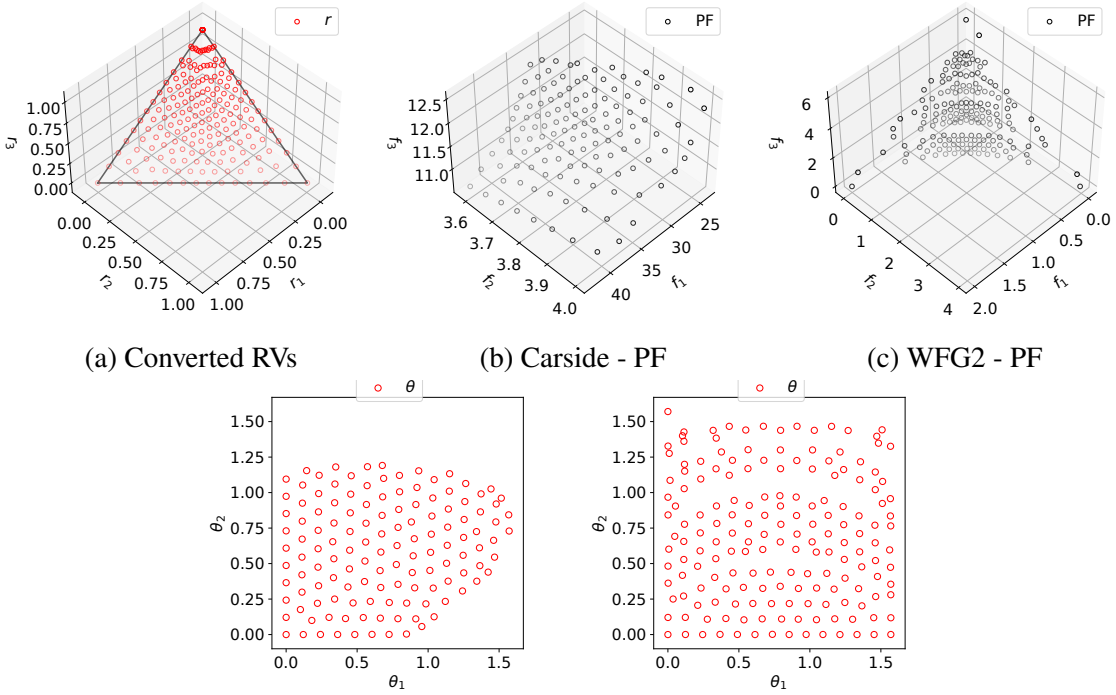
RVs, as shown in Figure 3.8b. We call this *Converted RV* NSGA-III. This conversion will lead to a biased initialization of RVs with baseline NSGA-III. Although this conversion is simple owing to the conversion relationships derived in the earlier sections, the choice of identifiers for conversion is not straightforward. For example, for a three-objective case, converting projection RVs (**P**) to **r** will lead to a loss of reference points as only the central triangle will be present on the non-negative unit simplex. Similarly, converting nadir point based RVs to **r** can lead to issues in convergence as the RVs will lie on a small inverted triangle in the middle. This *incomplete* set of RVs could cause issues in convergence.

Table 3.3 Median hypervolumes of Customized and Converted RV NSGA-III.

	Problem	M	Customized NSGA-III		Converted RV NSGA-III	
			R	θ	**R**	θ
1	DTLZ1	3	9.314E-01	9.579E-01	8.324E-01	**9.582E-01**
2	DTLZ2	3	5.702E-01	5.894E-01	4.653E-01	**5.906E-01**
3	C2DTLZ2	3	5.034E-01	5.176E-01	3.514E-01	**5.210E-01**
4	InvertedDTLZ1	3	2.190E-01	*1.967E-01*	**2.192E-01**	*1.970E-01*
5	InvertedDTLZ2	3	**7.079E-02**	5.651E-02	5.805E-02	5.297E-02
6	WFG2	3	1.052E+00	**1.071E+00**	9.437E-01	1.071E+00
7	MaF3	3	1.090E+00	1.107E+00	1.044E+00	**1.107E+00**
8	DTLZ7	3	4.138E-01	**4.220E-01**	3.382E-01	4.169E-01
9	Carside	3	9.044E-01	9.062E-01	8.541E-01	**9.117E-01**
10	Crashworthiness	3	8.937E-01	**9.080E-01**	7.710E-01	9.023E-01

Median hypervolumes of Customized and Converted RV NSGA-III are shown in Table 3.3. The key observation here is that while both algorithms are supposed to achieve similar results, for the nadir point RV case, the customized version achieves a better value of HV than its Converted RV counterpart for most problems. This could be because converting optimal nadir RVs to ideal point RV space leads to an incomplete unit simplex that further hinders optimization. On the other hand, the Converted RV NSGA-III works similar to the Customized NSGA-III case for converted angle vectors - Converted RV performs better in 5 of 10 cases, Customized NSGA-III performs better in 4 cases and they are statistically similar in 1 of 10 problems. This small discrepancy could be due to the normalization procedure used in the two methods. Uniformly distributed angle vectors converted to ideal point based RVs are shown in Figure 3.12a. Using this as RVs in NSGA-III

leads to a biased distribution of PO solutions as seen in Figures 3.12b-3.12c. Their corresponding angle vectors after optimization are shown in Figures 3.12d-3.12e. This is because *optimal* angle vectors form a biased yet complete set of reference identifiers for NSGA-III. It could be interesting to develop identifiers specifically compatible with this kind of conversions.



(a) Converted RVs (b) Carside - PF (c) WFG2 - PF

(d) Carside - Polar (e) WFG2 - Polar

Figure 3.12 NSGA-III after converting uniformly distributed angle vectors to RVs.

3.4.3 Combined identifier spaces

Previously discussed methods are focused on achieving a good distribution of solutions in one identifier space. It is always possible that either due to preferences of the DM or due to prior knowledge about the shape of the PF, we might want to achieve a good distribution of solutions in multiple identifier spaces. In this section, we present a method that can achieve this by generating *reference identifiers* in multiple spaces and ensuring a good distribution in the overlapping regions using the Riesz's energy [53, 61] method. The aim is to perform optimization using a set of reference identifiers that when converted to other identifiers are reasonably well distributed.

Based on the mathematical relationships derived in the earlier sections, a good distribution of

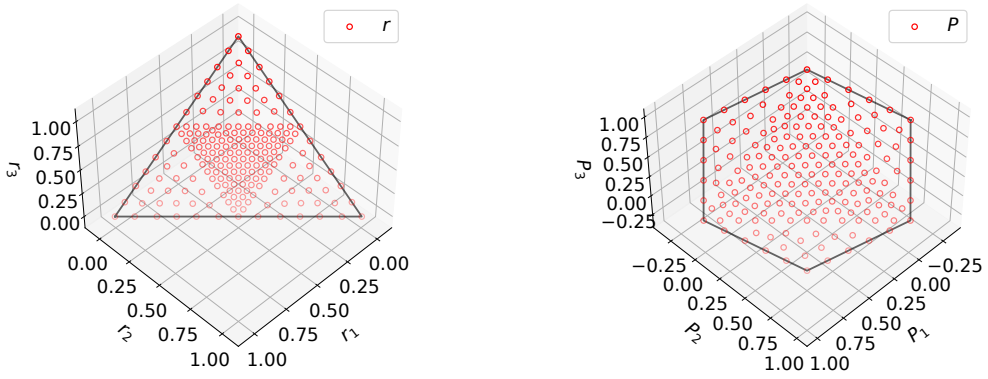
solutions in one identifier space might mean the identifiers are oddly distributed in other spaces. Similarly, it is also possible that reference identifiers in one space might not be valid points in the other identifier space. To handle these issues, we propose a method in which uniform points are generated in each identifier space, and then the overlapping regions are further optimized to be well distributed. The aim is to have a set of RVs that can satisfy the requirements of DMs if they prefer more than one identifier as their choice of space for decision-making. Along with the choices of combinations of identifiers, the identifier that encompasses the other should be preferred as the space where optimization is to be performed. This ensures that RVs are not lost during conversion to perform optimization.

The procedure for combined Riesz Energy optimization for the two-identifier case with population size N is as follows:

1. Initialize $N/2$ uniformly distributed vectors for each identifier.
2. Convert the identifiers to a common identifier space.
3. Combine the vectors and delete any common points in the boundaries of each identifier space.
4. Freeze the boundary points.
5. Until convergence or maximum number of iterations: Perform one Riesz Energy optimization step in each identifier space and perform a weighted average of the identifier vectors in the common identifier space. Weights for this weighted average are $\frac{1}{F_E}$ where F_E is the potential *energy* objective value in the common identifier space.

This procedure can be easily extended to a higher number of identifiers. Examples of combined RVs generated using the above procedure are shown in Figure 3.13. Specifically, Figure 3.13a is generated by a combination of ideal and nadir point based RVs where optimization is intended to be performed in the ideal RV space. We can easily observe the dense set of points in the overlapping region - the small inverted triangle. Similarly, Figure 3.13b show RVs generated as a combination

of projection based RVs and ideal point RVs. Again, we see the dense set of RVs in the overlapping region - the central triangular region (unit simplex) of the hexagon.



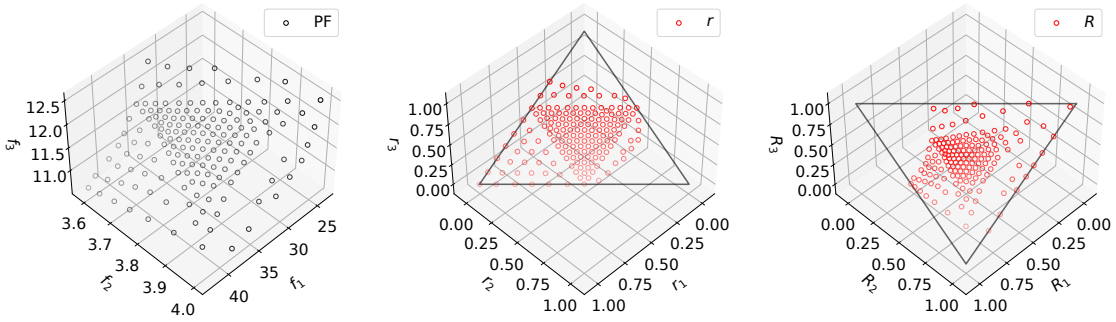
(a) **r** and **R** (b) **P** and **r**

Figure 3.13 Combined RVs.

Table 3.1 and Figure 3.14 show optimization results with combined identifiers. We can clearly observe the influence of the combined RVs in both PFs shown. In order to understand the *qualitative* properties of the distributions in the identifier spaces, we compute IGD in the identifier spaces, as shown in Table 3.4. Here, lower IGDI values indicate that the solutions are *better* arranged in the identifier space. Each value in the table is dependent on two identifiers: one that was used during optimization and one that was used to compute the IGDI metric. Measurements in each identifier that can be compared with each other are shaded in the same color. The best value in each computed identifier is marked in bold along with other statistically similar values. The question of which identifier is the most suitable (based on the IGDI values) is dependent on the shape of the PF of the problem. For example, we see that baseline NSGA-III (which optimizes in **r**-space) has the best IGDI values when measured in **r**-space for problems with typical PF shapes. On the other hand, for inverted PF problems, Customized NSGA-III optimized in **R** provides the best IGDI values in **R**-space. Problems that do not conform to the unit simplex shape, such as DTLZ7 and Crashworthiness show **r**-space IGDI values that are best for multiple cases.

A key observation here is that the combined RV cases have IGDI values that fall in between the two cases. For example, for problems where the PF shape matches that of the unit simplex (like

DTLZ1, DTLZ2 and C2DTLZ2), when computed in **r**-space, IGDI values are the best for baseline NSGA-III, worst for **R**-based Customized NSGA-III and in the middle for **r**-**R** based Combined RV NSGA-III case. We see a similar observation for Carside impact problem for **P**, **r**and **r**-**P** case for IGDI measured in **P**-space. This effect is not as evident because **r**-space and **P**-space are very closely related to one another in terms of properties compared to the **r**-**R** case.



(a) **r** - **R** RVs - Carside - PF (b) **r** - **R** RVs - Carside - **r** (c) **r** - **R** RVs - Carside - **R**

(d) **P** - **r** RVs - Carside - PF (e) **P** - **r** RVs - Carside - **P** (f) **P** - **r** RVs - Carside - **r**

Figure 3.14 PF and identifiers achieved using combined RVs with NSGA-III.

Table 3.4 Median IGDI values: low values indicate that the final distribution of solutions achieved are optimal in that identifier space.

		Customized NSGA-III						Combined RV NSGA-III						Baseline NSGA-III		
	Optimized in	**R**			**P**			**r-R**			**P-r**			**r**		
	IGD computed in	**r**	**R**	**P**	**r**	**R**	**P**	**r**	**R**	**P**	**r**	**R**	**P**	**r**	**R**	**P**
1	DTLZ1	5.19E-02	1.85E-01	1.23E-01	3.80E-02	1.82E-01	1.12E-01	2.99E-02	1.78E-01	1.06E-01	3.48E-02	1.80E-01	1.12E-01	**1.29E-02**	**1.77E-01**	**1.02E-01**
2	DTLZ2	4.77E-02	9.86E-02	6.81E-02	3.17E-02	9.86E-02	**5.31E-02**	2.99E-02	**9.54E-02**	**5.58E-02**	3.28E-02	9.80E-02	6.00E-02	**1.29E-02**	9.68E-02	**5.44E-02**
3	C2DTLZ2	6.64E-02	1.36E-01	1.14E-01	5.47E-02	1.36E-01	**1.05E-01**	5.18E-02	**1.29E-01**	9.88E-02	5.49E-02	1.32E-01	**1.03E-01**	**4.01E-02**	1.33E-01	**1.03E-01**
4	InvertedDTLZ1	**1.78E-01**	**1.23E-02**	**1.03E-01**	1.81E-01	3.53E-02	**1.08E-01**	1.79E-01	3.42E-02	1.09E-01	1.80E-01	3.41E-02	1.11E-01	**1.77E-01**	4.92E-02	1.16E-01
5	InvertedDTLZ2	**2.27E-01**	**9.56E-03**	**1.70E-01**	2.28E-01	5.42E-02	1.75E-01	2.29E-01	7.00E-02	1.82E-01	2.29E-01	4.68E-02	1.76E-01	2.30E-01	9.48E-02	1.88E-01
6	WFG2	6.98E-02	**1.76E-01**	1.38E-01	4.61E-02	**1.83E-01**	**1.28E-01**	3.12E-02	2.08E-01	1.49E-01	4.01E-02	1.92E-01	1.36E-01	**1.73E-02**	2.03E-01	1.42E-01
7	MaF3	8.99E-02	2.29E-01	1.82E-01	5.59E-02	2.27E-01	1.76E-01	2.99E-02	2.29E-01	1.75E-01	4.71E-02	**2.27E-01**	1.72E-01	**1.37E-02**	**2.27E-01**	**1.70E-01**
8	DTLZ7	1.04E-01	**7.51E-02**	**1.00E-01**	**1.03E-01**	8.12E-02	**9.88E-02**	1.05E-01	8.26E-02	1.05E-01	**1.03E-01**	8.09E-02	**1.02E-01**	**1.03E-01**	8.79E-02	1.08E-01
9	Carside	8.29E-02	**5.63E-02**	6.88E-02	7.89E-02	6.58E-02	**5.75E-02**	7.87E-02	8.06E-02	7.59E-02	7.88E-02	7.15E-02	6.90E-02	**7.02E-02**	7.75E-02	7.26E-02
10	Crashworthiness	1.47E-01	1.68E-01	2.08E-01	1.41E-01	1.72E-01	2.07E-01	1.45E-01	1.55E-01	**1.91E-01**	**1.37E-01**	1.73E-01	2.06E-01	**1.38E-01**	**1.59E-01**	**1.93E-01**

3.5 Summary

In this chapter, we have compared and discussed the advantages and disadvantages of several distinct identifiers for Pareto-optimal solutions: ideal point RVs, nadir point RVs, projection RVs, pseudo-weight vectors, angle vectors and RadViz coordinates from the perspective of optimization, visualization and decision-making. We have demonstrated that different identifiers provide varying properties and lead to different distributions of the same PO solutions. These identifiers have different interpretations and can lead to distinct experiences during decision-making or visualization. For example, RV-based identifiers provide a geometric perspective, pseudo-weight vectors represent priorities of objectives, and angle vectors can describe direct objective-wise relationships. Hence, pseudo-weight vectors and angle vectors can be of use at higher dimensions when a geometric interpretation of the PO front is not possible. An important aspect of using identifiers for visualization and decision-making is identifying apparent gaps and discontinuities in PO fronts. Gaps in the PO front will lead to gaps in the identifier space. However, gaps in the identifier space might not always mean gaps in the PO front, as the shape of the Pareto front can lead to gaps or incomplete regions in the identifier space. Thus, a proper understanding of the mapping of PO points and the respective identifier space, as derived through mathematical relationships in this study, is important.

We have also demonstrated that in cases where the DM has a preference over the identifiers, EMaO algorithms, like NSGA-III can be modified, generating what is here called Customized NSGA-III, to perform diversity preserving operations in a suitable identifier space instead of using the typical ideal point RVs. This can lead to a superior distribution of solutions in the identifier space for visualization and decision-making purposes. We consider converting other identifiers to ideal point based RVs before performing optimization as an easier solution and we discuss the flaws of this approach. We also consider the situation of a DM preferring to perform decision-making using multiple identifiers and propose a Riesz Energy based method to find a uniform distribution of RVs while combining identifier spaces. We achieve this by combining a uniform set of RVs in each space and optimizing the overlapping regions in all identifier spaces.

CHAPTER 4

POST-OPTIMAL MACHINE LEARNING APPLICATION USING PSEUDO WEIGHT IDENTIFIERS

In this chapter, we propose an ML-assisted post-optimization multi-criterion decision-making procedure in which patterns between a Pareto set and pseudo-weights are learned, and new PO solutions are created based on user-desired pseudo-weights. Most of this chapter is based on [62]. The trained ML model is used for several functionalities, as listed below:

Task 1: The trained model can be verified by predicting $\hat{\mathbf{x}}$-vectors by using a test set of K input-output pairs from the true PO set: $(\mathbf{w}^{(k)}, \mathbf{x}^{(k)})$, for $k = 1, \ldots, K$.

Task 2: The trained model can be used to fill apparent intermediate *gaps* and find new ND-solutions. $\hat{\mathbf{x}}$-vectors can be predicted from pseudo-weights generated at the gaps.

Task 3: The trained model can be used to fill apparent gaps in the extremities on the PO front and find new extreme ND-solutions.

Task 4: The trained model can be used to fill apparent gaps observed on the EMO/EMaO-obtained ND front and validate whether the gaps are true. This can be achieved by predicting $\hat{\mathbf{x}}$-vector from pseudo-weights in the gaps and evaluating $\mathbf{f}$-vectors of the corresponding predictions. Domination of predicted $\mathbf{f}$-vectors would imply that the apparent gap is true.

Task 5: As a continuation of Task 4, a true gap arising due to the infeasibility of solutions in the gap can be predicted. This can be achieved by predicting $\hat{\mathbf{x}}$-vector from pseudo-weights in the gaps and evaluating constraints.

Task 6: The trained model can be used for quickly populating PO front in desired or sparse regions. This can be helpful in decision-making or visualization tasks.

4.1 Literature survey

Machine learning (ML) methods have been actively used in tandem with MOEAs in the recent past. The use of machine learning in MOEAs can be roughly divided into the following categories:

- **Surrogate assistance**: ML methods are used to approximate computationally expensive objectives and exhaustive optimization is performed on this approximation [63]. This helps in drastically reducing the total number of high-fidelity evaluations needed to solve an optimization problem.

- **Generative operators**: ML methods are used to create new offspring during optimization. These methods exploit patterns in high-performing solutions in order to find new unseen solutions to drive optimization forward. These methods are also referred to as *innovization* in the literature. Generative operators have been employed based on supervised learning [34, 35], inverse mapping [32, 33] and using generative models [64].

In this study, we focus on the generative operators category. These methods can be further split into two major sub-categories:

1. **Post optimization:** ML methods act as data analysis tools to get the best out of the generated solutions. This can be to understand the solutions better or to identify new solutions.

2. **During optimization:** ML methods are used to accelerate the optimization process by learning patterns amongst high-performing solutions.

4.1.1 Patterns to be learned in MOO

Before the use of ML in various aspects of MOO, it is important to understand the existence of patterns in data collected from MOEAs. We discuss two such areas where theoretical and experimental studies have proven that patterns exist that can potentially be exploited by ML algorithms.

4.1.2 Structure in PO front

It is well known in the community that under mild regularity conditions, the PO set of an M-objective problem lies at most on an (M-1)-dimensional piece-wise continuous manifold [29]. This property is derived based on Karush-Kuhn-Tucker (KKT) conditions for multi-objective optimization. Jin et al. [65] theorized that the order of the polynomial function describing the PO set in $\mathbf{x}$-space is lower than or equal to that of a function describing the PO-front in $\mathbf{f}$-space. Hence,

if a set of ND solutions is close enough to the actual PO front, they can potentially be modeled by (M-1) variables using machine learning.

4.1.3 Post optimization Machine Learning and analysis for MCDM

The idea of using ML at the end of optimization for the purpose of MCDM has been used before for the purpose of creating new solutions on the PO front. Giagkiozis et al. [31] learned the mapping between reference vectors and decision variables and found new ND solutions on the existing PO front. The authors used Radial Basis Neural Networks (RBNN) [66] to learn the corresponding mapping. Similarly, Takagi et al. [67, 68] used Kriging [69] and RBNNs to map reference vectors to decision variables and objective functions. Using reference directions as inputs makes it convenient to visualize the geometry of the PO front but requires some knowledge about the shape of the PO front to generate the desired solutions. Also, these methods lack specialized means to handle constrained optimization problems. Another well-studied way of using ML for creating solutions is inverse modeling techniques [30, 32, 33, 70, 71, 72] that use ML methods to map from $\mathbf{f}$ to $\mathbf{x}$. Gaspar-Cunha et al. [71] used ANNs to build inverse models based on perturbations in the objective space. IM-MOEA [32] uses GPR based inverse models to sample more candidates from objective space as required to achieve better diversity and convergence. To simplify the modeling process, the authors decomposed the mapping model into multiple univariate models. Random grouping of variables was also considered to increase the chance of correlated variables being considered together. Though these methods have been used for creating solutions during optimization, the authors show that the model can also be used to sample more points a posteriori to get more ND solutions. Bidgoli et al. [30] use generative methods like Latin Hypercube Sampling and Opposition-based Learning to create new non-dominated solutions close to existing PO solutions to achieve a well-distributed PO front. They apply these generative methods either in the decision space (to be evaluated for objectives) or in the objective space (to be mapped to the decision space and then evaluated). However, knowledge of the location of a true $\mathbf{f}$-vector is required to obtain a meaningful $\mathbf{x}$ vector. For example, an arbitrarily chosen $\mathbf{f}$-vector on the interpolated gap in the EMO/EMaO-obtained ND front, as demanded by the decision-maker, may

not yield a meaningful **x**-vector. The feasible **f**-vector must be specified to have a corresponding **x**-vector. A slight deviation from the feasible and PO objective vector is enough to make such inverse methods vulnerable to failure. This may cause difficulties in decision-making and may require several iterations with the **f**-vector to obtain a feasible **x**-vector of interest. Though not a targetted method, He et al. [64] used a Generative Adversarial Network (GAN) [73], trained on existing solutions to sample candidate solutions during an EMO procedure. This method can also be applied at the end of optimization to create new solutions on the PO front, however with no control over the solutions being created. There have also been other studies, such as [74, 75], to parameterize the PO front. These approximations can be used to sample more points on the obtained PO-front.

In the subsequent sections, we describe the proposed methodology and discuss the effects of using different ML methods. Apart from this, we also propose an effective way to deal with constrained optimization problems and study the scalability of the proposed method.

4.2 Proposed post optimal ML method

Figure 4.1 shows a sketch of a two-objective feasible objective space (in blue color) with the respective PO set shown as a red line on the lower left boundary of the objective space. A specific

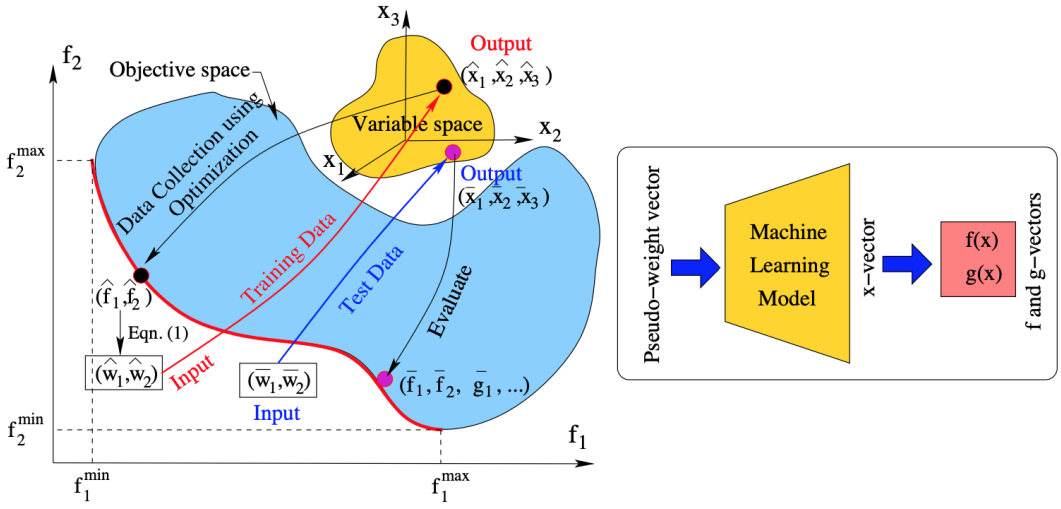


Figure 4.1 Training data generation and test data to create new Pareto-optimal solution using pseudo-weight vectors.

PO solution $\hat{\mathbf{x}} = (\hat{x}_1, \hat{x}_2, \hat{x}_3)$ in the three-dimensional variable space (marked in golden color)

produces a PO solution $(\hat{f_1}, \hat{f_2})$ marked with a black dot in the figure. Corresponding pseudo-weight vectors ($\hat{\mathbf{w}} = (\hat{w}_1, \hat{w}_2)$) can be computed using Equation 2.1 by using the minimum and maximum objective values of the PO set, shown in the figure. Thus, the specific pseudo-weight vector $\hat{\mathbf{w}}$ can be associated with a specific PO variable vector $\hat{\mathbf{x}}$. Similarly, other PO solutions can also be assigned an equivalent pseudo-weight vector. Care should be taken while curating **w**-**x** dataset for training ML models as existance of Dominant Resistant Solutions (DRS) in the dataset can make pseudo-weight values of actual PO front very close to each other. This can lead to difficulties in training ML models. Hence, DRS solutions can be filtered out first, using methods like cone domination [76], before computing pseudo-weights.

After forming the **w**-**x** data from EMO/EMaO obtained ND solutions, an ML model can be trained to capture the relationships between input vector **w** and output vector **x**. The steps are enumerated below:

1. Run an MOEA and collect K non-dominated solutions.
2. Compute pseudo-weight vectors ($\mathbf{w}^{(k)}$) for all K solutions ($k = 1, \ldots, K$) of the ND-set using Equation 2.1.
3. Prepare training data with input-output pairs $(\mathbf{w}^{(k)}, \mathbf{x}^{(k)})$ for $k = 1, \ldots, K$.
4. Train an ML model ($f(\mathbf{w}) = \mathbf{x}$) using the prepared training data.
5. Use the ML model to predict the **x**-vector for any desired **w**-vector. Objectives (**f**-vectors) and constraints (**g**-vectors) can be computed from the predicted **x**-vectors by evaluating them to determine their domination level and feasibility.

4.3 Training of ML methods

The task at hand is to predict an n-dimensional **x**-vector from an M-dimensional **w**-vector, where M is the number of objectives and n is the number of variables. Since in most real-world problems, $M \ll n$, this ML model learning task can be challenging. This challenge is also exacerbated by the fact that we are dealing with only a small amount of data (K is at most a few hundreds) here,

since the training data is the ND-set of an EMO/EMaO. Hence, the current problem is a small-data oriented learning task and is distinct from deep-learning frameworks where data is abundant. There is another challenge worth mentioning. In many practical problems, variable x_i representing one of the output entities would be intricately related to other variables x_j. Thus, ideally, the chosen ML model must use a method that is capable of learning the correlations among the output entities. We explore two ML approaches for this purpose: (i) Artificial Neural Network (ANN), (ii) Gaussian Process Regression (GPR). ANNs capture correlations between outputs but usually require more data. Vanilla GPR is usually trained for one output at a time but is highly effective in a sparse data setting. Multi-task GPR captures the best of both worlds. We restrict most of our study to GPR and ANN-based ML methods as a proof of concept and compare them with the multi-task GPR approach for a scale-up study. A better customized ML method can also be used here, depending on dataset size, and numbers of objectives and variables. In these approaches, **x**-vectors are normalized to zero mean and unit variance as necessary. Since pseudo-weight vectors are always within $[0, 1]$, they need not be normalized further.

The ND front is generated using NSGA-III [2] in this study and is used as for training, validation, and test sets. A reasonable population size of $N = 110M + 10$ is chosen for the proof-of-concept study. Of the N points, $100M$ are used as training set and $10M$ are used as test set. The remaining 10 points are used as a validation set for ANNs but are discarded for the GPR method.

4.4 Handling variable bounds and constraints

During the optimization process, the lower and upper bounds of decision variables are usually known. However, machine learning methods like ANN and GPR approach may not restrict outputs to certain bounds. Hence, **x**-vectors predicted from new pseudo-weights might have variables that do not belong within the bounds. In this study, the output values are clipped to the problem's variable bounds. Although activation functions, like sigmoid, can be incorporated into ANN approach to restrict values to certain bounds, we do not pursue that approach as all types of variable bounds might not have a corresponding activation function.

Constraint satisfaction is another necessity in any optimization task. Since all final ND solutions

from an EMO/EMaO run are expected to be feasible (satisfying all constraints), the **w**-**x** data will correspond to feasible solutions only. Thus, the ML model is expected to learn how to come up with a feasible **x**-vector for any given **w**-vector. However, not every predicted **x**-vector may satisfy all constraints in reality, due to the inaccuracies of the learning process or because the associated **x**-vector for the given **w**-vector is expected to fall in the infeasible region. A more sophisticated learning model may need to be designed to ensure that every predicted **x**-vector is feasible. In this study, we propose a simple approach of using additional data of infeasible solutions, such that in addition to learning the **w**-**x** mapping, an ML model can also learn to predict whether a solution is feasible. For example, the values of the constraint functions, $G(\mathbf{x}) = [g_1(\mathbf{x}), ..., g_J(\mathbf{x})]^T$ can be additional outputs for the ML method and these predicted constraint values can be used to decide whether the solution needs to be discounted or evaluated.

The final ND set provided by the EMO/EMaO method will not contain any infeasible solutions, leading to a biased dataset for predicting feasibility. Hence, we propose two approaches to circumvent this dataset bias. The first approach is similar to SAO, in which predicted **x**-vectors are evaluated and their constraint values are added to the training dataset for predicting feasibility. As this dataset grows, accuracy of predicted feasibility or constraint violation will improve. The second proposed approach is a simple yet effective archiving strategy to ensure that both feasible and infeasible solutions are present in the final training dataset. The proposed strategy can be readily included with any existing EMO algorithm without changing the algorithm or function evaluations. This strategy is based on the hypothesis that the EMO method's genetic operators produce offspring that are infeasible and yet on the same manifold as ND solutions. Collecting these solutions during the EMO procedure will provide a more *complete* dataset for training the ML models to predict feasibility. The steps at the end of each generation for archiving are enumerated below. The archive is empty to start with.

1. Add offspring population to the archive.

2. Find non-dominated and feasible solutions (ND-feasible) of the current generation's popula-

tion (this is usually the surviving population of an EMO/EMaO approach).

3. Ignore constraint values and retain solutions from the archive that are *non-dominated* with the ND-feasible set.

4. To prevent the archive from retaining solutions far away from the actual PF, filter solutions in the archive based on the current nadir and ideal-point.

5. Use reference-direction based survival (while ignoring constraints) [2] to retain a certain number of solutions in the archive.

The final archive must contain decision variables, $\mathbf{x}$, objective values, $F(\mathbf{x})$, constraint values, $G(\mathbf{x})$ and feasibility of each population member. This dataset can now be used to train ML models with corresponding pseudo-weights as input and constraint values, $G(\mathbf{x})$ or feasibility as output.

In this proof-of-principle study, we briefly discuss the effectiveness of the proposed archiving strategy. We use this archive dataset only for training models that predict either constraint values or feasibility. These ML models are trained with twice the amount of data compared to $\mathbf{w}$-vectors to $\mathbf{x}$-vectors mapping models. We use the kNN-classifier for the binary classification problem and GPR approach for predicting $G(\mathbf{x})$ values. For testing purposes, we obtain an $\mathbf{x}$-vector for any desired $\mathbf{w}$-vector and evaluate constraint values at $\mathbf{x}$. We report the solution's feasibility based on its actual constraint values.

4.5 Results

We first present results on two-objective unconstrained (ZDT) [77], and constrained (BNH [78], OSY [79] and TNK [25]) test problems. Then, we show results on three-objective problems (DTLZ [55], WFG2 [56], Car side impact [58] and crash-worthiness [59]), followed by a few many-objective problems (DTLZ). All experiments are conducted 31 times with different random seeds and the mean results are presented. In the end, we conduct a scale up study to understand the efficacy of the proposed method for higher number of variables.

Tables 4.1 to 4.7 show Mean Absolute Errors (MAE), that is, mean of absolute difference between prediction and target, of the test set on multi- and many-objective problems with ANN and

GPR approaches. The mean **x** shows the average MAE scaled between variable bounds between target **x**-vectors (from the PO set) and predicted **x**-vectors. Similarly, mean **f** is the average MAE scaled based on range of objectives on the PO front (in $[\mathbf{f}^{ideal}, \mathbf{f}^{nadir}]$) between the target **f**-vectors and **f**-vectors from evaluating predicted **x**-vectors of the test set. Their standard deviation across the test set is also presented in the table. 'Random' signifies experiments in which the test set is uniformly taken from the PO set (Task 1). 'Continuous' and 'Edge' describe the cases in which the test sets are created from a continuous region on the PO front, either in the middle or towards the extreme parts (Task 2), respectively, on the obtained ND set. It is clear that in all problems, both ANN and GPR approaches are able to predict the **x** and **f**-vectors with small errors. The error is smaller for continuously and randomly created test data, compared to the test data from the edge of the ND set. This is expected, mainly due to the issues related to the normalization process and we discuss more about it in Section 4.5.1.2.

4.5.1 Two-objective Optimization Problems

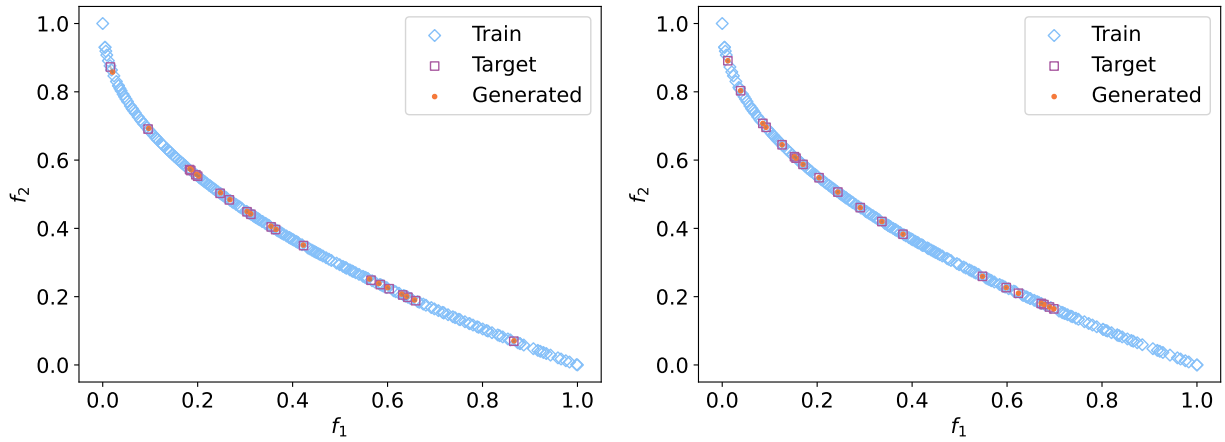
Prediction errors in **x** and **f**-vectors of the test data-set by the ANN approach are shown in Table 4.1 for unconstrained ZDT [77] and constrained (BNH, OSY and TNK) problems. We can clearly see that the error values are low for all test problems.

Table 4.1 Performance of ANN approach on two-objective problems.

Prob.	Test Data	Mean MAE in **x**	Mean MAE in **f**	Std. Dev. MAE in **x**	Std. Dev. MAE in **f**
ZDT1	Continuous	8.548E-04	6.889E-03	1.761E-03	3.253E-03
	Edge	3.839E-03	7.689E-02	1.099E-02	4.371E-02
	Random	9.466E-04	1.889E-02	4.380E-03	1.414E-02
ZDT2	Continuous	1.131E-03	1.729E-02	3.025E-03	8.506E-03
	Edge	3.970E-03	5.919E-02	1.811E-02	4.218E-02
	Random	7.784E-04	8.564E-03	2.517E-03	9.227E-03
ZDT3	Continuous	1.267E-03	4.480E-02	5.948E-03	4.779E-02
	Random	4.619E-04	1.757E-02	2.591E-04	1.434E-02
BNH	Continuous	6.366E-03	5.039E-03	3.879E-03	1.696E-03
	Random	4.039E-03	2.852E-03	4.623E-03	4.368E-03
OSY	Continuous	2.281E-03	1.910E-02	2.949E-03	1.451E-02
	Random	8.177E-03	1.804E-02	2.626E-02	2.870E-02
TNK	Continuous	1.022E-02	3.224E-02	5.815E-03	1.834E-02
	Random	3.189E-03	1.006E-02	3.720E-03	1.173E-02

4.5.1.1 Validating Task 1

Figure 4.2a shows corresponding **f**-vectors from predicted **x**-vectors from randomly chosen pseudo-weights (test set). We can clearly see that the ANN generated **f**-vectors (red filled circles) fall almost inside the target **f**-vector (red square). This is not an easy feat as the ANN predicts correct **x**-vectors (that are evaluated to find **f**-vectors for plotting) from pseudo-weights not present in the training data (blue diamonds).



(a) ANN approach

(b) GPR approach

Figure 4.2 ANN and GPR approaches on random test data for ZDT1 problem, demonstrating Task 1.

We apply the GPR and Multi-task GPR approaches and tabulate the results in Table 4.2 and 4.3 respectively. It is interesting to note that much smaller errors are observed for GPR and multi-task GPR approach compared to the ANN approach. The multi-task GPR approach performs slightly worse compared to the standard GPR approach. Figure 4.2b clearly shows that GPR approach can also obtain PO points very close to target **f**-vectors.

4.5.1.2 Validating Task 2

We then investigate the efficacy of the proposed methods on continuous intermediate gaps in the PO front produced by an EMO algorithm (Task 2). Figure 4.4 shows that GPR approach can produce points in the middle of the PO front on ZDT2 and ZDT3 problems. Similarly, Figure 4.5 shows that multi-task GPR approach can also produce solutions in the middle of the PO front for ZDT1 and ZDT2 problems. We show the **x**-space of training and predicted solutions (from the gap)

Table 4.2 Performance of GPR approach on two-objective unconstrained and constrained problems.

Prob.	Test Data	Mean MAE in **x**	Mean MAE in **f**	Std. Dev. MAE in **x**	Std. Dev. MAE in **f**
ZDT1	Continuous	1.091E-08	1.022E-07	5.642E-08	6.062E-08
	Edge	8.081E-06	1.434E-03	8.897E-05	5.463E-03
	Random	2.101E-08	2.008E-06	3.106E-07	1.209E-05
ZDT2	Continuous	1.140E-04	3.796E-03	7.076E-04	2.190E-03
	Edge	4.598E-04	7.340E-03	3.294E-03	1.059E-02
	Random	6.498E-05	2.152E-03	7.820E-04	4.541E-03
ZDT3	Continuous	2.754E-05	3.260E-03	1.855E-04	3.749E-03
	Random	2.357E-06	2.070E-04	4.162E-05	7.697E-04
BNH	Continuous	1.890E-03	2.914E-04	1.497E-03	1.336E-04
	Random	2.237E-03	6.407E-04	3.230E-03	2.094E-03
OSY	Continuous	1.677E-03	5.743E-03	4.052E-03	8.513E-03
	Random	2.379E-03	4.377E-03	1.299E-02	1.847E-02
TNK	Continuous	7.907E-03	2.493E-02	9.725E-03	3.066E-02
	Random	1.372E-04	4.327E-04	4.278E-04	1.349E-03

Table 4.3 Performance of Multi-task GPR approach on unconstrained and constrained two-objective problems.

Prob.	Test Data	Mean MAE in **x**	Mean MAE in **f**	Std. Dev. MAE in **x**	Std. Dev. MAE in **f**
ZDT1	Continuous	3.114E-07	8.251E-06	1.630E-06	9.122E-07
	Edge	8.097E-05	7.689E-03	5.862E-04	1.474E-02
	Random	1.097E-06	8.391E-05	1.496E-05	3.852E-04
ZDT2	Continuous	1.305E-06	4.384E-05	7.745E-06	2.030E-05
	Edge	1.199E-03	1.980E-02	7.218E-03	2.045E-02
	Random	4.267E-05	7.856E-04	5.268E-04	1.923E-03
ZDT3	Continuous	5.595E-04	6.857E-02	3.440E-03	6.652E-02
	Random	4.036E-04	2.853E-02	2.992E-03	4.052E-02
BNH	Continuous	1.634E-03	1.974E-05	1.278E-03	1.029E-05
	Random	1.877E-03	3.991E-04	2.545E-03	2.030E-03
OSY	Continuous	9.016E-03	2.683E-02	1.257E-02	2.856E-02
	Random	6.316E-03	3.289E-02	2.845E-02	1.313E-01
TNK	Continuous	1.424E-04	4.490E-04	6.865E-05	2.165E-04
	Random	2.125E-04	6.701E-04	6.160E-04	1.942E-03

for ZDT1 problem in Figure 4.3. Here, we can clearly observe that x_1 that is distributed in the full range of the variable space with predicted values in the middle. As expected, all other variables (x_i for $i \in 2, 3, ..., 30$) are at 0.0. We show only the first 5 variables and ommit the remaining 25 in the figure due to space constraints. The remaining 25 variables are at 0.0. For a two-objective

problem, (M-1)-manifold will be one dimensional and this can be evidently seen here as only one variable is varying. Here, too, the points in the gaps were not used during training of the models. Pseudo-weights corresponding to the gap were held-out and used during testing.

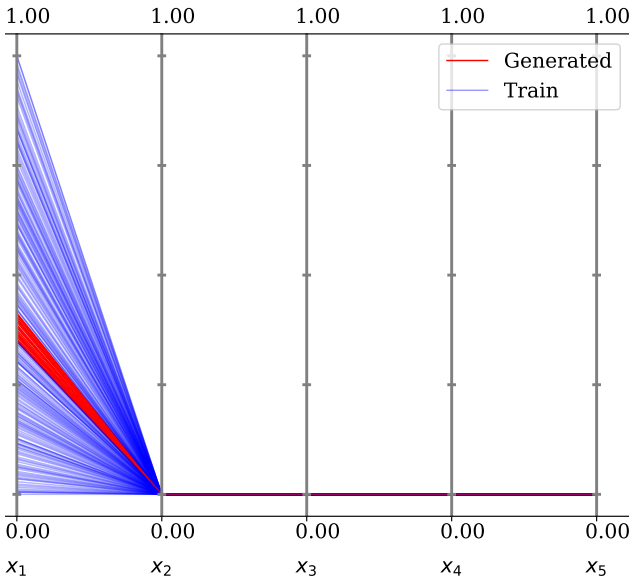
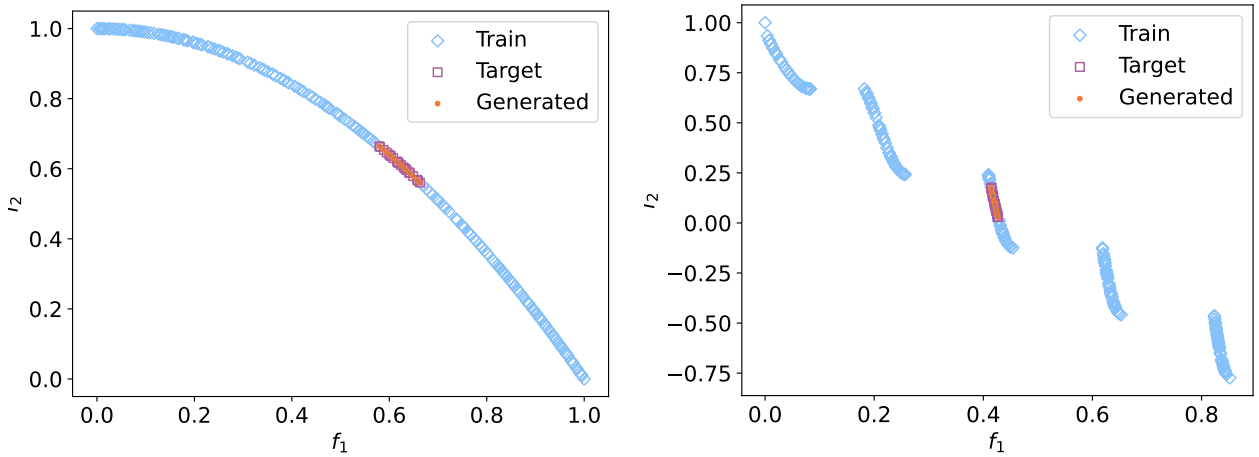


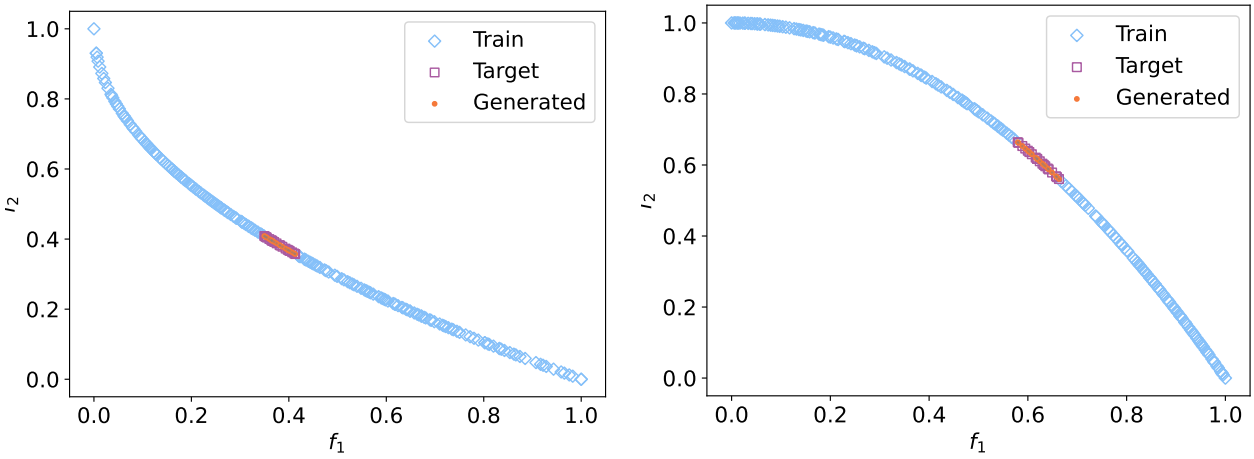
Figure 4.3 PCP plot of **x**-space of training and generated solutions for ZDT1 with GPR approach and continuous gap.



(a) ZDT2 (b) ZDT3

Figure 4.4 GPR approach on continuous gap in middle of PO front for ZDT2 and ZDT3 problems, demonstrating Task 2.

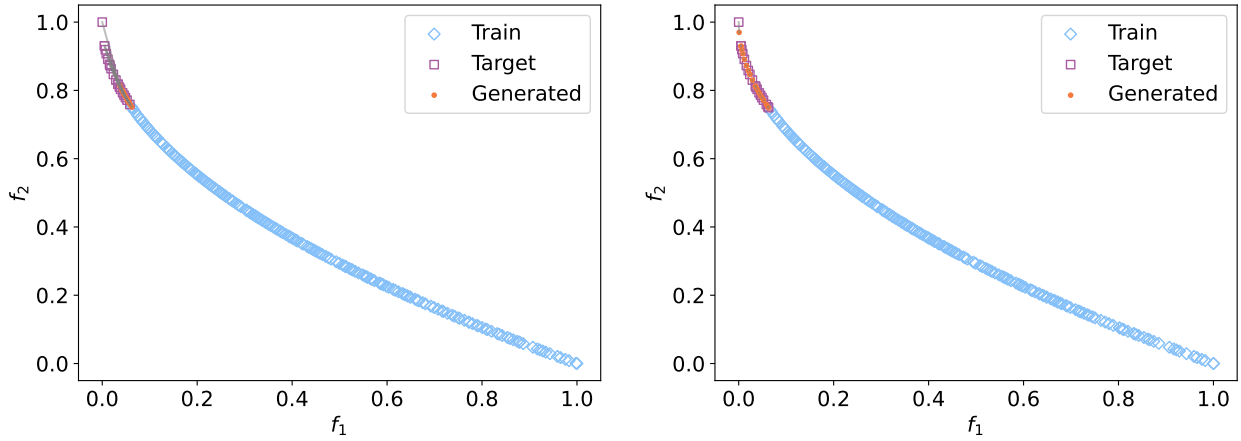
Figure 4.6 shows points produced by ANN and GPR approach for ZDT1 problem, where, a gap (used as the test set) is created at one of the extremes of the PO front. Here, the pseudo-weights



(a) ZDT1 (b) ZDT2

Figure 4.5 Multi-task GPR approach on continuous gap in middle of PO front for ZDT1 and ZDT2 problems, demonstrating Task 2.

are calculated based on the knowledge of the true ideal and nadir points. Corresponding error



(a) ANN approach (b) GPR approach

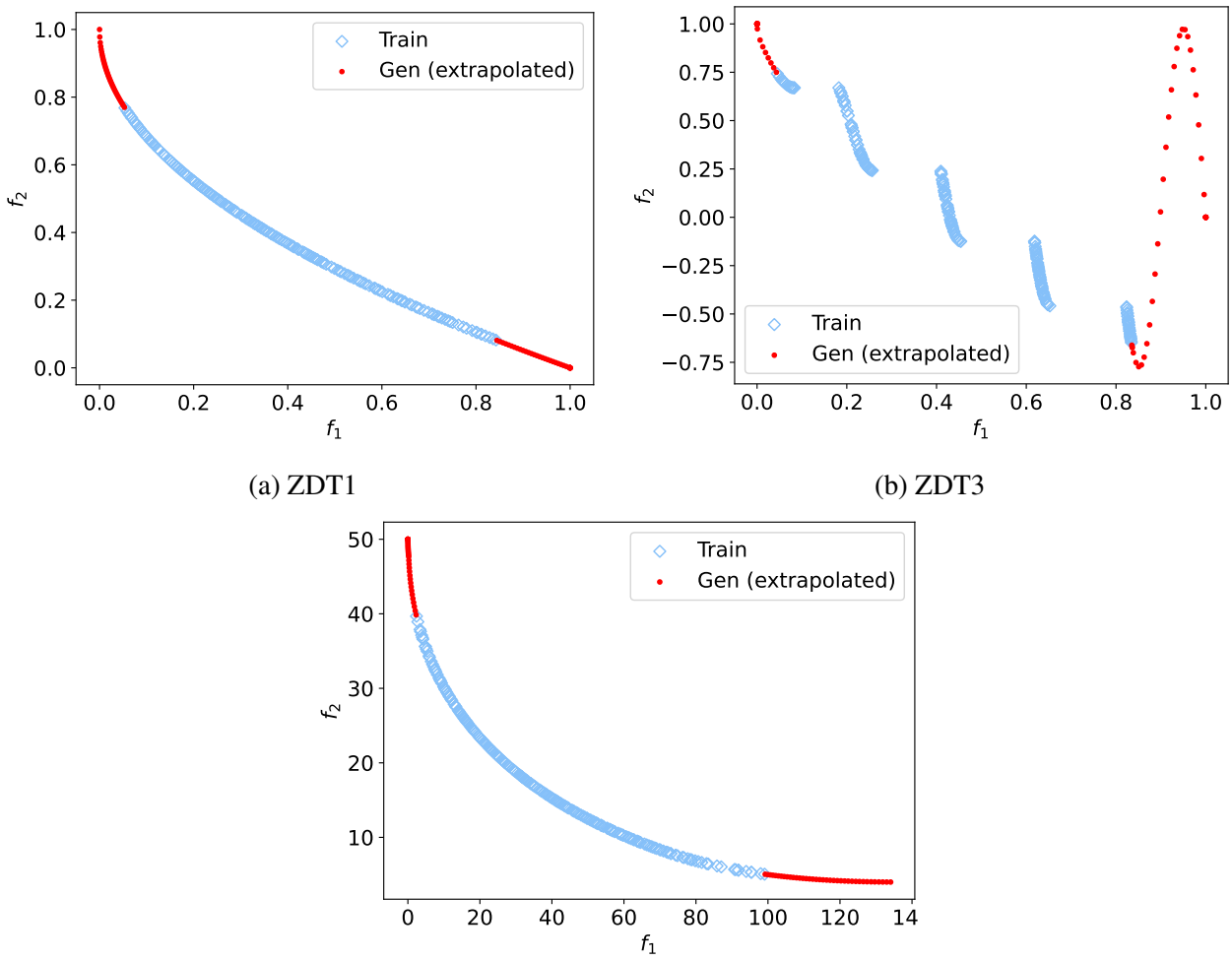
Figure 4.6 ANN and GPR approaches in finding edge gap points on ZDT1 problem, demonstrating Task 2.

values are shown in Table 4.1, 4.2 and 4.3 under the 'Edge' case. Here, only the blue points are used for training the models and the red points are used for testing. This task requires the ML models to learn the relationship between pseudo-weights and **x**-vectors in such a way that they can be extrapolated. This is always a relatively harder task.

4.5.1.3 Validating Task 3

In validating extreme points in Task 2, the true ideal and nadir points were used for computing pseudo-weights. Thus, the test set contained positive pseudo-weights even for extreme points. In practice, the pseudo-weight computation should use the extremes of the training data. Thus, if the true extreme points are not found by an EMO/EMaO algorithm and then if the user wants to identify any un-discovered extreme points during testing, negative **w**-vectors will result, indicating that points to the left and right of the training data are desired. These can be achieved with pseudo-weights $(-w_1, 1 + w_1)$ and $(1 + w_1, -w_1)$, respectively, for $w_1 > 0$ for two-objective problems.

We demonstrate this aspect on ZDT1 and ZDT3 problems where the both extremes of the PO front can be assumed missing in the training data, simulating the case that the EMO/EMaO algorithm could not find these extreme points (Figure 4.7). The pseudo-weight computation and training of ML model are done with the EMO-obtained partial PO solutions, indicating that if the extreme solutions are to be retrieved by the ML model, negative and larger-than-one pseudo-weight values need to be used. We generate a uniform set of pseudo-weights in the range $w_1 \in [0, 0.3]$ so that weights for extreme points lie in $(-w_1, 1 + w_1)$ and $(1 + w_1, -w_1)$, and predict corresponding **x**-vectors. Note that the training of the ML model is done with pseudo-weights in $[0, 1]$. Since the pseudo-weights for testing are negative, the learned GPR model now needs to extend the PO front on both sides by effectively extrapolating. Figure 4.7a shows that the learned GPR model is able to create a few points on either side of the given ND front. Similarly, as seen on Figure 4.7b, new ND points are created on each extreme using the learned GPR model on ZDT3 problem. A part of these right-extreme predicted solutions are non-dominated with the EMO-obtained solutions and the remaining are dominated. We see that this strategy is effective even for problems where the extremes are not easily identified, like BNH, as shown in Figure 4.7c. Using negative pseudo-weights, we are able to extend the PO front on both sides and reach the actual extremes. This clearly shows that our approach can be used to verify whether the extremes of EMO-obtained trade-off set are in fact extremes of the PO front. If the extrapolated **f**-vectors are dominated or infeasible, it can be assumed that there exist no points beyond the given ND-set.



(a) ZDT1

(b) ZDT3

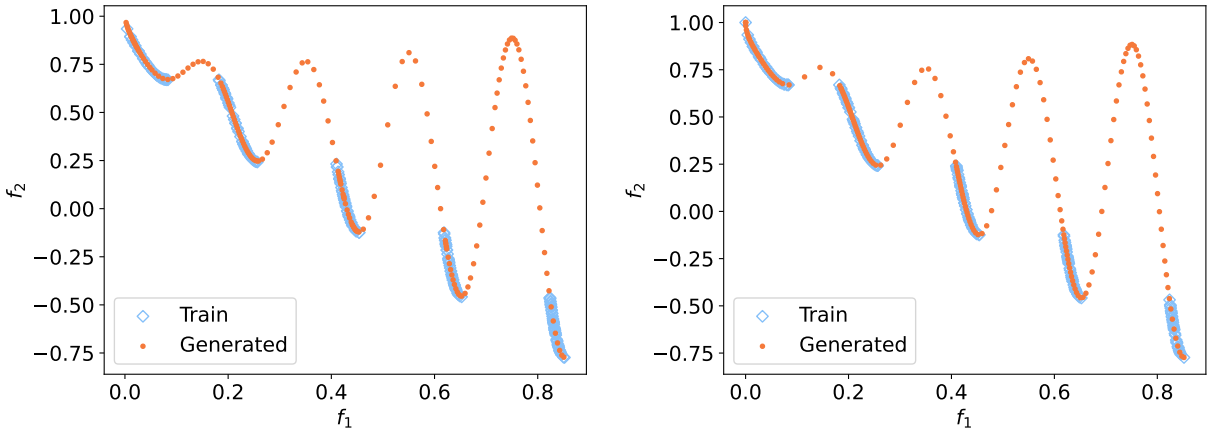
(c) BNH

Figure 4.7 Extrapolation of extremes of the PO front using GPR approach on ZDT1, ZDT3 and BNH problems.

4.5.1.4 Validating Task 4

Some problems, like ZDT3 contain natural gaps in the PO front. Hence, there will be gaps in pseudo-weights computed for the entire PO front. We investigate what our trained ML methods would produce if pseudo-weights from these natural gaps are supplied as inputs. This emulates Tasks 2 and 3, where an EMO run yields PO front with gaps and we wish to understand if the gaps are true gaps or merely a shortcoming of the EMO algorithm. We train both ANN and GPR models on the PO front from ZDT3 problem. We then sample 200 uniformly distributed pseudo-weights as test data and predict their respective **x**-vectors. It is important to note that some of these sampled pseudo-weights lie in the natural gaps of the PO front. Figure 4.8 shows **f**-vectors

produced by evaluating the predicted **x**-vectors. Similar results are observed for the multi-task GPR approach as well. We can observe that the models find the entire PO front (that was used in



(a) ANN on ZDT3

(b) GPR on ZDT3

Figure 4.8 Pseudo-weights on true gaps produce dominated solutions for ZDT3, but pseudo-weights on true PO front produce PO solutions, demonstrating Task 3.

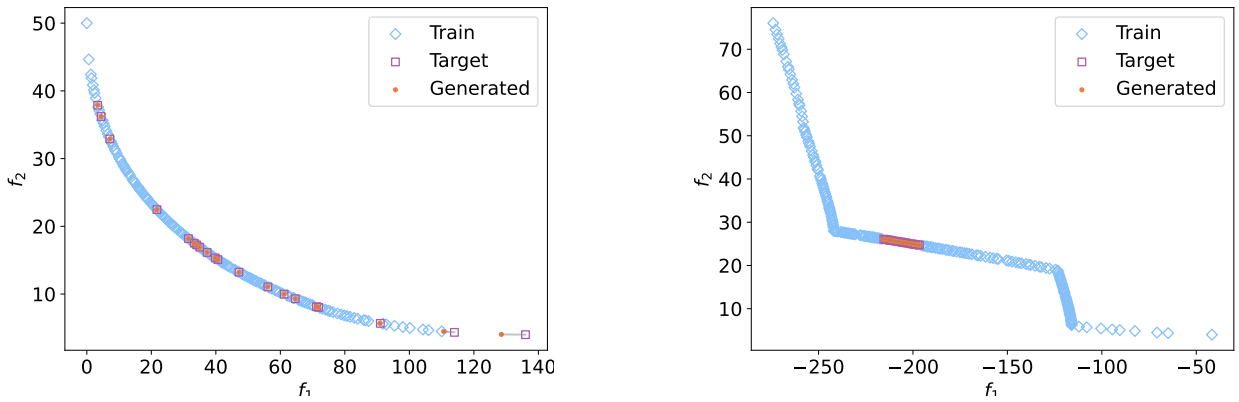
the training) and pseudo-weights corresponding to the natural gaps produce dominated solutions on the lower boundary of the objective space. This demonstrates that the models have learned the low-dimensional manifold where the PO solutions lie and can effectively interpolate between them. Hence, by removing the dominated solutions to the EMO/EMaO solutions, this method can be used to verify the reality of gaps in the EMO-obtained PO front.

4.5.2 Constrained Two-objective Optimization Problems

Tables 4.1 and 4.2 show test set errors for constrained problems, BNH, OSY and TNK. Though both ANN and GPR approaches has low errors, GPR approach performs better for these constrained problems. Figure 4.9 shows that the GPR approach is able to produce the target **f**-vectors for 'Random' and intermediate 'Continuous' case test sets.

4.5.2.1 Validating Task 5

For constrained problems, the **x**-vectors obtained from the learned ML model is checked for their constraint violations. Table 4.4 shows the percentage of predicted solutions that are feasible. For the predicted infeasible solutions, mean of the minimum, average and maximum constraint



(a) Random points on BNH

(b) Continuous gap on OSY

Figure 4.9 GPR results showing the discovery of PO points by the GPR approach on two constrained problems, demonstrating Tasks 1 and 2.

violation values [25, 80] over 31 runs are also presented. It can be observed that for BNH problem, a large number of (close to 100%) test (new) pseudo-weights predict *feasible* solutions on the PO front. However, for OSY, the respective percentage of predicted feasible solutions is small, except for the GPR approach. However, it is interesting that the constraint violation values for predicted infeasible solutions are small, except in the 'Edge' case for OSY, indicating that the ML model is able to learn to produce **x**-vectors close to the active constraint boundaries. Similarly, TNK problem also leads to predicted solutions that are infeasible and yet are close to the active constraint boundary.

4.5.3 Three-objective Optimization problems

Since GPR approach yielded better error values, in general, for two-objective problems, we use only GPR approach for three- and many-objective problems. Table 4.5 shows scaled MAE along with their standard deviations for **x**-vectors and **f**-vectors for the test sets.

Figure 4.10 shows 'Random' and 'Continuous' case test set predictions for DTLZ2 problem using the GPR approach. For the 'Continuous' case, test pseudo-weights are found from the gaps observed in pseudo-wrights found from EMO solutions. We can clearly observe that the GPR model is able to predict **x**-vectors that places **f**-vectors reasonably on the target objective values with low errors. This effectively validates Task 3 and Task 4 for three-objective problems. Similarly, Figure 4.11 shows 'Random' and 'Continuous' case predictions for the car side-impact problem.

Table 4.4 Constraint violation for ML methods for BNH and OSY problems for 'Random', 'Continuous' and 'Edge' cases.

ML	Prob.	Test gap	% Feas.	CV for Infeasible solutions		
				Min.	Avg.	Max.
ANN	BNH	Rand.	100	0.000E+00	0.000E+00	0.000E+00
		Cont.	100	0.000E+00	0.000E+00	0.000E+00
		Edge	99.8	0.000E+00	3.040E-10	6.079E-09
	OSY	Rand.	16.0	3.529E-04	5.868E-02	4.168E-01
		Cont.	15.3	1.054E-02	2.223E-02	3.472E-02
		Edge	9.0	2.607E-01	6.958E-01	1.270E+00
	TNK	Rand.	40.4	4.886E-03	1.697E-02	3.653E-02
		Cont.	24.08	0.000E+00	5.969E-02	1.599E-01
GPR	BNH	Rand.	99.84	0.000E+00	4.904E-06	9.808E-05
		Cont.	100	0.000E+00	0.000E+00	0.000E+00
		Edge	90.8	0.000E+00	5.624E-03	2.582E-02
	OSY	Rand.	63.1	0.000E+00	1.210E-02	1.858E-01
		Cont.	86.3	0.000E+00	8.604E-03	3.651E-02
		Edge	59.8	5.819E-03	8.312E-02	1.549E-01
	TNK	Rand.	72.9	4.798E-07	8.184E-04	5.027E-03
		Cont.	35.9	0.000E+00	4.561E-02	5.778E-01
Multi-task GPR	BNH	Rand.	100	0.000E+00	0.000E+00	0.000E+00
		Cont.	100	0.000E+00	0.000E+00	0.000E+00
		Edge	100	0.000E+00	0.000E+00	0.000E+00
	OSY	Rand.	44.4	0.000E+00	6.099E-02	9.267E-01
		Cont.	6.9	4.558E-03	8.145E-02	1.387E-01
		Edge	56.0	3.843E-02	1.369E-01	4.224E-01
	TNK	Rand.	72.4	8.749E-07	1.076E-03	4.810E-03
		Cont.	51.6	0.000E+00	2.397E-04	1.005E-03

DTLZ7 problem has gaps in its PO front. Figure 4.12 shows 'Random' and 'Continuous' case test points falling on the PO set despite having gaps.

4.5.4 Revisiting Task 4

To visualize the resulting predicted **f**-vector for any test **w**-vector, we generate 1,200 points on the unit simplex and filter them based on maximum values of the **w**-vector in corresponding objectives. This ensures that reasonable number of points are sampled within the relevant region in the unit simplex while avoiding points outside the boundary of the PO front. On executing this step on Crash-worthiness problem, we observe that not all pseudo-weights within the relevant region of the unit simplex result in a PO solution. Figure 4.13 shows that when uniformly distributed

Table 4.5 Performance of GPR approach on unconstrained and constrained three-objective problems.

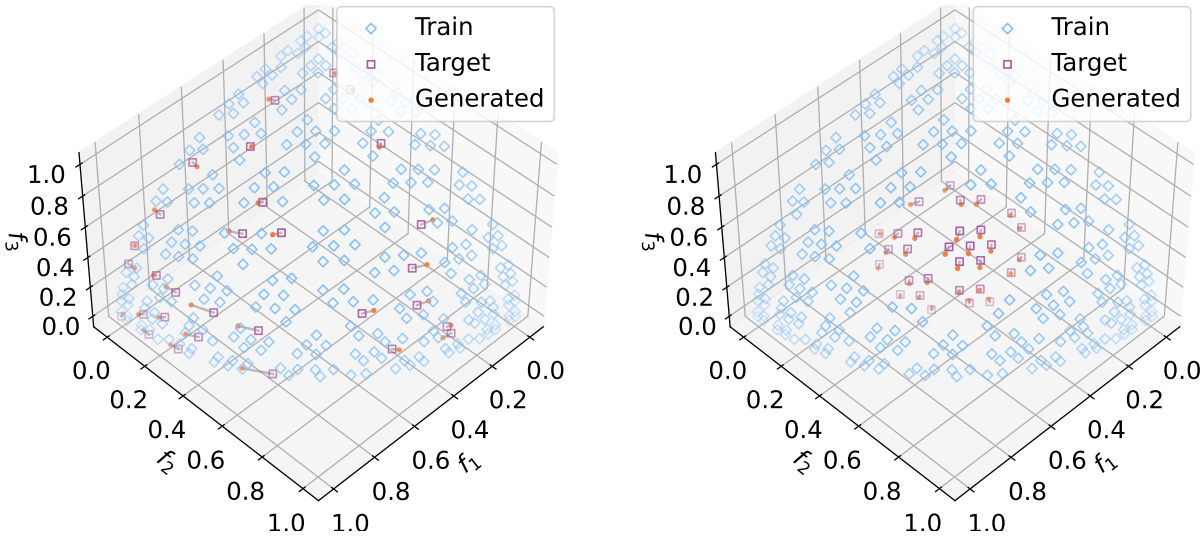
Prob.	M	Test data	Mean MAE in **x**	Mean MAE in **f**	Std. Dev. MAE in **x**	Std. Dev. MAE in **f**
DTLZ2	3	Cont.	4.355E-03	2.149E-02	1.079E-02	1.720E-02
		Edge	8.757E-03	4.873E-02	2.351E-02	5.193E-02
		Random	3.864E-03	1.225E-02	1.753E-02	1.648E-02
		Sparse	3.622E-03	1.889E-02	1.043E-02	2.412E-02
DTLZ5	3	Cont.	2.971E-02	2.701E-02	9.438E-02	1.941E-02
		Random	3.730E-02	6.010E-04	1.146E-01	1.562E-03
DTLZ7	3	Cont.	1.662E-03	8.666E-03	3.619E-03	3.969E-03
		Random	3.414E-04	2.182E-03	1.926E-03	6.275E-03
WFG2	3	Cont.	7.699E-02	9.134E-02	1.240E-01	1.548E-01
		Random	5.725E-02	9.086E-02	1.147E-01	1.542E-01
Carside	3	Cont.	2.321E-02	1.177E-02	4.570E-02	1.122E-02
		Random	1.364E-02	4.416E-03	3.273E-02	4.978E-03
Crash-worthiness	3	Cont.	1.253E-02	9.142E-03	1.653E-02	7.972E-03
		Random	1.129E-02	5.125E-03	3.977E-02	1.170E-02
		Sparse	6.641E-03	4.186E-03	9.921E-03	4.250E-03
C2DTLZ2	3	Cont.	1.777E-03	8.944E-03	4.057E-03	7.167E-03
		Random	2.955E-03	7.658E-03	1.821E-02	1.257E-02

pseudo-weights are tested on the learned ML model, the resulting **x**-vectors which are not on the PO set are dominated, thereby confirming the boundary of the original PO front. The GPR approach is able to correctly associate pseudo-weights with true PO solutions and remaining pseudo-weights with dominated solutions, despite the latter's absence in the training data set.

Similar results are shown on DTLZ7 problem in Figure 4.14. For pseudo-weights inside the gap regions, the learned GPR finds **x**-vectors that are dominated by PO set, confirming the existence of gap regions in the DTLZ7 PO front.

4.5.5 Revisiting Task 5

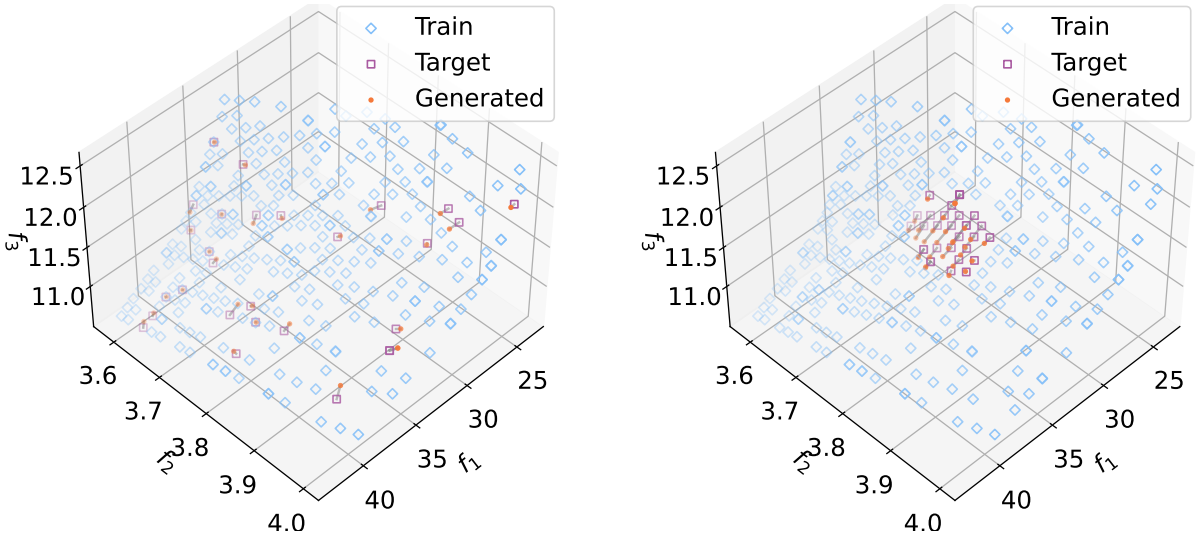
Similar to 4.5.4, 1,200 uniformly distributed unit simplex points are generated and then filtered based on maximum values of pseudo-weights in corresponding objectives. These pseudo-weights are then used to predict **x**-vectors for C2DTLZ2 and car side-impact problems. Figure 4.15a shows predicted solutions on the pseudo-weight space. It can be observed that the predicted **x**-vectors for pseudo-weights specified at the gaps are infeasible, confirming that there are gaps in the original



(a) Random points (Task 1)

(b) Continuous gap (Task 2)

Figure 4.10 Random and continuous gap points found by the GPR approach on three-objective DTLZ2 problem.

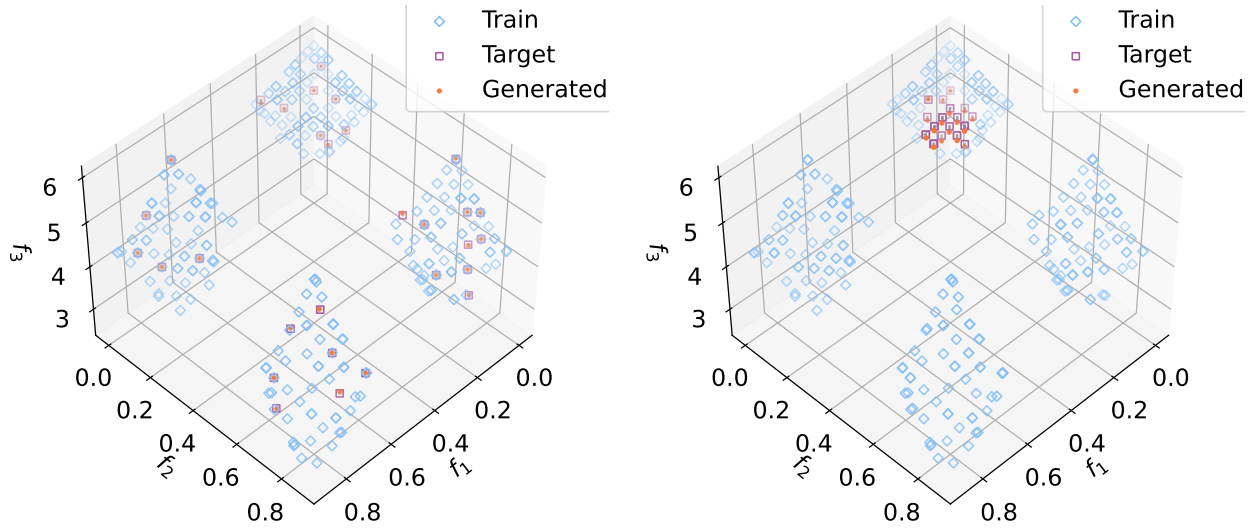


(a) Random points (Task 1)

(b) Continuous gap (Task 2)

Figure 4.11 Random and continuous gap points found by the GPR approach on Car side impact problem.

PO front. A similar observation is made on the car side-impact problem (Figure 4.15b), where almost all produced points are feasible solutions, except a few infeasible points (shown in red) and some dominated yet feasible points (shown in green). The figure shows that the latter points are not overlapping with the original PO solutions (blue points), meaning that these pseudo-weights may not have a corresponding PO solution. But since these pseudo-weights are included during testing,



(a) Random points (Task 1)

(b) Continuous gap (Task 2)

Figure 4.12 Random and continuous gap points found by the GPR approach on three-objective DTLZ7 problem.

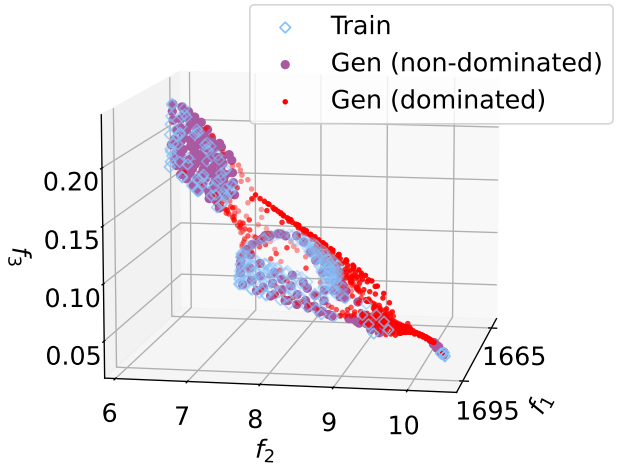


Figure 4.13 Testing with uniformly distributed pseudo-weights reveals that pseudo-weights outside the EMaO-obtained PO front (shown with red points) correctly predict solutions which are *dominated* by the blue PO points for the crash-worthiness problem.

the trained GPR model predicts them as feasible but dominated solutions. These results confirm that our proposed approach is able to learn non-linear mappings between $\mathbf{w}$ and $\mathbf{x}$ in real-world problems and predict infeasible and dominated solutions, as the case may be, for arbitrary pseudo-weights that do not result in any true PO solution. This is a remarkable feature of our proposed

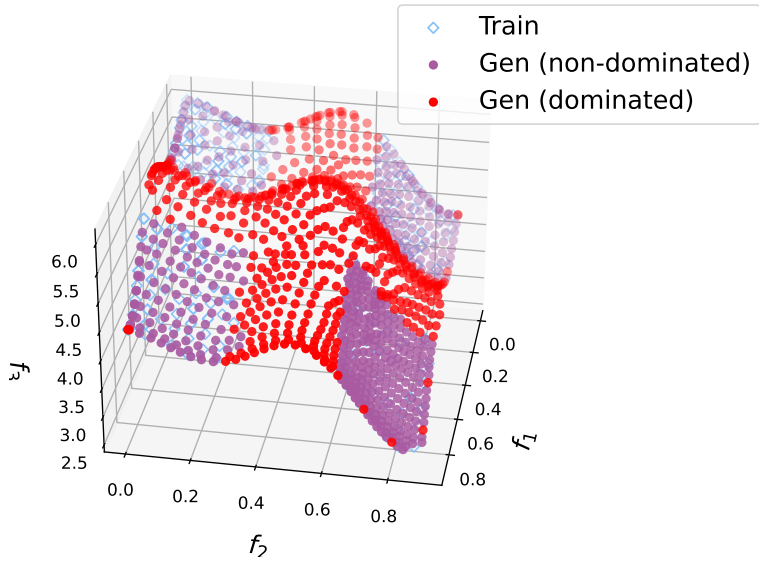


Figure 4.14 Testing of DTLZ7 problem with 1,200 points showing that **x**-vectors in the gap region are dominated, confirming a disjointed PO front.

approach.

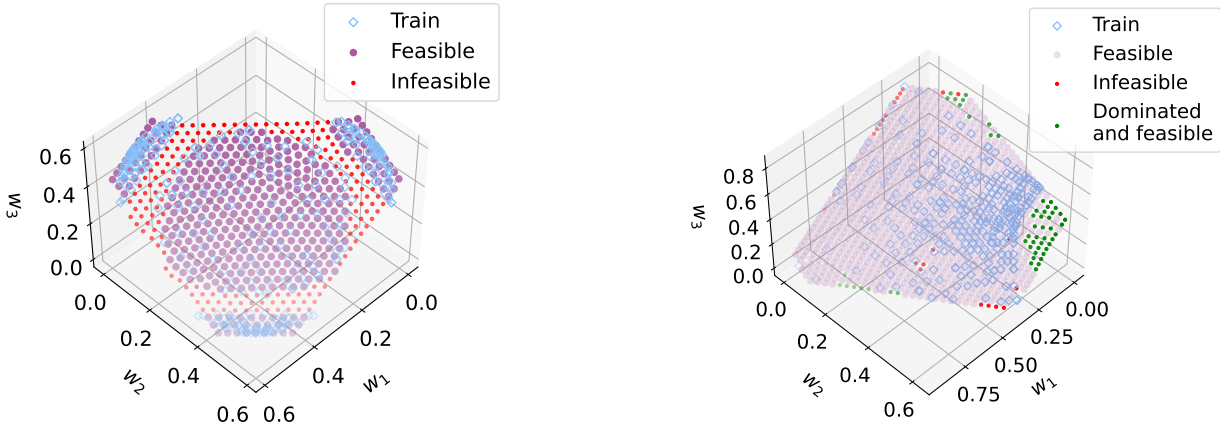
The alternate way of dealing with Task 5 would be to predict feasibility or constraint values from pseudo-weights before actually evaluating the constraint values. For this, we use our proposed *archiving strategy*, described in Section 4.4 to curate a more *complete* dataset. Figure 4.16 shows such an archive collected during NSGA-III execution. We can clearly see that the archive contains solutions in the gaps or missing portions of the PO front containing infeasible solutions and Pareto feasible solutions. The mix of feasible and infeasible solutions in the archive makes a superior dataset for training ML methods to predict feasibility or constraint values. Notice that there exists many infeasible solutions below the region marked by the archive points and they will dominate the PO points. But we ignore these infeasible solutions which dominate the PO points from the archive for training the ML models.

We apply this constraint handling method on two-objective TNK and three-objective C2DTLZ2 problems. This archiving strategy leads to a tractable mapping problem between pseudo-weights and constraint values as shown in Table 4.6. Clearly, constraint values can be predicted with reasonably low errors eliminating the need for high-fidelity evaluation of every desired x-vector. The same datasets are also used for training binary classification models that achieve high cross-

Table 4.6 Performance of constraint prediction.

Problem	M	Mean MAE in constraints	Feasible/infeasible Class Acc %
TNK	2	1.085E-3	91.59
C2DTLZ2	3	6.847E-3	93.03

validation accuracy (more than 90%) for classifying pseudo-weights as feasible or infeasible. Figure 4.17 shows the predictions of the trained models for the C2DTLZ2 problem for uniformly distributed pseudo-weight vectors. Figure 4.17a shows the predicted constraint value where we can clearly observe the *feasible* regions (in the center and middle of the apex of the pseudo-weights). Similarly, Figure 4.17b shows the classification prediction for uniformly distributed pseudo-weights. This clearly demonstrates that these models can eliminate high fidelity evaluation of constraints while evaluating gaps in the PO front.



(a) C2DTLZ2

(b) Carside impact

Figure 4.15 Pseudo-weights of feasible and infeasible solutions generated by prediction of **x** on uniform set of pseudo-weights on two three-objective constrained problems. Feasibility was evaluated after prediction of **x**-vectors

4.5.6 Validating Task 6

Next, we simulate a case in which an EMO run produces a set of PO solutions that are sparse in a specific region, thereby making it challenging from a decision-making perspective. We avoid new solution evaluations by using our GPR approach to produce more PO solutions in the sparse region without resorting to any additional solution evaluations, except to evaluate the predicted

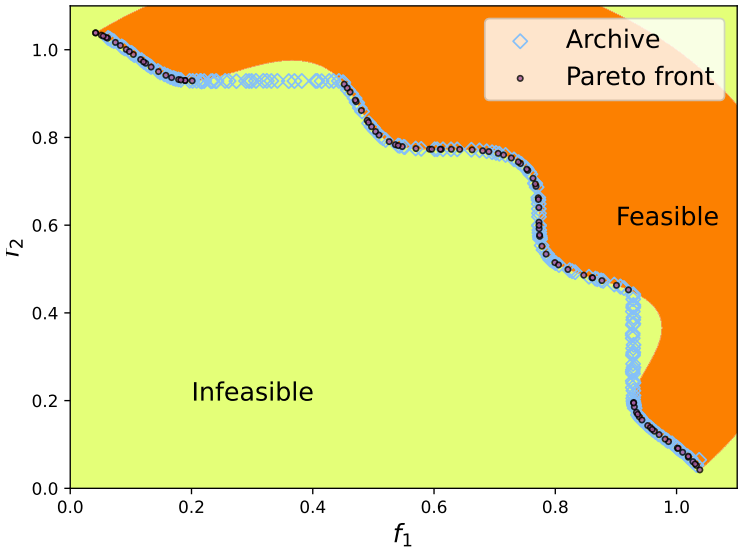
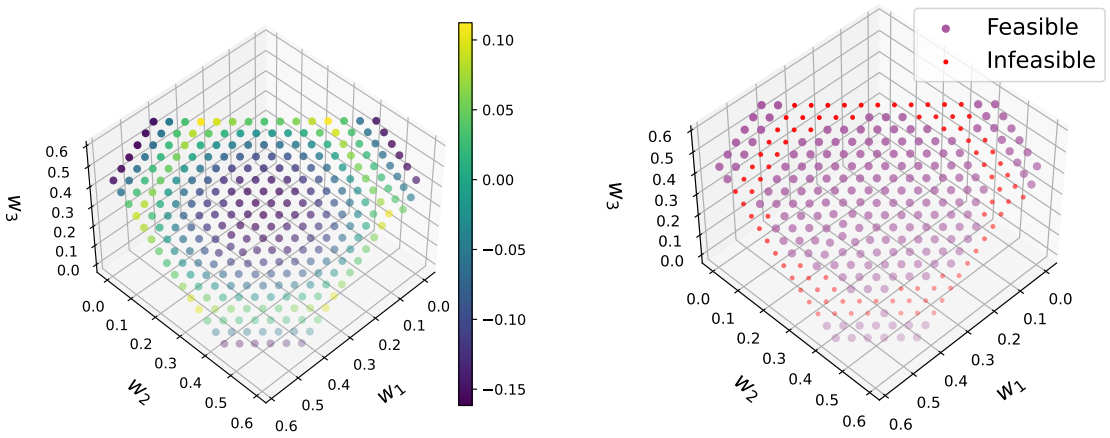


Figure 4.16 Archive of feasible and infeasible ND solutions for the TNK problem. The feasible and infeasible regions of the problem are shown.

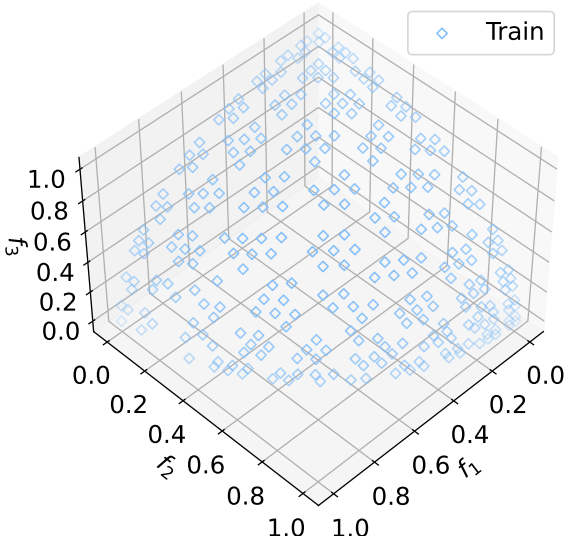


(a) Constraint value prediction, where, the color indicates the predicted constraint function value, $G(\mathbf{x})$.

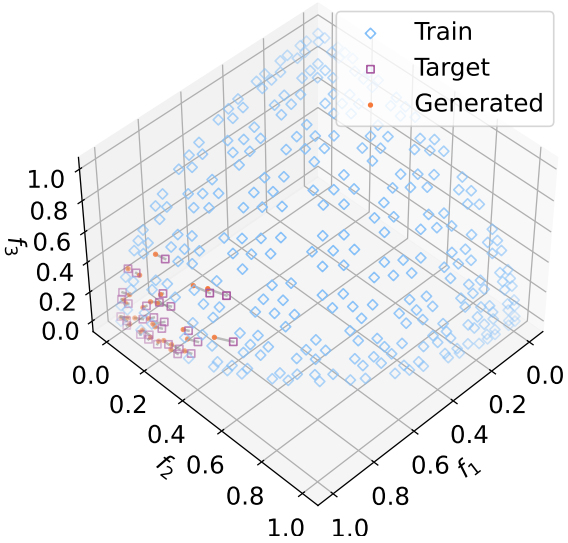
(b) Feasibility prediction, where, the color indicates the binary-classifier's prediction of feasibility.

Figure 4.17 Predicting constraint value and feasibility from pseudo-weight vectors using dataset curated from the archiving strategy on C2DTLZ2 problem.

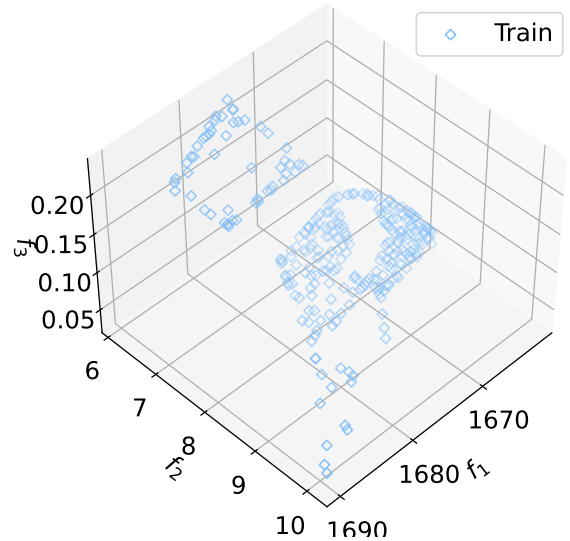
x-vector. Figure 4.18 shows that our GPR approach can effectively fill sparse regions with new PO solutions, as demonstrated on DTLZ2 and crash-worthiness problems.



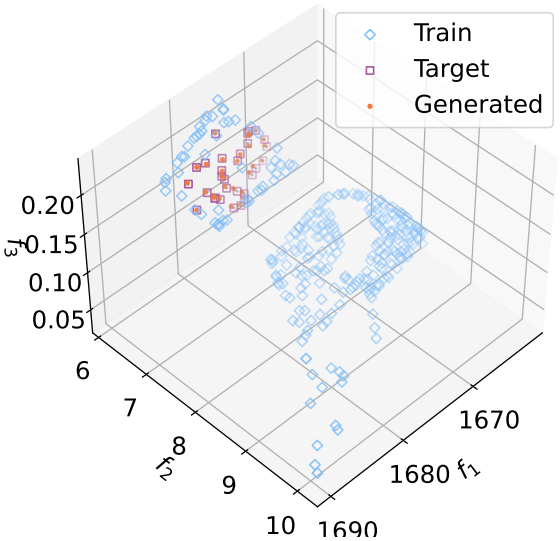
(a) NSGA-III on DTLZ2 finds low density of points on the bottom-left part of the PO front.



(b) GPR-based space filling on DTLZ2



(c) NSGA-III points in the middle of top PO segment are removed deliberately on Crash-worthiness problem.



(d) GPR-based space filling on crash-worthiness.

Figure 4.18 Additional solutions are supplied by the GPR approach for two three-objective problems at region of low-density of solutions, demonstrating Task 4. A few blue points were found by EMO on a part of PO front, but our ML approach has nicely replenished them.

4.5.7 Many-objective Optimization Problems

Finally, we apply our GPR approach to two 5-objective and two 10-objective problems, DTLZ2 and C2DTLZ2 [36], problems to demonstrate the efficacy of our proposed method. Table 4.7 shows the errors in **x**-vectors and subsequent **f**-vectors of the test set. The numerical values of the errors

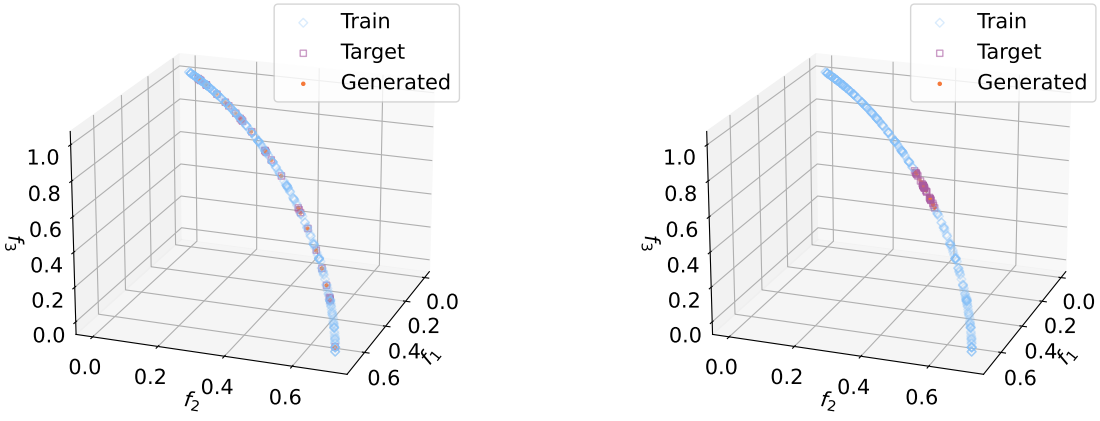
Table 4.7 Performance of GPR approach on unconstrained and constrained many-objective problems.

Prob.	M	Test Data	Mean MAE in **x**	Mean MAE in **f**	Std. Dev. MAE in **x**	Std. Dev. MAE in **f**
DTLZ2	5	Continuous	1.724E-02	4.237E-02	3.744E-02	4.557E-02
		Random	1.094E-02	1.359E-02	4.118E-02	1.907E-02
DTLZ2	10	Continuous	5.895E-02	5.321E-02	1.060E-01	7.066E-02
		Random	5.486E-02	1.698E-02	1.348E-01	3.044E-02
C2DTLZ2	5	Continuous	2.108E-02	5.804E-02	5.003E-02	6.039E-02
		Random	2.099E-02	3.978E-02	7.048E-02	6.056E-02
C2DTLZ2	10	Continuous	4.190E-02	6.027E-02	6.317E-02	6.275E-02
		Random	7.596E-02	5.958E-02	1.508E-01	9.631E-02

suggest that this proposed method is highly effective even for higher objective problems.

4.5.8 Learning in Problems with Degenerate PO fronts

It is important to evaluate the proposed method on problems with degenerate PO fronts with dimensionality lower than $(M-1)$. To understand this better, we use three-objective DTLZ5 as a test problem and evaluate Continuous and Random gap scenarios. As expected, both these cases have a low **x**-error similar to other three-objective problems. Although the dimensionality of the PO front to be learned is lower, the learning problem still remains tractable. Clearly, we can still learn the mapping between pseudo-weights and **x**-vectors.



(a) Random case

(b) Continuous case

Figure 4.19 GPR approach for DTLZ5 for random and continuous test case.

4.5.9 Learning in Problems with Variable Linkages and Scale-up Study

It is important to understand whether the proposed method can be effective in problems with variable linkages and how the efficacy is affected at higher dimensions. Here, since the output of the ML method are the decision variables, we evaluate the scalability of the proposed method with respect to number of decision variables for linked problems. For this task, we use two two-objective problems with variable linkages: L1 and L2 as proposed by Mittal et al. [34]. In Problem L1, all variables, $\mathbf{x}_i$ where $i \in 2, 3 \ldots, n$ are linked to x_1. Similarly, in Problem L2, variables are linked to x_1 depending on the parity of the index of the variable. The two problems are presented below for completeness:

Problem L1:

$$\begin{aligned} \text{Minimize} \quad & f_1(\mathbf{x}) = (1 + g(\mathbf{x}))x_1, \\ \text{Minimize} \quad & f_2(\mathbf{x}) = (1 + g(\mathbf{x}))(1 - x_1^{0.5}), \\ \text{Where} \quad & g(\mathbf{x}) = \textstyle\sum_{i=2}^{n} |x_i - (0.2 + 0.6x_1)^2|^{0.6}, \\ & x_i \in [0, 1], \quad \forall i = 1, 2, \ldots, n. \end{aligned} \tag{4.1}$$

Problem L2:

$$\begin{aligned} \text{Minimize} \quad & f_1(\mathbf{x}) = (1 + g(\mathbf{x}))x_1, \\ \text{Minimize} \quad & f_2(\mathbf{x}) = (1 + g(\mathbf{x}))(1 - x_1^{0.5}), \\ \text{Where} \quad & g(\mathbf{x}) = \textstyle\sum_{i \in I_1} |x_i - (0.2 + 0.6\cos(0.5\pi x_1))| \\ & \quad + \textstyle\sum_{i \in I_2} |x_i - (0.2 + 0.6\sin(0.5\pi x_1))|, \\ & I_1 = \{i | i \text{ is odd and } 2 \le i \le n\}, \\ & I_2 = \{i | i \text{ is even and } 2 \le i \le n\}, \\ & x_i \in [0, 1], \quad \forall i = 1, 2, \ldots, n. \end{aligned} \tag{4.2}$$

We also evaluate the proposed method on a problem where all variables are linked to each other. We achieve this by modifying ZDT1 problem to include a linear transformation of the variables [81] by a full rank matrix $\mathbf{A}$. That is, $\mathbf{y} = \mathbf{A}\mathbf{x}$, where $\mathbf{x}$ is the original variable vector and ZDT1 problem is evaluated on $\mathbf{y}$. The element a_{ij} is sampled from a standard normal distribution. This full rank linear transformation allows us to find the PO-set in the transformed domain by inverting

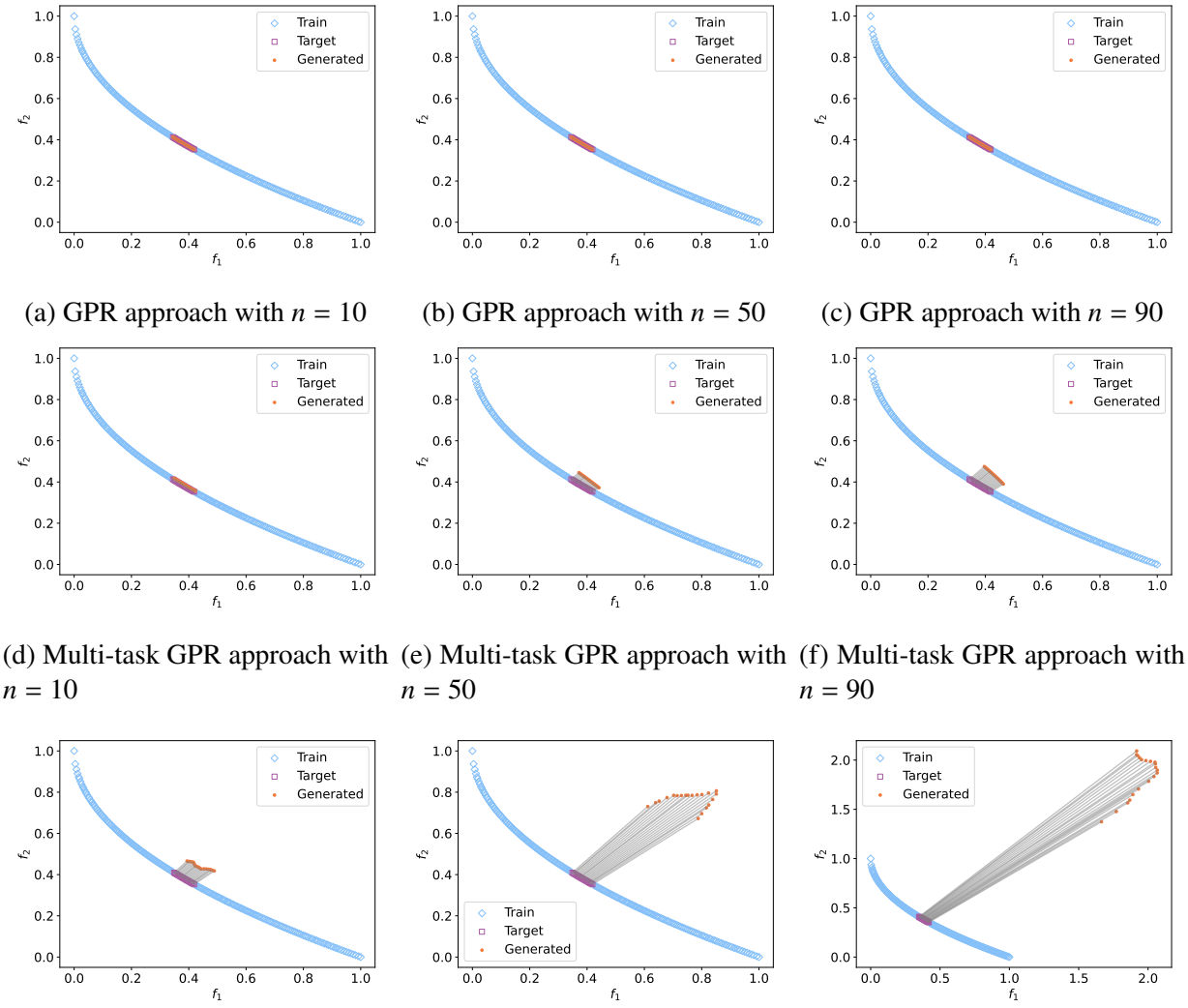
the transformation, $\mathbf{A}^{-1}\mathbf{y} = \mathbf{x}$. The LZDT1 problem is presented below:

$$
\begin{aligned}
&\text{Problem LZDT1:} \\
&\qquad \text{Minimize} \quad f_1(\mathbf{y}) = y_1, \\
&\qquad \text{Minimize} \quad f_2(\mathbf{y}) = g(\mathbf{y})h(f_1(\mathbf{y}), g(\mathbf{y})), \\
&\qquad \text{Where} \quad \mathbf{y} = \mathbf{A}\mathbf{x}, \\
&\qquad\qquad g(\mathbf{y}) = 1 + \tfrac{9}{n-1}\textstyle\sum_{i=1}^{n} y_i, \\
&\qquad\qquad h(f_1, g) = 1 - \sqrt{f_1/g}, \\
&\qquad\qquad 0 \le y_i \le 1, \quad i = 1, 2, \ldots, n.
\end{aligned}
\tag{4.3}
$$

Since EMO methods like NSGA-III with variable-wise Simulated Binary Crossover (SBX) [82] struggle to find a good set of trade-off solutions for these linkage problems, we analytically compute the PO-set and save the data for training and testing.

With respect to variable linkages, vector output models, like ANNs have an advantage as they can learn non-linear relationships among various outputs. However, as seen here, ANNs perform worse than GPR approach. We hypothesize that this is because ANNs require a lot more data to be able to learn a complex mapping as found in the current problem.

Table 4.8 shows the effect of scaling up the number of variables while keeping problem and number of training data-points similar. The table shows **f**-error for different types of modeling approaches. This clearly shows that the GPR performs much better than Multi-task GPR and ANNs. The current data consists of a many-to-one mapping with respect to inputs and outputs. That is, many pseudo-weights might have the same variables as output. Though GPR approach models each output (decision variable) independently and ignores any output correlations, the many-to-one mapping ensures that the mapping to be learned is tractable. An interesting observation is that the **f**-error does not increase with increasing number of variables for the GPR approach. We do not see this trend with ANN and multi-task GPR approaches as seen in Figure 4.20. For the ANN approach, the pseudo-weights created at the gap do not create PO points. The multi-task GPR performs better than ANN approach. This shows that modeling decision variables individually makes the learning problem feasible.



(a) GPR approach with $n = 10$ (b) GPR approach with $n = 50$ (c) GPR approach with $n = 90$

(d) Multi-task GPR approach with $n = 10$ (e) Multi-task GPR approach with $n = 50$ (f) Multi-task GPR approach with $n = 90$

(g) ANN approach with $n = 10$ (h) ANN approach with $n = 50$ (i) ANN approach with $n = 90$

Figure 4.20 Scale up study with a number of variables for different modeling approaches.

4.6 Summary

We demonstrated that the proposed ML assisted MCDM approach is able to create new PO solutions from unseen pseudo-weight vectors. The errors exhibited by these models is extremely low, indicating that this method can safely replace procedures that require new solution evaluations. Among the three ML algorithms considered, GPR performed the best compared to ANN and vector-GPR. We hypothesize that this could be due to the amount of data used in training. We showed that this relationship holds even after scaling up the number of variables. We also showed that the proposed archiving strategy can collect the necessary data to train ML models to predict

Table 4.8 Scale up study on problems with variable linkages.

ML Model	Problem	Number of variables n				
		10	30	50	70	90
GPR	L1	1.30E-05	4.27E-05	7.376E-05	1.037E-04	1.328E-04
	L2	4.514E-09	1.523E-08	2.559E-08	3.593E-08	4.671E-08
	LZDT1	2.731E-09	2.103E-09	4.834E-09	1.604E-09	1.870E-09
Multi-task GPR	L1	4.969E-03	1.824E-02	2.761E-02	4.146E-02	5.220E-02
	L2	3.220E-05	1.291E-04	1.915E-04	2.801E-04	3.489E-04
	LZDT1	2.060E-05	1.536E-05	4.604E-06	1.693E-05	1.787E-05
ANN	L1	8.491E-02	3.767E-01	7.189E-01	1.043E+00	1.311E+00
	L2	2.069E-02	7.601E-02	1.632E-01	2.045E-01	2.455E-01
	LZDT1	6.645E-03	1.142E-02	7.872E-03	1.287E-02	1.081E-02

feasibility from pseudo-weights, potentially avoiding costly and unnecessary function evaluations.

CHAPTER 5

FINDING IDENTIFIER BASED SOLUTIONS WITHIN MOO ALGORITHMS

In this chapter, we present a method for integrating MCDM concepts into optimization and propose an ML-assisted genetic operator that can create new promising candidates and record the data for post-optimal analysis.

5.1 Literature survey

In this section, we briefly introduce some existing literature on using machine learning (ML) as an integral part of MOO.

5.1.1 Online Innovization

Innovization [83] is a process of learning patterns in optimal solutions and using this information to create new promising candidates. Similarly, *online innovization* is a concept where patterns from high-performing solutions are learned during the optimization process from current and previous generations. These methods have been widely explored in literature [34, 35] and have demonstrated that high-performing solutions have patterns both regarding variable relationships as well as the trajectory of optimization.

5.2 Machine learning during optimization

ML methods also aid EMO algorithms to achieve better convergence. ML-based methods, called *innovization*, have been used to recognize patterns in existing solutions and propose meaningful candidate solutions for future iterations. Mittal et al. [34, 35] proposed a series of *innovization* based progress operators that used intra-generational solutions to learn favorable search advances and hence achieve better diversity and convergence. Innovized Progress (IP) operator [34], and IP2 [35] use ML methods to learn directional improvements of solutions in the design variables space. A mapping dataset is created based on the objective values and the same mapping requirements are employed in the decision variable space. Based on existing solutions (enabled by EMO approaches), these ML models learn pertinent directions of progress and can improve the current offspring. These approaches map from **x**-vectors to **x**-vectors, hence are different from the inverse

mapping approach. These IP operators have been further extended to handle extreme solutions and maintain the diversity-convergence balance.

Inverse modeling based methods [32, 33, 70, 71] create new solutions as and when needed during optimization in order to accelerate convergence. Gaspar-Cunha et al. [71] used ANNs to build inverse models based on perturbations in the objective space during optimization to create offspring. Similarly, IM-MOEA based methods used Gaussian Process Regression for this modeling. Though the established theory states that the solutions need to be on the PO front to exhibit regularity principles, the authors frame the problem as a simple regression task and attempt to learn the inverse of objectives. These methods have been continually improved in terms of efficiency using grouping [32], adaptive reference vectors [70] etc. Following the success of IM-MOEA [32], this method has been extended to handle irregular PFs [70] and dynamic environments [84]. Though these methods have shown great success, they are susceptible to failure due to the fact that exact **f**-values need to be known in order to sample the right **x**-variables. Error in the inputs can propagate to errors in outputs. Owing to the fact that the amount of data available is usually low, this error can quickly lead to the failure of the whole method. One generation of an IM-MOEA algorithm is shown in Figure 5.1 where g is the inverse function, X^P and Y^P are decision variables and objectives of the parent population and X^o and Y^o are offspring. Similarly, some methods also parameterize the PF as part of optimization [75]. Some recent works have proposed to learn ML models that can approximate the whole Pareto set, called Pareto Set Learning [85, 86]. The aim of these methods is the final model and is a generalization of decomposition based MOEAs.

5.3 Proposed operator

Current approaches either focus on accelerating and improving convergence or provide tools for quick decision-making post-optimization. This leaves a discernible gap in combining the two aforementioned approaches to have efficient methods that can not only aid in optimization but also assist during the decision-making process. In this study, we propose a new operator that integrates principles from MCDM into optimization. Here, the aim is also to induce confidence in the achieved PO solutions rather than only accelerating the optimization process. We exploit the

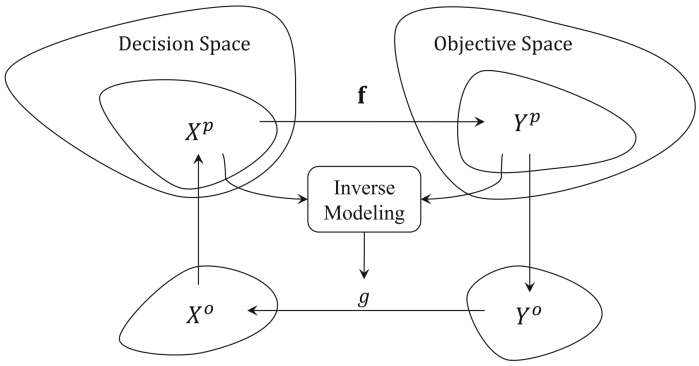


Figure 5.1 One generation of IM-MOEA. Figure adapted from [32].

fact that ML-based steps performed during the decision-making process can be vital in improving the optimization procedure also. Steps such as filling gaps in the PO front and improving diversity of solutions by populating sparse regions in the PO front can be beneficial during the optimization procedure. If these steps are repeatedly performed during optimization and their outcomes are documented for post-processing, there is no further need for analysis at the end of optimization during decision-making. This post-processed data can be presented as auxiliary data to the DM along with the PO solutions and can answers all subsequent questions regarding the PO front. In this study, we propose one such method and show that modifications like this can be performed on any MOEA without compromising the performance of the backbone algorithm. At the end of optimization, these methods provide the PO front along with crucial additional information regarding each RV (both captured and missed by the PO front). This additional data can help the DM make informed decisions regarding the perceived *flaws* of the obtained PO front.

The proposed method aims to streamline the process of MOO and MCDM and eliminate the need for post-optimal analysis and attempts to provide auxiliary information along with the obtained PO front that can answer any subsequent questions a decision-maker might have. This information can now be in the form of tables, plots or merely data for future reference. An operator to be used for this purpose must fulfill the following requirements:

1. **Aid in optimization:** It is crucial that the primary task of any new operator added to algorithms is optimization. This ensures that the act of performing this step along with the

optimization is more efficient than post-optimal analysis.

2. **Auxiliary data:** The operator added to the algorithm needs to create and collect relevant information regarding the PO frontier during the run.

3. **Efficiency:** It is important not to waste solution evaluations with futile attempts to find solutions that do not exist.

The proposed method has three main components: 1)the operator that is employed during the search, 2) tracking progress and stability of the algorithm to decide the use of the operator and 3) post-processing to summarize the collected information for the decision maker.

The proposed ML operators are based on regularity properties of the PO set, that is, the PO set (decision variables) lie on a low dimensional manifold. This property is exploited using ML by learning a mapping between reference-vectors and decision variables. Once these ML models are trained, desired RVs can be given as input to predict the decision variables. However, uncertainty remains regarding the choice of input RVs as they decide the region in which the predicted solutions will lie. These inputs should be chosen in such a way that they aid in optimization while also attempting to fix flaws that are implicitly created during an MOEA run. It is also crucial that solution evaluations are not wasted in irrelevant predicted solutions. By carefully assigning input RVs to these ML models, solutions with varying properties can be achieved. Based on these factors, we propose an ML based operator called the *Fill* operator. This operator uses the current non-dominated solutions as the training data.

5.3.1 Fill operator

The aim of the operator proposed here is to fill gaps in the currently achieved PO front and to improve the *diversity* of the solutions. We achieve this by detecting gaps and attempting to fill them. Conveniently, with reference vector based backbone optimization algorithm, we already have information regarding gaps in the ND front. RVs are associated with solutions during the Survival operation of NSGA-III. RVs that do not have any solutions associated with them can be considered as gaps. An example of identifying such gaps are shown in Figure 5.2a where reference line B does

not have solutions associated with it. Once these gaps are identified, they can be filled as shown in Figure 5.2b. Considerations here are that if the gap is a real gap, that is, if there is no available Pareto solution corresponding to that RV, we would be wasting solutions evaluations and valuable compute time to create dominated or infeasible solutions. Doing this repeatedly can be detrimental to the progress of the algorithm. On the other hand, trying to fill this gap and collecting information regarding the reality of the gap and providing it to the DM can be of use from the perspective of MCDM.

Given the number of solutions to be created, n, the Fill operator involves the following steps:

1. Train the ML model using $(\mathbf{r}, \mathbf{x})$ pairs belonging to non-dominated solutions.

2. Identify RVs that do not have solutions associated with them. Identify n RVs with the furthest associated solutions. Merge these two lists of RVs and select n random RVs to use for prediction.

3. Use the trained ML model to predict $\mathbf{x}$ for these RVs.

We identify two sets of RVs: RVs without solutions (gaps) and RVs with furthest associated solutions to ensure that we do not attempt to fill the same RVs in cases where the gaps are real. In cases where the number of gap RVs are few, this ML operator can still aid in optimization by creating solutions closer to RVs. It is important to note that the RVs assigned during training are computed from $\mathbf{f}$-vectors but the RVs used for creating new solutions are based on RVs from the backbone algorithm.

When these new solutions are added to the Offspring population, remaining offspring are generated using existing genetic operators and then evaluated. The Offspring generation procedure is shown in Algorithm 5.1.

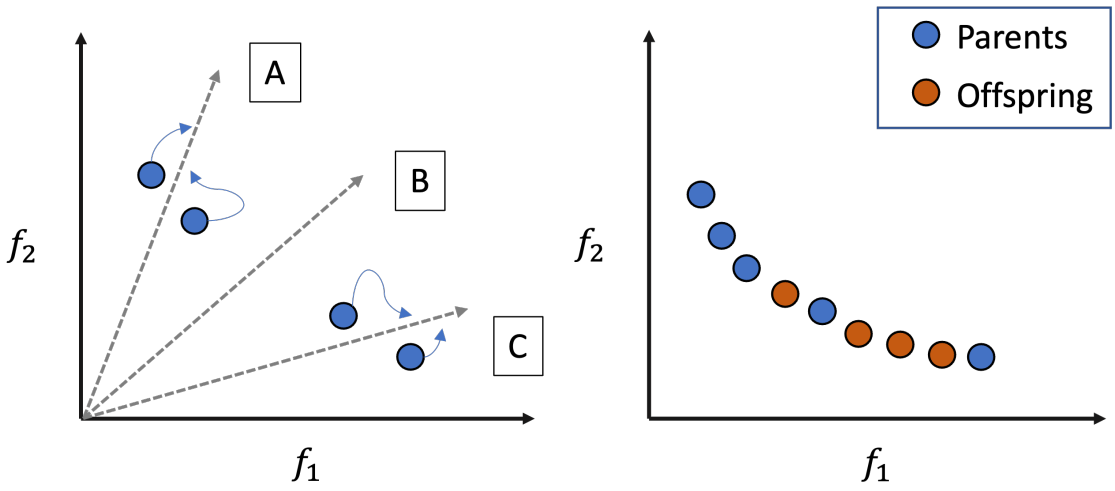
5.3.2 Algorithm tracking

ML based operators that exploit the manifold nature of Pareto set can be used only when the algorithm has reached the PF. If the algorithm is not sufficiently close to the PF, these ML models cannot be trained. Inspired from methods like [3], we also take the approach of letting the backbone

Algorithm 5.1 Offspring

Input: population of ith generation P^i, minimum data for ML N_{min}, movement threshold of ideal point λ

$P_{ND} \leftarrow NDSort(P)$
$n_{ND} \leftarrow len(P_{ND})$
$\Delta z^i_{ideal} \leftarrow z^{(i-1)}_{ideal} - z^{(i)}_{ideal}$
if $n_{ND} \geq N_{min}$ **then**
  if $\Delta z_{ideal} \leq \lambda$ **then**
    $Q_{ML} \leftarrow Fill(P_{ND})$
  end if
end if
$Q_{genetic} \leftarrow GeneticOperators()$
$Q^i \leftarrow Q_{ML} \cup Q_{genetic}$
return Q^i



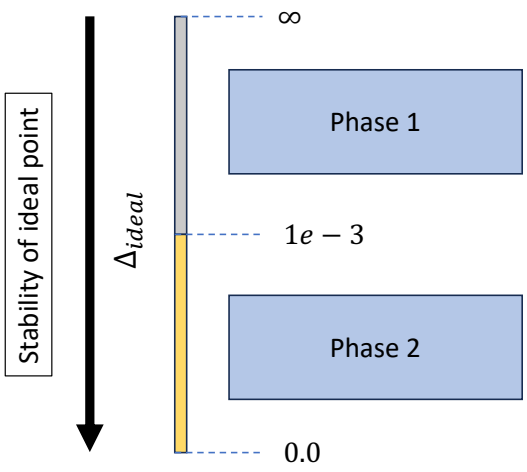
(a) Finding gaps

(b) Fill

Figure 5.2 Fill operator.

algorithm run until stability and then start fixing flaws of the obtained PF. Instead of fixing the number of solution evaluations for stability, we track the movement of ideal point, z_{ideal}, and assume stability and convergence when Δz_{ideal} falls below a certain threshold λ.

The algorithm tracking method is shown in Figure 5.3. Phase 1 is the backbone algorithm in which all offspring are created by the genetic operators (crossover and mutation). When ideal point movement is lower than the set threshold, Phase 2 starts, in which ML-based operator is activated. We choose NSGA-III as the backbone algorithm in the current study but this can be readily replaced with any other MOEA.



(a) Phases

Figure 5.3 Tracking the stability of the ideal point to activate Phase 2.

5.3.3 Postprocessing for auxiliary data

At every generation, we save the reference vectors, decision variables, objective vectors, and constraint values used for training ($\mathbf{r}_{tr}$, $\mathbf{x}_{tr}$, $\mathbf{f}_{tr}$ and $\mathbf{g}_{tr}$) and predicting new solutions ($\mathbf{r}_{ml}$, $\mathbf{x}_{ml}$, $\mathbf{f}_{ml}$ and $\mathbf{g}_{ml}$). Here, $\mathbf{r}_{ml}$ is identified using the FindRV procedure, $\mathbf{x}_{ml}$ is predicted by the ML method and $\mathbf{f}_{ml}$ and $\mathbf{g}_{ml}$ are evaluated objective and constraint values after all offspring are evaluated.

Postprocessing involves the following steps:

- Go back from the last generation until the operator was active and collect tuples of ($\mathbf{r}$, $\mathbf{f}$ and $\mathbf{g}$).
- Assign boolean values to dominated/non-dominated (F_d) and feasible/infeasible (F_{cv}) solutions belonging to different RVs. That is, assign 0 to non-dominated or feasible solutions and 1 to dominated or infeasible solutions.
- Aggregate (average) these values based on the input RVs.
- Amend domination (F_d) and feasibility (F_{cv}) values for solutions close to training data. Here, we eliminate error from data that are close to training samples as ground truth is known.

Now, at the end of the post-processing, these tuples of ($\mathbf{r}$, F_d and F_{cv}) can be visualized to

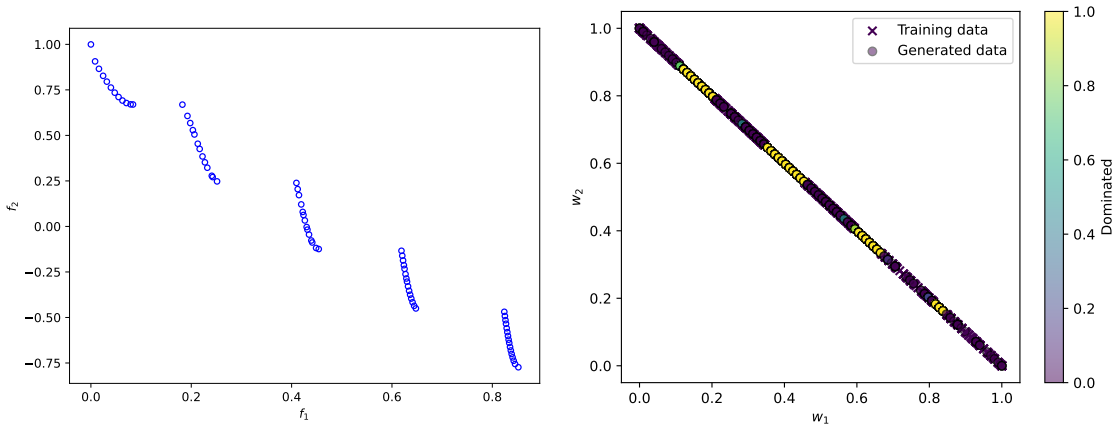
analyze the PF along with the flaws. RVs belonging to gaps will now contain aggregated values based on ML predicted solutions in different generations.

5.4 Results

We include this operator with NSGA-III and run experiments on a number of problems from DTLZ[55], ZDT[77], WFG[56] and MW[87] problem suites . We run each experiment 11 times and report median results along with statistical significance between baseline NSGA-III and ML-NSGA-III. We use population size of 100 and run upto 30,000 solution evaluations for ZDT and DTLZ problems and 60,000 solution evaluations for MW and WFG problems. We use GPR based ML models with λ (threshold for Δz_{ideal}) of 1e-3. We perform Wilcoxon ranksum testing to check for statistically significant difference between the algorithms.

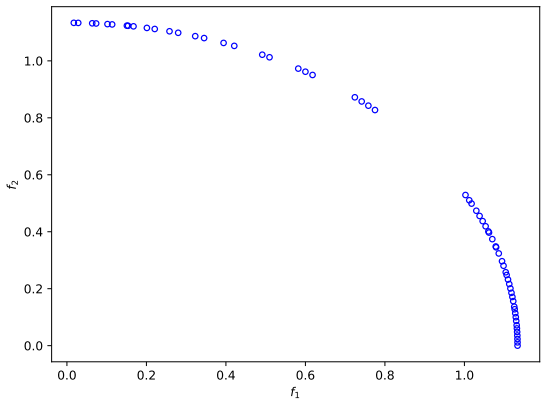
Median hypervolumes are shown in Table 5.1. Here, statistically better values are bolded and statistically similar values are italicized. Here, we see that for most problems (12 of 17), baseline NSGA-III and ML-NSGA-III have statistically similar final performances. This is indicative of the fact the proposed operator is not hindering optimization performance. On the other hand, proposed method is worse than baseline for 3 of 17 problems and better than baseline for 2 of 17 problems. This shows that for some problems, the process of collecting auxiliary data comes at a cost.

We present some postprocessed results from ML-NSGA-III in Figure 5.4. Here, Figure 5.4a, Figure 5.4c and Figure 5.4e show median PFs achieved during ML-NSGA-III optimization. Figure 5.4b shows RVs along with their domination criterion at the end of optimization for ZDT3 problem. We can see the corresponding gaps from the PF and interpret the meaning of the gaps. We can observe that the gaps are real and that the solutions belonging to those RVs are dominated. We can make similar observations from Figure 5.4d and Figure 5.4f that the gaps are real and that the solutions belonging to the gaps are infeasible. If the DM is provided a PF with gaps as shown in Figure 5.4a, Figure 5.4c and Figure 5.4e, the DM will want to know more about the gaps and discontinuities. The auxiliary information shown in Figure 5.4 can answer some of the concerns.

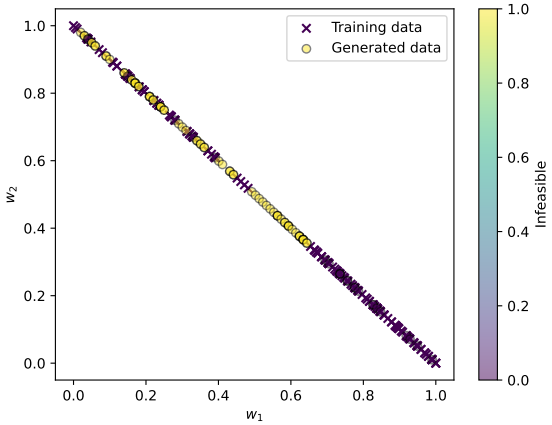


(a) ZDT3 - PF

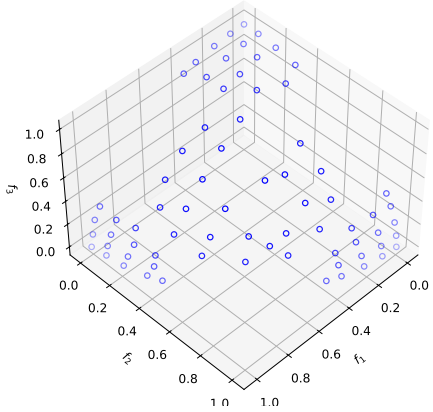
(b) ZDT3 - Domination



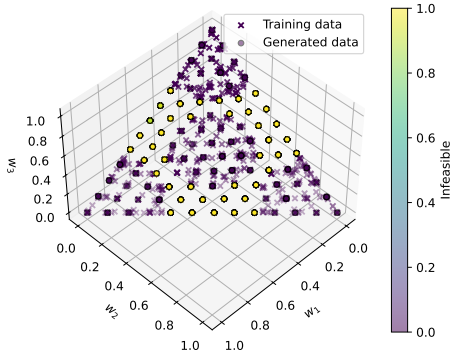
(c) MW6 - PF



(d) MW6 - Feasibility



(e) C2DTLZ2 - PF



(f) C2DTLZ2 - Feasibility

Figure 5.4 PF and identifiers achieved using combined RVs with NSGA-III.

5.5 Summary

In this chapter, we presented a ML-assisted operator that can be implemented with any MOEA. The operator uses ML to attempt to rectify flaws of the PF and saves the data. This data can be

Table 5.1 Median hypervolumes of 2 and 3 objective problems.

	M	problem	NSGA-III	ML-NSGA-III
1	2	ZDT2	*4.31E-01*	*4.31E-01*
2	2	ZDT3	*6.17E-01*	*6.17E-01*
3	2	MW5	*2.53E-01*	*2.52E-01*
4	2	MW6	**2.46E-01**	2.40E-01
5	2	MW7	*3.89E-01*	*3.88E-01*
6	2	MW13	*3.83E-01*	*3.83E-01*
7	3	DTLZ1	*9.44E-01*	*9.44E-01*
8	3	DTLZ2	*5.74E-01*	*5.74E-01*
9	3	DTLZ7	4.18E-01	**4.19E-01**
10	3	C2DTLZ2	*5.07E-01*	*5.06E-01*
11	3	WFG2	*1.06E+00*	*1.05E+00*
12	3	WFG4	**5.53E-01**	5.36E-01
13	3	WFG5	5.14E-01	**5.16E-01**
14	3	WFG7	*5.69E-01*	*5.62E-01*
15	3	WFG8	*4.75E-01*	*4.65E-01*
16	3	WFG9	*4.62E-01*	*5.10E-01*
17	3	MW14	**4.88E-01**	4.85E-01

post-processed to provide auxiliary information to the DM along with the PF. Now, if there are gaps or discontinuities in the PF, the auxiliary data can be used to understand the reason for the gaps thereby making this more convenient than performing post-optimal analysis.

CHAPTER 6

CONCLUSIONS AND FUTURE DIRECTIONS

6.1 Conclusions

This dissertation looks at the process of multi-objective optimization and multi-criterion decision-making in a comprehensive manner and proposes methods for convenient execution of the two steps.

Decision makers look for trade-offs and relative importance of objective values while evaluating obtained Pareto solutions. These solutions can be visualized and analyzed in alternate domains, called unique identifiers. These identifiers are designed to capture the inherent properties of the Pareto front and convey them to the DM. We defined the requirements for these unique identifiers and compared and discussed the advantages of different choices of identifiers from the perspective of optimization, visualization and decision-making. Based on the discussion, we propose guidelines on the choice of identifier based on preferences of the decision maker or prior information regarding the problem. Ideal point based RVs can be used to identify knee-like solutions with more dense points or if the DM is interested in the middle part of a convex-like PF, or uniform distribution in a concave PF that falls on the unit simplex. Nadir RV identifiers can be useful when the PF is inverted or if the DM is interested in uniform solutions in convex PFs. Projection based RVs can provide a middle ground between the two and help achieve a well spread out set of solutions in the objective space. Pseudo-weights allows the DM to set priorities or relative importance between objectives and can be vital in cases where the objective values themselves do not matter but only the preferences. Similarly, angle vectors can be used in cases in which trade-offs with specific objectives must to be analyzed. Both pseudo-weights and angle vectors eliminate the need to visualize or understand the geometry of the PF and hence can be helpful in problems with higher number of objectives when traditional visualization techniques cannot be used.

Based on the understanding of these identifiers, the DM might want to perform the decision-making process in one of these spaces. Then, it is prudent to provide the DM with a good distribution of solutions in these spaces. We propose modifications to NSGA-III that can help

achieve a good distribution of solutions in these identifier spaces that could be very useful during decision-making. We show that Riesz Energy method can be used to obtain reference identifiers in these spaces that could be used to obtain PFs that are oddly distributed in the objective space but are better suitable for DM. We also present a method to achieve a uniform distribution of solutions in multiple identifiers by combined optimization of Riesz Energy in the overlapping regions. This can be useful in cases where multiple identifiers need to be used for DM.

While these unique identifiers can help in a convenient MCDM process, we cannot guarantee that the obtained PF will be complete or uniformly distributed due to the complexities of the optimization problem at hand. Unexpected gaps or sparse regions could severely hinder the MCDM experience. To alleviate some of these issues, we proposed a machine learning-based MCDM method that can be used to create new PO solutions based on user demands by learning the mapping between pseudo-weights and PO-set. Apart from this, we also demonstrated that a similar ML model can be trained to predict the feasibility of solutions using only pseudo-weights for constrained problems. This can be used to filter out solutions before evaluating them. In order to curate the dataset for training the feasibility models, we proposed an archiving strategy that can collect the required data during an EMO run without extra solution evaluations. This was based on the hypothesis that infeasible yet non-dominated solutions (on the same manifold as PO front) are, in fact, created by the evolutionary operators but do not survive the evolutionary optimization procedure. Once these machine learning models are trained, they can be used to:

1. create new solutions as desired by the DM
2. fill apparent gaps and verify the existence of gaps
3. fill sparse regions for decision-making and visualization
4. try and extend the PO front on the desired sides to gain confidence in the obtained solutions
5. predict feasibility of solutions in unoccupied regions on PO front

We also propose an online *innovization* method that can conveniently answer some MCDM concerns at the end of optimization without further analysis. We exploit the idea that common MCDM concerns such as filling gaps or populating sparse regions are also valuable steps during the optimization process. We proposed an operator that can be included with any MOEA that uses ML-based methods to check for MCDM requirements during optimization and stores the results of these steps. This data is postprocessed and provided to the decision maker as auxiliary information. This can cater to subsequent concerns of the DM such as reason for gaps in the PF and can potentially eliminate the need for post optimal analysis.

6.2 Future directions

The current work can be extended in the following ways:

- More identifiers must be developed to improve the MCDM process. For example, an extreme-proximity identifier space in which an objective vector's Euclidean distance to best objective point of each of M objectives can be used a direct metric for decision-making preference. In some sense, this metric will be be opposite in sense to the pseudo-weight identifier space. While the current study focused on simple visualization techniques, more identifiers need to be developed that can be used in tandem with advanced visualization techniques such as PaletteViz [11], parallel coordinate plots, and others, for achieving more pragmatic DM-aided PO solutions.

- The convergence behavior of optimizing with different identifiers should be studied as these varying properties might affect the evolution process. Sensitivity of these mapping functions (to and from identifiers) can have an impact on the optimization trajectory.

- Adaptively switching between identifiers based on intermediate solution can be interesting for cases where the DM also does not know the suitable identifier space for DM. Since there is a relationship between problem (PF shape) and identifiers, it is pragmatic to develop methods that can ascertain the suitable identifier during optimization.

- The next step for ML-assisted MCDM would be to improve the method to be functional with much less data and scalable to more objectives and variables. It might also be interesting to perform this in other data structures such as Finite Element mesh data.
- Finally, these integrated MCDM and MOO methods can be extended to be interactive in nature such that the DM can decide the future trajectory of optimization from some intermediate point.

BIBLIOGRAPHY

[1] R. Cheng, Y. Jin, M. Olhofer, and B. Sendhoff, "A reference vector guided evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 5, pp. 773–791, 2016.

[2] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: Solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2013.

[3] K. Deb, C. L. d. V. Lopes, F. V. C. Martins, and E. F. Wanner, "Identifying pareto fronts reliably using a multistage reference-vector-based framework," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 1, pp. 252–266, 2024.

[4] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[5] H. Seada, M. Abouhawwash, and K. Deb, "Towards a better balance of diversity and convergence in nsga-iii: First results," in *Evolutionary Multi-Criterion Optimization: 9th International Conference, EMO 2017, Münster, Germany, March 19-22, 2017, Proceedings 9*, pp. 545–559, Springer, 2017.

[6] H. Seada and K. Deb, "A unified evolutionary optimization procedure for single, multiple, and many objectives," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 358–369, 2015.

[7] Q. Zhang and H. Li, "MOEA/D: a multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[8] K. Miettinen and M. Mäkelä, "Interactive bundle-based method for nondifferentiable multiobjective optimization: Nimbus," *Optimization*, vol. 34, no. 3, pp. 231–246, 1995.

[9] D. Yadav, D. Nagar, P. Ramu, and K. Deb, "Visualization-aided multi-criteria decision-making using interpretable self-organizing maps," *European Journal of Operational Research*, vol. 309, no. 3, pp. 1183–1200, 2023.

[10] A. Ibrahim, S. Rahnamayan, M. V. Martin, and K. Deb, "3D-RadVis: Visualization of pareto front in many-objective optimization," in *2016 IEEE Congress on Evolutionary Computation (CEC)*, pp. 736–745, IEEE, 2016.

[11] A. K. A. Talukder and K. Deb, "Paletteviz: A visualization method for functional understanding of high-dimensional pareto-optimal data-sets to aid multi-criteria decision making," *IEEE Computational Intelligence Magazine*, vol. 15, no. 2, pp. 36–48, 2020.

[12] K. Miettinen, *Nonlinear Multiobjective Optimization*. Boston: Kluwer, 1999.

[13] K. Deb, A. Sinha, P. Korhonen, and J. Wallenius, "An interactive evolutionary multi-objective optimization method based on progressively approximated value functions," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 5, pp. 723–739, 2010.

[14] M. Kadziński, M. Tomczyk, and R. Słowiński, "Preference-based cone contraction algorithms for interactive evolutionary multiple objective optimization," *Swarm and Evolutionary Computation*, vol. 52, p. 100602, 2020.

[15] Y. Tian, R. Cheng, X. Zhang, Y. Su, and Y. Jin, "A strengthened dominance relation considering convergence and diversity for evolutionary many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 331–345, 2018.

[16] K. Ikeda, H. Kita, and S. Kobayashi, "Failure of Pareto-based MOEAs: Does non-dominated really mean near to optimal?," in *Proceedings of IEEE Congress on Evolutionary Computation*, pp. 957–962, 2001.

[17] L. Chen, H.-L. Liu, and K. C. Tan, "Decomposition based dominance relationship for evolutionary many-objective algorithm," in *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–6, IEEE, 2017.

[18] M. Farina and P. Amato, "A fuzzy definition of "optimality" for many-criteria optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 34, no. 3, pp. 315–326, 2004.

[19] X. Zou, Y. Chen, M. Liu, and L. Kang, "A new evolutionary algorithm for solving many-objective optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 38, no. 5, pp. 1402–1412, 2008.

[20] H. Ishibuchi, Y. Setoguchi, H. Masuda, and Y. Nojima, "Performance of decomposition-based many-objective algorithms strongly depends on pareto front shapes," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 169–190, 2016.

[21] R. Denysiuk, L. Costa, I. Espírito Santo, and J. C. Matos, "MOEA/PC: Multiobjective evolutionary algorithm based on polar coordinates," in *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part I 8*, pp. 141–155, Springer, 2015.

[22] H. Sato, "Inverted PBI in MOEA/D and its impact on the search performance on multi and many-objective optimization," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pp. 645–652, 2014.

[23] Z. Wang, Q. Zhang, H. Li, H. Ishibuchi, and L. Jiao, "On the use of two reference points in decomposition based multiobjective evolutionary algorithms," *Swarm and Evolutionary Computation*, vol. 34, pp. 89–102, 2017.

[24] Q. Zhang, W. Zhu, B. Liao, X. Chen, and L. Cai, "A modified PBI approach for multi-objective optimization with complex pareto fronts," *Swarm and Evolutionary Computation*, vol. 40, pp. 216–237, 2018.

[25] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Chichester, UK: Wiley, 2001.

[26] K. Deb and J. Sundar, "Reference point based multi-objective optimization using evolutionary algorithms," in *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pp. 635–642, 2006.

[27] Y. Vesikar, K. Deb, and J. Blank, "Reference point based NSGA-III for preferred solutions," in *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1587–1594, IEEE, 2018.

[28] P. V. Pellicer, M. I. Escudero, S. F. Alzueta, and K. Deb, "Gap finding and validation in evolutionary multi-and many-objective optimization," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pp. 578–586, 2020.

[29] C. Hillermeier, *Nonlinear multiobjective optimization: a generalized homotopy approach*, vol. 135. Springer Science & Business Media, 2001.

[30] A. Asilian Bidgoli, S. Rahnamayan, B. Erdem, Z. Erdem, A. Ibrahim, K. Deb, and A. Grami, "Machine learning-based framework to cover optimal pareto-front in many-objective optimization," *Complex & Intelligent Systems*, vol. 8, no. 6, pp. 5287–5308, 2022.

[31] I. Giagkiozis and P. J. Fleming, "Pareto front estimation for decision making," *Evolutionary Computation*, vol. 22, no. 4, pp. 651–678, 2014.

[32] R. Cheng, Y. Jin, K. Narukawa, and B. Sendhoff, "A multiobjective evolutionary algorithm using gaussian process-based inverse modeling," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 6, pp. 838–856, 2015.

[33] L. R. Farias and A. F. Araújo, "IM-MOEA/D: an inverse modeling multi-objective evolutionary algorithm based on decomposition," in *2021 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 462–467, 2021.

[34] S. Mittal, D. K. Saxena, K. Deb, and E. D. Goodman, "A learning-based innovized progress operator for faster convergence in evolutionary multi-objective optimization," *ACM Transactions on Evolutionary Learning and Optimization*, vol. 2, nov 2021.

[35] S. Mittal, D. K. Saxena, K. Deb, and E. D. Goodman, "Enhanced innovized progress operator for evolutionary multi- and many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 5, pp. 961–975, 2022.

[36] H. Jain and K. Deb, "An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 602–622, 2013.

[37] A. P. Wierzbicki, "The use of reference objectives in multiobjective optimization," in *Multiple criteria decision making theory and application: Proceedings of the third conference Hagen/Königswinter, West Germany, August 20–24, 1979*, pp. 468–486, Springer, 1980.

[38] J. Blank and K. Deb, "Pymoo: Multi-objective optimization in python," *IEEE Access*, vol. 8, pp. 89497–89509, 2020.

[39] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*, pp. 8024–8035, 2019.

[40] J. Gardner, G. Pleiss, K. Q. Weinberger, D. Bindel, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[41] M. Balandat, B. Karrer, D. R. Jiang, S. Daulton, B. Letham, A. G. Wilson, and E. Bakshy, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," in *Advances in Neural Information Processing Systems 33*, 2020.

[42] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 2623–2631, 2019.

[43] J. Blank, "anyoptimization/ezmodel: A Common Interface for Models and Model Selection."

[44] A. Suresh and K. Deb, "Identifier spaces for representing pareto-optimal solutions in multi-objective optimization and decision-making," *Engineering Optimization*, vol. 57, no. 1, 2025.

[45] D. Kuang and J. Zheng, "Strategies based on polar coordinates to keep diversity in multi-objective genetic algorithm," in *2005 IEEE Congress on Evolutionary Computation*, vol. 2, pp. 1276–1281, IEEE, 2005.

[46] C. He, Y. Tian, Y. Jin, X. Zhang, and L. Pan, "A radial space division based evolutionary algorithm for many-objective optimization," *Applied Soft Computing*, vol. 61, pp. 603–621, 2017.

[47] A. Habib, H. K. Singh, T. Chugh, T. Ray, and K. Miettinen, "A multiple surrogate assisted decomposition-based evolutionary algorithm for expensive multi/many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 1000–1014, 2019.

[48] Y. Liu, H. Ishibuchi, N. Masuyama, and Y. Nojima, "Adapting reference vectors and scalarizing functions by growing neural gas to handle irregular pareto fronts," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 3, pp. 439–453, 2019.

[49] X. Ma, Y. Yu, X. Li, Y. Qi, and Z. Zhu, "A survey of weight vector adjustment methods for decomposition-based multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 4, pp. 634–649, 2020.

[50] Y. Qi, X. Ma, F. Liu, L. Jiao, J. Sun, and J. Wu, "MOEA/D with adaptive weight adjustment," *Evolutionary Computation*, vol. 22, no. 2, pp. 231–264, 2014.

[51] P. Hoffman, G. Grinstein, and D. Pinkney, "Dimensional anchors: a graphic primitive for multidimensional multivariate information visualizations," in *Proceedings of the 1999 workshop on new paradigms in information visualization and manipulation in conjunction with the eighth ACM internation conference on Information and knowledge management*, pp. 9–16, 1999.

[52] I. Das and J. E. Dennis, "Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems," *SIAM J. on Optimization*, vol. 8, no. 3, pp. 631–657, 1998.

[53] J. Blank, K. Deb, Y. Dhebar, S. Bandaru, and H. Seada, "Generating well-spaced points on a unit simplex for evolutionary many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 25, no. 1, pp. 48–60, 2020.

[54] M. Elarbi, S. Bechikh, C. A. C. Coello, M. Makhlouf, and L. B. Said, "Approximating complex pareto fronts with predefined normal-boundary intersection directions," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 5, pp. 809–823, 2019.

[55] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable test problems for evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, pp. 105–145, Springer, 2005.

[56] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[57] R. Cheng, M. Li, Y. Tian, X. Zhang, Y. Jin, and X. Yao, "A benchmark test suite for evolutionary many-objective optimization," *Complex and Intelligent Systems*, vol. 3, pp. 67–81, 2017.

[58] L. Gu, R. Yang, C.-H. Tho, M. Makowskit, O. Faruquet, and Y. L. Y. Li, "Optimisation and robustness for crashworthiness of side impact," *International Journal of Vehicle Design*, vol. 26, no. 4, pp. 348–360, 2001.

[59] X. Liao, Q. Li, X. Yang, W. Zhang, and W. Li, "Multiobjective optimization for crash safety design of vehicles using stepwise regression model," *Structural and Multidisciplinary Optimization*, vol. 35, pp. 561–569, 2008.

[60] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *MICAI 2004: Advances in Artificial Intelligence: Third Mexican International Conference on Artificial Intelligence, Mexico City, Mexico, April 26-30, 2004. Proceedings 3*, pp. 688–697, Springer, 2004.

[61] J. Blank and K. Deb, "Constrained bi-objective surrogate-assisted optimization of problems with heterogeneous evaluation times: Expensive objectives and inexpensive constraints," in *Evolutionary Multi-Criterion Optimization*, pp. 257–269, Springer International Publishing, 2021.

[62] A. Suresh and K. Deb, "Machine learning-based prediction of new pareto-optimal solutions from pseudo-weights," *IEEE Transactions on Evolutionary Computation*, vol. 28, no. 5, pp. 1351–1365, 2024.

[63] K. Deb, P. C. Roy, and R. Hussein, "Surrogate modeling approaches for multiobjective optimization: Methods, taxonomy, and results," *Mathematical and Computational Applications*, vol. 26, no. 1, 2021.

[64] C. He, S. Huang, R. Cheng, K. C. Tan, and Y. Jin, "Evolutionary multiobjective optimization driven by generative adversarial networks (GANs)," *IEEE Transactions on Cybernetics*, vol. 51, no. 6, pp. 3129–3142, 2020.

[65] Y. Jin and B. Sendhoff, "Fuzzy preference incorporation into evolutionary multi-objective optimization," in *Proceedings of the 4th Asia-Pacific Conference on Simulated Evolution and Learning*, vol. 1, pp. 26–30, 2002.

[66] J. Liu, *Radial Basis Function (RBF) neural network control for mechanical systems: design, analysis and Matlab simulation*. Springer Science & Business Media, 2013.

[67] T. Takagi, K. Takadama, and H. Sato, "Supervised multi-objective optimization algorithm using estimation," in *2022 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, IEEE, 2022.

[68] T. Takagi, K. Takadama, and H. Sato, "Pareto front upconvert by iterative estimation modeling and solution sampling," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 218–230, Springer, 2023.

[69] M. L. Stein, *Interpolation of spatial data: some theory for kriging*. Springer Science & Business Media, 1999.

[70] R. Cheng, Y. Jin, and K. Narukawa, "Adaptive reference vector generation for inverse model based evolutionary multiobjective optimization with degenerate and disconnected pareto fronts," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9018, pp. 127–140, 2015.

[71] A. Gaspar-Cunha and A. Vieira, "A multi-objective evolutionary algorithm using neural networks to approximate fitness evaluations.," *International Journal of Computers, Systems and Signals*, vol. 6, no. 1, pp. 18–36, 2005.

[72] A. Gupta, Y.-S. Ong, M. Shakeri, X. Chi, and A. Z. NengSheng, "The blessing of dimensionality in many-objective search: An inverse machine learning insight," in *2019 IEEE International Conference on Big Data (Big Data)*, pp. 3896–3902, IEEE, 2019.

[73] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," *Communications of the ACM*, vol. 63, no. 11, pp. 139–144, 2020.

[74] P. Bhardwaj, B. Dasgupta, and K. Deb, "Modelling the pareto-optimal set using b-spline basis functions for continuous multi-objective optimization problems," *Engineering Optimization*, vol. 46, no. 7, pp. 912–938, 2014.

[75] Y. Tian, X. Zhang, R. Cheng, C. He, and Y. Jin, "Guiding evolutionary multiobjective optimization with generic front modeling," *IEEE Transactions on Cybernetics*, vol. 50, no. 3, pp. 1106–1119, 2018.

[76] H. Ishibuchi, T. Matsumoto, N. Masuyama, and Y. Nojima, "Effects of dominance resistant solutions on the performance of evolutionary multi-objective and many-objective algorithms," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, pp. 507–515, 2020.

[77] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, vol. 8, pp. 173–195, 06 2000.

[78] T. T. Binh and U. Korn, "MOBES: A multiobjective evolution strategy for constrained optimization problems," in *The Third International Conference on Genetic Algorithms (Mendel 97)*, vol. 25, p. 27, 1997.

[79] A. Osyczka and S. Kundu, "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm," *Structural Optimization*, vol. 10, no. 2, pp. 94–99, 1995.

[80] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 2-4, pp. 311–338, 2000.

[81] K. Deb, A. Sinha, and S. Kukkonen, "Multi-objective test problems, linkages, and evolutionary methodologies," in *Proceedings of the 8th annual conference on Genetic and Evolutionary Computation*, pp. 1141–1148, 2006.

[82] K. Deb and R. B. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 9, no. 2, pp. 115–148, 1995.

[83] K. Deb and A. Srinivasan, "Innovization: Innovating design principles through optimization," in *Proceedings of the 8th annual conference on Genetic and Evolutionary Computation*, pp. 1629–1636, 2006.

[84] S. B. Gee, K. C. Tan, and C. Alippi, "Solving multiobjective optimization problems in unknown dynamic environments: An inverse modeling approach," *IEEE Transactions on Cybernetics*, vol. 47, no. 12, pp. 4223–4234, 2016.

[85] B. Li, Y. Lu, H. Qian, W. Hong, P. Yang, and A. Zhou, "Regularity model based offspring generation in surrogate-assisted evolutionary algorithms for expensive multi-objective optimization," *Swarm and Evolutionary Computation*, vol. 86, p. 101506, 2024.

[86] X. Lin, Z. Yang, X. Zhang, and Q. Zhang, "Pareto set learning for expensive multi-objective optimization," *Advances in Neural Information Processing Systems*, vol. 35, pp. 19231–19247, 2022.

[87] Z. Ma and Y. Wang, "Evolutionary constrained multiobjective optimization: Test suite construction and performance comparisons," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 6, pp. 972–986, 2019.