Nonconvex Many-Objective Optimisation

Ioannis Giagkiozis

Department of Automatic Control and Systems Engineering
The University of Sheffield

A dissertation submitted in partial fulfilment of the requirements for the degree of
Doctor of Philosophy in Control Systems

October 2012

Executive Summary

As many-objective optimisation problems become more prevalent, evolutionary algorithms that are based on Pareto dominance relations are slowly becoming less popular due to severe limitations that such an approach has for this class of problems. At the same time decomposition-based methods, which have been employed traditionally in mathematical programming, are consistently increasing in popularity. These developments have been led by recent research studies that show that decomposition-based algorithms have very good convergence properties compared to Pareto-based algorithms.

Decomposition-based methods use a scalarising function to decompose a problem with multiple objectives into several single objective subproblems. The subproblems are defined with the help of weighting vectors. The *location* on the Pareto front that each subproblem tends to converge, strongly depends on the choice of weighting vectors and the scalarising function. Therefore, the selection of an appropriate set of weighting vectors to *decompose* the multi-objective problem, determines the distribution of the final Pareto set approximation along the Pareto front. Currently a limiting factor in decomposition-based methods is that the distribution of Pareto optimal points cannot be directly controlled, at least not to a satisfactory degree. *Generalised Decomposition* is introduced in this thesis as a way to optimally solve this problem and enable the analyst and the decision maker define and obtain the desired distribution of Pareto optimal solutions.

Furthermore, many algorithms generate a set of Pareto optimal solutions. An interesting question is whether such a set can be used to generate more solutions in specific locations of the Pareto front. *Pareto Estimation* - a method introduced in this thesis - answers this question quite positively. The decision maker, using the Pareto Estimation method can request a set of solutions in a particular region on the Pareto front, and although not guaranteed to be generated in the exact location, it is shown that the spatial accuracy of the produced solutions is very high. Also the cost of generating these solutions is several orders of magnitude lower compared with the alternative to restart the optimisation.

Acknowledgements

I would like to thank my supervisor Peter J. Fleming for his confidence in my abilities, his patience and guidance. I would also like to thank Ricardo H.C. Takahashi for the most interesting discussions and his invaluable perspective with respect to the present work, during his visit to the University of Sheffield, while supported by a Marie Curie International Research Staff Exchange Scheme Fellowship within the 7th European Community Framework Programme. My thanks are also due to Jacob Mattingley for providing access to his tool CVXGEN [1], a convex optimisation solver generator that reduced considerably the computational effort of many of the experiments in this work. I would also like to thank Evan J. Hughes and Stephen P. Banks for granting me the honour of being my examiners. Lastly my thanks and love to my wife Meg, for her never ending support during the preparation of this thesis.

This thesis has been written in LaTeX2e, a free document preparation system (see `www.latex-project.org`).

Statement of Originality

Unless otherwise stated in the text, the work described in this thesis was carried out solely by the candidate. None of this work has already been accepted for any other degree, nor is it being concurrently submitted in candidature for any degree.

Candidate: _____

Ioannis Giagkiozis

Supervisor: _____

Peter J. Fleming

Creare, ergo sum.

Contents

List of Figures

List of Tables

Nomenclature

Roman Symbols

$\mathbf{F}(\mathbf{x})$	Vector objective function
f_i	Scalar objective function
$\mathbb{Z}$	The set of integers
k	Number of objectives
n	Number of decision variables
$\mathbb{N}$	The set of natural numbers, zero included
$\mathcal{N}$	Normal distribution
$\mathcal{D}$	Pareto set in decision space
$\mathcal{P}$	Pareto set in objective space
$\mathbb{R}^n$	n-dimensional Euclidean space
$\mathbb{R}^n_{++}$	Positive orthant in $\mathbb{R}^n$
$\mathbb{R}^n_{+}$	Non-negative orthant in $\mathbb{R}^n$
S	Feasible region in decision space
$\mathcal{U}$	Uniform distribution
$\mathbf{w}$	Weighting vector
$\mathbf{x}$	Decision vector
Z	Feasible region in objective space
$\mathbf{z}^\star$	Ideal objective vector
$\mathbf{z}^{nd}$	Nadir objective vector
$\mathbf{z}^{\star\star}$	Utopian objective vector
$\mathbf{z}$	Objective vector

Superscripts

c	Set complement
T	Matrix transpose

Other Symbols

$\lvert\cdot\rvert$	Absolute value for scalar operands, or cardinality for set operands
aff	Affine hull
bd	Boundary
cl	Closure
cone	Conic hull
conv	Convex hull
dom	Domain of definition
$\emptyset$	Empty set
epi	Epigraph
$\exists$	Exists
$\nexists$	Does not exist
$\forall$	For all
$\preceq$	Generalised inequality
$\prec$	Strict generalised inequality
$\gg$	Much larger than
∇	Gradient
hypo	Hypograph
$\in$	In (set)
inf	Infimum
∞	Infinity symbol
int	Interior
$\ll$	Much smaller than
$\notin$	Not in (set)
$\lVert\cdot\rVert_p$	ℓ_p-norm
rank	Rank of a matrix
$\cap$	Set intersection

$\setminus$ Set difference

$\cup$ Set union

$\subset$ Proper subset

$\subseteq$ Subset

sup Supremum

$\triangle$ Symmetric difference (set)

Acronyms

ACO Ant Colony Optimisation

AIS Artificial Immune System

AS Ant System

CE Cross Entropy

DE Differential Evolution

DM Decision Maker

EA Evolutionary Algorithm

ES Evolution Strategies

GA Genetic Algorithm

gD Generalised Decomposition

GDE Generalised Differential Evolution

GP Genetic Programming

IDEA Iterated Density Estimation Algorithm

MACE-gD Many-Objective Cross Entropy with Generalised Decomposition

MACE Many-Objective Cross Entropy

MAEA-gD Many-Objective Evolutionary Algorithm with Generalised Decomposition

MAP Many-Objective Problem

MIDEA Multi-Objective Iterated Density Estimation Algorithm

MISA Multi-Objective Immune System Algorithm

MOBES Multi-Objective Evolution Strategy

MO-CMA-ES Multi-Objective Covariance Matrix Adaptation Evolution Strategy

MOEA/D Mutli-Objective Evolutionary Algorithm Based on Decomposition

MOGA Fonseca-Fleming Multi-Objective Genetic Algorithm

MOP Multi-Objective Optimization Problem

MOP Multi-Objective Problem

MOPSO Multi-Objective Particle Swarm Optimisation

MSOPS-II Multiple Single Objective Pareto Sampling Algorithm 2

MSOPS Multiple Single Objective Pareto Sampling Algorithm

NNIA Nondominated Neighbour Immune Algorithm

NN Neural Network

NSGA-II Non-Dominated Sorting Genetic Algorithm 2

NSGA Non-Dominated Genetic Algorithm

PAES Pareto Archived Evolution Strategy

PBOT Population-Based Optimisation Technique

PDEA Pareto Differential Evolutionary Approach

PDE Pareto-Frontier Differential Evolution

PE Pareto Estimation method

PF Pareto Front

PSO Particle Swarm Optimization

PS Pareto Set

RM-MEDA Regularity Model-Based Multi-Objective Estimation of Distribution Algorithm

SPEA2 Strength Pareto Evolutionary Algorithm 2

SPEA Strength Pareto Evolutionary Algorithm

TSP Travelling Salesman Problem

VEDA Voronoi-Based Estimation of Distribution Algorithm

VEDE Vector Evaluated Differential Evolution

VEGA Vector Evaluated Genetic Algorithm

ZDT Zitzler, Deb, Thiele

Chapter 1

Introduction

1.1 Motivation

The focus of this work is on optimisation, specifically a subclass of optimisation methods whereby the quantity that is to be extremised is a vector and the generating mapping does not have nice properties such as convexity or differentiability. Such problems are usually referred to as nonlinear, however, this ignores a large class of problems that are nonlinear but are also convex, therefore their solution can be obtained efficiently and reliably. Therefore, in this work, problems are classified in to two major categories: (i) convex, and, (ii) nonconvex. A further complication is met, when the function that is to be extremised, presumably a function that describes in quantitative terms a real-world problem, is vector valued, namely it has multiple outputs all of which need to be minimised or maximised simultaneously. These outputs, in this context, are called objectives and are usually competing and possibly incommensurable. However, from this information it is unclear why would, such a subclass of optimisation problems, be interesting to study. The answer to this is that very often real-world problems can be expressed in such a form naturally, that is - without imposing a particular form onto the problem artificially. The more complex and large the problem, the more likely it is for the number of objectives to increase.

Both these problems, namely many objectives (more than 3) and nonconvex problems have been addressed, to some extent, in the literature of multi-objective nonlinear mathematical programming and evolutionary computation. The motivation to further explore this subject, stems from the fact that there are still several unresolved issues. One such example, which is considered in this work, is the selection of weighting vectors in algorithms that employ scalarising functions. Scalarising functions decompose a multi-objective problem into a set of many single objective problems. Such methods are called decomposition-methods, and are of interest because

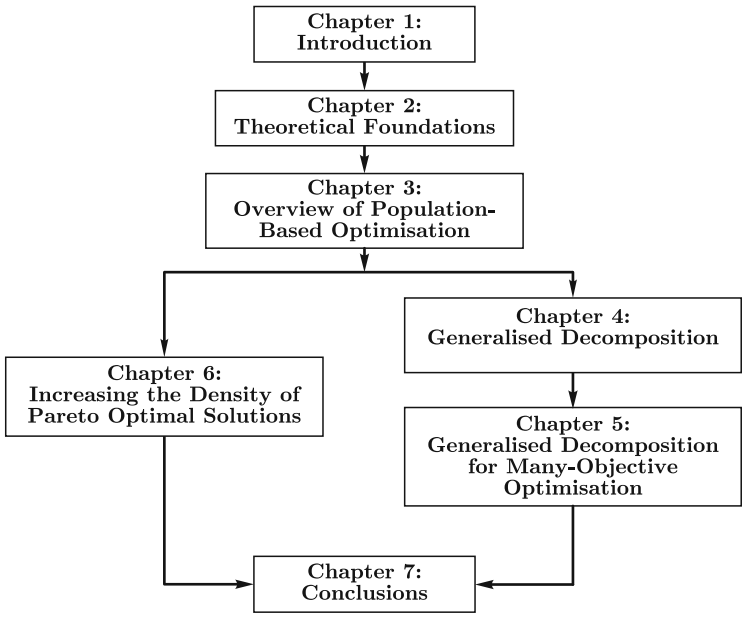

Figure 1.1: Thematic organisation of this thesis.

it has been shown in recent studies [2, 3] that they seem to be superior to their alternatives [4]. However, current decomposition-based methods cannot directly control the distribution of solutions across the *trade-off* surface[1] to the extent that is desirable by the analyst or the decision maker.

1.2 Outline

This work revisits fundamental principles and methods applied in mathematical optimisation and evolutionary algorithms to deal with many-objective optimisation problems. The two main threads of research in nonlinear optimisation are metaheuristics and convex optimisation methods. Each of these optimisation divisions has something to offer in terms of increasing our ability to answer questions, and, both are found useful in practice. An overview of this thesis can be seen in Fig. (1.1).

Chapter 2 lays the necessary theoretical foundations for the following chapters. A general formulation of single and multi-objective problems is given as well as a minimal introduction to concepts of convex analysis. Additionally the optimisation problem is considered from a broader view, which is designed to help the reader gain a perspective of what is standard practice and further illuminate the underlying motivation for exploring nonconvex many-objective problems.

[1]The term *trade-off* surface which is equivalent to Pareto front is defined in Section 2.5.2.

Chapter 3 presents an overview of the most prominent population-based algorithms and the methodologies used to extend them to multiple objective problems. Although not exact in the mathematical sense, it has long been recognised that population-based multi-objective optimisation techniques for nonconvex real-world problems are immensely valuable and versatile. These techniques are usually employed when exact optimisation methods are not easily applicable or simply when, due to sheer complexity, such techniques could potentially be very costly. Another advantage is that since a population of decision vectors is considered in each generation these algorithms are implicitly parallelisable and can generate an approximation of the entire Pareto front at each iteration. A critique of their capabilities is also provided.

In Chapter 4 a novel concept is presented, namely *Generalised Decomposition*. It is shown that using this approach to extend single objective optimisation algorithms, an extremely high precision can be achieved regarding the distribution of Pareto optimal points on the trade-off surface. Additionally, it is shown that Generalised Decomposition scales much more gracefully to many objectives compared with alternative methods. The importance of such a method is pronounced considering that recent research results show that Pareto-based algorithms cannot scale to many-objectives well and that their ability to converge to the Pareto front is progressively diminished in higher dimensions.

Chapter 5 explores the potential and benefits of applying Generalised Decomposition combined with a low-order statistics based estimation of distribution algorithm - the Cross Entropy method - to tackle many-objective problems. The aim of this chapter is twofold - (i) investigate strengths and potential weaknesses of Generalised Decomposition, and, (ii) obtain additional evidence regarding the hypothesis that estimation of distribution algorithms, based on low-order statistics, can have comparable performance with more elaborate algorithms. Furthermore, it is shown that nonlinear constraints in decision space can be handled gracefully with the proposed framework.

In Chapter 6 a method is presented that, when applied to a Pareto set approximation produced by any multi-objective algorithm, can generate more Pareto optimal solutions across the entire Pareto front or in specific locations. This method is called Pareto Estimation. The theoretical and empirical basis for Pareto Estimation is the observation that the Pareto set in decision space can, under certain regularity conditions, be described as a $(k-1)$-dimensional piecewise continuous manifold, where k is the number of objectives. This suggests that the mapping from objective space to decision space should be identifiable, given that the Pareto

set approximation is close to Pareto optimal. The Pareto Estimation method is tested against a set of difficult test problems resulting in an impressive increase in Pareto optimal solutions across the entire Pareto front. Additionally it is shown how Pareto Estimation can be used to identify Pareto optimal solutions in specific regions of the Pareto front which is demonstrated on a 3-objective portfolio optimisation problem.

In Chapter 7 future research perspectives, insight, and potential extensions of the introduced methods and concepts are discussed and this work is summarised and concluded.

1.3 Contributions

The main contributions of this work are:

- An overview of population-based optimisation methods. The overview considers a broad class of population-based metaheuristics, including Genetic Algorithms, Evolution Strategies, Differential Evolution, Particle Swarm Optimisation, Artificial Immune Systems and Estimation of Distribution algorithms. The way such techniques are applied to multi-objective optimisation problems is explored, and the most prominent methodologies are described in more depth. Lastly a critique of their performance for different problem types is discussed and summarised in a comparative matrix. This contribution is based on Chapter 3 of this thesis and has been submitted to a journal for publication.

- A novel concept has been introduced - Generalised Decomposition. This concept is applicable to decomposition based algorithms in general, namely to evolutionary algorithms or the more traditional gradient search methods. Generalised Decomposition provides the means to calculate the weighting vectors in decomposition methods, so that the distribution of Pareto optimal solutions is *optimal*, given a definition of what is considered optimal for the given context and there exists a metric which can measure the quality of the produced set according to the aforementioned definition of optimality. This contribution is based on Chapter 4 of this thesis and has been submitted to an international conference for publication.

- Generalised Decomposition and Cross Entropy methods for many-objective optimisation. Another major strength of Generalised Decomposition is that it can maintain its favourable properties for any number of objectives. This, combined with the fact that currently employed methods for selecting the weighting vectors in decomposition-based algorithms

are fundamentally flawed and do not scale well for many objective problems, motivated the creation of an algorithm based on Generalised Decomposition. The main algorithm used for this task was the Cross Entropy method (CE). The CE method can be classified as an estimation of distribution algorithm, a choice that enabled the exploration of a recently introduced hypothesis, namely, that estimation of distribution algorithms based on low-order statistics are a viable alternative to algorithms that use elaborate probabilistic models that may be very expensive to *train*. The result was the - Many Objective Cross Entropy with Generalised Decomposition (MACE-gD). The performance of MACE-gD compared with two other prominent algorithms (RM-MEDA and MOEA/D), is shown to be competitive. This contribution is based on Chapter 5 of this thesis and has been submitted to a journal for publication.

- Increasing the density of available Pareto optimal solutions. At the end of a multi-objective optimisation an approximation of the Pareto optimal set is returned. Given the Pareto set approximation, the question is: Is there salient information within that set, which can be used to generate more Pareto optimal solutions, and if so, could the produced solutions be generated in a particular region of the Pareto front that is of interest to the decision maker? In this work, a positive answer to this question is presented. Namely, a method has been developed with which the analyst is enabled to produce more Pareto optimal solutions in a specific region of interest on the Pareto front, subject to certain regularity conditions. Lastly, the utility of the proposed method is illustrated on a 3-objective portfolio optimisation problem. This contribution is based on Chapter 6 and has been submitted to a journal for publication.

In summary, the two main themes that this thesis explores are:

- The impact of weighting vectors on the distribution of Pareto optimal solutions on the Pareto front.

- Given an approximation of the Pareto front, as generated by any multi-objective optimisation evolutionary algorithm, does this approximation contain information that can be utilised to further explore the Pareto front.

Chapter 2

Theoretical Foundations

2.1 Introduction

A traditional classification of optimisation problems has been their separation into linear and nonlinear problems [5]. However, this ignores a large body of research in convex programming, which is a special class of methods that apply to a subclass of nonlinear optimisation problems, for which a solution can be obtained with relative ease, even for very large problems [6]. Therefore a more relevant distinction of optimisation problems would be their classification based on convexity, see Fig. (2.1). This is so because this distinction will radically impact the approach used in solving such problems and the expected quality of the identified solutions. In the fortunate case that a problem can be expressed in a convex form, then for all practical purposes it can almost be considered as solved. Furthermore the solution obtained for convex problems is guaranteed to be optimal, however it is not necessarily unique. Namely, a solution obtained for a convex program is the global minimum (maximum) for a minimisation (maximisation) problem. Also such a solution can be obtained quite efficiently [6]. These facts strongly motivate the additional effort required to attempt to identify whether a problem is convex. However the process of identifying a convex problem is non-trivial and is further complicated by the fact that different formulations can be more difficult to solve [6]. A constructive approach introduced in [7] is to attempt and recreate an *equivalent* formulation of the original problem. The authors introduce a set of composition rules. When these are adhered to, only convex problems are created. So the challenge is to reformulate the original problem using only the advocated set of rules in [7]. However, if an equivalent convex problem cannot be found, this does not necessarily mean that the original problem is nonconvex. Therefore this methodology is applicable only to a subset of all convex problems and depends heavily on the imagination of the user. Although

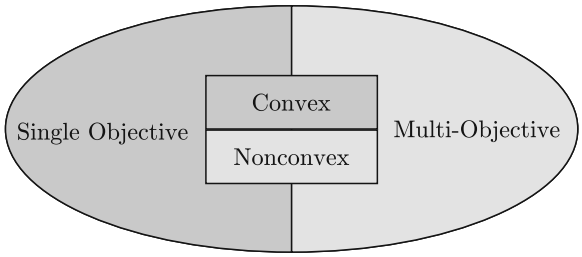

Figure 2.1: Overview of optimisation problems. **Light grey:** The focus of this work, namely nonconvex and multi-objective problems. **Dark grey:** Convex optimisation for single-objective problems is also considered in this work, however, it is used mainly as support to the main theme.

what is proposed in [7] does not solve all ensuing difficulties in the formulation of a convex problem, it is a very interesting development which can be of great aid to practitioners and has the potential to be generalised to a wider family of functions.

For nonconvex problems, guarantees about the obtained solution can only be given when an exhaustive search is performed. That is, only if the entire domain of definition of the objective function is explored. Naturally such a task can very easily become unmanageable and is often impossible to perform. However once the fact that a problem is nonconvex is established, there are several metaheuristics[1] that can be employed to obtain a solution. Although such a solution will most likely be suboptimal, metaheuristics perform very well in practice. Such methods are reviewed in Chapter 3 and are an integral part of the tool-set available to the practitioner, since, as it will become clear, many problems are nonconvex and even convex problems will turn into nonconvex at the slightest *provocation*. An example of this phenomenon is seen in machine learning, where kernel-based learning algorithms that have a *shallow* architecture, namely a single layer of kernels for which the weights are to be determined, are linear in the parameters and produce convex problems. However, such architectures seem to be inefficient for certain tasks while *deep* architectures, namely architectures with several layers of kernels, can learn more complex tasks but, the estimation of their parameters (learning) is nonconvex [8]. Such examples serve as feedback to practitioners not to become overly dependent on a particular method, but instead, carefully investigate the nature of the problem so as to select the *optimal* approach for its solution, a process that is nonconvex in itself.

Another important classification is the separation of optimisation problems into single objective, that is problems where the objective function is a mapping of the type $f : \mathbb{R}^n \to \mathbb{R}$, and,

[1] An algorithmic framework of heuristic optimisation algorithms. Heuristic is the Greek word for *search*. Metaheuristic algorithms commonly have a stochastic (Greek word for *random*) component.

multi-objective whereby the objective function mapping has the following form $f : \mathbb{R}^n \to \mathbb{R}^k$, see Fig. (2.1). This classification is important because in the latter case, there is no obvious or unique way to induce a complete ordering, see Appendix A.1.2. Without order a direction of search cannot be established. This is a well known issue in multi-objective optimisation and has been addressed to varying degrees of success by several researchers in the field of mathematical programming [9, 10, 11] and more recently in evolutionary computation [12, 13, 14]. In general there are two approaches employed to resolve this issue: Pareto-based and decomposition-based methods. In both methodologies and assuming, the *a posteriori* preference articulation paradigm (see Section 2.6), the relative importance of the objectives is considered to be unknown. In the case that preference information is given by the decision maker (DM), then using a decomposition method to combine the scalar objective functions can be used, see Section 2.5.3 and Section 4.3. An alternative is to distill the preference information given by the decision maker in a utility function [11, pp. 62]. Pareto-based methods use the Pareto-dominance relations [5] to induce partial ordering in the objective space. These relations initially introduced by Edgeworth [15] and further expanded by Pareto [16] are similar in nature with element-wise vector comparison. For example for two vectors $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$, a binary relation $<$ can be defined to mean, $\mathbf{a} < \mathbf{b}$ if all the elements in $\mathbf{a}$ are smaller than the corresponding elements in $\mathbf{b}$. This partial ordering is denoted as $\mathbf{a} \prec \mathbf{b}$, and, in the context of a minimisation problem this expression is read as: the vector $\mathbf{a}$ dominates $\mathbf{b}$. This type of ordering and its implications are further explored in Section 2.5.2, and basic concepts from set theory are explained in Appendix A.1.2.

Another way to induce partial ordering in multi-objective problems, that is predominantly used in mathematical programming [5], is by decomposition methods. These methods, use scalarising functions to decompose a multi-objective problem into several single objective subproblems. These subproblems are defined with the help of weighting vectors. The weighting vectors are k-dimensional vectors with positive components that sum to one. The *location* on the Pareto front that each subproblem tends to converge, strongly depends on the choice of weighting vectors (see Chapter 4). Therefore, the selection of an appropriate set of weighting vectors to *decompose* the multi-objective problem, determines the distribution of the final Pareto set approximation along the Pareto front. Although a clear definition of what is considered a good distribution of Pareto optimal solutions does not exist, there is a consensus about the features that must be present. First, assuming that a decision maker is not involved prior or during the optimisation process, the general tendency is to distribute the Pareto set approximation along

the entire Pareto front. A second implicit requirement is that Pareto optimal solutions are distributed evenly across the entire front. This emanates from the fact that the preference of the decision maker towards a particular region of the trade off surface is unspecified or unknown. Finally, the distance of the Pareto set approximation must be as close as possible to the true Pareto front. Convergence of the optimization algorithm is measured in terms of that distance.

The remainder of this chapter is organised as follows. In Section 2.2, fundamental concepts about convex sets and functions are presented. Section 2.3 introduces the epigraph of a function, which is a link between convex functions and convex sets. These two sections are essential for Chapters 4 and 5. In Section 2.4 and Section 2.5, single and multi-objective optimisation problems are defined. Lastly, in Section 2.6 an overview of the multi-objective optimisation process, from problem formulation to the selection of a single solution is presented.

2.2 Convex Sets and Functions

A set, $C \subseteq \mathbb{R}^n$, is a *subspace* if it is closed under scalar multiplication and vector addition, namely if,

$$\text{for all } \mathbf{x}, \mathbf{y} \in C,\ a\mathbf{x} + b\mathbf{y} \in C \text{ with } a, b \in \mathbb{R}. \tag{2.1}$$

An *affine set* is a subspace plus a shift, that is, if C is a subspace, then,

$$V = \{\mathbf{x} + \mathbf{b} : \mathbf{x} \in C\},\ \text{for any } \mathbf{b} \in \mathbb{R}^n, \tag{2.2}$$

is an affine set. Naturally the dimension of the vectors in the set C must be equal to the dimension of the vector $\mathbf{b}$, that is n. An example of a subspace in $\mathbb{R}^2$ and an affine set is shown in Fig. (2.2). A line through the origin is a one dimensional subspace of $\mathbb{R}^2$ this line plus a shift forms an affine set. A different view of an affine set, one that elucidates the connection of an affine set with a convex set, is obtained using two points $\mathbf{x}, \mathbf{y} \in V$, then,

$$\mathbf{y} + \theta(\mathbf{x} - \mathbf{y}) = \theta\mathbf{x} + (1 - \theta)\mathbf{y} \in V, \tag{2.3}$$

where $\theta \in \mathbb{R}$. The description of an affine set in (2.3) can be extended by induction to any number of points in V and scalars $\theta_1, \ldots, \theta_d \in \mathbb{R}$ as,

$$\sum_{i=1}^{d} \theta_i \mathbf{x}_i,\ \text{subject to } \sum_{i=1}^{d} \theta_i = 1, \tag{2.4}$$

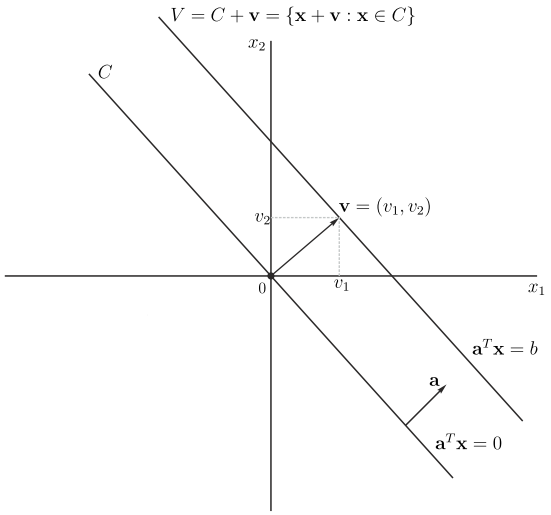

Figure 2.2: A subspace in $\mathbb{R}^2$, C, and an affine set, V. Note that the vector $\mathbf{v}$ is by no means unique, in fact, by subtracting any vector that is in the affine set, V, the original subspace C will be obtained.

this is referred to as an *affine combination*. The *affine hull* of a set V is defined as the set of all affine combinations,

$$\mathbf{aff}\, V = \left\{ \sum_{i=1}^{d} \theta_i \mathbf{x}_i : \mathbf{x}_i \in V, \sum_{i=1}^{d} \theta_i = 1,\ i = 1, \ldots, d \right\}. \tag{2.5}$$

A set $C \subseteq \mathbb{R}^n$ is *convex* if for any $\mathbf{x}, \mathbf{y} \in C$ and any $\theta \in [0, 1]$,

$$\theta\mathbf{x} + (1 - \theta)\mathbf{y} \in C. \tag{2.6}$$

By definition an affine set is also a convex set, however a convex set is not necessarily an affine set. The combination of the points $\mathbf{x}, \mathbf{y}$ in (2.6), is called a *convex combination* and can be extended to multiple points in a similar manner to the extension of affine combinations,

$$\sum_{i=1}^{d} \theta_i \mathbf{x}_i, \text{ with } \sum_{i=1}^{d} \theta_i = 1, \text{ and } \theta_i \geq 0, \text{ for all } i = 1, \ldots, d. \tag{2.7}$$

Some convex sets are illustrated in Fig. (2.3). The set of all convex combinations of a convex set C is the *convex hull* of that set and is defined as,

$$\mathbf{conv}\, C = \left\{ \sum_{i=1}^{d} \theta_i \mathbf{x}_i : \mathbf{x}_i \in C, \sum_{i=1}^{d} \theta_i = 1, \theta_i \geq 0, i = 1, \ldots, d \right\}, \tag{2.8}$$

see Fig. (2.4).

A set C is a *cone* if $\theta\mathbf{x} \in C$ for all $\mathbf{x} \in C$ and $\theta \geq 0$. A cone need not necessarily be a convex set, for instance the cone shown in Fig. (2.5) is nonconvex. To see this, notice that a

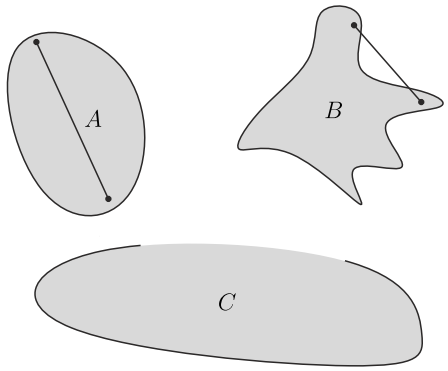

Figure 2.3: A,B and C are three sets of which only the set A is convex since the set B does not contain all convex combinations of its points, namely not all points on the line segment connecting the two points in B are in the set. The set C fails to be convex because part of its boundary (thick line about the set) is not included in the set.

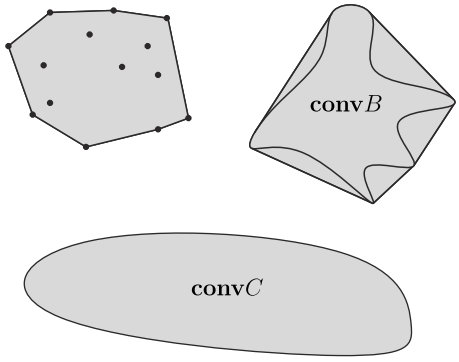

Figure 2.4: Convex hull of a set of points (**top left**) and the sets B and C in Fig. (2.3).

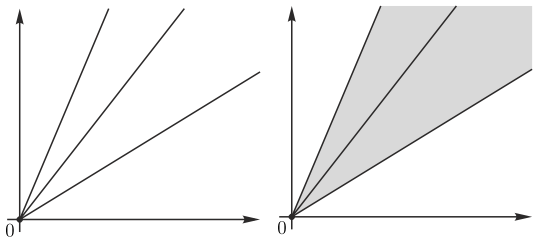

Figure 2.5: **Left:** The set of all points on the rays emanating from the origin is a cone. **Right:** The conic hull of the cone depicted in the left figure.

convex combination of a point on the *lower* ray with a point on the *higher* ray can produce infinitely many points that are not within the set, i.e. not on one of the two rays, therefore this is not a convex set. However if a set C that is a cone, is also a convex set, then the set C is called a *convex cone*. A convex cone C contains all the *conic combinations* in the set. A conic combination is a nonnegative combination of the elements in the set C. The *conic hull* of a set C is all conic combinations of the elements in the set. The conic hull is denoted as, $\mathbf{cone}\, C$, and is defined as,

$$\mathbf{cone}\, C = \left\{ \sum_{i=1}^{d} \theta_i \mathbf{x}_i : \mathbf{x}_i \in C, \theta_i \geq 0, i = 1, \ldots, d \right\}. \tag{2.9}$$

A cone $C \subset \mathbb{R}^n$ is called a *proper cone* if it is convex, closed[1], *pointed* and has nonempty interior. A cone is pointed if it contains no line, for example the cone, $C = \{(x, f(x)) \in \mathbb{R}^2 : f(x) > 0\}$, is not pointed since it contains an infinite number of lines: $f(x) = c, \forall x \in \mathbb{R}$, for any $c \in \mathbb{R}_+$. An example of a pointed cone is the nonnegative orthant $\mathbb{R}^n_+$, which is also a proper cone. Proper cones play a significant role in inducing partial ordering and are strongly related to the concept of Pareto optimality, see Section 2.5.2.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is said to be convex if its domain of definition, $\mathbf{dom}\, f$, is a convex set and $\forall\, \mathbf{x}, \mathbf{y} \in \mathbf{dom}\, f$ and $\theta \in [0, 1]$ we have,

$$f(\theta \mathbf{x} + (1 - \theta)\mathbf{y}) \leq \theta f(\mathbf{x}) + (1 - \theta) f(\mathbf{y}). \tag{2.10}$$

A function is strictly convex if the inequality in (2.10) is strict. Accordingly a function is concave if $-f$ is convex. A more interesting definition of convex and concave functions is formulated with the aid of the *epigraph* of a function, see Section 2.3.

2.3 Epigraph

The *epigraph* of a function $f : \mathbb{R}^n \to \mathbb{R}$, which is the Greek word for *above the graph*, is defined as,

$$\mathbf{epi}\, f = \{(\mathbf{x}, t) : \mathbf{x} \in \mathbf{dom}\, f, t \in \mathbb{R}, f(\mathbf{x}) \leq t\}, \tag{2.11}$$

consequently $\mathbf{epi}\, f \subset \mathbb{R}^{n+1}$. If the epigraph of a function is a convex set then the function is convex and vice versa. The *hypograph* of a function $f : \mathbb{R}^n \to \mathbb{R}$, meaning *below the graph*, is defined as,

$$\mathbf{hypo}\, f = \{(\mathbf{x}, t) : \mathbf{x} \in \mathbf{dom}\, f, t \in \mathbb{R}, f(\mathbf{x}) \geq t\}. \tag{2.12}$$

[1] See Appendix A.2 for the definition of a closed set.

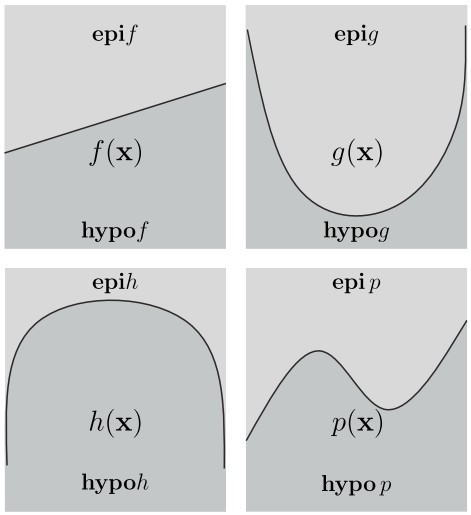

Figure 2.6: **Top left:** An affine function $f(\mathbf{x})$, affine functions are both convex and concave. **Bottom left:** A concave function $h(\mathbf{x})$. **Top right:** A convex function $g(\mathbf{x})$. **Bottom right:** A nonconvex function $p(\mathbf{x})$.

The epigraph and hypograph allow for the definition of convex functions to be expressed in terms of convex sets and in the present work are used to define the *shapes* of the Pareto front, see Section 2.5.1. If a function is concave, its hypograph is a convex set. In general a function $f : \mathbb{R}^n \to \mathbb{R}$ with a convex domain of definition is:

- Convex, if and only if **epi** f is a convex set. If in addition **hypo** f is nonconvex then, f is strictly convex.
- Concave, if and only if **hypo** f is a convex set. If in addition **epi** f is nonconvex then, f is strictly concave.
- Convex and concave, if both **epi** f and **hypo** f are convex. A concave and convex function is affine.
- Nonconvex, if both **epi** f and **hypo** f are nonconvex.

An illustration of the above definitions is shown in Fig. (2.6). Nonconvex functions can exhibit one or more of the following *pathologies*:

- Multi-modal functions have more than one extremum. An example of such a function is $f(x) = \sin(x)$ with domain of definition **dom** $f \in [0, 4\pi]$. Multi-modal functions can have many or infinitely many global extrema. For instance, the following function has infinitely

many global minima,

$$f(x) = \begin{cases} -1 & \text{if } x \in \left[\frac{\pi(12k+7)}{6}, \frac{\pi(12k+11)}{6}\right], k \in \mathbb{Z} \\ \sin(x) & \text{otherwise.} \end{cases} \tag{2.13}$$

- Discontinuous functions are nonconvex because their domain of definition is nonconvex. In general there are two types of discontinuities: (i) the left and right limit of the function about the discontinuity is the same, and (ii) the left and right limits are different. The latter is called a *jump* discontinuity.

- Functions that are a combination of convex functions and nonconvex functions. For example, a function that is affine everywhere except for a small part of its domain, is nonconvex. Such functions are usually employed in test problems as they can be deceptive.

The objective functions studied in this work are functions that are defined over an *uncountable* set, namely an infinite set.

2.4 Single-Objective Optimisation

2.4.1 Problem Setting

A single objective optimisation problem is defined as,

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x}) \\ \text{subject to } &\begin{cases} g_i(\mathbf{x}) \leq 0 & i = 1, \ldots, m \\ h_i(\mathbf{x}) = 0 & i = 1, \ldots, d \\ \mathbf{x} \in \mathbf{dom}\, f & \mathbf{dom}\, f \subset \mathbb{R}^n \end{cases} \end{aligned} \tag{2.14}$$

where m and d are the number of inequality and equality constraints respectively, n is the size of the decision vector $\mathbf{x}$ and $\mathbf{dom}\, f$ is the domain of definition of the objective function. The type of the functions f, g and h as well as the domain of f, are the key factors that determine the type of the problem defined by (2.14). For example when the objective function is convex and is defined over a convex domain, and, the inequality constraints are convex functions while the equality constraints are affine, then (2.14) is a convex optimisation problem [6]. An optimisation problem defined by (2.14) is considered to be solved when a decision vector $\tilde{\mathbf{x}} \in \mathbf{dom}\, f$ is found that,

$$\begin{aligned} &f(\tilde{\mathbf{x}}) \leq f(\mathbf{x}), \text{ for all } \mathbf{x} \in \mathbf{dom}\, f, \\ \text{and } &\begin{cases} g_i(\tilde{\mathbf{x}}) \leq 0 & i = 1, \ldots, m \\ h_i(\tilde{\mathbf{x}}) = 0 & i = 1, \ldots, d\,. \end{cases} \end{aligned} \tag{2.15}$$

Such a decision vector, $\tilde{\mathbf{x}}$, is said to be a *global minimum*. A local minimum is defined as,

$$f(\tilde{\mathbf{x}}) \leq f(\mathbf{x}), \text{ for all } \mathbf{x} \in \{\mathbf{x} : \|\mathbf{x} - \tilde{\mathbf{x}}\|_2 \leq r, r \in \mathbb{R}_+, \tilde{\mathbf{x}} \in \mathbf{dom}\, f\},$$
$$\text{and } \begin{cases} g_i(\tilde{\mathbf{x}}) \leq 0 & i = 1, \ldots, m \\ h_i(\tilde{\mathbf{x}}) = 0 & i = 1, \ldots, d\,, \end{cases} \tag{2.16}$$

and is usually what can be found for nonconvex problems, since, as mentioned in the introduction, it is usually impracticable to search the entire feasible set so as to ensure that the global minimum is found.

A problem is called unconstrained if the equality and inequality constraints in (2.14) are removed. Strictly speaking there remains the constraint that $\mathbf{x} \in \mathbf{dom}\, f$ but this could be overlooked since the domain of the objective function can be easily extended to $\mathbb{R}^n$ in the following way,

$$\tilde{f}(\mathbf{x}) = \begin{cases} f(\mathbf{x}) & \mathbf{x} \in \mathbf{dom}\, f, \\ \infty & \mathbf{x} \notin \mathbf{dom}\, f. \end{cases} \tag{2.17}$$

The function $\tilde{f}$ is called the extended value function, and is simply defined to be infinite outside the domain of definition of f. Interestingly, this *trick* can be applied to (2.14) as well, but this can make the original problem more difficult to solve since the information about the constraints is hidden by the extension, therefore, the identification of the problem as convex or not becomes virtually impossible if the *internal* description of the extended value function is not available.

Another useful device is the definition of the *feasible set* in decision space, namely,

$$S = \{\mathbf{x} : \mathbf{x} \in \mathbf{dom}\, f, g_i(\tilde{\mathbf{x}}) \leq 0, i = 1, \ldots, m,\ h_i(\tilde{\mathbf{x}}) = 0,\ i = 1, \ldots, d\,\}, \tag{2.18}$$

which is also called, the *feasible region*. With the help of (2.18), (2.14) can be rewritten in a more compact form,

$$\begin{aligned} &\min_{\mathbf{x}} f(\mathbf{x}) \\ &\text{subject to } \mathbf{x} \in S, \end{aligned} \tag{2.19}$$

which is a convention that is used in the remainder of this thesis.

2.4.2 Optimality Condition - Single Objective Problems

There are several prevalent optimality conditions, for instance the Fritz John type conditions [5, pp. 37] or the Karush-Kuhn-Tucker condition [5, pp. 39]. Such conditions have extensions to address non-differentiability and multiple objectives [5, pp. 45]. However, these conditions are necessary but not sufficient for nonconvex optimisation problems. The optimality condition that is employed in this work has been selected due to its intuitive geometric interpretation.

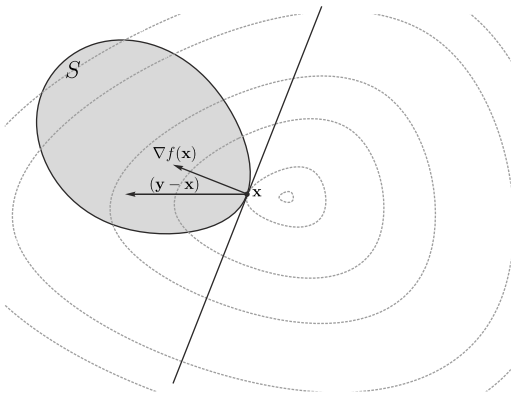

Figure 2.7: S is the feasible set in decision space and the level sets represent the objective function values $f(\mathbf{x}) = c$ where c is a constant and in this example $\mathbf{x} \in \mathbb{R}^2$, namely $f(\mathbf{x}) : \mathbb{R}^2 \to \mathbb{R}$. In this example the feasible set, S, is convex. A convex feasible set cannot *occupy* both sides of the halfspace defined by $\nabla f(\mathbf{x})$ and at the same time satisfy (2.20).

However, as it is applicable only to differentiable problems, it is used as an illustration device rather than an actual test for optimality.

For convex, differentiable and single objective problems, a decision vector, $\mathbf{x}$, is optimal if,

$$\begin{aligned} \nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \geq 0, \\ \forall \mathbf{y} \in S, \end{aligned} \tag{2.20}$$

a proof can be found in [6, pp. 139]. This definition of optimality, can be readily extended to multi-objective optimisation problems, see Section 2.5.4. The statement in (2.20), is that the inner product of the gradient of the objective function with the vector, $(\mathbf{y}-\mathbf{x})$, must be non-negative, namely their angle must be less than $\frac{\pi}{2}$, see Fig. (2.7). Therefore, since the gradient vector of a function has the direction of steepest ascent, in function value with respect to the decision vector, then the last statement simply means that all decision vectors, $\mathbf{y} \in S$, lie in the direction that the value of the function is increasing, hence $\mathbf{x}$ is optimal.

For nonconvex differentiable objective functions, the definition can be *localised* by using a sufficiently small, ϵ, to a part of the feasible region that is locally convex,

$$\begin{aligned} \nabla f(\mathbf{x})^T(\mathbf{y}-\mathbf{x}) \geq 0, \\ \forall \mathbf{y} = \{\mathbf{y} : \|\mathbf{y}-\mathbf{x}\| \leq \epsilon, \mathbf{y} \in S, \epsilon \in \mathbb{R}\}. \end{aligned} \tag{2.21}$$

Although (2.21) is perhaps of limited utility, it shows that, local optima can be identified, and therefore the geometric interpretation is still valid, albeit in a more restricted form. This concept can be extended to non-differentiable functions using subdifferentials in a fashion similar to the way the KKT conditions are extended, see [5, pp. 49].

2.5 Multi-Objective Optimisation

When the objective function is vector valued, that is $\mathbf{F} : \mathbb{R}^n \to \mathbb{R}^k$ then the optimisation problem becomes more complex. This complexity stems from the fact that now there exists the possibility that there is no single objective function value $\mathbf{F}(\tilde{\mathbf{x}}) \leq \mathbf{F}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{R}^n$, as was the case for single objective problems. Therefore, in all but the most trivial case where all the scalar objective functions are *harmonious*, namely when all the objectives are positively correlated [17], only a partial ordering can be induced without the preference structure of the decision maker. Some fundamental methods inducing such a partial order are discussed in Section 2.5.2 and Section 2.5.3. However, a complete ordering is required since only one solution can be usually implemented. For this reason the decision maker plays an integral part in the solution of a multi-objective optimisation problem. In Section 2.6 the entire optimisation process, from problem formulation to solution selection is discussed as well as common paradigms of interaction between the analyst and the decision maker.

2.5.1 Problem Setting

A multi-objective minimisation problem can be defined as follows:

$$\begin{gathered} \min_{\mathbf{x}} \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_k(\mathbf{x})) \\ \text{subject to } \mathbf{x} \in S, \end{gathered} \tag{2.22}$$

where k is the number of objective functions, S, is the feasible set in decision space, $\mathbf{x}$ is the decision vector and, $f_i(\mathbf{x})$, are scalar objective functions. Let $\mathbf{F} : S \to Z$, namely the forward image[1] of the objective function, then the set, Z, is the feasible set in objective space. The $\min_{\mathbf{x}} \mathbf{F}(\mathbf{x})$ notation is interpreted as: minimise the vector valued function $\mathbf{F}(\mathbf{x})$ over all $\mathbf{x} \in S$ and should not be confused with the min operator which returns the minimum element of a set. It should also be noted that this definition could be used to describe a constrained or unconstrained minimisation problem depending on how S is defined, see (2.19). Further the fact that minimisation is assumed is not restrictive because the problem of maximising $-f$ is equivalent to the problem of minimising f and vice versa. In the special case where $k = 1$, (2.22) becomes a single objective minimisation problem. It is implicitly assumed that the scalar objective functions are mutually competing and perhaps are incommensurable while the goal is to minimise all of them simultaneously. If this is not the case then no special treatment is needed

[1] See Appendix A.3.

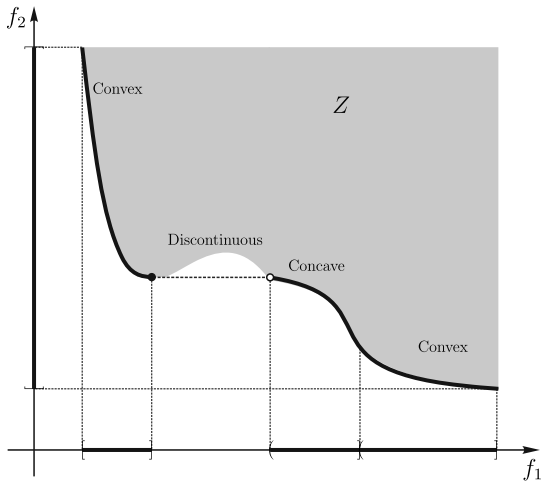

Figure 2.8: A Pareto front which is partially convex, partially concave and discontinuous. Notice that the frame of reference, which in this case is f_1, used to determine the convex and concave parts is arbitrary, namely the same parts of the Pareto front would be partially convex and concave, even if f_2 was chosen as the reference. However, discontinuities on the PF are not always *visible* from all frames of reference, i.e. the projection of the PF on the f_2 axis is continuous, while the projection on the f_1 axis is discontinuous.

since minimising one of the scalar objective functions automatically results in minimisation of the rest.

2.5.2 Partial Ordering - Pareto-Based Approach

The problem that arises in MOPs is that direct comparison of two objective vectors is not as straightforward as in single objective problems. In single objective problems when both $\mathbf{x}, \tilde{\mathbf{x}} \in S$ and $f(\tilde{\mathbf{x}}) < f(\mathbf{x})$ it is clear[1] that the decision vector $\tilde{\mathbf{x}}$ is superior to $\mathbf{x}$. This is not the case when two or more objectives are considered simultaneously and there exists no *a priori* preference toward a particular objective.

If the relative importance of the objectives is unspecified, one way to partially order the objective vectors, $\mathbf{z} \in Z$, is to use the Pareto[2] dominance relations, initially introduced by Edgeworth [15] and further studied by Pareto [16]. Dominance relations can be defined using generalised inequalities $(\prec, \preceq)$ and the help of a proper cone K, see Section 2.2. A commonly used cone for this is the non-negative orthant, $\mathbb{R}^k_+$. So, for $K = \mathbb{R}^k_+$ and $\mathbf{a}, \mathbf{b} \in \mathbb{R}^k$, $\mathbf{a} \prec_K \mathbf{b}$ is true when[3] $\mathbf{b} - \mathbf{a} \in \mathbf{int}\, K$, and, $\mathbf{a} \preceq_K \mathbf{b}$ when $\mathbf{b} - \mathbf{a} \in K$. However, since the non-negative orthant is

[1] For a minimisation problem.

[2] Referred to as Edgeworth-Pareto dominance relations by some authors.

[3] The notation $\mathbf{int}\, K$ is used to denote the *interior* of the set K, in this case the cone K. For further details see Appendix A.2.

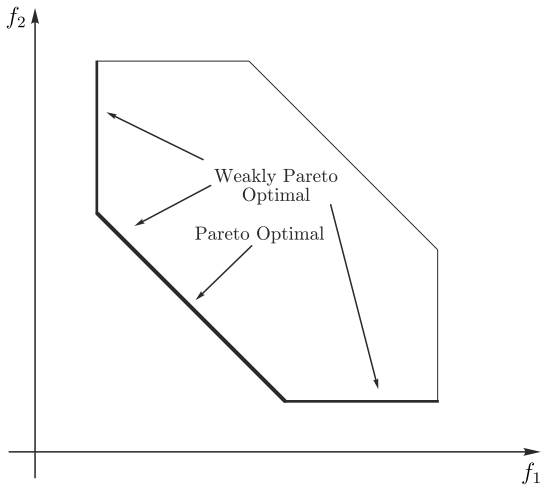

Figure 2.9: Pareto optimal set and weakly Pareto optimal set. Note that the Pareto optimal set is a subset of the weakly Pareto optimal set.

almost always used to define generalised inequalities the subscript, K, is usually omitted. This notational convention is adopted in this work, so a subscript in generalised inequalities will be used only when the proper cone, K, is other than the non-negative orthant or the meaning is unclear from the context.

Specifically, in a minimisation context, a decision vector $\tilde{\mathbf{x}} \in S$ is said to be **Pareto optimal** if there is no other decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) \leq f_i(\tilde{\mathbf{x}})$, for all i, and, $f_i(\mathbf{x}) < f_i(\tilde{\mathbf{x}})$ for at least one $i = 1, \ldots, k$. Namely there exists no other decision vector that maps to a clearly superior objective vector. Similarly, a decision vector $\tilde{\mathbf{x}} \in S$ is said to be **weakly Pareto optimal** if there is no other decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) < f_i(\tilde{\mathbf{x}})$ for all $i = 1, \ldots, k$, see Fig. (2.9). Furthermore, a decision vector $\tilde{\mathbf{x}} \in S$ is said to **Pareto-dominate** a decision vector $\mathbf{x}$ iff $f_i(\tilde{\mathbf{x}}) \leq f_i(\mathbf{x}),\ \forall i \in \{1, 2, \ldots, k\}$ and $f_i(\tilde{\mathbf{x}}) < f_i(\mathbf{x}),$ for at least one $i \in \{1, 2, \ldots, k\}$ then $\tilde{\mathbf{x}} \preceq \mathbf{x}$. So, in terms of generalised inequalities, if $\mathbf{F}(\tilde{\mathbf{x}}) \preceq \mathbf{F}(\mathbf{x})$ and $\mathbf{F}(\tilde{\mathbf{x}}) \neq \mathbf{F}(\mathbf{x})$, then $\tilde{\mathbf{x}} \preceq \mathbf{x}$. Also, a decision vector $\tilde{\mathbf{x}} \in S$ is said to **strictly dominate**, in the Pareto sense, a decision vector $\mathbf{x}$ iff $f_i(\tilde{\mathbf{x}}) < f_i(\mathbf{x}),\ \forall i \in \{1, 2, \ldots, k\}$ then $\tilde{\mathbf{x}} \prec \mathbf{x}$. That is, if $\mathbf{F}(\tilde{\mathbf{x}}) \prec \mathbf{F}(\mathbf{x})$, then $\tilde{\mathbf{x}} \prec \mathbf{x}$. It should be noted at this point, that when $\prec, \preceq$ are used in decision space, their meaning is mostly symbolic and is used to reflect the dominance relations in the objective space. For example, let $\mathbf{x}_1 = (0, 0, 0, 0), \mathbf{x}_2 = (3, 3, 3, 3)$ and $f(\mathbf{x}_1) = (4, 4), f(\mathbf{x}_2) = (1, 1)$. Clearly, for $K_s = \mathbb{R}^4_+$, $\mathbf{x}_1 \prec_{K_s} \mathbf{x}_2$, however according to the above definition of strict dominance it should be $\mathbf{x}_2 \prec \mathbf{x}_1$, because $\mathbf{F}(\mathbf{x}_2) \prec_{K_z} \mathbf{F}(\mathbf{x}_1)$ for $K_z = \mathbb{R}^2_+$. This can happen because the decision vectors are implicitly ordered according to their forward image in objective space, where the

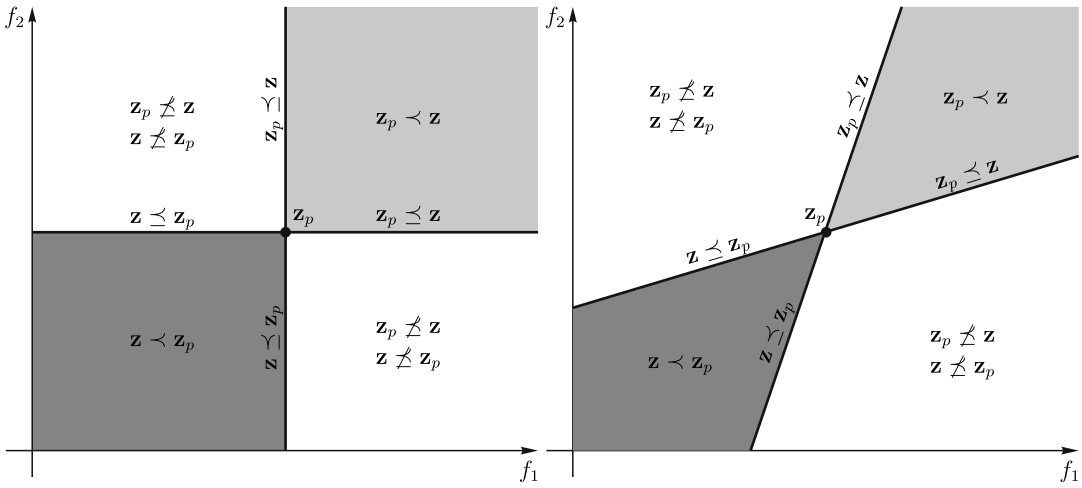

Figure 2.10: **Left:** Dominance relations defined by a cone $K = \mathbb{R}^k_+$, in this instance $K = \mathbb{R}^2_+$. **Right:** Dominance relations based on an *acute* proper cone $K = \{\mathbf{z} : \sum_{i=1}^{k} \theta_i \mathbf{z}_i, \theta_i \geq 0\}$ with $\mathbf{z}_i$ forming an acute angle with $\mathbf{z}_j$ for all $i \neq j$.

usual partial ordering, induced by the cone K_z is employed. Lastly, the ordering induced by the binary relations $\prec, \preceq$ is called *partial* because of the following possibility: $\mathbf{x}, \mathbf{y} \in Z$ but $\mathbf{x} \not\preceq \mathbf{y}$ and $\mathbf{y} \not\preceq \mathbf{x}$, in which case the vectors $\mathbf{x}, \mathbf{y}$ are said to be **incomparable**. For example, the vectors $\mathbf{x} = (3, 2, 1)$ and $\mathbf{y} = (1, 2, 3)$ are incomparable. Dominance relations induced by two different proper cones are depicted in Fig. (2.10). Pareto-dominance relations and dominance relations imposed by the *cone*[1] $K = \mathbb{R}^n_+ \setminus \mathbf{0}$ are equivalent [5, pp. 24]. Notice that the $\mathbf{0}$ element is removed from K, this means that Pareto-dominance relations are not reflexive, i.e. $\mathbf{x} \preceq_K \mathbf{x}$ does not hold as, $\mathbf{x} - \mathbf{x} = \mathbf{0} \notin K$.

Most multi-objective problem solvers attempt to identify a set of Pareto optimal solutions, this set is a subset of the **Pareto optimal set** (PS) which is also referred to as **Pareto front** (PF). The Pareto optimal set is defined as follows: $\mathcal{P} = \{\mathbf{z} : \nexists\, \tilde{\mathbf{z}} \preceq \mathbf{z}, \forall\, \tilde{\mathbf{z}} \in Z\}$, namely, it is the set of objective vectors that are not dominated by any objective vector in the feasible objective space. The decision vectors that correspond to the set, $\mathcal{P}$, are also called the Pareto set and are denoted as $\mathcal{D}$, namely $\mathbf{F} : \mathcal{D} \rightarrow \mathcal{P}$. The *geometry* of the Pareto front can significantly impact the *quality* of the produced Pareto set approximation and is further studied in Chapters 3-6. In general, the geometry of the PF can be: (i) *partially* concave, (ii) *partially* convex, (iii) discontinuous, and any *combination* thereof. For instance, the PF shown in Fig. (2.8), is discontinuous, partially convex and partially concave. Let the Pareto front be represented by a

[1]Note that according to the definition of a cone, see Section 2.2, the set K is not actually a cone, as it does not contain the $\mathbf{0}$ element.

piecewise continuous function, $g : \mathbb{R}^{k-1} \to \mathbb{R}$, where k is the number of objectives. With the aid of the function, g, the geometry of the Pareto front is defined as:

- Convex, if $\mathbf{epi}\, g$ is a convex set.
- Concave, if $\mathbf{hypo}\, g$ is a convex set.
- Affine, if both $\mathbf{epi}\, g$ and $\mathbf{hypo}\, g$ are convex.
- Discontinuous, if $\mathbf{dom}\, g$ is nonconvex or g is discontinuous.
- Partially convex, if g is convex over a convex subset of $\mathbf{dom}\, g$.
- Partially concave, if g is concave over a convex subset of $\mathbf{dom}\, g$.
- Partially affine, if g is convex and concave over a convex subset of $\mathbf{dom}\, g$.
- Piecewise convex, if g partially convex over all convex subsets of $\mathbf{dom}\, g$.
- Piecewise concave, if g partially concave over all convex subsets of $\mathbf{dom}\, g$.
- Piecewise affine, if g partially affine over all convex subsets of $\mathbf{dom}\, g$.

In mathematical programming, the Pareto dominance relations are mainly used for theoretical purposes. However, in evolutionary computation they are heavily used in fitness assignment. Fitness assignment has a similar function to the negative gradient in gradient search - it indicates a promising direction of search. In Chapter 3 a more in-depth view of how Pareto dominance relations have been employed in population-based optimisation methods is presented.

Pareto dominance based methods for fitness assignment have several difficulties to overcome, for instance a well distributed Pareto front is not guaranteed simply by using dominance relations. An answer to this problem has been presented in [18], where the authors introduce ε-dominance. In essence, ε-dominance creates a pseudo partial ordering based on the set $K_\varepsilon = \mathbb{R}^k_+ + \varepsilon$. In fact K_ε is not a cone as it does not contain the $\mathbf{0}$ vector (see the definition of a cone in Section 2.2). The reason that ε-dominance can only define a pseudo partial ordering is because it fails all axioms that a binary relation should satisfy to be a partial ordering, see Appendix A.1.2. For example, for $K_\varepsilon = \mathbb{R}^2_+ + \varepsilon$ and $\varepsilon \in \mathbb{R}^2_{++}$, the binary relation $\preceq_{K_\varepsilon}$ can be defined. However, $\preceq_{K_\varepsilon}$ is not reflexive, i.e. $\mathbf{x} \preceq_{K_\varepsilon} \mathbf{x}$ does not hold as $\mathbf{x} - \mathbf{x} = 0 \notin K$ and similarly for the other two properties shown in Section 2.2. Nevertheless ε-dominance is

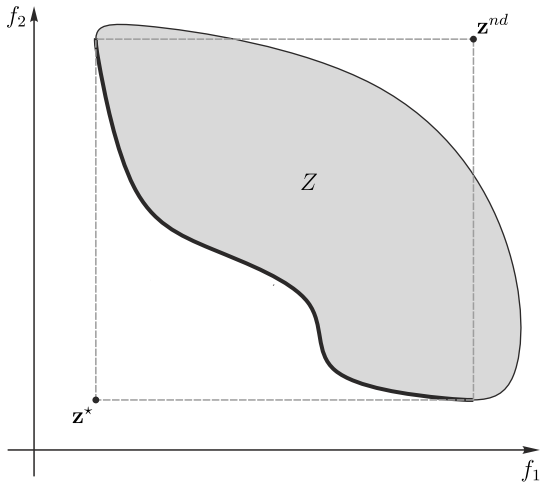

Figure 2.11: The ideal objective vector, $\mathbf{z}^{\star}$, and the nadir objective vector, $\mathbf{z}^{nd}$.

useful to maintain well distributed Pareto optimal solutions [18], however it cannot escape the deficiency that Pareto-based methods face in many-objective problems [4]. Another very interesting approach introduced in [19], termed cone ε-dominance, uses the union of an *acute* proper cone and the set $K_\varepsilon = \mathbb{R}^k_+ + \varepsilon$ to define a relation that is a partial ordering. Namely, they use ε-dominance in combination with an *acute* cone, $K_\alpha = \{\mathbf{z} : \sum_{i=1}^{k} \theta_i \mathbf{z}_i, \theta_i \geq 0\}$ with $\mathbf{z}_i$ forming an acute angle with $\mathbf{z}_j$ for all $i \neq j$. Therefore, cone ε-dominance is defined with the help of the set $K_c = K_\varepsilon \cup K_\alpha$. The partial ordering induced by the acute cone is shown in Fig. (2.10). The motivation for the introduction of this type of Pareto dominance is that the diversity of produced Pareto optimal solutions seems to be better. Namely, cone ε-dominance promotes a good spread of solutions across the entire Pareto front, and, their distribution seems to be more uniform. However, the problem reported in [4] persists for this type of dominance as well. In fact, since the regions where solutions become non-comparable are larger in cone ε-dominance it is expected that the number of non-dominated solutions increase more rapidly, compared to the Pareto dominance using $K = \mathbb{R}^k_+$, see Fig. (2.10).

2.5.3 Partial Ordering - Decomposition-Based Methods

As mentioned briefly in Section 2.1, decomposition methods employ scalarising functions to divide a multi-objective problem in to a set a single objective subproblems. The premise of this approach is that, upon successful optimisation of all the subproblems, a Pareto set will emerge, formed by the solutions of these subproblems. There are several scalarising functions that are available to the analyst [20], however in this chapter only the most fundamental method

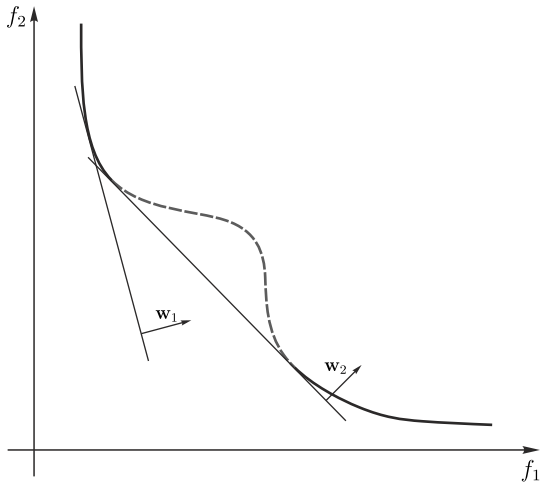

Figure 2.12: With the weighted sum, the dashed part of the Pareto front can never be obtained.

is discussed. This topic is further investigated in Section 4.2.1.

One of the simplest and perhaps most intuitive scalarising functions is the weighted sum method [5]:

$$\begin{aligned} &\min_{\mathbf{x}} \mathbf{w}^T \mathbf{F}(\mathbf{x}) \\ &\sum_{i=1}^{k} w_i = 1, \text{ and } w_i \geq 0, \end{aligned} \tag{2.23}$$

where $\mathbf{w} = (w_1, \ldots, w_k)$ is referred to as weighting vector and w_i are the weighting coefficients. The weighting coefficients can be viewed as factors that quantify the relative importance of the scalar objective functions in $\mathbf{F}(\cdot)$. The issue with the weighted sum approach is that its ability to produce Pareto optimal solutions depends strongly on the convexity of the Pareto front. For instance, for Pareto fronts of mixed type geometry, that is fronts that are partially convex and partially concave, this method cannot produce all Pareto optimal solutions [5], see Fig. (2.12). However the weighted sum method is still employed in practice, because (2.23) maintains differentiability. That is, if the scalar functions $f_i(\cdot)$ are differentiable then the scalar problem produced by the weighted sum method will also be differentiable.

Another issue with decomposition methods is that the selection of the weighting vectors, $\mathbf{w}$, is mostly based on *ad-hoc* methods. However, as it is shown in Chapter 4, the selection of the weighting vectors is very important as it determines the location of the produced Pareto optimal solutions and therefore also control the final distribution of the Pareto front approximation. Lastly, to avoid bias due to difference in scales of the objective functions, f_i, the objectives are usually normalised in the range $[0, 1]$, and this is accomplished with the help of two vectors - the

ideal objective vector and the **nadir objective vector**. The ideal objective vector, $\mathbf{z}^\star$, is defined as the vector with elements $\{\inf(f_1), \ldots, \inf(f_k)\}$, while the nadir objective vector, $\mathbf{z}^{nd}$, is the vector with elements $\{\sup(f_1), \ldots, \sup(f_k)\}$, subject to f_i be elements of objective vectors in the Pareto optimal set, $\mathcal{P}$, these vectors are illustrated in Fig. (2.11).

2.5.4 Optimality Condition - Multi-Objective Problems

The optimality condition presented in Section 2.4.2, can be generalised to multi-objective problems in the following way, with the help of generalised inequalities and for a convex objective function:

$$\begin{aligned} D\mathbf{F}(\mathbf{x})(\mathbf{y}-\mathbf{x}) &\succeq 0, \\ \forall \mathbf{y} &\in S, \end{aligned} \tag{2.24}$$

where $D\mathbf{F}(\mathbf{x})$, is the Jacobian of the objective function defined as,

$$D\mathbf{F}(\mathbf{x}) = \begin{bmatrix} \nabla f_1(\mathbf{x})^T \\ \nabla f_2(\mathbf{x})^T \\ \vdots \\ \nabla f_k(\mathbf{x})^T \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_1(\mathbf{x})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_k(\mathbf{x})}{\partial x_1} & \cdots & \frac{\partial f_k(\mathbf{x})}{\partial x_n} \end{bmatrix}. \tag{2.25}$$

Although (2.24) is not as easy to visualise as was the case in Section 2.4.2, there are some interesting observations to be made. The matrix, $D\mathbf{F}(\mathbf{x}) \in \mathbb{R}^{k\times n}$, which means that its maximum rank is, $\min\{k, n\}$[1]. Also, notice that the number of decision variables is usually much larger than the number of objectives, i.e. $n \gg k$, meaning that the maximum rank of $D\mathbf{F}(\mathbf{x})$ can be at most k. Therefore there can be at most k linearly independent vectors in, $D\mathbf{F}(\mathbf{x})$, and at least $n-k$ free variables. And from the fundamental theorem of linear algebra[2] the dimension of the null space will be at least $n-k$. So there will be an infinite number of vectors that span an $n-k$ dimensional subspace of $\mathbb{R}^n$ that would have absolutely no effect on the value of the objective function at the Pareto optimal point, $\mathbf{x}$. In other words, the optimality of the decision vector, $\mathbf{x}$, can depend at most on k number of elements in its neighbourhood where the Jacobian is a *good* approximation of the objective function. This fact holds for every Pareto optimal decision vector, $\mathbf{x}$, however it need not be the case that the independent vectors in, $D\mathbf{F}(\mathbf{x})$, be the same for every Pareto optimal point. However, this observation substantiates the empirical results presented in [21], where the authors discovered that Pareto optimal solutions in decision space, under certain regularity conditions can be represented by a $(k-1)$-dimensional manifold. Based

[1] See Section A.4.2 for further details.
[2] See Appendix A.4.2.

on the argument presented here, this manifold cannot possibly be more than k-dimensional. This observation is the basis of the Pareto estimation method presented in Chapter 6.

2.6 Multi-Objective Optimisation - A Conceptual Overview

The procedure used to solve a real-world problem is in general comprised of following stages:

- Problem definition and modeling (Analyst and Decision Maker).
- Approach selection and optimisation (Analyst).
- Complete ordering of Pareto optimal solutions and selection of the *best* solution (Decision maker).

The last stage could be omitted in the case that the decision maker has a set of preferences that can be distilled in a utility function [11, pp. 62], in which case a unique solution is output by the optimisation procedure. However, this description is greatly simplified as the interaction of the decision maker and the analyst during the optimisation stage need not be as separate as it is portrayed in the above-mentioned stages. A prevalent classification of the paradigm of interaction of the decision maker with the analyst is the following [5, pp. 63]:

- The decision maker is satisfied with one, any one, solution as long as it is Pareto optimal. This is the *no preference* paradigm.
- The decision maker has no involvement in the optimisation process, it is only after a set of Pareto optimal solutions have been generated that such a decision maker expresses a preference. From the resulting set one solution is selected by the decision maker. This is the *a posteriori* paradigm of interaction, and, is in predominant use. This is mainly due to the separation of the decision maker and the analyst, which allows both parties to operate independently, or nearly so.
- Prior to optimisation procedure, the decision maker, provides the analyst with a complete set of instructions that would enable him to select a single solution among all alternatives. Namely, the decision maker must provide the means for a definition of a complete ordering in the objective space, prior to the solution of the problem. This paradigm is called *a priori*.

- Lastly, there exists a mode of interaction of the decision maker and the analyst whereby there is exchange of information during the optimisation procedure. That is, the decision maker is intimately involved in the process and is responsible for *steering* the search direction on-the-fly. This paradigm is termed *interactive*. In essence the preferences of the decision maker are expressed progressively, which is why such methods are also referred to as progressive preference articulation methods [22].

Chapter 3

Overview of Population-Based Optimisation

3.1 Introduction

This chapter presents an overview of the most prominent population-based multi-objective optimisation methodologies, serving as a map or a starting point in the search for a suitable technique. As in cartography, maps are meant to contain important landmarks but in order to enhance legibility a vast amount of detail has been abstracted or omitted. In that spirit, hybrid algorithms, for example memetic algorithms [23], are not considered here, without any implications for the contribution they make. Additionally, although Genetic Programming (GP) instigated by Koza [24] is based on the same principles as Genetic Algorithms [25] it is also not considered here, since in the view of the author, the main focus of GP is not multi-objective optimisation.

Population-based optimisation techniques (PBOTs) have received much attention in the past 30 years. This can be attributed to their inherent ability to escape local optima, their extensibility to multi-objective problems (MOPs) and their ability to incorporate the handling of linear and nonlinear inequality and equality constraints in a straightforward manner. They are applicable to a wide set of problems and impose very few constraints on the problem structure. For example convexity, continuity and differentiability of the objective functions or constraints, are not required. PBOTs also perform well on NP-hard[1] problems where exhaustive search is impractical or simply impossible. Since there is a population of decision vectors, the Pareto front (PF) can be approximated (theoretically) to arbitrary precision.

Assuming that the problem under consideration is being *solved* for practical purposes, that

[1] Non-deterministic Polynomial time (NP)

is, it is a real-world problem and a design choice is to be made. Even if multiple competing objectives are under consideration only one solution can usually be employed. This requires a human expert or decision maker (DM) to resolve the situation by selecting a small subset of the solutions presented by the algorithm. So in this scenario, or more formally *a posteriori* preference articulation[1], the algorithm is endowed with the task of finding a set of alternative solutions. Subsequently, the DM will evaluate these solutions according to his/her preferences and make a selection. To facilitate this process the algorithm guides a population of decision vectors toward a tradeoff surface, be it partially convex, partially concave, or even discontinuous[2]. This tradeoff surface should enable the DM to choose a solution that they believe would serve their purposes well. So it becomes apparent that in MOPs a DM plays an integral role in the solution procedure. The various paradigms of interaction involving the DM and the solution process are discussed in Section 2.6, but in the remainder of this work *a posteriori* preference articulation is the main focus. For further details on the other types see [5, pp. 61] and the references therein. This choice is based on the fact that this particular method of interaction with the DM allows a greater degree of separation between the algorithm and the decision making process. Also this is invariably the reason as to why this paradigm is usually employed by researchers when developing new optimisation techniques: it enables the testing process to be conducted independently of the problem or application and most importantly it need not involve a DM at this stage. It should also be noted that specific applications are not mentioned in this chapter. The only exception to this rule is when an application explicitly results in the creation of an algorithm highly regarded in the community. The reason for this decision is that this approach facilitates a more concise and conceptual presentation of population-based optimisation techniques for nonconvex problems.

The remainder of this chapter is structured as follows. In Section 3.2 a historical regression of important developments in PBOTs is presented. In Section 3.3 an overview of the common characteristics of PBOTs is given and in Sections 3.4 - 3.10, seven of the most prominent algorithm families are described as well the methods used to extend them to multi-objective problems. In Section 3.11 a critique of the strengths and weakness of the aforementioned algorithm families is presented. Lastly, this chapter is summarised in Section 3.12.

[1] See Section 2.6.
[2] See Section 2.5.2 for the types of Pareto front geometries.

3.2 Chronology

The origins of the inspiration to use a population of trial solutions in an optimisation algorithm can be traced back to Holland [26] as a mixture of automata theory and theoretical genetics. Since then many algorithms have been developed that utilise a population of decision vectors and by means of applying variation and exploration operators this population evolves to *adapt* to the *environment*. The *environment*, represented by an evaluation (or objective) function, dictates which individuals in the population would be fit to enter the next generation. In 1975 Holland [25] formalised the idea presenting a more concrete framework for optimisation which he named Genetic Algorithms (GA). The scheme quickly gained acceptance and by 1985 Shaffer [12] proposed an algorithm, the vector evaluated genetic algorithm (VEGA), based on Holland's GA to tackle multiple objectives simultaneously. The following year Farmer and Packard [27] published a paper describing how knowledge about the human immune system can be used to create adaptive systems. This Artificial Immune System (AIS) shares many commonalities with Hollands' classifier system [25]. Two years later, in 1989, Goldberg [28] introduced a conceptual algorithm of how Pareto dominance relations could be utilised to implement a multi-objective GA. This idea is heavily used to this day when extending single objective algorithms to address MOPs.

Independently, and in parallel with Holland, Rechenberg [29] introduced Evolution Strategies (ES) in the mid-1960s. The early versions of ES had only two individuals in their population due to the limited computational resources available at the time. However, in subsequent years multi-membered versions of ES were introduced by Rechenberg and further developed and formalised by Schwefel [30] in 1975. During the same period a similar family of algorithms, the so called Evolutionary Programming, was introduced by Fogel [31].

Another important approach was introduced by Dorigo et al. [32] in 1991 which they called the Ant System (AS). The AS, inspired by the ability of ants to find the shortest route from a food source to their nest, was used to solve combinatorial problems and almost a decade later Dorigo et al. [33] presented a generalisation of the AS which they named Ant Colony Optimisation (ACO). In the mid 90s Storn and Price [34] brought forward a novel heuristic, Differential Evolution (DE). Its main characteristic was its conceptual simplicity and its wide applicability to a variety of problems with continuous decision variables. Later that same year, Eberhart and Kennedy [35] introduced Particle Swarm Optimisation (PSO), a new family of

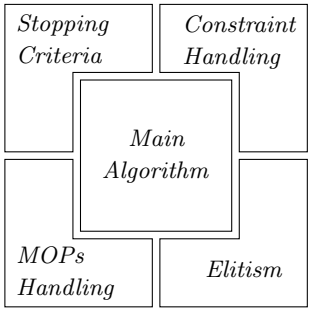

Figure 3.1: PBOT components.

optimisers inspired by the flocking behaviour of pack animals. Almost a full circle is complete by the latest addition to PBOTs, namely Estimation of Distribution algorithms (EDAs) [36, 37]. A full circle in the sense that concepts derived from nature were employed as inspiration for more than two decades, helping researchers solve complex problems. EDAs are one of the first metaheuristic algorithms not to rely on nature inspired mechanisms to drive their search. Instead, EDAs depend on probabilistic models of better performing individuals in the population to generate the entire or part of the next generation.

3.3 General Structure

Population-based optimisation metaheuristics, although diverse in their problem solving approach, share more similarities than differences. In general, they are comprised of five parts, (see Fig. (3.1)): the main algorithm, an extension to tackle MOPs, an extension to deal with constrained optimisation problems, a part to maintain promising solutions and a part to halt the algorithm execution based usually on some notion of convergence. The main algorithm deals primarily with single objective optimisation, and is comprised of three parts. Those three parts effectively drive the search combining information within the population (recombination), randomly perturbing some individuals to enhance search space exploration (mutation) and an operator to select promising individuals to be part of the new generation (selection). Recombination is a superset of operators, usually crossover, that utilise two or more decision vectors to form one or more new decision vectors. All three parts of the main algorithm have some deterministic and stochastic features. If these three operations are chosen correctly the population $X^{(G+1)}$ will have a better chance being superior, in terms of Pareto optimality, to that of the previous generation $X^{(G)}$. The population is defined as the set, $X = \{\mathbf{x}_i : i = 1, \ldots, N\}$, where

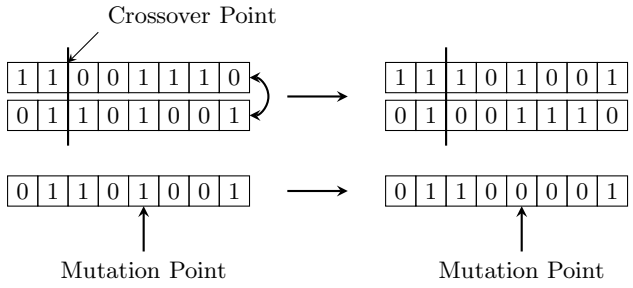

Figure 3.2: One point crossover and mutation operator in Genetic Algorithms.

$\mathbf{x}_i$ is the i^{th} decision vector and N is the number of individuals in the population.

Algorithm 3.1 PBOT Conceptual Algorithm

1: Initialise population
2: Evaluate objective function
3: **repeat**
4: Evaluate population quality
5: Recombine good individuals
6: Mutate some individuals
7: Evaluate objective function
8: **until** Termination condition is met

The general algorithm in PBOTs can be seen in *Alg. 3.1*. The population is usually initialised within the feasible region, if possible, and uniformly distributed in the absence of prior information. Most population-based metaheuristics are either, (i) based on empirical knowledge and techniques that would steer the search toward desirable regions of the decision space, or, (ii) directly replicate mechanisms found in nature.

3.4 Genetic Algorithms

GAs are a family of stochastic metaheuristic search algorithms with links to cybernetics [13, 26] and philosophically connected to Darwin's theory of evolution [38]. Social dynamics are simulated based on the premise that the *survival of the fittest* paradigm does produce qualitatively better *individuals* with the passage of several generations. Individuals are in effect decision vectors whose fitness is evaluated using the objective function. The representation of decision vectors in the basic GA is a binary *string* termed *chromosome* comprised of *genes* (decision variables). Each chromosome is mapped to a *phenotype*. A phenotype is a representation that the objective function can directly utilise to evaluate the fitness of an individual. The main op-

Algorithm 3.2 Simple Genetic Algorithm

```
1: Initialise population
2: repeat
3:     Evaluate objective function
4:     Apply crossover and mutation
5:     Select solutions for next population
6: until Termination Condition is met
```

erators used in GAs are crossover and mutation. These two operators are combined in a fashion to balance exploitation and exploration of the search space. Crossover and mutation, shown in Fig. (3.2), are applied to the chromosomes, thus facilitating the creation of an abstraction layer with respect to the phenotype. This layer of abstraction insulates the internal implementation of GAs from the problem structure, specifically the representation type of the decision variables. The main operators in GAs have been extended to deal directly with continuous decision variables [39, 40, 41] which is useful when full machine precision is required and all decision variables are continuous.

3.4.1 Multi-Objective Problems

Since their introduction GAs have been extended in numerous ways to solve MOPs. The first extension is due to Schaffer in 1985 [12]. Later based on the idea of Pareto dominance, Goldberg [28] described a conceptual algorithm, however no implementation was presented. Based on this description several approaches appeared, of which the most prominent[1] ones were due to Fonseca and Fleming [14] in 1993, Srinivas and Deb [42] in 1994 and Horn and Nafpliotis [43] in 1994. Zitzler and Thiele [44] introduced the so-called strength Pareto evolutionary algorithm (SPEA) that can be classified as one of the first algorithms to incorporate *elitism* in a very efficient manner. Later Zitzler et al. [45] introduced several modifications creating the SPEA2 algorithm. SPEA2 uses a more elaborate fitness assignment scheme. Mating of individuals is restricted to members of the population that are part of the archive and the archive is allowed to contain dominated individuals when certain conditions are met [45]. In 2002, Deb et al. [46] introduced an elitism based algorithm, the so-called non-dominated GA II (NSGA-II). The main innovative feature in NSGA-II is that the parent and offspring population is combined and non-dominated sorting is applied to this superset. Then the population is reduced back to its original size by discarding directly solutions that are part of *fronts* exceeding N. If the size

[1] "Prominent" in this context means that these approaches had significant impact on MOEA research. This is supported by the number of citations and the improvements/extensions of these methods.

of the population is still larger than the maximum allowed, the remaining population is sorted by its crowding distance [47, pp. 247] and the lowest ranked individuals are simply discarded. NSGA-II was very successful and has become a standard comparator in benchmarking. Lastly, several algorithms based on the hypervolume indicator [44] which was initially introduced as a Pareto set quality metric have been introduced in the last decade [48, 49, 50] and [51, 52] the so-called hypervolume estimation algorithm (HypE) which is further discussed in Section 3.4.6.

3.4.2 First Attempt to Extend GA to MOPs

Shaffer [12] was the first to extend GAs to become multi-objective problem solvers. In retrospect, his approach may appear simple, but for its time it was quite a conceptual leap since VEGA considered all objectives simultaneously. VEGA, see *Alg. 3.3*, partitioned the population, X, into k equally sized randomly selected sub-populations, where k is the number of objectives. Then each partition is assigned to an objective and individuals are ranked according to their performance for the corresponding objective and a mating pool is formed using proportionate selection as described in [28]. Crossover and mutation operate on the entire population in the

Algorithm 3.3 VEGA

1: $X^{(1)} \leftarrow Initialise$
2: $G \leftarrow 1$
3: $ps \leftarrow N/k$ ▷ Partition size
4: **repeat**
5: $X^{(G)} \leftarrow \text{Shuffle}_{rows}(X^{(G)})$
6: **for** $i \leftarrow 1, k$ **do** ▷ Partition the population
7: $D_i \leftarrow X^{(G)}_{ps\times(i-1)+1\rightarrow i\times ps}$
8: $E_i \leftarrow \text{Evaluate}(D_i)$
9: $S_i \leftarrow \text{Proportionate Selection}(E_i, D_i)$
10: **end for**
11: $\tilde{X}^{(G)} \leftarrow Join(S_{1,\ldots,k})$
12: $\tilde{X}^{(G)} \leftarrow \text{Crossover}(\tilde{X}^{(G)})$
13: $X^{(G+1)} \leftarrow \text{Mutate}(\tilde{X}^{(G)})$
14: $G \leftarrow G + 1$
15: **until** $G \leq MaxGenerations$

hope that linear combinations of the fitness functions would arise thus estimating the entire PF. This, however, was to some extent problematic since a phenomenon called *speciation* [12] did on some occasions occur. For instance, in objective functions with a concave Pareto front geometry, VEGA fails to approximate the entire PF as parts of the population drift toward the edges favouring one of the objective functions.

3.4.3 Fonseca-Fleming Genetic Algorithm

Fonseca and Fleming [14] were the first to use Pareto dominance relations in a multiple objective genetic algorithm (MOGA), while incorporating progressive preference articulation enabling the decision maker to guide the search interactively. The dominance relation was used to rank the individuals in a population in a manner similar to a set of selection methods proposed by Fourman [53]. Every individual in the population, after evaluation of the objective function, is ranked using the following relation,

$$r_i = 1 + p_i \tag{3.1}$$

where p_i is the number of individuals dominating the decision vector $\mathbf{x}_i$. The idea behind this method of ranking, see Fig. (3.3), is that misrepresented sections of the PF will increase the selection pressure to that direction of the front. As an example, the objective vector with rank 3 as seen in Fig. (3.3) is dominated by two individuals while the objective vector with rank 2 is dominated only by one. Alternatively, this means that within area A there is only one solution, while in area B there are two, suggesting higher concentration of solutions. This information is used to induce better spread in the objective vectors that are not part of the current PF approximation so as to maintain a relatively even *supply* of objective vectors in all regions of the PF. For the objective vectors that are part of the PF approximation, Fonseca and Fleming introduced an adaptive factor, σ_{share} to penalise objective vectors that are less than σ_{share} apart. Lastly, a facility for progressive preference articulation was embedded in MOGA providing the means for the DM to narrow down the search to *interesting* regions of the PF. This facility is implemented by incorporating a goal attainment method in the ranking procedure. Let $\mathbf{g} = (g_1, \ldots, g_k)$ be the goal vector and $\mathbf{z}_1 = (z_1, \ldots, z_k)$ and $\mathbf{z}_2$ be two objective vectors. A *preference* relation can be expressed such that when $k - s$ of the k objectives are met then $\mathbf{z}_1$ is *preferable* to $\mathbf{z}_2$ if and only if,

$$\begin{aligned} z_{(1,1\ldots k-s)} \preceq z_{(2,1\ldots k-s)} \text{ or} \\ \{(z_{(1,1\ldots k-s)} = z_{(2,1\ldots k-s)}) \wedge \\ [(z_{(1,k-s+1\ldots k)} \preceq z_{(2,1\ldots k-s+1\ldots k)}) \vee \\ (z_{(2,k-s+1\ldots k)} \not\preceq g_{(k-s+1\ldots k)})]\} \end{aligned} \tag{3.2}$$

The rest of the MOGA utilised Gray encoding for the chromosomes, two-point reduced surrogate crossover and the standard binary mutation.

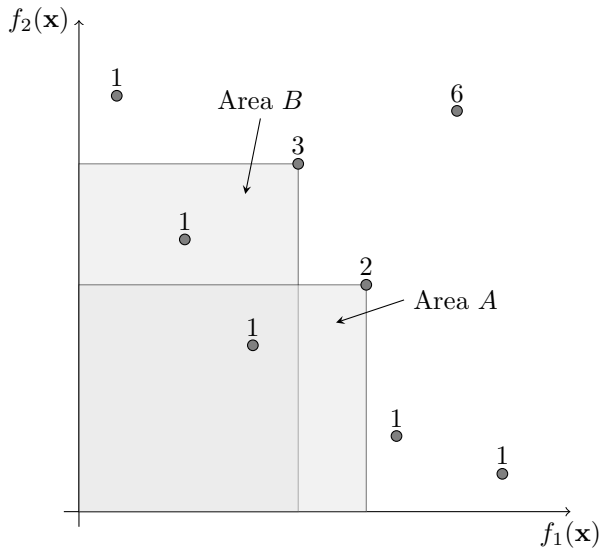

Figure 3.3: Ranking method used in MOGA. The numbers above the points represent the rank of the individual that results in that objective vector. The worst rank possible is N.

3.4.4 Non Dominated Sorting Genetic Algorithm

Another prominent algorithm, the non-dominated sorting GA (NSGA), that utilised the Pareto dominance relations was proposed by Srinivas and Deb [42]. This method is almost identical to the non-dominated sorting idea proposed by Goldberg [28]. Non-dominated sorting is very similar to a well established concept in non-cooperative game theory in which candidates need to select a *winning* strategy while considering what their opponents' strategy might be. This idea is usually referred to as *eliminating dominated strategies* which is an iterative process of deleting dominated strategies that, if chosen, would lead to lower payoffs relative to the strategies available to the opponents and thus result in unfavourable outcomes. Interested readers are referred to [54]. Non-dominated sorting, see Fig. (3.4), works as follows:

Step 1 Evaluate the objective function for each individual in the population X.

Step 2 Find the non-dominated individuals in the population, X_{nd}, and remove them from the current population $X_{new} = \{\mathbf{x} : \mathbf{x} \in (X \cap X_{nd})^c\}$.

Step 3 Repeat **Step 2** until no solutions remain in X_{new}.

Each non-dominated set identified by this process is labelled as a *front* and given a rank according to the position it has in the iteration. For example, the first identified *front* would be *front 1*,

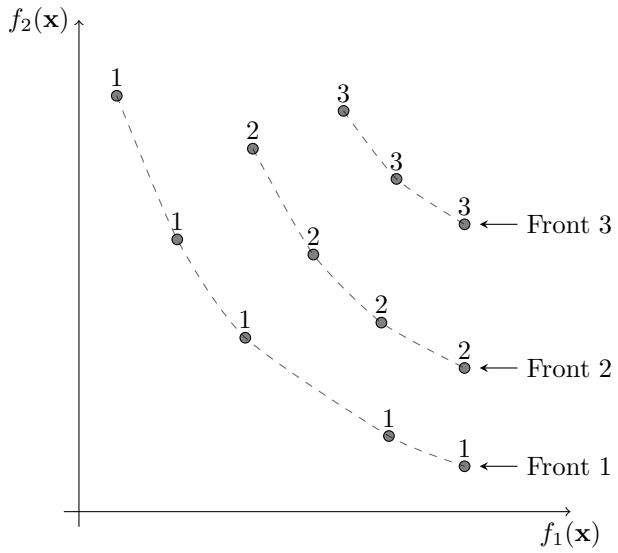

Figure 3.4: Non-dominated sorting method as used in NSGA.

the second *front 2* and so on. Subsequently the individuals of each front are assigned fitness values using the following procedure starting from the first *front*,

Step 1 Assign to all the individuals in *front 1* a fitness value N, where N is the number of individuals in the population.

Step 2 Use sharing to penalise clustered solutions in the current *front*.

Step 3 Assign the lowest fitness minus a small value ϵ to all individuals in the next *front* and go to **Step 2** until all individuals in all fronts have been assigned a fitness value.

The sharing distance σ_{share} is calculated based on distance in the decision variable space rather than the objective space and is held fixed throughout the algorithm execution. Finally NSGA employed the roulette-wheel selection operator [28] and the usual bitwise mutation operator, see Fig. (3.2). The fact that NSGA uses sharing based on the distance of the decision vectors, seems counter-intuitive since equally spaced decision vectors do not necessarily map to equally spaced objective vectors unless the mapping is affine which is usually not the case for most objective functions. This can potentially mislead the algorithm as to which solutions are densely clustered and which are not, resulting in exactly the opposite effect. Although, as the authors state, sharing can be performed based on the objective vectors [42], they do not mention how σ_{share} would be selected in that case.

3.4.5 Strength Pareto Evolutionary Algorithm

One problem identified in multi-objective GAs was that good solutions could be lost if they were not retained using some mechanism and the cost to rediscover them can potentially be prohibitive. This situation is exacerbated when the cost, computational or otherwise, of evaluating the objective function is high; this is often the case in real-world applications. A solution to this predicament is the use of *elitism*, in other words the retention of highly performing individuals in the population. In 1999 Zitzler and Thiele [44] introduced an algorithm named the strength pareto evolutionary algorithm (SPEA) that used elitism. Their approach was not the first to utilise an external archive to retain good performing individuals, however it was certainly one of the earliest and most elegant attempts.

SPEA, in addition to the current population X, maintains an archive population, $\tilde{X}$, of maximum size $\tilde{N}$ which is usually smaller than the population size N. Zitzler and Thiele [44] suggest $\tilde{N} = 0.25N$. At the start of the algorithm the external archive is empty and the population is initialised randomly within the decision variable limits or as appropriate to the problem. After the objective function has been evaluated the population is searched for non-dominated solutions which are copied to the archive, $\tilde{X} = X_{nd}$. This direct assignment is performed only on the first iteration of the algorithm. In subsequent iterations the archive updating procedure is different if the total population in the archive exceeds $\tilde{N}$. When the archive size becomes larger than $\tilde{N}$, the superfluous solutions are removed, based on a crowding algorithm that the authors of SPEA introduced. This crowding algorithm maintains the *elite* solutions that have maximal spread along the PF. After the archive size is properly reduced to its maximum size, the population and the *elite* individuals are assigned fitness values in the following way:

Step 1 Assign fitness values, called *strength* (s), to the population in the archive $\tilde{X}$, using the following relation, $s_i = \frac{n_i}{N+1}$, where n_i is the number of individuals that the i^{th} solution in the archive dominates in the regular population, X.

Step 2 The regular population, X, is then assigned fitness values according to, $\mathfrak{f}_i = 1 + \sum_{\mathbf{x}_j \preceq \mathbf{x}_i} s_i$. Note that in SPEA lower fitness value is considered to be better. Therefore, if a decision vector $\mathbf{x}_i$ is not dominated by any individual in the population X, its fitness, $\mathfrak{f}_i$, would be equal to 1.

The way the archive is maintained and used in SPEA ensures that the population is rewarded

when it approaches the PF. In a way the archive is used as a *moving target* to which the regular population aspires. Again, the rest of the algorithm utilises roulette wheel selection and random bit mutation.

3.4.6 Hypervolume-Based Evolutionary Algorithm

The need for comparative analyses regarding the strengths and weaknesses of different algorithms led to the introduction of several *indicators*, see [55], measuring various *qualities* of the resulting Pareto optimal set approximation. With the advent of such indicators, some researchers grasped the opportunity to utilise these within the evolution process, see [48],[56]. One important indicator is a derivative of the Lebesgue measure [57], initially introduced in this context by Zitzler and Thiele [44]; this indicator is now commonly referred to as the hypervolume indicator [51]. The most favourable quality of the hypervolume indicator is its monotonicity with regard to Pareto dominance [51]. This implies that the hypervolume indicator can be used in place of Pareto dominance relations for fitness assignment. However, this indicator is not without drawbacks since most known algorithms to calculate the Lebesgue measure are of exponential complexity with respect to the number of objectives [51]. This is a limitation that imposes a restriction on the number of objectives that can be considered in an algorithm based on this indicator.

Recently Bader and Zitzler [51, 52] introduced an algorithm based on the hypervolume indicator for many-objective optimisation[1]. Their idea is that precise calculation of the hypervolume is not required, per se, to enable a useful ranking of the population. Under this premise they developed a methodology which uses Monte Carlo sampling to approximately determine the value of the hypervolume and subsequently use this to assign fitness to the decision vectors. The results produced were quite favourable for HypE and the authors demonstrated that their proposed technique can effectively tackle as many as fifty objectives [51].

3.5 Evolution Strategies

The introduction of ES is due to Rechenberg [29] in 1965. This early version of ES was a two-membered mutation based algorithm the so called $(1 + 1)$-ES. Although ES share similar features and inspirational background to GAs, there are two distinct differences. First, ES do not employ a crossover operator, and second, ES utilise real decision variables as opposed to GAs that originally employed binary encoded *strings*. However, these differences have blurred over

[1] Many meaning more than three in this context.

time due to developments on both sides. Effectively both methodologies pursued adaptation and were inspired by the cybernetic movement of that era.

There is a general consensus that both mutation and recombination operators perform an important role and eventually both were employed to various degrees in these two approaches. The main argument concerns their effect on the evolving population. Holland, with his famous building block hypothesis (BBH) [25], promoted the idea that the recombination operator had a positive influence on the evolving population due to short *schemas* with good *fitness* recombining to form even better individuals. Beyer argued that this was not the case; his suggestion was that the crossover operator had the effect of *genetic repair* [58]. Beyers' hypothesis was that the features that *survive* the crossover operator were the ones common to both parents and not the small *schemas* with desirable elements.

As previously mentioned, the first version of ES was $(1+1)$-ES. This notation refers to the fact that there are two members in the population, namely the *parent* and one *offspring* and the selection for the *surviving* individual is performed among the two. The + sign essentially refers to the selection mechanism. Some later versions of ES use the comma notation, denoting that the *parent* takes no part in the selection process and new individuals are selected only from the offspring pool. The main *search* operator in $(1+1)$-ES is the mutation operator defined as follows,

$$\mathbf{x}^{(G+1)} = \mathbf{x}^{(G)} + \mathcal{N}(\mathbf{0}, \sigma) \tag{3.3}$$

where $\mathcal{N}(\mathbf{0}, \sigma)$ is a vector of random numbers of size n drawn from the normal distribution with zero mean and σ standard deviation. Here $\mathbf{x}^{(G)}$ represents the parent solution and $\mathbf{x}^{(G+1)}$ the offspring. If $f(\mathbf{x}^{(G+1)}) \leq f(\mathbf{x}^{(G)})$ the offspring is retained and the parent is discarded. Otherwise the same parent is used in the next generation.

ES were extended to a *true* population-based optimisation methodology by Schwefel [59]. Schwefel introduced $(\mu+\lambda)$-ES as well as (μ, λ)-ES. Here, μ is the number of parents producing λ offspring. The resulting population, after the selection process, is reduced to the number of parents, μ. Schwefel combined these two variants to enable a parallel implementation of ES as well as to experiment with adaptation in the control parameters such as σ [60]. For further details on the $(\mu+\lambda)$-ES the reader is referred to [59, 60, 61].

3.5.1 Multi-Objective Problems

The earliest application of ES to MOPs that the author could trace is due to Kursawe [62] in 1991. Kursawe used *recessive* genes as a way to consider all objectives simultaneously. Several years later Binh and Korn [63] introduced an algorithm, multi-objective evolution strategy (MOBES), that estimated the nadir and ideal vectors[1] in order to partition the PF into several segments. The population was evenly distributed between these segments and selection was performed based on the relative performance of each individual in the corresponding segment. In 1998 Laumanns [64] introduced an algorithm inspired by the predator-prey game and showed how this paradigm can be exploited to aid the population in approaching the PF uniformly. Subsequently, Knowles and Corne [65] were the first researchers to utilise archiving and dominance relations to extend ES to MOPs; the resulting algorithm is known as Pareto Archived Evolution Strategy (PAES). In its simplest form, PAES is based on a $(1 + 1)$-ES, a decision resulting from the authors' observation that local search algorithms seemed to perform better than population-based techniques when applied to telecommunication network design problems. An attractive feature of PAES is its conceptual simplicity. Later, in 2001, Jin et al. [66] explored the weighted sum decomposition method [5, pp. 78] along with an archiving strategy to retain non-dominated individuals. This seemed necessary since the technique without archiving struggled to maintain Pareto optimal solutions. Several other techniques were introduced since 2001 by various authors, most notably by Igel et al. [67] in 2007 who produced an algorithm based on the covariance matrix adaptation evolution strategy, MO-CMA-ES. MO-CMA-ES is invariant to affine transformations applied to the objective function. This quality is deemed useful, especially for non-separable test problems [67].

3.5.2 First Attempt to Extend ES to MOPs

In a short paper Kursawe [62] illustrated a conceptual framework upon which a multi-objective ES can be based. Kursawe suggested using the entire population to optimise one objective in each generation. The choice as to which objective should be optimised in each generation was made probabilistically. So, in every generation every objective has a chance to be the fitness function according to which selection is made. The aforementioned probabilities, which are represented as a probability vector of dimension k, can either be predefined at the start of the algorithm or be randomly generated every t generations. By dynamically changing the probability vector and

[1] See Section 2.5.3 for the definition of the ideal and nadir objective vectors.

by altering the objective in almost every generation, the problem is transformed to a varying problem. To tackle this issue Kursawe used two independent chromosomes to describe every individual. These were termed as *dominant* and *recessive* genotypes. The individual then would be represented mainly using the *dominant* genotype but with a probability of 0.3 the *dominant* and *recessive* genes would be exchanged. However, comparative results with and without individuals with double genotypes were not presented. A further suggestion by the author was the use of the symmetric Weibull distribution by the mutation operator in the main ES algorithm, although a detailed justification is not presented.

3.5.3 Predator-Prey Model

An interesting approach was introduced by Laumanns et al. [64] based on the nature-inspired *predator-prey* model. The population of decision vectors assumes the role of the *prey* being chased by a number of *predator*, notionally both situated on the vertices of an undirected graph. All prey maintain their location on the graph while predators are allowed to move to neighbouring vertices according to a probabilistic transition rule [64]. Every predator is associated with a single objective function, hence there are at least $k \cdot M$ predators, where $M \in \mathbb{N}^+$. When a predator lands on a vertex, the weakest[1] prey in the neighbourhood is deleted from the population and a new one is created using the mutation operator. It is assumed that no prior preference information is available hence the search will not be biased toward any objective. This is achieved by choosing appropriately the structure of the graph so that every vertex has the same chance of being visited by a predator [64].

The number of predators is a multiple of the dimension of the objective vector since each predator is associated with a single objective. It should also be noted that, according to the authors, the use of a recombination operator in this technique produced worse results on the given problem set [64].

One potential weakness of this technique is that its scalability to more than two objectives is unclear.

3.6 Artificial Immune Systems

The immune system (IS) is a fine tuned concert of a vast array of cells acting in synchronism in order to protect the body from pathogens. Pathogens are substances or microorganisms that can

[1] In this context the term, *weakest*, refers to the prey with the worst objective function related to that predator.

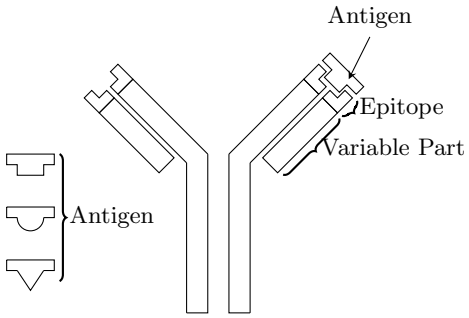

Figure 3.5: Antibody schematic diagram.

affect the balance of the body (*homeostasis*), impairing its normal functioning and even cause death. The immune system responds, *immune response*, to such threats creating specialised cells capable of neutralising pathogens and restoring balance. In order for this procedure to progress smoothly the immune system must attack the source of the disturbance while minimising the effect this action has on healthy cells in the body - *specificity*. Once a particular threat has been dealt with, the second time the body is exposed to the same pathogen the immune response will be swifter and stronger to such a degree that the organism might not even realise that it had been afflicted by a pathogen, so the immune system also exhibits *memory*.

Pathogens have certain features that the immune system can identify. These features are called *antigens*, and their presence in the organism is detected by *antibodies*. Antibodies are highly specialised molecules that have the capacity to identify specific antigens. An antibody, see Fig. (3.5), identifies an antigen by binding on it, thus serving as a beacon signalling other cells in the immune system to devour the intruder. Although antibodies are highly specialised, their diversity is enormous thus enabling the immune system to attack a vast variety of antigens. When a particular antigen is identified the antibody that successfully identified it will proliferate by cloning via a process termed *the clonal selection principle* [68, 69]. Once the pathogens have been eliminated, superfluous antigens are dissolved mainly via the process of *apoptosis* [70]. However some cells responsible for the proliferation of this particular type of antibodies survive as memory cells [71], and if the host is infected again by the same pathogen these memory cells will be activated to produce antibodies much faster compared with the first encounter, *primary response*, with this particular antigen.

Although this introduction to the immune system will serve the purpose of illustrating the parallelism of the biological system with Artificial Immune Systems (AIS), it does not even

scratch the surface of the delicate intricacies present in the immune system. The interested reader is referred to [72].

AIS have been employed in various applications, such as intrusion detections systems, virus detection, anomaly detection fault diagnosis, and pattern recognition [73]. AIS have been used in place of fitness sharing techniques in GAs to maintain diversity and distribute the population evenly along the PF [74]. Another interesting use is in time-dependent optimisation problems where AIS successfully *adapt* to a changing problem [75]. Despite these applications, direct use of AIS to optimisation problems prior to 1999 is scarce. This could potentially be attributed to the complexity involved in implementing an AIS.

3.6.1 Multi-Objective Problems

In 1999 Yoo and Hajela [76] implemented the first direct application of AIS to multi-objective optimisation when they applied their algorithm on a 2-bar truss and an I-beam structure optimisation [76]; this algorithm is described in the next section. Another approach based on the *clonal selection principle* was proposed by Coello and Cortés [77] in 2002, in which multiple objectives are handled using a Pareto dominance based approach. Archiving was also used, which the authors termed *secondary memory*, to enhance selection pressure toward the PF and retain non-dominated individuals. A later development was introduced by the same authors in 2005 [74]; this algorithm is also discussed later. Another algorithm based on the clonal selection principle was introduced by Zhang [78], utilising several other immune system inspired concepts. Other similar algorithms have also been proposed by various authors, mostly based on the clonal selection principle. Lastly, Gong et al. [79] introduced the Nondominated Neighbour Immune Algorithm (NNIA). NNIA is a Pareto-based algorithm that was shown to be competitive with several other evolutionary algorithms by its authors [79]. However, since NNIA is based on more elaborate concepts its implementation complexity is quite high, in comparison to other population-based algorithms, which is typical of immune based multi-objective algorithms.

3.6.2 First Attempt to Extend AIS to MOPs

The immune system based algorithm proposed by Yoo and Hajela [76] used the weighting method [5, pp. 27] to account for all the objective functions simultaneously. The weighting method

aggregates all the objective functions into a single function using the following relation,

$$\begin{gathered} \min_{\mathbf{x}} g(\mathbf{x}) = \sum_{i=1}^{k} w_i f_i(\mathbf{x}) \\ \sum_{i=1}^{k} w_i = 1, \text{ and, } w_i \geq 0, \text{ for all } i \\ \text{subject to } \mathbf{x} \in S, \end{gathered} \tag{3.4}$$

where w_i represents weight of the i^{th} objective function. When all of the objective functions are equally important to the decision maker, the weights can be set to $w_i = \frac{1}{k}$, where k is the number of objective functions. Using the weighting method results in an approximate solution for one point of the PF. To explore more points, various combinations of the weights have to be used. Yoo and Hajela randomly generated a set of weight combinations in an attempt to uniformly sample the PF and then used these different weighting vectors to guide the search. The procedure they used is as follows:

Step 1 The population X is generated at random (note that the decision vectors are encoded as binary *strings* with length M).

Step 2 Create a randomly distributed weighting vector set. A method to generate weighting vectors in this fashion is described in Section 4.2.2.

Step 3 For all the individuals in the population evaluate the objective function for all the weight vector sets and select a proportion of the best performing individuals X_{best}. Set X_{best} as *antigens*; the rest of the population in X is treated as *antibodies*.

Step 4 Select one *antigen* from X_{best}

Step 5 Select a portion of the *antibody* population. The authors suggest that the size of this be three times larger than the number of individuals in the *antigen* population (X_{best}).

Step 6 For all the selected *antibodies* from the previous step use $R = \sum_{i=1}^{M} u_i$, where u_i is 1 for a matching bit and 0 otherwise, to identify similarity strength among *antibodies* and *antigens*. Subsequently, using a predefined threshold on similarity strength, select *antibodies* that match the selected *antigen* and store their similarity strength R.

Step 7 Repeat the process from **Step 4** (the authors suggest this be repeated three times).

Step 8 Individuals from the population are selected based on the fitness value of the previous steps.

Step 9 Apply crossover and mutation operators to the population.

Step 10 Repeat from **Step 3** until convergence or some predefined stopping criterion.

Some difficulties encountered in the above implementation are that the weighting method is used and its weaknesses could affect the algorithm in several ways. For instance, evenly spaced weighting vectors do not necessarily produce an even distribution of points along the PF and two widely differing weight vectors need not produce significantly different points along the PF [5]. An additional shortcoming of the weighting method is that it cannot guarantee that all the Pareto optimal points can be obtained [5, pp. 80].

It is apparent, even in this early application of AIS to MOPs that there is a *jump* in algorithm complexity when compared to a GA. This complexity inevitably leads to an increased demand on computational resources. This cost will have to be justified especially since GAs, as can be seen in [14], can explicitly deal with constraints as well. However, the computational cost of most algorithms can often be ignored when compared with the cost of evaluating the objective function.

3.6.3 Multi-objective Immune System Algorithm

Coello and Cortés introduced another algorithm based on AIS, multi-objective immune system algorithm (MISA) [74]. MISA was compared in performance with three other algorithms, namely micro genetic algorithm (microGA) [80], NSGA-II [46] and PAES [65] and was found to perform similarly and on some occasions better. The MISA algorithm is far too elaborate to describe in detail here and therefore, only a conceptual overview will be given.

MISA initialises the population randomly, in common with almost every algorithm. There is an archive, called a *secondary population* which is initially empty. The algorithm is based on Pareto dominance relations. Specifically, for unconstrained problems 5% of the non-dominated individuals in the population are selected and then copied to the secondary population. The secondary population is *cloned*, i.e. copied several times. Then a mutation operator is applied to the resulting population. Prior to application of mutation, the individuals are ranked and non-dominated individuals, that also satisfy the constraints, are placed on *higher* rank compared to non-dominated individuals that fail to meet the constraints. The most fit individuals are

mutated on fewer bit positions than the less fit and then the population is decreased to its original size, N, by selecting the best performing individuals. This process is repeated until the stopping criterion is met.

3.7 Ant Colony Optimisation

Ants come from the same family as wasps and bees [81], namely *Hymenoptera Fromicidae*. As individuals, ants' capabilities are limited. Apart from their extraordinary strength their sight is very limited and their hearing and sense of smell depend on their antennas. Despite these facts, ants as a collective exhibit very interesting behaviour patterns [82]. In their search for food, ants explore their nest surroundings in a random-walk pattern. However, when an ant discovers food, it returns to the nest releasing pheromones [82] of varying intensity depending on the quality and quantity of the discovered food source. Other ants close to the pheromone trail are more likely to follow that path and, if there is still food to be found at the end of the trail, they release pheromones as well, thus reinforcing the pheromone scent which, in turn, increases the probability that more ants will follow that path. Alternatively, when an ant reaches the destination indicated by the pheromone trail and does not discover anything interesting, no pheromone is released and progressively the path becomes less and less attractive as pheromones evaporate.

In 1991, Dorigo et al. [83] introduced an algorithm, *Ant System* (AS), inspired by the emergent behaviour of ants. The AS was tailored to solve combinatorial problems and the travelling salesman problem (TSP) was used as a test bed, see Fig. (3.6). Almost a decade later Dorigo and Di Caro [33] formalised the AS and other similar methods in a unified framework, ant colony optimisation (ACO). ACO is comprised of N individuals called *ants*. These are distributed among μ cities and each ant progressively creates a candidate solution for the problem. The ants *decide* which city they will next visit based on the pheromone trail (τ) associated with each edge leading to a particular city and the separating distance of the cities. Assuming that an ant starts off from city A in Fig. (3.6), it has three options, namely to go to B, C or D. Each of these edges has a pheromone trail associated with it. The probabilistic transition rules are controlled by two parameters, namely τ and η. The parameter τ can be initialised using prior information, if available, or with the same value for all edges. Initially η represented the *visibility* of the ants which depended on the distance of a city from the next and was defined as, $\eta_{AB} = \frac{1}{d_{AB}}$ for the edge connecting city A with B. The smaller η_{AB} is, thus the larger the distance d_{AB} is, the less

likely is that an ant in city A to choose the city B for its next step in the tour, see (3.6). As ants *traverse* across the edges constructing different solutions the pheromone across the edges is updated as,

$$\tau_{i,j} = (1-\rho)\tau_{ij} + \sum_{s=1}^{N} \Delta\tau_{i,j}^{s} \tag{3.5}$$

where ρ is defined as the evaporation rate of the pheromones, and $\Delta\tau_{i,j}^{s}$ is the amount of pheromone added by ant s on the edge connecting the i^{th} and j^{th} node. Therefore, the larger the deposit of pheromones across an edge i, j the larger $\tau_{i,j}$ is, hence the probability that more ants will select this path is increased. The probabilistic transition rule, based on τ and η, that govern the *behaviour* of ants is summarised by the following relation,

$$p_{i,j} = \begin{cases} \dfrac{\tau_{i,j}^{\alpha}\eta_{i,j}^{\beta}}{\sum_{m=1}^{\mu}\tau_{i,m}^{\alpha}\eta_{i,m}^{\beta}} & \text{if } j \text{ is a valid destination} \\ 0 & \text{if } j \text{ is not a valid destination,} \end{cases} \tag{3.6}$$

where a valid destination for an ant is a node that the particular ant is allowed to visit at that particular stage. The probability that an ant on node i will visit a node j is given by $p_{i,j}$, and μ is the number of nodes in the problem. For the TSP problem a node is a *valid* destination if a particular ant has not visited that node. Also α and β are real and positive numbers used to favour either τ or η; if $\alpha = \beta$ the information gained from τ is considered equally important with η.

The general ACO algorithm has the following stages,

Step 1 Initialise pheromone values, τ, for all the edges. Usually all the edges receive the same pheromone value at the start of the algorithm unless prior information is available indicating that favouring certain edges would increase convergence speed.

Step 2 Construct a solution for each ant, $\mathbf{x}_s$.

Step 3 Update the pheromone values for each edge.

Step 4 Go to **Step 2** until stopping criteria are met.

The rules for constructing a solution $\mathbf{x}_s$ for each ant are problem-dependent.

Since 1991, ACO has been successfully applied to various combinatorial problems such as vehicle routing, sequential ordering, project scheduling, multiple knapsack and protein folding [83].

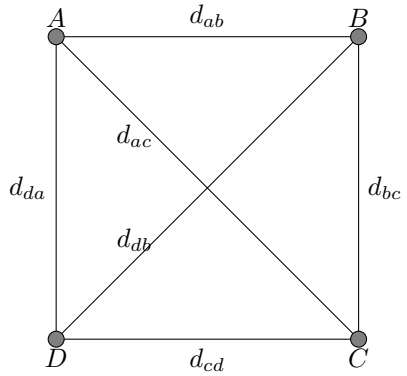

Figure 3.6: Travelling salesman problem with four cities. Although the edges connecting the cities seem equidistant they need not be. The distance of two cities is represented by d.

3.7.1 Multi-Objective Problems

ACO algorithms have been extended to two or three objectives but a methodology that is scalable with an increased number of objectives has yet to be presented [84]. The key difficulty is the way that the pheromone matrix is to be represented for more than two objectives [84]. The solutions to this issue can, in general, fall into two categories. The first use a single pheromone matrix (see Iredi et al. [85]), and the second utilise many pheromone matrices (see, for example, Doerner et al. [86]).

3.7.2 First Attempt to Extend ACO to MOPs

The first extension of ACO to MOPs was introduced by Gambardella et al. [87] in 1999 and was applied to a vehicle routing problem with two objectives, number of tours and total travel time; both objectives are to be minimised. The strategy adopted to extend ACO to MOPs was to use two independent ant groups with different pheromone matrices; the number of tours objective is considered more important relative to the total travel time, a procedure very similar to lexicographic minimisation. The algorithm is started using the nearest neighbour heuristic to create an initial feasible solution, ψ, and subsequently improvements to this solution are admitted when a new solution is found that decreases the number of vehicles used or utilises the same number of vehicles as ψ, but the total travel time is smaller.

3.7.3 Bi-Criterion Ant Colony Optimisation

Although Gambardella et al. [87] did successfully apply ACO to a MOP, their methodology was problem-specific and the decision maker had to order the objectives according to their

importance. This can often prove problematic if no prior information is available for the problem or that information is inadequate. Iredi et al. [85] were the first to consider all objectives simultaneously extending ACO to MOPs in a more general way, utilizing the weighting method to obtain different regions of the PF, see (3.4). The methodology presented by Iredi et al. had several alternative approaches. Here, only the one that produced the best results is described.

Iredi et al. used multiple ant colonies to approximate the PF, with all colonies having the same number of ants, $m = \frac{N}{p}$, where p is the number of colonies, and two pheromone matrices were used, $M^{(1)}$ and $M^{(2)}$, one for each objective. A major point is that only non-dominated individuals (ants) are allowed to update the pheromone matrices, thus solutions corresponding to dominated objective vectors progressively lose their *attractiveness* since their pheromone values are decreased. To achieve this behaviour the pheromone matrices are updated in two stages. In the first instance, evaporation is applied to all the pheromone values, and in the second stage, the only values updated are the ones that correspond to non-dominated individuals. Although there is no explicit fitness assignment in this algorithm, the described mechanism creates selection pressure toward better individuals. So all pheromone values are updated using the following,

$$\tau_{i,j} = (1 - \rho)\tau_{i,j} \tag{3.7}$$

where ρ is the evaporation rate, and non-dominated solutions receive an additional update,

$$\tau_{i,j} = \tau_{i,j} + \Delta \tag{3.8}$$

where Δ is a fixed amount that depends on the number of non-dominated solutions ν i.e., $\Delta = \frac{1}{\nu}$. The different weighting vectors $\mathbf{w}$ are generated so that a 50% overlap among ant colonies exists. This has a similar effect to that of a sharing function in genetic algorithms and helps maintain an evenly distributed PF. The $\mathbf{w}$ values are pre-set using the following expression,

$$w_{i,k} = \frac{i-1}{p+1} + \frac{2(k-1)}{p(p+1)} \tag{3.9}$$

where $i \in [1, p]$ is the i^{th} colony and $k = \left[1, \frac{N}{p}\right]$ is the weight vector associated with the k^{th} ant in that colony.

For a population of N ants, the algorithm outlined by Iredi et al. is the following:

Step 1 Generate p ant colonies, where N is the size of the entire ant population.

Step 2 Generate N weights $\mathbf{w}$, for each ant in all colonies.

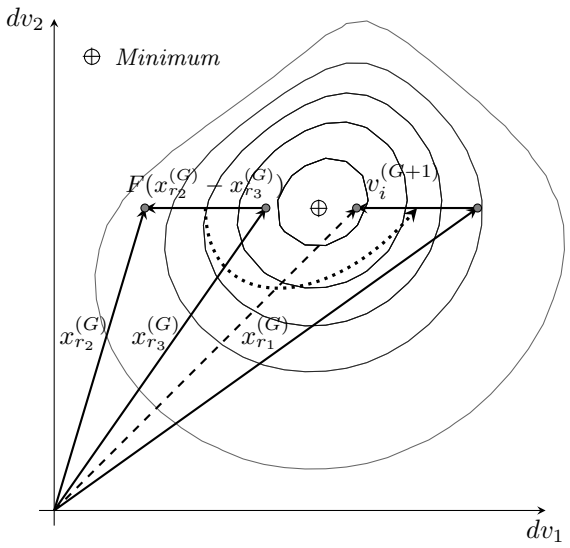

Figure 3.7: Illustration of the mutation operator in DE, in this case $F = 1$ and $\mathbf{x}_i^{(G+1)} = \mathbf{v}_i^{(G+1)}$ since for this instance $f(\mathbf{x}_i^{(G+1)}) < f(\mathbf{x}_i^{(G)})$, and, dv_1 and dv_2 stand for decision variable one and two respectively.

Step 3 Create solutions and evaluate.

Step 4 Find the non-dominated solutions X_{nd}.

Step 5 Update the pheromone matrices $M^{(1)}$ and $M^{(2)}$ using (3.7), evaporation.

Step 6 Update the pheromone elements corresponding to non-dominated solutions using (3.8).

Step 7 Repeat from **Step 3** until termination criteria are met.

This method performs reasonably well for two objectives, however as the authors mention, it is not easily extensible to more than two objectives.

3.8 Differential Evolution

DE was introduced by Storn and Price [34] as a global search method over real decision vectors, although later extended versions for integer [88] and binary representations [89] were introduced. The major strengths of DE are its conceptual simplicity, ease of use and implementation; also the number of tuning parameters is very small and the same parameter values with no or very little change are found to be applicable to a wide range of problems.

For a population of size N and decision vectors of dimension n, the current generation is

$$\mathbf{x}_i^{(G)} \text{ for } i = \{1, 2, \ldots, N\}. \tag{3.10}$$

The basic DE algorithm has three stages in its iteration phase: mutation, crossover and selection. During the mutation stage all decision vectors in the population are perturbed resulting in a temporary new population $\mathbf{V}^{(G+1)}$ of the same size with $X^{(G)}$. The mutation operator in DE, shown in Fig. (3.7), is described as,

$$\begin{gathered} \forall\, \mathbf{x}_i^{(G)},\, i = \{1, 2, \ldots, N\}, \\ \mathbf{v}_i^{(G+1)} = \mathbf{x}_{r_1}^{(G)} + F\left(\mathbf{x}_{r_2}^{(G)} - \mathbf{x}_{r_3}^{(G)}\right), \end{gathered} \tag{3.11}$$

and if $f(\mathbf{v}_i^{(G+1)}) < f(\mathbf{x}_i^{(G)})$ the newly formed parameter vector $\mathbf{v}_i^{(G+1)}$ is assigned to $\mathbf{x}_i^{(G+1)}$, otherwise $\mathbf{x}_i^{(G)}$ is retained in the next generation. The parameters $r_1, r_2, r_3 \in \{1, 2, \ldots, N\}$ are sampled at random without replacement from the set $\{1, \ldots, N\} \setminus \{i\}$ for each individual in the population. The parameter $F \in [0, 2]$ is a scaling factor controlling the variation $(\mathbf{x}_{r_2}^{(G)} - \mathbf{x}_{r_3}^{(G)})$.

The crossover operator in DE is defined as

$$\begin{gathered} \mathbf{u}_i^{(G+1)} = \left(u_{i,1}^{(G+1)}, u_{i,2}^{(G+1)}, \ldots, u_{i,n}^{(G+1)}\right), \\ \text{where} \\ u_{i,j}^{(G+1)} = \begin{cases} v_{i,j}^{(G+1)} & \mathcal{U}(0,1) \leq C_r \text{ or } j = ridx(i), \\ x_{i,j}^{(G)} & \mathcal{U}(0,1) > C_r \text{ and } j \neq ridx(i), \end{cases} \\ j = 1, 2, \ldots, n\,, \end{gathered} \tag{3.12}$$

and C_r is the crossover rate parameter, bounded in the range $[0, 1]$, $ridx(i)$ is a random index in the range $\{1, \ldots, n\}$,[1] and $\mathcal{U}(0, 1)$ is a random number sampled from a uniform distribution in the domain $[0, 1]$. Once the new decision vectors $\mathbf{u}_i^{(G+1)}$ have been generated, the selection is based on the *greedy* criterion, which can be stated as

$$\mathbf{x}_i^{(G+1)} = \begin{cases} \mathbf{u}_i^{(G+1)} & \text{if } f(\mathbf{u}_i^{(G+1)}) < f(\mathbf{x}_i^{(G)}) \\ \mathbf{x}_i^{(G)} & \text{otherwise.} \end{cases} \tag{3.13}$$

So the DE algorithm progresses exactly as *Alg. 3.2* but using the crossover, mutation and selection operators as defined above. Some general guidelines on tuning DE can be found in [90].

[1] As a reminder, n represents the number of decision variables.

3.8.1 Multi-Objective Problems

One of the early extensions of DE to MOPs was presented by Chang et al. [91] in 1999 in conjunction with tuning a fuzzy system controlling the operation of a train, using a Pareto dominance based approach. Two years later in 2001 a couple of new extensions were introduced. One was due to Abbass et al. [92] which they called Pareto-frontier Differential Evolution (PDE) where they preserved the greedy selection algorithm in the initial DE algorithm with the additional requirement that the offspring at least weakly dominate their parents. The second algorithm introduced in 2001 was due to Lampinen the so-called generalised DE (GDE) [93]. In 2002 another approach based on Pareto dominance appeared by Madavan [94], Pareto DE approach (PDEA), that used the non-dominated sorting algorithm in NSGA-II [46]. A decomposition-based extension of DE to many-objective problems was also presented in 2003 by Hughes [95], the so called multiple single objective Pareto sampling (MSOPS) and its updated version introducing a new fitness assignment process (MSOPS-II) was presented in 2007 also by Hughes [96]. It should be noted that although the problems used in [95] were 2 and 3-objective, MSOPS scales well for many-objective problems, see for example [56]. In 2004, Parsopoulos et al. [97] introduced the vector evaluated DE (VEDE) which used a similar approach to VEGA [12] to extend DE to MOPs. The same year, Kukkonen and Lampinen [98] introduced GDE-2 improving upon the diversity of the Pareto set with respect to the first algorithm. The following year, the same authors presented GDE-3 [99] further improving its performance. It should be noted that all the GDE algorithms did consider constraints explicitly in their main algorithm. Lastly, an approach based on decomposition methods was suggested by Zhang and Li [2]. This technique is further discussed later.

3.8.2 First Attempt to Extend DE to MOPs

The first algorithm to extend differential evolution to multi-objective problems is due to Chang et al. [91]. The algorithm was applied to a fuzzy train operation controller with three objectives, deviation from arrival time, energy consumption and passenger comfort, from which the first two are minimised and the last is maximised.

The approach adopted by Chang et al. [91] to extend DE to MOPs is divided into two phases excluding the standard DE operators. In the initial phase, an empty archive is created to contain non-dominated individuals. This archive is populated with solutions by comparing every individual in the current population $X^{(G)}$ with the individuals contained in the archive

A. If the individual is not dominated by any solution in the archive it is added to A and all the individuals, if any, that are dominated by the new individual are removed from the archive. In the second phase the objective vectors are scaled, in a process similar to fitness sharing in GAs, using the following,

$$D_k(\mathbf{x}_i) = \begin{cases} \frac{f_k(\mathbf{x}_i)}{\sum_j L(\mathbf{x}_i, \mathbf{x}_j)}, \ \forall i, j & \text{if } \sum_j L(\mathbf{x}_i, \mathbf{x}_j) \neq 0 \\ f_k & \text{otherwise ,} \end{cases} \tag{3.14}$$

where $L(\mathbf{x}_i, \mathbf{x}_j)$ is defined as:

$$L(\mathbf{x}_i, \mathbf{x}_j) = \begin{cases} 1 - \frac{d_{i,j}}{\sigma} & \text{if } d_{i,j} \leq \sigma \\ 0 & \text{if otherwise ,} \end{cases} \tag{3.15}$$

$d_{i,j}$ is the Euclidean distance of the individual $\mathbf{x}_i$ from $\mathbf{x}_j$ in the solution space and σ has the same effect as σ_{share} in GAs. Although this procedure bears some similarity to the way fitness sharing is used in GAs there is a significant difference. In GAs, a scalar fitness value is used for every individual. In this approach Chang et al. [91] used the objective vectors and if the decision vectors are within σ distance then the entire objective vector is scaled. Scaling is applied to a newly generated population which is compared with the individuals in the archive, using the same process as in phase one.

The algorithm presented by Chang et al. [91] is summarised as follows:

Step 1 Initialise the population at random and set the archive to, $A = \varnothing$

Step 2 Evaluate the population X using the objective functions.

Step 3 Compare all individuals in the population X with the population in the archive A. Add non-dominated individuals to the current archive population.

Step 4 Generate and evaluate a temporary population using crossover and mutation operators and use (3.14) and (3.15) to scale the objective vectors as appropriate. The individuals that are not dominated by any member of A are added to it and passed to the next generation; the rest are discarded.

Step 5 Repeat from **Step 2** until the termination criteria are met; the authors used a fixed number of iterations.

One potential difficulty with this algorithm is that since the sharing is performed in the decision variable space, as in NSGA, the resulting uniformity of the PF might suffer.

3.8.3 Generalised Differential Evolution

Another interesting multi-objective extension of DE was presented by Lampinen [93] in 2001. The advantage of this approach is that GDE was developed to deal with constraints as well. This renders it very useful for real-world applications, although that is not to say that all other algorithms cannot be extended to handle constraints. However GDE performance was overly sensitive to parameter selection [99], also no mechanism was employed to preserve diversity in the PF. These issues were later both addressed by Kukkonen and Lampinen [98] but these modifications affected conversely the convergence rate. The latest addition to the GDE algorithm was presented in 2005 by Kukkonen and Lampinen [99] providing sufficient evidence of improvement over previous versions of GDE and NSGA-II. This new algorithm is referred to as GDE3. Another interesting feature of GDE is that the main algorithm is clearly delineated from the part that enables GDE to handle constrained MOPs which can be useful if a modification to the main algorithm is desirable.

GDE3 has the ability to handle any number of constraints and objectives and in the case where there is only one objective and no constraints it gracefully falls back to the basic DE algorithm. The inherent handling of constraints in GDE3 is achieved by means of an *augmented* definition of Pareto dominance described as follows.

Definition 1. *A decision vector* $\mathbf{x}^\star$ *is said to* ***constraint-dominate*** *a decision vector* $\mathbf{x}$*, written as* $\mathbf{x}^\star \prec_c \mathbf{x}$*, if and only if one or more of the following conditions are true,*

- $\mathbf{x}^\star \in S$ *while* $\mathbf{x} \notin S$.
- $\mathbf{x}^\star, \mathbf{x} \notin S$ *but* $\mathbf{x}^\star$ *fails to satisfy a smaller number of constraints compared to* $\mathbf{x}$.
- $\mathbf{x}^\star, \mathbf{x} \in S$ *and* $\mathbf{x}^\star \prec \mathbf{x}$.

The definition for *weak-constraint-dominance* is as above with the exception that the last condition requires $\mathbf{x}^\star \preceq \mathbf{x}$. The GDE3 algorithm can be summarised as follows,

Step 1 Initialise the population X, usually randomly or from a predefined feasible set.

Step 2 Apply mutation and crossover operators using the standard DE algorithm or some alternative and evaluate the objective functions.

Step 3 Apply the following rule to all the individuals in the population to decide whether an individual will be accepted or not in the next generation $(G+1)$, $\mathbf{x}_i^{(G+1)} = \mathbf{u}_i^{(G)}$ if $\mathbf{u}_i^{(G)} \preceq_c \mathbf{x}_i^{(G)}$, otherwise $\mathbf{x}_i^{(G+1)} = \mathbf{x}_i^{(G)}$.

Step 4 Expand the population X by adding individuals, $\mathbf{u}_i^{(G)}$, that satisfy all of the following conditions,

- $\mathbf{u}_i^{(G)}$ satisfies all constraints,
- the i^{th} individual in the population X is unchanged, and
- $\mathbf{x}_i^{(G)} \not\prec \mathbf{u}_i^{(G)}$.

Step 5 Based on some crowding measure, reduce the population size back to N.

Step 6 Repeat from **Step 2** until the termination criteria are met.

3.8.4 Multi-Objective Evolutionary Algorithm based on Decomposition

As mentioned in Section 3.3, the main algorithm in PBOTs is usually built with single objective optimisation in mind and as such the extension to multiple-objectives, as will be apparent by now, requires a number of considerations. For instance, diversity preserving operations and *eliteness* preserving strategies inevitably result in higher computational costs. And for algorithms utilizing directly the non-dominated sorting strategy, the cost is even higher. Relatively recently a multi-objective evolutionary algorithm based on decomposition (MOEA/D) was introduced by Zhang and Li [2] as an alternative way of extending DE and evolutionary algorithms (EAs), in general, to deal with MOPs. The approach depends on one of several available decomposition techniques, - weighted sum, Chebyshev [5] and normal boundary intersection [100] decompositions - with each having its own strengths and weaknesses. The minimisation problem defined in Section 2.22, when using the Chebyshev decomposition, can be restated as follows,

$$\begin{aligned} \min_{\mathbf{x}} g_\infty(\mathbf{x}, \mathbf{w}^s, \mathbf{z}^\star) &= \max_{i=1\ldots k} (w_i^s |f_i(\mathbf{x}) - z_i^\star|) \\ &\text{for all } s = \{1, \ldots, N\}, \\ &\text{subject to } \mathbf{x} \in S, \end{aligned} \tag{3.16}$$

where $\mathbf{w}^s$ are N evenly distributed weighting vectors. The idea behind this is that g_∞ is a continuous function of $\mathbf{w}$ [2]. The authors' hypothesis was that for N evenly distributed weighting vectors the same number of single objective sub-problems is generated and their

simultaneous solution should result in evenly distributed Pareto optimal points, assuming the objectives are normalised [2]. Here is a description of the MOEA/D algorithm,

Step 1 Generate N equally spaced $\mathbf{w}^s$ vectors and create a matrix B containing the nearest neighbours of each $\mathbf{w}^s$.

Step 2 Initialise the ideal vector, $\mathbf{z}^\star$, to some large value.

Step 3 Evaluate the decision vectors using the objective function.

Step 4 Apply genetic operators, crossover and mutation, using individuals in the neighbourhood of each solution. The choice of which individuals are to be mated is made at random.

Step 5 Evaluate the decision vectors, if any evaluation results in better value for a particular objective than in the $\mathbf{z}^\star$ vector then update it.

Step 6 Update all individuals in the population for which the following is true: $g_\infty(\mathbf{x}_{i,new}) \leq g_\infty(\mathbf{x}_{i,old})$.

Step 7 Find the non-dominated solutions in the population and store them in the archive.

As can be seen from the above procedure, each individual in the population is assigned a particular $\mathbf{w}^s$ value and a sub-problem. Also the *ideal* vector $\mathbf{z}^\star$ is estimated iteratively at execution time. A good guess for $\mathbf{z}^\star$ does help but is not a prerequisite. It should be noted that, as the authors of MOEA/D state, the use of normalised objectives helps the algorithm better distribute Pareto optimal solutions along the entire PF.

3.9 Particle Swarm Optimisation

PSO, introduced in 1995 by Eberhart and Kennedy [35, 101] was inspired by the flocking behaviour of birds and swarm theory. In PSO, as observed in nature, each agent has a rather limited repertoire of behaviours while the collective exhibits complex expressions. In the initial PSO algorithm the particles (decision vectors) use a simple update rule to update the velocity and, consecutively, the position of each particle. An archive is maintained that contains the best achieved objective function values for each particle

$$P = \{\mathbf{p}_i : i = 1, \ldots, N\}, \tag{3.17}$$

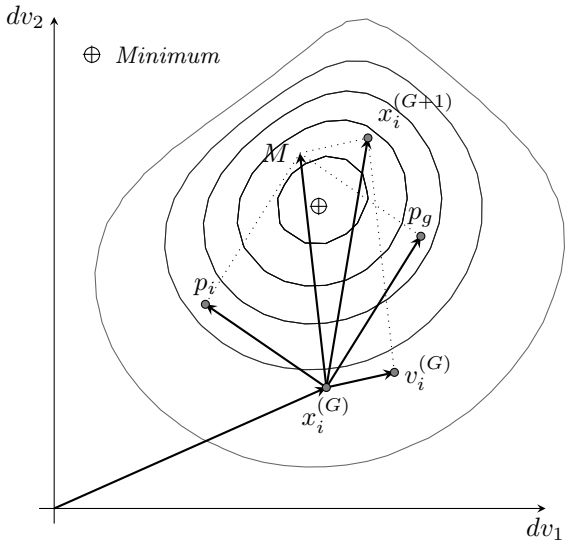

Figure 3.8: Particle update illustration in particle swarm optimisation. Where, $M = c_1\mathcal{U}(0,1)(\mathbf{p}_i - \mathbf{x}_i^{(G)}) + c_2\mathcal{U}(0,1)(\mathbf{p}_g - \mathbf{x}_i^{(G)})$ and $c_1\mathcal{U}(0,1), c_2\mathcal{U}(0,1) = 1$ for clarity. As before, dv_1 and dv_2 stand for decision variables 1 and 2 respectively.

so $f(\mathbf{p}_i) \leq f(\mathbf{x}_i^{(G)})$ and if $f(\mathbf{p}_i) > f(\mathbf{x}_i^{(G+1)})$, then $\mathbf{p}_i = \mathbf{x}_i^{(G+1)}$ and so on. Additionally, a global best position is maintained $\mathbf{p}_g$ for which the following condition must hold $f(\mathbf{p}_g) \leq f(\mathbf{x}_i^{(G+1)})$, for all $i \in \{1, 2, \ldots, N\}$. If this is not true, $\mathbf{p}_g$ is updated using the following rule $\mathbf{p}_g = \{\mathbf{x}_i : \min_i f(\mathbf{x}_i)\}$. The velocity update rule is:

$$\begin{aligned} \mathbf{v}_i^{(G+1)} = \mathbf{v}_i^{(G)} &+ c_1\mathcal{U}(0,1)\left(\mathbf{p}_i - \mathbf{x}_i^{(G)}\right) \\ &+ c_2\mathcal{U}(0,1)\left(\mathbf{p}_g - \mathbf{x}_i^{(G)}\right), \end{aligned} \tag{3.18}$$

where c_1 and c_2 are positive constants and $\mathcal{U}(0,1)$ is a random number sampled from the uniform distribution in the range $[0,1]$. A suggested value for c_1 and c_2, for problems when no prior information is available, is that both are set to 2 [101]. This would effectively result in a multiplier with a mean value of one, thus balancing in the mean the bias toward the point $\mathbf{p}_g$ and $\mathbf{p}_i$ for each particle [102]. Part of the standard PSO algorithm is illustrated in Fig. (3.8). A modification to (3.18) was presented by Shi and Eberhart [102] introducing a multiplying factor w to $\mathbf{v}_i^{(G)}$, the *inertia weight*, resulting in the following velocity update relation,

$$\begin{aligned} \mathbf{v}_i^{(G+1)} = w \cdot \mathbf{v}_i^{(G)} &+ c_1\mathcal{U}(0,1)\left(\mathbf{p}_i - \mathbf{x}_i^{(G)}\right) \\ &+ c_2\mathcal{U}(0,1)\left(\mathbf{p}_g - \mathbf{x}_i^{(G)}\right), \end{aligned} \tag{3.19}$$

where w can be constant, a function of the current generation G or even a function of a metric measuring the convergence of the algorithm. For example, the normalised hypervolume indicator

could be utilised. A value of 1 for w results in the regular velocity update rule, while a value below 1 progressively decreases the average velocity of the particles biasing the search to regions local to the particles. Alternatively, a value above 1 leads to a progressive increase in the velocity of particles resulting in a more explorative behaviour. Subsequently the position of the new particles is calculated in the following way,

$$\mathbf{x}_i^{(G+1)} = \mathbf{x}_i^{(G)} + \mathbf{v}_i^{(G+1)} \ . \tag{3.20}$$

PSO does not have an explicit selection operator, although the archive $\mathbf{p}$ could qualify as such.

Several variants of the described PSO algorithm have been devised, the interested reader is referred to [103, 104].

3.9.1 Multi-Objective Problems

The first attempts to extend PSO algorithms were presented in 2002 and at least four researchers independently suggested various methodologies to extend PSO. However the first extension seems to be due to Parsopoulos and Vrahatis [105] where the authors elected to use several variants of the weighting method and an approach similar to Shaffer's VEGA. Later the same year, Coello and Lechuga [106] introduced a Pareto-based extension which they subsequently compared with NSGA-II and PAES with similar results but lower computational cost. Another Pareto-based approach is due to Fieldsend and Singh [107]. An additional extension in 2002 was presented by Hu and Eberhart [108], where the personal best value $\mathbf{p}_i$ for the i^{th} individual is replaced only in the case when the new value dominates the previous archived value along with a technique to maintain diversity in the objective space using the Euclidean distance of objective vectors. The following year, Hu et al. [109] presented yet another extension to PSO based on the previously presented version [108] with minor improvements. In 2004 Coello et al. [110] presented a much more complete and elaborate extension of PSO with similar performance to three other evolutionary algorithms, namely NSGA-II, microGA and PAES.

Since 2004, several researchers introduced various other methodologies to extend PSO to handle MOPs but the philosophy of the applied methodologies and techniques to achieve this task does not vary significantly with respect to the already presented methods. For a comprehensive survey the reader is referred to [111].

3.9.2 First Attempt to Extend PSO to MOPs

Parsopoulos and Vrahatis [105] introduced a generalization of PSO to MOPs based on a methodology presented by Jin et al. [112] using some variations of the weighting method, see (3.4). Jin et al. [112] argued that weighting methods, or more specifically the conventional weighting method, could not approximate a concave PF due to the fact that the surface of a concave PF is *unstable* and attracts solutions toward the two extremes[1] for which the weights are $(0, 1)$ and $(1, 0)$ respectively. However, in real-world applications, the PF *shape* is usually not known a priori. To deal with this predicament Jin et al. [112] suggested that if the optimiser is run with weights of one of the two extremities of the PF, that is either $(0, 1)$ or $(1, 0)$ and then gradually or abruptly the weighting vectors are exchanged the optimiser should be able to traverse the entire PF. To capture the Pareto optimal solutions Jin et al. suggested an archiving technique that would, hopefully, result in a good approximation of the entire PF. The problem with this technique is that the optimiser is effectively segmented in two phases, the first one with the fixed weighting vectors and the second one where the population is allowed to *slide* along the PF. To successfully accomplish this task, a metric is needed to measure the convergence rate of the first phase so that the second phase is initiated. This task is not trivial because it presupposes that the minimum is already known. Another potential problem is that if disjoint regions in the decision variable space map to neighbouring objective values, this approach will have difficulty approximating the PF.

Despite these difficulties, the aforementioned approach seems to perform reasonably well for the test problems used in [105] and [112]. The proposed algorithm by Parsopoulos and Vrahatis [105] using the archiving strategy can be summarised as follows:

Step 1 Initialise the PSO population, X, and an empty archive, A; also set the weight vector to $(0, 1)$.

Step 2 Update the position of every particle according to (3.19) and (3.20).

Step 3 For each particle, update $\mathbf{p}_i$ only if the aggregated weight function of the new particle results in a smaller objective value, similarly for $\mathbf{p}_g$.

Step 4 Find the non-dominated particles in the population and add them to the archive substituting dominated solutions.

[1] In the case of two objectives.

Step 5 If a convergence criterion is not met go to **Step 2**, otherwise proceed.

Step 6 Set the weighting vector to $(1, 0)$, either gradually or abruptly and go to **Step 2**; in subsequent runs ignore **Step 5**.

At the end of the above process the archive A should contain an approximation of the PF. In a second approach Parsopoulos and Vrahatis [105] used a replication of the method used by Shaffer [12] in VEGA. The two methodologies compared and the authors concluded that their performance was very similar with no clear advantages of one over the other.

3.9.3 Multi-Objective PSO

Coello et al. [110] brought forward an improvement of a previously developed multi-objective PSO algorithm (MOPSO) [106]. The MOPSO algorithm used an archive to store non-dominated individuals and a mutation operator to enable PSO explore the whole search space; numerous suggestions are given on various effects the parameters have on the MOPSO algorithm performance. Additionally, MOPSO has the ability to handle constrained optimisation problems.

3.10 Estimation of Distribution Algorithms

In 1995, several crossover operators in GAs appeared that used probability distributions, for example, simulated binary crossover [39], unimodal normal distribution crossovers [40] and fuzzy recombination [41], to name but a few. These crossover operators, although they did not recombine more than two or three individuals from the population, did use a probability distribution that was based on the parent solutions to generate the offspring. Estimation of distribution algorithms can, in a way, be seen as an extension to the idea behind these crossover operators. The generalization is straightforward. Instead of using a crossover and mutation operator in the GA, a probability distribution over the most prominent of solutions can be estimated and then utilised to produce new individuals in the population [36]. A typical EDA, see *Alg. 3.4*, proceeds very similarly to a GA. The difference is that on every generation a subset of the population, usually the *better* half of the population, is selected and based on that subset a probabilistic model is created. Then this model is sampled and the resulting new individuals are merged with the old population while maintaining the population size constant.

While EDAs have successfully been applied to a diverse problem set, involving real and discrete decision variables, outperforming rival algorithms [113], it is not so trivial to generate the

Algorithm 3.4 Estimation of Distribution Algorithm

1: Initialise population $X^{(0)}$
2: Evaluate objective function
3: **repeat**
4: Select promising solutions X_p from $X^{(G)}$
5: Create a probabilistic model, P, based on X_p
6: Generate new solutions X_n by sampling P
7: Evaluate the objective function for X_n
8: Combine X_n and $X^{(G)}$ to create $X^{(G+1)}$
9: **until** Termination condition is met

required probabilistic model. Also, the model type strongly depends on the decision variable type, whether or not the decision variables are coupled, to what extent and in what way. Assuming that all decision variables are independent, while in fact there are multiple interdependencies, would grossly mislead the algorithm [114]. Additionally, these challenges increase in difficulty when dealing with MOPs [115],[116]. Despite these difficulties, further research in EDAs seems promising due to their inherent adaptive nature that enables them to scale well compared with other algorithms for some problems [117]. Another strong point is that prior knowledge can effectively be exploited by biasing directly the initial population [118] or by biasing the model creation procedure [119].

3.10.1 Multi-Objective Problems

In 2001 Thierens and Bosman [120] presented a simple multi-objective version of the previously presented EDA algorithm, the iterated density estimation algorithm (IDEA) [121]. The approach was simple, yet proved quite effective. The methodology used is further discussed in the next section. The following year Laumanns and Ocenasek [115], based on the Bayesian optimisation algorithm (BOA) [122], suggested a multi-objective EDA using a $(\mu + \lambda)$-strategy. In 2003 Costa and Minisci [123] used the Parzen method, a non-parametric estimation of distribution algorithm. The rest of the algorithm is very similar to the multi-objective iterated density estimation algorithm (MIDEA) by Thierens and Bosman [120], apart from the fact that, instead of a clustering algorithm, Costa and Minisci used the Parzen method to distribute solutions along the PF. An interesting approach presented in 2004 by Okabe et al. [124] is the Voronoi based EDA (VEDA). VEDA used Voronoi diagrams with the aid of clustering techniques to estimate models for disjoint regions in the decision variable space. The approach is fairly similar to the approach presented by Zhang et al. [125] in 2008 (RM-MEDA), where the authors used

certain regularity assumptions true for continuous MOPs.

3.10.2 First Attempt to Extend EDAs to MOPs

In 2000 Bosman and Thierens [121] introduced a framework, named Iterated Density Estimation Algorithm (IDEA). Based on the assumption that the problem might contain many high-order nonlinear interactions, the authors argue that even complex probabilistic models might fail to capture such behaviour, a view that seems to be gaining support [126]. Instead, a clustering technique is being used over the decision variable space and subsequently a probability distribution function is fitted over the cluster, weighted by the performance of the individuals in each cluster. It should be noted that only a certain percentage, the better performing part, was selected to participate in the described process. Later in 2001, Thierens and Bosman [120] expanded the IDEA framework to tackle MOPs, named MIDEA[1]. The main differences in the newly presented algorithm were the ranking of the individuals in the population and a clustering method used to maintain diversity in the PF. For each individual, a count is formed based on number of individuals that dominate it, so highly *fit* individuals have a relatively small *domination* count. To maintain diversity along the PF, a clustering algorithm is applied in the objective space.

3.10.3 Regularity Model-Based EDA

Another interesting approach was introduced by Zhang et al. [125]. The basis of the proposed approach is that under certain regularity conditions on the problem, used K piecewise continuous $(k-1)$-dimensional manifolds to approximate the probability model for decision vectors that result in the PF.

Zhang et al. [125] used inductively the Karush-Kuhn-Tucker condition [5] for continuous multi-objective problems, asserting that the PF of a problem with k objectives is defined by a $(k-1)$-dimensional manifold in the decision variable space. This assertion allowed Zhang et al. [125] to approximate this manifold with K piece continuous manifolds. To accomplish this task a $(k-1)$-dimensional local principal component analysis algorithm was used to partition the population in K disjoint clusters and then the centroid and its variance were estimated. The rest of the algorithm procedure is very similar to the one seen in *Alg. 3.4*.

RM-MEDA was compared, using the ZDT[2] test suite [127], with PCX-NSGA-II [128], GDE3 [99] and MIDEA [129] and, on average, outperformed those algorithms although as the authors

[1] Multi-objective Iterated Density Estimation Algorithm (MIDEA)
[2] Zitzler, Deb, Thiele (ZDT)

	Algorithm Family						
Features	GA	ES	AIS	ACO	DE	PSO	EDA
Continuous	5	5	2	1	5	5	4
Discrete	5	1	5	5	2	2	5
Mixed	1	1	-	1	1	1	1
Combinatorial	2	3	3	5	2	2	4
Complexity	4	4	3	2	5	4	1
Cost	4	4	3	3	5	5	1
No of Parameters	2	3	2	3	4	4	5
MOS	5	5	4	2	5	4	4
Prior Information	2	2	3	4	2	2	5

Table 3.1: Relative strengths and weaknesses of algorithm families for MOPs.

note, this performance comes at a much higher computational cost since a local principal component analysis algorithm has to be executed in every iteration of RM-MEDA.

3.11 Discussion

There is little doubt that evolutionary algorithms have progressed with great pace over the last 30 years, however there is little advice available to the practitioner faced with the challenge of choosing a suitable algorithm. In a way this choice is a multi-objective problem in itself, and quite a challenging one. Nevertheless, the fact that there is an absence of concrete guidelines is not accidental and can be attributed to several factors. For instance, not all the methodologies mentioned are easy to study analytically and, thus a rigorous and conclusive comparison is rather difficult. In this section, as an initial guide to the practitioner, the advantages and disadvantages of the algorithms are summarised: see Table 3.1.

Although every possible effort has been made to validate the features and their corresponding ratings in Table 3.1, the ratings presented inevitably have some bias introduced by the author, especially due to the fact that no experiments were conducted. This fact is ascribed to the sheer volume of material available relating to population-based multi-objective optimisation algorithms, rendering an attempt to thoroughly compare the different algorithms extremely difficult, if not impossible, in such a limited time frame.

The basis for the assigned ratings in Table 3.1 is now addressed. Factors that influenced these ratings are the following:

- Degree of Support for a Feature

  The degree of support for a feature is determined by how well established it is for algorithms

in a particular family. "Established" means, that the research community consistently improves upon the feature and that this process, relative to the study interval (the last 30 years), is not entirely novel. For example, AIS have a score of 2 for continuous problems since this extension is very recent [130, 131] and involves hybridization of the underlying algorithmic framework. For the same reason, all algorithmic families score poorly for mixed type decision variables (Table 3.1: Mixed), since systematic studies on this topic are extremely scarce.

- Reported Performance

  Another influencing factor is the reported performance of individual methods. For example, there is a clear way to *bias* the model building process in EDAs so as to encompass prior information when compared with any of the other algorithm families, hence they are assigned a top score in Table 3.1: Prior Information. To some extent, this argument is also true for ACO.

Explanation as to the meaning of the features in Table 3.1 is given below along with justification of the relative values attributed to each family of algorithms,

- *Continuous* - Problems with continuous decision vectors. A "-" indicates that continuous decision variables are not supported. At the other extreme, a value of 5 indicates excellent support for continuous decision variables.

  AIS-based algorithms have adopted continuous decision variables quite recently [130, 131]. However, the employed methodology resembles EDA algorithms since a Gaussian network model [130] is created to enable the underlying AIS method deal with this type of decision variable. This means that the method in [130] is in essence a hybrid of two algorithmic families, namely AIS and EDA. Admittedly, the reported performance in [131] is comparable with, and in some instances better than, the compared algorithms. For this reason, AIS score 2 in Table 3.1, since their ability to handle continuous decision variables is much better than ACO, but research involving AIS with continuous decision variables is quite sparse. ACO has relatively poor support for this type [132, 133]. EDA is ranked at 4 since its support for real decision variables is relatively recent [120, 121]. ES has supported this type since its inception [29], the same is true for PSO [35] and DE [34].

- *Discrete* - Problems with discrete decision vectors. The values have the same interpretation

as *Continuous*.

GAs initially used discrete representation [25] as well as AIS [27], EDAs [134] and ACO [135]. ES, DE and PSO have been developed for problems with continuous decisions variables [34, 35], however some extensions exist that extend these classes of algorithms to tackle problems requiring discrete decision variables, see [136] for DE, [137] for ES and [138] for PSO.

- *Mixed* - Problems that require mixed type decision variables, ie continuous and discrete. Values have the same meaning as *Continuous*.

  Since AIS does not support continuous decision variables it is disqualified from this category. Tracking published resources in this category proved challenging. Although, in the authors' experience, mixed type decision variables are absolutely necessary for real-world applications, the literature is very scarce on this topic. This fact is reflected in the relative rankings, which, for all algorithmic families, are the same. One extension of ACO to mixed type decision variables is presented in [139], and an application using EDAs in [140]. Regarding ES there are some studies dealing with mixed-integer decision variables [141, 142], however the bulk of the research is directed toward real decision variable problems. Additionally, some examples in DE and PSO are [88, 89] and [143] respectively.

- *Combinatorial* - This feature represents the ability to deal with combinatorial problems and how well; 5 indicates highly suited for tackling combinatorial problems and "-" means absolutely no support.

  To some extent all the studied algorithmic families support or have the ability to tackle combinatorial problems, although ACO has a strong lead in this type of problems [144]. Also ACO papers consistently feature in the top 10 most cited papers on combinatorial optimisation problems [145]. EDAs are ranked 4 only because there seems to be a smaller body of research compared with ACO in this category. However EDAs gain ground quite rapidly [145]. Some exemplars of successful applications can be found in [37, 113, 117]. Regarding the rest of the families and considering the fact that most are employed in combinatorial problems in a hybrid form with local search techniques [146, 147], ES and AIS are given a slight advantage due to their increased use [145]. However this point is debatable since it has been shown [2], that a framework based on decomposition, which

can utilise GAs or DEs as its main search algorithm, does perform admirably on multi-objective knapsack problems.

- *Complexity* - This refers to how complex it is to implement the *main algorithm*. The values are relative, with 1 meaning very difficult to implement and 5 a fairly straightforward implementation.

  By far the easiest techniques to implement are DE [34] and PSO [35] followed by ES [29], GAs and AIS. ACO is given a complexity of 2 due to the fact that the practitioner usually has to adapt the algorithmic process significantly to solve a particular problem [84] and EDAs are deemed to be the most complex to implement since quite elaborate techniques are required [113]. It should be noted that these ratings apply to the *main algorithm* and not to extensions to MOPs since the total complexity would be greatly affected by the selected methodology.

- *Cost* - This refers to the relative computational cost of the main algorithm, per iteration. This is intended to provide information on how easy, i.e. fast, it would be to test the algorithm and obtain some results quickly. Again, this is a comparative rating, 1 means very costly and 5 indicates an extremely low computational overhead. This does not refer to the computational time complexity of the algorithms, which is still an open issue for all but the simplest versions of the aforementioned algorithms [148, 149].

  This more or less mirrors the implementation complexity which is fairly reasonable. Although it should be stated that EDAs have much lower memory requirements [113], their computational cost per iteration is higher in comparison with the other of the algorithm families. This is especially true when more elaborate probabilistic models are created, which is the case for hierarchical-BOA [150] and RM-MEDA [125]. DE and PSO have the highest score, since their updating rules do not require elaborate calculations as can be seen in Section 3.8 and Section 3.9. Following these are ESs and GAs, which are still relatively not very demanding. For example, ES use normal distributions which are more expensive to calculate compared with the small number of addition and multiplication operations in DE and PSO, and GAs have some crossover operators that can slightly increase their cost, for example [39].

- *Number of parameters* - Indicates the relative number of parameters the algorithm requires

to be set by the user. A value of 1 translates to a high number of parameters need to be set, while a value of 5 indicates the exact opposite.

It should be emphasised that the main algorithms are assessed and not their many extensions to MOPs, with or without constraints. Here DE [34], EDAs [113] and ES [29] score highest due to their inherent adaptive nature. EDAs are awarded the top score since most of their updating rules are calculated during the algorithm execution based on some measure of optimality, and such measures can be found in abundance in statistics. For example, optimal updating rules can be formulated using the Kullback-Leibler divergence as is the case for the cross-entropy method [151]. Therefore, since there are ways to *optimally* update the *direction* of search, the need for controlling parameters is, relative to other families, less significant.

- *MOS - Multi-Objective scalability* - This is a measure of how easy it is, relative to other methodologies, to extend the *main algorithm* to tackle multiple objective optimisation problems.

  In this category, ACO is ranked 2 since it seems that extending ACO to more than 2 or 3 objectives is extremely difficult [84] or at least no attempts have been brought forward so far[1]. This is also supported by the fact that articles for multi-objective optimisation algorithms based on ACO do not appear in the top 20 cited papers [145]. For the rest, a rank of 4 was given if the methodologies to extend a particular family have been relatively recently addressed; such is the case for EDAs, PSO and AIS. It seems natural that GAs and ES score highly here since their techniques have the longest history. Additionally, regarding EDAs, the problem is that the method used to extend this family to MO problems affects the complexity of the probabilistic model. For instance, if a Bayesian network is used as in [152], it is not very easy to envisage how this will be directly extended to many objectives. However, since EDA literature is quite rich and the community is active, problems of this kind are relatively quickly addressed.

- *Prior information* - This is the ability to incorporate prior information at the start of the optimisation process and how well that information is utilised. A value of 5 signifies that the general methodology can, in principle, use prior information or expert knowledge very effectively while a value of 1 signifies the contrary.

[1] To the authors' best knowledge.

The highest scores for this feature are assigned to ACO and EDA-based algorithms. For EDAs this is due to the ease with which prior information can be incorporated [113]. This type of bias can be induced by altering the initial parameters of the probabilistic model [113]. ACO also has probabilistic transitional rules, hence to bias them towards promising regions is relatively straightforward and is common practice when the problem is formulated , for example, see [84]. In the remainder of the reviewed algorithm families, prior information about a problem has to be embedded, usually on an ad-hoc basis. For example, a GA/DE based algorithm MOEA/D [2], required significant changes in order to be applied to the multi-objective 0-1 knapsack problem.

3.12 Summary

Although the seven algorithmic families reviewed in this chapter have been considered as separate entities, it should be noted that clear boundaries cannot be explicitly drawn. For this reason, researchers start naming algorithms that fall into these categories, Evolutionary Algorithms. This choice is further justified in Section 3.3. Diffusion of information between the studied methodologies is quite rapid and common. For instance, ES have come to bear a great resemblance to GAs. Both techniques employ mutation, recombination and selection and the methods for extending GAs to multiple objective problems have successfully been applied in ES as well as in EDAs, AIS, DE and PSO. Some recombination operators in GAs have similar features to the recombination operator in DE. ACO uses probabilistic transition rules in a similar fashion to EDAs, especially when ACO has continuous decision variables.

It is also evident that more and more researchers lean towards the development of algorithmic processes that exhibit strong adaptive behaviour and provide more information than just the PF approximation at the end of the optimisation procedure. This is partly due to the fact that computational resources have become cheaper and more accessible and partly due to the ever increasing complexity of systems requiring more elaborate, effective and informative techniques. Additionally, as MOEA research moves toward many objectives, i.e. problems with more than three objectives, research is moving away from Pareto-based algorithms. This is mostly due to the difficulty that is posed in many dimensions for methods employing this ranking scheme [153, 154]. However there are still a number of issues with decomposition-based algorithms. For instance, as the number of dimensions increases the distance of the weighting vectors effectively increases, and recombination of neighbouring solutions become problematic as a result. Due

to this problem, many solutions that are potentially Pareto-optimal can be lost or remain unutilised. A suitable solution to this problem has not yet emerged and is a very interesting direction for future research. Several other approaches that seem promising have emerged, for example cone ε-dominance [19] and δ-ball dominance [155]. However these methods are quite novel and have not yet been tested for more than three objectives; further investigations are required to reveal their potential merit. Another promising research direction is that of many-objective optimisation in the presence of noise, extending to time-varying problems and real-time optimisation.

Arising from this overview, it becomes apparent that the current trend in population-based multi-objective algorithms is toward estimation of distribution algorithms and indicator-based algorithms. It is the authors' view that a combination of these two methodologies could produce interesting and innovative approaches for many-objective problems. Additionally, more attention from the research community should be ascribed to problems with mixed-type decision variables due to their frequent presence in real world problems. Another somewhat not fully developed aspect is notation, especially with regard to Pareto dominance relations, where there is no coherence in this field. Finally a unified mathematical framework seems possible and its development would be highly beneficial; some steps toward this direction can be seen in [156] and [157].

Chapter 4

Generalised Decomposition

4.1 Introduction

Decomposition-based methods have been used traditionally in mathematical programming to solve multi-objective problems [5]. These methods use scalarising functions to decompose a multi-objective problem into several single objective subproblems. These subproblems are defined with the help of weighting vectors. Weighting vectors are k-dimensional vectors with positive components that sum to one. The *location* on the Pareto front to which each subproblem tends to converge, strongly depends on the choice of weighting vectors. Therefore, the selection of an appropriate set of weighting vectors to *decompose* the multi-objective problem determines the distribution of the final Pareto set approximation along the Pareto front. Although a rigorous definition of what is considered a good distribution of Pareto optimal solutions does not exist, there is a consensus about the features that must be present. First, assuming that a decision maker is not involved prior or during the optimisation process, the general tendency is to distribute the Pareto approximation along the entire Pareto front. A second implicit requirement is that Pareto optimal solutions are distributed evenly across the entire front. This emanates from the fact that the preferences of the decision maker towards a particular region of the trade off, surface is unspecified or unknown. Finally, the distance of the Pareto set approximation must be as close as possible to the true Pareto front. Convergence of the optimisation algorithm is measured in terms of that distance.

In decomposition-based multi-objective algorithms, the first two properties, mentioned above, are directly controlled by the choice of weighting vectors. So, naturally there have been several suggestions as to their selection. However, this problem is not independent of the scalarising method used to decompose the multi-objective problem. Furthermore, this problem is nonlinear

in itself, so in most instances there is no guarantee that a unique solution can be found. In this chapter, it is shown that for a particular class of scalarising functions the weighting vectors can be *optimally* calculated, given that a clear definition of what is meant by *well* distributed Pareto optimal solutions is given. Also, knowledge of the Pareto front geometry can greatly increase the accuracy of the generated solutions, as will be explained later.

In contrast with mathematical programming, evolutionary algorithms employ mostly, although this trend seems to be slowly changing lately, Pareto-based methods to deal with multi-objective problems. However, recent studies strongly indicate that such methods become impractical when solving problems with more than three objectives. The reason for this is that as the dimensionality of the problem is increased, the ability of Pareto-based methods to discriminate solutions into *inferior* and *superior* becomes increasingly more difficult. This difficulty stems from the fact that Pareto dominance relations, the basis of Pareto-based methods, induce only a partial ordering. This means that two objective vectors can be either identical, superior or inferior with respect to the other or incomparable. In higher dimensions, the number of incomparable objective vectors increases to such levels that any meaningful selection is simply impossible [4]. An additional difficulty that Pareto-based algorithms are facing for many-objective problems is that the closer the Pareto set approximation is to the Pareto optimal front, the probability that a superior solution is generated decreases with the distance to the front [3].

These problems encountered with Pareto-based algorithms have led some researchers to more carefully investigate traditional methods, such as scalarising functions, to extend optimisation algorithms to multi-objective problems. Some examples of promising applications of decomposition-based methods for multi-objective problems are due to Jaszkiewicz [158], Hughes [95] and Zhang and Li [2]. However, a more careful inspection of the effect of the weighting vectors on the distribution of Pareto optimal solutions is considered in [95]. Most other methods either employ the paradigm presented by Das [100], that is, to evenly space the weighting vectors, or by Jaszkiewicz [158] where the weighting vectors are generated at random. What is shown in the present work is that neither approach has the capacity to yield satisfactory results according to the implicit requirements stated above.

4.2 Decomposition Methods

In the same way as Pareto-based methods, decomposition methods can be classified according to the necessary interaction with the decision maker. In this chapter the focus is *a posteriori*

methods, see Section 2.6. Therefore the aim is to generate a Pareto set approximation that portrays as faithfully as possible the entire Pareto front. As a reminder, *a posteriori* preference articulation methods operate under the assumption that the decision maker has no particular preference towards any region of the Pareto front, or if he has this information it is unknown prior to the optimisation process. Therefore a reasonable course of action is to produce solutions across the entire front, if possible. Furthermore, this assumption can be used to infer a *good* distribution of solutions across the front. For instance, some researchers assume that uniformly distributed solutions are preferable [158], while others advocate evenly distributed solutions [2]. However, a clear resolution of this matter is impossible as it depends on the decision maker, although it is apparent that the ability to change this distribution at will is potentially very helpful.

4.2.1 Scalarising Functions

So far only the weighted sum method has been introduced, see Section 2.5.3. The weighted sum method has several interesting properties, for instance if the objective function is differentiable, then so will be the scalar subproblem. However, not all Pareto optimal points can be obtained by the weighted sum method, as shown in Fig. (2.12). Furthermore, if the optimisation problem is nonconvex, there are no guarantees that the produced solutions will be Pareto optimal [5, pp. 98].

Another family of scalarising functions is based on the weighted metrics method [20]:

$$\min_{\mathbf{x}} \left(\sum_{i=1}^{k} w_i |f_i(\mathbf{x}) - z_i^\star|^p \right)^{\frac{1}{p}}. \tag{4.1}$$

Here as in (2.23), the weighting coefficients must be $w_i \geq 0$ and $\sum_{i=1}^{k} w_i = 1$, also $p \in [1, \infty)$. However, p is usually a positive integer or equal to ∞. The expression (4.1) can be read as, minimise the weighted *distance* of the objective functions f_i, where the meaning of the term - distance - depends on the chosen norm. An equivalent formulation of (4.1) is,

$$\min_{\mathbf{x}} \sum_{i=1}^{k} w_i |f_i(\mathbf{x}) - z_i^\star|^p, \tag{4.2}$$

which is commonly used since it easier to calculate [11, pp. 144] and is separable [6, pp. 294]. A potential drawback of weighted metrics based scalarising functions is that the ideal vector, $\mathbf{z}^\star$, has to be known *a priori*. However this vector can be estimated adaptively during the

process of optimisation [2]. Also the weighted metrics scalarising functions cannot guarantee Pareto optimal solutions for nonconvex problems [5, pp. 98], as is the case for the weighted sum method. For $p = \infty$ the Chebyshev scalarising function is obtained:

$$\min_{\mathbf{x}} \|\mathbf{w} \circ |\mathbf{F}(\mathbf{x}) - \mathbf{z}^\star|\,\|_\infty. \tag{4.3}$$

The $\circ$ operator denotes the Hadamard product which is element-wise multiplication of vectors or matrices of the same size. The key result that makes (4.3) very interesting is that for every Pareto optimal solution there exists a weighting vector with coefficients $w_i > 0$, for all $i = 1, \ldots, k$ [5]. This means that all Pareto optimal solutions can be obtained using (4.3). This result is quite promising, although the choice of weighting vectors is made primarily using ad hoc methods, see Section 4.2.2. Therefore direct control of the distribution of solutions on the Pareto front is very limited.

Concluding, the normal boundary intersection method (NBI) introduced by Das [100], presents another formulation of a scalarising function. The idea in NBI is that by maximising the distance of a vector normal to the simplex with vertices $\{\mathbf{v}_i : \mathbf{e}_i \circ \mathbf{z}^{nd}\}$, where $\mathbf{e}_i$ is a zero vector with the i^{th} component equal to 1, a solution that is *likely*[1] to be Pareto optimal is obtained. Using NBI the distribution of solutions on the Pareto front are directly related to the distribution of weighting vectors on the probability simplex[2], thus providing the analyst a clear path in distributing solutions on the Pareto front according to the needs of the decision maker, if these are known. The NBI method is stated as follows:

$$\begin{aligned} &\min_{\mathbf{x}} g_{nbi}(\mathbf{x}; \mathbf{w}, \mathbf{z}^\star) = d \\ &\text{subject to } \mathbf{z}^\star - d \cdot \mathbf{w} = \mathbf{F}(\mathbf{x}). \end{aligned} \tag{4.4}$$

The equality constraint in the formulation of the NBI method in (4.4) has to be satisfied in some way; a method proposed by Zhang and Li [2] is the use of a penalty function approach. Therefore (4.4) is reduced to the solution of the following:

$$\begin{aligned} \min_{\mathbf{x}} g_{nbi}(\mathbf{x}; \mathbf{w}^i, \mathbf{z}^\star) &= d_1 + p d_2 \\ d_1 &= \frac{\|(\mathbf{z}^\star - \mathbf{F}(\mathbf{x}))^T \mathbf{w}^i\|_2}{\|\mathbf{w}^i\|_2}, \\ d_2 &= \|\mathbf{F}(\mathbf{x}) - (\mathbf{z}^\star - d_1 \mathbf{w}^i)\|_2, \end{aligned} \tag{4.5}$$

where p is a tunable parameter which controls the relative importance of convergence, d_1, and position, d_2, in the penalty function. Unfortunately, (4.5) has three significant drawbacks.

[1] NBI does not have a guarantee of producing Pareto optimal solutions [100].

[2] The simplex with vertices $\mathbf{e}_i$ for all $i = 1, \ldots, k$, is commonly known as the probability simplex.

First, the normal-boundary intersection method does not guarantee that the solutions to the subproblems will be Pareto optimal [100]. Second, NBI has to be solved using a penalty method which introduces one more parameter that has to be tuned for every problem separately. Lastly, it is unclear how this decomposition method can be scaled for problems with many objectives.

4.2.2 Methods for Generating Weighting Vectors

To solve multi-objective problems using a decomposition method apart from the scalarising function, a set of weighting vectors has to be selected based on the criteria explained in Section 4.2, or perhaps other considerations pertaining to a particular problem. However the real interest is not actually in the weighting vectors but the Pareto optimal solutions that will result by solving the corresponding subproblems generated by the set of weighting vectors. So the question is, how to select the weighting vectors in such a way that the desired distribution of Pareto optimal solutions is generated. Surprisingly, this important question, appears to have mainly two answers.

The first, is to generate a set of weighting vectors that are evenly spaced. This is achieved by discretising every dimension of the objective space so that every weighting coefficient is allowed to assume any value within the set,

$$\left\{\frac{0}{H}, \frac{1}{H}, \ldots, \frac{H}{H}\right\}, \tag{4.6}$$

while ensuring that $\sum_{i=1}^{k} w_i = 1$. This approach, first seen in [11, pp. 234], has been adopted by Das [100] for use in NBI where a method to generate weighting vectors for an arbitrary number of objectives is also presented. So for a two objective problem and for $H = 2$ the set of weighting vectors is, $\{(0, 1), (0.5, 0.5), (0, 1)\}$. Although this method seems effective when combined with a normal boundary intersection scalarising function and perhaps others, its use with the Chebyshev scalarising function does not produce Pareto optimal solutions that are evenly spaced nor uniformly distributed. This can be seen in [2], and is further explored in Section 4.3.3. An alternative, based on *uniform design*, is proposed in [159]. The aim is still to generate evenly distributed weighting vectors, however using the method in [159], an arbitrary number of weighting vectors can be generated.

The second approach in generating a set of weighting vectors is due to Jaszkiewicz [158]. The idea is to generate a set of weighting vectors that are uniformly distributed on the probability simplex. The assumption is that for a uniformly distributed set of weighting vectors,

the corresponding solutions of the associated subproblems will be uniformly distributed on the Pareto front. To generate a set of weighting vectors according to the suggestions in [158], the following equation can be used:

$$\begin{gathered}\mathbf{w} = \{w_1, \ldots, w_k\}, \\ w_i = 1 - \sum_{m=1}^{i-1} w_m - (\mathcal{U}(0,1))^{i-k}\ , \text{ for all } i = 1, \ldots, k\ ,\end{gathered} \tag{4.7}$$

as many times as the required size of the weighting vector set. Here, $\mathcal{U}(0,1)$ is a sample from the uniform distribution in the domain $[0,1]$.

An interesting adaptive method to select the set of weighting vectors is presented in [160, 161]. The main idea is to identify the Pareto front geometry and then distribute a set of points on that surface in such a way so as to maximise the hypervolume indicator [44]. Subsequently, using the points from the previous step, find the weighting vectors that, upon minimisation of the resulting subproblems, would result in these points on the Pareto front. The idea seems hopeful, however, there are three major difficulties with this approach. First, the authors assume that the Pareto front can be parameterised using the following,

$$f_1^{p_1} + f_2^{p_2} = 1, \tag{4.8}$$

where, $p_i \in \mathbb{R}_{++}$ and the fact that (4.8) equals to one means that the objective functions are normalised in the range, $[0,1]$. The problem is that (4.8) is nonconvex but the authors of [160, 161] ignored this issue and used the Newton method to solve for the p_i parameters. Therefore, for Pareto front geometries where, $p_i \neq p_j$, $i \neq j$, this method will fail. This can be seen in [161] whereby a front described by: $f_1^2 + f_2 = 1$ is generated and the estimate using the Newton method is: $f_1^{1.445} + f_2^{1.445} = 1$. Therefore, the first part of the suggested method can mislead the entire procedure in [160, 161]. A solution to this problem is suggested in Section 7.4.3, however this issue is left for future research. The second problem, is that the weighting vectors that correspond to points on the identified Pareto front are formulated in a similar fashion to (4.8), hence the issue of nonconvexity of the problem formulation emerges again and the resulting weighting vectors will not produce subproblems that converge to the reference points. Lastly, the hypervolume indicator [44], which is used to ascertain the quality of the *reference* points on the PF, has exponential complexity in the number of objectives, which limits the method to approximately 4-objective problems, since the hypervolume must be calculated several times on every iteration of the algorithm. For these reasons, this method is not employed in the tests presented in Section 4.3.3.

4.3 Generalised Decomposition

As mentioned in Section 4.2, decomposition methods have two key components, first, the scalarising function and, second, a set of weighting vectors. The argument in the present work is that the choice of weighting vectors is very important with respect to the three main objectives of multi-objective optimisation. Namely, convergence to the Pareto front, coverage of the entire front and a well distributed Pareto optimal set. All these aspects are directly controlled by the choice of the set of weighting vectors that is used to decompose a multi-objective problem to a set of single objective subproblems. A method, which is referred to as *generalised decomposition*, is presented that provides an exact solution to the choice of this set of weighting vectors. The version presented here, is based on the Chebyshev scalarising function, due to its guarantee of producing a Pareto optimal solution for every weighting vector [5].

4.3.1 Optimal Selection of the Weighting Vector Set

First, it must clarified what is meant by *optimal* selection of the weighting vector set. The meaning of the term *optimal* in the present context is that, given a clear mathematical definition of what a well distributed Pareto optimal set is, and a way to measure the quality of a candidate set against this definition, then, by using generalised decomposition this quality measure can be maximised. This is subject to some prior information as is explained later.

As an example, starting with the Chebyshev scalarising function as defined in (4.3) and given a set of weighting vectors, a multi-objective optimisation problem can be decomposed in to N subproblems as:

$$\begin{aligned} \min_{\mathbf{x}} g_{\infty}(\mathbf{x}, \mathbf{w}^s, \mathbf{z}^{\star}) &= \| \mathbf{w}^s \circ |\mathbf{F}(\mathbf{x}) - \mathbf{z}^{\star}| \, \|_{\infty} \\ \forall\, s &= \{1, \ldots, N\}, \\ \text{subject to } \mathbf{x} &\in \mathbf{S}, \end{aligned} \tag{4.9}$$

with $w_i > 0$ for all $i = 1, \ldots, k$ and $\sum_{i=1}^{k} w_i = 1$, for all $\mathbf{w}^s$. For simplicity, let us assume that the ideal vector is $\mathbf{z}^{\star} = (0, \ldots, 0)$ and that the scalar objective functions are normalised in the range $[0, 1]$. This normalization implies that the nadir vector, $\mathbf{z}^{nd}$, is known. Then the question that needs to be answered for a Pareto optimal decision vector $\tilde{\mathbf{x}}$, is whether a weighting vector $\tilde{\mathbf{w}}$ exists, for which the following condition holds,

$$\begin{aligned} \| \tilde{\mathbf{w}} \circ \mathbf{F}(\tilde{\mathbf{x}}) \|_{\infty} &\leq \| \mathbf{w} \circ \mathbf{F}(\tilde{\mathbf{x}}) \|_{\infty} \\ \tilde{\mathbf{w}}, \mathbf{w} &\in \mathcal{W},\ \mathbf{F}(\tilde{\mathbf{x}}) \in \mathcal{P}, \end{aligned} \tag{4.10}$$

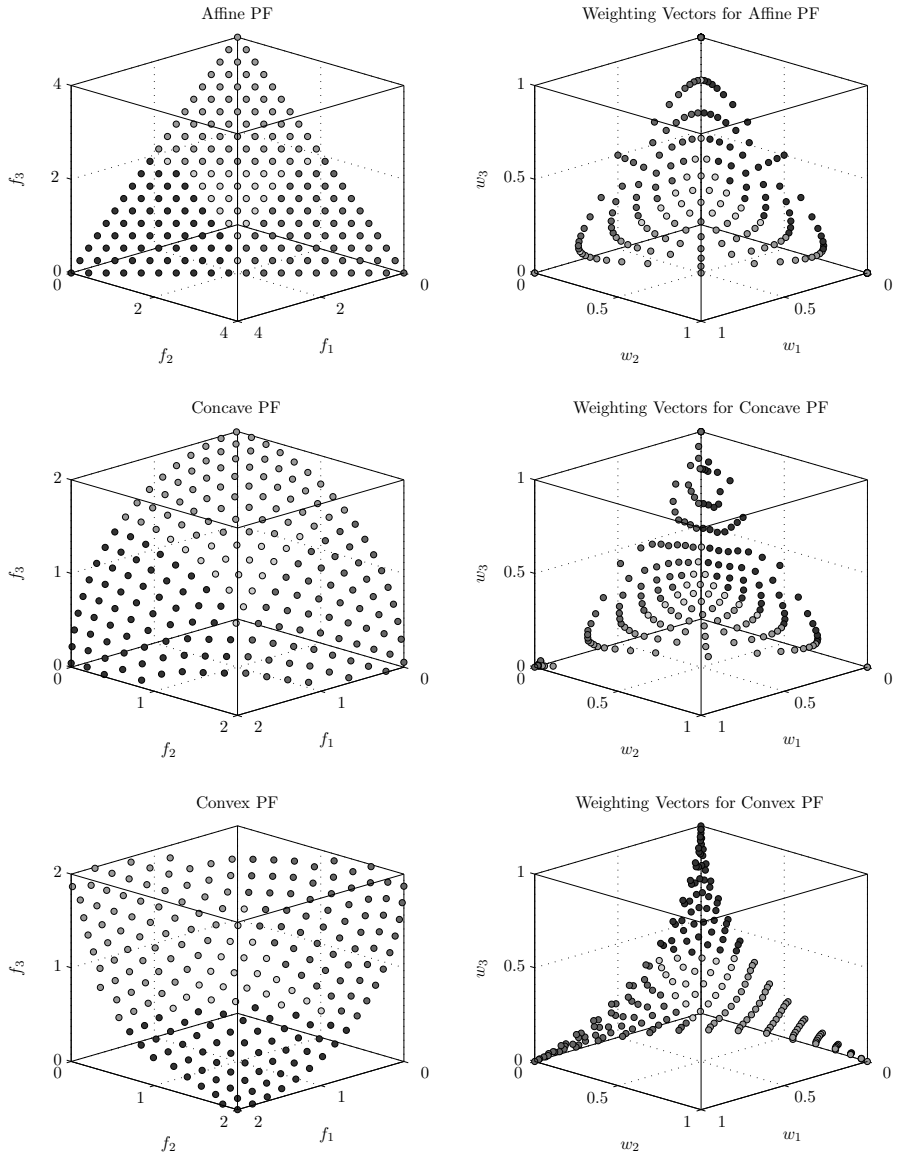

Figure 4.1: A reference Pareto front with affine geometry (**top left**) and the corresponding optimal weighting vector set (**top right**). A concave Pareto front (**middle left**) and the corresponding optimal weighting vectors (**middle right**). A convex Pareto front (**bottom left**) and the corresponding optimal weighting vectors (**bottom right**).

where $\mathcal{W}$ is the convex set defined by the vertices $\{\mathbf{e}_i : i = 1, \ldots, k\}$. If such a solution exists, it can be obtained by the solution of the following mathematical program,

$$\begin{aligned} &\min_{\mathbf{w}} \|\mathbf{w} \circ \mathbf{F}(\mathbf{x})\|_\infty, \\ &\text{subject to } \sum_{i=1}^{k} w_i = 1, \\ &\text{and } w_i \geq 0, \forall\, i \in \{1, \ldots, k\}, \mathbf{F}(\mathbf{x}) \geq 0. \end{aligned} \tag{4.11}$$

Notice that in (4.11), the optimisation is with respect to $\mathbf{w}$. In this setting, $\mathbf{F}(\mathbf{x})$ is simply a linear transformation of the vector $\mathbf{w}$, which means that since the weighting vector is convex, then the transformed vector is also part of a convex set. Additionally, all norms preserve convexity, hence the problem stated in (4.11) is convex. Subsequently there is a guarantee that a solution $\tilde{\mathbf{w}}$ exists [6], hence, this solution will satisfy (4.10). A decomposition method that selects the weighting vector set using (4.11), falls within the generalised decomposition framework. Note that the optimal weighting vectors can be obtained for the weighted metrics scalarising function for p other than infinity by using the appropriate norm in (4.11). Therefore to obtain the optimal weighting vectors for any member of the family of scalarising functions based on the weighted metrics the following problem is to be solved,

$$\min_{\mathbf{w}} \left(\sum_{i=1}^{k} w_i |f_i(\mathbf{x}) - z_i^\star|^p \right)^{\frac{1}{p}}. \tag{4.12}$$

In general, for any scalarising function that is convex with respect to the weighting vector $\mathbf{w}$, $g(\cdot)$, and any objective function whose feasible set of objective vectors is in $\mathbb{R}_+^k$ generalised decomposition can be used to obtain the optimal set of weighting vectors by solving the following mathematical program,

$$\min_{\mathbf{w}} g(\mathbf{w}; \mathbf{F}(\mathbf{x})). \tag{4.13}$$

Although there are analytical solutions for some scalarising functions, for example for the Chebyshev [95] and a point in objective space $\mathbf{z} = (z_1, \ldots, z_k)$ the optimal weighting vector is obtained by,

$$\mathbf{w} = \left(\frac{1}{z_1}, \ldots, \frac{1}{z_k} \right). \tag{4.14}$$

However, the importance of (4.13) is accentuated given the breadth of available scalarising functions designed to produce only Pareto optimal solutions and avoid weakly Pareto optimal solutions and the fact that generalised decomposition can be applied to any scalarising function that is convex with respect to $\mathbf{w}$. Two notable scalarising functions that guarantee to produce

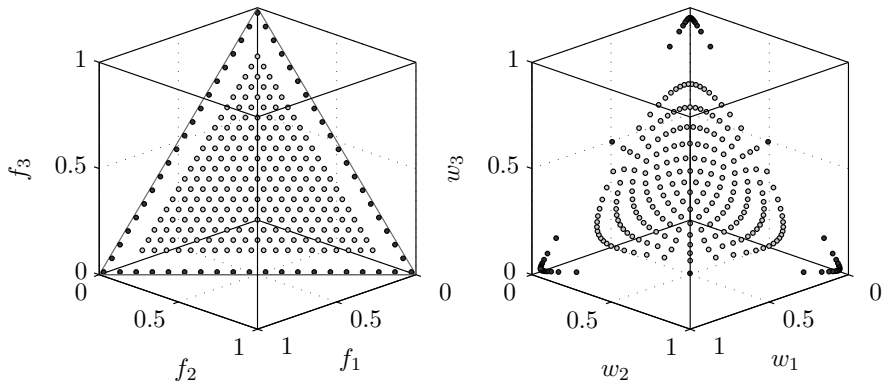

Figure 4.2: A reference Pareto front with affine geometry (left) and the corresponding optimal weighting vector set (right).

only Pareto optimal solutions are the augmented Chebyshev metric [162] and the modified Chebyshev metric [163] which is employed in Section 4.3.2 and the remainder of this chapter.

Continuing the example for the Chebyshev scalarising function, the assumption in (4.11) is, that there exists a reference Pareto set, $\mathcal{P}_r$, that exhibits the desired properties described in Section 4.2. Next, the mathematical program in (4.11) is solved for every vector in the set $\mathcal{P}_r$, thus obtaining the optimal weighting vector set. This weighting vector set can then be used with any optimisation algorithm, in order that a Pareto front with the desired properties to be obtained. An example of the application of generalised decomposition to a reference Pareto front is shown in Fig. (4.1). Any convex optimisation problem solver can be used for (4.11), however in the present work CVXGEN [1] is used as it is several orders of magnitude faster than any other solver. Some alternatives can be found in [164].

4.3.2 Practical Considerations

The problem that becomes evident with generalised decomposition is that solutions at the extremities of the Pareto front, that is Pareto optimal points where one of the objective functions is very close to 0, seem to be difficult to obtain. Although this situation is rare in practice, namely it is unusual that one of the scalar objectives in the objective vectors is reduced to zero, nevertheless it is important that the reasons behind this behaviour are understood. The cause of this behaviour is linked to the fact that a scalarising function is used and that the weighting vectors that represent solutions with one 0 component is unavoidably a weighting vector with zero components everywhere except at the location of the 0. For example, in Fig. (4.2), the left graph illustrates an affine Pareto front. The solutions that lie on the $f_1\,f_2$-plane have zero f_3

component, so they are of the form $(a, b, 0)$. Consequently, in this scenario, the mathematical program described in (4.11), will yield a weighting vector $\mathbf{w} = (0, 0, 1)$ for all solutions of the form $(a, b, 0)$. The same reasoning applies for solutions on the $f_1 f_3$-plane and $f_2 f_3$-plane. This behaviour is also present for problems with many objectives. This means that for all the solutions that lie on these planes the weighting vector that represents them is only one. In practical terms this prevents generalised decomposition from obtaining solutions of this type. This is directly linked to the fact that using the Chebyshev scalarising function, although there is a guarantee that all Pareto optimal solutions are obtainable for some weighting vector, there is no guarantee that these solutions will not be weakly Pareto optimal [5, pp. 99].

A solution to this is to modify the Chebyshev scalarising function and in extension generalised decomposition. Therefore the problem in (4.9) can be restated as [163],[5, pp. 101]:

$$\begin{gathered} \min_{\mathbf{x}} g_{\infty}(\mathbf{x}, \mathbf{w}^s, \mathbf{z}^{\star}) = \|\mathbf{w}^s \circ (|\mathbf{F}(\mathbf{x}) - (\mathbf{z}^{\star} - \epsilon)| + \rho \sum_{i=1}^{k} |f_i(\mathbf{x}) - (z_i^{\star} - \epsilon)|) \|_{\infty} \\ \forall\, s = \{1, \ldots, N\}, \\ \text{subject to } \mathbf{x} \in \mathbf{S}, \end{gathered} \tag{4.15}$$

where ρ and ϵ are sufficiently small scalars. Assuming that the scalar objective functions f_i are normalised in the range $[0, 1]$, the extra term in (4.15) will be almost a constant for all solutions in the case of a linear Pareto front geometry. For different Pareto front geometries it will vary within bounds, even if all objectives are normalised. However, the use of (4.15) will have a distorting effect on the resulting solutions, namely the obtained Pareto front will not be identical to the reference Pareto front. To avoid this, thereby preserving the desired distribution properties present in the reference front, generalised decomposition is restated as follows:

$$\begin{gathered} \min_{\mathbf{w}} \|\mathbf{w} \circ (\mathbf{F}(\mathbf{x}) + \rho \cdot C(k))\|_{\infty}, \\ \text{subject to } \sum_{i=1}^{k} w_i = 1, \\ \text{and } w_i \geq 0,\, \forall\, i \in \{1, \ldots, k\}, \mathbf{F}(\mathbf{x}) \geq 0, \end{gathered} \tag{4.16}$$

where ρ is a small scalar, as in (4.15), and $C(k)$ is a linear function of the number of objectives k. Intuitively the effect of the $\rho \cdot C(k)$ term is that it shifts the reference Pareto front slightly. This in extension eradicates solutions that have identically zero components and preserves the relative position of solutions in the reference Pareto front. The penalty for this modification is that all resulting solutions using the weighting vector set produced by (4.16) will be slightly

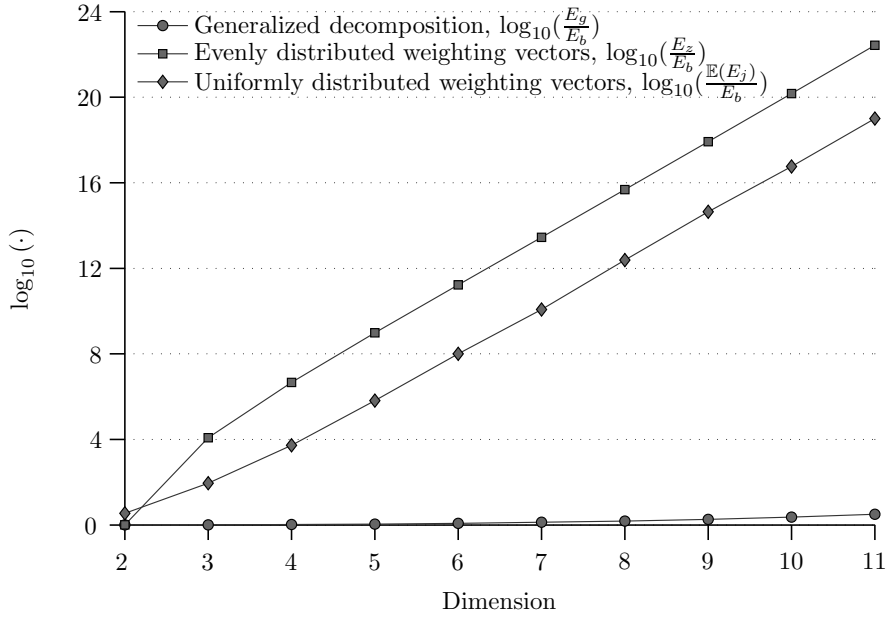

Figure 4.3: The $\log_{10}$ energy ratio of Pareto optimal solutions obtained according to the three methods for generating weighting vectors.

closer to one another. This effect is directly controlled by ρ and $C(k)$, and can be as small as the machine precision allows for.

An alternative to the modification of generalised decomposition seen in (4.16), is to simply remove the solutions in the reference Pareto front that have one zero component. This way the original definition, seen in (4.11), can be used. This alternative approach requires fewer parameters for its operation as ρ and $C(k)$ become unnecessary.

4.3.3 The Effect of Weighting Vector Choice in Many Objective Problems

In Section 4.3 it is stated that the choice of the weighting vector set is very important, and that this set directly controls the distribution of produced Pareto optimal points by a multi-objective optimisation algorithm. To test this hypothesis, first a definition and a measure of well distributed Pareto optimal solutions is required. A measure that is in common use for evenly distributed points on k-dimensional manifolds, is the Riesz kernel or s-energy [165], defined as:

$$\begin{aligned} E(\mathbf{Z}; s) &= \sum_{1 \leq i \leq j \leq N} \|\mathbf{z}_i - \mathbf{z}_j\|^{-s},\ s > 0 \\ \mathbf{z} &\in \mathbb{R}^k,\ \text{and},\ \mathbf{Z} = \{\mathbf{z}_i : i = 1, \ldots, N\}. \end{aligned} \tag{4.17}$$

It has been shown that for a k-dimensional manifold the s-energy is minimised when the distribution of points on that manifold is even, if $s \geq k$ [165]. Therefore since the Pareto front

Table 4.1: The number of objective vectors, N, for constant H used in the experiment seen in Fig. (4.3).

Obj. #	2	3	4	5	6	7	8	9	10	11
s	1	2	3	4	5	6	7	8	9	10
H	8	8	8	8	8	8	8	8	8	8
N	8	36	120	330	792	1716	3432	6435	11440	19448
$\rho \cdot C(k)$	0.001	0.002	0.003	0.004	0.005	0.006	0.007	0.008	0.009	0.01

of a k-objective problem is a $(k-1)$-dimensional manifold [2], the s parameter in the s-energy metric used for the following experiment is set to $k-1$, see Table 4.1. Generalised decomposition is compared with the methods suggested by [100], and later used by [2], that is, evenly distributed weighting vectors and the method suggested by Jaszkiewicz [158], namely the selection of a weighting vector set generated according to (4.7). The results, Fig. (4.3), are obtained according to the following procedure:

Step 1 A Pareto front with affine geometry has been selected for all test instances. This is mainly due to the fact that it is straightforward to generate a Pareto front of this geometry with the optimal distribution of solutions, so that these can be used as a reference. The way that these reference fronts have been generated is identical to the generation of weighting vectors, described in Section 4.2.2. For an example of a Pareto front with this geometry in 3 dimensions, see Fig. (4.1). Also, this enables a fair comparison with the scheme employed by Zhang and Li [2]. The number of solutions, N, generated in every dimension, is controlled by the H parameter (see Section 4.2.2. Since this parameter can be seen as the number of subdivisions per dimension, it has been kept constant, see Table 4.1, for all dimensions. The reason for this setting is to isolate only the effect that the dimensional increase has on the s-energy, and by extension the distribution of solutions on the Pareto front.

Step 2 For every dimension, a set of weighting vectors is generated according to the suggestions in [158] and [2]. Since the method suggested by Jaszkiewicz [158] is generating weighting vectors according to the uniform distribution, the s-energy calculated as described in the next step, is averaged over 50 independent weighting vector sets of size N and for every dimension. The weighting vector set for generalised decomposition is generated according to (4.16), with $\rho \cdot C(k)$ set as seen in Table 4.1. The reference Pareto front used is the affine front. As this is not always the case a default Pareto front to generate the weighting vectors can be a useful starting point. For details regarding this issue, see Section 4.3.4.

Step 3 Another benefit of using an affine Pareto front geometry, is that the solutions that will minimise the subproblems defined by the weighting vector set can be directly calculated, by solving the following mathematical program:

$$\begin{aligned} &\min_{\mathbf{F}(\mathbf{x})} \|\mathbf{F}(\mathbf{x}) \circ \tilde{\mathbf{w}}\|_\infty, \\ &\text{subject to } \sum_{i=1}^{k} f_i = 1, \\ &\text{and } f_i \geq 0 \,, \forall\, i \in \{1, \ldots, k\}. \end{aligned} \tag{4.18}$$

Note that (4.18) is a convex problem for the same reasons described in Section 4.3.1. Therefore, using (4.18) and a set of weighting vectors, the s-energy can be calculated for the resulting Pareto set. Also, a baseline energy using the actual Pareto front, E_b, is calculated for every problem instance.

Step 4 Concluding, the $\log_{10}$ ratio of the obtained s-energy according to every method and the base energy E_b is calculated for all objectives. The only exception is for the method proposed by Jaszkiewicz [158], where the expected energy, $\mathbb{E}(E_j)$ is used. The reason for this is explained in **Step 2**.

From the results seen in Fig. (4.3), it is apparent that generalised decomposition can follow very closely the desired distribution of solutions in the Pareto set. Additionally, the difference with alternative methods is striking, namely in the range of several orders of magnitude for problems with 3 or more objectives. These results refute the hypothesis of Zhang and Li [2] that by selecting an evenly distributed set of weighting vectors a *well*[1] distributed Pareto front can be obtained. Furthermore, it is shown that the method proposed by Jaszkiewicz [158], performs consistently better compared with evenly distributed weighting vectors.

The increasing ratio of the s-energy produced by solutions selected using evenly distributed weighting vectors can provide an explanation for the reason that MOEA/D [2] and derivative algorithms seem to perform well in many-objective problems. Namely, for an increasing number of objectives MOEA/D-based algorithms find solutions that are more clustered. This means that, relative to the entire Pareto front *area*, such algorithms only focus on a very small part of the front.

[1] Although the authors of [2] do not explain what they mean by *well* distributed Pareto optimal points, their selection scheme for the weighting vectors does not result in well distributed Pareto optimal solutions by any commonly used convention.

4.3.4 Reference Pareto Front

A limitation of generalised decomposition seems to be that, since a reference Pareto set is needed to generate the optimal weighting vectors, if that reference is unavailable due to lack of information about the Pareto front geometry, then the method cannot be employed. However, this is not entirely true. In the presence of a reference Pareto optimal set, the precision that can be achieved, regarding the desired distribution is exceptional. Although, even if an affine Pareto front is used to generate the weighting vectors, very good results can still be obtained. Good in the sense that the distribution of solutions on the Pareto front is close to even. This is because the affine Pareto front geometry seems to be in the middle ground, with respect to the shift in location of the weighting vectors, of concave and convex geometries. Although this hypothesis seems intuitive, it is still untested. Further investigation of that matter is left for future research.

To test whether a good distribution of solutions can be obtained using an affine Pareto front geometry as a reference for generalised decomposition, the implemented algorithm is primarily based on the description of MOEA/D [2]. The main differences are that the neighbourhood distance is measured in objective space, instead of the weighting space as is the case in MOEA/D. Additionally, the generation of weighting vectors is as follows:

- Generate N evenly distributed points on the probability simplex according to the method described in [100]. Subsequently use generalised decomposition, (4.11) or (4.16), to generate the weighting vector set. The evenly distributed points are used as the reference Pareto front.

This algorithm is referred to as a many-objective evolutionary algorithm based on generalised decomposition (MAEA-gD). This is because, as demonstrated in Section 4.3.3, generalised decomposition scales very well to many objectives. The results are seen in Fig. (4.4) for the 3-objective instances of the test problems DTLZ1 and DTLZ2 [166]. The results shown in Fig. (4.4) are indicative, but not conclusive. In the next chapter, thorough tests are performed.

4.4 Summary

A new concept has been presented, namely generalised decomposition (gD). Using gD the optimal distribution of solutions across the Pareto front can be achieved, given the geometry of

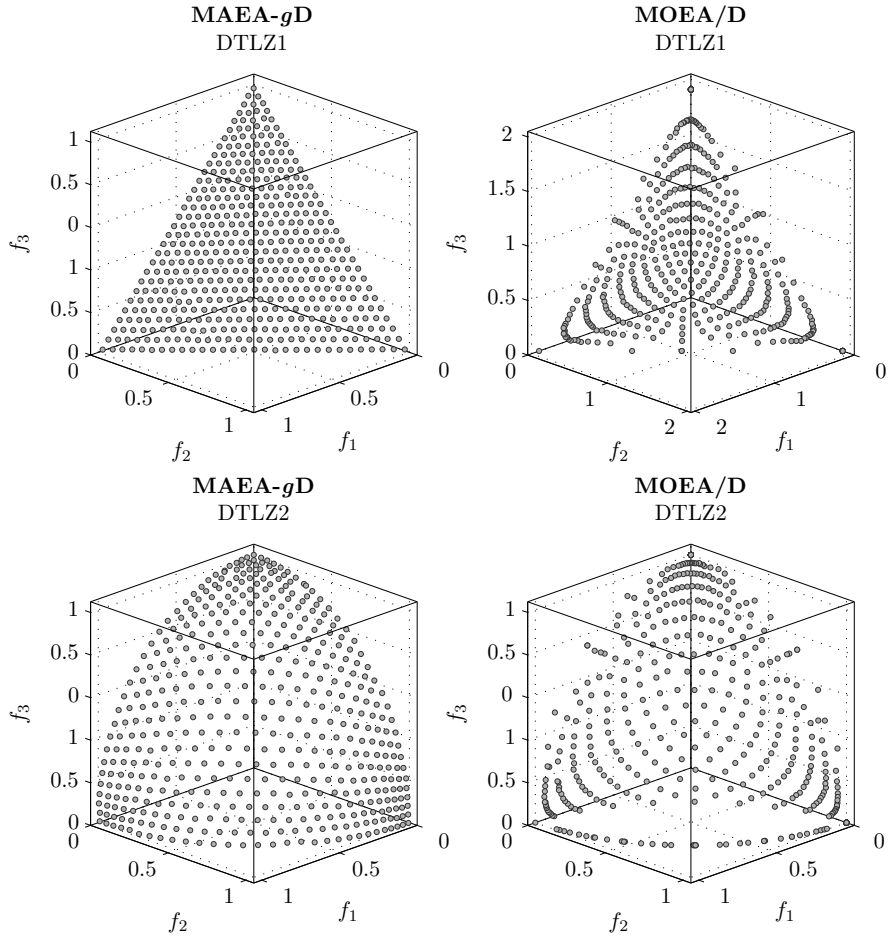

Figure 4.4: Attained Pareto optimal front for the DTLZ1 and DTLZ2 3-objective problems by MADE-gD and MOEA/D.

the front is known *a priori* and that the selected scalarising function is convex with respect to the weighting vector set. Furthermore, the obtained results suggest that the method maintains its favourable qualities even for many objectives. In contrast, the available methods perform substantially worse, in the sense that they are unable to produce evenly distributed Pareto optimal solutions even for 3 objectives. This is supported by the fact that the alternative methods have several orders of magnitude larger s-energy. Also, gD is not limited to producing evenly distributed solutions. Given a definition and a measure of what is meant by a *well* distributed Pareto set, generalised decomposition can produce optimal results according to that definition.

However the assumption that knowledge of the Pareto front geometry is available is to an extent restrictive. For this reason, it is suggested that the weighting vector set be produced using an affine Pareto front geometry as the reference front for generalised decomposition. Following this a modified version of the MOEA/D algorithm was created producing clearly superior results

for the given test set, when compared with the original version of MOEA/D.

Chapter 5

Generalised Decomposition for Many-Objective Optimisation

5.1 Introduction

Pareto-based methods are of limited utility when the number of dimensions is increased [153]. This is primarily because the number of non-dominated solutions increases as the dimensionality of the problem increases, and for dimensions greater than approximately 10 almost all the solutions are non-dominated [4]. Hence this type of partial ordering becomes of limited use in high dimensions since, if all the generated solutions are non-dominated, the EA has no objective measure on which to base its selection process. An alternative is the use of decomposition-based methods, see Chapter 4. Arguably, decomposition methods have not been explored in sufficient depth in relation to MAPs. For example, most researchers, with a few exceptions such as [66, 95, 96], have assumed that an even or uniform distribution of weighting vectors will result in an even distribution of Pareto optimal points, which is shown in Section 4.3.3 to be false. Therefore, the primary objective of this chapter is to explore the benefits and potential difficulties that a generalised decomposition algorithm may have for many-objective problems.

Evolutionary algorithms (EAs) have found numerous applications in MAPs [4]. This is because most EAs are population-based, in the sense that at each iteration an entire population of solutions is evaluated. This feature is quintessential to MAPs since, in *a posteriori* preference articulation, an entire family of solutions is required to describe the entire trade-off surface. This trade-off surface in objective space is also called Pareto front (PF). Another important reason for EA applicability is that they impose almost no constraints on the problem structure, for example continuity and differentiability are not required for EA operation. Due to these factors, MAP research is vibrant in the EA community, something that can be attested by the number

of EAs available for MAPs [2, 4, 52]. Specifically, EAs are comprised of a number of algorithm families, such as genetic algorithms (GAs) [25] and evolution strategies (ES) [30], as well as differential evolution (DE) [35] and others. Most of the aforementioned algorithm families are inspired by some naturally occurring process, such as DNA recombination and mutation [25]. However this presents certain difficulties, for example, it is very hard to analyse the behaviour of MOEAs analytically, thus their performance on a problem cannot be guaranteed prior to application. This is why EAs are usually evaluated experimentally using some test problem set [127, 166, 167].

More recently, a new family of algorithms has emerged, namely estimation of distribution algorithms (EDAs). EDAs stand in the middle ground between Monte-Carlo simulation and EAs. In EDAs a probabilistic model is built based on elite individuals, which subsequently is sampled producing a new population of *better*[1] individuals. From the EA point of view, EDAs can be traced back to recombination operators based on density estimators that use good performing individuals in the population as a sample [36]. The positive aspects of EDAs are that it is straightforward to integrate prior information into the optimisation procedure, thus reducing the time to convergence if such information is available. Also, the amount of heuristics, compared to other EAs, is reduced, easing the task of mathematical analysis of these algorithms. This is an important aspect which has been overlooked, due to inherent difficulties, in most heuristics for optimisation. Studies of this kind are usually applied to algorithms that are not used in practice [148, 149], therefore the practical value of such studies is limited. However EDAs are not a panacea since they heavily depend on the quality and complexity of the underlying probabilistic model [113]. For instance a simple EDA based on low order statistics, i.e. an EDA that does not account for variable dependencies, can be easily misled if, in fact, such dependencies exist in the underlying problem. To overcome such difficulties researchers proposed ever more elaborate models [113], which, of course, increase the complexity of the algorithm and, in some instances, the identification of the optimal model is of comparable complexity to that of the optimisation problem necessitating the use of heuristics in the model identification and training [168]. Acknowledging this problem has led some researchers to suggest hybridization of EDAs based on simple probabilistic models with some form of clustering [126]; this course is further supported by more recent studies [169].

The Cross Entropy method (CE) seems to be a good candidate as the main algorithm in

[1] Or more precisely, individuals that are more likely to be better than their predecessors.

the presented generalised decomposition based framework. The CE-method was introduced by Rubinstein [170], initially as a rare event estimation technique and subsequently as an algorithm for combinatorial and continuous optimisation problems. The most alluring feature of CE is that for a certain family of instrumental densities the updating rules can be calculated analytically, and are thus extremely efficient and fast. Also the theoretical background of CE is enabling theoretical studies of this method which can provide sound guidelines about the applicability of this algorithm to problem types.

5.2 Cross Entropy Method

The cross entropy method (CE) was introduced by Rubinstein [170], for single objective continuous and discrete optimisation problems. In its original form, CE was based on Kullback-Leibler cross-entropy, importance sampling and the Boltzmann distribution for continuous problems, while Markov chains are employed in the discrete case. It is interesting to note that in this form CE is similar, in principle, to probability collectives (PC), a method introduced by Wolpert et al. [171] for distributed control and optimisation.

In CE, the optimisation problem is cast as a rare event estimation and subsequently an adaptive technique, with the aid of importance sampling, is applied to update the parameters of an instrumental density. The derived problem is called the *associated stochastic problem* (ASP). The method then uses the ASP to implicitly solve the original optimisation problem. Generally speaking there are two steps involved in this iterative procedure:

- Generate a population[1] based on a prior distribution, g. The distribution g is uniquely defined by a parameter vector v. In the initial iterations of the algorithm it is usually the uniform distribution that is used, unless prior knowledge suggests otherwise.

- Update the parameter vector, v, to create the posterior distribution using an elite subset, $\mathcal{E}$, of the previous population.

Since its introduction, several studies expanding on the initial methodology have been presented. Most notably, the minimum cross-entropy (MCE) method [172], where a non-parametric instrumental distribution is used. Albeit, MCE is computationally more demanding compared with CE. Another interesting approach to extend CE is presented by Botev [173], termed generalised cross entropy (GCE). In GCE, quite elegantly, the ASP is transformed to a convex

[1]Note that the terms *population* and *samples* are used interchangeably in this work; unless stated otherwise.

program with the help of the χ^2 directed divergence. GCE overcomes the specification bias by using non-parametric density estimation. However, the computational cost of GCE is prohibitive when used in an optimisation setting.

Let us assume that the optimisation problem has a single objective which is to be minimised,

$$\min_{\mathbf{x}} f(\mathbf{x}) \tag{5.1}$$

where $\mathbf{x}$ is the decision variable vector and $f(\mathbf{x}^\star) = \gamma^\star$ is the minimum. Assuming $\mathbf{x}^\star$ is *rare* in S, (5.1) can be interpreted in a different way, i.e. as a rare event estimation. Rare in this context means that for, $C = \{\mathbf{x} : \|\mathbf{x}^\star - x\|_2 \leq \varepsilon, \varepsilon > 0\}$ and ε *small*, then the probability, $\mathbf{P}(\mathbf{x} \in C) = \int_C u(\mathbf{x}) d\mathbf{x} \ll 1$, where, u, is a density function. Therefore (5.1) can be restated as follows,

$$\mathbb{E}_{\mathbf{u}} I_{f(\mathcal{X}) \leq \gamma} = \mathbf{P}_{\mathbf{u}}(f(\mathcal{X}) \leq \gamma) = \ell, \tag{5.2}$$

ℓ is the probability of the *rare event*, I is the indicator function and $\mathbb{E}_{\mathbf{u}}$ is the expectation of a quantity distributed according to the density $g(\cdot; \mathbf{u})$ and $\mathbf{P}_{\mathbf{u}}(f(\mathcal{X}) \leq \gamma)$ is the probability for the function $f(\cdot)$ to have a smaller value than γ when the random variable vector $\mathcal{X}$ is distributed according to the density $g(\cdot; \mathbf{u})$. The random variable vector, $\mathcal{X}$, is associated with the decision variable vector $\mathbf{x}$. For notational compactness, $H(\mathcal{X}; \gamma) \equiv I_{f(\mathcal{X}) \leq \gamma}$ is defined as,

$$H(\mathcal{X}; \gamma) = \begin{cases} 1 & f(\mathcal{X}) \leq \gamma \\ 0 & f(\mathcal{X}) > \gamma. \end{cases} \tag{5.3}$$

Now, to estimate ℓ for some $\tilde{\gamma}$ such that $\|\tilde{\gamma} - \gamma^\star\| \leq \epsilon$, with ϵ small, $\mathbf{P}_{\mathbf{u}}(H(\mathcal{X}; \tilde{\gamma}))$ must be solved, which is non-trivial if the initial assumption is true, i.e. that the probability $\mathbf{P}_{\mathbf{u}}(H(\mathcal{X}; \tilde{\gamma}))$ is small when $\mathcal{X} \sim g(\cdot; \mathbf{u})$. In the trivial case that the aforementioned assumption is not true ℓ can be estimated using the *crude Monte Carlo* (CMC) estimator,

$$\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} H(\mathcal{X}; \gamma). \tag{5.4}$$

If, however, the prior assumption holds, then the indicator function $I_{f(\mathcal{X}) \leq \rho}$ in (5.4) will most likely be identically 0 for all $\mathcal{X}_i$, so a different approach is necessary. An alternative to CMC is the *importance sampling* (IS) [174] estimator which is defined as follows,

$$\hat{\ell} = \frac{1}{N} \sum_{i=1}^{N} W(\mathcal{X}_i; \mathbf{u}, \mathbf{v}) H(\mathcal{X}; \gamma), \tag{5.5}$$

where $W(\mathcal{X};\mathbf{u},\mathbf{v}) = \frac{g(\cdot;\mathbf{u})}{g(\cdot;\mathbf{v})}$ is the *likelihood ratio* (LR). Now the problem is to find the IS density $g(\cdot;\mathbf{v})$ that would minimise the variance of the estimator; theoretically the zero variance density is:

$$g^{\star}(\mathbf{x}) = \frac{f(\mathbf{x};\mathbf{u})H(\mathcal{X};\gamma)}{\ell}. \tag{5.6}$$

However (5.6) involves the quantity which we are trying to estimate (ℓ), hence its practical value is limited. Although it is possible, up to a multiplicative constant, to attempt to minimise the "distance" of $g(\cdot;\mathbf{v})$ with $g^{\star}(\cdot)$. And for this purpose, a convenient measure of "distance" is the Kullback-Leibler *distance* (KL), defined as:

$$\mathcal{D}(g,h) = \int g(\mathbf{x}) \ln\left(\frac{g(\mathbf{x})}{h(\mathbf{x})}\right) d\mathbf{x} \tag{5.7}$$

and upon expansion,

$$\begin{aligned}\mathcal{D}(g,h) = &\int g(\mathbf{x}) \ln g(\mathbf{x})\, d\mathbf{x} \\ &- \int g(\mathbf{x}) \ln h(\mathbf{x})\, d\mathbf{x}.\end{aligned} \tag{5.8}$$

Since the first term in (5.8) is constant, we only need to minimise the second term which is equivalent to maximising $\int g(\mathbf{x}) \ln h(\mathbf{x})\, d\mathbf{x}$. Therefore the optimal parameter vector $\mathbf{v}^{\star}$, in the minimum variance sense, is obtained by the solution of the following program:

$$\mathbf{v}^{\star} = \max_{\mathbf{v}} \mathbb{E}_{\tilde{\mathbf{v}}} H(\mathcal{X};\gamma) W(\mathcal{X};\mathbf{u},\tilde{\mathbf{v}}) \ln g(\mathcal{X};\mathbf{v}), \tag{5.9}$$

where $\mathcal{X}$ is independent and identically distributed (i.i.d) according to $g(\cdot;\tilde{\mathbf{v}})$. However $\mathbf{P}_{\mathbf{u}}(H(\mathcal{X};\gamma))$ is still a rare event; in CE this is overcome by substitution of γ with $\bar{\gamma} \geq \gamma$ equal to the ρ-quantile of $f(\mathcal{X})$ under $\mathbf{v}$. The program in (5.9) is solved for decreasing levels of $\bar{\gamma}$ until $\bar{\gamma} \leq \gamma$. So (5.9), in the CE-method, becomes:

$$\mathbf{v}_t = \max_{\mathbf{v}} \mathbb{E}_{\mathbf{v_{t-1}}} H(\mathcal{X};\gamma_{t-1}) W(\mathcal{X};\mathbf{u},\mathbf{v}_{t-1}) \ln g(\mathcal{X};\mathbf{v}), \tag{5.10}$$

whose stochastic counterpart is,

$$\mathbf{v}_t = \max_{\mathbf{v}} \frac{1}{N} \sum_{i=1}^{N} H(\mathcal{X}_i;\gamma_{t-1}) W(\mathcal{X}_i;\mathbf{u},\mathbf{v}_{t-1}) \ln g(\mathcal{X}_i;\mathbf{v}), \tag{5.11}$$

$\mathcal{X}_1,\ldots,\mathcal{X}_N$ are drawn from $g(\cdot;\mathbf{v}_{t-1})$. Typically (5.11) is convex and if the instrumental densities $g(\cdot;\cdot)$ are chosen from the *natural exponential family* (NEF) [151] then (5.11) can be solved analytically [172] by solving the following system of equations:

$$\max_{\mathbf{v}} \frac{1}{N} \sum_{i=1}^{N} H(\mathcal{X}_i) W(\mathcal{X}_i;\mathbf{u},\mathbf{v}_{t-1}) \nabla_{\mathbf{v}} \ln g(\mathcal{X}_i;\mathbf{v}) = 0 \tag{5.12}$$

This is a major strength in CE, the fact that the updating rules for the instrumental densities can be obtained analytically translates to a much lower computational overhead. Briefly, some distributions in the NEF family are the Gaussian, Poisson and the gamma distributions [175]. The procedure described by (5.10)-(5.12) will generate a sequence of γ values: $(\gamma_t > \gamma_{t+1} > \cdots > \gamma_{t+m})$ with the corresponding instrumental densities converging to the optimal parameter $\mathbf{v}$ for which the event $\mathbf{P}_{\mathbf{u}}(H(\mathcal{X};\tilde{\gamma}))$ is increasingly easier to estimate, i.e. it becomes more *likely* under the density $g(\cdot;\mathbf{v})$.

5.2.1 CE-Method for Continuous Optimisation

The procedure described so far is directly applicable to optimisation problems with the only difference being that the level γ is either the *a priori* minimum of the objective function $f(\cdot)$ or if this information is not available it is allowed to decrease ad infinitum. In practice, for bounded problems, the sequence $\{\gamma_t, \gamma_{t+1}, \dots\}$ converges to a value close to the minimum, hence the stopping criterion can be set to $|\gamma_t - \gamma_{t-1}| \leq \delta$ for some small δ.

A good candidate for the instrumental densities is the normal distribution,

$$g(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \tag{5.13}$$

and its truncated equivalent for problems with boundary constraints. It should be mentioned that the updating rules derived using (5.12) are identical for the regular and truncated Gaussian [173].

It is suggested in [172] that for the optimisation case IS is not very useful since the initial parameter $\mathbf{u}$ in the density $g(\cdot;\mathbf{u})$ is actually arbitrary, under the assumption that no information is available about the location of the optimum. However, such information may be available, hence maintaining the IS estimator allows prior information to be exploited. This can be achieved by setting the parameters $\mathbf{u}$ according to the information available, which should in turn help steer the search near optimal solutions faster. On the down side, if the prior information is not correct, this biasing can lead the optimisation procedure astray.

The CE-method for single objective problems can be summarised as follows:

Step 1 Initialise $\mathbf{v}_0$ to the uniform distribution and set $t = 1$.

Step 2 Sample the distribution $g(\cdot;\mathbf{v}_{t-1})$ to generate a random sample of size N and evaluate the objective function $f(\cdot)$.

Step 3 Select the top ρN performing samples and use them to estimate $\mathbf{v}_t$. Solving (5.12) the updating rules are obtained for the normal distribution $\mathbf{v}_t = \{\mu_t, \sigma_t\}$:

$$\hat{\mu}_t = \frac{\sum_{i=1}^{\rho N} W(\mathcal{X}_i; \mathbf{u}, \mathbf{v}_{t-1})\mathcal{X}_i}{\sum_{i=1}^{\rho N} W(\mathcal{X}_i; \mathbf{u}, \mathbf{v}_{t-1})}, \tag{5.14}$$

$$\hat{\sigma}_t = \left(\frac{\sum_{i=1}^{\rho N} W(\mathcal{X}_i; \mathbf{u}, \mathbf{v}_{t-1})(\mathcal{X}_i - \hat{\mu})^2}{\sum_{i=1}^{\rho N} W(\mathcal{X}_i; \mathbf{u}, \mathbf{v}_{t-1})} \right)^{\frac{1}{2}}, \tag{5.15}$$

where ρ is some small value, e.g. 0.1. The updating rules in (5.14) and (5.15) could lead to premature convergence [172] so a *smoothed* version is usually employed:

$$\begin{aligned} \mu_t &= \alpha\hat{\mu}_t + (1-\alpha)\mu_{t-1} \\ \sigma_t &= \beta_t\hat{\sigma}_t + (1-\beta_t)\sigma_{t-1}, \end{aligned} \tag{5.16}$$

where α and β_t are smoothing parameters with $\alpha \in (0.7, 1)$ and β_t is calculated as:

$$\begin{aligned} \beta_t &= \beta - \beta\left(1 - \frac{1}{t}\right)^q \\ \beta &\in (0.7, 1) \\ q &\in (5, 9). \end{aligned} \tag{5.17}$$

Step 4 If the stopping condition is not met go to **Step 2**, otherwise output the current μ_t as the estimate of the location of the optimum.

5.3 Generalised Decomposition-Based Many Objective Cross-Entropy

The proposed algorithm is based on the CE-method, see Section 5.2, and the newly introduced concept of generalised decomposition as described in Section 4.3. However, two versions have been introduced, namely many objective CE (MACE) and MACE based on generalised decomposition (MACE-gD). The difference is that the weighting vectors $\mathbf{w}$ in MACE are generated according to the suggestions in [2] to enable a clearer comparison with the MOEA/D framework and to evaluate the benefits and potential shortcomings of generalised decomposition. Therefore MACE employs a set of evenly spaced weighting vectors to further test validity of the hypothesis that this scheme does not result in an *even* distribution of Pareto optimal solutions on the PF, see Section 4.3.3. It is shown how such issues can be overcome using MACE-gD and present a method that can prove invaluable when the optimisation problem has many objectives. The general idea is that a set of weighting vectors can be generated near regions that are of interest,

thus avoiding waste of resources in search of Pareto optimal solutions away from such regions. The *main* algorithm in MACE and MACE-gD is the CE-method for continuous optimisation problems as described in Section 5.2.1. An overview of MACE-gD can be seen in *Alg. 5.1.* In

Algorithm 5.1 MACE-gD

1: $\mathbf{w} \leftarrow gD(\text{PF Shape})$ ▷ Initialise the weighting vectors using generalised decomposition.
2: $\mathcal{M}^{(1)} \leftarrow \min \mathbf{x} + \mathcal{U}(0,1)(\max \mathbf{x} - \min \mathbf{x})$ ▷ If possible, initialise the means in the feasible set.
3: $\mathcal{S}^{(1)} \leftarrow C(\max \mathbf{x} - \min \mathbf{x})$
4: $\mathbf{x}^{(1)} \leftarrow \mathcal{N}(\mathcal{M}, \mathcal{S})$
5: $\mathbf{E} \leftarrow \mathbf{F}(\mathbf{x}^{(1)})$
6: $\mathbf{z}^{\star} \leftarrow \min\{\mathbf{E}_{f_1}, \ldots, \mathbf{E}_{f_k}\}$
7: $t \leftarrow 1$
8: **repeat**
9: **for** $i \leftarrow 1, N$ **do**
10: $\mathbf{V}^{(t)} \leftarrow g(\mathbf{x}^{(t)}, \mathbf{w}_i, \mathbf{z}^{\star})$ ▷ $g(\cdot)$ is a convex scalarising function with respect to $\mathbf{w}$.
11: $Q \leftarrow Sort(\mathbf{V}^{(t)})$
12: $\mathcal{E} \leftarrow Q_{1,\ldots,\rho N}$
13: $\mathcal{M}_i^{(t)} \leftarrow \alpha\hat{\mu}_t + (1-\alpha)\hat{\mu}_{t-1}$
14: $\mathcal{S}_i^{(t)} \leftarrow \beta_t\hat{\sigma}_t + (1-\beta_t)\hat{\sigma}_{t-1}$
15: $\hat{\mathbf{x}}_i^{(t)} \leftarrow \mathcal{N}(\mathcal{M}_i^{(t)}, \mathcal{S}_i^{(t)})$
16: $\hat{\mathbf{V}}_i^{(t)} \leftarrow g(\hat{\mathbf{x}}_i^{(t)}, \mathbf{w}_i, \mathbf{z}^{\star})$
17: **if** $\hat{\mathbf{V}}_i^{(t)} \leq \mathbf{V}_i^{(t)}$ **then**
18: $\mathbf{V}_i^{(t+1)} \leftarrow \hat{\mathbf{V}}_i^{(t)}$
19: $\mathbf{x}_i^{(t+1)} \leftarrow \hat{\mathbf{x}}_i^{(t)}$
20: $\mathbf{z}^{\star} \leftarrow \min\left(\mathbf{z}^{\star}, \mathbf{F}\left(\mathbf{x}_i^{(t)}\right)\right)$
21: **end if**
22: **end for**
23: $t \leftarrow t + 1$
24: **until** $t \leq MaxGenerations$
25: $\mathbf{x} \leftarrow \mathcal{M}^{(t)}$

line *1*, the optimal weighting vectors are obtained according to prior information about the shape of the PF and the desired distribution of Pareto optimal solutions. This procedure is comprised of two steps, namely:

Step 1 Generate a set of solutions according to the PF shape of the given problem, for example for a concave PF, this reference front could be the one depicted in Fig. (4.1). The generation of this target front is mostly a matter of preference. To insulate the DM from different objective function scales, it is advisable that the objective functions are normalised in the range $[0, 1]$. This can be achieved if the ideal vector $\mathbf{z}^{\star}$ is known a priori or an adaptive method is used during the optimisation, such as in [2]. Note that this method can be

used only for bounded objective functions, since, generalised decomposition in its current formulation, only applies to such functions.

Step 2 Solve (4.11) for every point in the reference PF generated in **Step 1** to obtain the optimal weighting vectors $\mathbf{w}$.

The reference PF used in this work for the WFG4–9 test problems in Section 5.5.3 is a uniformly distributed set on a concave front using the method described in Appendix B.1. For the test problem WFG3, since the front is a line in any number of dimensions, an evenly spaced set of points were selected along this line and lastly for the WFG2 problem the optimal weighting vectors are evaluated using a random sample from the true PF. Next in lines *2–5* the starting population $\mathbf{x}^{(1)}$ is initialised by sampling the almost uniform distribution $\mathcal{N}(\mathcal{M}, \mathcal{S})$. In this work, for notational compactness, $\mathcal{N}(\mathcal{M}, \mathcal{S})$ has the following meaning:

$$\begin{pmatrix} \mathcal{N}(\mu_{1,1}, \sigma_{1,1}) & \cdots & \mathcal{N}(\mu_{1,n}, \sigma_{1,n}) \\ \vdots & \ddots & \vdots \\ \mathcal{N}(\mu_{N,1}, \sigma_{N,1}) & \cdots & \mathcal{N}(\mu_{N,n}, \sigma_{N,n}) \end{pmatrix} \tag{5.18}$$

where n is the number of decision variables and N the size of the population, which is the same as the number of subproblems and $\mathcal{N}$ is the truncated normal distribution in the domain of definition of the corresponding decision variables. The matrix, $\mathcal{M}^{(t)}$ contains the current estimate of the decision variables and in $\mathcal{S}^{(t)}$ are the standard deviation parameters. The $\mathcal{M}^{(t)}$ matrix is initialised at random within the decision variables' domain of definition or using some alternative method, for example, Latin hypercube sampling. The $\mathcal{S}^{(t)}$ matrix is initialised to some sufficiently large value so that the truncated normal distributions to be approximately equal to the uniform distribution at the first iteration, given no prior information is available. For this reason C is set to 10, see line *3*. Next the objective function, $\mathbf{F}(\cdot)$ is evaluated for the initial population $\mathbf{x}^{(1)}$ and the ideal vector $\mathbf{z}^{\star}$ is estimated using the minimum of the individual objectives in E. The main loop of the MACE-gD algorithm is in lines *8–24*. At each iteration and for every subproblem, $\mathbf{w}_i$, the entire population is evaluated using the Chebyshev decomposition. The population performance, $\mathbf{V}^{(t)}$ is sorted in ascending order[1] and the solutions in the ρ-percentile, $\mathcal{E}$ are used to update the instrumental density parameters of the i^{th} subproblem, $\mathcal{M}_i^{(t)}$ and $\mathcal{S}_i^{(t)}$. Following a new solution, $\hat{\mathbf{x}}_i^{(t)}$, is sampled from the parametric density using the updated parameters. This solution is evaluated and if its performance is superior to the

[1]For maximisation problems $\mathbf{V}^{(t)}$ is sorted in descending order.

previous one it is retained. The algorithm terminates once the maximum function evaluations are reached. Finally the PF approximation set is the matrix $\mathcal{M}^{(t)}$.

MACE and MACE-gD have similarities with MOEA/D [2] and derivatives [176, 177]. However there are fundamental differences which have been motivated by the results in Section 4.3.3. Namely, MACE and MACE-gD do not have a mating restriction, and there is no neighbourhood in weighting vector space. In fact only the top performing individuals for every subproblem are used, irrespective of their *origin* (see *Alg. 5.1*), namely the distribution that generated them. In contrast to that, MOEA/D derivatives insist on using a neighbourhood based on the distance of the weighting vectors. This choice seems reasonable when the relative location of the Pareto optimal solutions resulting from the set of subproblems is unknown. However, even if the Pareto front geometry is unknown *a priori*, this information can be extracted using generalized decomposition. For example, assuming an affine Pareto front geometry the neighbourhood can be calculated in objective space. The weighting vectors can be calculated using (4.11) and the neighbourhood structure can be as calculated for the above Pareto front. Here the assumption of an affine Pareto front is only limiting if the real Pareto front is discontinuous. However, this is also problematic for MOEA/D as defined in [2]. In any other case, the relative distance of the Pareto optimal solutions will be correct.

5.4 Algorithms Selected For Comparison

5.4.1 Multi-Objective Evolutionary Algorithm based on Decomposition

As already mentioned in Section 5.1 decomposition methods were usually applied in conjunction with gradient search methods, although there are examples of EAs based on this type of fitness assignment. One notable framework based on decomposition was introduced by Zhang et al. [2] the so called MOEA/D. The original version of MOEA/D was a decomposition-based algorithm with mating restriction and an archive preserving the best-so-far solution for every subproblem. The benefit of using scalarizing functions to extend an EA to MAPs is that considerations such as diversity preserving operators and *elite* preserving strategies become, to a great degree, redundant if the choice of weighting vectors and decomposition method is suitable for the problem in question. An additional benefit is that the computational cost tends to be lower compared to Pareto based algorithms [2]. MOEA/D depends on one of several available decomposition techniques, - weighted sum, Chebyshev [5] and normal boundary intersection [100] decompositions - with each having its own strengths and weaknesses. The minimisation problem

from Section 2.22, when using the Chebyshev decomposition is restated according to (4.9). In MOEA/D the vectors $\mathbf{w}^i$ are N evenly distributed weighting vectors. An MAP is decomposed to N number of sub-problems using $\mathbf{w}^i$. Each individual in the population is assigned to a single sub-problem, so N is also the size of the population. For example, for a 2 objective problem, the weighting vectors are defined as:

$$w_1^i = \frac{i}{H},\ w_2^i = 1 - w_1^i,\ i \in \{0, \ldots, H\}, \tag{5.19}$$

where the H parameter controls the number of subdivisions per dimension and $\mathbf{w}^i = \{w_1^i, w_2^i\}$. The argument is that since g^{tce} is a continuous function of $\mathbf{w}$, N evenly distributed weighting vectors should result in N evenly distributed Pareto optimal solutions, assuming that the objectives are normalised [2]. However this argument is only valid in the case that a boundary intersection (BI) approach is used, for instance the normal boundary intersection method (NBI) [100]. In NBI the following program is to be solved:

$$\begin{aligned} &\min_{\mathbf{x}} g_{nbi}(\mathbf{x}; \mathbf{w}^i, \mathbf{z}^\star) = d \\ &\quad \text{s.t. } \mathbf{z}^\star - \mathbf{F}(\mathbf{x}) = d \cdot \mathbf{w}^i, \end{aligned} \tag{5.20}$$

where Zhang et al. [2] suggest a penalty approach to handle the equality constraint. Thus (5.20) is transformed to:

$$\begin{aligned} \min_{\mathbf{x}} g_{nbi}(\mathbf{x}; \mathbf{w}^i, \mathbf{z}^\star) &= d_1 + p d_2 \\ d_1 &= \frac{\|(\mathbf{z}^\star - \mathbf{F}(\mathbf{x}))^T \mathbf{w}^i\|_2}{\|\mathbf{w}^i\|_2} \\ d_2 &= \|\mathbf{F}(\mathbf{x}) - (\mathbf{z}^\star - d_1 \mathbf{w}^i)\|_2, \end{aligned} \tag{5.21}$$

where p is a penalty parameter. It was shown that MOEA/D using (5.21) has the potential to produce truly evenly distributed Pareto optimal solutions [2]. Unfortunately (5.21) has three significant drawbacks. First, the normal-boundary intersection method does not guarantee that the solutions to the sub-problems will be Pareto optimal [100]. Second, NBI has to be solved using a penalty method which introduces one more parameter that has to be tuned for every test problem separately, and lastly it is unclear how this decomposition method can be scaled for MAPs. A description of the MOEA/D algorithm is now given:

Step 1 Generate N equally spaced $\mathbf{w}^i$ vectors according to (5.19). Create a matrix B containing the nearest neighbours of each $\mathbf{w}^i$ and initialise the ideal weighting vector $\mathbf{z}^\star$ to a large value.

Step 2 Evaluate the decision variable vectors $\mathbf{x}$ using the objective function $\mathbf{F}(\mathbf{x})$.

Step 3 Update the ideal vector $\mathbf{z}^\star = \min(\mathbf{z}^\star, \mathbf{F}(\mathbf{x}))$.

Step 4 For each individual $i \in \{1, \ldots, N\}$ execute the following procedure:

Step 4.1 Apply genetic operators, crossover and mutation, using individuals in the neighbourhood of each solution. The choice of individuals is random among neighbouring solutions.

Step 4.2 Evaluate the newly generated solution using (4.9).

Step 4.3 Update the ideal vector $\mathbf{z}^\star$.

Step 4.4 If the new solution is superior to the previous in the archive then swap the old solution to the i^{th} sub-problem with the new solution. Otherwise, retain the old solution.

Step 4.5 Check if the new solution is better for any of the neighbouring subproblems and substitute if that is the case.

Step 5 If the termination criteria are met output the non-dominated solutions, otherwise proceed to **Step 4**.

In this work the MATLAB code provided by the authors of MOEA/D is used [2].

5.4.2 Regularity model-based EDA

The second algorithm that is employed in this comparison study, see Section 5.5, is the regularity model-based multi-objective estimation of distribution algorithm (RM-MEDA) proposed by Zhang et al. [125]. The main idea in RM-MEDA is that, for continuous MAPs, the Pareto set can be viewed as a $(k-1)$-dimensional piecewise continuous manifold. So for two dimensions, the PF can be described with line segments, for three dimensions with planes etc.

Zhang et al. [125] used inductively the Karush-Kuhn-Tucker condition [5] for continuous multi-objective problems, asserting that the PF of a problem with k objectives is a $(k-1)$-dimensional manifold in the decision variable space. This assertion allowed Zhang et al. [125] to approximate this $(k-1)$-dimensional manifold with K piecewise continuous manifolds. To accomplish this task a $(k-1)$-dimensional local principal component analysis algorithm was used to partition the population in K disjoint clusters and then the centroid and its variance were estimated. The problem with this approach is that there is no objective measure to support

the choice of the number of clusters K for an unknown problem. Hence the practitioner must heavily depend on the *smoothness* of the objective function in the decision space. Alternatively, if it is known a priori that the MAP fulfills the smoothness criteria then RM-MEDA will be able to exploit that structure thus converge much faster.

In [125] RM-MEDA was evaluated against PCX-NSGA-II [128], GDE3 [99] and MIDEA [129], and on average outperformed the aforementioned algorithms on variants of the ZDT[1] test problems [127]. However the performance of RM-MEDA comes at the expense of increased computational cost due to the necessity of computing a local principal component analysis on each iteration. The implementation of RM-MEDA that is employed in this work is the publicly available version in MATLAB code provided by the authors [125].

5.4.3 Random Search

Random search is regarded as the absolute baseline algorithm in MOEAs. In random search absolutely no prior assumptions are made about the problem and during the optimisation the search is not affected by the *fitness* of the previous samples. Random search with memory, that is an algorithm that samples uniformly the decision variable space but does not revisit solutions previously sampled, enjoys asymptotical convergence [178]. However, since there is no mechanism to *steer* the search; the time to convergence is analogous to the problem complexity. Conversely, due to its simplicity and inability to *learn* it cannot be misled by the problem. The random search algorithm employed in the current work is in its most basic form. The objective function is evaluated for 25 000 uniformly sampled decision variable combinations, then the non-dominated solutions are extracted and a randomly selected subset is chosen for evaluation using the methodology described in Section 5.5.

5.5 Comparative Studies

5.5.1 Performance Indicator

The main performance metric for the comparative studies in this work is the generational distance (GD) indicator, see (5.22). This metric has been chosen since the main interest is in the convergence properties of the studied algorithms. Generational distance (GD), introduced in

[1] Zitzler, Deb, Thiele (ZDT)

Table 5.1: Value of the H parameter in MOEA/D and MACE and the corresponding population size N. The population size is the same for all algorithms. $|\mathcal{P}^\star|$ is the size of the Pareto front reference set, solutions in this set are uniformly distributed along the PF.

Obj. #	2	3	4	5	6	7	8	9	10	11
H	101	20	10	7	6	5	5	5	5	5
N	101	210	220	210	252	210	330	495	715	1001
$\lvert\mathcal{P}^\star\rvert$	500	1000	1500	2000	2500	3000	3500	4000	4500	5000

[179] is given by,

$$D(A, \mathcal{P}^\star) = \frac{\sum_{s \in A} \min\{\|\mathcal{P}_1^\star - s\|_2, \ldots, \|\mathcal{P}_N^\star - s\|_2\}}{|A|} \tag{5.22}$$

where $|A|$ is the cardinality of the set A. The GD metric measures the distance of the elements in the set A from the nearest of the reference PF. A is an approximation of the true Pareto front and $\mathcal{P}^\star$ is the reference Pareto optimal set.

5.5.2 Experiment Description

The goal of the comparative studies in this work is not to proclaim a *best* algorithm among variants of MACE and the aforementioned frameworks. The main aim is to explore the potential of generalised decomposition versus what is considered to be standard practice in many present day decomposition-based MOEAs. The additional benefit is that the generalised decomposition framework seems very suitable for the extension of EDAs to MAPs, something that enables us to evaluate whether the performance of the CE-method is comparable with established MOEAs. Therefore the selection of MOEA/D is only natural since this algorithm framework has become a baseline for comparison of decomposition-based MOEAs. Also, the good performance of RM-MEDA against other EDAs makes it a suitable candidate to evaluate the main EDA in MACE and MACE-gD algorithms.

In Section 4.3, it was illustrated that the three objectives that MOEAs have to achieve - namely convergence, diversity and PF coverage - can be reduced to only one, convergence, in the generalised decomposition framework. Therefore a quantity measuring convergence to the PF becomes very important, which is the reason that the GD metric is used, see (5.22).

The selected problem set for the experiments is the WFG toolkit [167], specifically problems WFG2–WFG9, since it contains several challenging problems, it is scalable and the PF is known a priori. For all test instances 32 decision variables were used and the k parameter is calculated as: $k = 4 + 2 \cdot (M - 1)$, the only exception being for the 2 objective instances of the test problems where it is set to 4; M is the number of objectives. Also the neighbourhood size T in

Table 5.2: Settings for MACE and MACE-gD.

ρ	α	β	q
0.1	0.9	0.9	7

MOEA/D was selected to be 10% of the population size N, since, according to [4], this appears to be a setting that produces good results for MAPs. The population size was the same for all the algorithms, see Table 5.1. The parameters of the CE-method are the same in MACE and MACE-gD and have been selected according to the suggestions in [172], see Table 5.2. Lastly, the reference Pareto fronts used in MACE-gD to produce the *optimal* weighting vectors for the test instances WFG2 and WFG3 were generated by a random sample of the true Pareto set and for the problems WFG4–WFG9, the method described in Appendix B.1 was employed for generating a concave Pareto optimal set. In practice, such information is usually not available before the application of the optimisation algorithm. This problem can be averted using an identification method to determine the PF shape during the optimisation; this methodology is left for future research. Lastly, as is probably evident from the selection of the reference PS for the generation of the weighting vectors in MACE-gD, it is assumed that the DM is interested in a PF that is uniformly distributed on that front. This is due to several considerations: first, if the method that is usually applied in MOEA benchmarking for generating the reference PF of concave geometry is followed, say for 3 dimensions, i.e. generate a set of evenly distributed weighting vectors and then project onto the first octant of the unit sphere, then for higher dimensions, due to the curvature of the hypersphere this will induce a large bias in the reference set. Namely, the density of Pareto optimal solutions will be higher near the edges of the PF compared to the density near the *centre*. Conversely, to produce a truly even distribution of Pareto optimal solutions in high dimensions is still an unresolved issue for an arbitrary number of points, even for PFs that have simple geometry, see [180, 181].

5.5.3 Experiment Results

A summary of the GD-metric performance of the algorithms is presented in Tables 5.3–5.10. The values in bold indicate the best performing algorithm for the particular instance of a test problem. The Kruskal-Wallis test is used at the 95% confidence level to verify if the mean performance of the studied algorithms is different. Furthermore, for each algorithm and for each problem instance the Wilcoxon two-sided rank sum test was employed with $\alpha = 0.05$ (95% confidence level). Every time an algorithm outperforms another in the test group, for a test

Table 5.3: GD-metric performance of the studied algorithms on the WFG2 problem for 2–11 objectives.

	WFG2				
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	0.0816 (3)	0.1027 (4)	0.0656 (2)	**0.0279** (1)	0.1687 (5)
3	**0.0353** (1)	0.0386 (2)	0.0444 (3)	0.0794 (4)	0.1929 (5)
4	0.0712 (2)	**0.0485** (1)	0.1283 (4)	0.1274 (3)	0.1998 (5)
5	0.0718 (2)	**0.0471** (1)	0.1717 (4)	0.1674 (3)	0.2125 (5)
6	0.0573 (2)	**0.0423** (1)	0.1489 (3)	0.1979 (4)	0.2228 (5)
7	0.0650 (2)	**0.0487** (1)	0.1081 (3)	0.2152 (4)	0.2335 (5)
8	0.0525 (2)	**0.0379** (1)	0.0806 (3)	0.2434 (4)	0.2649 (5)
9	0.0471 (2)	**0.0286** (1)	0.0791 (3)	0.2563 (4)	0.2638 (5)
10	0.0495 (2)	**0.0168** (1)	0.0658 (3)	0.2694 (4)	0.2785 (5)
11	0.0453 (2)	**0.0108** (1)	0.0814 (3)	0.2793 (4)	0.2867 (5)

instance, a 1 was added to its rank. Therefore since 5 algorithms are considered, the maximum rank for an algorithm is 4. A rank of 4 means that the algorithm in question performs better than all other algorithms for that particular test instance. In the case that no algorithm is clearly better, this is marked as a tie thus both algorithms are displayed in bold in Tables 5.3–5.10. An algorithm with a rank of 4 is denoted with a (1), one with a rank of 3 with a (2) and so forth, with (1) denoting the best performing algorithm and (5) the worst performer. These values are recorded to the right of the GD-metric performance in Tables 5.3–5.10.

Table 5.3 presents the results of the algorithms for 2–11 objective instances of the WFG2 test problem. WFG2 has the following features – it is non-separable, unimodal with respect to all objectives except the last which is multi-modal; there is no bias in the parameters and the PF geometry is piecewise convex[1]. In this problem MACE-gD performance is far better than all algorithms for more than 4 objectives. This performance is attributed to the fact that for PFs that have a convex geometry the optimal weighting vector set, see Fig. (4.1), is clustered near the centre region. So, using an even distribution of weighting vectors, the effective number of Pareto optimal solutions for which these vectors are optimal is reduced. This is especially true in higher dimensions, since the features seen in Fig. (4.1) are only accentuated. However, the MACE algorithm that utilised the same weighting vector selection as MOEA/D, outperforms the latter algorithm for all the instances except the 2-objective case. This in combination to the fact that MOEA/D consistently outperforms RM-MEDA, except for the 2-objective instance, leads to the hypothesis that Pareto-based algorithms potentially are not very well suited for problems with convex PF geometries in high dimensions. This hypothesis is further supported by the fact that RM-MEDA uses a variant of non-dominated sorting [125]. So in high dimensions

[1]See Section 2.5.1 for a definition of *piecewise convex* when referring to a Pareto front.

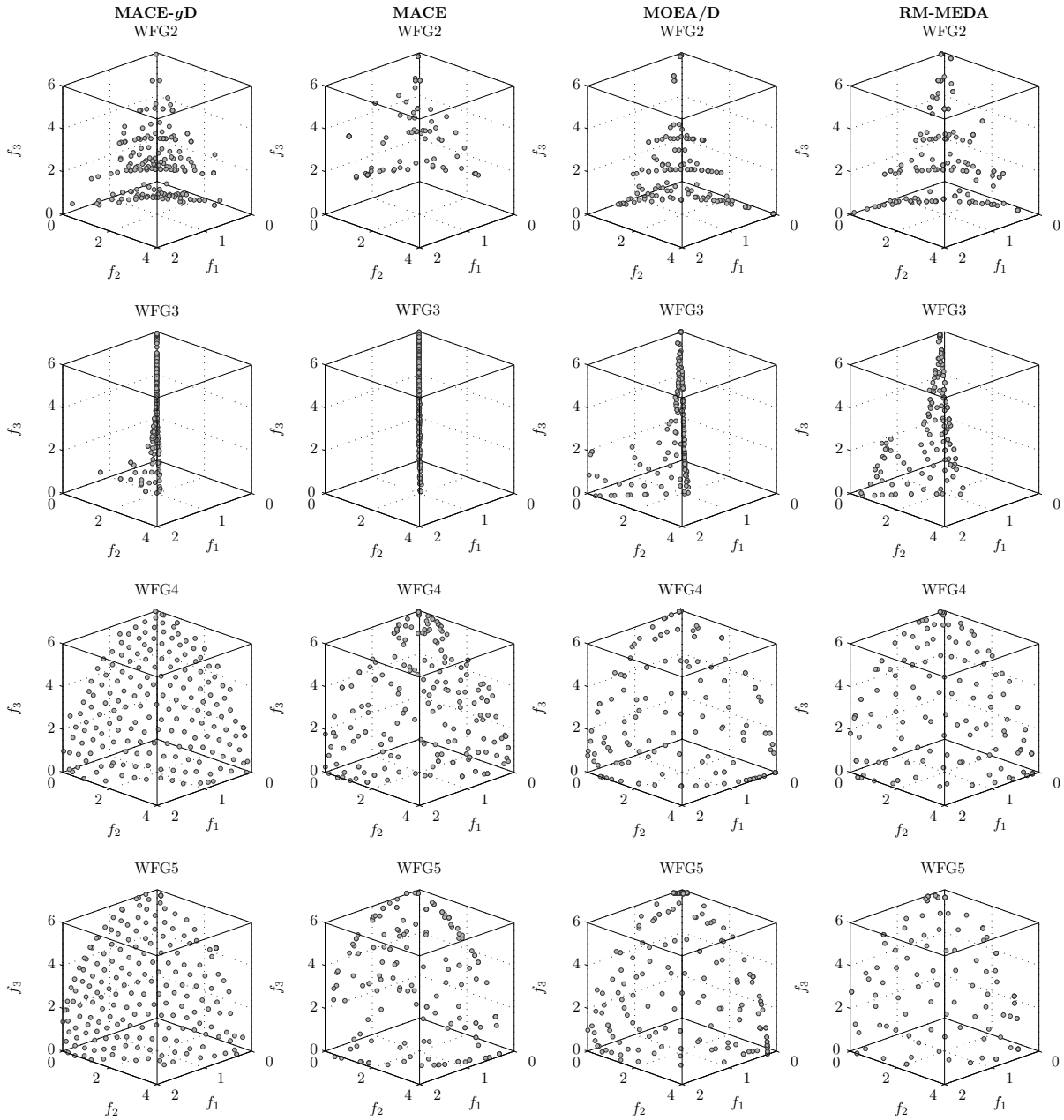

Figure 5.1: MACE-**g**D, MOEA/D and RM-MEDA Pareto front for 3 objective instances of the WFG2–WFG5 test problems.

Table 5.4: GD-metric performance of the studied algorithms on the WFG3 problem for 2–11 objectives.

	WFG3				
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	**0.0133** (1)	0.0194 (3)	0.0190 (3)	0.0215 (4)	0.2108 (5)
3	0.0699 (2)	**0.0231** (1)	0.1553 (3)	0.2419 (4)	0.2899 (5)
4	0.0841 (2)	**0.0338** (1)	0.2422 (3)	0.3474 (5)	0.3204 (4)
5	0.1023 (2)	**0.0230** (1)	0.3137 (3)	0.3885 (5)	0.3311 (4)
6	0.1146 (2)	**0.0209** (1)	0.2701 (3)	0.4091 (5)	0.3312 (4)
7	0.1033 (2)	**0.0340** (1)	0.2122 (3)	0.4346 (5)	0.3321 (4)
8	0.0921 (2)	**0.0290** (1)	0.1912 (3)	0.4356 (5)	0.3350 (4)
9	0.0848 (2)	**0.0237** (1)	0.1728 (3)	0.4342 (5)	0.3364 (4)
10	0.0760 (2)	**0.0135** (1)	0.1512 (3)	0.4314 (5)	0.3371 (4)
11	0.0702 (2)	**0.0117** (1)	0.1317 (3)	0.4283 (5)	0.3379 (4)

the closer the estimated PF is to the true PF, the fewer the solutions that are part of the first and second non-dominated fronts, which means that the availability of *good* solutions to the model creation process is reduced in RM-MEDA. Therefore, the closer the algorithm is to the actual PF, the more difficult it becomes for further progress to be achieved.

The results for the WFG3 instances are seen in Table 5.4. The WFG3 problem is non-separable, unimodal with no bias in the parameters and its PF geometry is affine degenerate, i.e. the front is always a line for any number of dimensions. In this problem as well, the MACE-gD algorithm has far superior performance, except for the 2-objective instance, where the performance of all algorithms is comparable. However MACE has statistically better performance for 2 objectives. MACE-gD outperforms other approaches on the WFG3 problems mainly due to its geometry. Since its geometry is affine, the optimal weighting vectors direct the algorithm towards the correct location on the PF, while other algorithms are exploring the entire objective space assuming that the front is some hypersurface which is to be populated with solutions. This *focus* illustrates what can be the advantages of generalised decomposition. It is also encouraging that MACE performs very well, which means that if the information about the geometry of the PF is not very accurate the algorithm can still achieve acceptable results. Additionally, the results of RM-MEDA on WFG3 further support the hypothesis about its selection scheme; notably its performance is much degraded compared to WFG2. Lastly a curiosity is that for increasing number of dimensions, MACE-gD is not only better compared with other algorithms but the GD metric becomes smaller, something that is counter-intuitive. However the explanation for that is rather simple, namely since the Pareto front in WFG3 is a line in any number of dimensions, the necessity of employing a larger population is diminished. Since the population size is increased, and the optimal weighting vectors are known, the density

Table 5.5: GD-metric performance of the studied algorithms on the WFG4 problem for 2–11 objectives.

WFG4					
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	0.0345 (3)	0.0344 (3)	**0.0211** (1)	0.0392 (4)	0.1161 (5)
3	0.0617 (3)	0.0522 (2)	**0.0316** (1)	0.0939 (4)	0.1302 (5)
4	0.0749 (3)	0.0740 (2)	**0.0655** (1)	0.1336 (4)	0.1358 (5)
5	0.1438 (3)	**0.1048** (1)	0.1653 (5)	0.1464 (4)	0.1407 (2)
6	**0.1358** (1)	0.1414 (2)	0.1959 (5)	0.1668 (4)	0.1549 (3)
7	0.2349 (4)	0.1997 (3)	0.2739 (5)	0.1898 (2)	**0.1770** (1)
8	0.3176 (4)	0.2351 (3)	0.3371 (5)	0.2172 (2)	**0.2025** (1)
9	0.3995 (5)	0.3028 (3)	0.3958 (4)	**0.2495** (1)	0.2568 (2)
10	0.3791 (4)	0.3265 (3)	0.4001 (5)	0.2718 (2)	**0.2577** (1)
11	0.4839 (5)	0.3875 (3)	0.4644 (4)	**0.3162** (1)	0.3540 (2)

Table 5.6: GD-metric performance of the studied algorithms on the WFG5 problem for 2–11 objectives.

WFG5					
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	0.0393 (2)	0.0523 (4)	**0.0276** (1)	0.0433 (3)	0.1947 (5)
3	0.1052 (3)	0.0962 (2)	**0.0321** (1)	0.2168 (5)	0.2114 (4)
4	0.1533 (2)	0.1845 (3)	**0.0655** (1)	0.2652 (5)	0.2268 (4)
5	**0.1537** (2)	0.2221 (3)	**0.1540** (2)	0.2604 (5)	0.2307 (4)
6	0.1579 (2)	0.2313 (3)	**0.1558** (1)	0.2556 (5)	0.2346 (4)
7	**0.1872** (1)	0.2286 (2)	0.2455 (4)	0.2588 (5)	0.2372 (3)
8	0.2620 (3)	**0.2340** (1)	0.3262 (5)	0.2646 (4)	0.2441 (2)
9	0.3357 (4)	0.2685 (2)	0.4007 (5)	0.2748 (3)	**0.2598** (1)
10	0.3497 (4)	0.2789 (2)	0.3813 (5)	0.2911 (3)	**0.2706** (1)
11	0.4479 (4)	0.3203 (3)	0.4792 (5)	0.3096 (2)	**0.3036** (1)

of solutions along the WFG3 PF is effectively increased, hence the GD metric value decreases.

In Table 5.5 the results for the WFG4 problem are presented. WFG4 is a separable problem, multi-modal with no bias and its PF geometry is concave. In this problem the major influence in algorithm performance seems to be the fact that this problem is multimodal. From the MACE and MACE-gD perspective, the fact that the instrumental densities used are Gaussian appears to have a significant effect. Namely, the multi-modal nature of the problem is misleading to all the algorithms. However, the more elaborate model employed in RM-MEDA helps the algorithm scale much better compared to the rest. This conclusion is based on the performance of random search on this problem and the fact that RM-MEDA follows this much more *smoothly* relative to all other algorithms. For example, for the 11 objective instance, while random search achieves a mean value for the GD-metric of 0.3540, MACE-gD, MOEA/D and MACE have much worse performance, although the positive effect of generalised decomposition is clearly visible when comparing MACE-gD to MACE. For instances with 2–4 objectives MOEA/D exhibits the best performance, however it is closely followed by MACE-gD and MACE. This leads to the hypothesis that a more elaborate EDA coupled with generalised decomposition could potentially

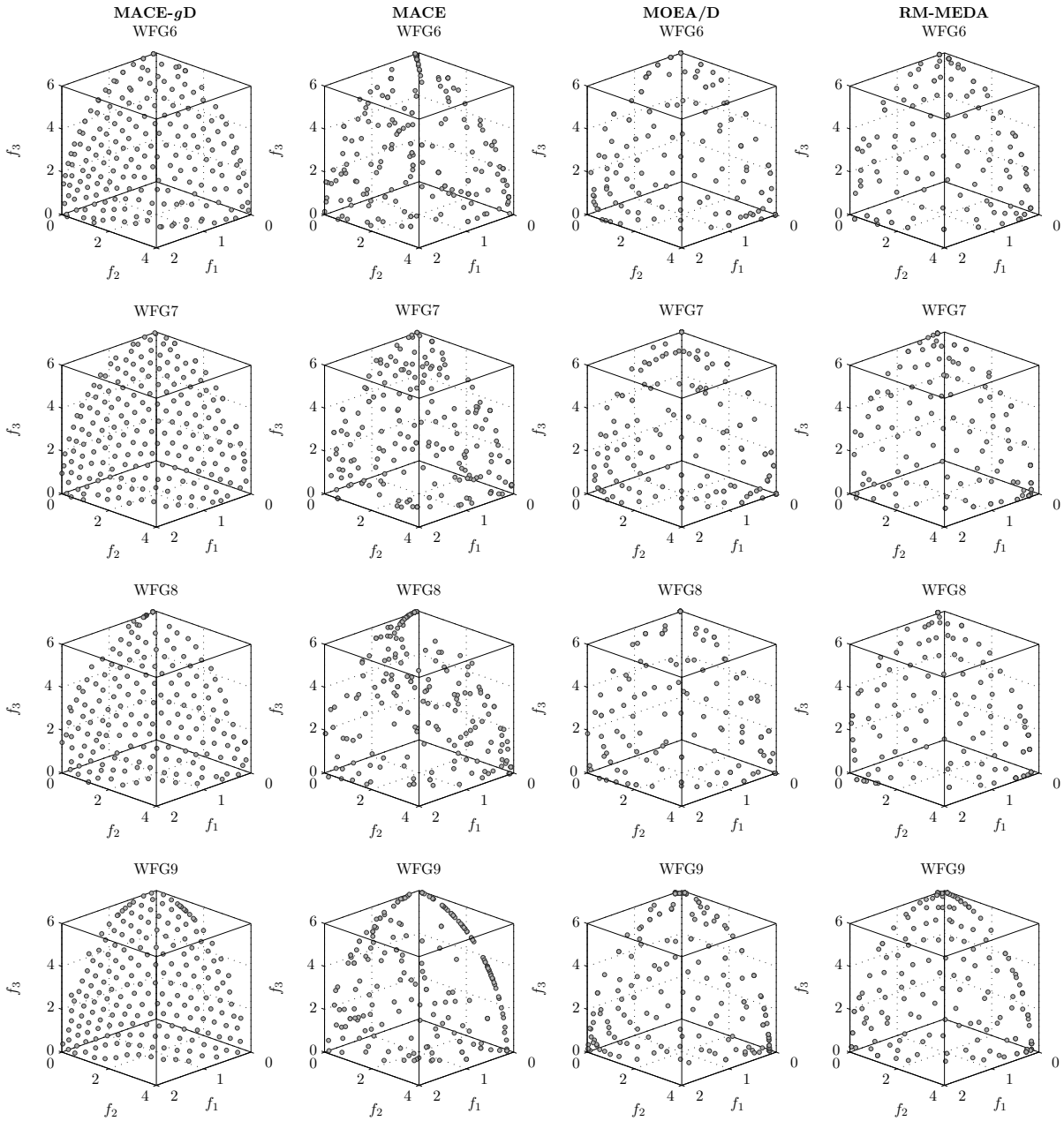

Figure 5.2: MACE-**g**D, MOEA/D and RM-MEDA Pareto front for 3 objective instances of the WFG6–WFG9 test problems.

Table 5.7: GD-metric performance of the studied algorithms on the WFG6 problem for 2–11 objectives.

WFG6					
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	**0.0162** (2)	0.0226 (3)	0.0293 (4)	**0.0164** (2)	0.2465 (5)
3	0.0489 (2)	0.0499 (3)	**0.0318** (1)	0.1417 (4)	0.2666 (5)
4	0.0782 (2)	0.0836 (3)	**0.0624** (1)	0.2441 (4)	0.2865 (5)
5	0.1459 (2)	**0.1182** (1)	0.1644 (3)	0.2532 (4)	0.2940 (5)
6	0.1960 (3)	**0.1491** (1)	0.1962 (3)	0.2574 (4)	0.2936 (5)
7	0.2531 (3)	**0.1897** (1)	0.2506 (2)	0.2608 (4)	0.2881 (5)
8	0.3094 (4)	**0.2215** (1)	0.3234 (5)	0.2759 (2)	0.2885 (3)
9	0.3890 (5)	**0.2716** (1)	0.3520 (4)	0.2888 (2)	0.2951 (3)
10	0.3762 (5)	**0.3004** (1)	0.3758 (5)	0.3078 (3)	0.3032 (2)
11	0.4632 (5)	0.3577 (3)	0.4233 (4)	0.3257 (2)	**0.3201** (1)

Table 5.8: GD-metric performance of the studied algorithms on the WFG7 problem for 2–11 objectives.

WFG7					
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	0.0075 (2)	0.0144 (3)	**0.0040** (1)	0.0158 (4)	0.1707 (5)
3	0.0363 (3)	0.0309 (2)	**0.0261** (1)	0.1159 (4)	0.1889 (5)
4	0.0819 (3)	0.0740 (2)	**0.0732** (1)	0.1742 (4)	0.1998 (5)
5	0.1374 (2)	**0.1086** (1)	0.1760 (3)	0.1915 (4)	0.2013 (5)
6	0.1541 (2)	**0.1434** (1)	0.2150 (5)	0.2050 (4)	0.2046 (4)
7	0.2587 (4)	**0.1889** (1)	0.2839 (5)	0.2191 (3)	0.2142 (2)
8	0.3269 (4)	0.2282 (2)	0.3704 (5)	0.2432 (3)	**0.2270** (1)
9	0.3954 (4)	0.2838 (3)	0.4359 (5)	0.2632 (2)	**0.2508** (1)
10	0.3803 (4)	0.3092 (3)	0.4052 (5)	0.2844 (2)	**0.2633** (1)
11	0.4812 (4)	0.3704 (3)	0.4875 (5)	**0.3115** (1)	0.3153 (2)

overcome the difficulties present in problems similar to WFG4.

Table 5.6 presents the results for the WFG5 problem. WFG5 is a unimodal, separable and deceptive problem with no bias and concave PF. It is most interesting that for this test problem, contrary to what was anticipated, RM-MEDA performs consistently worse compared with random search, the only exception being the 2-objective test instance. However for more than 9 objectives random search dominates in all other algorithms in performance. Also when compared to RM-MEDA, both MACE and MACE-gD perform significantly better for all instances with 2–10 objectives, a fact that supports the theory presented in [126] that EDAs using low-order statistics with some form of clustering have potential. Of course clustering is not used in the presented versions of the MACE algorithm which is left for future research. Another important feature is that MOEA/D strongly outperforms all algorithms on this test problem for 2–6 objectives although its performance is heavily degraded for higher numbers of objectives performing much worse than random search. However this rapid relative degradation in performance is not seen in MACE. This phenomenon is very likely to be related in some way with the control

Table 5.9: GD-metric performance of the studied algorithms on the WFG8 problem for 2–11 objectives.

	WFG8				
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	0.0598 (2)	0.0697 (3)	**0.0582** (1)	0.0875 (4)	0.2043 (5)
3	0.0857 (3)	0.0797 (2)	**0.0562** (1)	0.1671 (4)	0.2147 (5)
4	0.1201 (3)	0.1165 (2)	**0.0790** (1)	0.2596 (5)	0.2436 (4)
5	0.1453 (2)	**0.1349** (1)	0.1966 (3)	0.2982 (5)	0.2635 (4)
6	0.1835 (2)	**0.1528** (1)	0.1961 (3)	0.3005 (5)	0.2657 (4)
7	0.2524 (2)	**0.1888** (1)	0.2804 (4)	0.3002 (5)	0.2652 (3)
8	0.3214 (4)	**0.2237** (1)	0.3594 (5)	0.3134 (3)	0.2703 (2)
9	0.3762 (4)	**0.2706** (1)	0.3929 (5)	0.3246 (3)	0.2852 (2)
10	0.3698 (4)	0.2995 (2)	0.4050 (5)	0.3401 (3)	**0.2912** (1)
11	0.4669 (5)	0.3601 (3)	0.4658 (4)	0.3560 (2)	**0.3254** (1)

parameters in MOEA/D, leading to the conclusion that MACE, MACE-gD and RM-MEDA are more robust with respect to their controlling parameters. This is in accord with recent studies that show that the sweet spot of configuration parameters *shrinks* with the dimensional increase of the problem [182].

Table 5.7 presents the results of the GD-metric performance for the WFG6 test problem. WFG6 is a non-separable, unimodal problem with no bias and concave PF geometry. These results further strengthen the hypothesis that the CE method performs very well on unimodal problems. Generally, the performance over all test problems that are unimodal, for MACE and MACE-gD is similar, see Tables 5.6-5.9. The exception to this are the results for the WFG3 problem (see Table 5.4), however the geometry of WFG3 is influencing the performance of the algorithms greatly, so that a MACE-gD that has prior information of the *correct* direction of search can massively exploit this feature. In WFG6, RM-MEDA performs worse than random search for all instances except the 2-objective. This must be due to the fact that this problem is non-separable, as is the case for WFG2–3 and WFG8–9, see Tables 5.3–5.4 and Tables 5.9–5.10. For 2–3 objectives MOEA/D has superior performance to all algorithms and for 4–10 objectives MACE-gD is the top performer. It is interesting to note that in that range of objectives MACE and MOEA/D have similar performance, which further suggests that the decomposition method has a strong influence on algorithm performance.

Table 5.8 and Table 5.9 correspond to the mean GD-metric value of the compared algorithms for the problems WFG7 and WFG8. The demonstrated performance is similar to the results reported in Tables 5.3–5.7.

Lastly Table 5.10 presents the results for the WFG9 test problem which is non-separable, multi-modal and deceptive. WFG9 has also parameter dependent bias and its PF geometry is

Table 5.10: GD-metric performance of the studied algorithms on the WFG9 problem for 2–11 objectives.

	WFG9				
Obj. #	MACE	MACE-gD	MOEA/D	RM-MEDA	RAND
2	0.0223 (2)	0.0259 (3)	0.0286 (4)	**0.0179** (1)	0.1925 (5)
3	0.0390 (3)	**0.0366** (2)	**0.0365** (2)	0.0657 (4)	0.2410 (5)
4	0.0653 (3)	**0.0592** (1)	0.0607 (2)	0.1636 (4)	0.2764 (5)
5	0.1494 (3)	**0.0987** (1)	0.1468 (2)	0.2442 (4)	0.2982 (5)
6	0.1441 (3)	**0.1349** (1)	0.1369 (2)	0.2655 (4)	0.3073 (5)
7	0.2193 (2)	**0.1843** (1)	0.2270 (3)	0.2769 (4)	0.3070 (5)
8	0.3055 (4)	**0.2223** (1)	0.3122 (5)	0.2889 (2)	0.3058 (4)
9	0.3657 (4)	**0.2742** (1)	0.3685 (5)	0.3039 (2)	0.3110 (3)
10	0.3514 (4)	**0.2999** (1)	0.3547 (5)	0.3214 (3)	0.3199 (2)
11	0.4473 (4)	0.3488 (3)	0.4506 (5)	0.3416 (2)	**0.3346** (1)

concave. The results in Table 5.10 seem contradictory with the results for the WFG4 problem (Table 5.5), which is also multimodal. The expectation was that, since WFG9 is multimodal and deceptive, the more elaborate algorithm (RM-MEDA) to be the superior performer [125]. Instead, for more than ~ 6 objectives the performance of RM-MEDA is very close to that of random search and worse in the last two instances, i.e. for 10 and 11 objectives. In contrast, for 3-7 objectives MACE, MACE-gD and MOEA/D have relatively similar performance - with MACE-gD in the lead. For 8–10 objectives this lead is significantly increased and this is attributed to generalised decomposition, since the performance of the CE method for multimodal problems is moderate, or so it would seem.

5.5.4 Sensitivity of MACE and MACE-gD to the ρ Parameter

Although a complete sensitivity analysis of algorithm performance with respect to all control parameters in the MACE and MACE-gD algorithms is beyond the scope of this work, it is important to investigate how convergence is affected by the ρ parameter. This parameter controls the percentage of the individuals in the previous generation that are used in the updating process of the μ and σ parameters of the instrumental densities in the CE method. Intuitively, since every instrumental density is sampled only once for every subproblem, this parameter controls the amount of information sharing between different subproblems. In that context it is similar to the T parameter in MOEA/D. However the *neighbourhood* for the MACE algorithms does not depend on the closeness of weighting vectors but depends only on the similarity of performance of different subproblems. Hence, it is not fixed as it is in MOEA/D.

To test how the GD metric performance of MACE and MACE-gD is affected for various values of ρ, 50 independent trials were performed for $\rho = \{0.1, 0.2, \ldots, 0.9\}$ on the WFG9

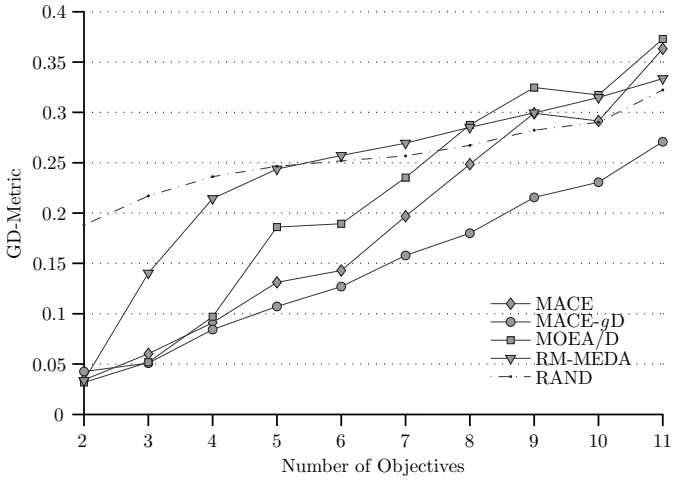

Figure 5.3: Mean GD-metric performance of studied algorithms over WFG2–9 for 2–11 objectives.

problem. All other parameters are identical to those employed in Section 5.5.3. The results can be seen in Fig. (5.4) – Fig. (5.6). In Fig. (5.4) and Fig. (5.5) the mean performance of the two algorithms over 2–11 objectives for different values of the ρ parameter is illustrated. The fact that the mean performance of MACE-gD, see Fig. (5.5), is better compared to MACE, see Fig. (5.4), is expected given the results in Table 5.10. MACE and MACE-gD exhibit similar variation in terms of their GD metric performance for the selected range of ρ. Namely the absolute value of the difference of the best performance less the worse one as seen in Fig. (5.4) and Fig. (5.5) is 2.79×10^{-3} and 2.96×10^{-3} for MACE and MACE-gD respectively. A comparison of these values with the absolute performance of the above algorithms shown in Fig. (5.6), suggests that MACE and MACE-gD are relatively robust to variations in the ρ parameter. Specifically, the mean performance over all objectives of MACE and MACE-gD for the WFG9 problem is 0.2109 and 0.1685 respectively which means that for $\rho \in \{0.1, \ldots, 0.9\}$ the variation in performance with respect to the GD metric of MACE and MACE-gD is 1.32% and 1.75% respectively. However their behaviour is qualitatively different.

MACE performs relatively better for all values of $\rho > 0.2$ with no consistent degradation or improvement past this threshold. Therefore any value for ρ that is greater than 0.2 should produce acceptable results. In contrast to MACE, the performance of MACE-gD varies in a much more coherent manner for different values of ρ, and, in general for $\rho < 0.5$ it performs consistently better than for $\rho > 0.5$. The lack of *coherency* in the improvement (or degradation) in GD performance for MACE could suggest that the algorithm is not affected as much as

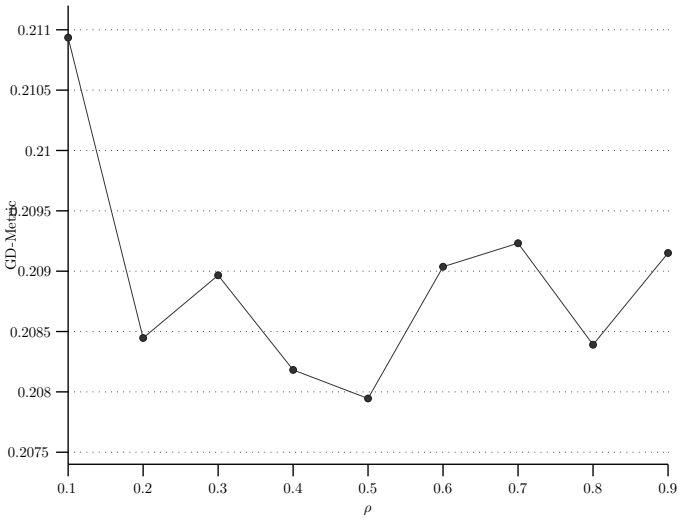

Figure 5.4: Mean GD-metric performance of MACE, over all objectives for the WFG9 test problem.

MACE-gD, by the ρ parameter. The question is: why is MACE less susceptible to variations in ρ? The hypothesis is that, since the weighting vectors in MACE are selected in the same fashion as in MOEA/D, subproblems are aggregated in a very small region of the PF, therefore sharing information with neighbouring solutions is less disruptive, for instance, see Fig. (4.1). Conversely, the weighting vectors in MACE-gD are distributed according to a uniformly distributed Pareto front, so that, as we increase ρ, the less likely it is to obtain *local* information from faraway solutions. Hence the convergence rate of the algorithm is somewhat inhibited for large ρ.

Additionally the GD-performance of MACE-gD appears to be a quasi-convex function of ρ, see Fig. (5.5). Namely there are two competing trends in MACE-gD, first, the larger ρ is, the more samples are used in the updating rules in (5.14) and (5.15), hence better estimates are obtained. However, exceeding a certain value for ρ which in this instance appears to be somewhere between $(0.5, 0.6)$, the GD-metric performance starts to degrade and this is due to the second trend, i.e. for large ρ samples obtained by disparate subproblems are used in the updating process, hence convergence to the PF becomes slower. This is consistent with the hypothesis that generalised decomposition successfully captures the density of the PF reference set used to generate the optimal weighting vectors.

In Fig. (5.6) the mean GD performance is illustrated over all ρ values for increasing numbers of objectives. Again this result is consistent with the experiments in Section 5.5.3. Additionally it seems that the linear scaling of performance of the MACE-gD algorithm as seen in Fig. (5.3), is

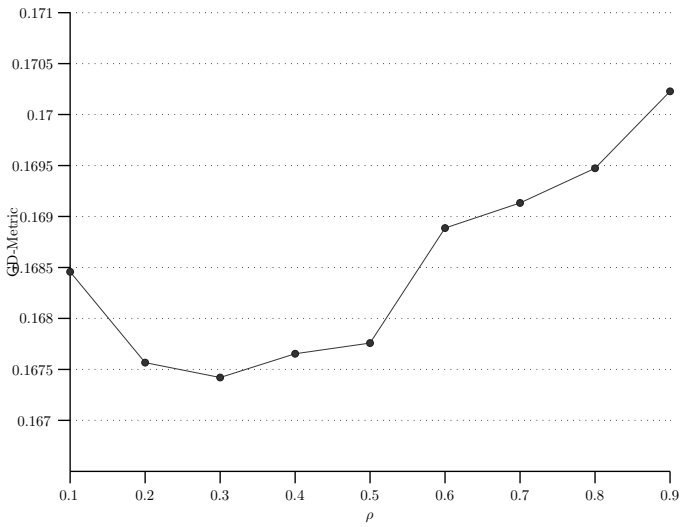

Figure 5.5: Mean GD-metric performance of MACE-gD, over all objectives for the WFG9 test problem.

persistent for a range of ρ values. This behaviour can be attributed to the fact that the Pareto optimal solutions are well distributed in MACE-gD while in MACE and MOEA/D, Pareto optimal solutions are clustered in a region that is getting smaller, relative to the dimension of the problem and not in absolute terms, with increasing number of dimensions. This has the effect that neighbouring solutions in MACE and especially in MOEA/D are virtually becoming duplicates of one another which leads to a relatively degraded performance for an increasing number of dimensions as shown in Fig. (5.3) and Fig. (5.6). This is further supported by the results in Chapter 4 and particularly Fig. (4.3).

5.6 Preference Articulation

Apart from convergence in MOEA algorithms, which is a relatively well defined concept, there can be no consensus on the meaning of a *well* distributed Pareto set. Apart from the theoretical difficulties, a proper definition of well distributed PF cannot be given mainly because it is contingent on the preferences of the decision maker (DM). Of what use would it be a Pareto optimal set if the solutions that are of interest to the DM are sparsely sampled if at all.

Generalised decomposition can be employed very effectively to resolve this problem, given that some information is available *a priori* about the general shape of the PF. To illustrate this 3-objective instances of WFG2-9 were used, with an evenly distributed reference PF for the generation of weighting vectors in MACE-gD, see Fig. (5.1) and Fig. (5.2). As it can be

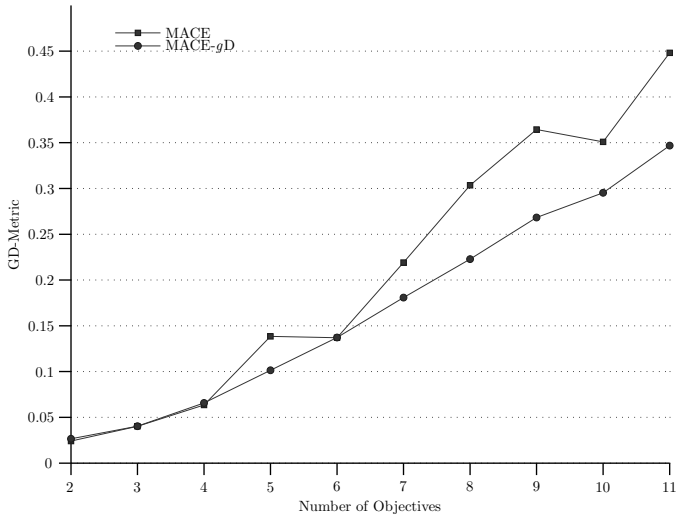

Figure 5.6: Mean GD-metric performance of MACE and MACE-gD, over all ρ values for the WFG9 test problem.

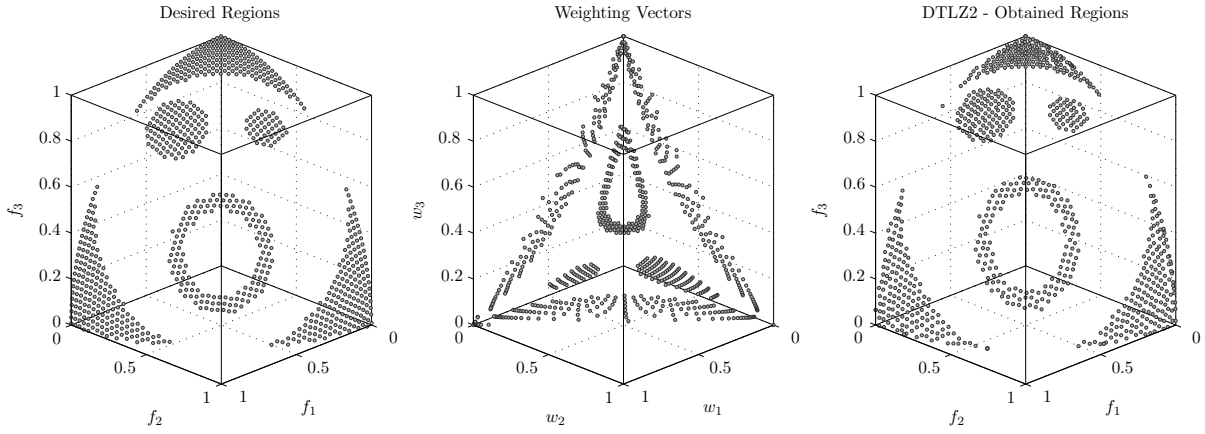

Figure 5.7: Left: Preferred regions of the Pareto front. Middle: Weighting vectors corresponding to the preferred PF regions. Right: Obtained Pareto optimal solutions on a 3-objective instance of the DTLZ2.

seen the solutions produced by MACE-gD are far *better* distributed compared to MOEA/D or RM-MEDA. It should be noted that, apart from a different reference PF for the generation of weighting vectors, all algorithm parameters are identical with the ones used in Section 5.5.3. Furthermore, a 3-objective DTLZ2 instance was also employed, a test problem with a concave PF, and a set of regions on an artificially generated PF selected manually, see Fig. (5.7). These regions represent the desired parts of the PF, potentially because other parts are of no interest to the DM. The set of points seen in the left figure in Fig. (5.7) is the set,

$$C = C_1 \cup C_2 \cup C_3 \cup C_4,$$

and the sets C_1, C_2, C_3, C_4 are defined as follows,

$$C_1 = \{\mathbf{z} : (z_1 - c_1)^2 + (z_2 - c_2)^2 + (z_3 - c_3)^2 \geq r^2\},$$

$$r^2 = 0.65,\ \mathbf{c} = (0.33, 0.33, 0.33),$$

$$C_2 = \{\mathbf{z} : (z_1 - c_1)^2 + (z_2 - c_2)^2 + (z_3 - c_3)^2 \leq r^2\},$$

$$r^2 = 0.15,\ \mathbf{c} = (0.53, 0.23, 0.8),$$

$$C_3 = \{\mathbf{z} : (z_1 - c_1)^2 + (z_2 - c_2)^2 + (z_3 - c_3)^2 \leq r^2\},$$

$$r^2 = 0.1,\ \mathbf{c} = (0.23, 0.53, 0.8),$$

and,

$$C_4 = C_a \cap C_b,$$

$$C_a = \{\mathbf{z} : (z_1 - c_1)^2 + (z_2 - c_2)^2 + (z_3 - c_3)^2 \geq r_a^2\},$$

$$C_b = \{\mathbf{z} : (z_1 - c_1)^2 + (z_2 - c_2)^2 + (z_3 - c_3)^2 \leq r_b^2\},$$

$$r_a^2 = 0.2,\ r_b^2 = 0.27,\ \mathbf{c} = (0.63, 0.63, 0.38).$$

Subsequently, (4.11) is solved to obtain the weighting vectors corresponding to these regions and using these weighting vectors MACE-gD was able to generate a PF that highly resembles the initially chosen regions, see Fig. (5.7). This concept extends directly to MAPs as it utilises generalised decomposition which is shown to perform very well for many-objectives (see Section 4.3.3), however the results are much more difficult to visualise. It is worth mentioning that such functionality can be conceptually incorporated using a different paradigm, for instance MSOPS [95] in combination with the vector angle distance scaling (VADS) metric, locates the objective front and therefore allows areas of the Pareto front that may be discontinuous to be identified. This information allows the decision maker to steer the MSOPS algorithm search away from discontinuities and if also desired, towards areas of designer preference. MSOPS-II [96] also provides identification of the objective front and allows designer choice of where

to perform the search, but additionally is able to generate the set of objective space search vectors automatically and adaptively to cover the objective front, allowing the Pareto front to be identified more efficiently when no a-priori knowledge of the objective surface structure is available.

Lastly, although it is useful to know the geometry of the PF, it is sufficient if its general shape is known. The boundary for which the weighting vectors radically change position is the transition from concave geometry to convex geometry, see Fig. (4.1).

5.7 Summary

In this chapter, generalised decomposition, a new concept introduced in Chapter 4, was used in combination with the cross entropy method, creating MACE-gD, an algorithm for many-objective problems. Using gD it is illustrated how the weighting vectors can be selected optimally to satisfy specific requirements in the distribution of the Pareto optimal solutions along the Pareto front. Also this approach allows decomposition-based MOEAs to focus on only one performance objective, that of convergence to the PF. This can be a significant advantage over other MOEAs that have to tackle 3 performance objectives simultaneously, i.e. PF coverage, even distribution of Pareto optimal solutions on the PF and convergence. Based on gD and the CE-method, a many-objective optimisation framework was presented, whose performance with respect to the GD-metric is competitive to that of MOEA/D and RM-MEDA, for the selected problem set. Additionally, it is shown how decision maker preferences can be articulated with the help of the presented framework.

Further, promising future directions of research have been identified. One of these research directions is to further test the hypothesis presented in [126], that EDAs based on low order statistics and clustering can be used as an alternative to complex probabilistic models. The obtained results in Section 5.5.3 further support this hypothesis. Finally an adaptive scheme to identify the PF shape for problems for which such information is not available would greatly increase the application range of gD. This seems feasible since, for most problems, the shape of the PF is well formed before the algorithm has fully converged.

Chapter 6

Increasing the Pareto Front Density

6.1 Introduction

As mentioned in Section 2.6, in *a posteriori* preference articulation, the aim is to generate a representative approximation of the Pareto front. This objective is common to Pareto-based [14, 42, 45, 46], and decomposition-based algorithms [2, 154, 183]. However, if the decision maker (DM) is not completely satisfied with the obtained Pareto front, he/she can recourse to, either, a different algorithm, or, restart the preferred algorithm with different parameters in the hope that the new PF will more closely satisfy the requirements. Progressive-preference articulation algorithms [5, 14], offer an alternative approach - however the drawback is that the DM must be *in-the-loop* for the algorithm execution [14] and this can be rather demanding.

Pareto estimation, the proposed method in this chapter, alleviates these difficulties for continuous MOPs by producing more and, usually, better distributed Pareto optimal solutions along the entire Pareto front (PF). Additionally, an important feature that may be helpful to both the analyst and the decision maker is that if there is a specific region of interest on the PF, the generation of solutions can be focused on that region. This can be helpful in situations where there is a set of solutions about a part of the PF that the DM is interested in, but no solution is found by the algorithm in that region. The proposed method achieves this result by estimating the mapping of a convex set to the decision vectors corresponding to the Pareto optimal solutions obtained on the final iteration of a multi-objective optimisation algorithm. This convex set is used in lieu of the objective vectors for reasons that are clarified in Section 6.2 and Section 6.3. This mapping, identified here using a radial basis function neural network (RBFNN), is then used to generate estimates of decision vectors that would lead to Pareto optimal solutions in the neighbourhood of the original solutions. It should be noted that a RBFNN is chosen mainly

because it is computationally efficient to train and the produced results are reasonable for the selected test problems. However this choice is not restrictive and does not characterise the presented methodology, as any modelling or metamodelling method can be used instead, should the situation require it.

The idea that supports the Pareto estimation method, is that, for continuous MOPs, it can be deduced from the Karush-Kuhn-Tucker optimality conditions that the PS is piecewise continuous in the decision variables, as previously noted in [125]. This fact, combined with a reasonable approximation of the PS, can be used constructively to infer the mapping of the above-mentioned convex set to decision variables that produces Pareto-optimal solutions.

The main contributions of this chapter can be summarised as follows:

- A method, which is referred to as Pareto Estimation (PE), is presented. Given a Pareto set approximation, PE can be used to increase the number of Pareto optimal solutions, for 2 and 3-objective problems. This can be useful in a situation where the evolutionary algorithm has not produced a solution close enough to the desired location on the PF. Furthermore, the Pareto estimation method does not necessitate any alteration to the optimization algorithm that is used to produce the Pareto set and is dependent on it only as far as the quality of the PS is concerned.

- The effectiveness of PE is validated using a set of test problems, commonly used in the MOEA community, for 2 and 3-objectives. It is shown that PE can produce more Pareto optimal solutions across the entire PF with a much lower cost compared to the alternative of restarting the optimization or using an alternative algorithm to solve the MOP. Also, it is much more flexible compared with progressive preference articulation methods [14]. Although this is not the main purpose of the PE method, if it can produce more solutions on the entire PF then it should be able to increase the number of solutions in specific regions as well. Increasing the number of solutions in specific regions of the PF is the main utility of PE.

- A real-world problem, namely a 3-objective portfolio optimization problem is addressed, whereby, an increased number of Pareto optimal solutions is produced along the entire PF as well as in specific regions,with the help of the Pareto estimation method.

The remainder of this chapter is organised as follows. Related concepts and motivating ideas for the proposed method are discussed in Section 6.2. In Section 6.3 the Pareto estimation

method is described for Pareto and decomposition-based algorithms. The method is tested against a set of multi-objective optimization problems and these tests are reported in Section 6.4 and in Section 6.5 PE is applied to a 3-objective portfolio optimization problem. Lastly, in Section 6.6 difficulties, potential solutions and ideas related to the PE method are discussed and in Section 6.7 the chapter is summarised and concluded.

6.2 Related Work

Multi-objective evolutionary algorithms have had tremendous success in solving real-world problems. They have been applied in control systems [184, 185, 186], economics, finance [187, 188, 189, 190] and aerospace systems [191, 192]. This can be attributed to the fact that evolutionary algorithms (EAs) perform well for a wide range of problems for which classical methods, such as convex optimization [6], are inapplicable. However the robustness of EAs does not come for free. For example, in contrast to convex optimization, there is no guarantee of global optimality for solutions produced by evolutionary algorithms. In practice, however, there is strong evidence that very good approximations of Pareto optimal solutions are generated.

6.2.1 Metamodelling Methods in Multi-Objective Optimisation

An additional challenge that MOEAs face is that the cost in objective function evaluations for a single Pareto optimal solution to be found is relatively high. This, coupled with objective functions that can take hours or days to evaluate, is a severe limitation which is widely acknowledged in the MOEA community [193, 194, 195, 196]. A prevalent methodology employed by researchers to tackle this issue is the use of metamodelling methods in optimisation. The insight is that, if a surrogate model of the actual objective function can be created with relatively few samples, then this surrogate model can be used instead of the objective function in the optimisation process. The assumption is that the surrogate model is representative of the process and since it is relatively easy to compute, it can be used repeatedly in the optimisation process at a reduced cost compared to a more accurate model of the process.

Since the purpose of the surrogate model is to relieve the EA from evaluating an expensive objective function as much as possible, the primary selection criteria for a surrogate model are adapted accordingly. Namely, the suitability of a modelling method is judged according to: (i) the ease with which the model parameters can be identified and, (ii) the cost of one evaluation of the surrogate model which must be much smaller than that of the actual objective

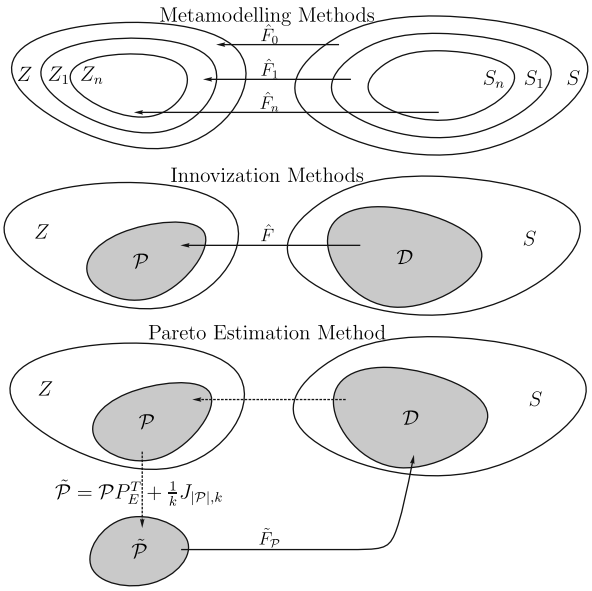

Figure 6.1: Metamodelling methods in EAs gradually refine a surrogate model and then use it to find a better Pareto set approximation. Innovization methods use the final Pareto set approximation to identify *design* rules, namely decision vector relations that map to Pareto optimal solutions. Pareto Estimation, proposed in this Chapter, proceeds in the reverse direction by mapping a surrogate set, $\tilde{\mathcal{P}}$, of a Pareto front approximation, $\mathcal{P}$, to the decision vector set that maps to $\mathcal{P}$.

function. Therefore, for a metamodelling method that satisfies the above criteria, a large number of objective function evaluations can be substituted with calls to the surrogate model, hence reducing the total cost of the optimisation. Another criterion that is definitive in the success of the aforementioned procedure is the model precision. Model precision is important because if the surrogate model cannot capture important features of the objective function the search will be grossly misled, although caution should be exercised not to overcomplicate the surrogate model to a degree that its cost becomes comparable to the original objective function. In a way, a surrogate model function can viewed as a low-pass filter, hopefully separating the *noise* from the important features of the objective function, which are its minima (or maxima). This is why such methods have been employed in noisy optimisation problems as well [197].

The general approach when substituting the real objective function with a surrogate model for use in an EA, has the following structure:

Step 1 Sample the real objective function.

Step 2 Using the obtained samples create a surrogate model.

Step 3 Use the surrogate model in the optimisation.

Step 4 If the convergence criteria are met, evaluate the solutions with the real objective function to verify their optimality and then stop; if not, go to **Step 1**.

An illustration of this iterative procedure can be seen in Fig. (6.1), where an ever more accurate mapping of the decision space, S, to the objective space Z, is created in every iteration, $\{\hat{F}_0, \hat{F}_1, \dots\}$. This approach was initially limited to serial implementations [198, 199], however later advances in metamodelling-based EAs employed local models [200], thus reinstating a key strength of EAs, their potential to be executed in parallel.

Box and Wilson [201] first used surrogate models in lieu of the true model of a process when they employed polynomial basis functions to create a model from data. This approach is commonly referred to in the literature as the response surface method (RSM). Other examples of modelling methods used in combination with an evolutionary algorithm are neural networks [195, 202, 203] (multi-layer perceptrons as well as radial basis function networks), Kriging or Gaussian processes generalised response surface methods [204], as well as Bayesian regression [205].

6.2.2 Innovization Methods

Another issue that has not yet been satisfactorily addressed, especially for many-objective problems[1], concerns the fact that the final Pareto set contains information that can be used to infer relationships in decision space that result in Pareto optimal solutions. A method that attempts to answer this question was presented by [206], which the authors call *innovization*. The authors argue that by identifying a set of *design rules* the multi-objective problem will not have to be solved again. Although this premise seems intriguing, to generate such design rules requires great effort on the behalf of the analyst, and thus is limited to very low dimensional problems in decision and objective space [207]. Another difficulty with this method is that the optimisation algorithm has to be specifically tailored to the process [206, 207, 208]. To deal with this shortcoming, further work presented in [207] attempts to resolve this by partially automating the procedure. The objective in such methods is to identify a mapping from decision space to objective space that, will guarantee that the resulting solutions will be Pareto optimal, see Fig. (6.1). This amounts to identifying a set of constraints/relationships in decision space that, if adhered to, will produce the desired results. However in these methods there is not a clear way to obtain Pareto optimal solutions in a specific region on the Pareto front, except by manually

[1] Problems with more than three objectives.

constructing different relationships on different parts of the front, something that can easily become unmanageable for even the smallest problems. This fact can be attested by the size of the problems selected in [206, 207] which never exceed $2-5$ decision variables and 2 objectives.

6.2.3 Pareto Estimation Method - Motivation

Multi-objective optimisation algorithms, in the *a posteriori* preference articulation paradigm, attempt to find a *good* approximation of the Pareto optimal set. The reason for finding this set can be due to several reasons, two of which are, (i) the decision maker does not have a clear idea of what solutions he or she prefers, and, (ii) even if preference information is available it is not guaranteed that a feasible solution exists that can satisfy the decision maker. Furthermore, there is no clear way in obtaining a specific solution on the Pareto front, even if the above-mentioned information is available.

In this chapter, a question that seems to be ignored by the literature[1] is brought forward and resolved, to some extent. Namely given an approximation of the Pareto front by any MOEA, is there a way to obtain solutions, in specific parts of the PF, that are not present in the given set, and if the answer is positive, how can this be achieved?

However, to appreciate the importance of this question, let us embark on a thought experiment. Assume that there exists a function,

$$G(\mathbf{z}) = \begin{cases} \mathbf{x} & \text{if and only if } F(\mathbf{x}) = \mathbf{z}, \text{ and } \mathbf{z} \in \mathcal{P} \\ 0 & \text{otherwise.} \end{cases} \tag{6.1}$$

The function, G, returns the corresponding Pareto optimal decision vector if a Pareto optimal solution, $\mathbf{z}$, is used and 0 otherwise. If the analyst had no information about the shape and location of the Pareto front, the function, G, would be of limited use. The function, G, is a special indicator function with domain of definition the Pareto optimal set, $\mathcal{P}$, and range the Pareto optimal decision vectors, $\mathcal{D}$. Therefore given such a function and the information about the exact location of the Pareto front, it would be simply a matter of evaluating (6.1) in order to obtain the decision vector that would result in a Pareto optimal solution. Such a description of the Pareto front geometry can be given by a parametric or non-parametric model if the problem has already been successfully solved by some method. A potential issue with such an approach is that a different description of the PF will be required for different problems. Although this seems troubling, there is nothing to preclude the existence of a function with a *convenient* domain of

[1] To the authors' best knowledge.

definition, that would map to the Pareto front of any given problem. Naturally such a function must depend, and adapt to, the Pareto optimal set or some approximation of it, and hopefully a procedure can be found to map the former to the latter. Strictly speaking, such a function would perform the following task:

$$\Pi(\mathbf{w}) = \mathbf{z},\ \mathbf{z} \in \mathcal{P}. \tag{6.2}$$

Additionally, it would be even more convenient if the mapping, Π, was predictable in the sense that, for a given $\mathbf{w}$, the location on the PF of the resulting $\mathbf{z}$ is not very hard to predict, as this would ease the complexity of using the function (6.2). A natural candidate for such a task would be an affine function, that is, a linear function plus an offset.

The final piece of this puzzle lies in the domain of definition of the function described in (6.2). The requirements on such a domain would be: (i) that points within the domain of definition of the function, Π, should be easy to obtain and, (ii) any convex combination of the points in the set must still be in the set, that is to say, the set must be convex. By adhering to these requirements, and if relations similar to (6.1) and (6.2) could be identified, then by the following procedure, a Pareto optimal solution could be obtained at any desired location on the PF:

- Choose a $\mathbf{w}$ that would produce the desired $\mathbf{z}$. This is verified by (6.2), if the resulting $\mathbf{z}$ is not the intended one; it would be sufficient to change $\mathbf{w}$ a little. In this step the *predictability* of the mapping, Π, is exploited.

- Use the obtained $\mathbf{z}$ in (6.1) to obtain the decision vector, $\mathbf{x}$, that would produce the objective vector $\mathbf{z}$.

- Evaluate the actual objective function, F, using the obtained $\mathbf{x}$ to verify that $F(\mathbf{x}) = \mathbf{z}$. Strictly speaking, should the mappings described in (6.1) and (6.2) be exact, this step is redundant.

So, if such a procedure was available in practice, there would be a way to obtain the decision vector that satisfies the requirements of the decision maker exactly, instead of repeatedly solving a multi-objective optimisation problem, in the hope of obtaining a solution that closely satisfies the aforementioned requirements.

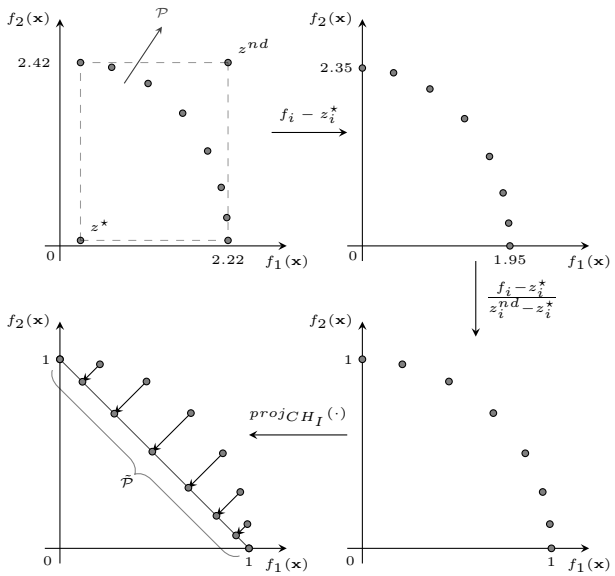

Figure 6.2: Illustration of the Π^{-1} mapping for a hypothetical Pareto set $\mathcal{P}$.

6.3 Pareto Estimation Method

6.3.1 Overview

The question posed in Section 6.2.3, is interesting because depending on how well it can be answered, the information that is in the analysts' possession increases dramatically, thus allowing the analyst to cater to more specific requests from the decision maker. This is so because given absolute knowledge of the aforementioned functions, (6.1) and (6.2), a multi-objective problem is virtually solved, as any solution on the Pareto front could be obtained with very little additional expense and yielding high precision. Obtaining the entire Pareto optimal set may be infeasible in practice, however, this is the predominant definition of what it means to *solve* a multi-objective optimisation problem [5, pp. 61].

However, such a relationship is usually unknown for real-world problems and sometimes it is unknown even for test problems. Most multi-objective optimisation algorithms strive to generate a PS which possesses two key properties; first, it should produce objective vectors as close as possible to the true PF and, second, these objective vectors should be evenly spread across the PF hyper-surface. Under the assumption that the optimisation algorithm of choice has succeeded, to a reasonable degree, in producing a PS that possesses the aforementioned properties, then the mapping, $F_{\mathcal{P}}$, of Pareto optimal objective vectors, $\mathcal{P}$, onto their corresponding decision variables $\mathcal{D}$,

$$F_{\mathcal{P}} : \mathcal{P} \to \mathcal{D} \ , \tag{6.3}$$

can be identified using a modelling method [204]. A theoretical argument based on the Karush-Kuhn-Tucker (KKT) optimality conditions, which further fosters the idea that the mapping in (6.3) should be identifiable, was proposed in [125] and is further supported by [21, 209]. The authors stated that for continuous multi-objective problems the Pareto optimal set is piecewise continuous in decision space. This point is revisited in Section 6.6. In the present work, a radial basis function neural network (RBFNN) is used for this purpose, since it is both robust and accurate for a wide set of problems [210]. The structure and further details regarding the way this type of neural network is employed is discussed in Section 6.3.2.

However, even if the mapping, $F_{\mathcal{P}}$, was explicitly known, it is still unclear how the desired Pareto optimal objective vectors should be generated in order to obtain their corresponding decision variables, using $F_{\mathcal{P}}$. This problem is related to the issue encountered in Section 6.2.3 with the function G. For example, assume that the exact mapping $F_{\mathcal{P}}$ for a multi-objective problem is given, with the only restriction being that the exact coordinates of Pareto optimal points have to be provided. This information can only be provided if the exact shape of the PF is known, meaning a mathematical description of the PF hyper-surface must be available for all potential problems. If such information is available for the given problem, then all decision variables corresponding to the PF could be obtained using $F_{\mathcal{P}}$. This point becomes clearer if the mapping $F_{\mathcal{P}}$ is seen as the inverse of the objective function $\mathbf{F}^{-1}(\cdot) = F_{\mathcal{P}}(\cdot)$, which leads to,

$$F_{\mathcal{P}}(\mathbf{F}(\mathbf{x})) = \mathbf{x}. \tag{6.4}$$

Even if the function, $\mathbf{F}(\cdot)$, is not one-to-one, a mapping $G : \mathcal{P} \to \mathcal{D}$ can still be obtained but can no longer be called the inverse image of $\mathbf{F}$; however for practical purposes its function would be the same. Therefore it is relatively *safe* to ignore for the moment the fact that the objective function $\mathbf{F}(\cdot)$ may be many-to-one; this issue is further discussed in Section 6.6.

Now, assume that the set $\mathcal{P}$ can be transformed to a set $\tilde{\mathcal{P}}$, with the only difference being that the elements of the new set are very easy to obtain and manipulate and that any element in $\tilde{\mathcal{P}}$ is mapped exactly to one element in the Pareto optimal set $\mathcal{P}$. Namely, it is required that the mapping $\Pi^{-1} : \mathcal{P} \to \tilde{\mathcal{P}}$ is one-to-one. In which case the inverse transform $\Pi : \tilde{\mathcal{P}} \to \mathcal{P}$ is obtained, and,

$$F_{\mathcal{P}}(\Pi(\tilde{\mathcal{P}})) = \mathcal{D}, \tag{6.5}$$

would enable the DM to generate any required solution. One way to produce such a mapping

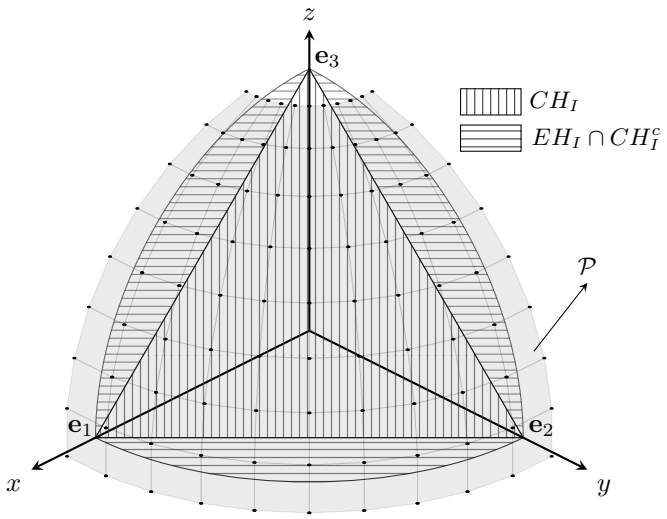

Figure 6.3: Illustration of the Π^{-1} mapping for a Pareto set $\mathcal{P}$ with 3 objectives. The points on the outer grid are in $\mathcal{P}$, while the transformed $\tilde{\mathcal{P}}$ set is within the hashed regions.

is to initially normalise the objective vectors in $\mathcal{P}$ according to,

$$\tilde{f}_i = \frac{f_i - \mathbf{z}_i^\star}{\mathbf{z}_i^{nd} - \mathbf{z}_i^\star}, \tag{6.6}$$

where $\mathbf{z}^\star$ and $\mathbf{z}^{nd}$ are estimated from the set $\mathcal{P}$. This normalisation scales the objectives in the range $[0, 1]$. The Π^{-1} mapping is illustrated in Fig. (6.2). After the normalisation the resulting objective vectors are projected onto EH_I; for problems with two objectives this is the same as CH_I. Subsequently the mapping $\tilde{F}_{\mathcal{P}}$,

$$\tilde{F}_{\mathcal{P}} : \tilde{\mathcal{P}} \to \mathcal{D} \tag{6.7}$$

is identified using a RBFNN, as shown in Fig. (6.1). This model in essence subsumes the composition of the mapping $F_{\mathcal{P}}$ and Π in (6.5).

Π^{-1} effectively takes a set of vectors in $\mathbb{R}^k$, $\mathcal{P}$, and creates its corresponding set in EH_I, $\tilde{\mathcal{P}}$. For two dimensions, vectors in $\tilde{\mathcal{P}}$ will be part of the convex set CH_I and this set will be identical to EH_I, see Fig. (6.2). For more than two dimensions, both EH_I and CH_I are still convex sets, but a more elaborate procedure will be required to obtain points on the EH_I due to its geometry, see Fig. (6.3).

For example, consider a concave Pareto front such as the one shown in Fig. (6.3). This front is the first octant of a sphere centred at the origin with radius 1.2. If the Π^{-1} transform is applied to this Pareto optimal set, the resulting $\tilde{\mathcal{P}}$ set will be on the union of the striped areas in Fig. (6.3), i.e. EH_I. The part of $\tilde{\mathcal{P}}$ in CH_I is the set within the triangle with vertices $\mathbf{e}_1$, $\mathbf{e}_2$

and $\mathbf{e}_3$. The remaining points in $\tilde{\mathcal{P}}$ are part of[1] $EH_I \cap CH_I^c$, and, since the edges of the EH_I set are curved, it is no longer straightforward to generate points within this set that are evenly distributed. Therefore the desired property of the function, Π, discussed in Section 6.2.3, that is the ability to easily generate points within its domain, would be restricted. A partial solution to this is simply to bound the domain of definitions of the Π mapping to the CH_I artificially. This would maintain the aforementioned desirable property but such a restriction would limit the method in producing solutions that their projection is within the CH_I. The solutions in $EH_I \cap CH_I^c$ correspond to *extreme* Pareto optimal points which are, potentially, of low interest [100]. However, if this assumption is not true and the decision maker requires solutions within these regions, the method described in Section 6.5.2 could be employed to obtain estimates from the PE method. This can be achieved as the entire set, $\tilde{\mathcal{P}}$, is used in the model creation process (see Section 6.3.2).

Finally, to generate the estimated Pareto optimal solutions, a set of evenly spaced convex combinations of the set $C = \{\mathbf{e}_1, \ldots, \mathbf{e}_k\}$ is created, this set is referred to as, $\mathcal{E}$. Subsequently this set can be used as an input to a model of $\tilde{\mathbf{F}}_{\mathcal{P}}$. The resulting decision vectors may then be used in the objective function to verify that they correspond to Pareto optimal objective vectors. An alternative is to create $\mathcal{E}$ for a specific region of interest in the PF, for example using points that are within the $\mathbf{conv}\, C$.

6.3.2 Radial Basis Function Neural Networks

Neural networks, or more precisely artificial neural networks[2], are widely used in an array of different disciplines [211, 212, 213]. They are well known for their *universal approximator* property [214]. Furthermore, a subclass of NNs, namely radial basis function neural networks (RBFNNs), have been shown to be robust and accurate predictors when compared to Kriging, multivariate adaptive splines and polynomial regression methods [210]. RBFNNs have a single hidden layer and an output layer. Their output layer is often comprised of linear functions since this guarantees a unique solution to their weights w [215] without the need to resort to the renowned back-propagation algorithm [216].

RBFNNs usually employ basis functions that are radially symmetric about their centres μ, for the chosen norm, and decreasing as $\mathbf{x}$ drifts away from μ. A commonly used basis function

[1] CH_I^c is the complement of the set CH_I.

[2] Artificial neural networks are simply referred to as neural networks or NNs for convenience.

is the Gaussian [215], given in its general form by,

$$\phi_i(\mathbf{x}) = \exp\left(\frac{\|\mathbf{x} - \mu_i\|^2}{2\sigma_i^2}\right), \tag{6.8}$$

where the norm $\|\cdot\|$ is often the Euclidean (ℓ_2-norm). Perhaps, at this point a difficulty associated with RBFNNs is evident, namely that, although the output layer is comprised of linear functions, the hidden layer is highly non-linear in the parameters μ and σ, which can prove a challenge in the selection of their optimal values. Various techniques are suggested in the literature addressing this problem [215]. As the number of Pareto optimal points produced in evolutionary algorithms is usually relatively low ($\ll 10\,000$), all the available Pareto optimal points are used as centres for the radial basis functions, ϕ_i. Therefore the number of basis functions is equal to the number of training vectors used. Additionally, a uniform value for the parameter, σ, is used for all basis functions, and it is set to $5 \cdot \bar{d}_\mu$, where $\bar{d}_\mu$ is the mean distance of solutions in $\tilde{\mathcal{P}}$ to their nearest neighbour. This value for σ was chosen experimentally. Intuitively, this guarantees that the basis functions have a significant overlap, thus minimising the number of regions in the interior of the set $\tilde{\mathcal{P}}$ for which no basis function is *active*. Therefore (6.8) becomes,

$$\phi_i(\mathbf{x}) = \exp\left(\frac{\|\mathbf{x} - \tilde{\mathcal{P}}_i\|_2^2}{2(5\bar{d}_\mu)^2}\right). \tag{6.9}$$

Arguably, this is the simplest way to choose the parameters of the basis functions and was used to retain focus on the proposed methodology. For a more elaborate and comprehensive methodology on selection of the parameters of RBFNNs, the reader is referred to [217].

The output of a RBFNN is a linear combination of the basis functions $\phi_i(\cdot)$,

$$y_m(\mathbf{x}) = \sum_{i=0}^{|\tilde{\mathcal{P}}|} w_{m,i}\phi_i(\mathbf{x}), \tag{6.10}$$

where $\phi_0(\cdot) = 1$ is the output layer bias term and $m \in \{1, \ldots, n\}$, where n is the number of outputs, i.e. the number of decision variables.

To validate the created neural network $(n-1)$-cross validation was used as suggested in [193]. Namely, for a Pareto set of size N, N NNs where created using $(N-1)$ samples for the training set and the remaining sample was used to estimate the generalisation error. This procedure is repeated until all the solutions in the Pareto set have been used as a test sample and then the mean square error is calculated. After estimating the NN expected generalisation error using cross validation, the final NN is generated using the entire Pareto set.

6.3.3 Pareto Dominance-Based Algorithms

The method described in Section 6.3.1 introduced the general procedure of the proposed technique, however certain details were abstracted. Optimisation algorithms based on Pareto dominance for fitness assignment have several control parameters. One of these parameters is the size of the population to be used in the optimisation process. This parameter effectively provides an upper bound on the resulting number of Pareto optimal solutions in the final set $\mathcal{P}$. One requirement for the methodology to function correctly for the entire PF is that there be a sufficient number of non-dominated solutions in the final population. An additional requirement, that is evident from experiments, is that the non-dominated set produced by the algorithm is well spread across the PF, i.e. the solutions are diverse and the mean distance from their neighbours has small variance. This simply states that the performance of the proposed method is dependent on the performance of the algorithm used to solve the MOP.

Once the execution of a multi-objective evolutionary algorithm (MOEA) has come to an end, the non-dominated solutions of the resulting set, constitute the set $\mathcal{P}$, with corresponding decision variables $\mathcal{D}$. Then each objective in $\mathcal{P}$ is normalised according to (6.6) in the range $[0, 1]$ and the ideal and nadir vectors are estimated from the set $\mathcal{P}$ as follows,

$$\mathbf{z}^{\star} = (\min\{f_1\}, \ldots, \min\{f_k\}), \tag{6.11}$$

$$\mathbf{z}^{nd} = (\max\{f_1\}, \ldots, \max\{f_k\}), \tag{6.12}$$

where f_i is the i^{th} objective function and its corresponding values for different solutions are found in the i^{th} column of $\mathcal{P}$. Note that since the produced Pareto set approximation has finite size, the inf and sup operators are replaced by the min and max operators, which return the minimum and maximum element of a set respectively. Next, the normalised set is projected onto the hyperplane E defined by $\{\mathbf{e}_1, \ldots, \mathbf{e}_{k-1}\}$ where $\mathbf{e}_i$ is a vector of zeros and a one in the i^{th} position. This is achieved by initially projecting onto the subspace[1] parallel to E and then shifting the result by $\frac{1}{k} J_{|\mathcal{P}|,k}$, where $J_{|\mathcal{P}|,k}$ is the $|\mathcal{P}| \times k$ unit matrix. To obtain the projection matrix, $k-1$ linearly independent vectors in the E plane are required. These vectors are obtained in the following way:

$$H = \left(\mathbf{e}_1 - \frac{1}{k}\mathbf{1} \cdots \mathbf{e}_{k-1} - \frac{1}{k}\mathbf{1} \right), \tag{6.13}$$

[1] The parallel plane to E that goes through the $\mathbf{0}$ vector.

where H is a $k \times (k-1)$ matrix. Subsequently the projection matrix P_E is obtained by,

$$P_E = H(H^T H)^{-1} H^T, \tag{6.14}$$

where P_E is a $k \times k$ matrix with rank $k-1$. The transformed Pareto set $\tilde{\mathcal{P}}$ is,

$$\tilde{\mathcal{P}} = \mathcal{P} P_E^T + \frac{1}{k} J_{|\mathcal{P}|,k}. \tag{6.15}$$

Finally, the neural network used to identify the mapping $\tilde{F}_P$, is created as described in Section 6.3.2, using $\tilde{\mathcal{P}}$ and $\mathcal{D}$ as the training inputs and outputs respectively.

Once the neural network is trained it can be used to create additional solutions for a new set of convex combinations $\mathcal{E}$. However, this set has to be generated by the DM according to his/her preference in a particular region of the PF; alternatively, a more densely and evenly spaced convex set spanning the entire PF could be created. The first option is likely to be preferred when the cost of evaluating the objective function is considerable or there is a clear preference towards a particular region of the PF.

The described procedure is summarised as follows:

Step 1 Obtain the non-dominated individuals from the final population of a Pareto-based MOEA, $\mathcal{P}$, and its corresponding decision variables $\mathcal{D}$.

Step 2 Normalise $\mathcal{P}$ according to (6.6).

Step 3 Project the normalised $\mathcal{P}$ onto the $k-1$ hyperplane going through $\{\mathbf{e}_1, \ldots, \mathbf{e}_{k-1}\}$ according to (6.13), (6.14) and (6.15), to produce $\tilde{\mathcal{P}}$. For 2 objectives this is the line through $(0,1)^T$ and $(1,0)^T$, and for 3 objectives, it is the plane through $(1,0,0)^T$, $(0,1,0)^T$ and $(0,0,1)^T$.

Step 4 Identify the mapping $\tilde{F}_{\mathcal{P}}$ using $\tilde{\mathcal{P}}$ and $\mathcal{D}$ as inputs and outputs, respectively, to train a RBFNN as described in Section 6.3.2.

Step 5 Create the set $\mathcal{E}$; in this case this is a set of evenly spaced convex vectors.

Step 6 Use the set $\mathcal{E}$ as inputs to the NN created in **Step 5**, to obtain estimates of decision vectors $\mathcal{D}_{\mathcal{E}}$.

Step 7 The set $\mathcal{D}_{\mathcal{E}}$ can be used with the objective function $\mathbf{F}(\cdot)$ to verify that the produced solutions are acceptable.

6.3.4 Decomposition-Based Algorithms

The multi-objective optimisation problem (2.22) is restated in the following way with the aid of the Chebyshev decomposition,

$$\begin{aligned} \min_{\mathbf{x}} g_{\infty}(\mathbf{x}, \mathbf{w}^s, \mathbf{z}^\star) &= \| \mathbf{w}^s \circ |\mathbf{F}(\mathbf{x}) - \mathbf{z}^\star| \, \|_{\infty}, \\ \forall s &= 1, \ldots, N \ , \\ \text{s.t.} \quad \mathbf{x} &\in S, \end{aligned} \tag{6.16}$$

where $\mathbf{w}^s$ are N evenly distributed weighting vectors and N is the population size and g_{∞} is the scalar objective function. The $\circ$ operator denotes the Hadamard product which is element-wise multiplication of vectors or matrices. The intuition behind this is that since g_{∞} is a continuous function of $\mathbf{w}$ [2], N evenly distributed weighting vectors should produce a well-distributed set of Pareto optimal solutions.

Consequently, since decomposition-based algorithms already have a set of convex combinations, namely the weighting vectors $\mathbf{w}$, and the correspondence of weighting vectors to objective vectors is clear, the set $\tilde{\mathcal{P}}$ can be substituted with the weighting vectors $\mathbf{w}$ that produce Pareto optimal solutions. This has the potential to greatly simplify the described procedure in Section 6.3.3. However, although this simplifies the algorithm, the choice of input vectors, $\mathbf{w}_{\mathcal{E}}$, by the DM is more difficult because of its indirect nature compared to the general method described in Section 6.3.3, and this problem becomes increasingly more difficult for increased number of objectives.

Therefore, although the method described for Pareto-based algorithms can be applied directly to decomposition-based algorithms, it is interesting to explore what would be the effect, if weighting vectors $\mathbf{w}$ are used instead of creating the set $\tilde{\mathcal{P}}$,

$$\tilde{F}_{\mathcal{P}} : \mathbf{w} \rightarrow \mathcal{D}. \tag{6.17}$$

Thus, a simplification to the Pareto estimation method is available when the MOEA used is based on decomposition, and is summarised as follows:

Step 1 Obtain the weighting vectors, $\mathbf{w}$, corresponding to non-dominated solutions.

Step 2 Identify the mapping $\tilde{F}_{\mathcal{P}}$ using $\mathbf{w}$ and $\mathcal{D}$ as inputs and outputs, respectively, to train a RBFNN.

Step 3 Generate a new set of weighting vectors $\mathbf{w}_{\mathcal{E}}$ in the PF region of interest, or using one of the methods discussed so far.

Step 4 Use the set $\mathbf{w}_{\mathcal{E}}$ as inputs to the neural network created in **Step 2**, to obtain estimates of decision vectors $\mathcal{D}_{\mathbf{w}}$.

Step 5 The set $\mathcal{D}_{\mathbf{w}}$ can be used with the objective function $\mathbf{F}(\cdot)$ to verify that the produced solutions are acceptable.

6.4 Experiment Results

To test the merits of the proposed method, the Pareto-based algorithm was chosen to be NSGA-II [46] and the decomposition-based algorithm was chosen to be MOEA/D [2]. The algorithms were run 50 times, using a different seed for the random number generator on every run, for six MOPs with two and three objectives. The population size used for both algorithms was set to 101 for the two objective problems and to 276 for the three objective problems, as these values are commonly employed in benchmarks [2]. Additionally, the algorithms were allowed to run for 300 generations for the WFG problems and for 500 generations for the DTLZ problems. The DTLZ test problems are, DTLZ1 and DTLZ2 for two and three objectives. For completeness a definition of the DTLZ1–2 test problems is given:

- DTLZ1, see [166]

$$
\begin{aligned}
f_1(\mathbf{x}) &= (1 + g(\mathbf{x}))x_1x_2, \\
f_2(\mathbf{x}) &= (1 + g(\mathbf{x}))x_1(1 - x_2), \\
f_3(\mathbf{x}) &= (1 + g(\mathbf{x}))(1 - x_1), \\
g(\mathbf{x}) &= 100(n - 2)+ \\
&\qquad 100\sum_{i=3}^{n}\left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5))\right),
\end{aligned}
$$

  where n is the number of decision variables, here $n = 10$. The two dimensional problem is $F(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))^T$.

- DTLZ2, see [166]

$$
\begin{aligned}
f_1(\mathbf{x}) &= (1 + g(\mathbf{x}))\cos\left(\frac{x_1\pi}{2}\right)\cos\left(\frac{x_2\pi}{2}\right), \\
f_2(\mathbf{x}) &= (1 + g(\mathbf{x}))\cos\left(\frac{x_1\pi}{2}\right)\sin\left(\frac{x_2\pi}{2}\right), \\
f_3(\mathbf{x}) &= (1 + g(\mathbf{x}))\sin\left(\frac{x_1\pi}{2}\right), \\
g(\mathbf{x}) &= \sum_{i=3}^{n} x_i^2,
\end{aligned}
$$

with $n = 10$.

Additionally, the test problems WFG2-3 and WFG6-7 from the WFG toolkit [167] were used. The settings used for these test problems can be seen in Table 6.1. The parameters k and l in Table 6.1 are the *position* and *distance* related parameters respectively as defined in [167]. This

Table 6.1: Test problem settings summary.

2-Obective Problem Instances					
	# Generations	N	n	k	l
WFG	300	101	24	4	20
DTLZ1	500	101	10	-	-
DTLZ2	500	101	10	-	
3-Objective Problem Instances					
	# Generations	N	n	k	l
WFG	300	276	24	4	20
DTLZ1	500	276	10	-	-
DTLZ2	500	276	10	-	-

particular collection of test problems was chosen with several considerations in mind. First, the problem set had to be broadly used and recognised by the MOEA community. Second, the problems should be challenging and diverse. It is hoped that future research will provide further validation of the proposed methodology through experiments on more test problems as well as real-world problems. More specifically, DTLZ1 and DTLZ2 [166] have been used in numerous studies [218],[219],[2], something that is also true for the WFG toolkit [218],[219]. Furthermore, each of these problems pose a different challenge. For instance, WFG2 has a discontinuous Pareto front and is non-separable. WFG3 is also non-separable and its Pareto front is linear for two dimensions and degenerate for three or more. WFG6 has a concave Pareto front and is non-separable and unimodal; and, lastly, WFG7 is separable with a concave Pareto front and has parameter dependent bias [167]. The settings for the two algorithms were chosen in a similar fashion.

The hypothesis is that, by using the Pareto estimation methodology the number of Pareto optimal solutions available to the DM can be increased significantly and, despite the fact that on many test instances the estimated Pareto set actually turns out to be superior to the initial set this is not the intended purpose of the method and can be treated as a positive side effect. For performance assessment purposes, the ratio of the following indices is used as the main focus is the relative quality of the Pareto set, produced by MOEA/D and NSGA-II, before

the application of the proposed method and after, and not the performance of the employed algorithms in absolute terms:

- Inverted Generational Distance (IGD), introduced in [179],

$$D(A, \mathcal{P}^\star) = \frac{\sum\limits_{s \in \mathcal{P}^\star} \min\{\|A_1 - s\|_2, \ldots, \|A_N - s\|_2\}}{|\mathcal{P}^\star|}, \tag{6.18}$$

  where $|\mathcal{P}^\star|$ is the cardinality of the set $\mathcal{P}^\star$ and A is an approximation of the PF. The IGD metric measures the distance of the elements in the set A from the nearest point of the actual PF. The ratio of this metric was used as,

$$D_R(A, B) = \frac{D(A, \mathcal{P}^\star)}{D(B, \mathcal{P}^\star)}, \tag{6.19}$$

  where B is another PF approximation set. Here B is the estimated PF using the Pareto estimation methodology.

- Mean Distance to Nearest Neighbour,

$$S(A) = \frac{\sum\limits_{i=1}^{|A|} d_i}{|A|}, \tag{6.20}$$

  where d_i is,

$$d_i = \min_j \{\|f_1(\mathbf{x}_i) - f_1(\mathbf{x}_j)\|_2 + \cdots + \|f_k(\mathbf{x}_i) - f_k(\mathbf{x}_j)\|_2\}.$$

  This metric can serve as a measure of the density of solutions. Again, the ratio of this metric is used as,

$$S_R(A, B) = \frac{S(A)}{S(B)}. \tag{6.21}$$

The coverage metric, described below, was used exactly as defined in [44],

- Coverage Metric (C-Metric)

$$C(A, B) = \frac{|\{u \in B | \exists v \in A : v \preceq u\}|}{|B|}, \tag{6.22}$$

  $C(A, B) = 0$ is interpreted as: there is no solution in A that dominates any solution in B. And $C(A, B) = 1$ is interpreted as the exact opposite, i.e. all the solutions in B are dominated by at least one solution in A.

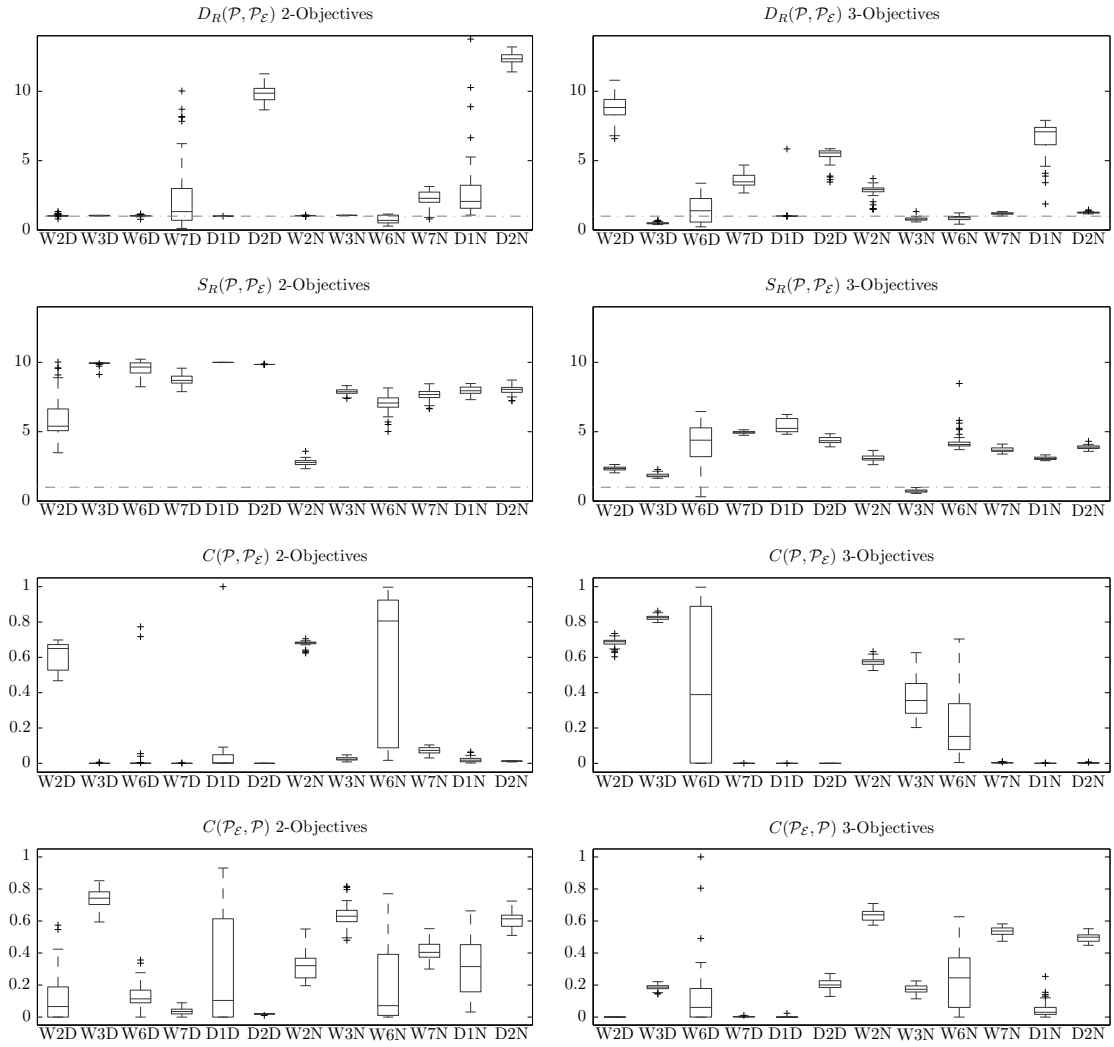

Figure 6.4: Boxplots of the experiment results of the Pareto estimation method using Pareto set approximations generated by MOEA/D and NSGA-II. The labels have the following format *Problem family:Problem number:Algorithm used*, where W refers to the WFG problem set and D to the DTLZ problem set. Also the postfix D means that the Pareto set used was produced by MOEA/D, while N by NSGA-II. For example the label *W6N* refers to results obtained for the WFG6 test problems using NSGA-II. The horizontal line in the top 4 plots marks the value 1.

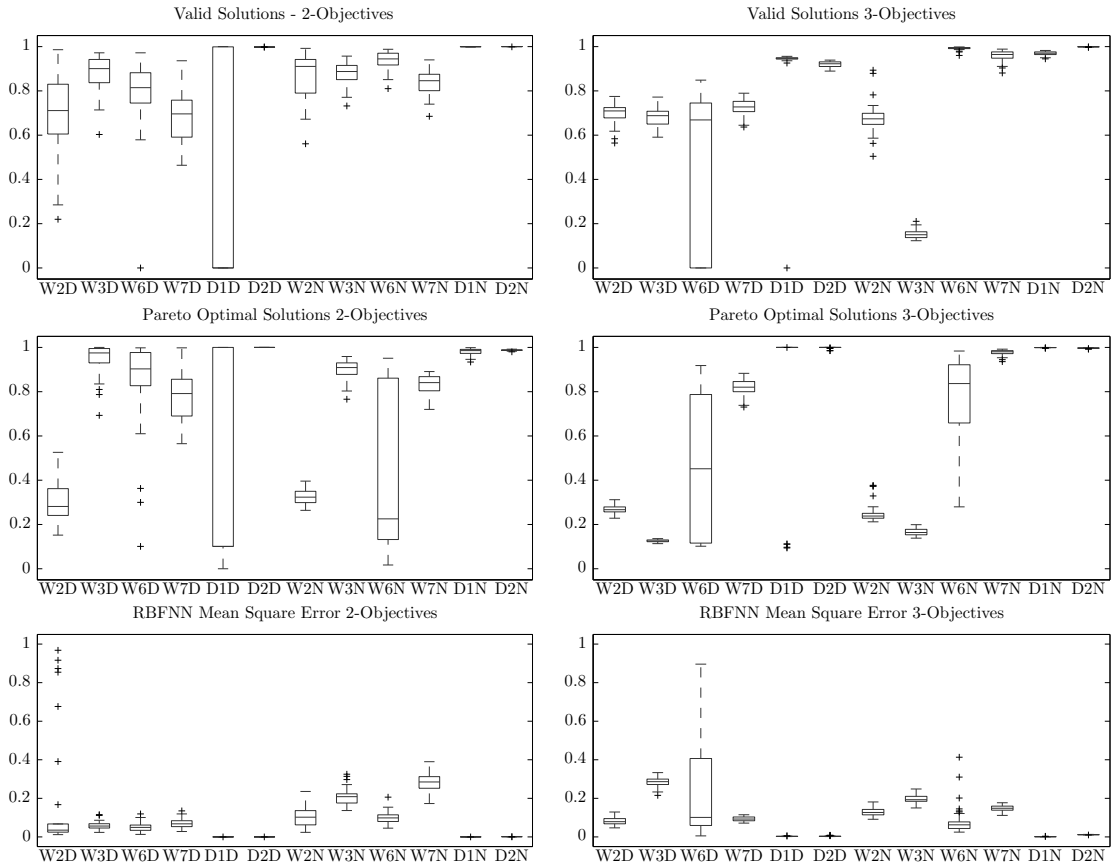

Figure 6.5: **Top row:** The number of valid solutions produced by the RBFNN in the Pareto estimation method for 2 and 3-objective problems instances, normalised to the $[0, 1]$ interval. So a value of 1 means that all produced solutions are valid, and a value of 0 that no valid solution was produced. **Middle row:** Number of Pareto optimal solutions generated by the RBFNN in the PE method, here too the values are normalised to the $[0, 1]$ interval. **Bottom row:** The mean square error (MSE) of the RBFNN. Note that all outputs of the NN have been normalised to the $[0, 1]$ interval before calculating the MSE. The labels on the x-axis have the same interpretation as in Fig. (6.4).

6.4.1 Pareto Dominance Based Algorithms

For every run of NSGA-II, with settings as explained in Section 6.4, the proposed method was applied using an evenly spaced convex set $\mathcal{E}$ of size ≈10 times greater than the initial population used in the optimisation algorithm. The set $\mathcal{E}$ was used as input to the identified mapping $\tilde{F}_{\mathcal{P}}$ resulting in the estimated decision vectors $\mathcal{D}_{\mathcal{E}}$. Subsequently $\mathcal{D}_{\mathcal{E}}$ was used with the objective function generating the objective vectors $\mathcal{P}_{\mathcal{E}}$.

Specifically, for the 2-objective test problems, the size of the set $\mathcal{P}_{\mathcal{E}}$ was set to 1 000 and for the 3-objective problems the size of the set $\mathcal{P}_{\mathcal{E}}$ was set to 3 003. The original Pareto optimal solutions used in the estimation process can be seen in Fig. (6.6) and Fig. (6.8), and the corresponding estimates $\mathcal{P}_{\mathcal{E}}$ are shown in Fig. (6.7) and Fig. (6.9). It should also be noted, as is perhaps apparent from the figures, that the entire estimated population, $\mathcal{P}_{\mathcal{E}}$, is presented and not a non-dominated subset. The same procedure was performed for all 50 runs of NSGA-II for all test problems for two and three objectives and the results are summarised in Tables 6.2-6.5 and their non-parametric counterparts are presented in Fig. (6.4). Furthermore, the number of valid solutions produced by the RBFNN, the number of Pareto optimal solutions and the RBFNN estimated generalisation error using cross validation (see Section 6.3.2), are presented in Fig. (6.5).

Table 6.2 presents the ratio of the IGD index $D_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}})$, and the mean distance to the nearest neighbour $S_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}})$ for problems with 2 objectives. The IGD index, in principle, attains smaller values the closer the set under testing is to the known PF. Additionally, if the set does not cover certain regions of the PF, this will cause the value of the IGD index to increase, signifying a degraded performance. Therefore, for this problem set, the proposed methodology is consistent in producing solutions that are at least of the same distance from the actual PF. Values of $D_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}}) > 1$ mean that the set $\mathcal{P}_{\mathcal{E}}$ produce better values for the IGD index compared to the original set $\mathcal{P}$, and for $D_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}}) < 1$ the converse is true. Regarding the mean nearest neighbour distance ratio $S_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}})$, values of $S_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}}) > 1$ mean that the mean distance from a solution to its nearest neighbour is smaller in $\mathcal{P}_{\mathcal{E}}$ compared to $\mathcal{P}$, and for $S_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}}) < 1$ the converse is true. In all cases the mean distance of neighbouring solutions in $\mathcal{P}_{\mathcal{E}}$ is much smaller. This fact combined with the results for $D_R(\mathcal{P}, \mathcal{P}_{\mathcal{E}})$ strongly indicates that the density of the available Pareto optimal solutions has significantly increased using the proposed method.

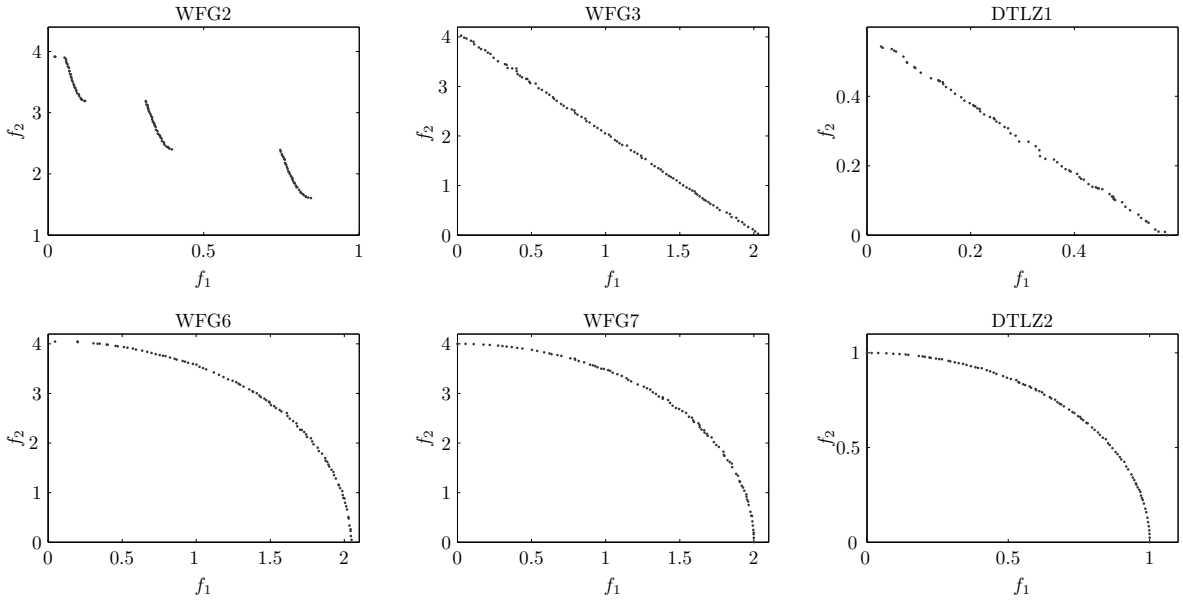

Figure 6.6: Pareto front solutions found by NSGA-II for the 2-objective problem set.

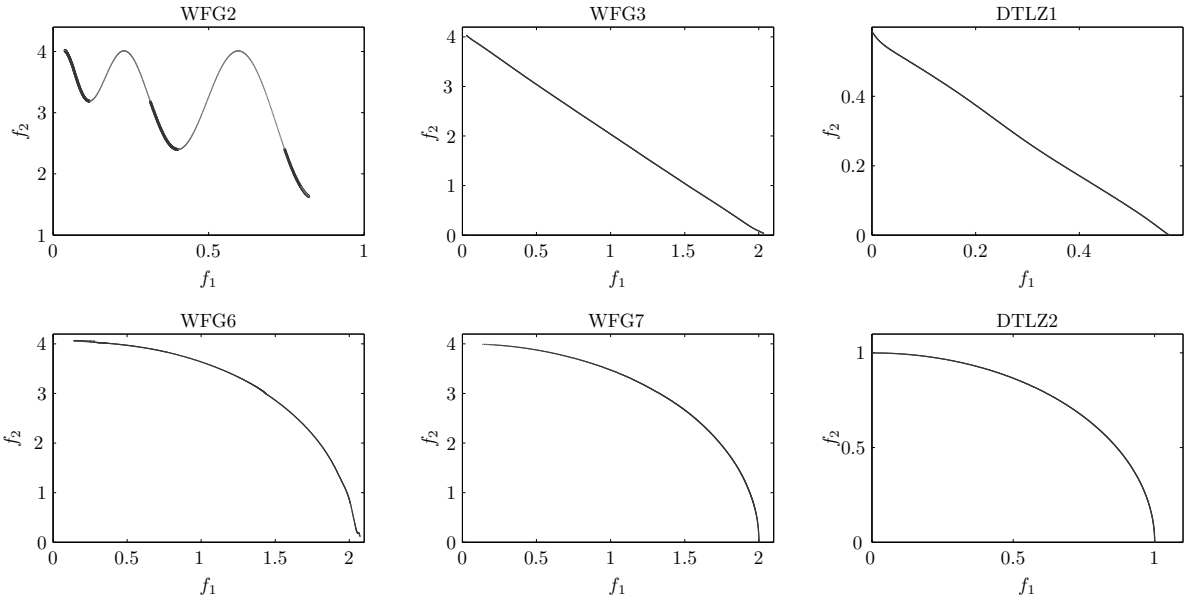

Figure 6.7: Estimated solutions $\mathcal{P}_E$ ($|\mathcal{P}_E| = 1\,000$) from the non-dominated solutions found by NSGA-II for the 2-objective problem set. The dominated solutions, for the WFG2 problem, are drawn in gray.

Table 6.2: $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ values of the solutions found by NSGA-II, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 2-objective problem set.

	$D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$		
Problem	min	mean	std	min	mean	std
WFG2	0.9879	1.0370	0.0174	2.3355	2.7844	0.2177
WFG3	1.0488	1.0589	0.0046	7.3917	7.8959	0.2286
WFG6	0.2834	0.7504	0.2730	5.0093	7.0383	0.6354
WFG7	0.7962	2.2765	0.5875	6.6541	7.6369	0.3695
DTLZ1	1.0772	4.0822	8.6262	7.3109	7.9676	0.2790
DTLZ2	11.3970	12.3377	0.3990	7.2193	8.0198	0.3061

Table 6.3: C-Metric values of the solutions found by NSGA-II, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 2-objective instances of the selected problem set.

	$C(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$C(\mathcal{P}_\mathcal{E}, \mathcal{P})$		
Problem	min	mean	std	min	mean	std
WFG2	0.6244	0.6789	0.0153	0.1959	0.3154	0.0756
WFG3	0.0080	0.0253	0.0096	0.4796	0.6306	0.0761
WFG6	0.0170	0.6046	0.3884	0.0000	0.1778	0.2115
WFG7	0.0305	0.0726	0.0185	0.3000	0.4150	0.0551
DTLZ1	0.0020	0.0192	0.0139	0.0316	0.3222	0.1814
DTLZ2	0.0080	0.0122	0.0020	0.5102	0.6108	0.0494

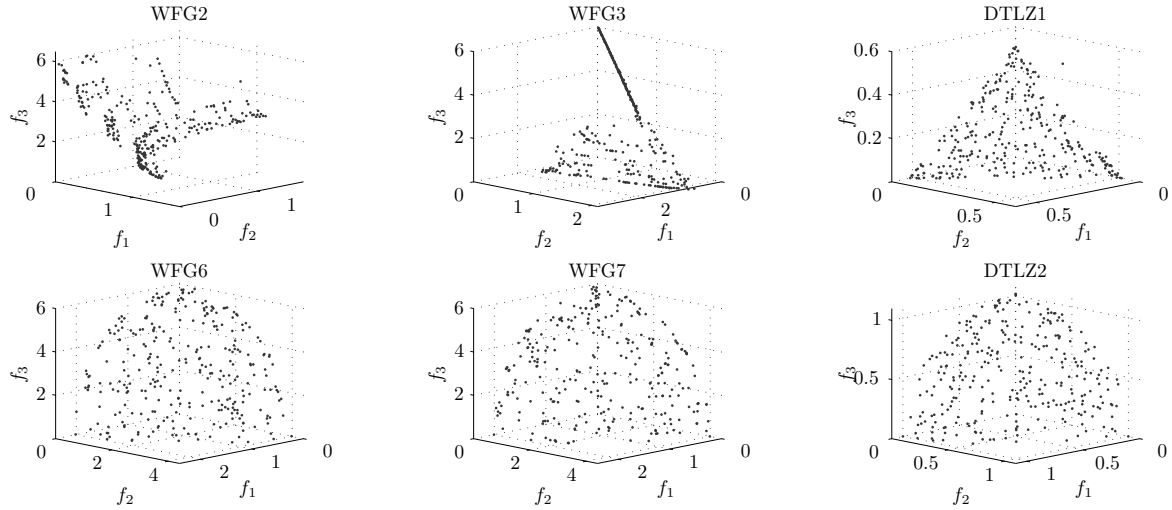

Figure 6.8: Pareto front solutions found by NSGA-II for the 3-objective problem set.

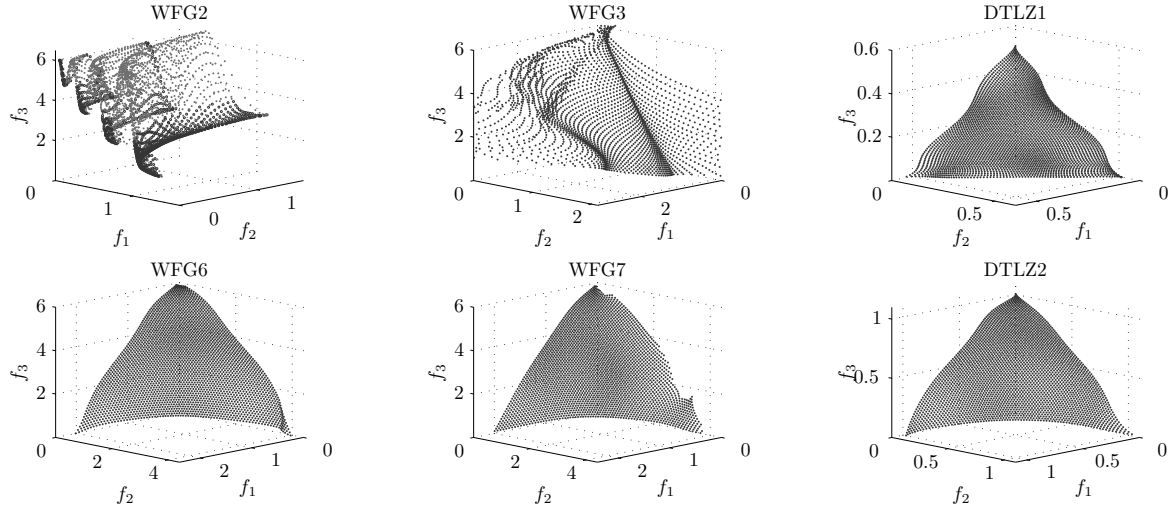

Figure 6.9: Estimated solutions $\mathcal{P}_E$ ($|\mathcal{P}_E| = 3\,003$) from the non-dominated solutions found by NSGA-II for the 3-objective problem set. The non-dominated solutions in the WFG2 test problem are the represented by the darker points on the plot.

Table 6.4: $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ values of the solutions found by NSGA-II, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 3-objective problem set.

	$D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$		
Problem	min	mean	std	min	mean	std
WFG2	1.4909	2.8047	0.4878	2.6249	3.1172	0.2306
WFG3	0.5846	0.8013	0.1292	0.5519	0.7331	0.1036
WFG6	0.4242	0.8790	0.1679	3.7070	4.2835	0.7461
WFG7	1.0079	1.1950	0.0720	3.3899	3.7008	0.1752
DTLZ1	1.8795	6.6262	1.2512	2.9125	3.0811	0.1057
DTLZ2	1.1649	1.2655	0.0587	3.5877	3.8767	0.1289

Table 6.5: C-Metric values of the solutions found by NSGA-II, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 3-objective instances of the selected problem set.

	$C(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$C(\mathcal{P}_\mathcal{E}, \mathcal{P})$		
Problem	min	mean	std	min	mean	std
WFG2	0.5253	0.5753	0.0197	0.5738	0.6342	0.0348
WFG3	0.2026	0.3773	0.1153	0.1141	0.1739	0.0262
WFG6	0.0043	0.2137	0.1768	0.0000	0.2417	0.1887
WFG7	0.0002	0.0035	0.0019	0.4733	0.5360	0.0280
DTLZ1	0.0000	0.0003	0.0004	0.0000	0.0463	0.0468
DTLZ2	0.0011	0.0027	0.0012	0.4482	0.4952	0.0264

In Table 6.3 the results for the C-metric are given for $C(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $C(\mathcal{P}_\mathcal{E}, \mathcal{P})$ for the 2-objective test problems. This metric was employed to further verify the consistency of the method. And as can be seen for all problems, except WFG2 and WFG6, the results are favourable. However it is interesting to explore the potential reasons for the less impressive performance in these two problems. Regarding WFG2, since the entire set $\mathcal{P}_\mathcal{E}$ was used, the identified PF is, as can be seen in Fig. (6.7), an oscillating function which includes dominated solutions; this is exactly the PF directly obtained from the WFG2 problem. Therefore, in a way, the method did actually perform rather well in identifying the front. A remedy to avoid such a behaviour would be that the requested solutions $\mathcal{E}$ are reasonably close to the transformed set $\tilde{\mathcal{P}}$ of the original Pareto optimal solutions $\mathcal{P}$; more elaborate methods are left for future research. And regarding the test problem WFG6, combined with the same moderate results in Table 6.2, it seems that Pareto estimation has consistent difficulties with this particular problem instance. A potential cause for these difficulties is perhaps the simplicity of the modelling technique.

Table 6.4 presents $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ indices for the 3-objective case. Again $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ has acceptable values, meaning that there is no significant sign of degradation of the IGD index. $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ shows that the mean neighbour distance is consistently lower for $\mathcal{P}_\mathcal{E}$. One noticeable feature for the values of $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ is that they are almost half of their counterparts for the 2-objective case, as seen in Table 6.2. This can partly be attributed to the *curse of dimensionality*, in the sense that to obtain similar results to Table 6.2, approximately $\mathcal{O}(n^2)$ order of solutions more than in the 2-objective case would have to be produced. This is not the case for the 3-objective instances of WFG2 and WFG3, which is mainly due to their PF geometry.

In Table 6.5 the results for the C-metric are given for $C(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $C(\mathcal{P}_\mathcal{E}, \mathcal{P})$ for the 3-objective problems. Again the results are consistent, with the observed performance for the WFG2 problem that is rather moderate for the same reasons as for the 2-objective case. The surprising fact is that for the 3-objective WFG6 problem Pareto estimation performs extremely well.

6.4.2 Decomposition-Based Algorithms

The same experimental procedure as in Section 6.4.1 is applied for the decomposition-based version of the MOEA. As previously mentioned, for this test case MOEA/D [2] was used with the same population size as NSGA-II. Instead of a set $\mathcal{E}$, an evenly distributed set of weighting vectors $\mathbf{w}_\mathcal{E}$ was used, as described in Section 6.3.4. In all other respects the experimental setup

is identical. The original Pareto optimal solutions used in the estimation process can be seen in Fig. (6.10) and Fig. (6.12), and the corresponding estimates $\mathcal{P}_\mathcal{E}$ in Fig. (6.11) and Fig. (6.13). As before, the entire estimated population $\mathcal{P}_\mathcal{E}$ is presented and used for the calculation of the statistical results. Also the results are summarised in Fig. (6.4) and Fig. (6.5).

Table 6.6 presents the ratio of the IGD index $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$, and the mean distance to the nearest neighbour $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ for problems with 2 objectives. A distinctive pattern, when compared with the corresponding values in Table 6.2, is that when the $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ index is very close to 1 the mean value for $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ is very close to 10, which is almost equal to the scaling factor that was used to increase the size of the set $\mathcal{P}_\mathcal{E}$ relative to the initial set $\mathcal{P}$. One possible reason for this behaviour, which no doubt is desirable, is that the solutions produced by MOEA/D are very well distributed across the PF. If the 2-dimensional PF is seen as a function, the fact that solutions are well distributed can be seen as sampling the function at regular intervals, hence their mean distance has low variance. This enables the employed modelling technique to better estimate the mapping, since a uniform σ value was chosen for all the basis functions, see Section 6.3.2. Another interesting fact is that, although the minimum value of $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ for the problem WFG2, is less than 1, the mean value is 1.0341 and the standard deviation is relatively small. This indicates that, in general, the method is producing good results with low deviations, for this problem instance.

In Table 6.7 the results for the C-metric are given for $C(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $C(\mathcal{P}_\mathcal{E}, \mathcal{P})$ for the 2-objective test problems. The results are very consistent, for all problems except WFG2 which is to be expected due to the shape of its PF. $C(\mathcal{P}, \mathcal{P}_\mathcal{E})$ is very close to 0, signifying that a very small number of the solutions in $\mathcal{P}_\mathcal{E}$ are dominated by solutions in the original Pareto set $\mathcal{P}$.

Table 6.8 presents $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ indices for the 3-objective case. In line with the results in Table 6.6 the $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ index is satisfactory. However, for problems WFG3 and WFG6 it seems to be somewhat low. This is to a certain extent also reflected in the $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ index. This behaviour, regarding problem WFG3, can be attributed to the fact that the real PF was not successfully identified by the algorithm, which for WFG3 is a line in 3-dimensions. This conclusion is further supported by the fact that the corresponding values for $C(\mathcal{P}, \mathcal{P}_\mathcal{E})$ in Table 6.9 are very close to 0 attesting to the fact that the produced estimated Pareto set $\mathcal{P}_\mathcal{E}$, does in fact model the given set rather well. Therefore this behaviour could be remedied by choosing the non-dominated solutions in the set $\mathcal{P}_\mathcal{E}$. However in this instance this option was avoided since this would *mask* such deficiencies, thus disallowing further insight for possible

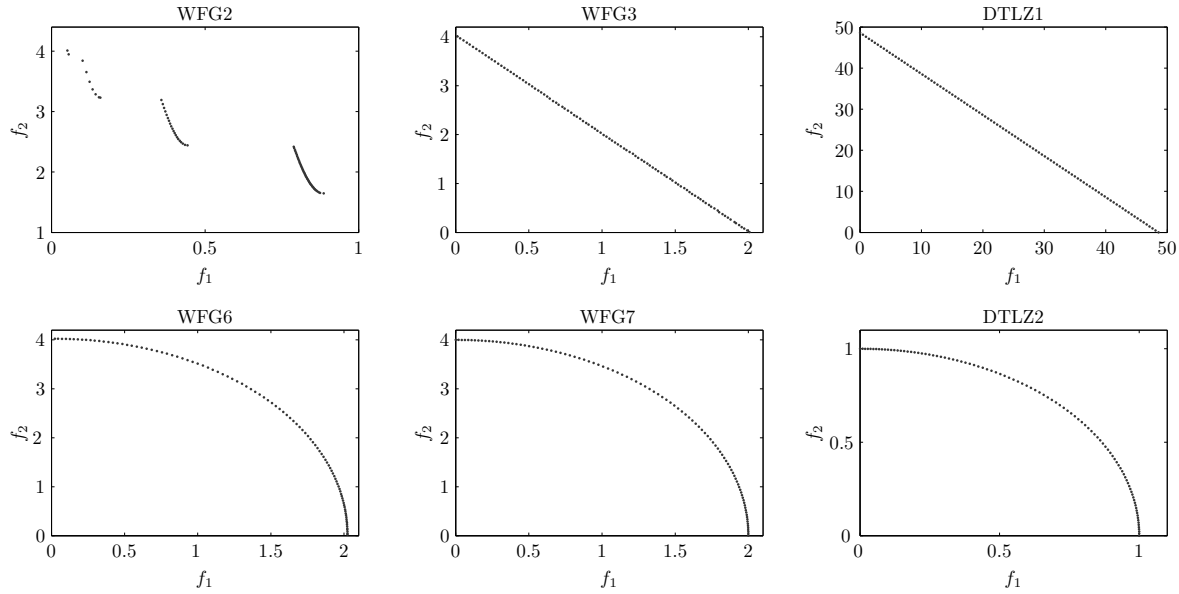

Figure 6.10: Pareto front solutions found by MOEA/D for the 2-objective problem set.

Table 6.6: $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ values of the solutions found by MOEA/D, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 2-objective problem set.

	$D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$		
Problem	min	mean	std	min	mean	std
WFG2	0.7984	1.0341	0.0742	3.4859	6.1070	1.6286
WFG3	1.0037	1.0401	0.0216	9.1222	9.9152	0.1197
WFG6	0.7774	1.0288	0.0514	8.2379	9.5327	0.5143
WFG7	0.1164	2.3724	2.5573	7.8871	8.7620	0.3967
DTLZ1	1.0000	1.0000	0.0000	9.9820	9.9932	0.0049
DTLZ2	8.6618	9.8542	0.6573	9.8320	9.8454	0.0062

improvements of the proposed methodology.

In Table 6.9 the results for the C-metric are given for $C(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $C(\mathcal{P}_\mathcal{E}, \mathcal{P})$ for the 3-objective problems.

6.5 Pareto Estimation Applied to Portfolio Optimisation

The seminal work of Markowitz [220] changed drastically the way that managers and investors decide on what portfolio of securities is appropriate for a given tolerance of risk. The main idea is that given a portfolio composition, there are two main objectives to be considered. First, the expected return is to be maximised and second, the variance of the expected return is to be minimised. Variance of a portfolio allocation is essentially a metric of risk. What was shown by Markowitz is that these two objectives are competing, namely if an investor wants extremely

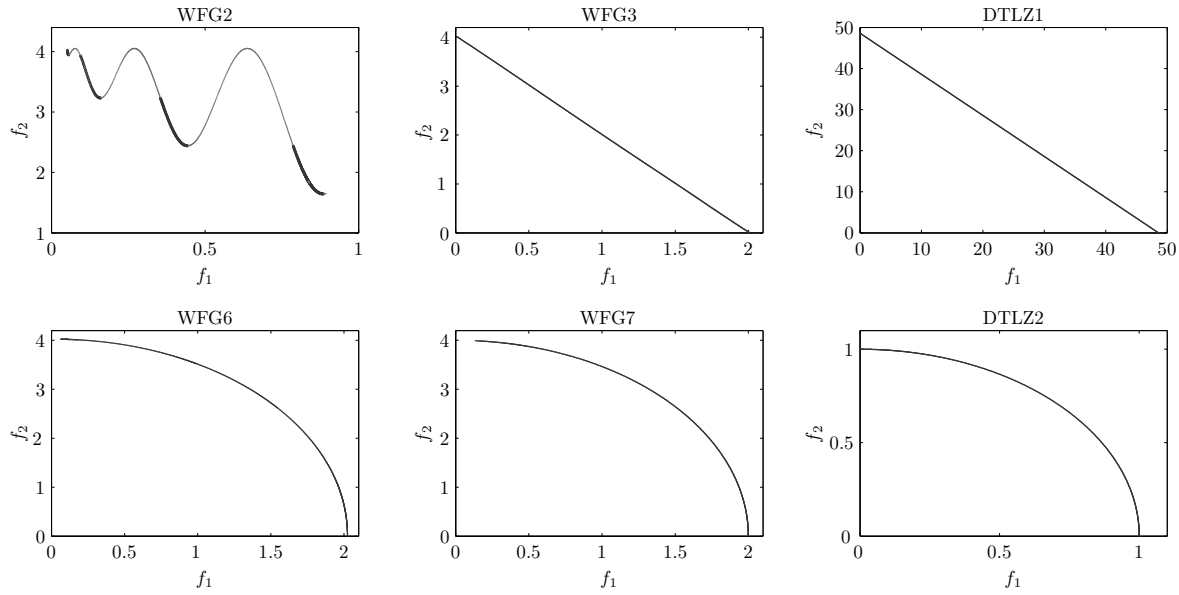

Figure 6.11: Estimated solutions $\mathcal{P}_E$ ($|\mathcal{P}_E| = 1\,000$) from the non-dominated solutions found by MOEA/D for the 2-objective problem set. The parts of the PF, for the WFG2 problem, drawn in gray represent dominated solutions.

Table 6.7: C-Metric values of the solutions found by MOEA/D, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 2-objective instances of the selected problem set.

	$C(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$C(\mathcal{P}_\mathcal{E}, \mathcal{P})$		
Problem	min	mean	std	min	mean	std
WFG2	0.4675	0.6101	0.0787	0.0000	0.1174	0.1405
WFG3	0.0000	0.0003	0.0009	0.5941	0.7430	0.0591
WFG6	0.0000	0.0323	0.1474	0.0000	0.1289	0.0745
WFG7	0.0000	0.0001	0.0003	0.0000	0.0362	0.0209
DTLZ1	0.0000	0.0441	0.1414	0.0000	0.2875	0.3366
DTLZ2	0.0000	0.0000	0.0000	0.0099	0.0196	0.0014

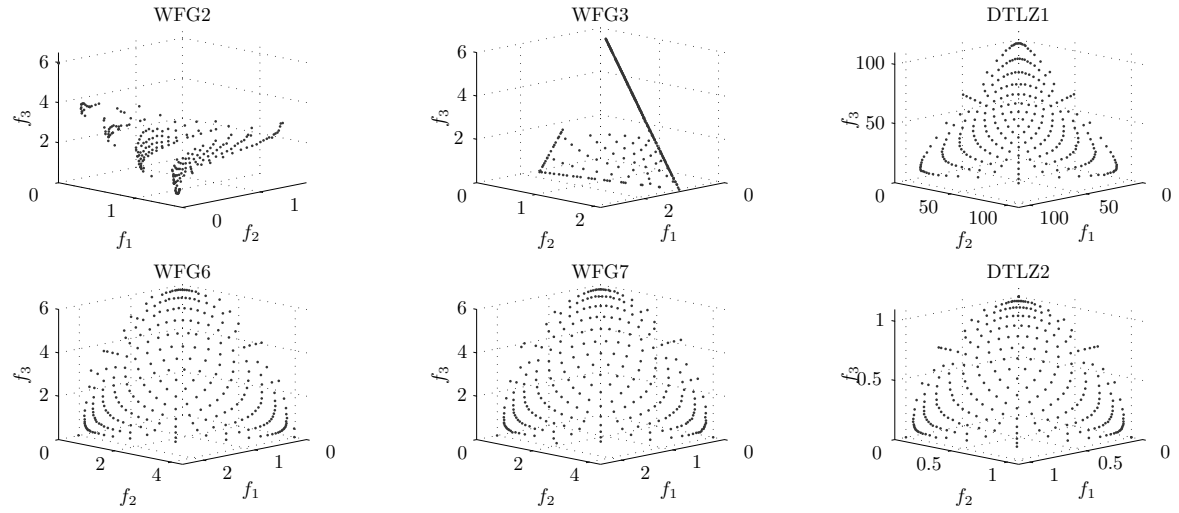

Figure 6.12: Pareto front solutions found by MOEA/D for the 3-objective problem set.

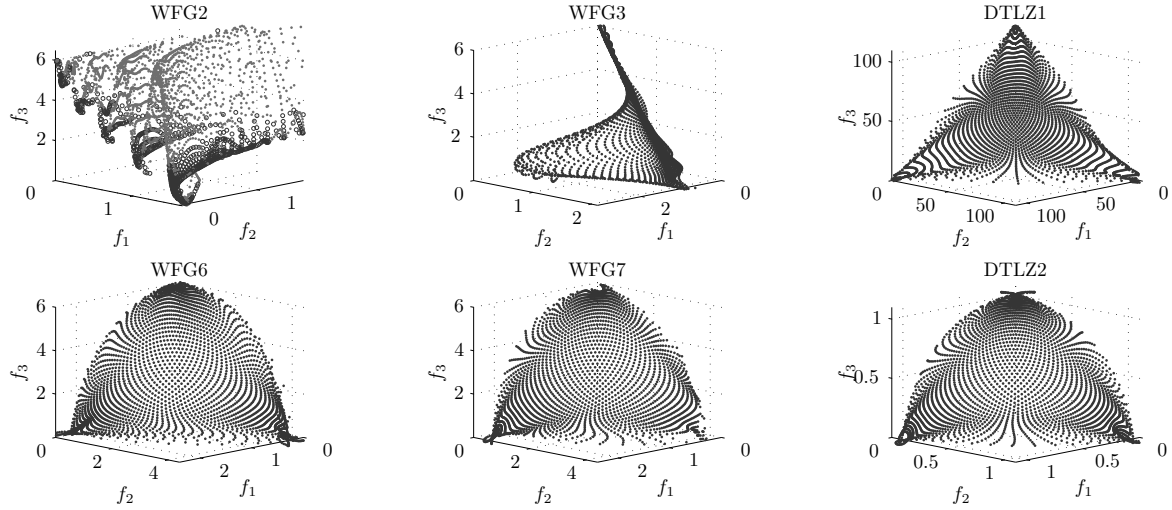

Figure 6.13: Estimated solutions $\mathcal{P}_E$ ($|\mathcal{P}_E| = 3\,003$) from the non-dominated solutions found by MOEA/D for the 3-objective problem set. The non-dominated solutions in the WFG2 test problem are the represented by the darker points on the plot.

Table 6.8: $D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ and $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ values of the solutions found by MOEA/D, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 3-objective problem set.

	$D_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$		
Problem	min	mean	std	min	mean	std
WFG2	6.5869	8.8543	0.8840	2.0372	2.3451	0.1331
WFG3	0.3963	0.4957	0.0709	1.6416	1.8425	0.1406
WFG6	0.2327	1.4326	0.8999	0.3148	4.3185	1.2262
WFG7	2.6739	3.5314	0.4975	4.7421	4.9538	0.0914
DTLZ1	1.0003	1.0983	0.6850	4.8150	5.4396	0.5015
DTLZ2	3.4609	5.3924	0.5535	3.9083	4.3850	0.2362

Table 6.9: C-Metric values of the solutions found by MOEA/D, $\mathcal{P}$, and the estimated set $\mathcal{P}_E$, for the 3-objective instances of the selected problem set.

	$C(\mathcal{P}, \mathcal{P}_\mathcal{E})$			$C(\mathcal{P}_\mathcal{E}, \mathcal{P})$		
Problem	min	mean	std	min	mean	std
WFG2	0.4675	0.6101	0.0787	0.0000	0.1174	0.1405
WFG3	0.0000	0.0003	0.0009	0.5941	0.7430	0.0591
WFG6	0.0000	0.0323	0.1474	0.0000	0.1289	0.0745
WFG7	0.0000	0.0001	0.0003	0.0000	0.0362	0.0209
DTLZ1	0.0000	0.0441	0.1414	0.0000	0.2875	0.3366
DTLZ2	0.0000	0.0000	0.0000	0.0099	0.0196	0.0014

high expected returns, then he or she must concede a high level of risk which could mean that the chance for the entire portfolio to be diminished is increased. Although not without its critics, Markowitz portfolio theory has taken the financial markets by storm and is today employed virtually by every investor. However, a criticism of this approach in selecting an *optimal* allocation of a portfolio of stocks is that the measure of risk, namely the variance of the portfolio, is not entirely realistic due to the assumption that the expected returns are normally distributed. This assumption is usually not entirely true, and as can be seen by the recent market crash, can often prove to be devastatingly flawed.

6.5.1 Portfolio Optimisation - Problem Definition

Even in the initial formulation of the portfolio optimisation problem by Markowitz [220], where the problem is actually convex, and hence it can be solved efficiently, to obtain one solution may require approximately $20 - 30$[1] objective function evaluations. Therefore, to obtain 200 Pareto optimal solutions, $4\,000 - 6\,000$ function evaluations will be necessary, on average. The Pareto estimation method, can be used to significantly decrease this number, if the decision maker has preference at a specific region of the Pareto front. However, as mentioned in Section 6.5, this simple formulation is insufficient to capture the *risk* adequately, so an extended portfolio optimisation is used.

The classical portfolio optimisation problem extended with an additional measure of risk as a third objective, namely the value-at-risk (VaR), can be defined as,

$$\begin{aligned} &\min_{\mathbf{x}} \mathbf{F}(\mathbf{x}) = (R(\mathbf{x}), V(\mathbf{x}), M(\mathbf{x}))\,, \\ &\text{subject to } \sum_{i=1}^{n} x_i = 1, \text{ and } x_i \geq 0,\ i = 1, \ldots, n, \end{aligned} \tag{6.23}$$

where the decision vector $\mathbf{x}$ represents the allocation of capital on n securities. The constraint imposed on the decision vector in (6.23) means that no gearing[2]is allowed as the maximum allocation must be equal to the available capital and the non-negativity constraint in the allocation (decision vector components) means that short positions are not allowed. A short position is one in which the investor *borrows* a security and sells it, in the hope that he can later buy it at a lower price, repay the loan by returning the security to the lender and makes a profit from the difference. Furthermore, the scalar objective functions in (6.23) are the negative of the expected return, $R(\mathbf{x})$, the portfolio variance, $V(\mathbf{x})$, and the value at risk calculated from historical data,

[1] This is an approximate number obtained by the authors.
[2] Gearing or leveraging is when securities are purchased on credit.

$M(\mathbf{x})$. The problem defined in (6.23) closely follows the formulation used in [189]. However, contrary to the work in [189], a non-parametric method to calculate the portfolio VaR is employed, instead of using the simplified VaR. Specifically these objectives are defined as follows,

$$\begin{gathered} R(\mathbf{x}) = -\frac{1}{N-1}\sum_{i=1}^{N-1} \ln\left(\frac{\mathbf{x}^T\mathbf{r}_{i+1}}{\mathbf{x}^T\mathbf{r}_i}\right), \\ \mathbf{r}_i = (r^i_{s_1}, \ldots, r^i_{s_n}), \end{gathered} \tag{6.24}$$

where $r^i_{s_n}$ is the return of the security s_n at time i. The expression in (6.24) represents the negative of the expected compounded return. The second objective, namely the portfolio variance, is defined as:

$$V(\mathbf{x}) = \mathbf{x}^T\Sigma\mathbf{x}, \tag{6.25}$$

where, Σ, is the covariance matrix of the underlying securities. The covariance matrix is calculated using historic data, as is the case for the value-at-risk, see Section 6.5.3. Lastly, the third objective is the value-at-risk calculated by a non-parametric method via historic simulation, see for example [221, 222],

$$\begin{gathered} M(\mathbf{x}) = VaR^{t+1}_\alpha, \\ VaR^{t+1}_\alpha = -\inf_y\left\{y \in \mathbb{R} : P\left(\ln\left(\frac{\mathbf{x}^T\mathbf{r}_{t+1}}{\mathbf{x}^T\mathbf{r}_t}\right) \leq y\right) \geq \alpha\right\}, \\ \text{if } VaR^{t+1}_\alpha < 0, \text{ then } M(\mathbf{x}) = 0, \\ \alpha \in (0,1), \end{gathered} \tag{6.26}$$

where α is the probability of a return smaller than y. In essence, VaR quantifies the potential loss in a portfolio with probability, α. Also if $M(\mathbf{x})$ becomes negative, this translates to positive returns ($y > 0$) in the *worst case* scenario, which means there is no risk in the investment, as far as VaR is concerned, so $M(\mathbf{x})$ is assigned to 0. A negative value could be assigned, however this has the potential to reduce the portfolio diversification which is generally undesirable [222]. For example, if a security has never had extreme variations in its price, then it would appear that it is *safe*, so if $M(\mathbf{x})$ is allowed to be negative (i.e. guaranteed positive returns), then a clear strategy would be to allocate a large proportion of the capital to this security. However, this will reduce the portfolio diversification and increase its sensitivity to the aforementioned security. So, should this security exhibit a large negative *swing*, the entire portfolio would follow. An even more conservative approach would be to assign a lower bound on VaR for securities whose historic price has never exhibited extreme variations. Since VaR can fail to account for risk due

to lack of portfolio diversification [222] and the variance of the portfolio is insensitive to extreme events, the two objectives $V(\mathbf{x})$ and $M(\mathbf{x})$ complement each other well.

6.5.2 Decision Making Procedure

Given a Pareto set approximation, $\mathcal{P}$, and using the Pareto estimation method, the decision maker has the opportunity to request a solution that is not present in the original Pareto set approximation. To illustrate this, consider the following scenario. Let us assume that the decision maker is interested in a solution, $\tilde{\mathbf{z}} \notin \mathcal{P}$, that is within the convex hull of the following solutions, $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3 \in \mathcal{P}$. Without the PE method, a solution to this would be to re-start the optimisation process, use another optimisation algorithm or involve the decision maker in the optimisation procedure using some preference articulation method, for instance [14]. All these alternatives have a high cost in function evaluations and are not guaranteed to produce the desired results. However, while the PE method cannot guarantee positive results either, it does enable the analyst to try and satisfy the DM's request at a much lower cost. A way to leverage the Pareto estimation method could be the following:

- Request, the decision maker to specify the regions of interest by selecting points from the obtained Pareto set.
- For each region select 3 points $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3 \in \mathcal{P}$ that fully enclose the preferred region on the Pareto front. For 2-objective problems, 2 points would suffice.
- Project the points on to CH_I. Let these points be $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$.
- Generate points within the $\mathbf{conv}\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\}$, namely the convex hull of the set of points $\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\}$. A way to achieve this is to create a set of evenly spaced weighting vectors, as described in [2]. Let W be an $N \times k$ matrix of N evenly spaced weighting vectors and $k = 3$ in this example, then:

$$\tilde{W} = W \cdot \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \mathbf{z}_3 \end{pmatrix}, \tag{6.27}$$

  where the resulting matrix, $\tilde{W}$, will be comprised of points within the $\mathbf{conv}\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\}$ by definition [223].
- Use the general version of the Pareto estimation method (see Section 6.3.3) to identify the mapping $\tilde{F}_{\mathcal{P}}$.

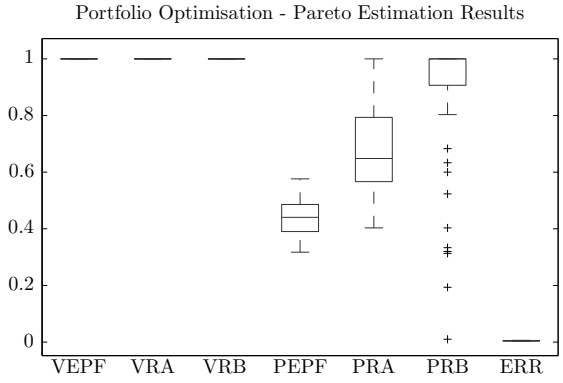

Figure 6.14: **VEPF:** Number of valid solutions generated by the PE method when considering the entire Pareto front. **VRA:** Number of valid solutions generated by the PE method for region A. **VRB:** Number of valid solutions generated by the PE method for region B. **PEPF:** Number of Pareto optimal solutions generated by the PE method when considering the entire Pareto front. **PRA:** Number of Pareto optimal solutions generated by the PE method for region A. **PRB:** Number of Pareto optimal solutions generated by the PE method for region B. **ERR:** Neural network generalisation error calculated using cross validation.

- Use the points in $\tilde{W}$ as input to the identified mapping, $\tilde{F}_{\mathcal{P}}$, to obtain a set of decision vectors, $\mathcal{D}_{\mathcal{E}}$, that will generate Pareto optimal points in the convex hull of the region enclosed by $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3$.

- Finally, using the objective function verify that the set, $\mathcal{D}_{\mathcal{E}}$, does indeed produce Pareto optimal solutions.

Following this procedure any region of interest on the Pareto front can be further explored without incurring the high cost of restarting the optimisation algorithm.

6.5.3 Portfolio Optimisation Experiments

To evaluate the Pareto estimation method on the portfolio optimisation problem defined in (6.5.1), NSGA-II was used with $N = 300$ for 350 generations, totaling 105 000 function evaluations. This procedure was performed for 50 independent runs. Furthermore, the dimension of the decision vector was set to $n = 20$, and was comprised of 20 randomly selected securities. The historic data used for the calculation of the objective function are daily opening prices for the past 3 000 trading days and were obtained from Yahoo! Finance [224]. Subsequently the PE method was used to obtain more Pareto optimal solutions for the entire Pareto front using the method described in Section 6.3.3 and two pre-specified regions using the procedure described

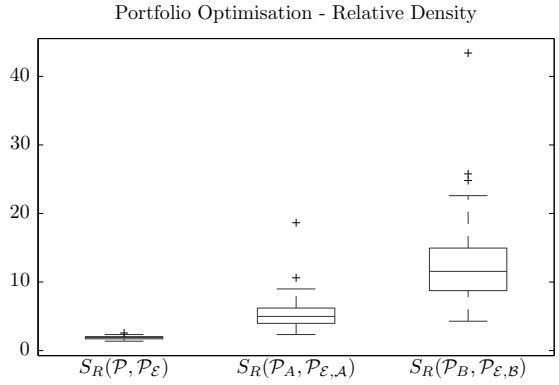

Figure 6.15: Mean distance to nearest neighbour ratio of: (i) $S_R(\mathcal{P}, \mathcal{P}_\mathcal{E})$ entire Pareto front approximation produced by NSGA-II, $\mathcal{P}$, divided by the set obtained by the PE method, $\mathcal{P}_\mathcal{E}$, for the entire PF, (ii) $S_R(\mathcal{P}_A, \mathcal{P}_{\mathcal{E},\mathcal{A}})$ the Pareto optimal solutions in the neighbourhood of region A, $\mathcal{P}_A$, divided by the set of solutions obtained by the PE method in region A, $\mathcal{P}_{\mathcal{P},\mathcal{A}}$, (iii) $S_R(\mathcal{P}_B, \mathcal{P}_{\mathcal{E},\mathcal{B}})$ the Pareto optimal solutions in the neighbourhood of region B, $\mathcal{P}_B$, divided by the set of solutions obtained by the PE method in region B, $\mathcal{P}_{\mathcal{P},\mathcal{B}}$.

in Section 6.5.2. The number of requested solutions for the entire Pareto front were 3 003 and for regions A and B 300 additional points were generated within the aforementioned regions. These results are shown in Fig. (6.16).

In Fig. (6.14) the statistics of the output of the PE method are shown. Notice that for all regions, namely the entire Pareto front and the regions A and B, all generated solutions are valid. Furthermore, the ratio of Pareto optimal solutions to dominated solutions for the case of the entire Pareto front seems to be lower when compared to regions A and B. However its median is approximately 0.41, which translates to 4 Pareto optimal solutions for every 10 generated solutions. This seems to be a fairly good ratio, since for only 3 003 function evaluations an additional 1231 Pareto optimal solutions are generated. Also, notice that for regions A and B (see Fig. (6.16)) this ratio is significantly higher. This is potentially due to the size of the requested region and the quality of the model in these parts of the PF. However the important benefit of the PE method is seen from the accuracy in location of the generated solutions in the above-mentioned regions. So, for a cost of 300 extra objective function evaluations the decision maker has obtained more than 160 additional Pareto optimal solutions in regions A and B, which greatly increase the chance that a specific solution would satisfy his or her preferences assuming that the regions were selected according to Section 6.5.2. Furthermore, the mean nearest neighbour distance in the entire Pareto front as well as for the regions A and B is shown

in Fig. (6.15), and, although the increase in density of Pareto optimal solutions for the entire Pareto front is modest (1.8 to 2.7 times larger density), the increase in density in regions A and B is phenomenal. In real terms, and given the fact that the solutions are very well distributed within the above regions (see Fig. (6.16)), this increase in the density of Pareto optimal solutions means that for any desired solutions within these regions the DM will be able to find one that is 4 to 6 and 9 to 15 times[1] closer to the exact location of the preferred Pareto optimal solution within region A and B respectively.

6.6 Discussion

This study has shown that the question posed in Section 6.2.3 is far from impossible to answer. In fact it can be answered with relative precision, as is strongly indicated by the results for the selected test problems, shown in Fig. (6.4), Fig. (6.5) and even more so for the portfolio optimisation problem, whose results are shown in Fig. (6.14) and Fig. (6.15). However, the Pareto estimation method is not without its problems. For instance, since the quality of the produced solutions depends on the employed modelling method, which in turn depends on the quality of the produced Pareto set approximation, it is to be expected that when both these factors are satisfied to a higher degree, better results are to follow. This is related to the observation in [21], about the connectedness of the Pareto optimal set in decision space for continuous multi-objective problems. Namely, if the Pareto set approximation is not *close* to the true Pareto set, this argument need not necessarily hold. For instance, such a Pareto set approximation need not necessarily be piecewise continuous, in decision space, as the Karush-Kuhn-Tucker condition would not obtain for the aforementioned PS.

As mentioned in Section 6.2.1, there are many alternative methods for identifying the mappings used in the Pareto estimation method, however since the cost of more elaborate methods renders them prohibitive for repetitive testing it is difficult to quantify the benefits of using more sophisticated identification methods and even more difficult to discern whether the results are due to the affinity of the modelling method to the particular problem set. However, when applying the Pareto estimation method to a specific real-world problem, the analyst has several options on how to proceed to identify the required mappings used in PE. An excellent work that addresses modelling issues and proposes a comprehensive approach based on neural networks is [217], wherein the entire procedure is systematised for producing high-quality models.

[1] These numbers refer to the 25th to 75th percentile in Fig. (6.15).

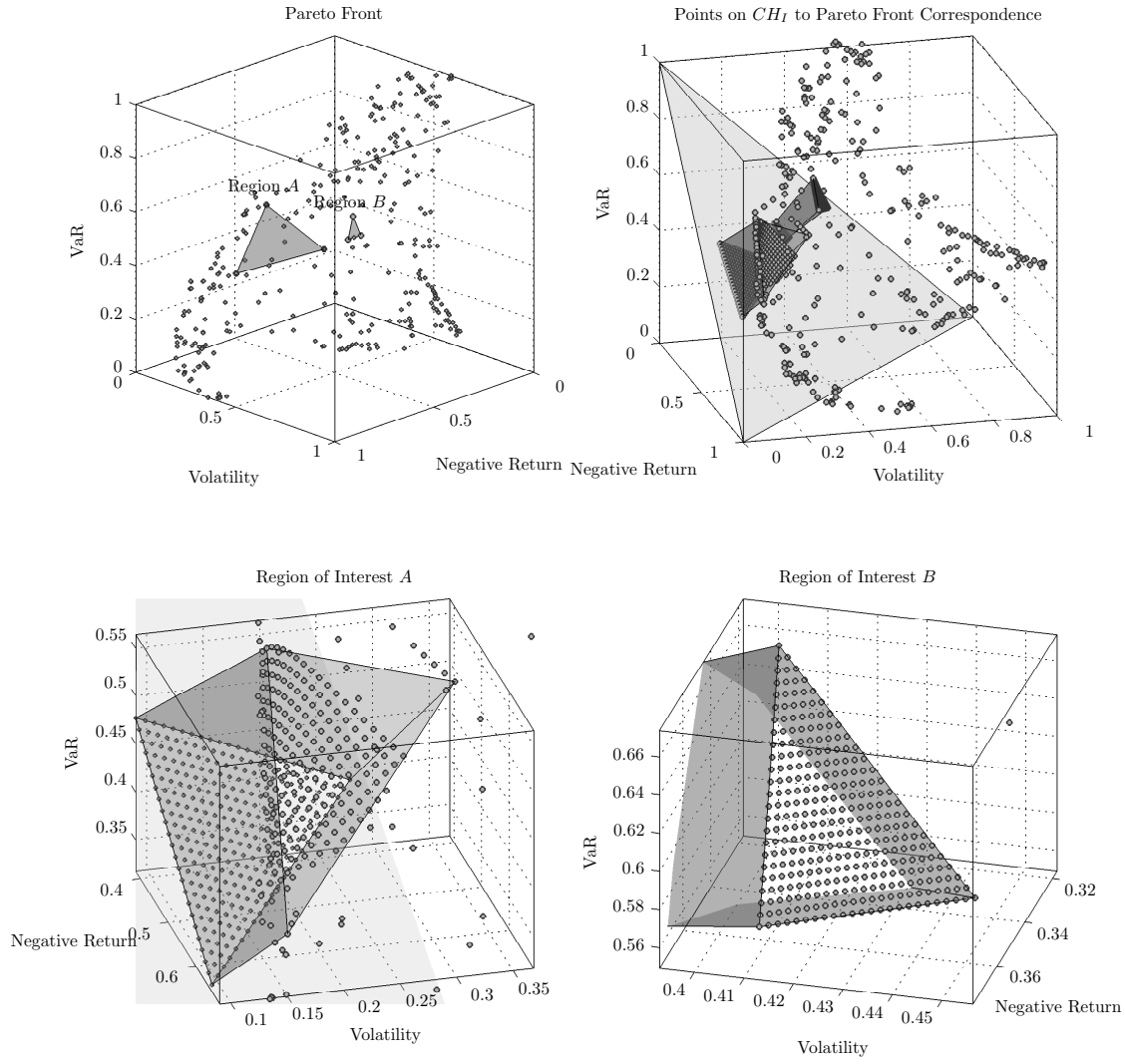

Figure 6.16: **Top left:** Portfolio optimisation Pareto front and the two regions of interest. **Top right:** The Pareto estimation method applied to identify more solutions in region A and B, the correspondence of points on the CH_I to the generated Pareto optimal solutions is marked by the shaded regions. **Bottom left:** A closer view of the generated Pareto optimal solutions for region A. **Bottom right:** A closer view of the generated Pareto optimal solutions for region B. Note that for illustration purposes, in the bottom and top right figures the Pareto front has been shifted by 0.1 in all dimensions.

Although it should be noted that, based on the results in this chapter, the radial basis function neural network proposed in Section 6.3.2, has a more than acceptable performance given the small amount of data that is usually available in a Pareto set approximation, therefore it is an excellent starting point.

Another aspect that has become evident, especially when comparing the results produced using the Pareto sets produced by NSGA-II and MOEA/D is that the distribution of the Pareto optimal solutions on the Pareto front, disregarding their convergence, seems to be an important factor determining the quality of the model. So, it would appear that if some active learning method such as that in [225] could be used, the results could potentially be improved. However, the problem of direct control of the distribution of Pareto optimal points in the PS is a very difficult one.

Lastly, the modelling employed in the Pareto estimation method operates under the assumption that the mapping from objective to decision space is a bijection, which seems to be limiting if in fact the objective function, $\mathbf{F}$, is many-to-one. However, careful consideration of this issue shows that this is not limiting to the Pareto estimation method. On the contrary, it can be rather helpful. This is based on the fact that a many-to-one objective function is *viewed* from the objective space to the decision space, for every objective vector there are one or more decision vectors to be found. This means that the probability of finding one decision vector for a specific objective vector is increased, which is to the benefit of the modelling method as there are many alternatives. Also, given the way multi-objective evolutionary algorithms operate, that is, they distribute Pareto optimal solutions across the entire Pareto front, this one-to-many relationship would be impossible to discern, as MOEAs do not preserve solutions that result in identical objective vectors. So it would be highly unlikely for a Pareto set approximation to have such *alternatives* as this in clash with MOEA objectives.

6.7 Summary

Multi-objective optimisation problem solvers seek to generate a satisfactory set of Pareto-optimal solutions to enable a decision-maker to select suitable solution. Here, a novel methodology that increases the density of available Pareto optimal solutions has been described. Using this method, the number of available solutions along the trade-off surface is increased, thereby greatly enhancing the ability of the DM to identify a suitable solution with accuracy.

This is accomplished by identifying the mapping of a transformed set, derived from an

approximation of the Pareto optimal set, to the corresponding decision vectors. This mapping was identified with the aid of a radial basis function neural network which was subsequently used to infer a number of Pareto optimal solutions $\mathcal{P}_{\mathcal{E}}$. The proposed method was presented in two forms. The first is a general formulation that is widely applicable to any multi-objective optimisation algorithm. This formulation was applied to a Pareto-based algorithm, NSGA-II, with a ten-fold increase in Pareto optimal solutions. The second form of the proposed method applies to decomposition-based algorithms. This form is motivated by the fact that by using the weighting vectors $\mathbf{w}$ in place of $\tilde{\mathcal{P}}$ the operations required to generate the latter are avoided. Both versions of the proposed method were experimentally tested against a set of well-known test problems and the results strongly indicate that the suggested methodology shows great promise.

Furthermore, the results in Section 6.4.1 and Section 6.4.2, suggest that the choice of weighting vectors in MOEA/D is not optimal, i.e. an even distribution of Pareto optimal points is not produced by the algorithm. An even distribution of Pareto optimal points is one which minimises the s-energy. The s-energy has has been shown to solve the best packing problem for a sphere, see [165, 226] for further details. This effect is transferred to the results of the proposed method that used an approximation of the PF produced by MOEA/D, see Fig. (6.12) and Fig. (6.13). In contrast with MOEA/D, the Pareto-based method produced much more uniform results, see Fig. (6.9). However, there are obvious *edge* effects, which are explained by the fact that solutions are generated only within the CH_I, see Section 6.3.1 and Fig. (6.3). This can be averted if Pareto estimation is used for specific regions, as is seen in Section 6.5.2.

Finally, although the concepts presented in this chapter can be further developed, it is the author's belief that they can alter the definition of what is currently considered to be a well distributed approximation of the PF. This is primarily due the fact that, if an inverse mapping can be identified, then the main issue becomes that of the *optimal* allocation of Pareto optimal solutions on the PF such that the process of identifying a suitable solution is facilitated. An optimal allocation would be an approximation set of the PF that provides the most information about the underlying PF. This, still unknown distribution, need not necessarily be an *even* distribution of Pareto optimal solutions. This issue is deferred to future research along with the exploration of the applicability of the presented method for many-objective optimisation problems.

Chapter 7

Conclusions and Further Research

7.1 Population-Based Multi-Objective Optimisation

A very interesting paper appeared in 1997 by Wolpert and Macready titled *"No Free Lunch Theorems for Optimization"* [178]. The impact of this work on the research direction of the evolutionary computation community was certainly significant, but in the authors' view, this work has been systematically misinterpreted. What Wolpert and Macready stated was that in the space of all functions, namely optimisation problems, there is no single algorithm that is consistently better when compared to all other optimisation algorithms. In a way, this statement seems to have a parallel in finance, specifically with respect to the *Efficient Markets* hypothesis popularised by Eugene Fama [227]. The essence of this hypothesis is that markets encompass all available information and that this is directly reflected in the price of the security, which means that there is no arbitrage opportunity. Namely, there exist no potential for profit due to under-priced or over-priced securities. This however is in contradiction to observed performance of many investors, although admittedly is in accord to the performance of many more. To abridge this parallel the investors can be represented as the optimisation algorithms and the market as the set of all optimisation problems. So the question is: why is it that some optimisation algorithms perform extremely well for a certain class of problems? The answer seems to be in both cases (algorithms and investors) the same - *information*. At least in the optimisation case, there is nothing to prevent a practitioner from using an algorithm that seems to outperform all others in a specific problem. Following this practice, namely applying the best algorithm for a specific problem, would result in a systematic increase of performance for all optimisation problems. Still, this does not mean there is a *free lunch* (or arbitrage) opportunity, it simply means that research effort needs to be spent towards identifying the *best* (algorithm,problem

type) pair, instead of trying to create a single algorithm that will be able to tackle all problems.

In line with this perspective, the information presented in Chapter 3, is a step towards this direction. Namely, it is an attempt to capture the strengths and weaknesses of certain algorithm families when applied to a specific problem class. A lack of such a broad study seems almost justified, since the number of available algorithms, even in a single family, is extremely high. The seven algorithmic families discussed in Chapter 3, capture the main body of research in evolutionary computation. However, as the progress of evolutionary algorithms has been quite rapid over the past 30 years, it would be impossible to closely examine all the presented ideas. Furthermore this pace in progress, has created a void of available *advice* to a practitioner seeking to solve a particular problem, in the sense that the amount of information available is such, that makes any sifting process very difficult and lengthy. The accomplishment of Chapter 3 is that the general concepts of evolutionary algorithms are distilled, and an *evaluation* of the performance of every algorithm family for specific problem types is presented, see Section 3.11 and Table 3.1. As mentioned in Section 3.11 the results presented in Table 3.1 are based on reported performance, frequency that a particular family is used for a specific problem and the amount of research that is conducted for a specific class of problems. Also, part of this work is admittedly subjective and most likely to be outdated given the rapid pace of EA research. However, this is merely a device to enable practitioners gain a fair understanding of fundamental concepts in EAs and provide practical guidance.

In order to further systematise such studies, a different approach in introducing novel ideas and concepts in evolutionary computation is required. Namely, it is currently very common for researchers to introduce *new algorithms* and compare their performance with other methods. However this lacks depth and rigour, as it is not always clear what is the component of the new method that makes it superior for the selected problem set. Interestingly, this idea has been proposed in the past in [182, 228], however it would seem that it has not been taken up by the EA community; perhaps the recent work of Michalewicz [229] can provide an additional incentive towards this direction.

7.2 Generalised Decomposition Many-Objective Problems

High dimensional multi-objective nonconvex problems in decision as well as in objective space are becoming more common. Evolutionary algorithms have traditionally been employed in the solution of nonconvex problems, and with considerable success. However, perhaps due to historical reasons, most EAs utilise Pareto-based methods as the means to extend the main algorithm to multi-objective problems, see Chapter 3. Such methods have been shown that lack the ability to scale gracefully for an increasing number of objectives. As shown in [4], problems begin to manifest themselves for as few as 3-4 objectives and for more than 10 objectives almost all generated solutions are incomparable. Therefore, fitness assignment becomes nearly impossible using the classical paradigm and different methods must be employed. Although, several attempts have been proposed to amend the Pareto dominance based approach, for example ε-dominance [18] and cone ε-dominance [19], the difficulties seem to persist for many-objective problems.

An alternative is found in decomposition-based methods, which use scalarising functions. Interestingly, decomposition methods have been used in nonlinear mathematical programming for more than 60 years [9]. Regardless of this fact, decomposition-based methods in evolutionary computation have started to gain more attention only in the past decade [66, 95, 96, 112, 158, 230] and, arguably due to the work of Zhang and Li [2] which rekindled the interest of the evolutionary computation community in decomposition-based methods and especially the Chebyshev scalarising function which has very interesting properties, see Section 4.2.1 and [5, pp. 98]. This seemingly disregarding attitude towards decomposition methods was not entirely unjustified. For example Das and Dennis [100] made a courageous attempt by introducing the normal boundary intersection method (NBI). NBI despite its inability to guarantee that the defined subproblems would result in Pareto optimal solutions, laid the foundation for understanding what are exactly the shortcomings of these methods. The work of Das and Dennis [100] demonstrated that there is significant difficulty in controlling the distribution of the generated Pareto front. It would seem that there was no straightforward way in selecting the weighting vectors for the classic scalarising functions, for example the weighted sum and weighted metrics methods. The understanding was that it was an issue with the scalarising functions that prevented the generation of a uniform Pareto front. However, in this work it has been shown that this is not entirely true, namely the distribution of Pareto optimal solutions across the Pareto front depends on the selected set of weighting vectors as well as the scalarising function. Addition-

ally, given a measure and a scalarising function that is convex in the weights, $\mathbf{w}$, then a set of weighting vectors that produce Pareto optimal solutions which are optimal with respect to the aforementioned measure can be found, see Chapter 4 and Chapter 5. Fortunately commonly used scalarising functions such as the weighted sum and weighted metrics for all norms including the ℓ_∞-norm which results in the Chebyshev scalarising function, are all convex. This was the key observation that led to the development of generalised decomposition, a method introduced in this work that can be used to identify a set of *optimal* weighting vectors. It should also be noted that generalised decomposition unifies and extends approaches that have appeared in the literature that provide a solution only for one or two scalarising functions, for example Hughes [95] presents a solution for the simple Chebyshev scalarising function. Another example is the recent work of Gu et al. [231] where the authors present a solution for the weighting sum and the Chebyshev function whose approach is very similar to [95]. Additionally it is shown in Giagkiozis and Fleming [232] that the selection of the norm in the weighted metrics scalarising function can have a profound effect on the convergence of the algorithm, thus must be selected with care. Furthermore the results in [232] show that the use of the Chebyshev scalarising function results in convergence rates, under certain assumptions, that are identical to Pareto-based algorithms. This result seems contradictory with reported performance of decomposition-based methods when compared to Pareto-based methods, see for example [4, 218, 233].

However, a more careful inspection of the aforementioned results reveals that while the relative performance difference of decomposition-based algorithms (often a comparison of the Chebyshev scalarising function, with Pareto-based algorithms) is in favour of decomposition-based algorithms, this performance difference is not especially significant. This small advantage might be attributed to the fact that decomposition-based algorithms employ a set of weighting vectors, which is not changing constantly, hence the problem remains fixed throughout the optimisation procedure, while in Pareto-based methods such clear direction does not exist. However, generalised decomposition in combination with the results in [232] can be used to increase the advantage of decomposition-based algorithms, subject to the provision of some prior information.

The ability to find the optimal set of weighting vectors, for a given definition of optimality and the majority of scalarising functions, can be advantageous for many-objective problems. For example, this enables theoretical studies of the effect of commonly used methods in selecting the weighting vector set for high dimensional problems in objective space (Section 4.2.2). A result from such a study performed on a weighting vector set that is evenly distributed, showed

that such a set results in subproblems that, once solved, will produce Pareto optimal points that are far from uniformly or evenly distributed, see Fig. (4.3) and Section 4.3.3. The results in Fig. (4.3) suggest that the produced Pareto optimal points are tightly clustered, and are not spread across the entire front.

Another interesting application of generalised decomposition is that for problems with known Pareto front geometry, and, a measure of optimality of well distributed solutions across the PF, the set of weighting vectors used to decompose the problem in a set of single objective subproblems can be calculated exactly. To demonstrate this an algorithm is created, many-objective cross entropy with generalised decomposition (MACE-gD), that is shown to perform very well for many-objective problems. However, the geometry of the Pareto front must be known prior to the optimisation process which can be limiting, as this information is usually not available for real-world problems. Until a better solution to this issue is found, an affine Pareto front geometry can be assumed to calculate the set of weighting vectors which results in satisfactory performance, for instance see Fig. (4.4). A solution to this would be the identification of the Pareto front geometry during the optimisation, and a promising idea that could lead to the resolution of this problem in generalised decomposition is discussed in Section 7.4.3.

7.3 Pareto Estimation

An interesting question that seems not to have been posed before is the following. Given a Pareto set approximation, is there a way to generate more Pareto optimal solutions in specific regions of the Pareto front? If such a question can be answered, then the Pareto front could be explored after the optimisation process, which is quite convenient and as shown in Chapter 6 much more efficient than restarting the optimisation algorithm until a solution that satisfies the decision maker is found. Of course at this point this question is only rhetorical, since it has been demonstrated that an answer can be found even for difficult test problems such as the WFG toolkit [167]. However, the fact that this question can be answered has several interesting implications.

One implication is that the hypothesis, first brought forward by Jin and Sendhoff [21], that the Pareto set in decision space is a $(k-1)$-dimensional piecewise continuous manifold, is further supported by the results in Chapter 6. This means that the forward or inverse mapping, such as the one presented in Section 6.3, of the Pareto optimal set can be identified. A requirement for such an identification is that the Pareto set approximation is *close* to the real Pareto set, as

the results in [21] and Section 2.5.4 are not necessarily valid for a set of solutions that fails this requirement.

Another implication is that even for problems that are expensive, in terms of the cost of one objective function evaluation, if a Pareto set is identified, then the decision maker and the analyst can collaborate much more easily if the Pareto estimation method is employed. Namely, by using Pareto estimation the decision maker can explore the Pareto front of the problem in question much more efficiently. However, as the mapping identified in the Pareto estimation method is not exact, once a desirable set of solutions is identified, it must be verified using the true objective function. However, even with that extra cost the Pareto estimation method is more efficient compared to the alternatives.

7.4 Future Perspectives

7.4.1 Disciplined Evolutionary Optimisation

It is advocated in Section 7.1 that there is a need for a more *systematic* approach in the development and presentation of new ideas and concepts in evolutionary optimisation. The main issue is that there is a tendency in this field to create methods that are pertinent to only a very small number of problems. This can be attributed to the fact that the domain of problems that EAs attempt to address is much wider, compared for instance with convex optimisation. However, there seems to be no active effort in identifying the problem features that impact the performance of the algorithm. That is to say, that there should be a systematic way to identify problem structures that can be used to identify the most appropriate method for the task. This information is important, as it can enable the automation of the entire procedure of algorithm selection and tuning, for a particular problem. In turn, once such automation is in place, the user will be much less burdened by the details of the *solver* and will have more time to properly formulate the problem and explore the resulting set of solutions. This subtle improvement can have a very distinct impact in the way that EAs are used and applied, and, the *user* will no longer have to be an expert in this field. Admittedly, this ideal is far from straightforward to achieve, as it requires a very clear overview of a wide variety of problems and algorithms in order for this *matching* to take place.

7.4.2 Estimation of Distribution Algorithms

An apparent issue with EDAs is that the probabilistic model, which is in the *core* of the main algorithm, tends to rival in complexity the actual problem [126]. In essence the *solution* process, namely the algorithm, becomes more difficult to address than the problem. Acknowledging this issue Emmendorfer and Pozo [126] suggested that an EDA based on low-order statistics and clustering can have comparable performance to high-order statistics based EDAs. Although the main topic of this work has not been estimation of distribution algorithms, it was very tempting to try and incorporate a test that would either support or reject the premise. This was achieved by using the CE method in Chapter 5 as the main algorithm of MACE and MACE-gD. Interestingly the results of MACE were competitive compared to RM-MEDA, an algorithm that is considered state-of-the-art in estimation of distribution algorithms. The evidence suggests that the way that the problem is decomposed to single objective problems (weighting vectors) is as important as the main algorithm used in the search. However since no clustering was used, the only conclusion that can be drawn is that an increase in model complexity in EDAs is not the only way to achieve performance improvements.

7.4.3 Generalised Decomposition

A problem associated with generalised decomposition is that, in its present formulation, the geometry of the Pareto front is required to be known prior to the optimisation procedure. Although, as is mentioned in Section 4.3.4, a reference Pareto front with affine geometry seems to produce better results than the alternative of selecting a evenly distributed set of weighting vectors. However, this is not utilising generalised decomposition to its full potential. Therefore it is important that an adaptive method is developed to identify the Pareto front geometry during an optimisation run. This seems to be feasible, since the geometry of the Pareto front, at least for connected fronts, is imprinted in the Pareto set approximation well before the algorithm has had the chance to converge. Therefore, if the geometry could be expressed in parametric form and converted to a convex problem, then, it would be trivial to identify the parameters, hence the geometry of the front. A parametrisation that is commonly used is the following:

$$\mathbf{z}^{p_1} + \cdots + \mathbf{z}^{p_k} = C, \tag{7.1}$$

and usually, $C = 1$ when all the objectives are normalised in the range $[0, 1]$, see for example [160, 161]. However (7.1) is a nonconvex problem, a fact that the authors of [161] failed to notice

as they attempted to solve (7.1) using the simplex method. This is reflected in the resulting solution which fails to identify models that have, $p_i \neq p_j$, for $i \neq j$. A way to sidestep this issue, is by transforming (7.1) to an infinite dimensional problem. For instance every factor, $\mathbf{z}^{p_1}$, could be represented by a set of basis functions $C_{p_1} = \{\mathbf{z}^p : p \in [a, b]\}$. If the interval, $p = [a, b]$, is discretised as, $p_d = \{a + \frac{i}{N}b : i = 1, \ldots, N\}$, then the problem in (7.1) can be expressed as,

$$a_1\mathbf{z}^{p_{1,1}} + \cdots + a_N\mathbf{z}^{p_{1,N}} + \cdots + z_1\mathbf{z}^{p_{k,1}} + \cdots + z_N\mathbf{z}^{p_{k,N}} = C, \tag{7.2}$$

which is linear in the parameters, $a_1, \ldots, a_N, \ldots, z_1, \ldots, z_N$, therefore it can be expressed in the standard $\min_{\mathbf{x}} \|\mathbf{Ax} - \mathbf{b}\|_p$ form, and solved using least squares if, $p = 2$, or in general, using convex optimisation for any, $p \geq 1$. Namely,

$$\mathbf{Ax} - \mathbf{b} = \begin{bmatrix} \mathbf{z}_1^{p_{1,1}} & \ldots & \mathbf{z}_1^{p_{1,N}} & \ldots & \mathbf{z}_1^{p_{k,1}} & \ldots & \mathbf{z}_1^{p_{k,N}} \\ \mathbf{z}_2^{p_{1,1}} & \ldots & \mathbf{z}_2^{p_{1,N}} & \ldots & \mathbf{z}_2^{p_{k,1}} & \ldots & \mathbf{z}_2^{p_{k,N}} \\ \vdots & & \vdots & & \vdots & & \vdots \\ \mathbf{z}_M^{p_{1,1}} & \ldots & \mathbf{z}_M^{p_{1,N}} & \ldots & \mathbf{z}_M^{p_{k,1}} & \ldots & \mathbf{z}_M^{p_{k,N}} \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ \vdots \\ a_N \\ \vdots \\ z_1 \\ \vdots \\ z_N \end{bmatrix} - C \cdot \begin{bmatrix} 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \\ \vdots \\ 1 \end{bmatrix}, \tag{7.3}$$

where M is the number of Pareto optimal solutions available. A problem that may arise, is that the solution of (7.3) depends on the quality of the selected basis. That is, if the true p_i, are not part of the bases then the solution to (7.3) will not be sparse. Which is quite intuitive as it will try and *recreate* a p_i using the available bases functions. Therefore, nonsparse solutions give the information that different bases are required, and, some adaptive scheme can be used to approximate the true $\{p_1, \ldots, p_k\}$ as well as possible.

7.4.4 Pareto Set Distributions and Notions of Optimality

The two main contributions of this work, namely Pareto estimation and generalised decomposition, can be used to complement each other in an interesting fashion. It is mentioned in Section 6.6 that an active learning method such as the one proposed in [225] could be instrumental to increase the effectiveness of the Pareto estimation method. The idea is that if the distribution of Pareto optimal solutions can be adapted to a distribution that results in a set of points that are maximally *informative* about the manifold of Pareto optimal solutions, then the estimated model would be better. So, if there was a way to identify this unknown distribution, that perhaps is different for every problem, generalised decomposition could be employed to gen-

erate this distribution of solutions, hence increasing the effectiveness of the Pareto estimation method.

Concluding Remarks

The transition from single objective convex problems to many-objective nonconvex problems is similar to the transition from linear dynamical systems to nonlinear systems. However when practitioners seek solution to a nonconvex problem they usually resort to a heuristic often without first considering whether there is a way to reformulate it in a form that exact solutions can be found. In the author's view this is a oversight that can be very costly. Therefore, prior to the use of a metaheuristic, a solid foundation in classic optimisation methods such as convex optimisation is necessary.

Appendix A

Mathematical Background

The purpose of this appendix is to provide a convenient and concise summary of the necessary mathematical background. The Sections A.1.1, A.1.2 and A.3 are from Rudin [234], Kolmogorov and Fomin [235] and some definitions in Section A.1 are from Shen and Vereshchagin [236]. Section A.2 is from Boyd and Vanderberghe [6]. Lastly, the definitions in Section A.4.2 are from [237].

A.1 Set Theory

A.1.1 Set Notation and Operations

A **set** is a collection of *elements*. Sets in the present work are denoted with capital letters, for example A, C, D. The notation $C = \{a, b, c, \dots\}$, means that the set, C, is comprised of the elements $a, b, c, \dots, p$. The **empty set**, namely the set that contains no elements is denoted as $C = \emptyset$. An element, a, that is part of the set, C, is denoted by $a \in C$, similarly, $a \notin C$ signifies that the element, a, is not part of the set C. Another way to define a set, apart from listing all its elements, is with the help of expressions, for instance the following expression defines a set, $C = \{x : x \in \mathbb{R}_+\}$, which is read as: the set C is comprised of all *elements* x for which the following condition is true: $x \in \mathbb{R}_+$. Furthermore, given two sets A, B, we say that $A \subseteq B$, that is, A is a **subset** of B, meaning that every element of A is also an element of B but not necessarily all elements of B are elements of A, note that $A = B$ is a possibility. An equivalent expression is to say that B is a superset of A. A **proper subset** is denoted as $A \subset B$, and is defined as $A \subseteq B$ and $A \neq B$. Two sets are said to be equal, $A = B$, if they are comprised of exactly the same elements, otherwise they are denoted as $A \neq B$.

For any two sets $A, B \subseteq \mathbb{R}$, the following operations are defined:

i. Set intersection,

$$A \cap B = \{x : x \in A \text{ and } x \in B\}. \tag{A.1}$$

ii. Set union,

$$A \cup B = \{x : x \in A \text{ or } x \in B\}. \tag{A.2}$$

iii. Set difference,

$$A \setminus B = \{x : x \in A \text{ and } x \notin B\}. \tag{A.3}$$

iv. Symmetric difference,

$$A \triangle B = (A \setminus B) \cup (B \setminus A). \tag{A.4}$$

v. Set complement,

$$A^c = A \setminus \mathbb{R}^n. \tag{A.5}$$

A.1.2 Ordered Sets

A binary relation[1] R $(\leq, \preceq)$ on a set C is said to be a **partial ordering** if,

i. R is reflexive: xRx for every $x \in C$.

ii. R is transitive: if xRy and $yRz \implies xRz$.

iii. R is antisymmetric, namely if xRy and $yRx \implies x = y$.

If a partial ordering is defined on a set C, then it is said to be **partially ordered** or **poset**. For a partially ordered set, it may happen that the relation R does not hold for all elements of the set, so the binary relation $\leq$ the following is a possibility: for $x, y \in C$, $x \not\leq y$ and $x \not\leq y$, in which case the elements x, y are incomparable - this is exactly the reason why the relation is called *partial* ordering for the set C. For example one way to extend the $\leq$ relation from $\mathbb{R}$ to $\mathbb{R}^2$ is to define it as the application of the common $\leq$ relation to the elements of the vectors in $\mathbb{R}^2$, namely, $(x_1, x_2) \leq (y_1, y_2) \iff x_1 \leq y_1$ and $x_2 \leq y_2$. Then this relation is a partial ordering for the set $C = \mathbb{R}^2$, that is for $x = (2, 2), y = (3, 4), z = (6, 5)$, $x \leq x$, also since $x \leq y$ and $y \leq z \implies (2, 2) \leq (6, 5) = x \leq z$.

If there are no incomparable elements in the set under the binary relation R, then C is said to be **ordered** (equivalently, **linearly** or **completely** or **simply** ordered) and R is said to be a **complete ordering** on the set C, or equivalently **linear** or **simple** ordering.

[1] See Section A.3.

A.2 Analysis

An element $x \in C \subseteq \mathbb{R}^n$ is called an *interior* point of C if there exists $\epsilon > 0$ for which,

$$\{y : \|y - x\|_2 \leq \epsilon\} \subseteq C, \tag{A.6}$$

i.e., there exists a ball centred at x that lies entirely in C. The set of all points interior to C is called **interior** of C and is denoted $\mathbf{int}\, C$. A set C is **open** if $\mathbf{int}\, C = C$, namely, every point in C is an interior point. A set $C \subseteq \mathbb{R}^n$ is **closed** if its complement $\mathbb{R}^n \setminus C = \{x \in \mathbb{R}^n : x \notin C\}$ is open.

The **closure** of a set C is defined as,

$$\mathbf{cl}\, C = \mathbb{R}^n \setminus \mathbf{int}\, (\mathbb{R}^n \setminus C),$$

i.e., the complement of the interior of the complement of C. A point x is in the closure of C if for every $\epsilon > 0$, there is a $y \in C$ with $\|x - y\|_2 \leq \epsilon$.

The **boundary** of the set C is defined as,

$$\mathbf{bd}\, C = \mathbf{cl}\, C \setminus \mathbf{int}\, C.$$

A **boundary point** x ($x \in \mathbf{bd}\, C$) satisfies the follwoing property: For all $\epsilon > 0$, there exist $y \in C$ and $z \notin C$ with

$$\|y - x\|_2 \leq \epsilon, \quad \|z - x\|_2 \leq \epsilon,$$

i.e., there exist arbitrarily close points in C, and also arbitrarily close points not in C. We can characterise closed and open sets in terms of the boundary operation: C is closed if it contains its boundary, i.e., $\mathbf{bd}\, C \subseteq C$. It is open if it contains no boundary points, i.e., $C \cap \mathbf{bd}\, C = \emptyset$.

A.3 Functions

The **Cartesian product** of any two sets, A, B, is defined as:

$$A \times B = \{(x, y) : x \in A, \text{ and } y \in B\}, \tag{A.7}$$

also, any subset $R \subset A \times B$ is called a **binary relation**. For example, let the set $A = \mathbb{R}^2$ and $B = \mathbb{R}$ then their Cartesian product is,

$$A \times B = \{((x, y), z) : (x, y) \in A \text{ and } z \in B\}.$$

A relation $f \subset A \times B$ is called a **partial function** (or a **partial mapping**) from A to B, if f does not contain two pairs (x, y_1) and (x, y_2) with $y_1 \neq y_2$. Namely, f is a partial function from A to B if for any $x \in A$ there exists at most one element $y \in B$ such that $(x, y) \in f$. If f is defined on all elements of A, we write $f : A \to B$ and say that f is a total function, or more commonly **function** or **mapping**.

The domain of definition, or simply domain of the function f, denoted as $\mathbf{dom}\, f$, is the set of all $x \in A$, for which a $y \in B$ exists and the **range** of the function f is the set of all $y \in B$. This can also be expressed in set notation for a total function, as: $f : A \to B$, whereby in this context the set B is the **image** (or **forward image**) of the set A under the mapping f, and, the set A is the **preimage** (or **inverse image**) of the set B under the mapping f^{-1}, denoted as $f^{-1}(B)$. A function is said to be an **into** mapping if, $f(A) \subset B$, and, an **onto** mapping if, $f(A) = B$. An onto mapping, f, is said to be **one-to-one** or a **bijection**, if every element $y \in B$ has a unique preimage $x \in A$, in which case the mapping, f^{-1}, is said to be the **inverse** of the mapping, f. Notice however, that a mapping f^{-1} usually exists, however, it is qualified as the inverse of f only if the mapping f is onto and one-to-one.

A.4 Linear Algebra

A.4.1 Notation

The notational convention that is followed in this work, in relation to matrices and vectors is quite *standard*. However, as the standard depends on background it is worthwhile to simply define the notation to promote clarity.

A vector is denoted with a bold type lower case letter, $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, and is defined to mean a column vector, namely:

$$\mathbf{x} = (x_1, x_2, \ldots, x_n) = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}.$$

The symbol T, as a superscript to a vector or a matrix, means the transpose. For example the transpose of the vector $\mathbf{x}$ is,

$$\mathbf{x}^T = (x_1, x_2, \ldots, x_n)^T = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix},$$

which is a row vector. A matrix is denoted with an upper case bold type letter, just to avoid confusion with the set notation, for example $\mathbf{A}, \mathbf{B}, \mathbf{D}$. An $m \times n$ matrix $\mathbf{A}$, is a matrix that

has m rows and n columns, and when necessary its size is specified as $\mathbf{A}_{m\times n}$. For example the matrix, $\mathbf{A}_{5\times 3}$, is a matrix with 5 rows and 3 columns. A matrix is comprised of entries, each of which is at a specific position in the matrix,

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & \cdots & a_{1,n} \\ \vdots & \ddots & \vdots \\ a_{m,1} & \cdots & a_{m,n} \end{bmatrix},$$

so the subscript to the entries identifies the location of that entry within the matrix $\mathbf{A}$, for instance the entry $a_{4,6}$, is located in row 4, column 6 of the matrix. A single subscript to a vector, $\mathbf{x}_i$, identifies an entire column, or, row if it is a transposed vector, of a matrix. Therefore a matrix can be defined as,

$$\mathbf{A} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_n \end{bmatrix},$$

or using row vectors as follows,

$$\mathbf{A} = \begin{bmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \vdots \\ \mathbf{x}_n^T \end{bmatrix}.$$

A.4.2 Fundamentals

Let a matrix, $\mathbf{A} \in \mathbb{R}^{m\times n}$, then the **column space** (or **range**) of A, denoted $\mathbf{C}(\mathbf{A})$ is defined as,

$$\mathbf{C}(\mathbf{A}) = \{\mathbf{A}\mathbf{x} : \mathbf{x} \in \mathbb{R}^n\}. \tag{A.8}$$

The **null space** (or **kernel**) of A, denoted $\mathbf{N}(\mathbf{A})$, is defined as,

$$\mathbf{N}(\mathbf{A}) = \{\mathbf{x} : \mathbf{A}\mathbf{x} = 0\}. \tag{A.9}$$

The **row space**, denoted as $\mathbf{C}(\mathbf{A}^T)$, is defined as,

$$\mathbf{C}(\mathbf{A}^T) = \{\mathbf{A}^T\mathbf{y} : \mathbf{y} \in \mathbb{R}^m\}. \tag{A.10}$$

Lastly, the **left null space**, denoted as $\mathbf{N}(\mathbf{A}^T)$, is defined as,

$$\mathbf{N}(\mathbf{A}^T) = \{\mathbf{y} : \mathbf{A}^T\mathbf{y} = 0\}. \tag{A.11}$$

These are the four fundamental subspaces in linear algebra. The **rank** of a matrix is defined as the number of pivots in the reduced row echelon form, $\mathbf{R}$, of the matrix $\mathbf{A}$. The rank of the matrix is denoted as $\mathbf{rank}\,\mathbf{A}$. The matrix $\mathbf{R}$ is obtained using Gauss-Jordan elimination [237]. The fundamental theorem of linear algebra states that:

1. The dimension of the column space, $\mathbf{C}(\mathbf{A})$, is equal to the rank of the matrix, $\mathbf{rank}\,\mathbf{A}$.
2. The dimension of the null space, $\mathbf{N}(\mathbf{A})$, is equal to: $n - \mathbf{rank}\,\mathbf{A}$.
3. The dimension of the row space, $\mathbf{C}(\mathbf{A}^T)$, is equal to the dimension of the columns space, namely, $\mathbf{rank}\,\mathbf{A}^T = \mathbf{rank}\,\mathbf{A}$.
4. The dimension of the left null space, $\mathbf{N}(\mathbf{A}^T)$, is equal to, $m - \mathbf{rank}\,\mathbf{A}$.

Furthermore, the maximum rank of a matrix is, $\min\{m, n\}$, in which case the matrix, $\mathbf{A}$, is said to have **full rank**. For example, the rank of a matrix $A \in \mathbb{R}^{6\times 2}$, can be at most 2.

Appendix B

B.1 Generating an n-dimensional Uniformly Distributed Concave or Convex Pareto Front

A moderately efficient method, but highly convenient, for generating uniformly distributed points on the unit hypersphere of arbitrary dimension is presented by Marsaglia [238]. Let n be the number of dimension of the hypersphere, then the method can be summarised as follows:

- Generate $\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n$ independent random deviates distributed according to $\mathcal{N}(0,1)$. $\mathcal{N}(0,1)$ is the normal distribution with mean 0 and variance 1.
- Calculate $S = \mathcal{X}_1^2 + \mathcal{X}_2^2 + \cdots + \mathcal{X}_n^2$, the point defined as:
$$U = \left(\frac{\mathcal{X}_1}{\sqrt{S}}, \frac{\mathcal{X}_2}{\sqrt{S}}, \ldots, \frac{\mathcal{X}_n}{\sqrt{S}} \right) \tag{B.1}$$
is uniformly distributed on the n-dimensional hypersphere [238].

With this method we can sample points on the unit hypersphere that are uniformly distributed, however these points are not Pareto optimal. To obtain a concave Pareto front with uniformly distributed points all that is necessary is to select the points that all their components are positive. If we select the points U that all their components are negative and add the vector $\mathbf{1}$, we can obtain a Pareto front with convex geometry.

However there is a limitation to the described method. Namely since we're selecting a subset of the generated solutions U, for higher dimensions in order to obtain the same number of Pareto optimal points it required that the number of uniformly distributed solutions in U is constantly increased. The required number of points in U so that a specific number of Pareto optimal solutions is obtained can be derived from the following relation that follows directly from geometric considerations,

$$|\mathcal{P}| \approx \frac{1}{2^k}|U|, \tag{B.2}$$

where $\approx$ becomes an equality in the limit as $|U| \to \infty$. For example if we require approximately 100 uniformly distributed solutions for a concave PF in 11 dimensions, then we would need 204 800 uniformly distributed vectors U on the 11 dimensional unit hypersphere, which translates to $\sim 2.2 \times 10^6$ samples from the normal distribution $\mathcal{N}(0, 1)$. So this method can easily become impractical for dimensions greater than ~ 11.

References

[1] J. Mattingley and S. Boyd, "CVXGEN: A Code Generator for Embedded Convex Optimization," *Optimization and Engineering*, pp. 1–27, 2012.

[2] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[3] R. Takahashi, E. Carrano, and E. Wanner, "On a Stochastic Differential Equation Approach for Multiobjective Optimization up to Pareto-Criticality," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science. Springer Berlin, 2011, vol. 6576, pp. 61–75.

[4] H. Ishibuchi, N. Tsukamoto, and Y. Nojima, "Evolutionary Many-Objective Optimization: A Short Review," in *IEEE Congress on Evolutionary Computation*, june 2008, pp. 2419 –2426.

[5] K. Miettinen, *Nonlinear Multiobjective Optimization*. Springer, 1999, vol. 12.

[6] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, 2004.

[7] M. Grant, S. Boyd, and Y. Ye, "Disciplined Convex Programming," vol. 84, pp. 155–210, 2006.

[8] Y. Bengio and Y. LeCun, "Scaling Learning Algorithms Towards AI," *Large-Scale Kernel Machines*, vol. 34, 2007.

[9] H. Kuhn and A. Tucker, "Nonlinear Programming," in *Proceedings of the second Berkeley symposium on mathematical statistics and probability*, vol. 1. University of California Press, Berkeley, 1951, pp. 481–492.

[10] A. M. Geoffrion, "Solving Bicriterion Mathematical Programs," *Operations Research*, vol. 15, no. 1, pp. pp. 39–54, 1967. [Online]. Available: http://www.jstor.org/stable/168510

[11] C. Vira and Y. Haimes, *Multiobjective Decision Making: Theory and Methodology*. North-Holland, 1983, no. 8.

[12] J. Schaffer, "Multiple Objective Optimization with Vector Evaluated Genetic Algorithms," in *Conference on Genetic Algorithms*. L. Erlbaum Associates Inc., 1985, pp. 93–100.

[13] D. Goldberg and J. Holland, "Genetic Algorithms and Machine Learning," *Machine Learning*, vol. 3, no. 2, pp. 95–99, 1988.

[14] C. Fonseca and P. Fleming, "Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization," in *Conference on Genetic Algorithms*, vol. 423, 1993, pp. 416–423.

[15] F. Edgeworth, *Mathematical Psychics: An Essay on the Application of Mathematics to the Moral Sciences*. CK Paul, 1881, no. 10.

[16] V. Pareto, "Cours D'Économie Politique," 1896.

[17] R. C. Purshouse and P. J. Fleming, "Conflict, Harmony, and Independence: Relationships in Evolutionary Multi-Criterion Optimisation," in *Conference on Evolutionary Multi-Criterion Optimization*. Berlin: Springer, 2003, pp. 16–30.

[18] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining Convergence and Diversity in Evolutionary Multiobjective Optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[19] L. Batista, F. Campelo, F. Guimarães, and J. Ramírez, "Pareto Cone ε-Dominance: Improving Convergence and Diversity in Multiobjective Evolutionary Algorithms," in *Evolutionary Multi-Criterion Optimization*. Springer, 2011, pp. 76–90.

[20] K. Miettinen and M. Mäkelä, "On Scalarizing Functions in Multiobjective Optimization," *OR Spectrum*, vol. 24, no. 2, pp. 193–213, 2002.

[21] Y. Jin and B. Sendhoff, "Connectedness, Regularity and the Success of Local Search in Evolutionary Multi-Objective Optimization," in *Congress on Evolutionary Computation*, vol. 3, dec. 2003, pp. 910 – 1917.

[22] C. Fonseca and P. Fleming, "Multiobjective Genetic Algorithms Made Easy: Selection Sharing and Mating Restriction," in *International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*. IET, 1995, pp. 45–52.

[23] P. Moscato, "On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms," Tech. Rep., 1989.

[24] J. Koza, *On the Programming of Computers by Means of Natural Selection*. MIT press, 1996, vol. 1.

[25] J. Holland, "Adaptation in natural and artificial systems," 1975.

[26] J. Holland, "Outline for a logical theory of adaptive systems," *Journal of the ACM (JACM)*, vol. 9, no. 3, pp. 297–314, 1962.

[27] J. Farmer, N. Packard, and A. Perelson, "The immune system, adaptation, and machine learning," *Physica D: Nonlinear Phenomena*, vol. 22, no. 1-3, pp. 187–204, 1986.

[28] D. Goldberg, "Genetic Algorithms in Search, Optimization, and Machine Learning," 1989.

[29] I. Rechenberg, "Cybernetic Solution Path of An Experimental Problem," 1965.

[30] H. Schwefel, "Evolutionsstrategie und Numerische Optimierung," Ph.D. dissertation, Technische Universität Berlin, 1975.

[31] L. Fogel and G. Burgin, "Competitive Goal-Seeking Through Evolutionary Programming," DTIC Document, Tech. Rep., 1969.

[32] M. Dorigo, V. Maniezzo, and A. Colorni, "The ant system: An autocatalytic optimizing process," *TR91-016, Politecnico di Milano*, 1991.

[33] M. Dorigo and G. Di Caro, "Ant Colony Optimization: A New Meta-Heuristic," in *Congress on Evolutionary Computation*, vol. 2. IEEE, 1999.

[34] R. Storn and K. Price, "Differential Evolution - A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces," *International Computer Science Institute-Publications TR*, 1995.

[35] R. Eberhart and J. Kennedy, "A New Optimizer Using Particle Swarm Theory," in *International Symposium on Micro Machine and Human Science*. IEEE, 1995, pp. 39–43.

[36] H. Mühlenbein and G. Paass, "From Recombination of Genes to the Estimation of Distributions I. Binary Parameters," *Parallel Problem Solving from Nature*, pp. 178–187, 1996.

[37] P. Larranaga and J. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Netherlands, 2002, vol. 2.

[38] C. Darwin, *The origin of species*. Hayes Barton Press, 1858, no. 811.

[39] K. Deb and R. Agrawal, "Simulated binary crossover for continuous search space," *Complex Systems*, vol. 50, no. 9, pp. 115–148, 1994.

[40] I. Ono and S. Kobayashi, "A real-coded genetic algorithm for function optimization using unimodal normal distribution crossover," *Journal of Japanese Society for Artificial Intelligence*, vol. 14, no. 6, pp. 246–253, 1997.

[41] H. Voigt, H. Mühlenbein, and D. Cvetkovic, "Fuzzy recombination for the breeder genetic algorithm," in *Proc. Sixth Int. Conf. on Genetic Algorithms*. Citeseer, 1995.

[42] N. Srinivas and K. Deb, "Muiltiobjective Optimization Using Nondominated Sorting in Genetic Algorithms," *Evolutionary Computation*, vol. 2, no. 3, pp. 221–248, 1994.

[43] J. Horn, N. Nafpliotis, and D. Goldberg, "A niched pareto genetic algorithm for multiobjective optimization," in *Evolutionary Computation, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the First IEEE Conference on*. Ieee, 1994, pp. 82–87.

[44] E. Zitzler and L. Thiele, "Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

[45] E. Zitzler, M. Laumanns, L. Thiele *et al.*, "SPEA2: Improving the Strength Pareto Evolutionary Algorithm," in *EUROGEN*, no. 103, 2001, pp. 1–21.

[46] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[47] K. Deb, *Multi-objective optimization using evolutionary algorithms*. Wiley, 2001, vol. 16.

[48] E. Zitzler and S. Künzli, "Indicator-Based Selection in Multiobjective Search," in *Parallel Problem Solving from Nature*. Springer, 2004, pp. 832–842.

[49] M. Emmerich, N. Beume, and B. Naujoks, "An emo algorithm using the hypervolume measure as selection criterion," in *Evolutionary Multi-Criterion Optimization*. Springer, 2005, pp. 62–76.

[50] N. Beume, B. Naujoks, and M. Emmerich, "Sms-emoa: Multiobjective selection based on dominated hypervolume," *European Journal of Operational Research*, vol. 181, no. 3, pp. 1653–1669, 2007.

[51] J. Bader and E. Zitzler, "HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization," Computer Engineering and Networks Laboratory (TIK), ETH Zurich, Tech. Rep., 2008.

[52] J. Bader and E. Zitzler, "HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 19, no. 1, pp. 45–76, 2011.

[53] M. Fourman, "Compaction of symbolic layout using genetic algorithms," in *Proceedings of the 1st International Conference on Genetic Algorithms*. L. Erlbaum Associates Inc., 1985, pp. 141–153.

[54] B. Bernheim, "Rationalizable Strategic Behavior," *Econometrica*, pp. 1007–1028, 1984.

[55] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, and V. da Fonseca, "Performance Assessment of Multiobjective Optimizers: An Analysis and Review," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 117–132, 2003.

[56] T. Wagner, N. Beume, and B. Naujoks, "Pareto-, Aggregation-, and Indicator-Based Methods in Many-Objective Optimization," in *Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 742–756.

[57] P. Halmos, *Measure Theory*. Springer, 1974, vol. 18.

[58] H. Beyer, "An Alternative Explanation for the Manner in Which Genetic Algorithms Operate," *BioSystems*, vol. 41, no. 1, pp. 1–15, 1997.

[59] H. Schwefel, "Numerical Optimization of Computer Models," 1981.

[60] T. Bäck, F. Hoffmeister, and H.-P. Schwefel, "A Survey of Evolution Strategies," in *International Conference on Genetic Algorithms*. Morgan Kaufmann, 1991, pp. 2–9.

[61] H. Beyer and H. Schwefel, "Evolution Strategies - A Comprehensive Introduction," *Natural computing*, vol. 1, no. 1, pp. 3–52, 2002.

[62] F. Kursawe, "A Variant of Evolution Strategies for Vector Optimization," *Parallel Problem Solving from Nature*, pp. 193–197, 1991.

[63] T. Binh and U. Korn, "MOBES: A Multiobjective Evolution Strategy for Constrained Optimization Problems," in *Conference on Genetic Algorithms*, 1997, pp. 176–182.

[64] M. Laumanns, G. Rudolph, and H. Schwefel, "A Spatial Predator-Prey Approach to Multi-Objective Optimization: A preliminary Study," in *Parallel Problem Solving from Nature*. Springer, 1998, pp. 241–249.

[65] J. Knowles and D. Corne, "The Pareto Archived Evolution Strategy: A New Baseline Algorithm for Pareto Multiobjective Optimisation," in *IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 1999.

[66] Y. Jin, T. Okabe, and B. Sendho, "Adapting Weighted Aggregation for Multiobjective Evolution Strategies," in *Evolutionary Multi-Criterion Optimization*. Springer, 2001, pp. 96–110.

[67] C. Igel, N. Hansen, and S. Roth, "Covariance Matrix Adaptation for Multi-Objective Optimization," *Evolutionary Computation*, vol. 15, no. 1, pp. 1–28, 2007.

[68] R. Mohler, C. Bruni, and A. Gandolfi, "A systems approach to immunology," *Proceedings of the IEEE*, vol. 68, no. 8, pp. 964–990, 1980.

[69] D. Dasgupta and S. Forrest, "Artificial Immune Systems in Industrial Applications," in *International Conference on Intelligent Processing and Manufacturing of Materials*, vol. 1. IEEE, 1999, pp. 257–267.

[70] P. Krammer, "Cd95's deadly mission in the immune system," *NATURE-LONDON-*, pp. 789–795, 2000.

[71] L. Parijs and A. Abbas, "Homeostasis and self-tolerance in the immune system: turning lymphocytes off," *Science*, vol. 280, no. 5361, p. 243, 1998.

[72] P. Parham and C. Janeway, *The immune system*. Garland Science New York, 2005.

[73] D. Dasgupta and N. Attoh-Okine, "Immunity-Based Systems: A Survey," in *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 1. IEEE, 1997, pp. 369–374.

[74] C. Coello and N. Cortés, "Solving Multiobjective Optimization Problems Using An Artificial Immune System," *Genetic Programming and Evolvable Machines*, vol. 6, no. 2, pp. 163–190, 2005.

[75] A. Gasper and P. Collard, "From GAs to Artificial Immune Systems: Improving Adaptation in Time Dependent Optimization," in *IEEE Congress on Evolutionary Computation*, vol. 3. IEEE, 1999.

[76] J. Yoo and P. Hajela, "Immune network simulations in multicriterion design," *Structural and Multidisciplinary Optimization*, vol. 18, no. 2, pp. 85–94, 1999.

[77] C. Coello and N. Cortés, "An Approach to Solve Multiobjective Optimization Problems Based on an Artificial Immune System," in *First International Conference on Artificial Immune Systems (ICARIS2002)*, 2002, pp. 212–221.

[78] Z. Zhang, "Immune optimization algorithm for constrained nonlinear multiobjective optimization problems," *Applied Soft Computing*, vol. 7, no. 3, pp. 840–857, 2007.

[79] M. Gong, L. Jiao, H. Du, and L. Bo, "Multiobjective Immune Algorithm with Nondominated Neighbor-Based Selection," *Evolutionary Computation*, vol. 16, no. 2, pp. 225–255, 2008.

[80] C. A. C. Coello and G. T. Pulido, "A Micro-Genetic Algorithm for Multiobjective Optimization," in *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, ser. EMO '01. London, UK: Springer-Verlag, 2001, pp. 126–140.

[81] J. Choe and D. Perlman, "Social Conflict and Cooperation Among Founding Queens in Ants (Hymenoptera: Formicidae)," *The evolution of social behavior in insects and arachnids*, p. 392, 1997.

[82] J. Deneubourg, S. Aron, S. Goss, and J. Pasteels, "The self-organizing exploratory pattern of the argentine ant," *Journal of Insect Behavior*, vol. 3, no. 2, pp. 159–168, 1990.

[83] M. Dorigo, M. Birattari, and T. Stutzle, "Ant Colony Optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[84] M. López-Ibáñez and T. Stützle, "An Analysis of Algorithmic Components for Multiobjective Ant Colony Optimization: A Case Study on the Biobjective TSP," *Artifical Evolution*, pp. 134–145, 2010.

[85] S. Iredi, D. Merkle, and M. Middendorf, "Bi-criterion optimization with multi colony ant algorithms," in *Evolutionary Multi-Criterion Optimization*. Springer, 2001, pp. 359–372.

[86] K. Doerner, W. Gutjahr, R. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection," *Annals of Operations Research*, vol. 131, no. 1, pp. 79–99, 2004.

[87] L. Gambardella, É. Taillard, and G. Agazzi, "Macs-vrptw: A multiple colony system for vehicle routing problems with time windows," in *New ideas in optimization*. Citeseer, 1999.

[88] J. Lampinen and I. Zelinka, "Mixed integer-discrete-continuous optimization by differential evolution," in *Proceedings of the 5th International Conference on Soft Computing*. Citeseer, 1999, pp. 71–76.

[89] G. Pampara, A. Engelbrecht, and N. Franken, "Binary Differential Evolution," in *IEEE Congress on Evolutionary Computation*. IEEE, 2006, pp. 1873–1879.

[90] R. Storn, "On the Usage of Differential Evolution for Function Optimization," in *Conference of the North American Fuzzy Information Processing Society*. IEEE, 1996, pp. 519–523.

[91] C. Chang, D. Xu, and H. Quek, "Pareto-optimal set based multiobjective tuning of fuzzy automatic train operation for mass transit system," in *Electric Power Applications, IEE Proceedings-*, vol. 146, no. 5. IET, 1999, pp. 577–583.

[92] H. Abbass, R. Sarker, and C. Newton, "PDE: A Pareto-Frontier Differential Evolution Approach for Multi-Objective Optimization Problems," in *Congress on Evolutionary Computation*, vol. 2. IEEE, 2001, pp. 971–978.

[93] J. Lampinen, "Des selection rule for multiobjective optimization," *Lappeenranta University of Technology, Department of Information Technology, Tech. Rep*, pp. 03–04, 2001.

[94] N. Madavan, "Multiobjective Optimization Using a Pareto Differential Evolution Approach," in *IEEE Congress on Evolutionary Computation*, vol. 2. IEEE, 2002, pp. 1145–1150.

[95] E. Hughes, "Multiple Single Objective Pareto Sampling," in *Congress on Evolutionary Computation, 2003*, vol. 4. IEEE, 2003, pp. 2678–2684.

[96] E. Hughes, "MSOPS-II: A General-Purpose Many-Objective Optimiser," in *IEEE Congress on Evolutionary Computation*, sept. 2007, pp. 3944 –3951.

[97] K. Parsopoulos, D. Tasoulis, N. Pavlidis, V. Plagianakos, and M. Vrahatis, "Vector Evaluated Differential Evolution for Multiobjective Optimization," in *IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 2004, pp. 204–211.

[98] S. Kukkonen and J. Lampinen, "An extension of generalized differential evolution for multi-objective optimization with constraints," in *Parallel Problem Solving from Nature-PPSN VIII*. Springer, 2004, pp. 752–761.

[99] S. Kukkonen and J. Lampinen, "GDE3: The Third Evolution Step of Generalized Differential Evolution," in *IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 2005, pp. 443–450.

[100] I. Das and J. Dennis, "Normal-Boundary Intersection: An Alternate Method for Generating Pareto Optimal Points in Multicriteria Optimization Problems," DTIC Document, Tech. Rep., 1996.

[101] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *IEEE International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[102] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," in *IEEE International Conference on Evolutionary Computation*. IEEE, may 1998, pp. 69 –73.

[103] Y. del Valle, G. Venayagamoorthy, S. Mohagheghi, J. Hernandez, and R. Harley, "Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 171–195, 2008.

[104] P. Tripathi, S. Bandyopadhyay, and S. Pal, "Multi-Objective Particle Swarm Optimization with Time Variant Inertia and Acceleration Coefficients," *Information Sciences*, vol. 177, no. 22, pp. 5033–5049, 2007.

[105] K. Parsopoulos and M. Vrahatis, "Particle swarm optimization method in multiobjective problems," in *Proceedings of the 2002 ACM symposium on Applied computing*. ACM, 2002, pp. 603–607.

[106] C. Coello Coello and M. Lechuga, "MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization," in *IEEE Congress on Evolutionary Computation*, vol. 2. IEEE, 2002, pp. 1051–1056.

[107] J. Fieldsend, E. Uk, and S. Singh, "A multi-objective algorithm based upon particle swarm optimisation, an efficient data structure and turbulence." 2002.

[108] X. Hu and R. Eberhart, "Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization," in *IEEE Congress on Evolutionary Computation*, vol. 2. IEEE, 2002, pp. 1677–1681.

[109] X. Hu, R. Eberhart, and Y. Shi, "Particle swarm with extended memory for multiobjective optimization," in *Swarm Intelligence Symposium, 2003. SIS'03. Proceedings of the 2003 IEEE*. IEEE, 2003, pp. 193–197.

[110] C. Coello, G. Pulido, and M. Lechuga, "Handling Multiple Objectives with Particle Swarm Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 8, no. 3, pp. 256–279, 2004.

[111] M. Reyes-Sierra and C. Coello, "Multi-objective particle swarm optimizers: A survey of the state-of-the-art," *International Journal of Computational Intelligence Research*, vol. 2, no. 3, pp. 287–308, 2006.

[112] Y. Jin, M. Olhofer, and B. Sendhoff, "Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does it Work and How?" 2001.

[113] M. Hauschild and M. Pelikan, "A survey of estimation of distribution algorithms," 2011.

[114] M. Pelikan, D. Goldberg, and F. Lobo, "A Survey of Optimization by Building and Using Probabilistic Models," *Computational Optimization and Applications*, vol. 21, no. 1, pp. 5–20, 2002.

[115] M. Laumanns and J. Ocenasek, "Bayesian optimization algorithms for multi-objective optimization," *Parallel Problem Solving from NaturePPSN VII*, pp. 298–307, 2002.

[116] K. Sastry, D. Goldberg, and M. Pelikan, "Limits of Scalability of Multiobjective Estimation of Distribution Algorithms," in *IEEE Congress on Evolutionary Computation*, vol. 3. IEEE, 2005, pp. 2217–2224.

[117] R. Shah and P. Reed, "Comparative Analysis of Multiobjective Evolutionary Algorithms for Random and Correlated Instances of Multiobjective D-Dimensional Knapsack Problems," *European Journal of Operational Research*, vol. 211, no. 3, pp. 466–479, 2011.

[118] K. Sastry, "Efficient Cluster Optimization Using Extended Compact Genetic Algorithm with Seeded Population," in *Conference on Genetic and Evolutionary Computation*, 2001.

[119] H. Muhlenbein and T. Mahnig, "Evolutionary Optimization and the Estimation of Search Distributions with Applications to Graph Bipartitioning," *International journal of approximate reasoning*, vol. 31, no. 3, pp. 157–192, 2002.

[120] D. Thierens and P. Bosman, "Multi-Objective Mixture-Based Iterated Density Estimation Evolutionary Algorithms," pp. 663–670, 2001.

[121] P. Bosman and D. Thierens, "Expanding from Discrete to Continuous Estimation of Distribution Algorithms: The IDEA," in *Parallel Problem Solving from Nature*. Springer, 2000, pp. 767–776.

[122] M. Pelikan, D. Goldberg, and E. Cantú-Paz, "Bayesian optimization algorithm, population sizing, and time to convergence," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2000, pp. 275–282.

[123] M. Costa and E. Minisci, "Moped: A multi-objective parzen-based estimation of distribution algorithm for continuous problems," in *Evolutionary Multi-Criterion Optimization*. Springer, 2003, pp. 71–71.

[124] T. Okabe, Y. Jin, B. Sendoff, and M. Olhofer, "Voronoi-Based Estimation of Distribution Algorithm for Multi-Objective Optimization," in *IEEE Congress on Evolutionary Computation*, vol. 2. IEEE, 2004, pp. 1594–1601.

[125] Q. Zhang, A. Zhou, and Y. Jin, "RM-MEDA: A Regularity Model-Based Multiobjective Estimation of Distribution Algorithm," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 1, pp. 41–63, 2008.

[126] L. Emmendorfer and A. Pozo, "Effective Linkage Learning Using Low-Order Statistics and Clustering," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 6, pp. 1233–1246, 2009.

[127] E. Zitzler, K. Deb, and L. Thiele, "Comparison of Multiobjective Evolutionary Algorithms: Empirical Results," *Evolutionary Computation*, vol. 8, no. 2, pp. 173–195, 2000.

[128] K. Deb, A. Sinha, and S. Kukkonen, "Multi-Objective Test Problems, Linkages, and Evolutionary Methodologies," in *Conference on Genetic and Evolutionary Computation*. ACM, 2006, pp. 1141–1148.

[129] P. Bosman and D. Thierens, "The Naive MIDEA: A Baseline Multi-Objective EA," in *Evolutionary Multi-Criterion Optimization*. Springer, 2005, pp. 428–442.

[130] P. Castro and F. Von Zuben, "GAIS: A Gaussian Artificial Immune System for Continuous Optimization," *Artificial Immune Systems*, pp. 171–184, 2010.

[131] P. Castro and F. Von Zuben, "A Gaussian Artificial Immune System for Multi-Objective optimization in continuous domains," in *Conference on Hybrid Intelligent Systems*. IEEE, 2010, pp. 159–164.

[132] G. Bilchev and I. Parmee, "The ant colony metaphor for searching continuous design spaces," *Evolutionary Computing*, pp. 25–39, 1995.

[133] K. Socha and M. Dorigo, "Ant colony optimization for continuous domains," *European Journal of Operational Research*, vol. 185, no. 3, pp. 1155–1173, 2008.

[134] S. Baluja and R. Caruana, "Removing the Genetics from the Standard Genetic Algorithm," in *Machine Learning International Workshop then Conference*, 1995, pp. 38–46.

[135] A. Colorni, M. Dorigo, V. Maniezzo *et al.*, "Distributed optimization by ant colonies," in *Proceedings of the first European conference on artificial life*, vol. 142, 1991, pp. 134–142.

[136] Q. Pan, M. Tasgetiren, and Y. Liang, "A Discrete Differential Evolution Algorithm for the Permutation Flowshop Scheduling Problem," in *Conference on Genetic and Evolutionary Computation*. ACM, 2007, pp. 126–133.

[137] J. Cai and G. Thierauf, "Evolution Strategies for Solving Discrete Optimization Problems," *Advances in Engineering Software*, vol. 25, no. 2-3, pp. 177–183, 1996.

[138] J. Kennedy and R. Eberhart, "A Discrete Binary Version of the Particle Swarm Algorithm," in *IEEE International Conference on Systems, Man, and Cybernetics*, vol. 5. IEEE, 1997, pp. 4104–4108.

[139] K. Socha, "ACO for Continuous and Mixed-Variable Optimization," *Ant Colony Optimization and Swarm Intelligence*, pp. 53–61, 2004.

[140] J. Ocenasek and J. Schwarz, "Estimation Distribution Algorithm for Mixed Continuous-Discrete Optimization Problems," *Intelligent technologies: theory and applications: new trends in intelligent technologies*, vol. 76, p. 227, 2002.

[141] T. Bäck, M. Schütz, T. Ack, and M. Utz, "Evolution Strategies for Mixed-Integer Optimization of Optical Multilayer Systems," *Evolutionary Programming*, p. 33, 1995.

[142] M. Emmerich, M. Grötzner, B. Groß, and M. Schütz, "Mixed-Integer Evolution Strategy for Chemical Plant Optimization with Simulators," *Evolutionary Design and Manufacture-Selected papers from ACDM*, pp. 55–67, 2000.

[143] Z. Gaing, "Constrained Optimal Power Flow by Mixed-Integer Particle Swarm Optimization," in *Power Engineering Society General Meeting*. IEEE, 2005, pp. 243–250.

[144] J. Bell and P. McMullen, "Ant Colony Optimization Techniques for the Vehicle Routing Problem," *Advanced Engineering Informatics*, vol. 18, no. 1, pp. 41–48, 2004.

[145] SCOPUS, "http://www.scopus.com/," dec 2012. [Online]. Available: http://www.scopus.com/

[146] J. Puchinger and G. R. Raidl, "Combining Metaheuristics and Exact Algorithms in Combinatorial Optimization: A Survey and Classification," in *Lecture Notes in Computer Science*, vol. 3562, 2005, pp. 41–53.

[147] A. Hertz and M. Widmer, "Guidelines for the use of Meta-Heuristics in Combinatorial Optimization," *European Journal of Operational Research*, vol. 151, no. 2, pp. 247–252, 2003.

[148] J. He and X. Yao, "Drift Analysis and Average Time Complexity of Evolutionary Algorithms," *Artificial Intelligence*, vol. 127, no. 1, pp. 57–85, 2001.

[149] T. Chen, K. Tang, G. Chen, and X. Yao, "Analysis of Computational Time of Simple Estimation of Distribution Algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 1–22, 2010.

[150] M. Pelikan, D. Goldberg, and S. Tsutsui, "Hierarchical Bayesian Optimization Algorithm: Toward A New Generation of Evolutionary Algorithms," in *SICE Annual Conference*, vol. 3. IEEE, 2005, pp. 2738–2743.

[151] P. De Boer, D. Kroese, S. Mannor, and R. Rubinstein, "A Tutorial on the Cross-Entropy Method," *Annals of Operations Research*, vol. 134, no. 1, pp. 19–67, 2005.

[152] M. Pelikan and K. Sastry, *Scalable Optimization via Probabilistic Modeling: From Algorithms to Applications*. Springer Verlag, 2006, vol. 33.

[153] R. Purshouse and P. Fleming, "Evolutionary Many-Objective Optimisation: An Exploratory Analysis," in *IEEE Congress on Evolutionary Computation*, vol. 3. IEEE, 2003, pp. 2066–2073.

[154] E. J. Hughes, "Evolutionary Many-Objective Optimisation: Many Once or One Many?" in *IEEE Congress on Evolutionary Computation*, vol. 1, 2005, pp. 222–227.

[155] Y. Chen, X. Zou, and W. Xie, "Convergence of Multi-Objective Evolutionary Algorithms to a Uniformly Distributed Representation of the Pareto Front," *Information Sciences*, vol. 181, no. 16, pp. 3336–3355, 2011.

[156] M. Laumanns, E. Zitzler, and L. Thiele, "A Unified Model for Multi-Objective Evolutionary Algorithms with Elitism," in *IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 2000, pp. 46–53.

[157] B. Liu, L. Wang, Y. Liu, and S. Wang, "A unified framework for population-based metaheuristics," *Annals of Operations Research*, pp. 1–32, 2011.

[158] A. Jaszkiewicz, "On the Performance of Multiple-Objective Genetic Local Search on the 0/1 Knapsack Problem - A comparative Experiment," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 4, pp. 402–412, 2002.

[159] Y. yan Tan, Y. chang Jiao, H. Li, and X. kuan Wang, "MOEA/D + uniform design: A new version of MOEA/D for optimization problems with many objectives," *Computers & Operations Research*, no. 0, 2012.

[160] S. Jiang, Z. Cai, J. Zhang, and Y.-S. Ong, "Multiobjective Optimization by Decomposition with Pareto-Adaptive Weight Vectors," in *International Conference on Natural Computation*, vol. 3, july 2011, pp. 1260 –1264.

[161] S. Jiang, J. Zhang, and Y. Ong, "Asymmetric Pareto-adaptive Scheme for Multiobjective Optimization," in *Advances in Artificial Intelligence*, ser. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2011, vol. 7106, pp. 351–360.

[162] R. Steuer and E. Choo, "An interactive weighted tchebycheff procedure for multiple objective programming," *Mathematical Programming*, vol. 26, no. 3, pp. 326–344, 1983.

[163] I. Kaliszewski, "A modified weighted tchebycheff metric for multiple objective programming," *Computers & operations research*, vol. 14, no. 4, pp. 315–323, 1987.

[164] M. Grant and S. Boyd, "CVX: Matlab Software for Disciplined Convex Programming," 2008. [Online]. Available: http://cvxr.com/cvx/

[165] D. Hardin and E. Saff, "Discretizing Manifolds via Minimum Energy Points," *Notices of the AMS*, vol. 51, no. 10, pp. 1186–1194, 2004.

[166] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, "Scalable Multi-Objective Optimization Test Problems," in *Congress on Evolutionary Computation*, vol. 1, may 2002, pp. 825 –830.

[167] S. Huband, P. Hingston, L. Barone, and L. While, "A Review of Multiobjective Test Problems and A Scalable Test Problem Toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[168] M. Pelikan, "Bayesian Optimization Algorithm," *Hierarchical Bayesian Optimization Algorithm*, pp. 31–48, 2005.

[169] C. Echegoyen, Q. Zhang, A. Mendiburu, R. Santana, and J. Lozano, "On the Limits of Effectiveness in Estimation of Distribution Algorithms," in *IEEE Congress on Evolutionary Computation*. IEEE, 2011, pp. 1573–1580.

[170] R. Rubinstein, "The Cross-Entropy Method for Combinatorial and Continuous Optimization," *Methodology and Computing in Applied Probability*, vol. 1, no. 2, pp. 127–190, 1999.

[171] D. Wolpert, "Information Theory - The Bridge Connecting Bounded Rational Game Theory and Statistical Physics," *Complex Engineered Systems*, pp. 262–290, 2006.

[172] R. Rubinstein, "A Stochastic Minimum Cross-Entropy Method for Combinatorial Optimization and Rare-event Estimation," *Methodology and Computing in Applied Probability*, vol. 7, no. 1, pp. 5–50, 2005.

[173] Z. Botev, D. Kroese, and T. Taimre, "Generalized Cross-Entropy Methods with Applications to Rare-Event Simulation and Optimization," *Simulation*, vol. 83, no. 11, p. 785, 2007.

[174] P. Glynn and D. Iglehart, "Importance sampling for stochastic simulations," *Management Science*, vol. 35, no. 11, pp. 1367–1392, 1989.

[175] C. N. Morris, "Natural Exponential Families with Quadratic Variance Functions," *The Annals of Statistics*, vol. 10, pp. 65–80, 1982.

[176] H. Ishibuchi, Y. Sakane, N. Tsukamoto, and Y. Nojima, "Effects of Using Two Neighborhood Structures on the Performance of Cellular Evolutionary Algorithms for Many-Objective Optimization," in *IEEE Congress on Evolutionary Computation*, may 2009, pp. 2508 –2515.

[177] H. Li and Q. Zhang, "Multiobjective Optimization Problems with Complicated Pareto Sets, MOEA/D and NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284–302, 2009.

[178] D. Wolpert and W. Macready, "No Free Lunch Theorems for Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 67–82, 1997.

[179] D. Van Veldhuizen, "Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations," in *Evolutionary Computation*, 1999.

[180] E. Saff and A. Kuijlaars, "Distributing Many Points on a Sphere," *The Mathematical Intelligencer*, vol. 19, no. 1, pp. 5–11, 1997.

[181] S. B. Damelin and P. J. Grabner, "Energy Functionals, Numerical Integration and Asymptotic Equidistribution on the Sphere," *Journal of Complexity*, vol. 19, no. 3, pp. 231 – 246, 2003.

[182] R. Purshouse and P. Fleming, "On the Evolutionary Optimization of Many Conflicting Objectives," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 770 –784, dec. 2007.

[183] A. Jaszkiewicz, "Do Multiple-Objective Metaheuristics Deliver on Their Promises? A Computational Experiment on the Set-Covering Problem," *IEEE Transactions on Evolutionary Computation*, vol. 7, no. 2, pp. 133–143, 2003.

[184] C. Fonseca and P. Fleming, "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms. I. A Unified Formulation," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 28, no. 1, pp. 26–37, 1998.

[185] C. Fonseca and P. Fleming, "Multiobjective Optimization and Multiple Constraint Handling with Evolutionary Algorithms. II. Application Example," *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, vol. 28, no. 1, pp. 38–47, 1998.

[186] P. Fleming and R. Purshouse, "Evolutionary Algorithms in Control Systems Engineering: A Survey," *Control Engineering Practice*, vol. 10, no. 11, pp. 1223–1241, 2002.

[187] J. Shoaf and J. Foster, "The Efficient Set GA for Stock Portfolios," in *IEEE International Conference on Evolutionary Computation*, may 1998, pp. 354 –359.

[188] R. Armananzas and J. Lozano, "A Multiobjective Approach to the Portfolio Optimization Problem," in *IEEE Congress on Evolutionary Computation*, vol. 2, sept. 2005, pp. 1388 – 1395 Vol. 2.

[189] R. Subbu, P. Bonissone, N. Eklund, S. Bollapragada, and K. Chalermkraivuth, "Multiobjective Financial Portfolio Design: A Hybrid Evolutionary Approach," in *IEEE Congress on Evolutionary Computation*, vol. 2, sept. 2005, pp. 1722 – 1729 Vol. 2.

[190] M. Tapia and C. Coello, "Applications of Multi-Objective Evolutionary Algorithms in Economics and Finance: A Survey," in *IEEE Congress on Evolutionary Computation*, vol. 2007, 2007, pp. 532–539.

[191] Y.-S. Ong, P. Nair, and K. Lum, "Max-Min Surrogate-Assisted Evolutionary Algorithm for Robust Design," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 392 – 404, aug 2006.

[192] T. Goel, R. Vaidyanathan, R. Haftka, W. Shyy, N. Queipo, and K. Tucker, "Response Surface Approximation of Pareto Optimal Front in Multi-Objective Optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 4, pp. 879–893, 2007.

[193] D. R. Jones, M. Schonlau, and W. J. Welch, "Efficient Global Optimization of Expensive Black-Box Functions," *Journal of Global Optimization*, vol. 13, pp. 455–492, 1998. [Online]. Available: http://dx.doi.org/10.1023/A:1008306431147

[194] K. Liang, X. Yao, and C. Newton, "Evolutionary Search of Approximated N-Dimensional Landscapes," *International Journal of Knowledge Based Intelligent Engineering Systems*, vol. 4, no. 3, pp. 172–183, 2000.

[195] Y. Jin, M. Olhofer, and B. Sendhoff, "A Framework for Evolutionary Optimization with Approximate Fitness Functions," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 5, pp. 481 – 494, oct 2002.

[196] J. Knowles, "ParEGO: A Hybrid Algorithm with On-Line Landscape Approximation for Expensive Multiobjective Optimization Problems," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 1, pp. 50–66, 2006.

[197] Y. Jin and J. Branke, "Evolutionary Optimization in Uncertain Environments - A Survey," *IEEE Transactions on Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.

[198] M. Farina, "A Neural Network Based Generalized Response Surface Multiobjective Evolutionary Algorithm," in *IEEE Congress on Evolutionary Computation*, vol. 1. IEEE, 2002, pp. 956–961.

[199] M. Emmerich, A. Giotis, M. Özdemir, T. Bäck, and K. Giannakoglou, "Metamodel - Assisted Evolution Strategies," in *Parallel Problem Solving from Nature*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2002, vol. 2439, pp. 361–370.

[200] Y. Ong, P. Nair, and A. Keane, "Evolutionary Optimization of Computationally Expensive Problems via Surrogate Modeling," *AIAA journal*, vol. 41, no. 4, pp. 687–696, 2003.

[201] G. E. P. Box and K. B. Wilson, "On the Experimental Attainment of Optimum Conditions," *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 13, no. 1, pp. pp. 1–45, 1951. [Online]. Available: http://www.jstor.org/stable/2983966

[202] C. Poloni, A. Giurgevich, L. Onesti, and V. Pediroda, "Hybridization of A Multi-Objective Genetic Algorithm, A Neural Network and A Classical Optimizer for A Complex Design Problem in Fluid Dynamics," *Computer Methods in Applied Mechanics and Engineering*, vol. 186, no. 24, pp. 403 – 420, 2000.

[203] S. Adra, T. Dodd, I. Griffin, and P. Fleming, "Convergence Acceleration Operator for Multiobjective Optimization," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 4, pp. 825–847, 2009.

[204] Y. Jin, "A Comprehensive Survey of Fitness Approximation in Evolutionary Computation," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 9, pp. 3–12, 2005.

[205] Y. Ong, P. Nair, A. Keane, and K. Wong, "Surrogate-Assisted Evolutionary Optimization Frameworks for High-Fidelity Engineering Design Problems," *Knowledge Incorporation in Evolutionary Computation*, pp. 307–332, 2004.

[206] K. Deb and A. Srinivasan, "Innovization: Innovating Design Principles Through Optimization," in *Conference on Genetic and Evolutionary Computation*. ACM, 2006, pp. 1629–1636.

[207] S. Bandaru and K. Deb, "Automated Innovization for Simultaneous Discovery of Multiple Rules in Bi-Objective Problems," in *Evolutionary Multi-Criterion Optimization*. Springer, 2011, pp. 1–15.

[208] S. Bandaru and K. Deb, "Automating Discovery of Innovative Design Principles Through Optimization," *KanGAL Report*, no. 2010001, 2010.

[209] O. Schütze, S. Mostaghim, M. Dellnitz, and J. Teich, "Covering Pareto Sets by Multilevel Evolutionary Subdivision Techniques," in *Evolutionary Multi-Criterion Optimization*, ser. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2003, vol. 2632, pp. 10–10.

[210] R. Jin, W. Chen, and T. Simpson, "Comparative Studies of Metamodelling Techniques Under Multiple Modelling Criteria," *Structural and Multidisciplinary Optimization*, vol. 23, no. 1, pp. 1–13, 2001.

[211] H. Maier and G. Dandy, "Neural Networks for the Prediction and Forecasting of Water Resources Variables: A Review of Modelling Issues and Applications," *Environmental Modelling and software*, vol. 15, no. 1, pp. 101–124, 2000.

[212] A. Atiya, "Bankruptcy Prediction for Credit Risk Using Neural Networks: A Survey and New Results," *IEEE Transactions on Neural Networks*, vol. 12, no. 4, pp. 929–935, 2001.

[213] B. Wong and Y. Selvi, "Neural Network Applications in Finance: A Review and Analysis of Literature (1990-1996)," *Information & Management*, vol. 34, no. 3, pp. 129–139, 1998.

[214] K. Hornik, M. Stinchcombe, and H. White, "Multilayer Feedforward Networks are Universal Approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[215] C. Bishop, "Neural Networks for Pattern Recognition," 1995.

[216] D. Rumelhart, G. Hintont, and R. Williams, "Learning Representations by Back-Propagating Errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[217] S. A. Billings, H.-L. Wei, and M. A. Balikhin, "Generalized Multiscale Radial Basis Function Networks," *Neural Networks*, vol. 20, no. 10, pp. 1081 – 1094, 2007.

[218] D. Hadka and P. Reed, "Diagnostic Assessment of Search Controls and Failure Modes in Many - Objective Evolutionary Optimization," *Evolutionary Computation*, 2011.

[219] R. Purshouse, C. Jalbă, and P. Fleming, "Preference-Driven Co-Evolutionary Algorithms Show Promise for Many-Objective Optimisation," in *Evolutionary Multi-Criterion Optimization*. Springer, 2011, pp. 136–150.

[220] H. Markowitz, "Portfolio Selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, mar. 1952. [Online]. Available: http://www.jstor.org/stable/2975974

[221] J. Danielsson and C. G. D. Vries, "Value-at-Risk and Extreme Returns," *Annals of Economics and Statistics*, no. 60, pp. 239–270, dec. 2000.

[222] K. Kuester, S. Mittnik, and M. S. Paolella, "Value-at-Risk Prediction: A Comparison of Alternative Strategies," *Journal of Financial Econometrics*, vol. 4, no. 1, pp. 53–89, 2006.

[223] R. Rockafellar, *Convex Analysis*. Princeton University Press, 1970, vol. 28.

[224] "Yahoo! Finance," http://uk.finance.yahoo.com/, aug. 2012.

[225] D. Cohn, L. Atlas, and R. Ladner, "Improving Generalization with Active Learning," *Machine Learning*, vol. 15, pp. 201–221, 1994.

[226] A. Katanforoush and M. Shahshahani, "Distributing Points on the Sphere, I," *Experimental Mathematics*, vol. 12, no. 2, pp. 199–210, 2003.

[227] E. Fama, "Efficient Capital Markets: A Review of Theory and Empirical Work," *The Journal of Finance*, vol. 25, no. 2, pp. 383–417, 1970.

[228] R. Purshouse and P. Fleming, "Why use Elitism and Sharing in a Multi-Objective Genetic Algorithm," in *Proceedings of the Genetic and Evolutionary Computation Conference*, 2002, pp. 520–527.

[229] Z. Michalewicz, "Quo vadis, evolutionary computation?" *Advances in Computational Intelligence*, pp. 98–121, 2012.

[230] I. Kim and O. De Weck, "Adaptive Weighted Sum Method for Multiobjective Optimization: A New Method for Pareto Front Generation," *Structural and Multidisciplinary Optimization*, vol. 31, no. 2, pp. 105–116, 2006.

[231] F. Gu, H. Liu, and K. Tan, "A Multiobjective Evolutionary Algorithm Using Dynamic Weight Method," *International Journal of innovative Computing, Information and Control*, vol. 8, no. 5B, pp. 3677–3688, may 2012.

[232] I. Giagkiozis and P. Fleming, "Methods for many-objective optimization: An analysis," Research Report No. 1030, November 2012.

[233] H. Ishibuchi, N. Akedo, H. Ohyanagi, and Y. Nojima, "Behavior of EMO Algorithms on Many-Objective Optimization Problems with Correlated Objectives," in *IEEE Congress on Evolutionary Computation*, june 2011, pp. 1465 –1472.

[234] W. Rudin, *Principles of Mathematical Analysis*. McGraw-Hill New York, 1976, vol. 3.

[235] A. Kolmogorov and S. Fomin, *Introductory Real Analysis*. Dover Publications, 1975.

[236] N. Vereshchagin and A. Shen, *Basic Set Theory*. American Mathematical Society, 2002, vol. 17.

[237] G. Strang, *Introduction to Linear Algebra*. Wellesley Cambridge Press, 2003.

[238] G. Marsaglia, "Choosing a Point from the Surface of a Sphere," *The Annals of Mathematical Statistics*, vol. 43, pp. 645–646, 1972.