Part I

Optimisation Background

Chapter 2

Formal Definitions

"We should forget about small efficiencies, say about 97% of the time: premature optimization is the root of all evil." – Donald E. Knuth

In the modern world of today optimisation occurs in many aspects and areas of everyday life. For example, a manufacturer wants to increase his profit and therefore the cost of the manufacturing process has to be as low as possible. If this is approached as an unconstrained SOOP, it can be defined as follows:

Example 2.1: A manufacturer wants to minimise the cost of the manufacturing process.

However, many optimisation problems have more than one goal and some problems occur in a changing environment. Example 2.1 can be defined with more than one goal to increase a manufacturer's profit, namely minimising the cost of the manufacturing process and maximising the number of manufactured goods produced per day.

This chapter provides a theoretical overview of optimisation, presenting theory and definitions that are needed throughout the thesis. It does not give a complete overview of all aspects of MOO, dynamic single-objective optimisation (DSOO) and DMOO. However, this chapter highlights the most important information that is required to understand concepts discussed in later chapters. Section 2.1 discusses the main concepts of optimisation theory, highlighting the different types of optima and characteristics of optimisation problems. The theory of MOO is summarised in Section 2.2, where a MOO problem is defined and the goal of solving MOO problems is clarified. Section 2.3 discusses DSOO and highlights the various types of environments for DSOOPs. DMOO, and the different types of DMOO problems, are presented in Section 2.4.

2.1 Single Objective Optimisation

This section discusses the main concepts of SOO. Section 2.1.1 discusses SOO theory that is required to understand the main concepts of MOO theory and Section 2.1.2 discusses the type of solutions that can be obtained for SOOPs.

2.1.1 Optimisation Concepts

Each optimisation problem contains one or more objective functions and a set of decision variables and most optimisation problems contain a set of constraints. Optimisation problems can be classified according to a number of characteristics, including the number of decision variables, the type of decision variables, the degree of linearity of the objective functions, the type of constraints, the number of optimisation criteria or objectives and the number of optima [36, 55]. These concepts are discussed in more detail below.

The **objective function** represents the quantity to be optimised, i.e. the quantity to be minimised or maximised. The objective function is also referred to as the *cost function* or *optimisation criterion*. If the problem that has to be optimised is expressed using only one objective function, it is referred to as a SOOP. However, if a problem has more than one objective that have to be optimised simultaneously, it is called a MOOP.

Each objective function has a vector of **decision variables** that influence the value of the objective function. Therefore, a search algorithm iteratively modifies the value of these variables to find the optimum for the objective function. If $\mathbf{x}$ represents the set of variables, the value of the objective function for the specific values of the variables can be quantified by $f(\mathbf{x})$. Therefore, $f(\mathbf{x})$ also quantifies the quality of the candidate solution, $\mathbf{x}$.

A problem with only one decision variable to optimise (only one variable influences the objective function) is referred to as a *univariate* problem. A *multivariate* problem is a problem where more than one variable influence the objective function. When the type of decision variables is taken into account and a problem's decision variables have only continuous values, i.e. $x_k \in \mathbb{R}$, $\forall k = 1, \ldots, n_x$, the problem is referred to as a *continuous-valued* problem. The domain of a *discrete-valued* optimisation problem has a limited number of discrete values. Combinatorial problems are problems were solutions are permutations of integer-valued variables. When the decision variables can only have 0 or 1 as value, the problem is called a *binary-valued* problem.

When an objective function is linear in its variables, the problem is a *linear* problem. A *quadratic* problem has a quadratic objective function. However, when any other non-linear objective functions are used, the problem is referred to as a *non-linear* problem.

If an optimisation problem has constraints, the set of **constraints** restricts the values that can be assigned to the set of decision variables. Equality constraints restrict a variable to a specific value, for example $g(x_2) = 3$. Inequality constraints can take one of two forms, namely:

- Boundary constraints that restrict the domain of values that can be assigned to each variable and thereby define the *search space*. For example, $-1 \leq x_1 \leq 1$ restricts the value that variable x_1 can have to values between -1 and 1.
- Constraints of the form $c(\mathbf{x}) \leq 0$ or $c(\mathbf{x}) \geq 0$.

Values of $\mathbf{x}$ that satisfy the constraints form the *feasible search space* that is a subset of the search space. Problems that only use boundary constraints are generally referred to as *unconstrained* problems. However, when problems also have equality or inequality constraints, these problems are referred to as *constrained* optimisation problems.

When solving an optimisation problem with either equality or inequality constraints, the optimisation method's goal is to assign values from the specified domain to the decision variables in order to optimise the objective function and to satisfy the constraints. Therefore, the optimisation algorithm searches for a solution in the feasible search space, $\mathbf{x} \in F \subseteq S \subseteq \mathbb{R}^{n_x}$, that will obtain the smallest possible objective function value, $f(\mathbf{x})$, for a minimisation problem (or largest possible value for a maximisation problem). Throughout this thesis, unless stated differently, minimisation is assumed.

Mathematically, a SOOP is defined as:

$$\begin{aligned} \mathit{minimise}: \quad & f(\mathbf{x}) \\ \mathit{subject\ to}: \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, n_g \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, n_h \\ & \mathbf{x} \in [\mathbf{x}_{min}\,, \mathbf{x}_{max}]^{n_x} \end{aligned} \tag{2.1}$$

where n_x is the number of decision variables; $\mathbf{x} = (x_1, x_2, \ldots, x_{n_x}) \in S \subseteq \mathbb{R}^{n_x}$; n_g is the number of inequality constraints, $\mathbf{g}$; n_h is the number of equality constraints, $\mathbf{h}$; and $\mathbf{x} \in [\mathbf{x}_{min}\,, \mathbf{x}_{max}]^{n_x}$ refers to the boundary constraints (domain of $\mathbf{x}$), with $\mathbf{x}_{min}$ and $\mathbf{x}_{max}$ referring to the lower- and upper bounds of the feasible values for decision variables $\mathbf{x}$. The research in this thesis focuses on unconstrained optimisation problems.

The next section discusses the various types of solutions that can be found when solving a SOOP.

2.1.2 Types of Solutions

This section discusses the type of solutions with various degrees of quality that can be obtained when solving SOOPs.

Solutions found by an optimisation algorithm can be classified according to their quality, where the main types of solutions for a minimisation problem are the global minimum and local minimum. The various degrees of solution quality, in terms of the global minima and local minima, are defined below.

Definition 2.1. Global minima: The solution $\mathbf{x}_i^* \in F$, with $F \subseteq S$, is a global minimum of the objective function f, if

$$f(\mathbf{x}_i^*) \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in F,\ \mathbf{x}_i^* \neq \mathbf{x},\ \forall i = 1, \ldots, q \tag{2.2}$$

where q is the number of global minima of the SOOP.

Therefore, the best candidate solutions that lead to the smallest value of the objective function is called the global minima. The various types of minima are illustrated in Figure 2.1, with the point $\mathbf{x}_2$ as the global minimum of the function. It is important to note that an optimisation problem can have more than one global minimum. A problem with only one solution (or optimum) is a *uni-modal* problem, but if more than one optimum exists, the problem is referred to as a *multi-modal* problem.

Local minima can be either strong or weak, defined as follows:

Definition 2.2. Strong local minima: The solution $\mathbf{x}^*_{N_i} \in N \subseteq F$ is a strong local minimum of the objective function f, if

$$\mathbf{x}^*_{N_i} < f(\mathbf{x}), \quad \forall \mathbf{x} \in N,\ \mathbf{x}^*_{N_i} \neq \mathbf{x},\ \forall i = 1, \ldots, q \tag{2.3}$$

where $N \subseteq F$ is a subset of points in the feasible space that is in the neighbourhood of $\mathbf{x}^*_N$ and q is the number of strong local minima of the SOOP. The point $\mathbf{x}_1$ in Figure 2.1 is a strong local minimum.

Definition 2.3. Weak local minimum: The solution $\mathbf{x}^*_{N_i} \in N \subseteq F$ is a weak local minimum of the objective function f, if

$$f(\mathbf{x}^*_{N_i}) \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in N,\ \mathbf{x}^*_{N_i} \neq \mathbf{x},\ \forall i = 1, \ldots, q \tag{2.4}$$

where q is the number of weak local minima of the SOOP. Point $\mathbf{x}_3$ in Figure 2.1 is a weak local minima.

2.2 Multi-objective Optimisation

Many optimisation problems have more than one objective. The manufacturing example given earlier in Example 2.1 can be extended to a MOOP as follows:

Example 2.2: A manufacturer wants to maximise its profit. However, many factors have an influence on profit, for example the time required to manufacture a specific number of products, the time that a specific machine is idle and the cost of the manufacturing process. Therefore, the goals or objectives of the manufacturer are to minimise the time required to manufacture a specific number of products, to minimise the time that a specific machine is idle, and to minimise the cost of the manufacturing process.

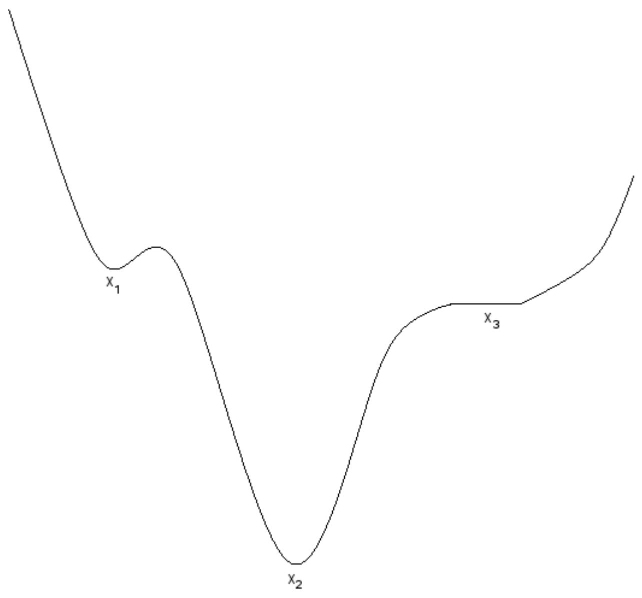


Figure 2.1: Optima of a minimisation function

However, using a specific machine can be more expensive to use than another, and the more expensive machine may require less time to manufacture the same number of products than a machine that is cheaper to operate. Therefore, in order to manufacture the maximum number of products in a certain time, using the more expensive machine will minimise the time required, but will increase the cost.

This example highlights an important problem with many MOOPs, namely that the objectives are in conflict with one another – minimising the time that the more expensive machine is idle increases the operational cost and vice versa. In this thesis, when referring to MOO, MOOPs with conflicting objectives are implied.

This section discusses the theory and definitions with regards to MOO [36, 55]. A MOOP is defined in Section 2.2.1 and the concept of optima is extended for MOO in Section 2.2.2. Section 2.2.3 discusses the goal when solving a MOOP and how this goal differs from situations when solving a SOOP.

2.2.1 Multi-objective Optimisation Problems

This section extends the mathematical definition of a SOOP (refer to Equation 2.1) to mathematically define a MOOP.

Let a single objective function be defined as $f_k : \mathbb{R}^{n_x} \to \mathbb{R}$. Then $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_{n_k}(\mathbf{x})) \in O_{space} \subseteq \mathbb{R}^{n_m}$ represents an *objective vector* containing n_k objective function evaluations, and O_{space} is the *objective space*.

Using the notation defined above, a MOOP can be mathematically defined as follows:

$$\begin{aligned} \textit{minimise}: \quad & \mathbf{f}(\mathbf{x}) \\ \textit{subject to}: \quad & g_i(\mathbf{x}) \leq 0, \quad i = 1, \ldots, n_g \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \ldots, n_h \\ & \mathbf{x} \in [\mathbf{x}_{min}, \mathbf{x}_{max}]^{n_x} \end{aligned} \tag{2.5}$$

2.2.2 Pareto-optimal Set and Pareto Optimal Front

For SOOPs, where only one objective is optimised, local and global optima are defined as presented in Section 2.1.2. However, when dealing with a MOOP, the various objectives are normally in conflict with one another, i.e. improvement in one objective leads to a worse solution for at least one other objective. For the manufacturing example (Example 2.2 in Section 2.2) the various objectives, namely to minimise the time required to manufacture a specific number of products, to minimise the time that a specific machine is idle, and ro minimise cost, are in conflict with one another. MOOPs do not have specific optima, but trade-off solutions. Therefore, for MOOPs, the definition of optimality has to be re-defined. This section discusses the new definition of optimality for MOO.

When solving a MOOP the goal is to find a set of trade-off solutions where for each of these solutions no objective can be improved without causing a worse solution for at least one of the other objectives. These solutions are referred to as *non-dominated solutions* and the set of such solutions is called the *non-dominated set* or *Pareto-optimal set (POS)*. The corresponding objective vectors in the objective space that lead to the non-dominated solutions are referred to as the *POF* or *Pareto-front*. These concepts and definitions are now discussed in more detail.

For MOOPs, when one decision vector dominates another, the dominating decision vector is considered as a better decision vector. Decision vector domination is defined as follows:

Definition 2.4. Decision Vector Domination: A decision vector $\mathbf{x}_1$ dominates another decision vector $\mathbf{x}_2$, denoted by $\mathbf{x}_1 \prec \mathbf{x}_2$, if and only if

- $\mathbf{x}_1$ is at least as good as $\mathbf{x}_2$ for all the objectives, i.e. $f_k(\mathbf{x}_1) \leq f_k(\mathbf{x}_2)$, $\forall k = 1, \ldots, n_k$; and
- $\mathbf{x}_1$ is strictly better than $\mathbf{x}_2$ for at least one objective, i.e. $\exists k = 1, \ldots, n_k \colon f_k(\mathbf{x}_1) < f_k(\mathbf{x}_2)$.

where n_k is the number of objective functions.

A slightly less strict comparison can be made between two decision vectors using the concept of weak domination, defined as:

Definition 2.5. Weak Decision Vector Domination: A decision vector $\mathbf{x}_1$ weakly dominates another decision vector $\mathbf{x}_2$, denoted by $\mathbf{x}_1 \preceq \mathbf{x}_2$, if and only if

$\mathbf{x}_1$ is at least as good as $\mathbf{x}_2$ for all the objectives, i.e. $f_k(\mathbf{x}_1) \leq f_k(\mathbf{x}_2)$, $\forall k = 1, \ldots, n_k$

The decision vectors that lead to the best trade-off solutions, are called Pareto-optimal, defined as follows:

Definition 2.6. Pareto-optimal: A decision vector $\mathbf{x}^*$ is Pareto-optimal if there does not exist a decision vector $\mathbf{x} \neq \mathbf{x}^* \in F$ that dominates $\mathbf{x}^*$, i.e. $\nexists k \colon f_k(\mathbf{x}) < f_k(\mathbf{x}^*)$. If $\mathbf{x}^*$ is Pareto-optimal, the objective vector, $\mathbf{f}(\mathbf{x}^*)$, is also Pareto-optimal.

Together, all the Pareto-optimal decision vectors form the Pareto-optimal set (POS), defined as:

Definition 2.7. Pareto-optimal Set: The POS, P^*, is formed by the set of all Pareto-optimal decision vectors, i.e.

$$P^* = \{\mathbf{x}^* \in F \,|\, \nexists \mathbf{x} \in F \colon \mathbf{x} \prec \mathbf{x}^*\} \tag{2.6}$$

The POS contains the best trade-off solutions for the MOOP. The corresponding objective vectors form the Pareto-optimal front (POF), which is defined as follows:

Definition 2.8. Pareto-optimal Front: For the objective vector $\mathbf{f}(\mathbf{x})$ and the POS P^*, the POF, $PF^* \subseteq O_{space}$ is defined as

$$PF^* = \{\mathbf{f} = (f_1(\mathbf{x}^*), f_2(\mathbf{x}^*), \ldots, f_{n_k}(\mathbf{x}^*))\}, \; \forall \mathbf{x}^* \in P^* \tag{2.7}$$

Therefore, the POF contains the set of objective vectors that corresponds to the POS, i.e. the set of decision vectors that are non-dominated. The POF can have various shapes, e.g. a convex POF or a concave POF, as can be seen in Section 3.1.

Some MOO algorithms make use of ϵ-domination and an ϵ-approximate POF, proposed by Laumanns *et al.* [101], which are extensions of Definitions 2.4 and 2.8 above. With ϵ-domination, a decision vector $\mathbf{x}$ dominates not only all decision vectors as defined in Definition 2.4, but also all decision vectors that are within a distance ϵ of $\mathbf{x}$. The ϵ value can be selected by the decision maker to control the size of the set of solutions [80]. Furthermore, ϵ-domination provides a way for algorithms to find solutions that converge to the POF and that has a good diversity [101]. ϵ-domination for decision vectors and objective vectors, and an ϵ-approximate POF are defined below in Definitions 2.9, **??** and 2.10 respectively.

Definition 2.9. Decision Vector ϵ-Domination: A decision vector $\mathbf{x}_1$ ϵ-dominates another decision vector $\mathbf{x}_2$, denoted by $\mathbf{x}_1 \prec_\epsilon \mathbf{x}_2$, if and only if

- $f_k(\mathbf{x}_1)/(1+\epsilon) \leq f_k(\mathbf{x}_2), \; \forall k = 1, \ldots, n_k, \;\; \epsilon > 0$; and
- $\exists k = 1, \ldots, n_k \colon f_k(\mathbf{x}_1)/(1+\epsilon) < f_k(\mathbf{x}_2), \; \epsilon > 0$.

Definition 2.10. ϵ-approximate Pareto-optimal Front: For the objective vector $\mathbf{f}(\mathbf{x})$ and an $\epsilon > 0$, the ϵ-approximate POF, $PF^*_\epsilon \subseteq O_{space}$, contains all objective vectors which are not ϵ-dominated by any other objective vector and is therefore defined as

$$\begin{aligned} PF^*_\epsilon = \{\mathbf{f} = (f_1(\mathbf{x}^*), f_2(\mathbf{x}^*), \ldots, f_{n_k}(\mathbf{x}^*)), \; \forall \mathbf{x}^* \in P^* \,|\, \nexists \mathbf{x} \in F : \\ f(\mathbf{x}) \prec_\epsilon f_k(\mathbf{x}^*), \; \forall k = 1, \ldots, n_k\} \end{aligned} \tag{2.8}$$

2.2.3 Solving a Multi-objective Optimisation Problem

When solving a MOOP, the goal is to approximate the true POF. If the problem requires a single solution, the best trade-off solution is selected for the specific problem from the set of solutions represented by the POF. Therefore, the goal is to find an approximation of the true POF such that:

- The distance between the found POF and the true POF is minimised.
- The set of non-dominated solutions is as diverse as possible and as evenly spread out along the found POF as possible.
- The set of non-dominated solutions contains as many solutions as possible.
- The solutions that have been found and that forms the found POF are stored for later reference.

Similar to a SOOP having global and local optima, a MOOP can have a global POF or local POFs. Definitions 2.1 to 2.3 for SOO is extended for MOO as follows:

Definition 2.11. Global POF: PF_g^* is the global POF of a DMOOP, $\mathbf{f}$, if

$$\mathbf{f}(\mathbf{x}^*) \prec \mathbf{f}(\mathbf{x}), \quad \forall \mathbf{x} \in F | \mathbf{x} \notin P^*, \, \forall \mathbf{x}^* \in P^*, \, \mathbf{x}^* \neq \mathbf{x} \tag{2.9}$$

where P^* is the POS of $\mathbf{f}$.

Therefore, the best candidate solutions that lead to the best trade-off solutions, form the POS and the corresponding values in the objective space result in the global POF or the true POF. A MOOP can have many local POFs, with a local POF defined as follows:

Definition 2.12. Local POF: $PF_{l_i}^*$ is a local POF of a DMOOP, $\mathbf{f}$, if

$$\mathbf{f}(\mathbf{x}_{N_i}^*) \prec \mathbf{f}(\mathbf{x}), \quad \forall \mathbf{x} \in N | \mathbf{x} \notin P^*, \, \mathbf{x}_{N_i}^* \neq \mathbf{x}, \, \mathbf{x}_{N_i}^* \in N, \, \forall i = 1, \ldots, q \tag{2.10}$$

where $N \subseteq F$ is a subset of points in the feasible space that is in the neighbourhood of $\mathbf{x}_{N_i}^*$ and q is the number of local POFs.

When a MOOP has local POFs, an algorithm can become stuck in one of the local POFs and this will prevent the algorithm from converging to the global POF.

2.3 Dynamic Single-objective Optimisation

In many real-world situations the objective function that has to be optimised is not static. A change in the objective function and/or the constraints can lead to a change in the environment. The change in the objective function and/or constraints causes a change in the search landscape, S, and/or the feasible space F, and causes changes to the optima of the problem, i.e. optima can change in position or value, or optima can disappear while new optima can appear. The manufacturing example, Example 2.1, can be extended to illustrate a DSOOP as follows:

Example 2.3: A manufacturer wants to minimise the cost of the manufacturing process. If the cost is calculated by taking the cost of using the machines into account, then if one machine breaks down, the environment changes. There will be idle time while the machine is being replaced and the new machine may not be exactly the same as the previous one – the new machine may be more expensive to use and/or may need longer time to complete the manufacturing process. Therefore, the previous solution cannot be used anymore, and a new solution for the changed situation has to be found.

This section discusses the theory and definitions [55] with regards to DSOO. A DSOOP is mathematically defined in Section 2.3.1 and Section 2.3.2 discusses the various classifications of dynamic environments.

2.3.1 Dynamic Single-objective Optimisation Problem

A DSOOP can formally be defined as follows:

$$\begin{aligned} &Minimise: && f(\mathbf{x},t), && \mathbf{x}=(x_1,\ldots,x_{n_x}) \\ &Subject\ to: && g_i(\mathbf{x},t)\leq 0, && i=1,\ldots,n_g \\ & && h_j(\mathbf{x},t)=0, && j=1,\ldots,n_h \\ & && \mathbf{x}\in[\mathbf{x}_{min};\ \mathbf{x}_{max}]^{n_x} && \end{aligned} \tag{2.11}$$

In order to solve the DSOOP, the goal is to find

$$\mathbf{x}^*(t) = min_{\mathbf{x}\in F(t)}\, f(\mathbf{x},t) \tag{2.12}$$

where $\mathbf{x}^*(t)$ is the optimum at time step t and $F(t)$ is the feasible space at time t.

Since the optima change with time, the goal of an optimisation algorithm for dynamic environments is to locate an optimum and track its trajectory as closely as possible, and to find new optima that may appear.

2.3.2 Dynamic Environment Types

Dynamic environments or DSOOPs can change in various ways over time. When a change occurs in the environment, *temporal severity* refers to the frequency of change that the environment experiences and *spatial severity* refers to the extend of change in the position of the optima.

Based on real-world problems, De Jong [91] identified four types of changes that can occur in a dynamic environment:

- **Drifting landscapes**, where the optima moves gradually over time, for example aging equipment in a large production plant.
- **Significant changes in the optima location**, where peaks of high fitness shrink and new regions of high fitness emerge that was previously uninteresting regions, for example competitive market places where opportunities for high profit fluctuate as the levels of competition change over time.
- **Cyclic patterns** in the landscape, where a relatively small number of states re-occur over time, for example seasonal climate changes.
- **Abrubt and discontinuous changes** in the landscape, for example a power station failure on a distribution grid.

Eberhart and Shi [53] defined the following three generic dynamic environment types for SOO:

- **Type I environments** where the location of the optimum in the problem space, $\mathbf{x}^*(t)$, changes, but $f(\mathbf{x}^*(t))$ remains unchanged. The spatial severity, ζ, measures the change in $\mathbf{x}^*(t)$.
- **Type II environments** where $\mathbf{x}^*(t)$ remains unchanged, but the objective function value at $\mathbf{x}^*(t)$, $f(\mathbf{x}^*(t))$, changes.
- **Type III environments** where both $\mathbf{x}^*(t)$ and $f(\mathbf{x}^*(t))$ changes. The change in $\mathbf{x}^*(t)$ is indicated by ζ.

These three types are summarised in Table 2.1.

Table 2.1: Dynamic environment types as defined by Eberhart and Shi [53]

	Optimum Location	
Optimum Value	No Change	Change
No Change	Static	Type I
Change	Type II	Type III

Branke [12] categorised dynamic environments according to the following characteristics:

- **Frequency of change** or temporal severity that determines how often the environment changes.
- **Severity of change** or spatial severity that are normally measured as the distance between the current and the previous optimum.
- **Predictability of change** that indicates whether the changes occur randomly or with a pattern that can be learned or predicted by an algorithm.
- **Cycle length or cycle accuracy** that indicates how long it takes before the environment returns to a previous state and how accurate or similar the returned state is with regards to the previous state.

More recently, Duhain [50] classified dynamic environments as follows:

- **Static environments**, where the environment does not change over time or the changes to the environment have such a small influence on the problem that they do not affect the performance of the algorithm for the duration of the simulation.
- **Progressively changing environments**, where the temporal severity is high, but the spatial severity (change in $\mathbf{x}^*(t)$) is low. Therefore, the environment changes in a progressive manner. Algorithms that solve problems with a progressively changing environment can use knowledge that was obtained earlier (the previous optima) to find the new optima that will be in close proximity of the previous optima.
- **Abruptly changing environments**, where the temporal severity is low, but the

spatial severity is high. Therefore, previous knowledge will not be as useful as in the case of a progressively changing environment.

- **Chaotic environments** where both the temporal and spatial severity are high.

These four types are summarised in Table 2.2. Duhain's classification is similar to De Jong, but more generic and using the concepts of temporal severity and spatial severity.

If the temporal severity is high, the environment changes frequently and therefore an algorithm would have to converge to the optima at a specific time step quickly and adapt quickly after a change to find the new optima. A high spatial severity occurs when $\mathbf{x}^*(t+1)$ differs severely from $\mathbf{x}^*(t)$ and therefore an algorithm has to find the new optima that is far from the previous location in the search space. It is important to note that not all problems' environment will remain one type for the whole duration of the simulation, but can change over time from one type of environment to another.

Table 2.2: Dynamic environment types as defined by Duhain [50]

	Spatial Severity	
Temporal Severity	Low	High
Low	Static	Abrupt
High	Progressive	Chaotic

2.4 Dynamic Multi-objective Optimisation

In most situations the optimisation problem is not static, and has more than one objective. Example 2.2 (refer to Section 2.2) can be extended to illustrate a DMOOP as follows:

Example 2.4: A manufacturer wants to maximise its profit. Therefore, the goals or objectives of the manufacturer are to minimise the time required to manufacture a specific number of products, to minimise the time that a specific machine is idle, and to minimise the cost of the manufacturing process. When a machine breaks down, the environment changes. This change in the environment may also influence more than one objective function. The breakdown of a machine can occur quite frequently and other changes can also occur. For example, the operational cost of a specific machine may change when it breaks down and is replaced by another machine that is not exactly the same as the replaced machine, the time required to complete the manufacturing process may take longer for a machine as it gets older, etc. Since this manufacturing problem is not static in nature, but dynamic, the previous solutions or POF will not be valid anymore and a new POF has to be found.

This section discusses DMOO in more detail. Section 2.4.1 provides a mathematical definition of a DMOOP and the various types of dynamic DMOOPs are discussed in Section 2.4.2.

2.4.1 Dynamic Multi-objective Optimisation Problem

This section provides a mathematical definition of a DMOOP.

A DMOOP can be defined as:

$$\begin{aligned} Minimise: &\quad \mathbf{f}(\mathbf{x},t), &\quad \mathbf{x} = (x_1,\dots,x_{n_x}) \\ Subject\ to: &\quad g_i(\mathbf{x},t) \le 0, &\quad i = 1,\dots,n_g \\ &\quad h_j(\mathbf{x},t) = 0, &\quad j = 1,\dots,n_h \\ &\quad \mathbf{x} \in [\mathbf{x}_{min};\ \mathbf{x}_{max}]^{n_x} \end{aligned} \tag{2.13}$$

Unlike DSOOPs with only one objective function, DMOOPs have many objective functions. Therefore, in order to solve the DMOOP the goal is to track the POF over time, i.e.

$$PF^*(t) = \{\mathbf{f}(t) = (f_1(\mathbf{x}^*,t), f_2(\mathbf{x}^*,t),\dots,f_{n_k}(\mathbf{x}^*,t)\},\ \forall \mathbf{x}^* \in P^*(t) \tag{2.14}$$

The next section discusses the various types of DMOOPs, as well as the various ways in which the POF can be affected when a change occurs in the environment.

2.4.2 Dynamic Environment Types

This section discusses the categorisation of DMOOPs, as well as the possible influences of a change in the environment on the POF.

Similar to the classification of dynamic environment types for DSOOPs (refer to Section 2.3.2), Farina *et al.* [58] classified dynamic environments for DMOOPs into four categories, namely:

- **Type I environment** where the POS (optimal set of decision variables) changes, but the POF (corresponding objective function values) remains unchanged.
- **Type II environment** where both the POS and the POF change.
- **Type III environment** where the POS remains unchanged, but the POF changes.
- **Type IV environment** where both the POS and the POF remain unchanged, even though an objective function or a constraint may have changed.

These four types are summarised in Table 2.3.

Table 2.3: Dynamic Environment Types for DMOO problems

	POS	
POF	No Change	Change
No Change	Type IV	Type I
Change	Type III	Type II

When a change occurs in the environment, the POF can change as follows over time:

1. Existing solutions in the POF becomes dominated and therefore are not part of the POF any more.
2. The shape of the POF remains the same, but its location in the objective space change over time. In these cases the POF shifts over time. This kind of change of the POF occurs with type I DMOOPs and are the easiest kind of DMOOPs to solve.
3. The shape of the POF changes over time. For example:
   - The POF changes from convex to concave or vice versa.
   - The POF changes from a continuous front to a disconnected front, i.e. various disconnected continuous-valued areas.

   This kind of change of the POF occurs with either type II or type III DMOOPs. When the shape of the POF changes over time, an algorithm has to track the changing POF and obtain a diverse set of solutions for the new shape of the POF. Therefore, if the shape of the POF changes over time, an algorithm may struggle to find a diverse set of solutions after a change has occurred.

4. The density of the solutions in the POF changes over time. For example:
   - The solutions in the POF becomes more/less dense.
   - The number of solutions in the POF becomes more/less.

   This kind of change in the POF can occur with all types of DMOOPs. When the number of solutions or the densitiy of the solutions in the POF change overtime, algorithms may struggle to find a diverse set of solutions.

2.5 Summary

This chapter discussed aspects of optimisation relevant to this thesis. Section 2.1.1 discussed optimisation problems and their characteristics with regards to the problem's objective functions, decision variables and constraints. Different types of solutions exist for an optimisation problem of which the main types are global and local minima, as defined in Section 2.1.2. Section 2.2.1 defined a MOOP and in order to re-define the optima for a MOOP, the concepts of a POS and POF were discussed in Section 2.2.2. Since most MOOPs do not have a single solution because of conflicting objectives, the goal when solving MOOPs were summarised in Section 2.2.3. Furthermore, the concepts of local and global optima for SOO have been extended to define local and global POFs for MOO in Section 2.2.3.

In real life, optimisation problems are not static in nature and change over time. Therefore, both DSOO and DMOO were introduced in this chapter. DSOO was discussed in Section 2.3 and a DSOOP was defined in Section 2.3.1. The environment of a DSOOP can change in various ways, as discussed in Section 2.3.2. However, many dynamic optimisation problems do not have only one objective and therefore DMOO was introduced in Section 2.4 and a DMOOP was defined in Section 2.4.1. Similar to DSOOPs, the environment of a DMOOP and the POF can change in various ways over time, as discussed in Section 2.4.2.

There exist many different approaches that are used to solve optimisation problems:

Population-based algorithms within the field of computational intelligence (CI), such as evolutionary algorithms (EAs), PSO algorithms, and ant algorithms, are widely used to solve optimisation problems. Various population-based approaches that are used to solve MOO and DMOO problems are discussed in Chapters 6, 7 and 8.

The next chapter discusses bechmark functions that are used to evaluate whether an algorithm can solve DMOOPs.

Chapter 3

Analysis of Dynamic Multi-objective Optimisation Benchmark Functions

"Without a standard there is no logical basis for making a decision or taking action." – Joseph M. Juran

Dynamic multi-objective optimisation problems are created by adjusting MOOPs in one or more of the following ways: changing the objective functions over time or changing the constraints over time. This thesis focusses on unconstrained DMOOPs with static boundary constraints and objective functions that change over time.

In order to determine whether an algorithm can solve DMOOPs efficiently, DMOOPs should be used that test the ability of the algorithm to overcome certain difficulties. These DMOOPs are called benchmark functions. One of the main problems in the field of DMOO is a lack of standard benchmark functions. This chapter seeks to address this problem by evaluating the current state-of-the-art benchmark functions presented in the DMOO literature to establish whether they efficiently evaluate the abilities of DMOO algorithms.

MOO benchmark functions adapted to develop DMOOPs and characteristics that an ideal set of MOO benchmark functions should have are discussed in Section 3.1. Current benchmark functions used in the DMOO literature are discussed in Section 3.2. Furthermore, approaches to develop DMOOPs with either an isolated or deceptive POF are proposed. New DMOOPs with complicated POSs, i.e. POSs that are defined by non-linear functions and where each decision variable has a different POS are introduced. Characteristics that an ideal DMOO benchmark function suite should have, are also presented and benchmark functions are suggested for each identified characteristic. Finally, a summary of this chapter is provided in Section 3.3.

3.1 Multi-objective Optimisation Benchmark Functions

Benchmark functions test how well an algorithm can overcome various types of difficulties when trying to find the true POF. When an algorithm solves a MOOP its goal is to find solutions that are as close as possible to the true POF and that have an uniform spread. Therefore, benchmark problems should test whether an algorithm can achieve this goal when faced with either multi-modality, deception (such as local POFs and isolated optima that may prevent the algorithm from converging towards the true POF; or a POF that is non-convex, discontinuous or non-uniform that may prevent the algorithm from finding an uniform spread of solutions [36, 49].

Section 3.1.1 discusses characteristics of ideal benchmark functions suites. Furthermore, two MOO benchmark function suites, namely the ZDT [38] and DTLZ functions [49], that were adapted to develop DMOOPs are discussed in Sections 3.1.2 and 3.1.3 respectively.

3.1.1 Ideal Benchmark Function Characteristics

This section discusses characteristics that an ideal benchmark function suite should exhibit.

Deb *et al.* [49] constructed the ZDT [38, 169] and DTLZ [49] MOOP suites in such a way that the benchmark functions are:

- easy to construct,
- scalable in terms of the number of decision variables as well as the number of objective functions,

- producing a POF that is easy to understand with the POF's shape and location known, and
- hindering an algorithm to converge to the true POF and to produce a good distribution of solutions.

According to Deb *et al.* [38], an algorithm can be hindered in converging to the true POF when a benchmark function is multi-modal, deceptive, has an isolated optimum, or contains noise. Deceptive functions have at least two optima in the search space, but the search space favours the deceptive optimum, which is a local POF and not the true global POF. If a function is multi-modal, it has many POFs and a DMOO algorithm can become stuck in a local POF. If an open subset of decision variable values maps to a single objective function value, the objective function is referred to as an objective function with *flat regions*, i.e. regions where small perturbations of the decision variable values do not change the objective function value. The lack of gradient information for the flat regions may cause an algorithm to struggle to converge to the optima. For DMOOPs, if the majority of the fitness landscape is fairly flat and no useful information is provided with regards to the location of the POF, the POF is referred to as being *isolated*. Therefore, if the DMOOP has an isolated POF, a DMOO algorithm may struggle to converge towards it. Even if the POF is not completely isolated from the rest of the search space, i.e. the majority of the fitness landscape is not fairly flat, an algorithm may struggle to converge towards the POF if the density of solutions close to the POF is significantly less than in the rest of the search space.

The following properties of the true POF may cause difficulty for an algorithm to find a diverse set of solutions: convexity or non-convexity in the POF, a discontinuous POF, or a non-uniform spacing of solutions in the POS or POF [38, 40]. When a POF is convex, it may be difficult to solve the DMOOP by algorithms that assign a solution's fitness based on the number of solutions that the solution dominates (Pareto ranking) [38]. This fitness assignment favours intermediate or middling solutions that perform reasonably well with regards to all objective functions more than solutions that perform very good with regards to one objective and not so good with regards to the other objectives. Therefore, this fitness assignment may cause bias towards certain portions of the POF that contain intermediate solutions. If the POF is discontinous and has a set of disconnected continuous sub-regions, an algorithm may struggle to find all regions of the POF. Even though an algorithm may find solutions within each region, when the solutions compete amongst each other (for storage in the archive or for a rank), solutions from certain sub-regions may be outranked and therefore may be removed from the non-dominated solution set. If the POS or POF is not uniformly spaced, an algorithm may struggle to find a diverse set of non-dominated solutions [40].

3.1.2 ZDT Functions

Deb introduced a tunable two-objective optimisation problem, defined as [38]:

$$\begin{aligned}
\text{Minimise:} \quad & f(\mathbf{x}) = (f_1(\mathbf{x_I}), f_2(\mathbf{x})) \\
\text{Subject to:} \quad & f_1(\mathbf{x_I}) = f_1(x_1, x_2, \ldots, x_m) \\
& f_2(\mathbf{x_{II}}) = g(\mathbf{x_{II}}) \cdot \ h(f_1(\mathbf{x_I}), g(\mathbf{x_{II}})) \\
& \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n)
\end{aligned} \tag{3.1}$$

where $f_1, g > 0$. MOOPs with specific features can be created by changing the f_1, g and h functions:

- the selected h function influences the convexity or discontiuity of the POF.
- a difficult g function affects the level of difficulty that an algorithm experiences when converging to the true POF.
- the selected f_1 function affects the diversity or spread of solutions in the POF.

Based on this two-objective optimisation problem and the guidelines produced by Deb *et al* [38] as discussed in Section 3.1.1, Zitzler, Deb and Thiele introduced six benchmark functions referred to as the ZDT functions (first letter of the surnames of the three authors) [169]. Each of the functions are structured according to Equation (3.1) and addresses one of the six difficulties discussed in Section 3.1.1. The mathematical equations (Equations (3.2) to (3.7)) of these functions are presented below:

$$\text{ZDT1} = \begin{cases} Minimise: f(\mathbf{x}) = (f_1(x_1), g(\mathbf{x_{II}}) \cdot h(f_1(x_1), g(\mathbf{x_{II}}))) \\ f_1(\mathbf{x_I}) = x_1 \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{i=2}^{m} \frac{x_i}{m-1} \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n), \;\; x_i \in [0,1] \end{cases} \tag{3.2}$$

where $m = 30$. ZDT1 has a convex POF that is formed with $g(\mathbf{x_{II}}) = 1$. Therefore the POF of ZDT1 is $1 - \sqrt{f_1}$ and the POS is $x_i = 0, \; \forall i \in \mathbf{x_{II}}$.

$$\text{ZDT2} = \begin{cases} Minimise: f(\mathbf{x}) = (f_1(x_1), g(\mathbf{x_{II}}) \cdot h(f_1(x_1), g(\mathbf{x_{II}}))) \\ f_1(x_1) = x_1 \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{i=2}^{m} \frac{x_i}{m-1} \\ h(f_1(x_1), g(\mathbf{x_{II}})) = 1 - \left(\frac{f_1}{g}\right)^2 \\ \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n), \;\; x_i \in [0,1] \end{cases} \tag{3.3}$$

where $m = 30$. The POF is non-convex with POF $= 1 - f_1^2$. The POS of ZDT2 is $x_i = 0, \; \forall i \in \mathbf{x_{II}}$.

$$\text{ZDT3} = \begin{cases} Minimise: f(\mathbf{x}) = (f_1(x_1), g(\mathbf{x_{II}}) \cdot h(f_1(x_1), g(\mathbf{x_{II}}))) \\ f_1(x_1) = x_1 \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{i=2}^{m} \frac{x_i}{m-1} \\ h(f_1(x_1), g(\mathbf{x_{II}})) = 1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi f_1) \\ where: \\ x_1 \in [0,1], \;\; \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n) \in [-5,5] \end{cases} \tag{3.4}$$

where $m = 10$. ZDT3 has a discrete POF that consists of several discontinuous convex parts. The sine function in h causes discontinuity in the POF, but not in the decision space. The POF is $1 - \sqrt{f_1} - f_1\sin(10\pi f_1)$. The POS of ZDT3 is $x_i = 0, \; \forall i \in \mathbf{x_{II}}$.

$$\text{ZDT4} = \begin{cases} Minimise: f(\mathbf{x}) = (f_1(x_1), g(\mathbf{x_{II}}) \cdot h(f_1(x_1), g(\mathbf{x_{II}}))) \\ f_1(x_1) = x_1 \\ g(\mathbf{x_{II}}) = 1 + 10(m-1) + \sum_{i=2}^{m}(x_i^2 - 10\cos(4\pi x_i)) \\ h(f_1(x_1), g(\mathbf{x_{II}})) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ x_1 \in [0,1], \ \ \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n) \in [-5,5] \end{cases} \tag{3.5}$$

where $m = 10$. The POF of ZDT4 has 21^9 local POFs and therefore tests the algorithm's ability to deal with multi-modality. The global POF is formed with $g(\mathbf{x_{II}}) = 1$ and is $1 - \sqrt{f_1}$. The global POS is $x_i = 0$, $\forall i \in \mathbf{x_{II}}$. The best local POF can be found with $g(\mathbf{x_{II}}) = 1.25$.

$$\text{ZDT5} = \begin{cases} Minimise: f(\mathbf{x}) = (f_1(x_1), g(\mathbf{x_{II}}) \cdot h(f_1(x_1), g(\mathbf{x_{II}}))) \\ f_1(x_1) = u(x_1) \\ g(\mathbf{x_{II}}) = 1 + 9\sum_{i=2}^{m} v(u(x_i)) \\ h(f_1(x_1), g(\mathbf{x_{II}})) = \frac{1}{f_1} \\ where: \\ x_1 \in \{0,1\}^{30}, \ \ \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n) \in \{0,1\}^5 \\ v(u(x_i)) = \begin{cases} 2 + u(x_i), & \text{if } u(x_i) < 5 \\ 1, & \text{if } u(x_i) = 5 \end{cases} \end{cases} \tag{3.6}$$

where $m = 11$. ZDT5 is a deceptive problem where x_i is represented by a binary string. The global POF is formed with $g(\mathbf{x_{II}}) = 10$. The best deceptive POF can be found where $g(\mathbf{x_{II}}) = 11$. The global and local POFs are convex.

$$\text{ZDT6} = \begin{cases} Minimise: f(\mathbf{x}) = (f_1(x_1), g(\mathbf{x_{II}}) \cdot h(f_1(x_1), g(\mathbf{x_{II}}))) \\ f_1(x_1) = 1 - exp(-4x_1)\sin^6(6\pi x_1) \\ g(\mathbf{x_{II}}) = 1 + 9\left(\frac{\sum_{i=2}^{m} x_i}{m-1}\right)^{0.25} \\ h(f_1(x_1), g(\mathbf{x_{II}})) = 1 - \left(\frac{f_1}{g}\right)^2 \\ where: \\ \mathbf{x_{II}} = (x_{m+1}, \ldots, x_n), \ \ x_i \in [0,1] \end{cases} \tag{3.7}$$

where $m = 10$. ZDT6 causes two difficulties for algorithms because of the non-uniformity of the search space, namely: (i) the solutions are non-uniformly distributed along the global POF, and (ii) the solutions are the least dense close to the POF and most dense away from the POF. ZDT6 has a non-convex POF $1 - f_1^2$. The POS of ZDT6 is $x_i = 0, \forall i \in \mathbf{x_{II}}$.

The ZDT functions are all two-objective optimisation problems. Therefore, Deb *et al.* [49] introduced test problems that can be scaled in terms of the number of objective functions.

3.1.3 DTLZ Functions

This section discusses two approaches, as well as a benchmark function generator, that were used to develop the Deb, Thiele, Laumanns and Zitzler (DTLZ) benchmark functions.

Spherical Coordinates Approach

Deb *et al.* [49] defined a test problem that has a POF in the first quadrant of a sphere with radius one and where all objective functions have non-negative values (add figure to refer to). Mathematically, using spherical coordinates (θ, γ and $r = 1$), the POF is defined as

$$\text{POF} = \begin{cases} f_1(\theta,\gamma) = \cos\theta\cos\left(\gamma + \frac{\pi}{4}\right) \\ f_2(\theta,\gamma) = \cos\theta\sin\left(\gamma + \frac{\pi}{4}\right) \\ f_3(\theta,\gamma) = \sin(\theta) \\ \text{where } \ 0 \le \theta \le \frac{\pi}{2},\ \frac{-\pi}{4} \le \gamma \le \frac{\pi}{4} \end{cases} \tag{3.8}$$

Any two points of the surface defined by Equation (3.8) are non-dominated if all three objective functions are minimised. By defining the rest of the search space above this surface, the POF is defined as the unit sphere. This can be done by constructing the rest of the search space parallel to the surface defined in Equation (3.8) as follows:

$$\text{POF} = \begin{cases} Minimise: \\ \quad f_1(\theta,\gamma,r) = (1+g(r))\cos\theta\cos\left(\gamma+\frac{\pi}{4}\right) \\ \quad f_2(\theta,\gamma) = (1+g(r))\cos\theta\sin\left(\gamma+\frac{\pi}{4}\right) \\ \quad f_3(\theta,\gamma) = (1+g(r))\sin(\theta) \\ where: \\ \quad 0 \leq \theta \leq \frac{\pi}{2},\ \frac{-\pi}{4} \leq \gamma \leq \frac{\pi}{4} \\ \quad g(r) \geq 0 \end{cases} \tag{3.9}$$

where the POS is $0 \leq \theta^* \leq \frac{\pi}{2}$, $\frac{-\pi}{4} \leq \gamma^* \leq \frac{\pi}{4}$, $g(r)^* = 0$. Although this three-objective problem has three independent variables (θ, γ and r), the variables can be meta-variables and can be considered as a function of n decision variables, i.e. $\theta = \theta(x_1, \ldots, x_n)$, $\gamma = \gamma(x_1, \ldots, x_n)$, $r = r(x_1, \ldots, x_n)$. These functions must adhere to the lower and upper bounds of the three variables and can be used to introduce difficulties to the optimisation problem.

Constraint Surface Approach

Another approach used by Deb *et al.* to develop benchmark functions are based on a constraint surface [49]. Firstly, a search space is defined as follows:

$$\begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) \\ \quad \ldots \\ \quad f_M(\mathbf{x}) \\ where: \\ \quad f_i^L \leq f_i(\mathbf{x}) \leq f_i^U,\ \forall\, i = 1, 2, \ldots, M \end{cases} \tag{3.10}$$

where f_i^L and f_i^U refers to the lower bound and upper bound of the objective function f_i respectively. The POS has only one solution, namely a solution that consists of the lower bound value of each objective, namely $(f_1^L, f_2^L, \ldots, f_M^L)^T$.

A set of constraints, that can be linear or non-linear, can be added to the problem in Equation (3.10), where each constraint eliminates a portion of the original rectangular region. Therefore, the optimisation problem of Equation (3.10) becomes:

$$\begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) \\ \quad \dots \\ \quad f_M(\mathbf{x}) \\ where: \\ \quad f_i^L \leq f_i(\mathbf{x}) \leq f_i^U \;\; \forall\, i = 1, 2, \dots, M \\ \quad g_j(f_1, f_2, \dots, f_M) \geq 0, \;\; \forall\, j = 1, 2, \dots, J \end{cases} \tag{3.11}$$

In order to solve this MOOP, the goal of an algorithm becomes to find the non-dominated part of the feasible space's boundary. The density of solutions in the search space can be modified by using non-linear functions for f_i.

Benchmark Function Generator

Based on the constraint surface approach, Deb [40, p.361] suggested a generic MOOP generator where the number of objectives can be scaled. Mathematically, the generator is defined as

$$\begin{cases} Minimise: \\ \quad f_1(x_1) \\ \quad \vdots \\ \quad f_{M-1}(\mathbf{x_{M-1}}) \\ \quad f_M(\mathbf{x}) = g(\mathbf{x_M}) \cdot h(f_1, \dots, f_{M-1}, g) \\ where: \\ \quad x_i \in \mathbb{R}^{|x_i|}, \; \forall i = 1, 2, \dots, M \end{cases} \tag{3.12}$$

where POF $= f_M = g \,\text{x}\, h(f_1, f_2, \dots, f_{M-1})$.

Using the concepts of Equations 3.9 and 3.12, Deb *et al.* [49] presented the DTLZ functions. The mathematical equations (Equations 3.13 to 3.19) of these functions are presented below:

$$\text{DTLZ1} = \begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) = \frac{1}{2}x_1x_2\dots x_{M-1}(1+g(\mathbf{x_M})) \\ \quad f_2(\mathbf{x}) = \frac{1}{2}x_1x_2\dots(1-x_{M-1})(1+g(\mathbf{x_M})) \\ \quad \vdots \\ \quad f_{M-1}(\mathbf{x}) = \frac{1}{2}x_1(1-x_2)(1+g(\mathbf{x_M})) \\ \quad f_M(\mathbf{x}) = \frac{1}{2}(1-x_1)(1+g(\mathbf{x_M})) \\ where: \\ \quad g(\mathbf{x_M}) = 100\left(|\mathbf{x_M}| + \sum_{x_i\in\mathbf{x_M}}(x_i-0.5)^2 - \cos(20\pi(x_i-0.5))\right) \\ \quad 0 \le x_i \le 1,\ \forall i = 1,2,\dots,n \\ \quad |\mathbf{x_M}| = k\ \ n = M+k-1 \end{cases} \tag{3.13}$$

where $k = 5$. The POF of DTLZ1 is a linear hyperplane with a POS of $x_i^* = 0.5,\ \ \forall x_i \in \mathbf{x_M}$. The POF of DTLZ1 is $\sum_{m=1}^{M} f_m^* = 0.5$. DTLZ1 introduces the difficulty of deception, since the search space has $(11^k - 1)$ local POFs.

$$\text{DTLZ2} = \begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos\left(\frac{x_1\pi}{2}\right)\dots cos\left(\frac{x_{M-1}\pi}{2}\right) \\ \quad f_2(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos\left(\frac{x_1\pi}{2}\right)\dots\sin\left(\frac{x_{M-1}\pi}{2}\right) \\ \quad \vdots \\ \quad f_M(\mathbf{x}) = (1+g(\mathbf{x_M}))\sin\left(\frac{x_{M-1}\pi}{2}\right) \\ where: \\ \quad g(\mathbf{x_M}) = \sum_{x_i\in\mathbf{x_M}}(x_i-0.5)^2 \\ \quad 0 \le x_i \le 1,\ \forall i = 1,2,\dots,n \\ \quad |\mathbf{x_M}| = k;\ \ n = M+k-1 \end{cases} \tag{3.14}$$

where $k = 10$. The POF of DTLZ2 is a sphere of radius one, namely $\sum_{m=1}^{M}(f_m^*)^2 = 1$. The POS is $x_i^* = 0.5,\ \ \forall x_i \in \mathbf{x_M}$.

$$\text{DTLZ3} = \begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos\left(\frac{x_1\pi}{2}\right)\ldots cos\left(\frac{x_{M-1}\pi}{2}\right) \\ \quad f_2(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos\left(\frac{x_1\pi}{2}\right)\ldots \sin\left(\frac{x_{M-1}\pi}{2}\right) \\ \quad \vdots \\ \quad f_M(\mathbf{x}) = (1+g(\mathbf{x_M}))\sin\left(\frac{x_{M-1}\pi}{2}\right) \\ where: \\ \quad g(\mathbf{x_M}) = 100\left(|\mathbf{x_M}| + \sum_{x_i\in\mathbf{x_M}} (x_i-0.5)^2 - \cos(20\pi(x_i-0.5))\right) \\ \quad 0 \le x_i \le 1,\ \forall i = 1,2,\ldots,n \\ \quad |\mathbf{x_M}| = k;\ \ n = M+k-1 \end{cases} \tag{3.15}$$

where $k = 10$. Similar to DTLZ2, the POF of DTLZ3 is a sphere of radius one, namely $\sum_{m=1}^{M}(f_m^*)^2 = 1$ with a POS of $x_i^* = 0.5,\ \forall x_i \in \mathbf{x_M}$. However, this MOOP has many local POFs and will test an algorithm's ability to converge to the global POF in the presence of many local POFs.

$$\text{DTLZ4} = \begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos\left(\frac{y_1\pi}{2}\right)\ldots cos\left(\frac{y_{M-1}\pi}{2}\right) \\ \quad f_2(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos\left(\frac{y_1\pi}{2}\right)\ldots \sin\left(\frac{y_{M-1}\pi}{2}\right) \\ \quad \vdots \\ \quad f_M(\mathbf{x}) = (1+g(\mathbf{x_M}))\sin\left(\frac{y_{M-1}\pi}{2}\right) \\ where: \\ \quad g(\mathbf{x_M}) = \sum_{x_i\in\mathbf{x_M}} (x_i-0.5)^2 \\ \quad y_i = x_i^{\alpha} \\ \quad 0 \le x_i \le 1,\ \forall i = 1,2,\ldots,n \\ \quad |\mathbf{x_M}| = k;\ \ n = M+k-1 \end{cases} \tag{3.16}$$

where $k = 10$ and $\alpha = 100$. Similar to DTLZ2 and DTLZ3, the POF of DTLZ4 is a sphere of radius, $\sum_{m=1}^{M}(f_m^*)^2 = 1$ and the POS is $x_i^* = 0.5,\ \forall x_i \in \mathbf{x_M}$. However, by introducing the mapping of the x-variables, a dense set of solutions exists near the $f_M - f_1$ plane.

$$\text{DTLZ5} = \begin{cases} Minimise: \\ \quad f_1(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos(\theta_1)\cos(\theta_2)\ldots\cos(\theta_{M-1}) \\ \quad f_2(\mathbf{x}) = (1+g(\mathbf{x_M}))\cos(\theta_1)\cos(\theta_2)\ldots\sin(\theta_{M-1}) \\ \quad \vdots \\ \quad f_M(\mathbf{x}) = (1+g(\mathbf{x_M}))\sin(\theta_{M-1}) \\ where: \\ \quad g(\mathbf{x_M}) = \sum_{x_i \in \mathbf{x_M}} x_i^{0.1} \\ \quad \theta_i = \frac{\pi}{4(1+g(r))}(1+2g(r)x_i),\ \forall i = 1,2,\ldots,n \\ \quad 0 \leq x_i \leq 1,\ \forall i = 1,2,\ldots,n \\ \quad |\mathbf{x_M}| = k;\ \ n = M+k-1 \end{cases} \tag{3.17}$$

where $k = 10$. The POF of DTLZ5 is a degenerated curve.

$$\text{DTLZ6} = \begin{cases} Minimise: \\ \quad f_1(x_1) = x_1 \\ \quad \vdots \\ \quad f_{M-1}(\mathbf{x_{M-1}}) = x_{M-1} \\ \quad f_M(\mathbf{x}) = g(\mathbf{x_M})\ \cdot h(f_1,\ldots,f_{M-1},g) \\ where: \\ \quad g(\mathbf{x_M}) = 1 + \frac{9}{|\mathbf{x_M}|}\sum_{x_i \in \mathbf{x_M}} x_i \\ \quad h = M - \sum_{i=1}^{M-1} \frac{f_i}{1+g}(1+\sin(3\pi f_i)) \\ \quad x_i \in \mathbb{R}^{|x_i|},\ \forall i = 1,2,\ldots,M \end{cases} \tag{3.18}$$

where $k = 20$. DTLZ6 is based on Equation (3.12) and has 2^{M-1} disconnected Pareto optimal regions in the search space.

$$\text{DTLZ7} = \begin{cases} \mathit{Minimise:} \\ f_j(\mathbf{x}) = \dfrac{1}{\lfloor \frac{n}{M} \rfloor} \displaystyle\sum_{\lfloor (j-1)\frac{n}{M} \rfloor}^{\lfloor j\frac{n}{M} \rfloor} x_i, \;\; j = 1, \dots, M \\ \mathit{where:} \\ g_j(\mathbf{x}) = f_M(\mathbf{x}) + 4f_j(\mathbf{x}) - 1 \geq 0, \forall j = 1, \dots, (M-1) \\ g_M(\mathbf{x}) = 2f_M(\mathbf{x}) + \displaystyle\min_{i,j=1; i \neq j}^{M-1} [f_i(\mathbf{x}) + f_j(\mathbf{x})] - 1 \geq 0 \\ h = M - \displaystyle\sum_{i=1}^{M-1} \frac{f_i}{1+g}(1 + \sin(3\pi f_i)) \\ 0 \leq x_i \leq 1, \; \forall i = 1, 2, \dots, n \end{cases} \tag{3.19}$$

where $n = 10M$. DTLZ7 is based on Equation (3.11) and has M constraints. Its POF is a combination of a hyperplane (represented by constraint g_M) and a straight line (intersection of the first (M-1) constraints with $f_1 = f_2 = \ldots = f_{M-1}$).

Many benchmark functions for DMOO were based on the ZDT and DTLZ static MOO (SMOO) benchmark functions. The next section discusses DMOO benchmark functions that were proposed in the DMOO literature and the gaps that can be identified in the currently available DMOOPs.

3.2 Dynamic Multi-Objective Optimisation Benchmark functions

This section discusses benchmark functions used to evaluate the performance of DMOO algorithms. Benchmark functions that have been proposed in the DMOO literature are discussed in Section 3.2.1. Sections 3.2.2 and 3.2.3 present new approaches to develop DMOOPs with an isolated POF and deceptive POF respectively. New DMOOPs with complicated POSs are introduced in Section 3.2.4. Characteristics of an ideal set or suite of benchmark functions are presented in Section 3.2.5 and DMOOPs are suggested for each characteristic.

3.2.1 Dynamic Multi-Objective Optimisation Benchmark Functions Currently Used

This section discusses benchmark functions used in the DMOO literature to evaluate whether DMOO algorithms can efficiently solve DMOOPs.

Due to space constraints, only POSs and POFs with different characteristics will be illustrated in this section. In all two-objective figures f_2 refers to gh.

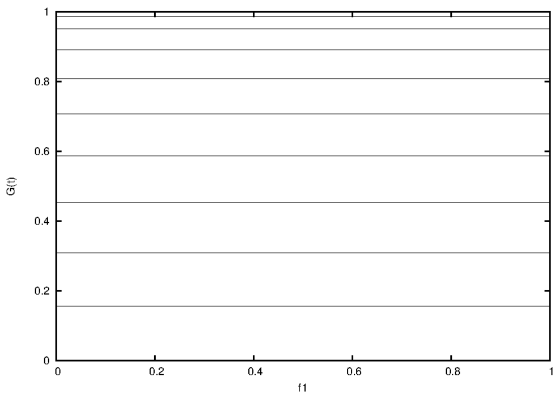
Guan *et al.* [74] suggested to create DMOOPs by replacing objective functions with new objective functions over time. The advantage of Guan *et al.*'s approach is that the new objective function(s) can cause a severe change in the DMOOP and by selecting the objective functions carefully, various types of changes can be incorporated into the DMOOP. Recently, Wang and Li [156] presented a DMOOP where the one subfunction of an objective function changes over time. When objective functions are changed over time, as in the approaches followed by Guan *et al.* and Wang and Li, the objective functions should be selected carefully to ensure that the resulting objective functions hinder the algorithm in finding the POF in various ways as discussed in Section 3.1.1. Another approach was followed by Jin and Sendhoff [90], where a two-objective DMOOP is constructed from a three-objective MOO function. The approach of Jin and Sendhoff has been used by various researchers [110, 111, 112, 108]. However, the adherence to the guidelines of Deb *et al.* by the benchmark functions suggested by Guan *et al.*, Wang and Li, and Jin and Sendhoff will depend on the specific objective functions that are used.

Based on the ZDT [38, 169] and DTLZ [49] functions, Farina *et al.* [58] developed the first suite of DMOOPs, namely the FDA benchmark functions. The FDA functions are constructed in such a way that they are one of the first three DMOOP types of DMOOPs, where either the POS or POF changes over time, or both the POS and POF change over time.

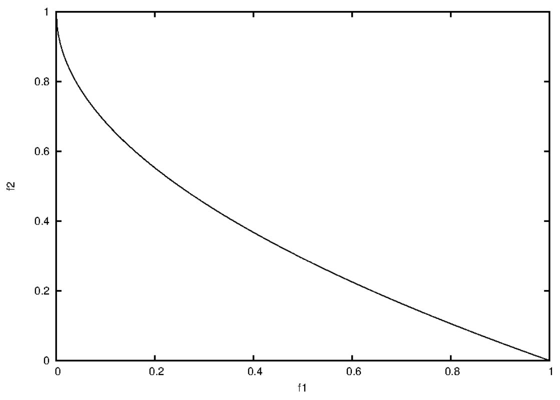
The DMOOPs of the FDA DMOOP suite are easy to construct and the number of decision variables are easily scalable. FDA4 and FDA5 are constructed in such a way that they are easily scalable with regards to both the number of decision variables and the number of objective functions. The FDA benchmark functions are of Type I, II and III DMOOPs and the POF of these DMOOPs is either convex, non-convex or changes from convex to concave over time. Therefore, the FDA DMOOP suite exhibits the characteristics that benchmark functions should have, as defined by Deb *et al.* [38]. The five FDA DMOOPs are defined as follows:

$$\mathsf{FDA1} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x}, t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}, t) \cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}, t))) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}, t) = 1 + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}} (x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \sin(0.5\pi t), \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathsf{x}_{\mathsf{I}} \in [0, 1]; \;\; \mathbf{x}_{\mathsf{II}} = (x_2, \ldots, x_n) \in [-1, 1]^{n-1} \end{cases} \tag{3.20}$$

For FDA1, values in the decision variable space (POS) change over time, but the values in the objective space (POF) remain the same. Therefore, it is a Type I DMOOP. It has a convex POF with $POF = 1 - \sqrt{f_1}$, as illustrated in Figure 3.1(b). The POS is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$ as illustrated in Figure 3.1(a). Appendix C explains how to determine the POS and POF of a DMOOP.



(a) POS

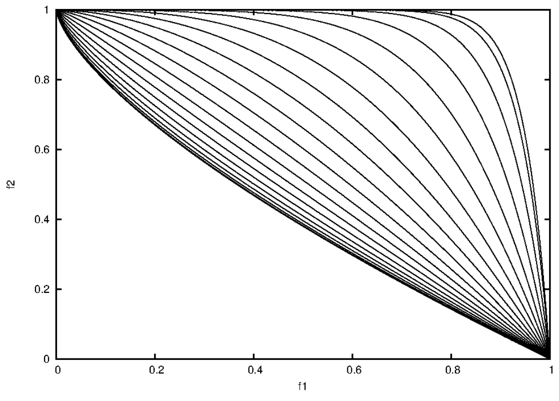


(b) POF

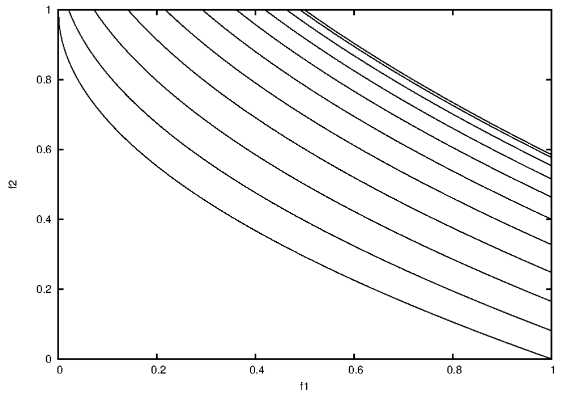
Figure 3.1: POS and POF of FDA1 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$\mathsf{FDA2} = \begin{cases} Minimize: f(\mathsf{x},t) = (f_1(\mathbf{x}_\mathsf{I}), g(\mathbf{x}_\mathsf{II}) \cdot h(\mathbf{x}_\mathsf{III}, f_1(\mathbf{x}_\mathsf{I}), g(\mathbf{x}_\mathsf{II}), t)) \\ f_1(\mathbf{x}_\mathsf{I}) = x_1 \\ g(\mathbf{x}_\mathsf{II}) = 1 + \sum_{x_i \in \mathbf{x}_\mathsf{II}} x_i^2 \\ h(\mathbf{x}_\mathsf{III}, f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H_2(t)} \\ where: \\ H(t) = 0.75 + 0.75\sin(0.5\pi t),\;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ H_2(t) = \left(H(t) + \sum_{x_i \in \mathbf{x}_\mathsf{III}} (x_i - H(t))^2\right)^{-1} \\ \mathbf{x}_\mathsf{I} \in [0,1];\;\; \mathbf{x}_{\mathsf{II}_\mathsf{i}}, \mathbf{x}_{\mathsf{III}_\mathsf{i}} \in [-1,1] \end{cases} \tag{3.21}$$

FDA2 has a POF that changes from convex to concave. It is a Type II DMOOP, since both the POS and POF change over time. For FDA2, $POF = 1 - f_1^{H(t)^{-1}}$, as illustrated in Figure 3.2(a). The POS of FDA2 is $x_i = 0$, $\forall x_i \in \mathbf{x}_\mathsf{II}$ and $x_i = H(t)$, $\forall x_i \in \mathbf{x}_\mathsf{III}$. It should be noted that many researchers refer to FDA2 as a Type III DMOOP due to an error at the DMOOP definition in [58]. However, before the definition of FDA2 in [58], the explanation of the effect of the h function on the DMOOP states that the h function in FDA2 causes the POF to only change through a change in $\mathbf{x}_{III}$ and that FDA2 is therefore a Type II DMOOP.



(a) POF of FDA2



(b) POF of FDA3

Figure 3.2: POF of FDA2 and FDA3 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$\text{FDA3} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x}, t) = (f_1(\mathbf{x}_{\mathsf{I}}, t), g(\mathbf{x}_{\mathsf{II}}, t) \cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}, t))) \\ f_1(\mathbf{x}_{\mathsf{I}}, t) = \sum_{x_i \in \mathbf{x}_{\mathsf{I}}} x_i^{F(t)} \\ g(\mathbf{x}_{\mathsf{II}}, t) = 1 + G(t) + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}} (x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = |sin(0.5\pi t)| \\ F(t) = 10^{2\sin(0.5\pi t)}, \ \ t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathbf{x}_{\mathsf{I}_i} \in [0,1]; \ \ \mathbf{x}_{\mathsf{II}_i} \in [-1,1] \end{cases} \tag{3.22}$$

FDA3 has a convex POF and both the values of the POS and POF change. Therefore it is called a Type II DMOOP. For FDA3, $POF = (1+G(t))\left(1 - \sqrt{\frac{f_1}{1+G(t)}}\right)$, as illustrated in Figure 9.5. The POS is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$, similar to the POS of FDA1 (refer to Figure 3.1(b)). The f_1 function of the two-objective FDA DMOOPs regulate the spread of solutions in objective space. Therefore, when f_1 changes over time, as is the case with FDA3, the spread of solutions in the POF changes over time.

$$\text{FDA4} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x}, t) = (f_1(\mathbf{x}, g(\mathbf{x}_{\mathsf{II}}, t)), \ldots, f_k(\mathbf{x}, g(\mathbf{x}_{\mathsf{II}}, t))) \\ f_1(\mathbf{x}, g, t) = (1 + g(\mathbf{x}_{\mathsf{II}}, t)) \prod_{i=1}^{M-1} \cos\left(\frac{x_i \pi}{2}\right) \\ f_k(\mathbf{x}, g, t) = (1 + g(\mathbf{x}_{\mathsf{II}}, t)) \left(\prod_{i=1}^{M-1} \cos\left(\frac{x_i \pi}{2}\right)\right) \\ \ \sin\left(\frac{y_{M-k+1}\pi}{2}\right), \forall k = 2, \ldots, M-1 \\ \vdots \\ f_m(\mathbf{x}, g, t) = (1 + g(\mathbf{x}_{\mathsf{II}}, t)) \prod_{i=1}^{M-1} \sin\left(\frac{x_1 \pi}{2}\right) \\ where: \\ g(\mathbf{x}_{\mathsf{II}}, t) = \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2 \\ G(t) = |sin(0.5\pi t)|, \ \ t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathbf{x}_{\mathsf{II}} = (x_M, \ldots, x_n); \ \ x_i \in [0,1], \ \forall i = 1, \ldots, n \end{cases} \tag{3.23}$$

For FDA4, values in the decision variable space (POS) change over time, but the values in the objective space (POF) remain the same. Therefore, it is a Type I DMOOP. It has a non-convex POF with the true POF (POF) defined as $f_1^2 + f_2^2 + f_3^2 = 1$ for three objective functions, as illustrated in Figure 3.3. The POS of FDA4 is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$, similar to FDA1 (refer to Figure 3.1(b)).

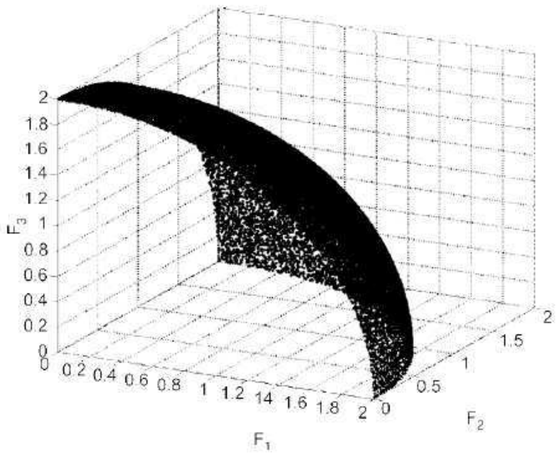


Figure 3.3: POF of FDA4 with three objective functions [58]

$$\mathsf{FDA5} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}, g(\mathbf{x}_{\mathsf{II}},t)), \dots, f_k(\mathbf{x}, g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x},g,t) = (1+g(\mathbf{x}_{\mathsf{II}},t)) \prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right) \\ f_k(\mathbf{x},g,t) = (1+g(\mathbf{x}_{\mathsf{II}},t)) \left(\prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right)\right) \\ \sin\left(\frac{y_{M-k+1}\pi}{2}\right), \forall k = 2, \dots, M-1 \\ \vdots \\ f_m(\mathbf{x},g,t) = (1+g(\mathbf{x}_{\mathsf{II}},t)) \prod_{i=1}^{M-1} \sin\left(\frac{y_1 \pi}{2}\right) \\ where: \\ g(\mathbf{x}_{\mathsf{II}},t) = G(t) + \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2 \\ G(t) = |sin(0.5\pi t)|, \ \ t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ y_i = x_i^{F(t)}, \ \ \forall i = 1, \dots, (M-1) \\ F(t) = 1 + 100 \sin^4(0.5\pi t) \\ \mathbf{x}_{\mathsf{II}} = (x_M, \dots, x_n) \\ x_i \in [0,1], \ \forall i = 1, \dots, n \end{cases} \quad (3.24)$$

FDA5 has a non-convex POF, where both the values in the decision variable space (POS) and the objective space (POF) change over time. Therefore, it is a Type II DMOOP. Furthermore, the spread of solutions in the POF changes over time. For FDA5 with three objective functions, the POF is $f_1^2 + f_2^2 + f_3^2 = (1 + G(t))^2$ as illustrated in Figure 3.4. The POS of FDA5 is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\text{II}}$, similar to FDA1 (refer to Figure 3.1(b)).

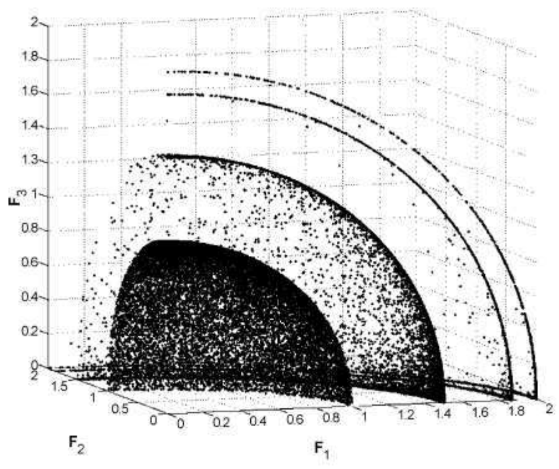


Figure 3.4: POF of FDA5 with three objective functions for four time steps [58]

Many researchers have used the FDA DMOOPs over the years as highlighted in Table 3.1. In Table 3.1 the symbol M indicates that the authors have used a modified version of the specific FDA DMOOP, *I* indicates that the authors have introduced the specific DMOOPs and the column *Other* indicates whether the authors have used DMOOPs other than the FDA set. Table 3.1 shows that most researchers used the FDA1 DMOOP, which is of Type I where the POS changes over time, but the POF remains the same. Clearly, FDA1 is the easiest DMOOP of the FDA suite to solve. Therefore, using the FDA1 DMOOP alone to test whether an algorithm can solve DMOOPs is not sufficient.

Several researchers have used the FDA2 DMOOP. However, the POF of FDA2 changes from a convex to a concave shape only for specific values of the decision variables [46, 117], as can be seen for example in [77, 78] and Figure 4.2. Therefore, even if an algorithm finds Pareto-optimal solutions, it may find a convex POF instead of a concave POF. To address this issue, several modifications to the h or g function of FDA2 have been suggested, as shown in Table 3.2. Underlying problems with FDA3 also lead to several modifications to FDA3 being suggested, as indicated in Table 3.3. In order to test an algorithm's ability to solve Type III DMOOPs, Talukder [144] modified FDA5 to a Type III DMOO, as indicated in Table 3.4.

A generalisation of the FDA functions, DTF, was suggested by Mehnen *et al.* [117]:

$$\mathsf{DTF} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}_\mathsf{I},t), g(\mathbf{x}_\mathsf{II},t)\cdot h(f_1(\mathbf{x}_\mathsf{I},t), g(\mathbf{x}_\mathsf{II},t),t)) \\ f_1(\mathbf{x}_\mathsf{I},t) = x_1^{\beta(t)} \\ g(\mathbf{x}_\mathsf{II},t) = 1 + \sum_{x_i\in\mathbf{x}_\mathsf{II}}((x_i-\gamma(t))^2 - \cos(\omega(t(\tau)))\pi(x_i-\gamma(t))+1) \\ h(f_1,g,t) = 2 - \left(\frac{f_1}{g}\right)^{\alpha(t)} - \left(\frac{f_1}{g}\right)|\sin(\psi(t)\pi f_1))|^{\alpha(t)} \\ where: \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_\mathsf{I}\in[0,1],\ \ \mathbf{x}_{\mathsf{II}_i}\in[-1,1] \end{cases} \tag{3.25}$$

where β represents the spread of solutions, α the curvature of the POF, γ the optimal decision variable values or POS, ψ the number of POF sections, and ω the number of local POFs. For example, a Type II DMOOP can be constructed from DTF by setting the following parameter values: $n = 20$, $\alpha(t) = 0.2 + 4.8t^2$, $\beta(t) = 10^{2\sin(0.5\pi t)}$, $\gamma(t) = \sin(0.5\pi t)$, $\psi(t) = ts$ with $s \in \mathbb{R}$ and $\omega(t) \propto \psi(t)$.

DTF is constructed in such a way that the number of disconnected continuous POF sections, the number of local POFs, the curvature of the POF, the spread of the solutions, and the optimal decision variable values that represent the POS can be easily specified.

Table 3.2: Usage of modified FDA2 DMOOP to test algorithms' performance

Year	Authors	Changes	Modified FDA2 DMOOP
2006	Mehnen *et al.* [117]	Changed the g and H_2 functions to develop a Type III DMOOP. POF is $1 - f_1^{H_2(t)}$ and the POS is $x_i = 0,\ \forall x_i \in \mathbf{x}_\mathsf{II}$ and $x_i = -1,\ \forall x_i \in \mathbf{x}_\mathsf{III}$.	$\begin{cases} f_1(\mathbf{x}_\mathsf{I}) = x_1 \\ g(\mathbf{x}_\mathsf{II}) = 1 + \sum_{x_i\in\mathbf{x}_\mathsf{II}} x_i^2 + \\ \qquad \sum_{x_i\in\mathbf{x}_\mathsf{III}}(x_i+1)^2 \\ h(\mathbf{x}_\mathsf{III}, f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H_2(t)} \\ where: \\ H_2(t) = 0.2 + 4.8t(\tau)^2 \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_\mathsf{I}\in[0,1];\ \ \mathbf{x}_{\mathsf{II}_i}, \mathbf{x}_{\mathsf{III}_i}\in[-1,1] \end{cases}$ (3.26)

Continued on next page

Year	**Authors**	**Changes**	**Modified FDA2 DMOOP**
2007 2010	Deb *et al.* [46] and Liu *et al.* [113]	Developed a Type III DMOOP by changing the h, H and H_2 functions and the calculation of t. The POF is $1-(f_1^2)^{H_2(t)}$ and the POS is $x_i=0, \forall x_i \in \mathbf{x}_{\mathsf{II}}$ and $x_i=-1, \forall x_i \in \mathbf{x}_{\mathsf{III}}$.	$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1+\sum_{x_i\in\mathbf{x}_{\mathsf{II}}} x_i^2 \\ h(\mathbf{x}_{\mathsf{III}}, f_1, g, t) = 1-\left(\left(\frac{f_1}{g}\right)^2\right)^{H_2(t)} \\ where: \\ H_2(t) = H(t)+\sum_{x_i\in\mathbf{x}_{\mathsf{III}}}(x_i - H(t)/4)^2 \\ H(t) = 2\sin(0.5\pi(t-1)) \\ t = 2\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \frac{\tau_t}{\tau_{max}-\tau_t} \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}_i}, \mathbf{x}_{\mathsf{III}_i} \in [-1,1] \\ \tau_{max} = 200,\ \ \lvert\mathbf{x}_{II}\rvert = 5,\ \ \lvert\mathbf{x}_{III}\rvert = 7 \end{cases}$ (3.27)
2007	Zheng [165]	Changed the h function to develop a Type III DMOOP. POF is $(1-\sqrt{f_1})^{H_2(t)}$ and POS is $x_i=0, \forall x_i \in \mathbf{x}_{\mathsf{II}}, \mathbf{x}_{\mathsf{III}}$.	$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1+\sum_{x_i\in\mathbf{x}_{\mathsf{II}}} x_i^2 \\ h(\mathbf{x}_{\mathsf{III}}, f_1, g, t) = \left(1-\sqrt{\frac{f_1}{g}}\right)^{H_2(t)} \\ where: \\ H_2(t) = \left(H(t)+\sum_{x_i\in\mathbf{x}_{\mathsf{III}}}(x_i - H(t))^2\right)^{-1} \\ H(t) = 0.75+0.75\sin(0.5\pi t) \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}_i}, \mathbf{x}_{\mathsf{III}_i} \in [-1,1] \end{cases}$ (3.28)
2008 2009	Isaacs *et al.* [87] and Ray *et al.* [127]	Developed a Type III DMOOP by changing the H_2 function. Very similar to modification made by Mehnen *et al.* [117]. POF is $1-f_1^{H_2(t)}$ and the POS is $x_i=0, \forall x_i \in \mathbf{x}_{\mathsf{II}}$ and $x_i=-1, \forall x_i \in \mathbf{x}_{\mathsf{III}}$.	$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1+\sum_{x_i\in\mathbf{x}_{\mathsf{II}}} x_i^2 + \\ \qquad \sum_{x_i\in\mathbf{x}_{\mathsf{III}}}(x_i+1)^2 \\ h(\mathbf{x}_{\mathsf{III}}, f_1, g, t) = 1-\left(\frac{f_1}{g}\right)^{H_2(t)} \\ where: \\ H_2(t) = 0.2+4.8t^2 \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}_i}, \mathbf{x}_{\mathsf{III}_i} \in [-1,1] \end{cases}$ (3.29)

Continued on next page

Year	Authors	Changes	Modified FDA2 DMOOP
2009	Salazar Lechuga [102]	Changed the h function to develop a Type III DMOOP. $1 - f_1^{H_2(t)}$ is the POF and the POS is $x_i = 0,\ \forall x_i \in \mathbf{x}_{\mathsf{II}}$.	$$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1 + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}} x_i^2 \\ h(\mathbf{x}_{\mathsf{III}}, f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H_2(t)} \\ where: \\ H_2(t) = 0.75 + 0.75\sin(0.5\pi t) \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\quad \mathbf{x}_{\mathsf{II}_i}, \mathbf{x}_{\mathsf{III}_i} \in [-1,1] \end{cases} \quad (3.30)$$
2010	Cámara *et al.* [17] [16] [138]	Changed the H and H_2 functions to develop a Type III DMOOP. $1 - f_1^{H_2(t)}$ is the POF and the POS is $x_i = 0,\ \forall x_i \in \mathbf{x}_{\mathsf{II}}$ and $x_i = -1,\ \forall x_i \in \mathbf{x}_{\mathsf{III}}$.	$$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1 + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}} x_i^2 \\ h(\mathbf{x}_{\mathsf{III}}, f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H_2(t)} \\ where: \\ H_2(t) = H(t) + \sum_{x_i \in \mathbf{x}_{\mathsf{III}}} (x_i - H(t)/2)^2 \\ H(t) = z^{-\cos(\pi t/4)} \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\quad \mathbf{x}_{\mathsf{II}_i}, \mathbf{x}_{\mathsf{III}_i} \in [-1,1] \end{cases} \quad (3.31)$$

Tang *et al.* [149] suggested a similar approach than Farina *et al.*, constructing DMOOPs based on the ZDT functions of Deb *et al.* [38]. Three objective functions are constructed similar to the DMOOPs of Farina *et al.* and provide an additional explanation of how the POF is calculated. For two objective DMOOPs, the following format is used:

$$\begin{cases} Minimise: \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}_{\mathsf{I}}), f_2(\mathbf{x}_{\mathsf{II}})) \\ f_1(\mathbf{x}_{\mathsf{I}}) = f_1(\mathbf{x}_{\mathsf{I}}) \\ f_2(\mathbf{x}_{\mathsf{II}}) = u(t)g(\mathbf{x}_{\mathsf{II}})v(t)\left[h\left(f(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}})v(t)\right)\right] \end{cases} \tag{3.36}$$

with $u(t)$ and $v(t)$ functions of time t. The selection of $u(t)$ and $v(t)$ lead to the construction of various types of DMOOPs:

- $u(t) = 1$ and $v(t)$ that changes over time, create a DMOOP of Type I.
- $v(t) = 1$ and $u(t)$ that changes over time, create a DMOOP of Type III.
- $u(t)$ and $v(t)$ that change over time, create a DMOOP of Type II.

The formulation of the DMOOP using Equation (3.36) can therefore lead to the creation of various types of DMOOPs by changing the values of $v(t)$ and $u(t)$. It is very similar to the FDA DMOOPs, but by formulating the DMOOP in this way, the required

Table 3.1: Usage of FDA DMOOP to test algorithms' performance

Year	**Authors**	**FDA1**	**FDA2**	**FDA3**	**FDA4**	**FDA5**	**Other**
2004	Farina *et al.* [58] (*I*)	x	x	x	x	x	
2005	Amato and Farina [1]	x					
2005	Shang *et al.* [135]			x		x	
2006	Hatzakis and Wallis [76]	x					
2006	Mehnen *et al.* [117]	x	M		x		x
2006	Zheng *et al.* [160]	x	x	x			
2007	Bingul [10]	x					
2007	Cámara *et al.* [19] [18]	x	x				
2007	Deb *et al.* [46]		M				
2007	Liu and Wang [112]		x	x			x
2007	Zheng [165]	x	M	M	x	x	
2007	Zhou *et al.* [166]	x M					
2008	Greeff and Engelbrecht [72]	x			x		x
2008	Isaacs *et al.* [87]	x	M				
2008	Talukder [144] [96]		x	M		M	
2008	Tan and Goh [146]	x					
2008	Wang and Dang [153]	x	x	x			
2009	Chen *et al.* [23]	x				x	
2009	Goh and Tan [67] [66]	x					x
2009	Isaacs *et al.* [88]	x	M				
2009	Ray *et al.* [127]	x	M				
2009	Salazar Lechuga [102]	x	M				
2009	Wang and Li [155]	x					x
2010	Cámara *et al.* [17] [16] [138]	x	M	M	x	x	
2010	Greeff and Engelbrecht [71]	x	x		x	x	
2010	Koo *et al.* [100]	x		x			x
2010	Liu *et al.* [113]	x	M				x
2010	Liu *et al.* [110]		x				x
2010	Wang and Li [156]	x	x	x			x
2011	Helbig and Engelbrecht [78]	x	x	x			x

type of DMOOP can be easily created. Since these functions are based on the ZDT functions, they adhere to the characteristics of benchmark functions recommended by Deb *et al.* An example Type III DMOOP using Equation (3.36) where $v(t) = 1$ and $u(t) = t^2$ is:

$$\begin{cases} Minimise: \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}_{\mathsf{I}}), f_2(\mathbf{x}_{\mathsf{II}})) \\ f_1(\mathbf{x}_{\mathsf{I}}) = 1 - exp(-4x_1)\sin^6(6\pi x_1) \\ f_2(\mathbf{x}_{\mathsf{II}}) = t^2 g\left(1 - \left(\frac{f_1}{g}\right)^2\right) \\ where: \\ g = 1 + 9\left(\frac{\sum_{i=2}^{n} x_i}{n-1}\right)^{0.25} \\ x_i \in [0,1],\ \forall i = 1,2,\ldots,10 \end{cases} \tag{3.37}$$

Table 3.3: Usage of modified FDA3 DMOOP to test algorithms' performance

Year	Authors	Changes	Modified FDA3 DMOOP
2007	Zheng [165]	Modified the f_1 function to develop a Type II DMOOP. POF is $(1+G(t))\left(1-\sqrt{\frac{f_1}{1+G(t)}}\right)$ and POS is $x_i = G(t),\ \forall x_i \in \mathbf{x}_{\mathsf{II}}$.	$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}, t) = \frac{1}{\lvert\mathbf{x}_{\mathsf{I}}\rvert}\sum_{x_i \in \mathbf{x}_{\mathsf{I}}} x_i^{F(t)} \\ g(\mathbf{x}_{\mathsf{II}}, t) = 1 + G(t) + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}}(x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \lvert sin(0.5\pi t)\rvert \\ F(t) = 10^{2\sin(0.5\pi t)} \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}_i} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}_i} \in [-1,1] \end{cases}$ (3.32)
2008	Talukder [144] [96]	Changed FDA3 from a Type II to a Type III DMOOP by modifying the g function. The POF is $(1+G(t))\left(1-\sqrt{\frac{f_1}{1+G(t)}}\right)$ and POS is $x_i = 0,\ \forall x_i \in \mathbf{x}_{\mathsf{II}}$.	$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}, t) = \sum_{x_i \in \mathbf{x}_{\mathsf{I}}} x_i^{F(t)} \\ g(\mathbf{x}_{\mathsf{II}}, t) = 1 + G(t) + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}} x_i^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \lvert sin(0.5\pi t)\rvert \\ F(t) = 10^{2\sin(0.5\pi t)} \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}_i} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}_i} \in [-1,1] \end{cases}$ (3.33)
2010	Cámara *et al.* [17]	Modified the f_1 function to develop a Type II DMOOP. POF is $(1+G(t))\left(1-\sqrt{\frac{f_1}{1+G(t)}}\right)$ and POS is $x_i = G(t),\ \forall x_i \in \mathbf{x}_{\mathsf{II}}$.	$\begin{cases} f_1(\mathbf{x}_{\mathsf{I}}, t) = x_1^{F(t)} \\ g(\mathbf{x}_{\mathsf{II}}, t) = 1 + G(t) + \sum_{x_i \in \mathbf{x}_{\mathsf{II}}}(x_i - G(t))^2 \\ h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \lvert sin(0.5\pi t)\rvert \\ F(t) = 10^{2\sin(0.5\pi t)} \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}_i} \in [-1,1] \\ \mathbf{x}_{\mathsf{II}} = (x_2,\ldots,x_n) \end{cases}$ (3.34)

Table 3.4: Usage of modified FDA5 DMOOP to test algorithms' performance

Year	Authors	Changes	Modified FDA5 DMOOP
2008	Talukder [144]	Changed FDA5 from a Type II to a Type III DMOOP by modifying the g and F functions. POF is $\sum \mathbf{f}_k^2 = (1 + G(t))^2$ and POS is $x_i = 0,\ \forall x_i \in \mathbf{x}_{\text{II}}$.	$\begin{cases} f_1(\mathbf{x}, g, t) = (1 + g(\mathbf{x}_{\text{II}}, t)) \prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right) \\ f_k(\mathbf{x}, g, t) = (1 + g(\mathbf{x}_{\text{II}}, t)) \left(\prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right)\right) \\ \sin\left(\frac{y_{M-k+1}\pi}{2}\right), \forall k = 1, \ldots, M-1 \\ \vdots \\ f_m(\mathbf{x}, g, t) = (1 + g(\mathbf{x}_{\text{II}}, t)) \prod_{i=1}^{M-1} \sin\left(\frac{y_1 \pi}{2}\right) \\ where: \\ g(\mathbf{x}_{\text{II}}, t) = G(t) + \sum_{x_i \in \mathbf{x}_{II}} x_i^2 \\ G(t) = \lvert sin(0.5\pi t)\rvert,\ \ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ y_i = x_i^{F(t)},\ \ \forall i = 1, \ldots, (M-1) \\ F(t) = 10^{2\sin(0.5\pi t)} \\ \mathbf{x}_{\text{II}} = (x_M, \ldots, x_n) \\ x_i \in [0, 1],\ \forall i = 1, \ldots, n \end{cases}$ (3.35)

Wang and Li [155, 156] recently also suggested new Type I DMOOPs that are created by adapting the ZDT functions. These functions are shown in Table 3.6.

Based on the construction guidelines of Farina *et al.* [58], Goh and Tan [67] presented three DMOOPs, namely dMOP1, dMOP2 and dMOP3. dMOP1 and dMOP2 have a POF that changes from convex to concave over time, with dMOP1 being a Type III DMOOP and dMOP2 a Type II DMOOP. In the FDA DMOOP suite, FDA2 also has a POF that changes from convex to concave over time, and FDA2 is a Type II DMOOP. However, dMOP1 and dMOP2 do not suffer from the decision variable selection problem that FDA2 suffers from. dMOP1 tests whether a DMOO algorithm can solve problems where the POF changes from convex to concave but the POS remains the same over time, and dMOP2 adds the difficulty of solving this problem with a changing POS and POF. dMOP3 is very similar to FDA1, however the variable that controls the spread of the POF solutions, x_1 in FDA1, changes over time. This may cause an algorithm to struggle to maintain a diverse set of solutions as the POS changes over time. The dMOP benchmark functions are defined as follows:

$$
\mathsf{dMOP1} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}) \cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}), t)) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1 + 9\sum_{x_i \in \mathbf{x}_{\mathsf{II}}} (x_i)^2 \\ h(f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25 \\ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x}_{\mathsf{I}} = (x_1) \\ \mathbf{x}_{\mathsf{II}} = (x_2, \ldots, x_n) \end{cases} \tag{3.38}
$$

The POF of dMOP1 changes from convex to concave over time, but the POF remains the same. Therefore, it is a Type III problem, with $POF = 1 - f_1^{H(t)}$, as illustrated in Figure 3.5. The POS of dMOP1 is $x_i = 0$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$, similar to FDA2.

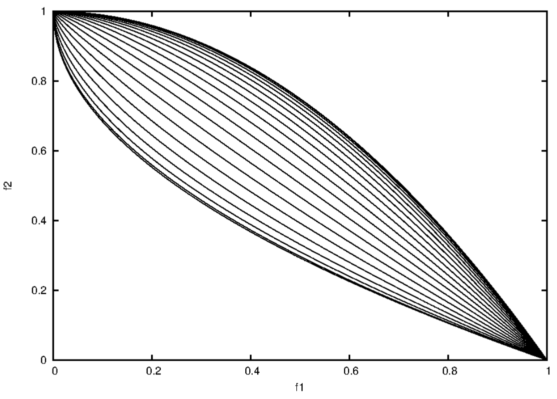


Figure 3.5: POF of dMOP1 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$
\mathsf{dMOP2} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t) \cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t), t)) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}},t) = 1 + 9\sum_{x_i \in \mathbf{x}_{\mathsf{II}}} (x_i - G(t))^2 \\ h(f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \\ G(t) = \sin(0.5\pi t) \\ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x}_{\mathsf{I}} = (x_1) \\ \mathbf{x}_{\mathsf{II}} = (x_2, \ldots, x_n) \end{cases} \tag{3.39}
$$

dMOP2 has a POF that changes from convex to concave, where the values in both the POS and POF change. Therefore, dMOP2 is a Type II problem, with $POF = 1 - f_1^{H(t)}$, similar to dMOP1 (refer to Figure 3.5). The POS of dMOP2 is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$, similar to FDA1 (refer to Figure 3.1(b)).

$$\mathsf{dMOP3} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t)\cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_r \\ g(\mathbf{x}_{\mathsf{II}},t) = 1 + 9\sum_{x_i\in\mathbf{x}_{\mathsf{II}}\setminus x_r}(x_i - G(t))^2 \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G(t) = \sin(0.5\pi t),\ \ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ x_i \in [0,1];\ \ r = \bigcup(1,2,\ldots,n) \end{cases} \tag{3.40}$$

dMOP3 has a convex POF where the POS changes over time, but the POF remains the same. dMOP3 is therefore a Type I DMOOP and the spread of the POF solutions changes over time. Similar to FDA1, for dMOP3, $POF = 1 - \sqrt{f_1}$ (refer to Figure 3.1(b)) and the POS is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}} \setminus x_r$ (refer to Figure 3.1(a)).

More recently, Li and Zhang [105] and Deb *et al.* [48] presented MOOPs with decision variable dependencies (or linkages). Zhou *et al.* [166] modified FDA1 to incorporate dependencies between the decision variables. The modified FDA1 DMOOP is defined as follows:

$$\mathsf{ZJZ} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t)\cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}},t) = 1 + \sum_{x_i\in\mathbf{x}_{\mathsf{II}}}\left(x_i - G(t) - x_1^{H(t)}\right)^2 \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ G(t) = \sin(0.5\pi t) \\ H(t) = 1.5 + G(t) \\ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1];\ \ \mathbf{x}_{\mathsf{II}} = (x_2,\ldots,x_n) \in [-1,2]^{n-1} \end{cases} \tag{3.41}$$

For ZJZ, the values of both the POS and POF change over time. Therefore, it is a Type II DMOOP. ZJZ's POF is similar to dMOP1 (refer to Figure 3.5) and changes from convex to concave over time, with $POF = 1 - f_1^{H(t)}$. However, there are non-linear dependencies between the decision variables that make the DMOOP more difficult to solve. The POS of ZJZ is $x_i = G(t) + x_1^{H(t)}$, $\forall x_i \in \mathbf{x}_{\text{II}}$, as illustrated in Figure 3.6. Changes made to FDA1 to develop new DMOOPs are summarised in Table 3.5.

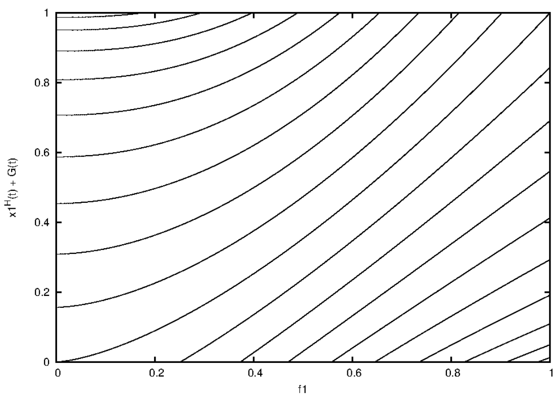


Figure 3.6: POS of ZJZ with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

Table 3.5: Usage of modified FDA1 DMOOP to test algorithms' performance

Year	Authors	Changes	Modified FDA1 DMOOP
2007	Zhou *et al.* [166]	Modified FDA1 from a Type I to a Type II DMOOP with non-linear dependencies between the decision variables. POF is $1 - f_1^{H(t)}$ and POS is $x_i = G(t) + x_1^{H(t)}$, $\forall x_i \in \mathbf{x}_{\text{II}}$.	$$\begin{cases} f_1(\mathbf{x}_{\text{I}}) = x_1 \\ g(\mathbf{x}_{\text{II}}, t) = 1 + \sum_{x_i \in \mathbf{x}_{\text{II}}} \left(x_i - G(t) - x_1^{H(t)}\right)^2 \\ h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ G(t) = \sin(0.5\pi t), \quad t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ H(t) = 1.5 + G(t) \\ \mathbf{x}_{\text{I}} \in [0,1]; \quad \mathbf{x}_{\text{II}} = (x_2, \dots, x_n) \in [-1,1]^{n-1} \end{cases} \tag{3.42}$$

Another shortcoming of the FDA DMOOP suite is that all DMOOP objective functions consist of decision variables with the same rate of change over time. Koo *et al.* [100] suggested two new benchmark functions where each decision variable has its own rate of change, except the variable x_1 that controls the spread of solutions. These two functions, DIMP1 and DIMP2, are defined as follows:

$$\mathsf{DIMP1} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t)\cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}},t) = 1 + \sum_{x_i\in\mathbf{x}_{\mathsf{II}}}(x_i - G_i(t))^2 \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^2 \\ where: \\ G_i(t) = \sin\left(0.5\pi t + 2\pi\left(\frac{i}{n+1}\right)\right)^2, \; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} = (x_1) \in [0,1], \; \mathbf{x}_{\mathsf{II}} = (x_2, x_3, \ldots, x_n) \in [-1,1]^{n-1} \end{cases} \tag{3.43}$$

The POS of DIMP1 changes over time, but the POF remains the same. Therefore, DIMP1 is a Type I DMOOP, with $POF = 1 - f_1^2$ (as illustrated in Figure 3.7) and the POS is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$, similar to FDA1 (refer to Figure 3.1(a)).

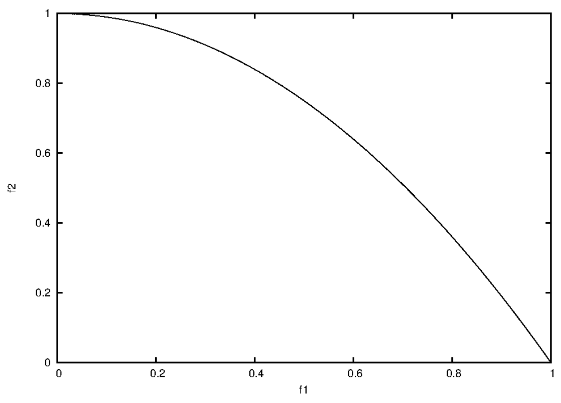


Figure 3.7: POF of DIMP1 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$\mathsf{DIMP2} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t)\cdot (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}},t) = 1 + 2(n-1) + \\ \quad \sum_{x_i\in\mathbf{x}_{\mathsf{II}}}[(x_i - G_i(t))^2 - \\ \qquad 2\cos(3\pi(x_i - G_i(t)))] \\ h(f_1,g) = 1 - \sqrt{\frac{f_1}{g}} \\ where: \\ G_i(t) = \sin\left(0.5\pi t + 2\pi\left(\frac{i}{n+1}\right)\right)^2, \; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ \mathbf{x}_{\mathsf{I}} \in [0,1], \; \mathbf{x}_{\mathsf{II}} \in [-2,2]^{n-1} \end{cases} \tag{3.44}$$

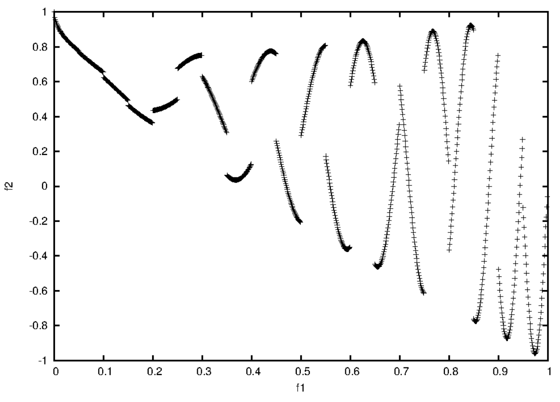
DIMP2 is a Type I problem, since its POS changes over time but its POF remains the same. Similar to FDA1, DIMP2's POF is $1-\sqrt{f_1}$ (refer to Figure 3.1(b)) and the POS is $x_i = G(t)$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$ (refer to Figure 3.1(a)).

The FDA and dMOP MOOPs only contain DMOOPs with a continuous POF. Two discontinous functions, namely $\mathsf{TP1}_{\mathsf{mod}}$ and $\mathsf{TP2}_{\mathsf{mod}}$, were presented by Greeff and Engelbrecht [72]. However, these two functions do not allow easy scalability of the number of decision variables. Therefore, $\mathsf{TP1}_{\mathsf{mod}}$ and $\mathsf{TP2}_{\mathsf{mod}}$ do not adhere to the characteristics of benchmark functions that are recommended by Deb *et al.* Recently, Helbig and Engelbrecht [78] presented two DMOOPs with a discontinuous POF, namely HE1 and HE2. These two functions are based on the ZDT3 [169] MOOP that was developed in such a way that it adheres to the characteristics recommended by *Deb et al.* HE1 and HE2 were developed by adapting ZDT3 to be dynamic and therefore adhere to the benchmark function characteristics recommended by Deb *et al.* HE1 and HE2 are defined as:

$$\mathsf{HE1} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}) \cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}), t)) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_1 \\ g(\mathbf{x}_{\mathsf{II}}) = 1 + \frac{9}{n-1}\sum_{x_i \in \mathbf{x}_{\mathsf{II}}} x_i \\ h(f_1, g, t) = 1 - \sqrt{\frac{f_1}{g}} - \frac{f_1}{g}\sin(10\pi t f_1) \\ where: \\ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x}_{\mathsf{I}} = (x_1); \;\; \mathbf{x}_{\mathsf{II}} = (x_2, \ldots, x_n) \end{cases} \tag{3.45}$$

$$\mathsf{HE2} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}) \cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}}), t)) \\ f_1(\mathbf{x}_{\mathsf{I}}) = x_i \\ g(\mathbf{x}_{\mathsf{II}}) = 1 + \frac{9}{n-1}\sum_{x_i \in \mathbf{x}_{\mathsf{II}}} x_i \\ h(f_1, g, t) = 1 - \left(\sqrt{\frac{f_1}{g}}\right)^{H(t)} - \left(\frac{f_1}{g}\right)^{H(t)}\sin(10\pi f_1) \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25; \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x}_{\mathsf{I}} = (x_1); \;\; \mathbf{x}_{\mathsf{II}} = (x_2, \ldots, x_n) \end{cases} \tag{3.46}$$

Both HE1 and HE2 have a discontinuous POF, with various disconnected continuous sub-regions. Both are Type III DMOOPs, since their POFs change over time, but their POSs remain the same. For HE1, $POF = 1 - \sqrt{f_1} - f_1\sin(10\pi t f_1)$ as illustrated in Figure 3.8(a), and for HE2, $POF = 1 - \left(\sqrt{f1}\right)^{H(t)} - f_1^{H(t)}\sin(0.5\pi f_1)$ as illustrated in Figure 9.9. The POS for both HE1 and HE2 is $x_i = 0$, $\forall x_i \in \mathbf{x}_{\mathsf{II}}$, similar to FDA2.


(a) POF of HE1

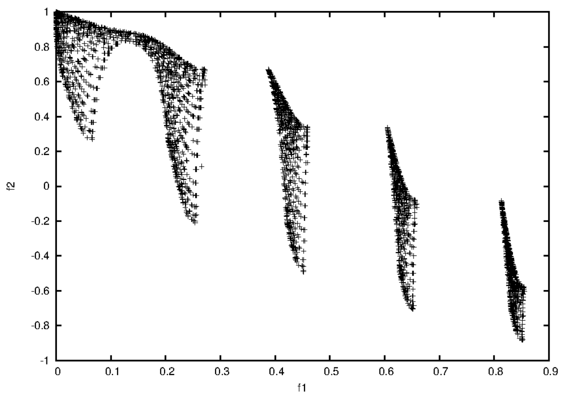

(b) POF of HE2

Figure 3.8: POF of HE1 and HE2 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

Avdagić *et al.* [2] introduced an adaptation of the DTLZ problems to develop the following types of benchmark functions: Type I DMOOP where the POS changes coherently over time, but the POF remains the same, Type II DMOOP where the shape of the POS continuously changes and the POF also changes over time, and a Type II DMOOP where the number of objective functions change over time [2]. These benchmark functions are developed from the following general equation:

$$DTLZ_{Av} = \begin{cases} Minimize: & q(\mathbf{x}) = (q_1(\mathbf{x}), \dots, q_m(\mathbf{x})) \\ q_1(\mathbf{x}) & = a_1 x_1^{c_1} x_2^{c_1} \dots x_{m-1}^{c_1} (1 - x_m)^{c_1} g_1(\mathbf{x}) + b_1 \\ q_2(\mathbf{x}) & = a_2 x_1^{c_2} x_2^{c_2} \dots (1 - x_{m-1})^{c_2} (1 - x_m)^{c_2} g_2(\mathbf{x}) + b_2 \\ \vdots & \\ q_{m-1}(\mathbf{x}) & = a_{m-1} x_1^{c_{m-1}} (1 - x_2)^{c_{m-1}} \dots (1 - x_{m-1})^{c_{m-1}} (1 - x_m)^{c_{m-1}} \\ & g_{m-1}(\mathbf{x}) + b_{m-1} \\ q_m(\mathbf{x}) & = a_m (1 - x_1)^{c_m} (1 - x_2)^{c_m} \dots (1 - x_{m-1})^{c_m} (1 - x_m)^{c_m} g_m(\mathbf{x}) \\ & + b_m \\ where: & \\ g_i = 1 - d_i \cos(20\pi x_i) & \\ a_i, b_i, c_i, d_i \in \mathbb{R} & \end{cases} \tag{3.47}$$

A Type I DMOOP with a continuously changing POS is created by using Equation (3.47) and setting the following parameter values: $a_i = 1$, $d_i = 0$, $b_i = b_i k$, where k represents the iteration and $c_i = 1$ or $c_i = 2$. Similarly, a Type II DMOOP with continuously changing POS and POF are developed by setting the following parameter values: $a_i = 1$, $b_i = b_i k$, $c_i k = 5 b_i k$ and $d_i = 0$. To develop a Type II DMOOP with a changing number of objectives, the same parameters are used as those spesified for the Type II DMOOP, with two objective functions being used for a certain number of iterations and then three objective functions are used for the other iterations. These additional types of DMOOPs, which are not part of the FDA benchmark function set, may become important if these kind of changes occur in a real-world problem.

Recently, Huang *et al.* [84] pointed out that all DMOOPs assume that the current found POS does not affect the future POS or POF. To the best knowledge of the author of this thesis, none of the suggested DMOOPs have a POS or POF that depends on the previous POS or POF. Furthermore, most DMOOPs consist of a static number of decision variables and objective functions. Therefore, Huang *et al.* [84] introduced four DMOOPs that incorporate these scenarios, defined as follows:

$$\mathsf{T1} = \begin{cases} Minimize : f(\mathbf{x},t) = (f_1(\mathbf{x},t), f_2(\mathbf{x},t)) \\ f_1(\mathbf{x},t) = \sum_{i=1}^{d_1(t)} \left(x_i^2 - 10\cos(2\pi x_i) + 10\right) \\ f_2(\mathbf{x},t) = (x_1 - 1)^2 + \sum_{i=2}^{d_2(t)} \left(x_i^2 - x_{i-1}\right)^2 \\ where : \\ d_1(t) = \lfloor n |\sin(t)| \rfloor \\ d_2(t) = \lfloor n |\cos^3(2t)| \rfloor \\ t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \end{cases} \tag{3.48}$$

with d_1 and d_2 varying the number of decision variables over time. The minimum for f_1 is 0 and the POS for f_1 is $x_i = 0$, $\forall i = 1, \ldots, d_1(t)$. The minimum for f_2 is 0 with the POS $x_i = 1$, $\forall i = 1, \ldots, d_2(t)$. Both the POF and POS remain static, but the number of decision variables changes over time. Therefore, T1 is a type IV DMOOP.

$$\mathsf{T2} = \begin{cases} Minimize : f(\mathbf{x},t) = (f_1(\mathbf{x},t), \ldots, f_m(\mathbf{x},t)) \\ f_1(\mathbf{x},t) = (1 + g(\mathbf{x}_{\mathsf{II}})) \prod_{i=1}^{m(t)-1} \cos\left(\frac{\pi x_i}{2}\right) \\ f_k(\mathbf{x},t) = (1 + g(\mathbf{x}_{\mathsf{II}})) \prod_{i=1}^{m(t)-k} \cos\left(\frac{\pi x_i}{2}\right) \quad \sin\left(\frac{\pi x_{m(t)-k+1}}{2}\right), \\ \forall k = 2, \ldots, m(t) - 1 f_m(\mathbf{x},t) = (1 + g(\mathbf{x}_{\mathsf{II}})) \prod_{i=1}^{m(t)-1} \sin\left(\frac{\pi x_1}{2}\right) \\ where : \\ g(\mathbf{x}_{\mathsf{II}}) = \sum_{i=1}^{m(t)} (x_i - 0.5)^2 \\ m(t) = \lfloor M |\sin(0.5\pi t)| \rfloor, \;\; t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathbf{x}_{\mathsf{i}} \in [0,1] \end{cases} \tag{3.49}$$

with M representing the maximum number of objective functions and m varying the number of objective functions over time. T2 is a Type III DMOOP, since its POF changes over time, but its POS remains the same. The POS of T2 is $x_i = 0.5,\ \forall i = 1, \dots, m(t)$ and the POF is $\sum_i^{m(t)} \mathbf{f}_i^2 = 1$.

$$\mathsf{T3} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x},t), f_2(\mathbf{x},t)) \\ f_1(\mathbf{x},t) = R(\mathbf{x},t)\cos\left(\frac{\pi x_1}{2}\right) \\ f_2(\mathbf{x},t) = R(\mathbf{x},t)\sin\left(\frac{\pi x_1}{2}\right) \\ where: \\ R(\mathbf{x},t) = \bar{R}(\mathbf{x},\mathrm{t}-1,t) + G(\mathbf{x},t) \\ \bar{R}(\mathbf{x},t) = \frac{1}{P}\sum_j^P R_j(\mathbf{x},t-1) \\ \bar{R}(\mathbf{x},-1) = 1 \\ G(\mathbf{x},t) = \sum_{i=2}^n \left(x_i - \bar{R}(\mathbf{x},t-1)\right)^2,\ \ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_1 \in [0,1],\ \ x_i \in [\bar{R}(\mathbf{x},t) - 100, \bar{R}(\mathbf{x},t) + 100],\ \ \forall i = 2, \dots, n \end{cases} \tag{3.50}$$

with the value of $R(\mathbf{x},t)$ depending on previous values of R. Therefore, if a slight error occurs with regards to the found value of R at time t, this error will increase over time, influencing the algorithm's ability to find the solutions at the next time steps. Both the POS and POF remain static. Therefore, T3 is a Type IV DMOOP. The POS is $x_i = \bar{R}(\mathbf{x},t-1),\ \forall i = 2, \dots, n$. The POF is $f_1^2 + f_2^2 = 1$. Similar to T1, T4 is a type IV DMOOP, defined as:

$$\mathsf{T4} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x},t), f_2(\mathbf{x},t)) \\ f_1(\mathbf{x},t) = \sum_{i=1}^n \left(x_i^2 - 10\cos(2\pi x_i) + 10\right) \\ f_2(\mathbf{x},t) = (x_1 - r(t))^2 + \sum_{i=2}^n \left(x_i^2 - x_{i-1}\right)^2 \\ where: \\ r(\mathbf{x},t) = \frac{1}{n}\sum_{x_i \in \mathbf{x}} (x_i - 0) \\ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \end{cases} \tag{3.51}$$

with r representing the average error of the decision variables of the selected POS (POS^*). Since the POS of T4 is $x_i = 0,\ \forall i = 1, 2, \dots, n$, the average error of the decision variables of POS^* is $r(\mathbf{x},t) = \frac{1}{n}\sum_{x_i \in \mathbf{x}} (x_i - 0)$. The selected trade-off solution set, POS^*, is derived from the current POS by a decision making mechanism used by the decision maker. Therefore, for T4, the POF depends on the decision making mechanism used at previous time steps.

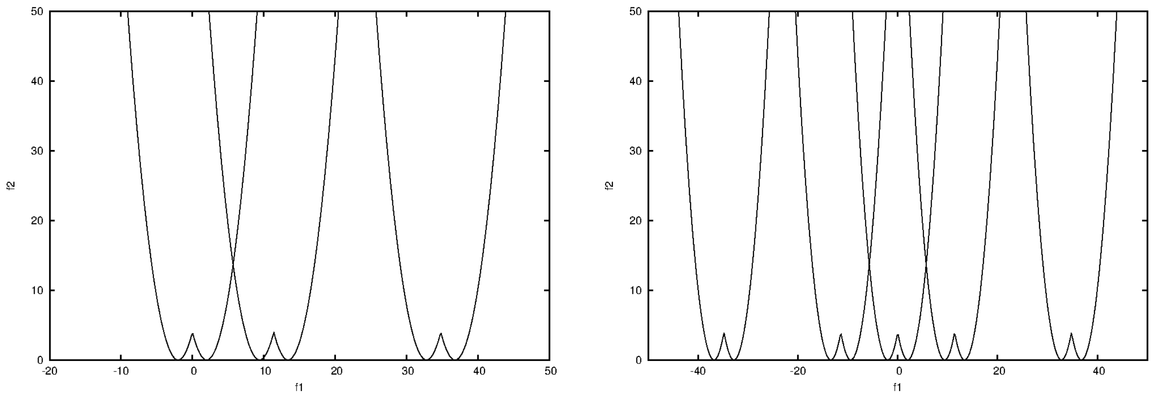
Mehnen *et al.* [117] suggested that simpler benchmark functions are required to analyse the effect of different dynamic properties in a more isolated manner. For this reason, they presented the DSW DMOOPs generator that are based on the static MOOP of Shaffer [131]. The DSW DMOOPs are parabolic and are similar to the sphere function that are typically used to test whether an algorithm can solve DSOOPs. The DSW benchmark generator is defined as:

$$\mathsf{DSW} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x},t), f_2(\mathbf{x},t)) \\ f_1(\mathbf{x},t) = (a_{11}x_1 + a_{12}|x_1| - b_1 G(t))^2 + \sum_{i=2}^{n} x_i^2 \\ f_2(\mathbf{x},t) = (a_{21}x_1 + a_{22}|x_1| - b_2 G(t) - 2)^2 + \sum_{i=2}^{n} x_i^2 \\ where: \\ G(t) = t(\tau)s, \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \end{cases} \tag{3.52}$$

with s representing the severity of change. Using Equation (3.52), the following three benchmark functions are created:

$$DSW1: \begin{cases} x \in [-50, 50]^n, \, a_{11} = 1, \, a_{12} = 0, \, a_{21} = 1, \\ a_{22} = 0, \, b_1 = 1, \, b_2 = 1 \end{cases} \tag{3.53}$$

DSW1 has a dynamic POF and POS, and is therefore a Type II DMOOP. The POS of DSW1 is $x_1 \in [G(t), G(t) + 2]$ and $x_i = 0, \; \forall i = 2, 3, \ldots, n$. The POF is $POF = \left(\sqrt{f_1} - 2\right)^2$ with $f_1 = (x_1 - G(t))^2$, as illustrated in Figure 3.9(a). DSW1 is similar to the spherical SOOP function where the center of the sphere is linearly shifted over time.



(a) POF of DSW1 (b) POF of DSW2

Figure 3.9: POF of DSW1 and DSW2 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$DSW2: \begin{cases} x \in [-50, 50]^n,\, a_{11} = 0,\, a_{12} = 1,\, a_{21} = 0, \\ a_{22} = 1,\, b_1 = 1,\, b_2 = 1 \end{cases} \tag{3.54}$$

Both the POS and POF of DSW2 change over time. Therefore, DSW2 is a Type II DMOOP. DSW2 has a disconnected POS, with $x_1 \in [-G(t)-2, -G(t)] \cup [G(t), G(t)+2]$ and $x_i = 0$, $\forall i = 2, 3, \ldots, n$. If a periodical $G(t)$ is used, the POSs will join and depart periodically. The POF of DSW2 is similar to that of DSW1, namely $POF = \left(\sqrt{f_1} - 2\right)^2$, but with $f_1 = (|x_1| - G(t))^2$, as illustrated in Figure 3.9(b).

$$DSW3: \begin{cases} x \in [-50, 50]^n,\, a_{11} = 1,\, a_{12} = 0,\, a_{21} = 1, \\ a_{22} = 0,\, b_1 = 0,\, b_2 = 1 \end{cases} \tag{3.55}$$

DSW3 has a changing POF and POS, and is therefore a Type II DMOOP. For DSW3 the POS is $x_1 \in [0, G(t)+2]$ and the POF is $POF = \left(\sqrt{f_1} - G(t) - 2\right)^2$ with $f_1 = x_1^2$. Setting $b_1 = 0$ causes one border of the POS interval for x_1, namely $G(t)+2$, to change over time, while the other border, 0, remains static.

The DMOOPs that have been discussed above are summarised in Table 3.6 (excluding the FDA and modified FDA functions summarised in Tables 3.1 to 3.4).

None of the DMOOPs discussed in this section have an isolated or deceptive POF. The next section discusses an approach to construct DMOOPs with an isolated POF.

Table 3.6: Usage of other DMOOP to test algorithms' performance

Year	**Authors**	**Other DMOOPs**	**DMOOPs Definition**
2004 2006 2007 2007 2010	Jin and Sendhoff [90] (I) Liu and Wang [111] Li *et al.* [108] Liu and Wang [112] Liu *et al.* [110]	Constructing two-objective DMOOPs from a three-objective MOOP. Various f_1 and f_2 functions can be used to create Type I to III DMOOPs.	$$\begin{cases} Minimize: \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), f_3(\mathbf{x})) \\ \text{Is changed to:} \\ Minimize: (F_1, F_2) \\ where: \\ F_1 = wf_1(\mathbf{x}) + (1-w)f_2(\mathbf{x}) \\ F_2 = wf_1(\mathbf{x}) + (1-w)f_3(\mathbf{x}) \\ \text{with } w \text{ changing over time} \end{cases} \tag{3.56}$$

Continued on next page

Year	Authors	Other DMOOPs	DMOOPs Definition
2005	Guan *et al.* [74]	DMOOPs created by replacing objective functions with new objective functions over time. G_1 is an example of a Type III DMOOP.	$$\mathsf{G}_1 = \begin{cases} Minimise: \\ \begin{cases} G=(f_1,f_2) \text{ for } t \\ G=(f_1,f_2') \text{ for } t^* \end{cases} \\ where: \\ f_1 = x_1 \\ f_2 = g(\mathbf{x})\left(1-\left(\frac{x_1}{g(\mathbf{x})}\right)^2\right) \\ f_2' = g(\mathbf{x})\left(1-\sqrt{\frac{x_1}{g(\mathbf{x})}}\right) \\ g(\mathbf{x}) = 1+\frac{9}{n-1}\sum_{i=2}^{n} x_i \\ x_i \in [0,1] \end{cases} \quad (3.57)$$ $$\mathsf{G}_2 = \begin{cases} Minimise: \\ \begin{cases} G=(f_1,f_2,f_3,f_4) \text{ for } t \\ G=(f_1,f_2,f_3,f_4') \text{ for } t^* \end{cases} \\ where: \\ f_1 = (x_1-2)^2+4x_2^2 \\ f_2 = x_1^2+(x_2-3)(x_3-3) \\ f_3 = x_2x_3x_4 \\ f_4 = x_1x_4+x_2x_3 \\ f_4' = 1/\left(x_2^{1.5}x_3^{2.5}x_4\right) \\ x_i \in [1,10] \end{cases} \quad (3.58)$$
2005	Guan *et al.* [74] (cont.)	DMOOPs created by replacing objective functions with new objective functions over time.	$$\mathsf{G}_3 = \begin{cases} Minimise: \\ \begin{cases} G=(f_1,f_2,f_3,f_4) \text{ for } t \\ G=(f_1,f_2,f_3',f_4') \text{ for } t^* \end{cases} \\ where: \\ f_1 = (x_1-2)^2+4x_2^2 \\ f_2 = x_1^2+(x_2-3)(x_3-3) \\ f_3 = 1-exp(-4x_1)\sin^6(6\pi x_1) \\ f_4 = x_1x_4+x_2x_3 \\ f_3' = x_2x_3x_4 \\ f_4' = 1/\left(x_2^{1.5}x_3^{2.5}x_4\right) \\ x_i \in [1,10] \end{cases} \quad (3.59)$$

Continued on next page

Year	Authors	Other DMOOPs	DMOOPs Definition
2006	Mehnen *et al.* [117]	DMOOP DTF enabling easy specification of the number of separated POF sections, the number of local POFs, the curvature of the POF, the spread of the solutions and the optimal decision variable values that represent the POS. Type I-III DMOOPs can be created. DSW DMOOP generator that is based on the static MOOP of Shaffer. DMOOP Types I-III can be created.	Equation (3.25) Equations (3.52) to (3.55)
2007	Tang *et al.* [149]	DMOOPs based on the ZDT functions of Deb *et al.* [38]. Can construct DMOOPs of Type I-III.	Equation (3.36)

Continued on next page

Year	Authors	Other DMOOPs	DMOOPs Definition
2008	Greeff and Engelbrecht [72]	TP1$_{\mathsf{mod}}$ and TP2$_{\mathsf{mod}}$ DMOOPs with discontinuous POFs. Both TP1$_{\mathsf{mod}}$ and TP2$_{\mathsf{mod}}$ are Type III DMOOPs.	$TP1_{mod}$: $$\begin{cases} Minimize: \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x})) \\ f_1(\mathbf{x}) = \begin{cases} -x & \text{for } x \leq 1 \\ -2+x & \text{for } 1 < x \leq 3 \\ 4-x & \text{for } 3 < x \leq 4 \\ -4+x & \text{for } x > 4 \end{cases} \\ f_2(\mathbf{x}) = (x-5)^2 + G(t) \\ where: \\ G(t) = \lvert \sin(0.5\pi t) \\ t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ -100 \leq x \leq 100 \end{cases} \quad (3.60)$$ $TP2_{mod}$: $$\begin{cases} Minimize: \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x})) \\ f_1(\mathbf{x}) = 2 + (x_2-1)^2 - 10c_1G(t) + \\ \qquad (x_1-2)^2 \\ f_2(\mathbf{x}) = 9x_1 + (x_2-1)^2 - 10c_2G(t) \\ where: \\ c_1(\mathbf{x}) = \begin{cases} c_1 & \text{for } c_1 \leq 0 \\ 0 & \text{for } c_1 > 0 \end{cases} \\ c_2(\mathbf{x}) = \begin{cases} c_2 & \text{for } c_2 \leq 0 \\ 0 & \text{for } c_2 > 0 \end{cases} \\ G(t) = \lvert \sin(0.5\pi t) \\ t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ x_1, x_2 \in [-20, 20] \end{cases} \quad (3.61)$$

Continued on next page

Year	Authors	Other DMOOPs	DMOOPs Definition
2009	Avdagić *et al.* [2]	Adapted the DTLZ problems to develop a Type II homogenous DMOOP where the POS changes uniformly at each iteration, a non-homogenous Type II DMOOP where the POS continuously changes and results in the POF that changes as well, and a non-homogenous Type II DMOOP where the number of objective functions change over time.	Equation (3.47)
2009	Goh and Tan [67] [66]	Three DMOOPs, namely dMOP1 (Type III), dMOP2 (Type II) and dMOP3 (Type I). dMOP1 and dMOP2 have a POF that changes from convex to concave over time. dMOP3 is very similar to FDA1, however the variable that controls the spread of the POF solutions changes over time.	Equations (3.38) to (3.40)

Continued on next page

Year	Authors	Other DMOOPs	DMOOPs Definition
2009 2010	Wang and Li [155] Wang and Li [156]	Modified ZDT functions to create the Type I DMZDT DMOOPs. POF of DMZDT1 is $1-\sqrt{f_1}$ and the POS is $\frac{\lvert x_i-t/n_t\rvert}{H(t)}=0$.	DMZDT1: $\begin{cases} Minimize: \mathbf{f}(\mathbf{x},t)=(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t)\cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x}_{\mathsf{I}})=x_1 \\ g(\mathbf{x}_{\mathsf{II}},t)=1+\frac{9\sum_{x_i\in\mathbf{x}_{\mathsf{II}}}\lvert y_i(t)\rvert}{D-1} \\ h(f_1,g)=1-\sqrt{\frac{f_1}{g}} \\ where: \\ y_i(t)=\frac{\lvert x_i-t/n_t\rvert}{H(t)}, \quad \forall i=2,\dots,D \\ H(t)=\max\{\lvert 1-\frac{t}{n_t}\rvert, \lvert -1-\frac{t}{n_t}\rvert\} \\ t=\left\lfloor\frac{f_c}{FES_c}\right\rfloor \\ \mathbf{x}_{\mathsf{I}}\in[0,1]; \quad \mathbf{x}_{\mathsf{II}}=(x_2,\dots,x_n)\in[-1,1]^{n-1} \end{cases} \quad (3.62)$
		POF of DMZDT2 is $1-f_1^2$ and the POS is $\frac{\lvert x_i-t/n_t\rvert}{H(t)}=0$.	DMZDT2: $\begin{cases} Minimize: \mathbf{f}(\mathbf{x},t)=(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t)\cdot h(f_1(\mathbf{x}_{\mathsf{I}}), g(\mathbf{x}_{\mathsf{II}},t))) \\ f_1(\mathbf{x}_{\mathsf{I}})=x_1 \\ g(\mathbf{x}_{\mathsf{II}},t)=1+\frac{9\sum_{x_i\in\mathbf{x}_{\mathsf{II}}}\lvert y_i(t)\rvert}{D-1} \\ h(f_1,g)=1-\left(\frac{f_1}{g}\right)^2 \\ where: \\ y_i(t)=\frac{\lvert x_i-t/n_t\rvert}{H(t)}, \quad \forall i=2,\dots,D \\ H(t)=max\{\lvert 1-\frac{t}{n_t}\rvert, \lvert -1-\frac{t}{n_t}\rvert\} \\ t=\left\lfloor\frac{f_c}{FES_c}\right\rfloor \\ \mathbf{x}_{\mathsf{I}}\in[0,1]; \quad \mathbf{x}_{\mathsf{II}}=(x_2,\dots,x_n)\in[-1,1]^{n-1} \end{cases} \quad (3.63)$

Continued on next page

Year	Authors	Other DMOOPs	DMOOPs Definition
2009 2010	Wang and Li [155] Wang and Li [156]	(continued) POF of DMZDT3 is $1-\sqrt{f_1}-f_1\sin(10\pi f_1)$. The POF is discontinuous. The POS is $\frac{\lvert x_i-t/n_t\rvert}{H(t)}=0$. DMZDT4 has many local POFs. POF of DMZDT4 is $1-\sqrt{f_1}$ and the POS is $\frac{\lvert x_i-t/n_t\rvert}{H(t)}=0$.	DMZDT3: $$\begin{cases} Minimize: \mathbf{f}(\mathbf{x},t)=(f_1(\mathbf{x}_\mathsf{I}),g(\mathbf{x}_\mathsf{II},t)\cdot \\ \quad h(f_1(\mathbf{x}_\mathsf{I}),g(\mathbf{x}_\mathsf{II},t))) \\ f_1(\mathbf{x}_\mathsf{I})=x_1 \\ g(\mathbf{x}_\mathsf{II},t)=1+\frac{9\sum_{x_i\in\mathbf{x}_\mathsf{II}}\lvert y_i(t)\rvert}{D-1} \\ h(f_1,g)=1-\sqrt{\frac{f_1}{g}}-\frac{f_1}{g}\sin(10\pi f_1) \\ where: \\ y_i(t)=\frac{\lvert x_i-t/n_t\rvert}{H(t)},\ \forall i=2,\dots,D \\ H(t)=max\{\lvert 1-\frac{t}{n_t}\rvert,\lvert -1-\frac{t}{n_t}\rvert\} \\ t=\left\lfloor\frac{f_c}{FES_c}\right\rfloor \\ \mathbf{x}_\mathsf{I}\in[0,1];\ \mathbf{x}_\mathsf{II}=(x_2,\dots,x_n)\in[-1,1]^{n-1} \end{cases} \quad (3.64)$$ DMZDT4: $$\begin{cases} Minimize: \mathbf{f}(\mathbf{x},t)=(f_1(\mathbf{x}_\mathsf{I}),g(\mathbf{x}_\mathsf{II},t)\cdot \\ \quad h(f_1(\mathbf{x}_\mathsf{I}),g(\mathbf{x}_\mathsf{II},t))) \\ f_1(\mathbf{x}_\mathsf{I})=x_1 \\ g(\mathbf{x}_\mathsf{II},t)=10D-9+ \\ \quad \sum_{x_i\in\mathbf{x}_\mathsf{II}}\left[y_i(t)^2-10\cos(4\pi\lvert y_i\rvert)\right] \\ h(f_1,g)=1-\sqrt{\frac{f_1}{g}} \\ where: \\ y_i(t)=\frac{\lvert x_i-t/n_t\rvert}{H(t)},\ \forall i=2,\dots,D \\ H(t)=max\{\lvert 1-\frac{t}{n_t}\rvert,\lvert -1-\frac{t}{n_t}\rvert\} \\ t=\left\lfloor\frac{f_c}{FES_c}\right\rfloor \\ \mathbf{x}_\mathsf{I}\in[0,1];\ \mathbf{x}_\mathsf{II}=(x_2,\dots,x_n)\in[-1,1]^{n-1} \end{cases} \quad (3.65)$$
2009 2010	Wang and Li [155] Wang and Li [156]	Type II DMOOP, WYL, where an objective changes over time.	WYL: $$\begin{cases} Minimize: \\ \begin{cases} DMZDT1 & \text{if } t\%4=0 \\ DMZDT2 & \text{if } t\%4=1 \\ DMZDT3 & \text{if } t\%4=2 \\ DMZDT4 & \text{if } t\%4=3 \end{cases} \\ where: \\ \%\text{ is the modulus operator} \end{cases} \quad (3.66)$$

Continued on next page

Year	**Authors**	**Other DMOOPs**	**DMOOPs Definition**
2010	Koo *et al.* [100]	Type I DMOOPs DIMP1 and DIMP2, where each decision variable has its own rate of change, except the variable x_1 that controls the spread of solutions.	Equations (3.43) and (3.44)
2010	Liu *et al.* [113]	DMOP3 is a three-objective Type I DMOOP similar to FDA4 of Farina *et al.* The three-objective POF is $f_1^2 + f_2^2 + f_3^2 = 1$ and the POS is $x_i = G(t),\ \forall x_i \in \mathbf{x}_{\text{II}}$.	DMOP3: $\begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}, g(\mathbf{x}_{\text{II}},t)), \\ \quad f_2(\mathbf{x}, g(\mathbf{x}_{\text{II}},t)), f_3(\mathbf{x}, g(\mathbf{x}_{\text{II}},t))) \\ f_1(\mathbf{x},g,t) = (1+g(\mathbf{x}_{\text{II}},t)))\cos(0.5\pi x_1) \\ \quad \cos(0.5\pi x_2) \\ f_2(\mathbf{x},g,t) = (1+g(\mathbf{x}_{\text{II}},t)))\cos(0.5\pi x_1) \\ \quad \sin(0.5\pi x_2) \\ f_3(\mathbf{x},g,t) = (1+g(\mathbf{x}_{\text{II}},t)))\sin(0.5\pi x_2) \\ where: \\ g(\mathbf{x}_{\text{II}},t) = \sum_{x_i \in \mathbf{x}_{II}} (x_i - G(t))^2 \\ G(t) = \lvert sin(0.5\pi t)\rvert \\ t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ \mathbf{x}_{\text{II}} = (x_3, \dots, x_n) \\ x_i \in [0,1],\ \forall i = 1, \dots, n \end{cases}$ (3.67)
2011	Huang *et al.* [84]	Type IV DMOOPs where the current found POS affects the future POS or POF, a Type IV DMOOP where the number of decision variables change over time and a Type II DMOOP where the number of objective functions change over time.	Equations (3.48) to (3.51)
2011	Helbig and Engelbrecht [78]	Type III DMOOPs HE1 and HE2 with a discontinuous POF and based on the ZDT3 [169] MOOP.	Equations (3.45) and (3.46)

3.2.2 Dynamic Multi-objective Optimisation Problems with an Isolated Pareto Optimal Front

Objective functions contain flat regions when an open subset of decision variable values maps to a single objective function value. The POF of DMOOPs with objective functions that have flat regions are also referred to as an isolated POF. The lack of gradient information for the flat regions may cause difficulty for a DMOO algorithm to converge to the POF. However, no DMOOPs with an isolated POF have been proposed. Therefore, this section proposes an approach that can be used to develop DMOOPs with an isolated POF.

Huband *et al.* introduced a suite of static MOOPs referred to as the WFG benchmark functions to address shortcomings of other MOO test suites [85]. One of the shortcomings that the WFG suite addresses, is the development of MOOPs where the objective functions have flat regions. This approach is adapted so that it can be applied to current DMOOPs.

The flat regions are created by mapping the decision variables to new values using the following equation [85]:

$$y_i(x_i, A, B, C) = A + \min(0, \lfloor x_i - B \rfloor)\frac{A(B - x_i)}{B} - \min(0, \lfloor C - y \rfloor)\frac{(1 - A)(x_i - C)}{1 - C} \quad (3.68)$$

where $A, B, C \in [0, 1], B < C,\ B = 0 \implies A = 0 \wedge C \neq 0,\ C = 1 \implies A = 1 \wedge B \neq 0$. All values of x_i between B and C are mapped to the value of A. Therefore, the region between B and C forms the flat region.

This mapping can be applied to existing DMOOPs. Two examples are provided below, namely the adjustment of FDA5 (refer to Equation (3.24)) and dMOP2 (refer to Equation (3.39)):

$$\text{FDA5}_{\text{iso}} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}, g(\mathbf{x_{II}},t)), \dots, f_k(\mathbf{x}, g(\mathbf{x_{II}},t))) \\ f_1(\mathbf{x},g,t) = (1+g(\mathbf{x_{II}},t)) \prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right) \\ f_k(\mathbf{x},g,t) = (1+g(\mathbf{x_{II}},t)) \left(\prod_{i=1}^{M-1} \cos\left(\frac{y_i\pi}{2}\right)\right) \\ \sin\left(\frac{y_{M-k+1}\pi}{2}\right), \forall k = 1,\dots,M-1 \\ \vdots \\ f_m(\mathbf{x},g,t) = (1+g(\mathbf{x_{II}},t)) \prod_{i=1}^{M-1} \sin\left(\frac{y_1\pi}{2}\right) \\ where: \\ g(\mathbf{x_{II}},t) = \sum_{x_j\in\mathbf{x_{II}}} (y_j - G(t))^2 \\ G(t) = |sin(0.5\pi t)|, \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ y_i = x_i^{F(t)}, \;\; \forall i = 1,\dots,(M-1) \\ y_j = y_j(x_j, A, B, C), \;\; \forall x_j \in \mathbf{x_{II}} \\ F(t) = 1 + 100\sin^4(0.5\pi t) \\ \mathbf{x_{II}} = (x_M,\dots,x_n), \;\; x_i\in[0,1], \; \forall i = 1,\dots,n \end{cases} \tag{3.69}$$

where y_j is calculated using Equation (3.68). A, B and C can, for example, be selected as $G(t)$, 0.001 and 0.05 respectively. Similar to FDA5 (refer to Equation (3.24)), FDA5$_{iso}$ is a Type II DMOOP and the POF of FDA5$_{iso}$ is $f_1^2 + f_2^2 + f_3^2 = (1+G(t))^2$ (as illustrated in Figure 3.4). The POS of FDA5$_{iso}$ is $x_i = G(t)$, $\forall x_i \in \mathbf{x_{II}}$, similar to FDA1 (refer to Figure 3.1(b)).

$$\text{dMOP2}_{\text{iso}} = \begin{cases} Minimize: f(\mathbf{x},t) = (f_1(\mathbf{x_I}), g(\mathbf{x_{II}},t)\cdot h(f_1(\mathbf{x_I}), g(\mathbf{x_{II}},t),t)) \\ f_1(\mathbf{x_I}) = x_1 \\ g(\mathbf{x_{II}},t) = 1 + 9\sum_{x_i\in\mathbf{x_{II}}}(y_i - G(t))^2 \\ h(f_1,g,t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ y_i = y_i(x_i,A,B,C), \;\; \forall x_i \in \mathbf{x_{II}} \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \\ G(t) = \sin(0.5\pi t), \;\; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ x_i \in [0,1]; \;\; \mathbf{x_I} = (x_1), \;\; \mathbf{x_{II}} = (x_2,\dots,x_n) \end{cases} \tag{3.70}$$

where y_i is calculated using Equation (3.68). Example values for A, B and C are $G(t)$, 0.001 and 0.05 respectively. Similar to dMOP2 (refer to Equation (3.39)), dMOP2$_{iso}$ is a Type II problem, with $POF = 1 - f_1^{H(t)}$ (refer to Figure 3.5). The POS of dMOP2$_{iso}$ is $x_i = G(t)$, $\forall x_i \in \mathbf{x_{II}}$, similar to FDA1 (refer to Figure 3.1(b)).

The next section discusses an approach that can be used to develop DMOOPs with a deceptive POF.

3.2.3 Dynamic Multi-objective Optimisation Problems with a Deceptive Pareto Optimal Front

DMOOPs with a deceptive POF have at least two optima, but the search space favours the deceptive POF, which is a local POF and not the global POF. Some of the benchmark functions discussed in Section 3.2.1 are multi-modal. However, none of the benchmark functions discussed in Section 3.2.1 has a deceptive optimum. This section proposes an approach that can be used to adjust existing DMOOPs in such a way that the DMOOPs have a deceptive POF.

The WFG suite of Huband *et al.* [85] also introduced an approach to develop MOOPs with a deceptive POF. Similar to their approach to develop MOOPs with isolated POFs, a transformation function is used as follows:

$$y_i(x_i, A, B, C) = \left(\frac{\lfloor y - A + B \rfloor \left(1 - C + \frac{A-B}{B}\right)}{A - B} + \frac{1}{B} + \right.$$
$$\left. \frac{\lfloor A + B - y \rfloor \left(1 - C + \frac{1-A-B}{B}\right)}{1 - A - B} \right) (|y - A| - B) + 1 \tag{3.71}$$

where $A \in (0,1)$, $0 < B << 1$, $0 < C << 1$, $A - B > 0$ and $A + B < 1$. A represents the value at which x_i is mapped to zero and therefore the global minimum of the transformation function. B is the “aperture” size of the basin leading to A and C is the value of the deceptive optimum.

By applying this transformation (or mapping) function to existing DMOOPs, DMOOPs with a deceptive POF can be developed. For example, by calculating y_j in Equation (3.69) and y_i in Equation (3.70) using Equation (3.71), FDA5$_{iso}$ and dMOP2$_{iso}$ will have deceptive POFs. Example values for A, B and C are 0.35, 0.001 and 0.05 respectively.

Li and Zhang [106] identified a shortcoming of MOO benchmark functions, namely that the POS is defined by a simple function, e.g. $x_i = sin(0.5\pi t)$. Therefore, they presented MOOPs that have complicated POSs, where the POS is defined by non-linear curves in decision space, e.g. $x_j = \sin\left(6\pi x_1 + \frac{j\pi}{n}\right)$, $\forall j = 2, 3, \ldots, n$. This shortcoming is also true for benchmark functions that were developed for DMOO. The next section introduces new DMOOPs with complicated POSs, based on the MOOPs of Li and

Zhang [106].

3.2.4 Dynamic Multi-objective Optimisation Problems with Complicated Pareto Optimal Sets

This section proposes new DMOOPs that have been developed based on the MOOPs of Li and Zhang [106]. The benchmark functions are constructed in such a way that the number of decision variables can be scaled easily, the resulting POFs are easily understood, and the DMOOPs hinder an algorithm to converge to the POF by requiring an algorithm to find a POS that are defined by non-linear curves. Therefore, they adhere to the benchmark function characteristics as defined by Deb *et al.*. The DMOOPs are defined as:

$$\mathsf{HE3} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t)\cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - x_1^{0.5\left(1.0+\frac{3(j-2)}{n-2}\right)}\right)^2 \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - x_1^{0.5\left(1.0+\frac{3(j-2)}{n-2}\right)}\right)^2 \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25,\ \ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\ \text{j is odd and } 2 \le j \le n\} \\ J_2 = \{j|\ \text{j is even and } 2 \le j \le n\} \\ x_i \in [0,1] \end{cases} \tag{3.72}$$

The POF changes over time, but the POS remains the same. Therefore, HE3 is a Type III DMOOP. The POS and POF of HE3 are:

$$POS: x_j = x_1^{0.5\left(\frac{3(j-2)}{n-2}\right)},\ \ \forall j = 2, 3, \ldots, n.$$

$$POF: y = (2-\sqrt{x_1})\left[1 - \left(\frac{x_1}{2-\sqrt{x_1}}\right)^{H(t)}\right]$$

The POF and POS of HE3 are illustrated in Figures 3.10 and 3.11 respectively. It is important to note that, unlike most of the other DMOOPs, the POS of HE3 to HE10 are different for each decision variable.

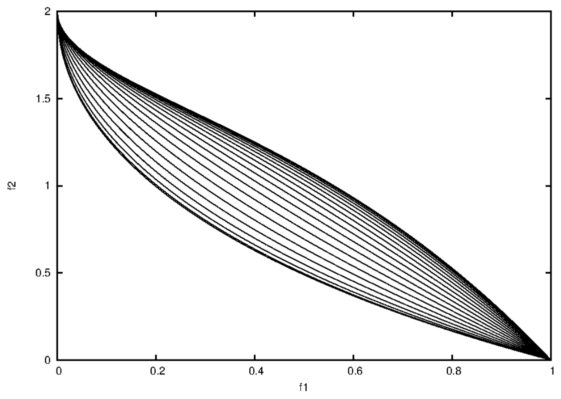
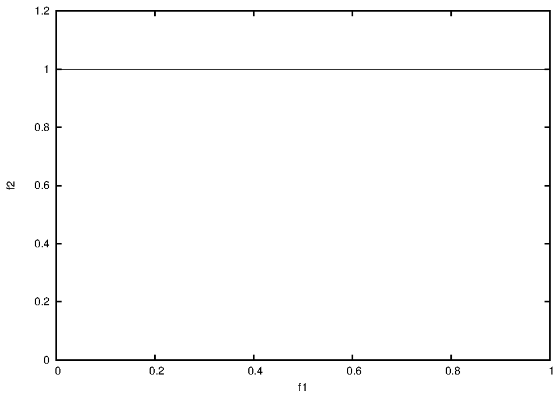
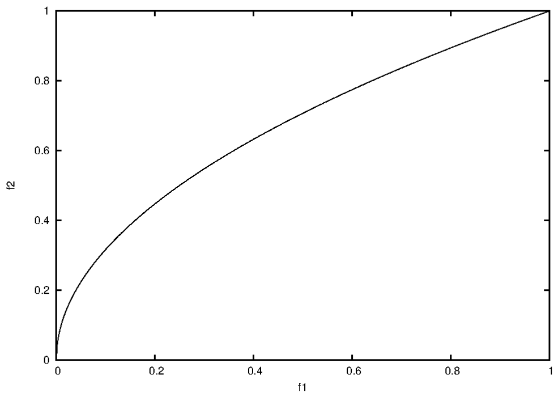


Figure 3.10: POF of HE3 with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations



(a) POS of x_2



(b) POS of x_5

Figure 3.11: POS of HE3 for two decision variables, x_2 and x_5, with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$\mathsf{HE4} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t)\cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25,\ \ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\text{ j is odd and } 2\le j\le n\} \\ J_2 = \{j|\text{ j is even and } 2\le j\le n\} \\ x_1\in[0,1],\ \ x_i\in[-1,1],\ \ \forall i = 2,3,\ldots,n \end{cases} \quad (3.73)$$

The POF of HE4 changes over time, but the POS remains the same. Therefore, HE4 is a Type III DMOOP. The POS and POF of HE4 are:

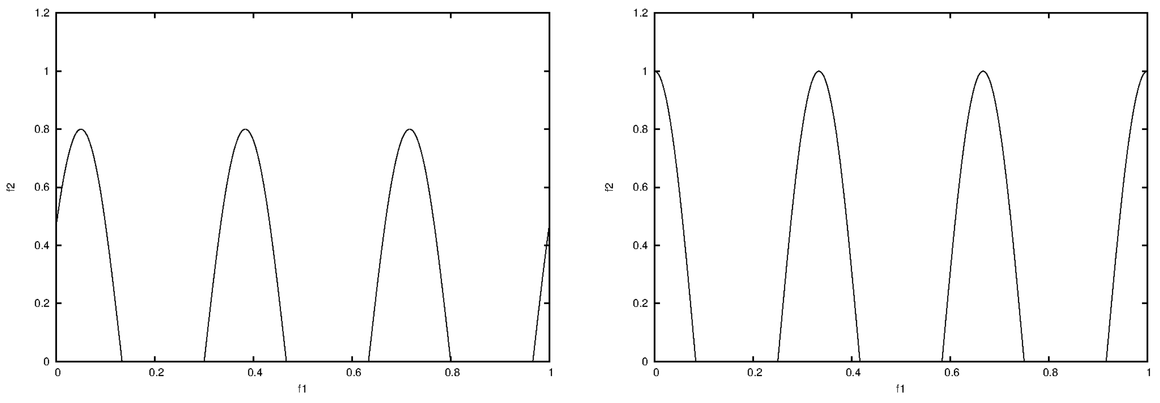
$$POS: x_j = \sin\left(6\pi x_1 + \frac{j\pi}{n}\right),\ \ \forall j = 2,3,\ldots,n.$$

$$POF: y = (2-\sqrt{x_1})\left[1 - \left(\frac{x_1}{2-\sqrt{x_1}}\right)^{H(t)}\right]$$

The POS of HE4 is illustrated in Figure 3.12. The POF is similar to the POF of HE3 (refer to Figure 3.10).

$$\mathsf{HE5} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t)\cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - 0.8x_1\cos\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2 \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - 0.8\cos\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2 \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25,\ \ t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\text{ j is odd and } 2\le j\le n\} \\ J_2 = \{j|\text{ j is even and } 2\le j\le n\} \\ x_1\in[0,1],\ \ x_i\in[-1,1],\ \ \forall i = 2,3,\ldots,n \end{cases} \quad (3.74)$$

HE5 is a Type III DMOOP, since the POF changes over time, but the POS remains the same. The POS and POF of HE5 are:



(a) POS of x_2 (b) POS of x_5

Figure 3.12: POS of HE4 for two decision variables, x_2 and x_5, with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

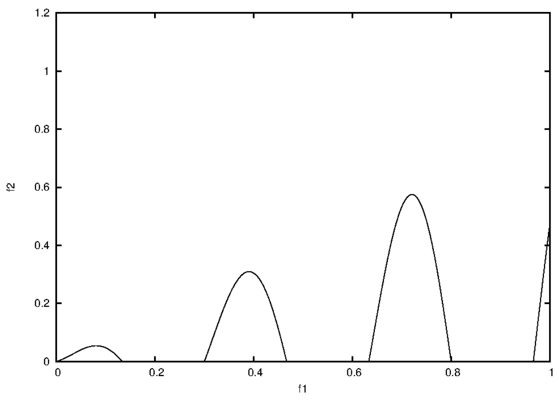
$$POS : x_j = \begin{cases} 0.8x_1 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_1 \\ 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \end{cases}$$

$$POF : y = (2 - \sqrt{x_1})\left[1 - \left(\frac{x_1}{2 - \sqrt{x_1}}\right)^{H(t)}\right]$$

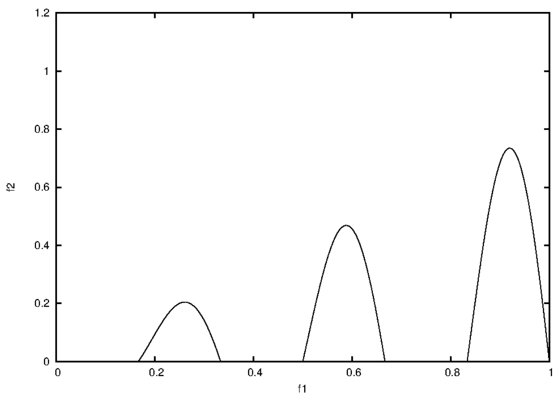
The POS of HE5 is illustrated in Figure 3.13. The POF is similar to the POF of HE3, illustrated in Figure 3.10.

$$\mathsf{HE6} = \begin{cases} Minimize : \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t) \cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j \in J_1}\left(x_j - 0.8x_1 \cos\left(\frac{6\pi x_1 + \frac{j\pi}{n}}{3}\right)\right)^2 \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j \in J_2}\left(x_j - 0.8 \cos\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2 \\ h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where : \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \;\; t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ J_1 = \{j | \text{ j is odd and } 2 \leq j \leq n\} \\ J_2 = \{j | \text{ j is even and } 2 \leq j \leq n\} \\ x_1 \in [0,1], \;\; x_i \in [-1,1], \;\; \forall i = 2,3,\ldots,n \end{cases} \tag{3.75}$$

For HE6, the POF changes over time, but the POS remains the same. Therefore, HE6



(a) POS of x_2



(b) POS of x_5

Figure 3.13: POS of HE5 for two decision variables, x_2 and x_5, with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

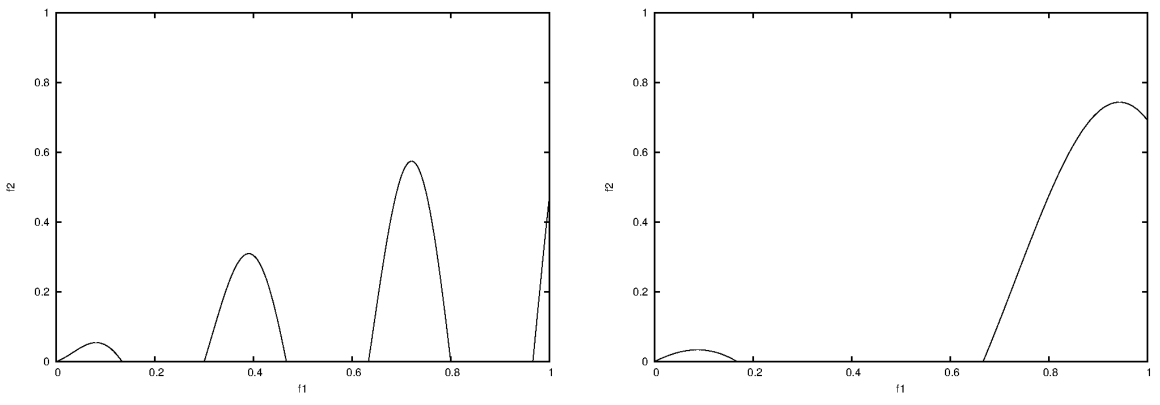
is a Type III DMOOP. The POS and POF of HE6 are:

$$POS : x_j = \begin{cases} 0.8x_1 \cos\left(\frac{6\pi x_1 + \frac{j\pi}{n}}{3}\right), & j \in J_1 \\ 0.8x_1 \sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \end{cases}$$

$$POF : y = (2 - \sqrt{x_1})\left[1 - \left(\frac{x_1}{2 - \sqrt{x_1}}\right)^{H(t)}\right]$$

The POF of HE6 is similar to the POF of HE3 (refer to Figure 3.10). The POS of HE6 is illustrated in Figure 3.14.

$$\text{HE7} = \begin{cases} Minimize : \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t) \cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \left[0.3x_1^2\cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right]\cos\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2 \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \left[0.3x_1^2\cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right]\sin\left(6\pi x_1 + \frac{j\pi}{n}\right)\right)^2 \\ where : \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \;\; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\text{ j is odd and } 2 \le j \le n\} \\ J_2 = \{j|\text{ j is even and } 2 \le j \le n\} \\ x_1 \in [0,1], \;\; x_i \in [-1,1], \;\; \forall i = 2,3,\ldots,n \end{cases} \tag{3.76}$$



(a) POS of x_2 (b) POS of x_5

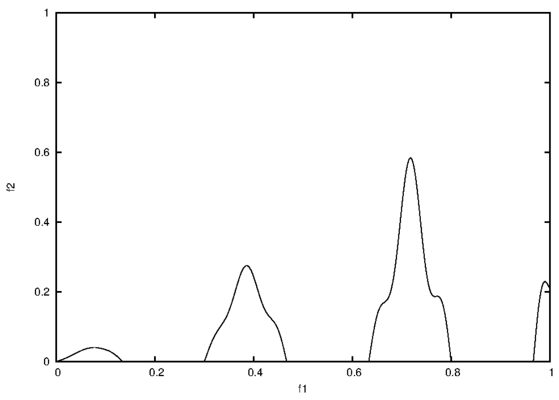
Figure 3.14: POS of HE6 for two decision variables, x_2 and x_5, with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

HE7 is a Type III DMOOP, since the POF changes over time, but the POS remains the same. The POS and POF of HE7 are:

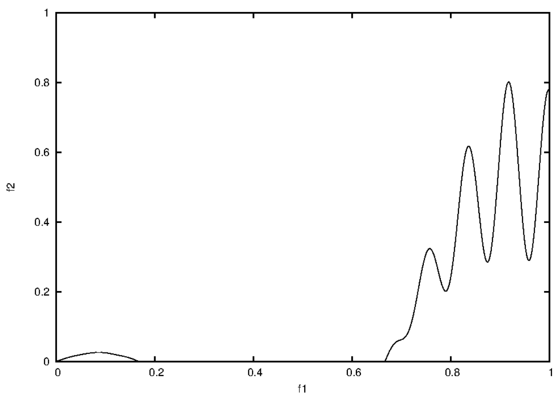
$$POS : x_j = \begin{cases} a\cos\left(\frac{6\pi x_1 + \frac{j\pi}{n}}{3}\right), & j \in J_1 \\ a\sin\left(6\pi x_1 + \frac{j\pi}{n}\right), & j \in J_2 \\ \text{with:} & \\ a = \left[0.3x_1^2\cos\left(24\pi x_1 + \frac{4j\pi}{n}\right) + 0.6x_1\right] & \end{cases}$$

$$POF : y = (2 - \sqrt{x_1})\left[1 - \left(\frac{x_1}{2 - \sqrt{x_1}}\right)^{H(t)}\right]$$

The POS of HE7 is illustrated in Figure 3.15. The POF is similar to the POF of HE3, as illustrated in Figure 3.10.



(a) POS of x_2



(b) POS of x_5

Figure 3.15: POS of HE7 for two decision variables, x_2 and x_5, with $n_t = 10$ and $\tau_t = 10$ for 1000 iterations

$$\text{HE8} = \begin{cases} Minimize: \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t) \cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(4y_j^2 - \cos(8y_i\pi) + 1.0\right) \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}\left(4y_j^2 - \cos(8y_i\pi) + 1.0\right) \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \;\; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\text{ j is odd and } 2 \le j \le n\} \\ J_2 = \{j|\text{ j is even and } 2 \le j \le n\} \\ y_j = x_j - x_1^{\left(0.5\left(1.0 + \frac{3(j-2)}{n-2}\right)\right)}, \;\; \forall j = 2,3,\ldots,n \\ x_i \in [0,1] \;\; \forall i = 1,2,\ldots,n \end{cases} \tag{3.77}$$

The POF of HE8 changes over time, but the POS remains the same. Therefore, HE8 is a Type III DMOOP. The POS (refer to Figure 3.11) and POF (refer to Figure 3.10) of HE8 are:

$$POS: x_j = x_1^{0.5\left(\frac{3(j-2)}{n-2}\right)}, \;\; \forall j = 2,3,\ldots,n.$$

$$POF: y = (2 - \sqrt{x_1})\left[1 - \left(\frac{x_1}{2-\sqrt{x_1}}\right)^{H(t)}\right]$$

$$\text{HE9} = \begin{cases} Minimize : \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t) \cdot h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}(4\sum_{j\in J_1} y_j^2 - \prod_{j\in J_1}\cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 2.0) \\ g(\mathbf{x}) = 2 - \sqrt{x_1} + \frac{2}{|J_2|}\sum_{j\in J_2}(4\sum_{j\in J_2} y_j^2 - 2\prod_{j\in J_2}\cos\left(\frac{20y_j\pi}{\sqrt{j}}\right) + 2.0) \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \;\; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\text{ j is odd and } 2 \le j \le n\} \\ J_2 = \{j|\text{ j is even and } 2 \le j \le n\} \\ y_j = x_j - x_1^{\left(0.5\left(1.0+\frac{3(j-2)}{n-2}\right)\right)}, \;\; \forall j = 2,3,\dots,n \\ x_i \in [0,1] \;\; \forall i = 1,2,\dots,n \end{cases} \tag{3.78}$$

For HE9, the POF changes over time, but the POS remains the same. Therefore, HE9 is a Type III DMOOP. The POS (refer to Figure 3.11) and POF (refer to Figure 3.10) of HE9 are:

$$POS : x_j = x_1^{0.5\left(\frac{3(j-2)}{n-2}\right)}, \;\; \forall j = 2,3,\dots,n.$$
$$POF : y = (2 - \sqrt{x_1})\left[1 - \left(\frac{x_1}{2-\sqrt{x_1}}\right)^{H(t)}\right]$$

$$\text{HE10} = \begin{cases} Minimize : \mathbf{f}(\mathbf{x},t) = (f_1(\mathbf{x}), g(\mathbf{x},t)\cdot \\ \qquad h(f_1(\mathbf{x}), g(\mathbf{x},t))) \\ f_1(\mathbf{x}) = x_1 + \frac{2}{|J_1|}\sum_{j\in J_1}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \\ g(\mathbf{x}) = 2 - x_1^2 + \frac{2}{|J_2|}\sum_{j\in J_2}\left(x_j - \sin(6\pi x_1 + \frac{j\pi}{n})\right)^2 \\ h(f_1,g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where: \\ H(t) = 0.75\sin(0.5\pi t) + 1.25, \;\; t = \frac{1}{n_t}\left\lfloor\frac{\tau}{\tau_t}\right\rfloor \\ J_1 = \{j|\text{ j is odd and } 2 \le j \le n\} \\ J_2 = \{j|\text{ j is even and } 2 \le j \le n\} \\ x_i \in [0,1] \;\; \forall i = 1,2,\dots,n \end{cases} \tag{3.79}$$

The POF of HE10 changes over time, but the POS remains the same. Therefore, HE10 is a Type I DMOOP. The POS (refer to Figure 3.12) and POF (refer to Figure 3.10) of HE10 are:

$$POS : x_j = sin\left(6\pi x_1 + \frac{j\pi}{n}\right), \ \forall j = 2, 3, \ldots, n.$$
$$POF : y = (2 - \sqrt{x_1})\left[1 - \left(\frac{x_1}{2 - \sqrt{x_1}}\right)^{H(t)}\right]$$

Taking into consideration the benchmark functions currently being used for DMOO (discussed in Section 3.2.1) and the ideal characteristics of benchmark functions (discussed in Section 3.1.1), it becomes clear that many different types of DMOOPs have been suggested to be used as benchmark functions. Therefore, when a new DMOO algorithm has been developed, the selection of benchmark functions to test the algorithm's ability to solve DMOOPs is a daunting task.

3.2.5 Ideal Set of Dynamic Multi-objective Optimisation Benchmark Functions

This section presents the characteristics of an ideal benchmark function set and suggests DMOOPs that can be used to sufficiently test an algorithm's ability to solve DMOOPs.

From Section 3.2.1 the following characteristics were identified that an ideal MOO (static or dynamic) set of benchmark functions should have:

1. The set of benchmark functions should test for the following difficulties to converge towards the POF:
   - Multimodality.
   - Deception.
   - Isolated optimum.
2. The set of benchmark functions should test for the following difficulties to obtain a diverse set of solutions:
   - Convexity or non-convexity in the POF.
   - Discontinuous POF, i.e. disconnected sub-regions that are continuous.
   - Non-uniform distribution of solutions in the POF.
3. The benchmark functions should have various types or shapes of POSs, where the POS is also non-linear curves and not only linear functions.

4. The benchmark functions should have decision variables with dependencies or linkages.

In addition, the following characteristics were identified that an ideal DMOOP benchmark function suite should have:

1. The set of benchmark functions should have a non-uniform distribution of solutions in the POF, where the distribution of solutions changes over time.
2. The shape of the POFs should change over time from convex to non-convex or vice versa.
3. The benchmark functions should have decision variables with different rates of change over time.
4. The benchmark functions should include cases where the POF depends on the values of previous POSs or POFs.
5. The benchmark functions should enable changing the number of decision variables over time.
6. The benchmark functions should enbale changing the number of objective functions over time.

For each characteristic a set of DMOOPs was identified from Sections 3.2.1, 3.2.2 and 3.2.3. The proposed ideal benchmark functions suite from which DMOOPs can be selected to evaluate the performance of dynamic MOAs (DMOAs) are presented in Tables 3.7 and 3.8.

When a selection of DMOOPs are made, it should be done in such a way that various types of DMOOPs are selected for each characteristic, or the benchmark suite should at least have type II DMOOPs for some characteristics. The reason for this is to ensure that an algorithm can overcome a certain difficulty in various types of DMOO environments.

In addition to the benchmark functions listed in Table 3.7, the generic benchmark function generators can be used to create benchmark functions of various types with specific characteristics as outlined in this section, for example DTF (refer to Equation (3.25)), DTLZ$_{Av}$ (refer to Equation (3.47)), DSW (refer to Equation (3.52)), and the DMOOP of Tang (refer to Equation (3.36)).

Table 3.7: Set of DMOO Benchmark Functions for each Identified Characteristic for MOOPs in general

Characteristic	**DMOOP Type: Suggested DMOOPs**
1. DMOOPs that cause difficulties to converge towards the POF:	
– Multi-modal DMOOPs	Type I: DMZDT4 (Equation (3.65))
– DMOOPs with a deceptive optimum	Various: DMOOPs developed according to Section 3.2.3
– DMOOPs with an isolated optimum	Various: DMOOPs developed according to Section 3.2.2
2. DMOOPs that cause difficulties to find a diverse set of solutions:	
– DMOOP with convex POF	• Type I: FDA1 (Equation (3.20)), DMZDT1 (Equation (3.62)) • Type II: Modified FDA3 functions (refer to Table 3.6) • Type III: dMOP1 (Equation (3.38))
– DMOOPs with non-convex POF	• Type I: DMZDT2 (Equation (3.63)), FDA4 (Equation (3.23)), DMOP3 (Equation (3.67)) • Type II: FDA5 (Equation (3.24)) • Type III: Modified FDA5 functions (Equation (3.35)
– DMOOPs with discontinuous POF	• Type I: DMZDT3 (Equation (3.64)) • Type III: HE1 (Equation (3.45)), HE2 (Equation (3.46))
– DMOOPs with non-uniform spread of solutions	• Type I: dMOP3 (Equation (3.40)) • Type II: FDA5 (Equation (3.24)), Modified FDA3 functions (refer to Table 3.6) • Type III: modified FDA5 functions (Equation (3.35)
3. DMOOPs with various types or shapes of POSs	• Type I, II: DTLZ$_{Av}$ (Equation (3.47)) • Type II: ZJZ (Equation (3.41)), DSW2 (Equation (3.54)), DSW3 (Equation (3.55)) • Type III: HE3-HE10 (Equations (3.72) - (3.79)), Modified FDA2 functions (Equations (3.26)- (3.31))
4. DMOOPs with dependencies or linkages between the decision variables	• Type II: ZJZ (Equation (3.41))

Table 3.8: Set of DMOO Benchmark Functions for each Identified Characteristic for DMOOPs

Characteristic	DMOOP
1. DMOOPs where the distribution of solutions changes over time	• Type I: dMOP3 (Equation (3.40)) • Type II: FDA5 (Equation (3.24)), Modified FDA3 functions (refer to Table 3.6) • Type III: modified FDA5 functions (Equation (3.35)
2. DMOOPs where the POF changes from convex to non-convex or vice versa over time	• Type II: dMOP2 (Equation (3.39)), ZJZ (Equation (3.41)) • Type III: dMOP1 (Equation (3.38)), Modified FDA2 functions (Equations (3.26)- (3.31))
3. DMOOPs with decision variables with different rates of change	• Type I: DIMP1 (Equation (3.43)), DIMP2 (Equation (3.44))
4. DMOOPs where the current POF depends on the previous POF	• Type IV: T3 (Equation (3.50)), T4 (Equation (3.51))
5. DMOOPs where the number of decision variables change over time	• Type IV: T1 (Equation (3.48))
6. DMOOPs where the number of objective functions change over time	• Type I, II: DTLZ$_{Av}$ (Equation (3.47)) • Type III: T2 (Equation (3.49))

3.3 Summary

This chapter provided an overview of the benchmark functions that have been used to test whether DMOO algorithms can overcome specific difficulties that can occur in real-world problems. MOO benchmark functions that have been adapted for DMOO, namely the ZDT and DTLZ MOOPs, were discussed. Since there is no standard DMOO benchmark functions yet, this chapter discussed the characteristics that an ideal benchmark function suite should have and suggested benchmark functions that can be used to test for each of these recommended characteristics.

The next chapter provides an overview of performance measures suggested for DMOO.

Chapter 4

Analysis of Dynamic Multi-objective Optimisation Performance Measures

"You get what you measure. Measure the wrong thing and you get the wrong behaviors." – John H. Lingle

In order to determine whether an algorithm can solve DMOOPs efficiently, it should be tested against DMOOPs that test the ability of the algorithm to overcome certain difficulties, called benchmark functions. However, to quantify the performance of the algorithm, and to compare the performance of one algorithm against that of another algorithm, performance measures are required.

One of the main problems in the field of DMOO is a lack of standard benchmark functions and standard performance measures. An analysis of benchmark functions for DMOO was presented in Chapter 3. This chapter evaluates the current performance measures presented in the DMOO literature to establish whether they efficiently evaluate the performance of DMOO algorithms.

Section 4.1 discusses performance measures that have been used for MOO and adapted for DMOO. Performance measures currently used in the DMOO literature are discussed in Section 4.2. Section 4.3 highlights issues with current performance measures that are frequently used to measure the performance of DMOO algorithms. Finally, a summary is provided in Section 4.4.

4.1 Static Multi-objective Optimisation Performance Measures

Performance measures enable the quantification and measuring of an algorithm's performance with regards to a specific requirement, such as the number of non-dominated solutions found, closeness to the true POF (accuracy), and the diversity or spread of the solutions. According to Zitzler *et al.* [173], a performance measure is defined as follows:

Definition 4.1. Performance Measure: A m-ary performance measure, P, is a function $P : \Omega^m \to \mathbb{R}$, that assigns each of the m approximated POFs, $POF_1^*, POF_2^*, \ldots,$ POF_m^* a real value $P(POF_1^*, POF_2^*, ..., POF_m^*)$.

This section discusses static MOO measures that have been adapted in the literature and used in DMOO. The discussion on static MOO performance measures is by no means complete, and the reader is referred to [40, 99, 114, 167] for detailed information on performance measures used for static MOO.

Outperformance relations that are used to evaluate performance measures are discussed in Section 4.1.1. Section 4.1.2 discusses performance measures that quantify an algorithm's performance with regards to accuracy, i.e. the found non-dominated solutions' (POF^*) closeness to the true POF (POF). Performance measures that measure the diversity or spread of the found solutions are discussed in Section 4.1.3. Section 4.1.4 discusses performance measures that measure the overall quality of the found solutions, taking into account both accuracy and diversity.

4.1.1 Outperformance Relations

This section discusses outperformance relations that performance measures should ideally adhere to. When an algorithm solves a MOOP where the objective functions are in conflict with one another, the algorithm tries to find the best possible set of non-dominated solutions, i.e. a set of solutions that are as close as possible to POF and where the solutions are diverse and evenly spread along POF^*. Once POF^* is found, a decision maker selects one of these solutions according to his/her own defined preferences.

Hansen and Jaszkiewicz [75] introduced an outperformance relation under the following assumptions:

- The preferences of the decision maker is not known a priori.
- Let POF_A^* and POF_B^* be two approximated POFs. Then, POF_A^* outperforms POF_B^* if the decision maker finds:
  - a better solution in POF_A^* than in POF_B^* for specific preferences, and
  - for another set of preferences the solution selected from POF_A^* is not worse than solutions found in POF_B^*.
- All possible preferences of the decision maker can be modelled with utility functions that belong to a set of utility functions, U.

Definition 4.2. Outperformance Relation (subject to a set of utility functions): Let A and B be two sets representing approximations of the same POFs. Let $U|A > B \subseteq U$ denote a subset of utility functions for which A is better than B, i.e. $U|A > B = \{u^* \in U|u^*(A) > u^*(B)\}$. Then A outperforms (O) B if there exists a non-empty subset of the utility functions set U for which A achieves better values than B, while the opposite is not true. Mathematically the outperformance relation is defined as: $A\ O_U\ B$ if $U(A > B) \neq \varnothing$ and $U(B > A) = \varnothing$.

The weakest assumption about the decision maker's preferences that is generally made when solving MOOPs is that the utility function is compatible with the dominance relation, i.e. the decision maker prefers non-dominated solutions [128]. Therefore, the decision maker can limit his/her selection of the best solution to the set $ND(A \cup B)$, i.e. the non-dominated solutions in $A \cup B$. Based on the dominance relation assumption, the following three dominance based relations were defined by Hansen and Jaszkiewicz [75]:

Definition 4.3. Weak Outperformance: A weakly outperforms (O_W) B if, for each solution in B, there exists a solution in A that is equal to or dominates the solution in B and at least one solution in A is not contained in B. Weak outperformance is defined as: $A\, O_W\, B$ if $A \neq B$ and $ND(A \cup B) = A$, where $ND(A \cup B)$ is the set of non-dominated solutions obtained from the unified set, $(A \cup B)$.

Definition 4.4. Strong Outperformance: A strongly outperforms (O_S) B if, for each solution in B, there exists a solution in A that is equal to or dominates the solution in B and at least one solution in B is dominated by a solution in A. Strong outperformance is mathematically defined as: $A\,O_S\,B \quad \text{if} \quad A \neq B, \quad ND(A \cup B) = A \quad \text{and} \quad B \backslash ND(A \cup B) = \varnothing$.

Definition 4.5. Complete Outperformance: A completely outperforms B if each solution in B is dominated by a solution in A. Complete outperformance is defined as: $A\,O_C\,B \quad \text{if} \quad A \neq B, \quad ND(A \cup B) = A \quad \text{and} \quad B \cap ND(A \cup B) = \varnothing$.

These outperformance relations only identify whether one set is better than another set, but doesn't quantify with how much the one set is better than the other. However, according to Knowles [99], performance measures that are not compatible with these outperformance relations, cannot be relied on to provide evaluations that are compatible with Pareto dominance. Based on the outperformance relations, Hansen and Jaszkiewicz [75] defined compatibility and weak compatibility with an outperformance relation:

Definition 4.6. Weak Compatibility: A performance measure is weakly compatible with an outperformance relation if, for each pair of non-dominated sets, A and B, where $A\,O\,B$, the performance measure will evaluate A as being not worse than B.

Definition 4.7. Compatibility: A performance measure is compatible with an outperformance relation if for each pair of non-dominated sets, A and B, where $A\,O\,B$, the performance measure will evaluate A as being better than B.

In addition to the outperformance relations, Knowles [99] introduced the concepts of monotony and relativity that are important when evaluating the efficiency of performance measures.

Definition 4.8. Monotony: Let C be a set containing a new non-dominated solution. Then the performance measure will evaluate $A \bigcup C$ as being not worse than A.

Definition 4.9. Relativity: Let D be a set containing the solutions of the true POF. Then the performance measure will evaluate D as being better than any POF^* found by the algorithms being evaluated.

Weak compability with O_W is sufficient for weak monotony and weak relativity [99].

From the above definitions it should be clear that $O_C \subset O_S \subset O_W$, i.e. complete outperformance is the strongest outperformance relation, and weak outperformance is the weakest outperformance relation. Therefore, it is most difficult for a performance measure to be compatible with O_W, and the easiest for a performance measure to be compatible with O_C [99]. According to Knowles [99], performance measures that are not compatible with these outperformance relations, cannot be relied on to provide evaluations that are compatible with Pareto dominance.

If a performance measure is compatible with the concept of monotony, it will not decrease a set's evaluation if a new non-dominated point is added, which adheres to the goal of finding a diverse set of solutions. Furthermore, if a performance measure does not adhere to the concept of relativity, it will evaluate an approximation set as being better than the true POF, which is not accurate.

Knowles [99] evaluated the performance measures frequently used in MOO according to their compatibility with the outperformance relations defined by Hansen and Jaszkiewicz. The MOO performance measures' compatability with the outperformance relations are highlighted below where the performance measures are discussed in more detail.

4.1.2 Accuracy Performance Measures

This section discusses performance measures that are used to measure the accuracy of POF^* that is found by a MOO algorithm, i.e. how close POF^* is to POF.

Generational Distance

The generational distance (GD) measures the convergence of the approximated set towards the true POF (POF). The GD is defined as:

$$GD = \frac{\sqrt{\sum_{i=1}^{n_{POF^*}} d_i^2}}{n_{POF^*}} \tag{4.1}$$

where n_{POF^*} is the number of solutions in POF^* and d_i is the Euclidean distance in the objective space between solution i of POF^* and the nearest member of POF'. POF' contains sampled solutions of POF that are used as a reference set. Therefore, GD determines how close POF^* is to the sampled solutions of POF.

GD is easy to calculate and intuitive. However, knowledge about POF is required and a reference set, POF', has to be available. It is important that the reference set contains a diverse set of solutions, since the selection of the solutions will impact on the results obtained from this performance measure. Furthermore, since the distance metric is used, scaling and normalisation of the objectives is required. GD is not weakly compatible with O_W, but is compatible with O_S. Unfortunately, this performance measure will rate a POF^* with only one solution that is on POF' better than another POF^* that has one hundred solutions that are very close to POF'. Therefore, GD does not adhere to the property of monotony. Furthermore, GD does not adhere to the concept of weak relativity, because any subset of POF' will not necessarily have the best GD value when compared to POF^*s found by MOO algorithms.

It should be noted that GD is computationally expensive, especially for large or unlimited archives or when DMOOPs with a large number of objectives are used.

Inverted Generational Distance

To overcome non-adherence to the concept of monotony by GD, Sierra and Coello Coello [137] introduced the inverse generational distance (IGD). The mathematical definition of IGD is the same as GD in Equation (4.1), except for the way in which the distance is calculated:

$$IGD = \frac{\sqrt{\sum_{i=1}^{n_{POF'}} d_i^2}}{n_{POF'}} \tag{4.2}$$

where $n_{POF'}$ is the number of solutions in POF' and d_i is the Euclidean distance in the objective space between solution i of POF' and the nearest member of POF^*.

IGD is compatible with relativity, since POF' obtains an IGD value of zero and POF^* will only receive an IGD value of zero if $POF^* = POF'$. Furthermore, IGD is compatible with monotony, because it will rate a POF^* with more non-dominated solutions that are close to POF as a better set than another POF^* that only has one solution that falls within POF'. However, IGD is computationally expensive to calculate for a larger POF' or a larger POF^*, since for each solution in POF', the distance between that solution and each of the solutions in POF^* has to be calculated. The usage of the distance function also requires scaling and normalisation of the objective function values, as is the case with GD.

Error Ratio

Van Veldhuizen [151] introduced the error ratio that measures the ratio of non-dominated solutions in POF^* that are elements of POF' to the non-dominated solutions in POF^* that are not elements of POF'. The error ratio is defined as

$$E = \frac{\sum_{i=1}^{n_{POF^*}} e_i}{n_{POF^*}} \tag{4.3}$$

where $e_i = 0$ if $x_i \in POF'$, $\forall x_i \in POF^*$ and $e_i = 1$ if $x_i \notin POF'$, $\forall x_i \in POF^*$. A small error ratio indicates a good performance.

If POF_A^* has two solutions with one solution in POF', $E = 0.5$. However, if POF_B^* has one hundred solutions with one solution in POF' and the other solutions very close to POF', $E = 0.99$. According to E, POF_A^* is a better set of solutions than POF_B^*. However, POF_B^* is more desirable. Therefore, E is only weakly compatible with O_C. E has weak relativity, because any subset of POF' will achieve the lowest E value, namely $E = 0$. It is not compatible with monotony, because if a non-dominated solution is added to POF^* that is not an element of POF', it will increase E.

The compatibility of the accuracy performance measures with the outperformance relations and the concepts of monotony and relativity is summarised in Table 4.1. In Tables 4.1 to 4.3 and Tables 4.4 to 4.11, M and R refer to the concepts of monotony and relativity respectively, C and W indicate that the performance measure is compatible or weakly compatible with the relation respectively and "–" indicates that the performance measure is neither compatible nor weakly compatible with the relation.

Table 4.1: Compatibility of accuracy performance measures

Performance Measure	$\mathbf{O_W}$	$\mathbf{O_S}$	$\mathbf{O_C}$	**M**	**R**
GD	–	C	C	–	–
IGD	W	C	C	W	C
E	–	–	W	–	W

4.1.3 Diversity Performance Measures

This section discusses performance measures that are used to measure the diversity of the solutions contained in POF^*. Diversity can be measured either by measuring how evenly the solutions are spread along POF^* or the extent of POF^*.

Number of Solutions

The easiest performance measure to calculate is the number of non-dominated solutions (NS) in POF^*. Van Veldhuizen [151] referred to this metric as the overall nondominated vector generation (ONVG). Even though this measure does not provide any information with regards to the quality of the solutions, it provides additional information when comparing the performance of various algorithms. For example, one algorithm may have a better GD value, but only half of the NS that have been found by the other algorithm.

NS is not weakly compatible with any of the outperformance relations. According to Knowles [99], weak compatability with O_W is necessary to ensure weak monotony. However, with NS this is not the case. Adding a non-dominated solution to POF^* increases, and thereby improves, NS. Therefore, NS is compatible with monotony. Furthermore, NS is weakly compatible with relativity only if the size of POF^* is smaller or equal to the size of POF'.

Spacing Metric of Schott

The *Spacing* metric, introduced by Schott [132], measures how evenly the points of POF^*, are distributed in the objective space. Spacing is calculated as:

$$\mathcal{S} = \sqrt{\frac{1}{n_{POF^*} - 1} \sum_{m=1}^{n_{POF^*}} (d_{avg} - d_m)^2}$$

with

$$d_m = min_{j=1,\ldots,n_{POF^*};j\neq i} \left\{ \sum_{k=1}^{n_k} |f_k(\mathbf{x}) - f_{kj}(\mathbf{x})| \right\} \tag{4.4}$$

where d_m is the minimum value of the sum of the absolute difference in objective function values between the m-th solution in POF^* and any other solution in POF^*, d_{avg} is the average of all d_m values and n_k is the number of objective functions. If $\mathcal{S} = 0$, the non-dominated solutions of POF^* is uniformly spread or spaced [40]. However, this does not mean that the solutions are necessarily good, since they can be uniformly spaced in POF^*, but not necessarily uniformly spaced in POF [99, 55].

The spacing metric of Schott is not even weakly compatible with O_W [99]. Adding a non-dominated solution to POF^* will not necessarily decrease the value of S and POF'

does not necessarily have the lowest spacing metric value. Therefore, S does not adhere to the principles of either monotony or relativity.

It should be noted that this performance measure was designed to be used with other performance measures, has a low computational cost, and can provide useful information about the distribution of the found solutions [99]. Since the Euclidean distance is used in the calculation of the measure, the objectives should be normalised before calculating the measure.

Spacing Metric of Deb

S provides information with regards to how evenly the non-dominated solutions are spaced on POF^*. However, it does not provide any information with regards to the extent of spread of the solutions. To address this shortcoming, Deb [42] introduced a measure of spread, defined as:

$$\Delta = \frac{\sum_{k=1}^{n_k} d_k^e + \sum_{i=1}^{n_{POF^*}} |d_i - d_{avg}|}{\sum_{k=1}^{n_k} d_k^e + n_{POF^*} d_{avg}} \tag{4.5}$$

with d_i any distance measure between neighbouring solutions, d_{avg} is the mean of these distance measures and d_k^e is the distance between the extreme solutions of POF^* and POF'.

Similar to S, Δ is not compatible with O_W and does not adhere to monotony or relativity.

Maximum Spread

Zitzler [167] introduced a measure of maximum spread that measures the length of the diagonal of the hyperbox that is created by the extreme function values of the non-dominated set. The maximum spread is defined as:

$$MS = \sqrt{\sum_{k=1}^{n_k} \left(\overline{POF_i^*} - \underline{POF_i^*} \right)^2} \tag{4.6}$$

where $\overline{POF_k^*}$ and $\underline{POF_k^*}$ is the maximum and minimum value of the k-th objective in POF^* respectively. A high MS value indicates a good extend (or spread) of solutions.

This measure can be normalised in the following way [40]:

$$MS_{norm} = \sqrt{\frac{1}{n_k} \sum_{k=1}^{n_k} \left(\frac{\overline{POF_k^*} - \underline{POF_k^*}}{\overline{POF_k} - \underline{POF_k}} \right)^2} \tag{4.7}$$

If POF_A^* outperforms POF_B^* (weakly, strongly or completely), but the non-dominated solutions of POF_B^* have a larger extent than the non-dominated solutions of POF_B^*, then POF_A^* will obtain a higher MS value. Therefore, MS is not weakly compatible with any of the outperformance relations. Adding a non-dominated solution to POF^* will not necessarily lead to a higher MS value. Therefore, MS adheres to weak monotony. POF' will obtain the maximum MS value, but even a POF^* that only has two non-dominated solutions at the extreme points of POF' will also obtain the maximum MS value. Therefore, MS adheres to weak relativity.

C-Metric

The set coverage metric (C-metric) introduced by Zitzler [167] measures the proportion of solutions in set B that are weakly dominated by solutions in set A. The C-metric is defined as:

$$C(A,B) = \frac{|\{b \in B| \, \exists a \in A\colon a \preceq b\}|}{|B|} \tag{4.8}$$

If $C(A,B) = 1$, all solutions in set B are weakly dominated by set A and if $C(A,B) = 0$ no solution in set B is weakly dominated by set A. Let POF_A^* and POF_B^* be the approximated POFs found by two algorithms with $POF_A^* \subset POF_B^*$, $ND(POF_B^*) = POF_B^*$. Then $C(POF_A^*, POF_B^*) < 1$ and $C(POF_B^*, POF_A^*) = 1$. Therefore, POF_B^* outperforms POF_A^*. Under the assumption that, if $C(A,B) = 1$ and $C(B,A) < 1$ evaluates set A as being better than set B, the C-metric is compatable with O_W [99].

It is important to note that the domination operator is not a symmetric operator, i.e. $C(A,B)$ is not necessarily equal to $1 - C(B,A)$. Therefore, if many algorithms are compared against each other, this metric would have to be calculated twice for each possible combination of algorithms. However, it should be noted that the C-metric is cycle-inducing, in other words if more than two sets are compared, the sets may not be ordered and in these cases no conclusions can be made [99].

If POF is known, set A can be selected as the set of sampled points of the true POF, POF', and set B as the POF^* found by the algorithm. Then the C-metric can be calculated separately for each algorithm. Let POF_A^* and POF_B^* be the approximated POFs found by two algorithms as defined above and POF' a reference set with $ND(POF') = POF'$ and $POF_B^* \subseteq POF'$. Then, $C(POF', POF_A^*) = C(POF_B^*) = 1$

and $C(POF_A^*, POF') < C(POF_B^*, POF')$. In order to ensure compatibility with O_W, POF_A^* has to be evaluated by the C-metric as being worse than POF_B^*. Therefore, the following assumption should be made: if $C(R, E) = C(R, E) = 1$ and $C(E, R) < C(D, R)$, then E performs worse than D with regards to the C-metric, where D and E are two sets that are compared with one another using the reference set R [99]. Under this aforementioned assumption, the C-metric is compatible with O_W when a reference set is used.

The C-metric does not adhere to the concept of monotony, since POF^* can add a non-dominated solution that is weakly dominated by the set that POF^* is compared against. However, the C-metric is weakly compatible with relativity, since $C(POF^*, POF')$ cannot obtain a higher C-metric value than $C(POF', POF^*)$.

U-measure

Leung and Wang [103] introduced the U-measure to measure the diversity of the found non-dominated solutions. Let $R = r_k$ be the set of reference points, where r_k is the extreme point of objective k of the union of all non-dominated solution of all POF^*'s found by the algorithms for the same POF that are compared with one another. Let χ be the set $\{d_i\}$ and $\overline{\chi}$ the set of $\{d_j\}$, where d_i is the distance between two neighbouring solutions and d_j is the distance between a reference point, r_k, and its nearest neighbour. Let d_{avg}^* be the average of the distances in χ and let $\overline{\chi}^*$ be the set $\{d_j' | d_j' = d_j + d_{avg}^*\}$. Then, the U-measure is defined as:

$$U = \frac{1}{n_{POF^*}} \sum_{j=1}^{n_{POF^*}} \left| \frac{d_j'}{d_{ideal}} - 1 \right|$$

with

$$d_{ideal} = \sum_{j=1}^{n_{POF^*}} \frac{d_j'}{n_{POF^*}} \tag{4.9}$$

A smaller U-measure value indicates better uniformity of the non-dominated solutions of POF^*. Since distances are calculated in the U-measure, the objectives have to be normalised. Similar to S and Δ, the U-measure is not weakly compatible with any of the outperformance relations and does not adhere to monotony or relativity.

Table 4.2 summarises the compatibility of the diversity performance measures with the outperformance relations and the concepts of monotony and relativity. In Tables 4.2 to 4.3, C* and W* indicate that the performance measure is either compatible or weakly compatible with the relation, but only under certain conditions.

Table 4.2: Compatibility of diversity performance measures

Performance Measure	O_W	O_S	O_C	M	R
NS	–	–	–	C	W*
S	–	–	–	–	–
Δ	–	–	–	–	–
MS	–	–	–	W	W
C	W*	C	C	–	W
U	–	–	–	–	–

4.1.4 Combined Performance Measures

This section discusses performance measures that measure the quality of the solutions of the found POF, by taking into account both the accuracy and diversity of the set of solutions.

Hypervolume

The hypervolume or S-metric (first referred to as "the size of the space covered") measures how much of the objective space is dominated by a non-dominated set [171, 172]. The definition of a dominated region and the traditional definition of the hypervolume are as follows:

Definition 4.10. Dominated Region: Let $\mathbf{f}_1 = \{f_{1_1}, f_{1_2}, \ldots, f_{1_k}\}$ be a solution in the objective space and $\mathbf{f}_{\mathsf{ref}}$ a reference vector dominated by $\mathbf{f}_1$. Then the region that is dominated by $\mathbf{f}_1$ and bounded by $\mathbf{f}_{\mathsf{ref}}$ is defined as the set,

$$R(\mathbf{f}_1, \mathbf{f}_{\mathsf{ref}}) \triangleq \left\{\mathbf{f}_r \,|\, \mathbf{f}_r \prec \mathbf{f}_{\mathsf{ref}} \text{ and } \mathbf{f}_1 \prec \mathbf{f}_r,\ \ \mathbf{f}_r \in \mathbb{R}^K\right\} \tag{4.10}$$

Let A be a non-dominated set of vectors, $\mathbf{f}_{\mathsf{i}}$, for $i = 1, \ldots, |A|$. Then the region dominated by A and bounded by the reference vector, $\mathbf{f}_{\mathsf{ref}}$, is defined as the set:

$$R(A, \mathbf{f}_{\mathsf{ref}}) \triangleq \bigcup_{i=1,\ldots,|A|} R(\mathbf{f}_{\mathsf{i}}, \mathbf{f}_{\mathsf{ref}}) \tag{4.11}$$

Definition 4.11. Hypervolume: The hypervolume (HV) or S-metric of set A with respect to the reference vector $\mathbf{f}_{\mathsf{ref}}$ is the hyperarea or Lebesgue integral of the set $R(A, \mathbf{f}_{\mathsf{ref}})$.

The reference vector can be any vector outside the feasible objective space, since this will result in a non-negative value for all possible non-dominated sets in the feasible objective space. Usually, the reference vector or reference point that is used in the HV calculation is the vector that consists of the worst value for each objective of the union of all non-dominated solutions of all POF^* that are compared against each other. It should be noted that the selected reference vector will affect the ordering of the non-dominated sets that are compared against each other, since all of the non-dominated sets use the same reference vector [99]. A high HV value indicates a good approximation set.

The HV is compatible with O_W if the upper boundary of the dominated region is set in such a way that all feasible non-dominated sets that are evaluated have a positive HV value. The HV is therefore compatible with the outperformance relations. The HV is weakly compatible with monotony and weakly compatible with relativity. It is scaling independent and no prior knowledge of the true POF is required. According to Zitzler *et al.* [168] the HV is the only performance measure in the literature that has the following two qualities:

- If an approximation set A dominates another set B the HV provides a strictly better value for A.
- If a set obtains the maximum possible HV value for a MOOP, it contains all Pareto-optimal objective values.

One flaw of the HV is that it is biased towards convex areas of the POF [168]. Furthermore, it is computationally expensive to calculate, with a computational cost of $O(n^{k+1})$ with k representing the number of objectives [99]. However, recent research developed algorithms that reduce the computational cost of the HV. For example, Fonseca *et al.* proposed an $O(|A| \log |A|)$ algorithm [62] and Beume and Rudolph proposed an algorithm with a complexity of $O(|A|^{k/2})$ [8], where A is the non-dominated set and k is the number of objectives.

Hypervolume Ratio

To overcome the bias of the HV towards convex regions of the POF, Van Veldhuizen [151] proposed the hypervolume ratio (HVR), defined as:

$$HVR = \frac{HV(POF^{*})}{HV(POF)} \tag{4.12}$$

The HVR normalises the HV and, assuming that the maximum HV is obtained by the true POF, the value of the HVR will be between 0 and 1. A high HVR indicates a good approximated POF. It should be noted that, for the HVR calculation, the reference vector is selected as the worst objective values for each objective from the union of the non-dominated solutions of all POF^{*} that are compared against each other, as well as POF'.

Similar to the HV, the HVR is compatible with O_W if the upper boundary of the dominated region is set in such a way that all feasible non-dominated sets that are evaluated have a positive HV value. Therefore, the HVR is compatible with the outperformance relations. Furthermore, the HVR is weakly compatible with monotony and relativity.

ϵ-metric

Zitzler *et al.* [173] presented the ϵ-metric to compare approximated sets. It measures the factor by which an approximation set is worse than another approximation set with respect to all objectives, i.e. it provides the factor ϵ where for any solution in set B there is at least one solution in set A that is not worse by a factor of ϵ in all objectives. The ϵ-measure uses the concept of ϵ-dominance.

Using the definitions of objective vector domination and objective vector ϵ-domination (refer to Section 2.2.2), the ϵ-metric is defined as:

$$I_{\epsilon}(A,B) = \inf_{\epsilon \in \mathbb{R}} \{\forall \mathbf{f}_2 \in B | \exists \mathbf{f}_1 \in A\colon f_1(\mathbf{x}_k) \prec_{\epsilon} f_2(\mathbf{x}_k)\} \tag{4.13}$$

The ϵ-metric is not weakly compatible with O_W, but is compatible with O_S and O_C. The ϵ-metric is not weakly compatible with monotonoy, but is weakly compatible with relativity.

The compatibility of the combined performance measures with the outperformance relations are summarised in Table 4.3.

The next section discusses how these MOO performance measures were adapted for DMOO. Performance measures developed specifically for DMOO are also discussed.

Table 4.3: Compatibility of combined performance measures

Performance Measure	$\mathbf{O_W}$	$\mathbf{O_S}$	$\mathbf{O_C}$	**M**	**R**
HV	C*	C*	C*	W*	W*
HVR	C*	C*	C*	W*	W*
I_ϵ	–	C	C	–	W

4.2 Current Dynamic Multi-objective Optimisation Performance Measures

This section discusses performance measures that are currently being used to evaluate the performance of DMOO algorithms. Section 4.2.1 discusses performance measures that measure the accuracy of the found POF. Performance measures that are used to measure the diversity of the non-dominated solutions are discussed in Section 4.2.2. Section 4.2.3 discusses the measurement of an algorithm's robustness after an environment change occurs. Combined performance measures that measure accuracy and diversity of the non-dominated solutions are discussed in Section 4.2.4.

4.2.1 Accuracy Performance Measures

This section discusses performance measures that are used to measure the accuracy of a POF^*.

GD Measure

Mehnen *et al.* [117] used the GD metric to evaluate the performance of algorithms solving DMOOPs. They calculated the GD metric in decision space, since the DMOOPs that were used in the study had POSs that dynamically shifted over time, and named the performance measure G_τ. If GD is calculated in decision space, GD measures the distance of the approximated POS, POS^*, to the true POS, POS. Zhou *et al.* [166] used the GD metric (and the variance of GD) in objective space for DMOO, but referred to the performance measure as the distance indicator D. A number of other researchers have used GD to evaluate DMOO algorithms, as shown in Table 4.5. Goh and Tan [67] adapted GD for DMOO as follows:

$$VD = \frac{1}{\tau}\sum_{t=1}^{\tau} VD(\tau)$$

with

$$VD(\tau) = \frac{\sqrt{n_{POF^*}\sum_{i=1}^{n_{POF^*}} d_i^2\ (\tau\%\tau_t)}}{n_{POF^*}}$$

where τ is the current iteration number and τ_t is the frequency of change. The performance measure, referred to as variational distance (VD), is calculated in the decision space every iteration just before a change in the environment occurs.

Similar to GD, VD is not weakly compatible with O_W, but is compatible with O_S and O_C. It is not weakly compatible with monotony, but is weakly compatible with relativity. Since distance is used in the calculation of VD, the objectives have to be normalised.

When solving DMOOPs, similar to VD, the performance measure is calculated every iteration just before an environmental change occurs. Therefore, prior knowledge of when changes occur is required. However, if a performance measure is calculated while the algorithm is running (also referred to as online calculation), prior knowledge about changes in the environment is not required. In this case the performance measure can be calculated on the non-dominated solutions that were obtained at the iteration just before the change occurred. Furthermore, if the performance measure is calculated after the algorithm has completed its runs (also referred to as offline calculation), the algorithm can keep record of the iterations when changes occurred.

Success Ratio

Similar to the error ratio (refer to Section 4.1.2), Mehnen *et al.* [117] used the success ratio to quantify the ratio of the found solutions that are members of the true POF. The success ratio is defined as

$$SC_\tau = \frac{|\{\mathbf{x}|f(\mathbf{x}) \in POF'\}|}{n_{POF^*}} \tag{4.14}$$

where a high success ratio, SC$_\tau$, indicates good performance.

If an algorithm finds many non-dominated solutions that are not pareto-optimal but very close to POF', the POF^* will obtain a lower SC$_\tau$ value than an algorithm that finds only one pareto-optimal solution. Therefore, SC$_\tau$ is only weakly compatible with O_C and is not weakly compatible with either O_W or O_S.

If a non-dominated solution is added to POF^* that is not Pareto-optimal, the value of SC$_\tau$ decreases and therefore SC$_\tau$ is not compatible with monotony. Since POF' will obtain the same SC$_\tau$ value than subsets of POF', SC$_\tau$ is weakly compatible with relativity.

The compatibility of the accuracy performance measures with the outperformance relations and the concepts of monotony and relativity is summarised in Table 4.4.

Table 4.4: Compatibility of accuracy performance measures

Performance Measure	**O$_\mathbf{W}$**	**O$_\mathbf{S}$**	**O$_\mathbf{C}$**	**M**	**R**
GD	–	C	C	–	W
SC$_\tau$	–	–	W	–	W

In Tables 4.5 to 4.8, x indicates that the performance measure was used, x* indicates that the performance measure was calculated in decision space and x$^\triangleright$ indicates that the variance of the performance measure was used. The usage of the accuracy performance measures in the DMOO literature is summarised in Table 4.5. Table 4.5 shows that most researchers have used the GD or VD performance measure to quantify the accuracy of POF^*.

4.2.2 Diversity Performance Measures

This section discusses performance measures that are used to measure the diversity of the solutions contained in the approximated POF.

***MS'* measure**

Goh and Tan [67] introduced an adapted version of MS (refer to Equation (4.7) in Section 4.1.3) to measure how well POF^* covers POF'. Contrary to MS, the adapted MS, MS', takes into account the proximity of POF^* to POF'. MS' is defined as:

$$MS' = \sqrt{\frac{1}{n_k}\sum_{k=1}^{n_k}\left[\frac{\min[\overline{POF_k^*}, \overline{POF_k'}] - \max[\underline{POF_k^*}, \underline{POF_k'}]}{\overline{POF_k'} - \underline{POF_k'}}\right]^2} \tag{4.15}$$

Table 4.5: Usage of DMOO accuracy performance measures

Year	Authors	Accuracy				
		GD	**IGD**	**VD**	**Acc**	**SC**
2004	Farina *et al.* [58]	x, x^*				
2005	Guan *et al.* [74]	x, $x^{\triangleright}$				
2006	Hatzakis and Wallace [76]	x, $x^{\triangleright}$, x^*, $x^{*\triangleright}$				
2006	Mehnen *et al.* [117]	x^*				x
2007	Cámara *et al.* [18] [15]				x	
2007	Li *et al.* [108]		x			
2007	Zhou *et al.* [166]	x, $x^{\triangleright}$				
2008	Isaacs *et al.* [87]	x, x^*				
2008	Tan and Goh [146]			x^*		
2009	Cámara *et al.* [16]				x	
2009	Chen *et al.* [23]	x				
2009	Wang and Li [155]		x, $x^{\triangleright}$			
2009	Goh and Tan [67] [66]			x^*		
2009	Isaacs *et al.* [88]	x, x^*				
2009	Salazar Lechuga [102]	x, $x^{\triangleright}$				
2009	Ray [127]	x, x^*				
2010	Cámara *et al.* [17] [138]				x	
2010	Koo *et al.* [100]			x^*		
2010	Wang and Li [156]		x			
2011	Helbig and Engelbrecht [78]			x	x	

Similar to MS, MS' is not weakly compatible with any of the outperformance relations. Adding a non-dominated solution to POF^* will not necessarily lead to a higher MS' value. Therefore, MS' adhere to weak monotony. POF' will obtain the maximum MS value, but even a POF^* that has only two non-dominated solutions at the extreme points of POF' will also obtain the maximum MS value. Therefore, MS' adheres to weak relativity.

PL **measure**

Since many diversity performance measures are based on the Euclidean distance and therefore do not take the shape of the POF into account, Mehnen *et al.* [117] introduced a performance measure, the PL measure, that is based on path lengths or path integrals. The length of the path between two solutions is defined as:

Definition 4.12. Length of Path between Two Solutions: Let γ be the path between two solutions in objective space, $\mathbf{a}$ and $\mathbf{b}$, that is differentiable in $[\mathbf{a}, \mathbf{b}]$. Then the length of a path between $[\mathbf{a}, \mathbf{b}]$ on γ is defined as:

$$L(\gamma, \mathbf{a}, \mathbf{b}) := \int_b^a |\dot{\gamma}| \, \mathrm{dt} = \int_b^a \sqrt{\dot{\gamma_1}^2 + \ldots + \dot{\gamma_m}^2} \, \mathrm{dt} \tag{4.16}$$

where $\dot{\gamma}$ is the derivative of γ and $|\dot{\gamma}|$ is the Euclidean norm of $\dot{\gamma}$.

The PL performance measure is the normalised product of the path between sorted neighbouring solutions on POF, defined as

$$\begin{aligned} PL_\tau &:= \frac{\ln\left(\prod_{\mathbf{f}(\mathbf{x}_i)\in POF} \zeta(\mathbf{x})\right)}{\ln e^{L_{POF}}} \\ &= \frac{\sum_{\mathbf{f}(\mathbf{x}_i)\in POF} \ln(\zeta(\mathbf{x}))}{L_{POF}} \end{aligned} \tag{4.17}$$

where $\zeta(\mathbf{x}_i) = L(\gamma, \mathbf{f}(\mathbf{x}_i), \mathbf{f}(\mathbf{x}_{i+1})) + 1$ and $\mathbf{f}$ represents the objective functions. For the calculation of PL, a solution is considered as being in POF if the solution is within an

Table 4.6: Usage of DMOO combined performance measures

Year	**Authors**	**Quality**					
		HV	**HVR**	**HVD**	**HV**$_{max}$	ϵ_{bin}	**OSPA**
2007	Deb *et al.* [46]		x				
2007	Cámara *et al.* [18] [15]	x					
2007	Li *et al.* [108]		x				
2007	Zheng [165]		x		x, x$^{\triangleright}$		
2007	Zhou *et al.* [166]			x			
2008	Greeff and Engelbrecht [72]	x					
2008	Talukder [144]	x					
2008	Talukder [96]		x				
2009	Avdagić *et al.* [2]	x					
2009	Cámara *et al.* [16]	x			x		
2010	Azevedo and Araújo [3]	x					
2010	Cámara *et al.* [17] [138]		x	x			
2010	Greeff and Engelbrecht [71]	x					
2010	Kim *et al.* [97]					x	
2010	Wang and Li [156]	x					
2011	Deb [41]		x				
2011	Helbig and Engelbrecht [78]		x				
2011	Tantar *et al.* [150]						x

ϵ-region near POF.

In order to calculate the PL performance measure, an analytic closed description of the true POF is required. However, according to Mehnen *et al.* the calculation of the PL measure is complicated when a DMOOP:

- has more than two objectives, or
- has a discontinuous POF.

In these situations Mehnen *et al.* [117] recommend the usage of S [132] (refer to Section 4.1.3).

PL is not weakly compatible with the outperformance relations. However, it is weakly compatible with monotony, since the value of PL increases when a new non-dominated solution that is within ϵ-distance of POF is added to POF^*.

Set Coverage Metric

Guan *et al.* [74] introduced a set coverage measure that is based on the S and D metrics introduced by Zitzler [167]. The HV of the objective space that is dominated by POF^* but not by POF', referred to as the D-metric, is defined as

$$D(POF^*, POF') = HV(POF^* + POF') - HV(POF') \tag{4.18}$$

The set coverage metric is then defined as

$$\eta = \frac{D(POF^*, POF')}{HV(POF')} + \frac{D(POF', POF^*)}{HV(POF')} \tag{4.19}$$

Therefore, the set coverage metric, η, is the normalised sum of the:

- HV of the objective space that is dominated by POF^* and not by POF', and
- HV of the objective space that is dominated by POF' and not by POF^*.

η is weakly compatible with O_W if the HV is weakly compatible with O_W. Therefore, η is weakly compatible with O_W if the reference vector is selected in such a manner that all feasible non-dominated sets that are evaluated have a positive HV value. If the reference vector is selected in this manner, η is compatible with all the outperformance relations. Furthermore, η is then weakly compatible with monotony and weakly compatible with relativity.

Pareto Front Extent

Zhang and Qian [164] introduced the coverage scope (CS) measure to quantify the average width or coverage of the non-dominated set. CS is calculated by averaging the maximum distance between each solution in POF^* and the other solutions in POF^*. Therefore, CS is defined as

$$CS = \frac{1}{n_{POF^*}} \sum_{i=1}^{n_{POF^*}} \max\{\| \mathbf{f}(\mathbf{x}_i) - \mathbf{f}(\mathbf{x}_j) \|\} \tag{4.20}$$

with $\mathbf{x}_i, \mathbf{x}_j \in POF^*$, $i \geq 1$ and $j \leq n_{POF^*}$.

A higher CS value indicates a better performance. CS is similar to S [132] (refer to Section 4.1.3), but use the maximum distance where S uses the minimum distance between the non-dominated solutions in POF^*. Similar to S, CS is not weakly compatible with the outperformance relations. Furthermore, CS is not weakly compatible with monotony, since adding a non-dominated solution to POF^* can decrease the CS value. The CS value of POF' can be less than the CS value of POF^*. Therefore, CS is not weakly compatible with relativity.

A summary of the compatibility of diversity performance measures is shown in Table 4.8. In Table 4.8, C_o presents C_{orig} and C_m presents C_{mod}. Table 4.8 indicates that most researchers used S and MS to quantify the diversity of the non-dominated solutions in POF^*.

Table 4.7: Compatibility of diversity performance measures

Performance Measure	**O_W**	**O_S**	**O_C**	**M**	**R**
MS'	–	–	–	W	W
PL	–	–	–	W	–
η	C*	C*	C*	W*	W*
CS	–	–	–	–	–

4.2.3 Robustness Performance Measures

This section discusses performance measures that quantify the robustness of the algorithm, i.e. how well the algorithm recovers after an environment change occurs.

Stability Measure

The effect of the changes in the environment on the accuracy (*acc* defined in Equation 4.23) of the algorithm can be quantified by the measure of stability that was introduced by Weicker [157] for DSOO and adapted for DMOO by Cámara *et al.* [138].

Table 4.8: Usage of DMOO diversity performance measures

Year	**Authors**	**Diversity/Spread**											
		NS	**C$_o$**	**C$_m$**	**PFE**	**Spac** (Schott)	**U**	**PL**	**MS**	**Spread** (Deb)	**Entropy**	**Accum**	**N$_E$**
2005	Guan *et al.* [74]	x		x									x
2006	Mehnen *et al.* [117]					x		x					
2006	Zheng *et al.* [160]									x			
2008	Greeff and Engelbrecht [72]	x				x							
2008	Tan and Goh [146]								x				
2008	Wang and Dang [153]		x				x						
2009	Avdagić *et al.* [2]		x										
2009	Goh and Tan [67] [66]								x				
2009	Chen *et al.* [23]										x		
2010	Azevedo and Araújo [3]										x	x	
2010	Greeff and Engelbrecht [71]					x							
2010	Koo *et al.* [100]								x				
2010	Liu [110]						x						
2011	Helbig and Engelbrecht [78]					x			x				
2011	Zang *et al.* [164]		x		x	x							

Stability is defined as

$$stab(t) = \max\{0, acc(t-1) - acc(t)\} \tag{4.21}$$

where a low *stab* value indicates good performance.

The compatibility of *stab* depends on the definition of *acc*. If *acc* as defined in Equation (4.23) is used, *stab* is compatible with O_W if *acc* is compatible with O_W. Under these conditions, *stab* is compatible with the outperformance relations and weakly compatible with monotony and relativity.

Reactivity Measure

Cámara *et al.* [138] presented a measure of reactivity based on the reactivity performance measure introduced by Weicker [157] for DSOO. Reactivity measures how long it takes for an algorithm to recover after a change in the environment, by determining how long it takes for an algorithm to reach a specified accuracy threshold. The reactivity performance measure is defined as

$$react(t, \epsilon) = \min\left\{t' - t | t < t' < \tau_{max}, t' \in \mathbb{N}, \quad \frac{acc(t')}{acc(t)} \geq (1-\epsilon)\right\} \tag{4.22}$$

where τ_{max} is the maximum number of iterations or generations.

Similar to *stab*, *react* is weakly compatible with O_W if *acc* is weakly compatible with O_W. *react*'s compatibility with monotony and relativity also depends on *acc*'s compatibility with monotony and relativity.

The compatibility with the outperformance relations by the robustness performance measures is summarised in Table 4.9.

Table 4.9: Compatibility of robustness performance measures

Performance Measure	$\mathbf{O_W}$	$\mathbf{O_S}$	$\mathbf{O_C}$	**M**	**R**
stab	C*	C*	C*	W*	W*
react	C*	C*	C*	W*	W*

Table 4.10 summarises the usage of performance measures that quantifies robustness in the DMOO literature.

Table 4.10: Usage of DMOO robustness performance measures

Year	**Authors**	**Robustness**	
		Stb	**React**
2007	Cámara *et al.* [18] [15]	x	x
2009	Cámara *et al.* [16]	x	x
2010	Cámara *et al.* [17] [138]	x	x
2011	Helbig and Engelbrecht [78]	x	

4.2.4 Combined Performance Measures

This section discusses performance measures used to quantify the overall quality of the found POF, i.e. they do not measure only one aspect such as convergence to the true POF or the diversity of the solutions.

Accuracy Measure

A measure of accuracy introduced by Weicker for DSOO [157] was adapted by Cámara *et al.* [138] for DMOO. This measure quantifies the quality of the solutions as a relation between the HV of POF^* and the maximum HV that has been found so far. The accuracy measure is defined as

$$acc(t) = \frac{HV(POF^*(t))}{HV_{max}(POF^*(t))} \tag{4.23}$$

The accuracy measure, acc, is compatible with O_W if the upper boundary of the dominated region is set in such a way that all feasible non-dominated sets that are evaluated have a positive HV value (refer to Section 4.1.4). Under these conditions, acc is compatible with the outperformance relations and weakly compatible with monotony and relativity.

Hypervolume Difference

Zhou *et al.* [166] suggested to use the hypervolume distance (HVD) to measure the quality of the found POF. HVD is defined as

$$HVD = HV(POF') - HV(POF^*) \tag{4.24}$$

However, when the true POF is unknown, the HVD cannot be used. Zheng used the maximum HV to measure the quality of the found POF [165].

Cámara *et al.* [17] extended the definition of their accuracy measure (Equation (4.23)) to use the HVD when the true POF is known. The alternative accuracy measure is defined as

$$acc_{alt}(t) = |HV(POF'(t)) - HV(POF^*(t))| \tag{4.25}$$

where $acc_{alt}(t)$ is the absolute HVD at time t. The absolute values ensure that $acc_{alt}(t) \geq 0$, even if $HV(POF^*) > HV(POF')$. HVD is compatible with the outperformance relations if HV is compatible with the outperformance relations.

Optimal Subpattern Assignment Measure

Recently Tantar *et al.* [150] introduced performance measures that are based on performance measures used in quantifying the tracking quality in multi-object tracking problems. The performance measures are developed based on the optimal subpattern assignment (OSPA) measure that can be used to compare sets with different cardinality [133].

Let $P = (F, X, N)$ define a DMOOP with F and X representing a set of objective functions and a set of decision variables respectively. N represents the neighbourhood function described by a ball of center c and radius r, defined as

$$N(c, r) = \{\mathbf{x} \in X |\ d(\mathbf{x}, c) < r \text{ and } \exists h| \mathbf{x}hc\} \tag{4.26}$$

where d is the distance between a solution, $\mathbf{x}$, and the center point of the neighbourhood, c, and $\exists h| \mathbf{x}hc$ indicates that the neighbourhood can be reached through a transformation h.

Let A and B respresent two approximated POFs with cardinality of m and n respectively. Then the following two performance measures are defined:

$$M_{loc}(X, Y) = \left(\frac{1}{n_{POF^*_B}} \min_{j \in P} \left\{ \sum_{i=1}^{n_{POF^*_A}} d(x_i, y_{j(i)})^p \right\} \right)^{\frac{1}{p}} \tag{4.27}$$

where $d(\mathbf{x}, \mathbf{y}) = min\{c, d(\mathbf{x}, \mathbf{y})\}$ is the minimum distance between two solutions that are cut off by c. When comparing A and B, the solutions from B that are in the neighbourhood of a given solution from A are determined by considering all permutations of solutions from B, referred to as the set P. M_{loc} quantifies the quality of the coverage of A as compared to B. A drawback of this performance measure is its computational cost, because of the calculation of permutations for each solution under consideration.

The other performance measure is defined as

$$M_{card}(X,Y) = \left(\frac{c^p(n_{POF_B^*} - n_{POF_A^*})}{n_{POF_B^*}}\right)^{\frac{1}{p}} \tag{4.28}$$

M_{card} is a cardinality penalty function that is used when $|A| \neq |B|$, and is zero if the two sets have the same cardinality. M_{card} measures the influence of the cardinality difference on the overall quality of the larger set, with the cut-off term as the error quantification factor.

The OSPA metric is then defined as:

$$OSPA(X,Y) = M_{loc}(X,Y) + M_{card}(X,Y) \tag{4.29}$$

$OSPA$ is not weakly compatible with the outperformance relations. However, $OSPA$ is weakly compatible with monotony.

Table 4.11 summarises the compatibility with the outperformance relations by the combined performance measures.

Table 4.11: Compatibility of combined performance measures

Performance Measure	$\mathbf{O_W}$	$\mathbf{O_S}$	$\mathbf{O_C}$	**M**	**R**
HVD	C*	C*	C*	W*	W*
acc_{alt}	C*	C*	C*	W*	W*
$OSPA$	–	–	–	W	–

When algorithms solve DMOOPs, five major issues should be taken into consideration when selecting performance measures to quantify the performance of the algorithms, namely: algorithms losing track of the changing POF, the effect of outlier solutions in the found POF, boundary constraint violations, calculating the performance measures in either the objective or decision space, and comparing the performance of the various algorithms. These issues are discussed in the next section.

4.3 Issues with Current Dynamic Multi-objective Optimisation Performance Measures

Section 4.2 discussed a number of performance measures that have been used to quantify the performance of algorithms on DMOOPs. Even though these measures have been used in a number of articles, they suffer from a number of problems mostly related to aspects of dynamic environments. These problems make general applicability of these performance measures to all DMOOPs not possible.

Section 4.3.1 discusses misleading results that can occur when algorithms lose track of the changing POF. The effect of outlier solutions in POF^* on the quantification of an algorithm's performance is discussed in Section 4.3.2. Section 4.3.3 discusses the effect of boundary constraints violations on the performance of DMOO algorithms. Furthermore, performance measures can be calculated in either the objective or decision space as discussed in Section 4.3.4. Finally, Section 4.3.5 discusses issues when comparing the performance of the various algorithms.

4.3.1 Losing Track of the Changing Pareto Optimal Front

When a DMOO algorithm loses track of the changing POF, and POF changes over time in such a way that its HV value decreases, many of the current performance measures will give misleading results. Figure 4.1 illustrates example POFs where the POF changes over time in such a way that, if the HV is calculated with the reference vector being the worst values of each objective, the HV will decrease over time. A decrease in the HV will occur if for each example the POF changes from convex to concave. Figure 4.1 illustrates three such POFs. In Figure 4.1, the first POF is represented by the bottom line and the last POF by the top line.

The problem of losing track of the POF was first observed by Helbig and Engelbrecht [77], where five algorithms were used to solve the FDA2 DMOOP. These algorithms included a dynamic VEPSO (DVEPSO) which uses clamping to manage boundary constraint violations (DVEPSO-A) [77], DVEPSO that uses per element re-initialisation to manage boundary constraint violations (DVEPSO-B) [77], NSGA-II where a percentage of individuals are randomly selected and replaced with newly created individuals if an environment change occurs (DNSGA-II-A) [46], NSGA-II where, after an environment change, a percentage of individuals are randomly selected and replaced with individuals that are mutated from existing individuals (DNSGA-II-B) [46], and dCOEA [67]. Figure 4.2 illustrates example POFs obtained by these algorithms in comparison with POF (Figure 4.2(f)). It is clear from these figures that DNSGA-II-A, DNSGA-II-B and dCOEA lost track of the changing POF, with the DVEPSO algorithms being more successful in tracking the POF. It is therefore expected that the values of the performance measures should be better for the DVEPSO algorithms than for the evolutionary algorithms.

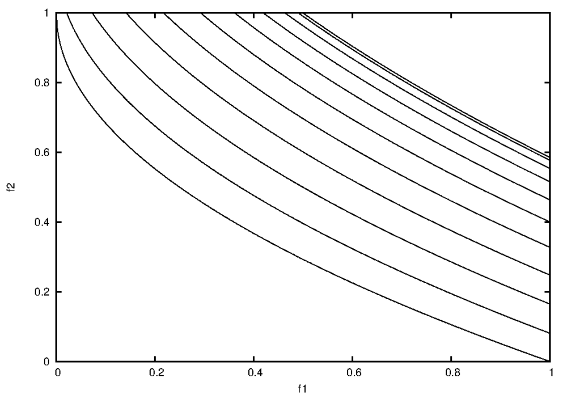
The performance measure values of these algorithms for a change frequency of ten are presented in Table 4.12. In Table 4.12 NS refers to the number of non-dominated solutions found, S refers to the spacing measure defined by Schott [132], HVR refers to the HV ratio [108], Acc and $stab$ refer to measures of accuracy and stability presented by Cámara *et al.* [18], and VD and MS refer to the adapted generational distance and maximum spread performance measures for dynamic environments proposed by Goh and Tan [67].

As shown in Table 4.12, performance measures VD and MS indicate the DVEPSO algorithms to be better than the evolutionary algorithms. However, the measures that make use of the HV, namely HVR, Acc and $stab$, rank the evolutionary algorithms as being better than the DVEPSO algorithms. This occurs since the HV value of POF decreases over time and therefore from the time where an algoritm loses track of the changing POF, its HV value is higher than that of POF and therefore higher than that of algorithms that are tracking the changing POF. Since the HV value of POF decreases over time, HVR (which divides the HV of POF^* by the HV of POF) still does not address this problem.

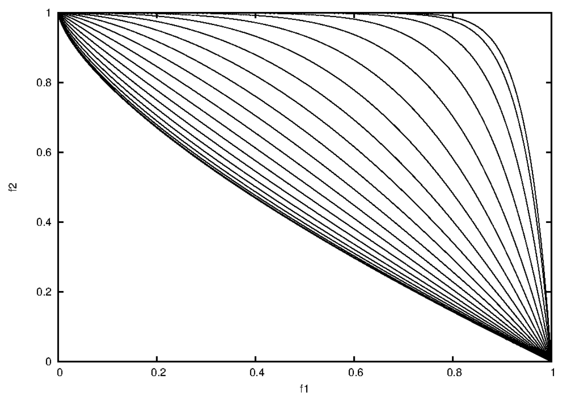
Tables 4.5 and 4.6 show that the following papers used the HV or HVR without using any accuracy measure that requires knowledge of the true POF: [2, 3, 15, 18, 16, 17, 41, 46, 72, 71, 96, 138, 144, 165]. Therefore, if any of the algorithms that were evaluated in these studies lost track of the changing POF, the performance measure values that were obtained may be misleading.

The issue of an algorithm losing track of the changing POF is unique to DMOO. In order to overcome this problem, acc_{alt} proposed by Cámara *et al.* (refer to Equation (4.25) in Section 4.2.4) should be used when the POF is known. Furthermore, if acc_{alt} is used for acc, $stab$ will also be reliable even if an algorithm loses track of POF.

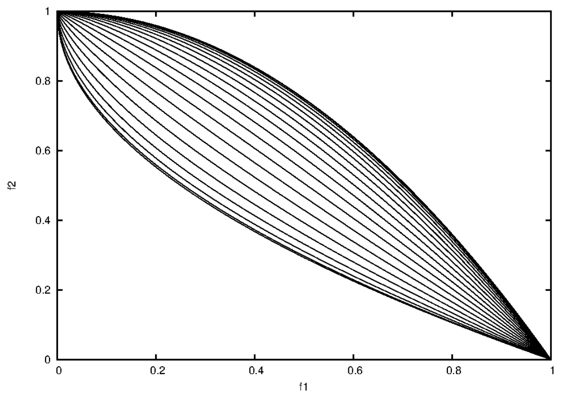
If POF is unknown, as is the case with most real-world problems, the deviation of the performance measures that use the HV measure should also be calculated. If



(a) FDA3



(b) FDA2



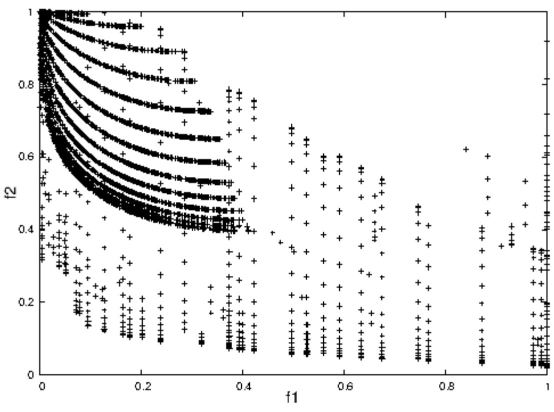
(c) dMOP2

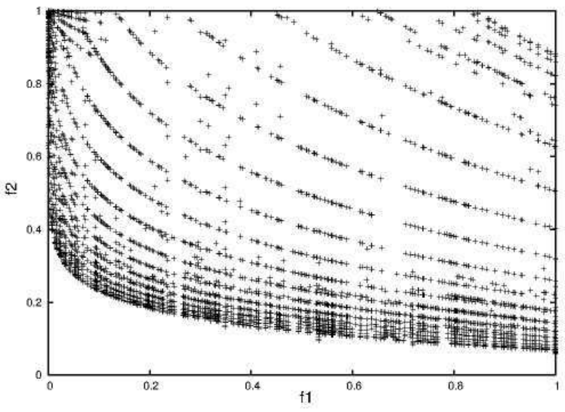
Figure 4.1: Examples of DMOOPs where the *HV* value of *POF* decreases over time

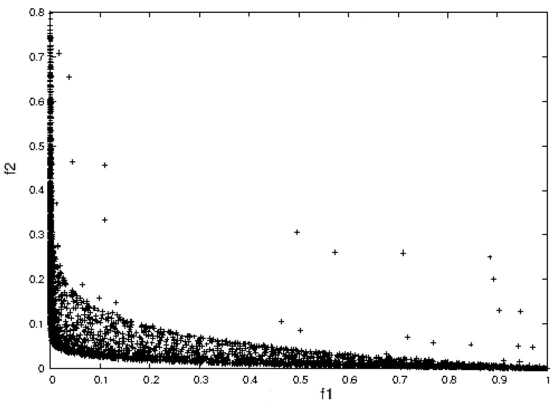
Table 4.12: Performance Measure Values for FDA2

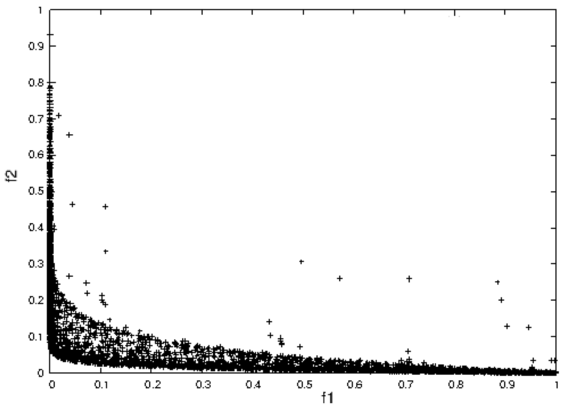
τ_t	**Algorithm**	**NS**	**S**	**HVR**	**Acc**	**Stab**	**VD**	**MS**	**R**
10	DVEPSO-A	**73.4**	0.00118	0.99533	0.97848	0.00049	0.45824	**0.90878**	4
10	DVEPSO-B	63	0.00391	0.99905	0.98157	0.00029	**0.43234**	0.88916	3
10	DNSGAII-A	39.4	0.00044	1.0044	0.98681	$9.565\text{x}10^{-06}$	0.71581	0.77096	2
10	DNSGAII-B	39.6	**0.00042**	**1.00441**	**0.98683**	$\mathbf{9.206x10^{-06}}$	0.71681	0.77866	**1**
10	dCOEA	38.4	0.00051	1.00209	0.98454	0.00122	0.70453	0.61923	5

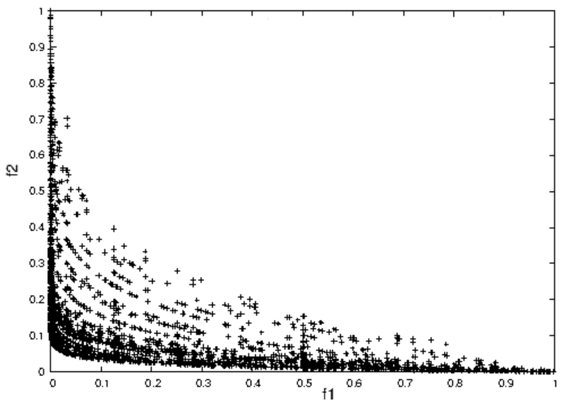
the performance measure's deviation varies much more for certain algorithms, it may indicate that one or more of the algorithms have lost track of the changing POF and


(a) POF^* found by DVEPSO-A


(b) POF^* found by DVEPSO-B


(c) POF^* found by DNSGA-II-A


(d) POF^* found by DNSGA-II-B


(e) POF^* found by dCOEA

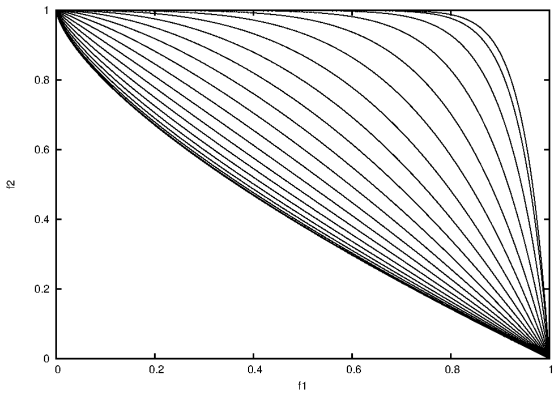

(f) POF of FDA2

Figure 4.2: POF and POF^* found by various algorithms for FDA2 with $n_t = 10$, $tau_t = 10$ and 1000 iterations

that the performance measure can not reliably be used to compare the performance of the different algorithms. Therefore, the graphs of POF^*s should be plotted and checked to determine whether any algorithm has lost track of the changing POF.

Even though various performance measures were used, the misleading performance measures can play a large enough role to influence the overall ranking of the algorithms. This is shown in Table 4.12. Even though the DVEPSO algorithms ranked the highest with regards to NS, VD and MS, the measures that make use of the HV value affected the ranking in such a way that the DVEPSO algorithms ranked as number 3 and 4 respectively and were outranked by the DNSGA-II algorithms - portraying an incorrect ordering.

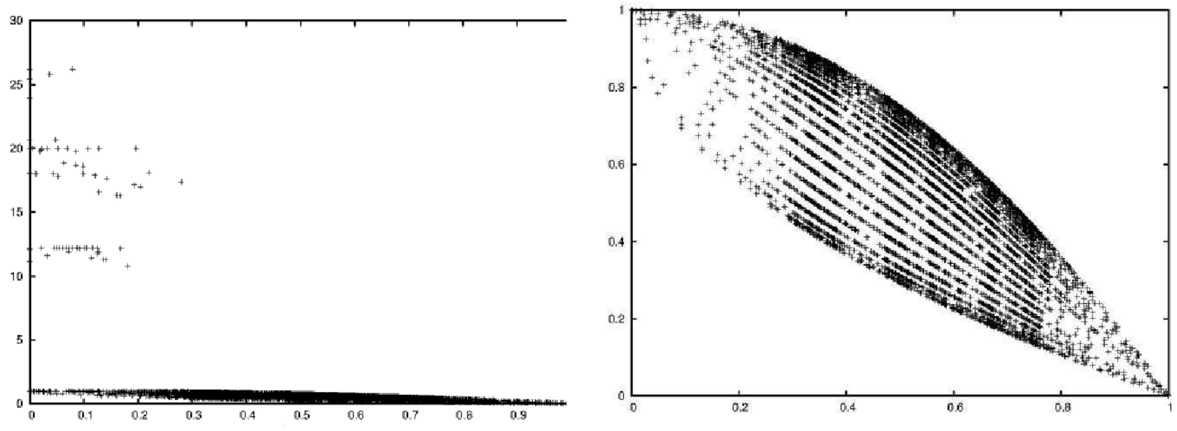
It should be noted that the stability measure, *stab*, proposed by Cámara *et al.* [18] measures the change in the values of the accuracy measure at two consecutive time steps (refer to Section 4.2.3). Under normal circumstances a low *stab* value indicates that the performance of the algorithm is not severely affected by the change in the environment. However, in situations where one or more algorithm(s) lost track of the changing POF, the lowest *stab* value will be obtained by the algorithms that lost track of the changing POF. Therefore, the results obtained with the *stab* performance measure will be misleading. Table 4.12 shows that the NSGA-II algorithms obtained a better *stab* value than the DVEPSO algorithms. Clearly, as indicated by the POFs shown in Figure 4.2, this is not the case.

4.3.2 Outliers in the Pareto Optimal Front

When algorithms solve DMOOPs and the environment changes frequently, the POF^* that has been found by the algorithm for a specific time step may contain outliers. This occurs because the algorithm found non-dominated solutions that are further away from the true POF within the number of iterations or generations available to the algorithm to solve the specific POF. In the time available before the environment changes, the algorithm did not find any solutions closer to the true POF that dominated these outlier solutions. Figure 4.3 illustrates an example POF^* that contains outliers.

Outliers will skew results obtained using:

- distance-based performance measures, such as GD, VD, PL, CS and M_{loc},


(a) POF^* of dMOP2 with outliers (b) Zoomed into POF region of (a)

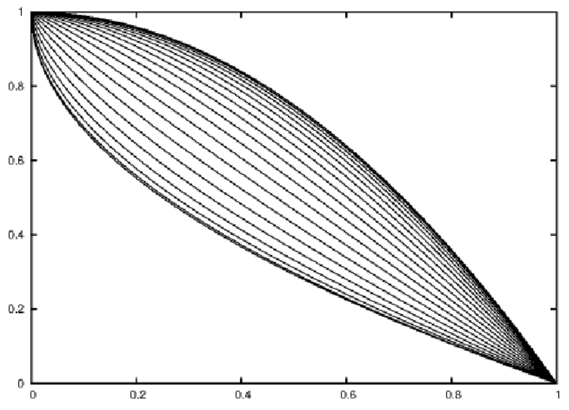

(c) POF of dMOP2

Figure 4.3: Example of a POF^* that contains outlier solutions.

- performance measures that measure the spread of the solutions, such as MS, and
- the HV performance measure.

The influence of outlier solutions on the calculation of GD and VD is illustrated in Table 4.13. Due to the large distance between the outliers and POF as shown in Figures 4.3 and 4.4, the resulting GD and VD is much larger with the outliers present compared to when the outliers are not present. However, it should be noted that the severity of the influence of outliers on distance calculations depends on the number of outliers and their distance from POF.

Furthermore, when a performance measure, such as MS of Cámara *et al.* [18], measures the extend or spread of the approximated POF, these outlier solutions may cause the performance measure to be misleading with regards to the performance of the algorithm. In Figure 4.4, the outlier solutions' f_1 and f_2 values will become the $\overline{PF_i^*}$ and $\underline{PF_i^*}$ in the f_2 and f_1 objective in Equation (4.7) respectively. In Figure 4.4, POF^* only contains non-dominated solutions with f_1 values in the range of $[0.2, 0.7]$ and f_2 values in the range of $[0, 0.5]$ without the outlier solutions. However, with the outlier solutions the f_1 values will be calculated as being in the range of $[0, 1.0]$ and f_2 values in the range of $[0.2, 3]$. This will result in the maximum MS value and will not give a true reflection of the diversity of solutions that has been found by the algorithm. The influence of the outliers on the value of MS is shown in Table 4.13.

When solving DMOOPs, many researchers use the HV performance measure, especially when POF is unknown. When comparing various algorithms' POF^*s, the same reference vector is used. HV values that are calculated with different reference vectors cannot be compared against each other. Furthermore, outlier solutions influence the reference vector values that are used to calculate the HV. Typically, the reference vector is chosen as the worst values obtained for each objective. Therefore, for POF^* in Figure 4.4 the reference vector for the HV is $[1.1, 3.1]$ and $[1.1, 1.1]$ with and without the outlier values respectively. This results in much larger HV values when outliers are present, as shown in Table 4.14. From Table 4.14 it is clear that HVR and acc_{alt} provide a more accurate representation of the performance of the algorithm, resulting in a better HVR value without outliers than with the outliers. However, when the HV is used, the POF^* with outliers obtain a better HV value than the POF^* without the outliers. Therefore, if POF is unknown and the HV is used, outlier solutions may lead to misleading results and algorithms being ranked incorrectly.

One approach to manage outliers in POF^* is to remove the outliers from POF^*. However, no consensus exists on the approach that should be followed to decide which non-dominated solutions in POF^* should be classified as outliers.

It should be noted that, as the number of objectives increases, more outlier solutions may become present in POF^*. This is the case, since as the number of objectives increases, more solutions that are found by the algorithm will be non-dominated with regards to the other solutions in POF^*. Furthermore, outliers in POF^* will cause the same problems when solving static MOOPs. However, since algorithms generally have longer time to converge towards POF with static MOOPs than with DMOOPs where the environment changes, the possibility of the occurrence of outliers increases when solving DMOOPs.

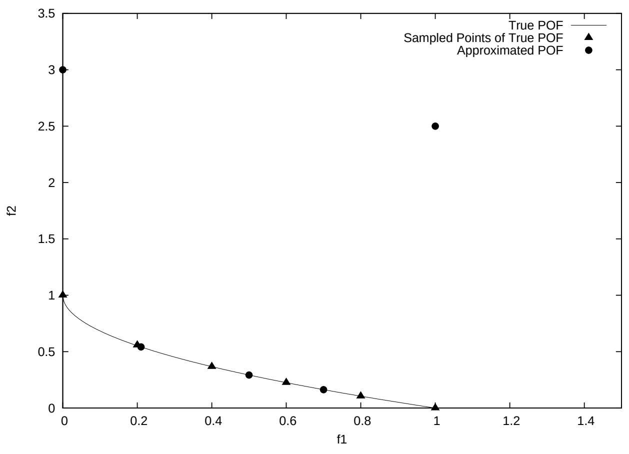


Figure 4.4: POF^* of FDA1 with outlier solutions

Table 4.13: GD, VD and MS values for FDA1

Outliers	**GD**	**VD**	**MS**
Yes	2.05565	4.596574	0.91833
No	0.00942	0.016311	0.4342

Table 4.14: HV, HVR and HVD values for FDA1

Outliers	**HV**	**HVR**	$\mathbf{acc}_{alt}$
Yes	2.49898	0.84461	0.45974
No	0.69798	0.91994	0.06074

4.3.3 Boundary Constraint Violations

The candidate solutions of certain computational intelligence algorithms tend to move outside the boundary constraints of an optimisation problem while searching for solutions. For example, it has been shown theoretically that most particles in a PSO algorithm [94] leave the bounds within the first few iterations [55, 63]. If a particle finds a better solution outside the bounds, its personal best position is set to this new position. Should this position be better than the current neighbourhood or global best, other particles are also pulled outside of the bounds. Consequently, particles may converge on a solution outside the bounds of the search space. This behaviour of particles was empirically analyzed by Engelbrecht [56].

If a GA [82] uses blending cross-over, such as parent-centric cross-over [44], offspring may be generated outside the boundaries of the search space due to the asymptotic tails of the distributions of the stochastic component of the blending process.

Most evolutionary programming [59] algorithms sample mutational step sizes from zero-mean distributions with tails that asymptotically approach infinity and negative infinity. Consequently, large mutational step sizes can potentially be added to parent individuals, moving offspring outside of the bounds. If such offspring have better fitness than parent individuals, these offspring survive to the next generation with a chance of obtaining a solution that does not lie within the bounds of the search space.

With differential evolution's [142] mutation operator, a weighted difference of two vectors are added to the parameter vector. If this weighted difference is large, it may cause the trial vector to move outside the boundary constraints of the optimisation problem.

Most unconstrained DMOOPs have boundary constraints that limit the search space. However, if an algorithm does not manage boundary constraint violations, infeasible solutions may be added to POF^*. These infeasible solutions may dominate feasible solutions in POF^*, which will cause the feasible solutions to be removed from POF^*. Furthermore, the infeasible solutions may cause misleading results with regards to an algorithm's performance.

Figure 4.5(a) illustrates a POF^* that was found by DVEPSO that did not manage boundary constraint violations (DVEPSO$_u$) when solving dMOP2, DVEPSO that manages boundary constraint violations (DVEPSO$_c$), and the true POF (POF). From Figure 4.5 it is clear that POF^* of DVEPSO$_u$ has a larger HV value than both POF^* of DVEPSO$_c$ (refer to Figure 4.5(b)) and POF (refer to Figure 4.5(c)). This is confirmed in Table 4.15. This incorrectly indicates that the POF^* that contains solutions that are outside the bounds of the search space to be better. Therefore, when comparing various algorithms with one another, it is important to check that all algorithms manage boundary contraint violations to ensure a fair comparison. It should be noted that the issue of boundary constraint violations is applicable to both SMOO and DMOO.

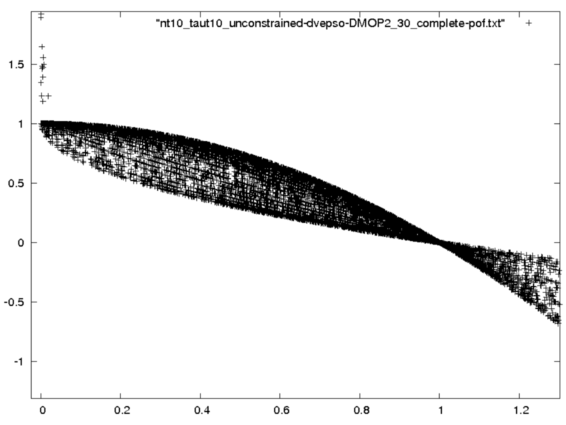
Table 4.15: HVR values for dMOP2

Algorithm	**HVR**
DVEPSO$_u$	1.00181
DVEPSO$_c$	0.99978

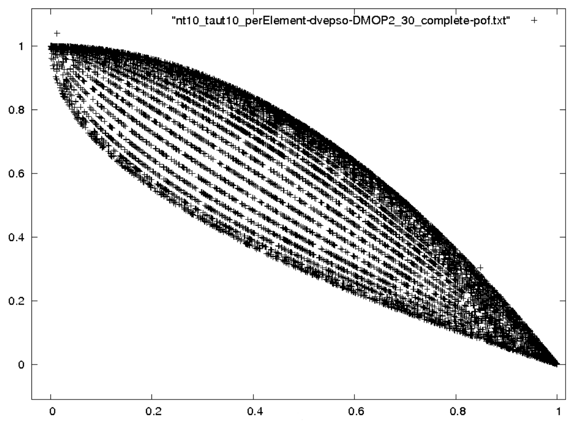
4.3.4 Objective Space versus Decision Space

Accuracy measures, such as VD or GD, can be calculated with respect to either the decision or the objective space. Using objective space, VD measures the distance between the non-dominated solutions of POF^* and POF'. Therefore, VD measures the closeness of POF^* to POF. Since one of the goals of solving a DMOOP is to track the changing POF, the accuracy should be measured in the objective space. If the VD measure is calculated in the decision space, the distance between POS^* and POS is calculated. Calculating the VD measure in the decision space may be useful to determine how close POS^* is from POS. However, if for a DMOOP a small change in the POS causes a big change in the POF, it may occur that even though the algorithm's POS^* is very close to POS, POF^* is quite far from POF. This is illustrated with an example DMOOP defined as:

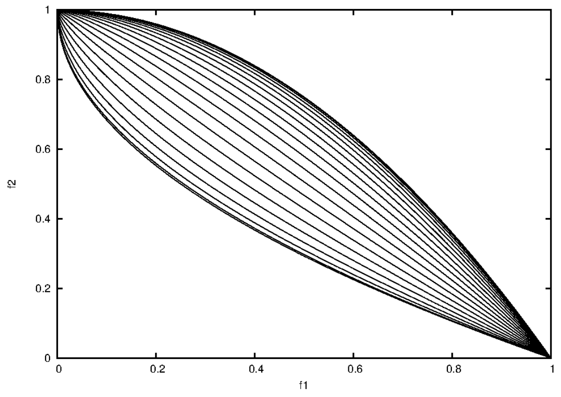
$$\mathsf{DMOOP}_1 = \begin{cases} Minimize : f(\mathrm{x}, t) = (f_1(\mathbf{x_I}), g(\mathbf{x_{II}})\cdot \\ \qquad\qquad h\left(\mathbf{x_{III}}, f_1(\mathbf{x_I}), g\left(\mathbf{x_{II}}\right), t\right)) \\ f_1(\mathbf{x_I}) = x_1 \\ g(\mathbf{x_{II}}) = 1 - \sum_{x_i \in \mathbf{x_{II}}} \sqrt{x_i - G(t)} - \\ \qquad \sum_{x_j \in \mathbf{x_{III}}} \left(x_j - G(t)\right)^2 \\ h(\mathbf{x_{III}}, f_1, g, t) = 1 - \left(\frac{f_1}{g}\right)^{H(t)} \\ where : \\ G(t) = \sin(0.5\pi t), \quad t = \frac{1}{n_t}\left\lfloor \frac{\tau}{\tau_t} \right\rfloor \\ H(t) = 1.5 + G(t) \\ \mathbf{x_I} \in [0, 1]; \quad \mathbf{x_{II}}, \mathbf{x_{III}} \in [-1, 1] \end{cases} \tag{4.30}$$



(a) POF^* of dMOP2 with feasible and infeasible solutions



(b) POF^* of dMOP2 with only feasible solutions



(c) POF of dMOP2

Figure 4.5: Example of a POF^* that contains infeasible solutions due to boundary constraint violations

For DMOOP$_1$, the POS and POF are:

$$POS: \; x_i = G(t), \; \forall x_i \in \mathbf{x_{II}}, \mathbf{x_{III}}$$
$$POF: \; y = 1 - f_1^{H(t)}$$

Let $\mathbf{x_{II}} = \{x_1, x_2, x_3\}$, $\mathbf{x_{III}} = \{x_4, x_5, x_6\}$, $t = 0.1$, $G(t) = 0.156$, $\mathbf{x}_1 = \{0.14, 0.16, 0.16, 0.16, 0.16, 0.16\}$ and $\mathbf{x}_2 = \{0.16, 0.16, 0.16, 0.16, 0.14, 0.16\}$. Then, measuring the distance between the solution and the true POS (i.e. in decision space), $d(\mathbf{x})_{dec}$, $\mathbf{x}_1$ and $\mathbf{x}_2$ obtains the same d_{dec} value. However, $\mathbf{x}_1$ and $\mathbf{x}_2$ produces the following gh values respectively: 0.937512 and 0.93183. The true POF value for $\mathbf{x}_1$ and $\mathbf{x}_2$ are 0.961453 and 0.951914 respectively. The difference between the gh values found by $\mathbf{x}_1$ and $\mathbf{x}_2$ and the true POF values, d_{obj}, are 0.023941 and 0.020084 respectively. Therefore, even though in the decision space the difference between $\mathbf{x}_1$ and $\mathbf{x}_2$ and the true POS produces the same d_{dec} value, their difference in objective space, d_{obj}, is different, with $\mathbf{x}_2$ being closer to the true POF than $\mathbf{x}_1$.

Measuring VD in the decision space will indicate how close the decision variable values are from POS. However, the VD value measured in decision space will not give a true reflection of the accuracy of POF^* with regards to POF. Therefore, measuring VD in decision space to determine the accuracy of the algorithm's found solutions only makes sense for DMOOPs of Type I where the POS changes over time, but the POF remains static. However, for DMOOPs of Type II and III, measuring VD in the decision space will not provide any information with regards to how well the algorithm has tracked the changing POF and therefore for these type of DMOOPs VD should be measured in objective space.

The following papers measured either GD or VD in only the decision space: [67, 66, 100, 117, 146]. In [146] only FDA1, which is a Type I DMOOP, was used and therefore measuring in the decision variable space makes sense. In [100], three DMOOPs of Type I (FDA1, DIMP1 and DIMP2) were used and one DMOOP of Type II (FDA3). For the Type II DMOOP, calculating in the decision space will only provide information with regards to tracking of the changing POS, but not with regards to tracking the changing POF. In [67, 66, 117], DMOOPs of Types I, II and III were used. For the DMOOP of Type III, measuring in the decision space only indicates whether POS^* remains close to

POS, which remains static over time. Therefore, it provides no information with regards to how well the algorithm has tracked the changing POF. The issue of calculating performance measures in either decision or objective space is unique to DMOO, since with SMOO both the POS and POF remain static.

4.3.5 Comparing Performance of Algorithms

When the performance of various algorithms are compared against one another, typically various performance measures are used. Some algorithms will perform very well with regards to certain performance measures and not so well with regards to the other performance measures. Therefore, typically each algorithm will be ranked according to its performance with regards to each performance measure. Then, for each algorithm its average rank is calculated. These averaged ranks are then used to determine how well each algorithm performed with regards to the other algorithms. However, the performance measures that are used for comparing various algorithms should be chosen with care. If the wrong performance measures are selected, it may lead to incorrect ordering as discussed in Section 4.3.1 and illustrated in Table 4.12 and [77]. Therefore, if POF is known, the usage of acc_{alt} is suggested. However, more research is required to determine the best performance measure(s) for cases where POF is unknown.

4.4 Summary

This chapter provided an analysis of performance measures for DMOO. Concepts of outperformance relations, compatibility, monotony and relativity were introduced that have been used to evaluate static MOO performance measures. Performance measures that were used for MOO to measure convergence to the true POF, diversity of the solutions, as well as overall quality of the approximated POF were discussed. Adaptation of the MOO performance measures for DMOO was presented, as well as performance measures that have been introduced specifically for DMOO. Furthermore, problems with current DMOO performance measures were discussed, indicating that algorithms that lose track of the POF, outliers in the found POF, and violation of the boundary constraints can produce misleading results with some performance measures that are currently used to measure the performance of DMOO algorithms.

The first part of the thesis provided background with regards to optimisation. The second part of the thesis discusses CI algorithms used to solve optimisation problems. The next chapter discusses population-based algorithms used to solve SOOPs.