Multi-guided Particle Swarm Optimization: A Multi-objective Particle Swarm Optimizer

by

Christiaan Scheepers

Submitted in partial fulfilment of the requirements for the degree
Doctor of Philosophy (Computer Science)
in the Faculty of Engineering, Built Environment and Information Technology
University of Pretoria, Pretoria

October 2017

Publication data:

Christiaan Scheepers. Multi-guided Particle Swarm Optimization: A Multi-objective Particle Swarm Optimizer. Doctoral dissertation, University of Pretoria, Department of Computer Science, Pretoria, South Africa, October 2017.

Electronic, hyperlinked versions of this dissertation are available online, as Adobe PDF files, at:

http://cirg.cs.up.ac.za/

http://upetd.up.ac.za/UPeTD.htm

Multi-guided Particle Swarm Optimization: A Multi-objective Particle Swarm Optimizer

by

Christiaan Scheepers
E-mail: tiaan.scheepers@gmail.com

Abstract

An exploratory analysis in low-dimensional objective space of the vector evaluated particle swarm optimization (VEPSO) algorithm is presented. A novel visualization technique is presented and applied to perform the exploratory analysis. The exploratory analysis together with a quantitative analysis revealed that the VEPSO algorithm continues to explore without exploiting the well-performing areas of the search space. A detailed investigation into the influence that the choice of archive implementation has on the performance of the VEPSO algorithm is presented. Both the Pareto-optimal front (POF) solution diversity and convergence towards the true POF is considered during the investigation. Attainment surfaces are investigated for their suitability in efficiently comparing two multi-objective optimization (MOO) algorithms. A new measure to objectively compare algorithms in multi-dimensional objective space, based on attainment surfaces, is presented. This measure, referred to as the porcupine measure, adapts the attainment surface measure by using a statistical test along with weighted intersection lines. Loosely based on the VEPSO algorithm, the multi-guided particle swarm optimization (MGPSO) algorithm is presented and evaluated. The results indicate that the MGPSO algorithm overcomes the weaknesses of the VEPSO algorithm and also outperforms a number of *state of the art* MOO algorithms on at least two benchmark test sets.

Keywords: Multi-guided particle swarm optimizer, vector evaluated particle swarm optimizer, attainment surface, porcupine measure, particle swarm visualization.

Supervisor : Prof. A. P. Engelbrecht
Department : Department of Computer Science
Degree : Doctor of Philosophy

"The important thing is not to stop questioning. Curiosity has its own reason for existing. One cannot help but be in awe when he contemplates the mysteries of eternity, of life, of the marvelous structure of reality. It is enough if one tries merely to comprehend a little of this mystery everyday."

Albert Einstein (1879 - 1955)

"You can teach a student a lesson for a day; but if you can teach him to learn by creating curiosity, he will continue the learning process as long as he lives."

Clay P. Bedford (1903 - 1991)

Acknowledgements

- **My dad, mother, and brother** without whose patience and support this work would never have been possible.
- **Professor Andries Engelbrecht** for his invaluable guidance and insight.
- **Christopher Cleghorn** for his assistance with the mathematical analysis of the stability criteria.
- **My friends at the Computational Intelligence Research Group** for supporting me, always asking insightful questions, and challenging my results.
- **My friends** for all their support and always being interested in what I was doing.

Contents

List of Figures

List of Algorithms

List of Tables

Chapter 1

Introduction

> *"On September 12, 1997, the Mars Global Surveyor (MGS) entered an elliptical orbit around Mars, beginning a 20-year period of continuous robotic scientific exploration at the Red Planet. After 18 months of aerobraking to circularize its orbit around the Red Planet, MGS began its science and mapping mission, during which it returned more than 240,000 images over 4.8 Martian years, contributing a wealth of new information about Mars, its atmosphere and its satellites. Although MGS stopped transmitting in November 2006, by that time it had been joined at Mars by three other orbiters and on the surface by two rovers."*

One of the most fascinating problems today is the optimization of interplanetary trajectory in the Solar System, using gravitational fly-bys to reduce fuel consumption and mission duration. Ideally, both fuel consumption and mission duration must be minimized. However, higher fuel consumption typically leads to short mission durations, and vice-versa. The lack of analytical solutions and the presence of discontinuity makes this a complex multi-objective optimization (MOO) problem. An exhaustive search of the space of input variables is impractical even with today's high-performance supercomputers.

The problem mentioned above is but one example of a class of problems, known as MOO problems, which are well-suited for solving using computational intelligence techniques. The objective of this thesis is to develop a particle swarm-based MOO

algorithm capable of solving such a MOO problem.

Section 1.1 provides the motivation for this study. Section 1.2 lists the objectives of this study, followed by a list of the contributions made throughout this study in Section 1.3. Finally, an outline for the remainder of this thesis is given in Section 1.4.

1.1 Motivation

Whether you are an aerospace engineer trying to find the optimal trade-off between the fuel consumption and mission duration for your spacecraft, or an investor trying to find the optimal trade-off between the risk and profit for a portfolio, finding the optimal trade-off solutions is critical to achieving success.

The field of multiple-criteria decision-making (MCDM) deals with this class of problems, referred to as MOO problems, which have multiple often contradictory solutions. The solution to a MOO problem is not a single solution, but rather a set of optimal trade-off solutions. The objective of a MOO algorithm is to find this optimal trade-off set of solutions.

This study presents a new multi-objective particle swarm-based algorithm capable of solving complex MOO problems. Past studies have shown that particle swarm-based algorithms are capable of solving MOO problems. However, the algorithms often deviate from the simplicity of the basic particle swarm optimization (PSO) algorithm leading to less desirable complex implementations. The algorithm presented in this study focuses on retaining the simplicity of the basic PSO algorithm.

1.2 Objectives

The main objective of this study is to develop a multi-objective particle swarm-based algorithm to solve MOO problems. In working towards this goal, the following sub-objectives have been identified:

- to provide an overview of existing computational intelligence techniques that can be used to solve a MOO problem.

- to investigate the exploration behavior of the vector evaluated particle swarm optimization (VEPSO) algorithm. The investigation includes the development of a novel visualization technique to visualize the candidate solutions, within decision space, of an algorithm over multiple iterations.
- to investigate the effect that different archive implementations has on the performance of the VEPSO algorithm.
- to develop a quantitative measurement that can be used to compare multiple Pareto-optimal fronts (POFs) found by two or more algorithms in a statistically sound manner through the use of attainment surfaces.
- to propose a multi-swarm multi-objective PSO algorithm to solve MOO problems based on the findings of the above-mentioned studies.
- to investigate the performance of the above-mentioned algorithm. The performance investigation includes usage of the quantitative comparative measurement developed in this study.
- to compare the performance of the above-mentioned algorithm with *state of the art* MOO algorithms.

1.3 Contributions

The main contributions of this study are:

- The introduction of a new PSO-based multi-swarm multi-objective algorithm capable of solving MOO problems.
- Theoretical derivation of the order-1 and order-2 stable regions for the proposed algorithm's control parameter values.
- The finding that the proposed algorithm is capable of solving MOO problems. The results show that the proposed algorithm exhibits increased exploitation when compared with the VEPSO.

- The finding that the proposed algorithm is highly competitive when compared with both other particle swarm based algorithms as well as the state of the art MOO algorithms.
- The introduction of a new quantitative measure, referred to as the porcupine measure, that can be used to compare multiple POFs found by two or more algorithms in a statistically sound manner through the use of attainment surfaces.
- The finding that the quantitative attainment surface-based measurement proposed by Knowles and Corne [67] may lead to misleading results for non-concave POFs.
- The introduction of a novel visualization technique to visualize the search behaviour of the candidate solutions over a number of iterations. The proposed visualization technique revealed previously undiscovered behavior of the VEPSO algorithm.
- The introduction of a particle movement diversity measurement capable of measuring the diversity of a particle's position over a number of iterations. The measure gives an indication of how much a particle is moving around.
- The introduction of three candidate solution dispersion measurements. The measurements, when used together, indicate the dispersion of the candidate solutions for a population-based MOO algorithm.
- The finding that the cause of the VEPSO's weak performance is related to the lack of exploitation of the already found well-performing regions of the search space.
- The first detailed analysis of the influence that the archive management algorithm has on an algorithm's POF solution diversity and convergence towards the true POF.
- The introduction of the hypersurface contribution bounded archive deletion approach. The hypersurface contribution deletion approach selects solutions to remove from the archive based on the size of the hypersurface region that the solution contributes to the POF.

- The finding that the spacing measurement by Schott [96] and the distribution measurement by Goh and Tan [44] both suffer from a solution pairing problem leading to misleading diversity measurement results.
- The introduction of the crowding distribution measurement that addresses the pairing problem identified in the spacing and distribution measurements.

1.4 Dissertation Outline

- **Chapter 2** covers all the relevant computational intelligence techniques and background on which the subsequent chapters build. PSO, MOO, multi-objective PSO, and multi-objective performance measures are discussed.
- **Chapter 3** presents an investigation into the exploration behavior of the VEPSO algorithm. The investigation shows that the VEPSO algorithm continues to explore and does not exploit the well-performing regions of the POF. Modifying the velocity update equation by adding a weighted archive guide is shown to increase exploitation.
- **Chapter 4** presents an investigation into the influence that the choice of archive implementation has on the VEPSO algorithm's performance. The diversity of the POF and overall closeness to the true POF is investigated.
- **Chapter 5** introduces the porcupine measure, a quantitative measure that can be used to compare the POFs found by two or more algorithms. The porcupine measure makes use of attainment surfaces to statistically analyze an algorithm's POF, and gives researchers additional insight into a MOO algorithm's overall performance.
- **Chapter 6** introduces multi-guided particle swarm optimization (MGPSO). A thorough performance analysis comparing MGPSO's performance with other MOO algorithms is presented. The comparison included both particle swarm-based and the *state of the art* MOO algorithms. The results show that MGPSO performs on-par and even exceeds the performance of the other algorithms.

- **Chapter 7** presents a parameter sensitivity analysis for the MGPSO's parameters. The optimization procedure and optimized parameter values, as used throughout this study, is presented.
- **Chapter 8** provides a summary of all the findings and conclusions of the presented work. Ideas for future research, based on the presented work, is also given.
- **Appendix A** provides a list of the important acronyms used or newly defined in the course of this work, as well as their associated definition.
- **Appendix B** lists and defines the mathematical symbols used in this work, categorized according to the relevant chapter in which they appear.
- **Appendix C** provides a list of the parameter configurations for the additional algorithms used throughout this study.
- **Appendix D** lists all the publications derived from this work.

Note that this thesis makes extensive use of color figures and is best read in color.

Chapter 2

Background

"Wonder is the beginning of wisdom."

Socrates (470 - 399 BC)

Development of a new multi-objective particle swarm-based algorithm requires the application a number of computational intelligence techniques. This chapter provides background insight into the various computational intelligence paradigms that influenced the work presented in this study. PSO, MOO, multi-objective PSO, multi-objective measurements, and multi-objective test sets are covered.

PSO, which forms the basis for the work presented in this study, is covered in Section 2.1. The section takes an in-depth look at all the equations and parameters involved in driving the particle swarm's search behavior. Different neighborhood structures and a discussion on convergence and stability is also given.

An overview of MOO is given in Section 2.2, followed by an in-depth look at the VEPSO algorithm, a multi-swarm, multi-objective PSO algorithm, in Section 2.3.

Measurements to quantify the performance of a multi-objective algorithm are discussed in Section 2.4. The multi-objective toolkits and test problems used throughout this study are discussed in Section 2.5, followed by a summary of this chapter in Section 2.6.

2.1 Particle Swarm Optimization

Kennedy and Eberhart [62] introduced PSO in 1995. PSO is a stochastic population-based single objective optimization algorithm with its roots in the simulation of the social behavior of birds in a flock. PSO has been shown to be more successful in solving complex problems than traditional evolutionary computation (EC) algorithms [63]. PSO has also been applied to a variety of real-world problems, including optimizing equipment design parameters and fuel expenditure models for space missions [51, 120].

The basic PSO algorithm is presented in Section 2.1.1. Variations of the PSO velocity update equation is discussed in Section 2.1.2. PSO neighborhoods are presented in Section 2.1.3. Finally, PSO parameter sensitivity and convergence are discussed in Section 2.1.4.

2.1.1 Basic Particle Swarm Optimization

PSO's population, referred to as a swarm, consists of individuals referred to as particles. Each particle is represented by an n_x-dimensional vector, $\mathbf{x}_i$, representing a candidate solution to an optimization problem. The quality of the candidate solution is determined by evaluating an objective function, $f(\mathbf{x}_i)$, also known as a fitness function.

Particles move through the search space guided by an inertia velocity component, a cognitive component and a social component. The cognitive component is a randomly weighted difference between the current particle position and the previously found best performing particle position, referred to as the particle's personal best position. The social component is a randomly weighted difference between the current particle position and the neighborhood best position. A particle's neighborhood is determined by a neighborhood topology and allows for information regarding the well-performing regions of the search space to be shared among particles. The social component represents the socio-psychological tendency to emulate the success of neighboring particles.

Kennedy and Eberhart [62] formally defined the particle position and velocity update equations as follows:

$$\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1) \tag{2.1}$$

with

$$\mathbf{v}_i(t+1) = \mathbf{v}_i(t) + \boldsymbol{\varphi}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + \boldsymbol{\varphi}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t)) \tag{2.2}$$

where $\mathbf{x}_i(t)$ is particle i's position at iteration t, $\mathbf{v}_i(t)$ is particle i's velocity at iteration t, $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$ are vectors with components sampled uniformly random between 0 and 2, $\mathbf{y}_i(t)$ is particle i's personal best position at iteration t, and $\hat{\mathbf{y}}_i(t)$ is particle i's neighborhood best position at iteration t.

Figure 2.1 illustrates the different components of the velocity update equation along with the resulting velocity for the next iteration.

Analysis of the PSO velocity update equation, as shown in Equation (2.2), showed that extreme differences between the personal best position, $\mathbf{y}_i(t)$, or the neighborhood best position, $\hat{\mathbf{y}}_i(t)$, and the particle position, $\mathbf{x}_i(t)$, may lead to an "explosion" in the particle's velocity, $\mathbf{v}_i(t)$, where the velocity increases towards infinity [105]. An explosive velocity is undesirable as it may cause particles to leave the search space and thus negatively influence the overall search performance. In order to prevent an explosive velocity, the velocity, $\mathbf{v}_i$, is clamped into the range $[\mathbf{v}_{min}, \mathbf{v}_{max}]$ with $\mathbf{v}_{min} = -\mathbf{v}_{max}$ [14]. Well-performing values for $\mathbf{v}_{min}$ and $\mathbf{v}_{max}$ are problem dependent [97]. Shi and Eberhart [97] noted that when one lacks prior knowledge, setting $\mathbf{v}_{min}$ and $\mathbf{v}_{max}$ proportional to the search domain is a good starting point. The ideal proportion is, however, problem dependent.

Eberhart and Kennedy [30] studied the vectors $\boldsymbol{\varphi}_1$ and $\boldsymbol{\varphi}_2$ and defined them as

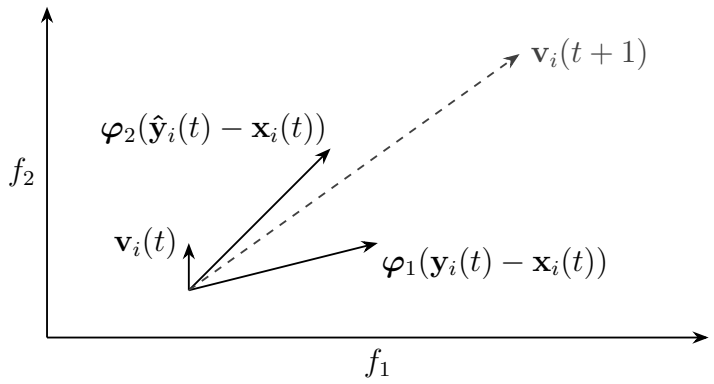


Figure 2.1: Particle velocity components.

$\boldsymbol{\varphi}_1 = c_1\mathbf{r}_1$ and $\boldsymbol{\varphi}_2 = c_2\mathbf{r}_2$, where $\mathbf{r}_1$ and $\mathbf{r}_2$ are vectors with components sampled uniformly random between 0 and 1. The constants c_1 and c_2 are positive real numbers referred to as the cognitive and social acceleration coefficients respectively. These constants regulate the maximum step size a particle can take per iteration. The cognitive acceleration coefficient, c_1, regulates the maximum step size in the direction of the particle's personal best position while the social acceleration coefficient, c_2, regulates the maximum step size in the direction of the neighborhood best position.

2.1.2 Velocity Variants

Shi and Eberhart [98] reasoned that the inertia velocity term, $\mathbf{v}_i(t)$, represents the PSO local search ability while the cognitive term, $c_1\mathbf{r}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t))$, and social term, $c_2\mathbf{r}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t))$, represent the global search ability. Shi and Eberhart further reasoned that different problems would benefit from different balances between the exploitation and exploration search ability. In order to regulate the trade-off between exploitation and exploration, Shi and Eberhart added an inertia weight, w, to the inertia velocity term. The inertia weight can be a positive constant or a positive linear or non-linear function of time. Early results indicated that the addition of the inertia weight improved PSO's performance [97]. The resulting PSO velocity update equation with the inertia weight incorporated is defined as follows:

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + c_2\mathbf{r}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t)) \tag{2.3}$$

Large inertia weights may also lead to an "explosion" in a particle's velocity due to a gradual buildup of velocity over time. An inertia weight less than 1 reduces the contribution made by the inertia velocity component and depending on the values of the acceleration coefficients may lead to a velocity that tends towards zero [105]. An inertia weight greater than 1 increases the contribution made by the inertia velocity component and will eventually lead to the maximum velocity being reached, leading to divergent particle behavior. The PSO algorithm with inertia weight incorporated is presented in algorithm listing 1. Note that algorithm listing 1 is for a synchronous PSO where all the personal best and neighborhood best positions are updated before updating the particle velocities and positions.

Clerc [12] proposed using a constriction factor to improve convergence of the PSO algorithm. The constriction factor, χ, is defined as a function of ϕ_1 and ϕ_2. The resulting PSO velocity update equation with the constriction factor incorporated is defined as follows:

$$\mathbf{v}_i(t+1) = \chi\big(\mathbf{v}_i(t) + \phi_1\mathbf{r}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + \phi_2\mathbf{r}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t))\big) \tag{2.4}$$

with

$$\chi = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} \tag{2.5}$$

where $\phi = \phi_1 + \phi_2, \phi > 4$.

Small values for the constriction factor, $\chi \approx 0$, encourage exploitation and faster convergence. Larger values for the constriction factor, $\chi \approx 1$, encourage exploration and slow convergence.

Clerc and Kennedy [14] found that clamping the particle velocity to $[\mathbf{v}_{min}, \mathbf{v}_{max}]$ is not necessary when adding the constriction factor to the velocity update equation.

Eberhart and Shi [31] found that the inertia weight in Equation (2.3) is functionally equivalent to the constriction factor in Equation (2.4) if the inertia weight, w, is set to χ, and $c_1 = \chi\phi_1$ and $c_2 = \chi\phi_2$ meet the conditions $\phi = \phi_1 + \phi_2, \phi > 4$. The PSO with the constriction factor can thus be considered a special case of the PSO with inertia weight.

2.1.3 Neighborhood Topologies

Kennedy and Eberhart's [62] proposed PSO made use of a fully connected, also known as the star, neighborhood. The star neighborhood connects all particles with all other particles, as illustrated in Figure 2.2(a). The entire swarm belongs to one neighborhood where every particle is attracted to the global best position. This fast sharing of information leads to faster convergence when compared to other neighborhoods. The faster convergence also increases the possibility of getting stuck in local optima [105]. PSOs that use the star neighborhood are referred to as global best, or *gbest*, PSOs.

Eberhart and Kennedy [30] also proposed a ring neighborhood where, in the case of the neighborhood size, $n_r = 2$, each particle only communicates with its immediate adjacent neighbors as illustrated in Figure 2.2(b). PSOs that use the ring neighborhood are referred to as local best, of *lbest*, PSOs. Particles attempt to imitate their best

Algorithm 1 Particle Swarm Optimization

Create and initialize a swarm, S, of n_s particles uniformly within a predefined hypercube of dimension n_x;
Let f be the objective function;
Let $\mathbf{y}_i$ represent the personal best position of particle i, initialized to $\mathbf{x}_i(0)$;
Let $\hat{\mathbf{y}}_i$ represent the neighborhood best position of particle i, initialized to $\mathbf{x}_i(0)$;
Initialize $\mathbf{v}_i(0)$ to $\mathbf{0}$;
Let $t = 0$;
repeat
  for each particle $i = 1, ..., n_s$ **do**
    if $f(\mathbf{x}_i) < f(\mathbf{y}_i)$ **then**
      $\mathbf{y}_i = \mathbf{x}_i(t)$;
    end if
    for particles $\hat{\imath}$ with particle i in their neighborhood **do**
      if $f(\mathbf{y}_i) < f(\hat{\mathbf{y}}_i)$ **then**
        $\hat{\mathbf{y}}_{\hat{\imath}} = \mathbf{y}_i$;
      end if
    end for
  end for
  for each particle $i = 1, ..., n_s$ **do**
    $\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + c_2\mathbf{r}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t))$;
    $\mathbf{x}_i(t+1) = \mathbf{x}_i(t) + \mathbf{v}_i(t+1)$;
  end for
  $t = t + 1$;
until stopping condition is true

performing neighbor moving towards the neighborhood best position. Note that, as shown in Figure 2.2(b), the neighborhoods for each particle overlap with that of its neighbor. This overlap facilitates the sharing of information among all the particles and helps to improve convergence on a single solution. Note that the *gbest* PSO can be seen as a special case of the *lbest* PSO where the neighborhood size is equal to the swarm size

minus one.

Kennedy [61] investigated the wheel neighborhood where all particles are effectively isolated from one another, and all the information exchange takes place through the focal particle. The wheel neighborhood is illustrated in figure 2.2(d). The focal particle compares fitnesses of all the particles and attempts to imitate the best particle in its neighborhood. If the focal particle's fitness improves, that improvement is propagated through to the rest of the particles. The focal particle serves as a buffer, slowing the speed at which information is propagated through the swarm.

Kennedy [60] proposed PSOs with no neighborhood, also referred to as *individual best* PSOs. In the individual best PSO, the velocity update equation does not have a social term. No information is exchanged between any of the particles. In effect, particles in the individual best PSO are hill-climbers. Particles may converge on multiple different solutions. Due to the lack of information sharing, the individual best PSO generally performs the worst of all the PSO neighborhoods.

Kennedy [60] also investigated small world inspired neighborhoods. Inspired by work done by Watts and Strogatz [112], small world neighborhoods randomize a small proportion of the connections, or shortcuts, between particles in order to achieve a high level of clustering with a greatly reduced average distance between particles. Watts and Strogatz called this high level of clustering with a reduced average distance the "small world" effect, in reference to the work done by Milgram [76], showing the hastened effect that small world shortcuts have on the spread of a disease through a population.

Kennedy and Mendes [64] introduced the Von Neumann neighborhood, a grid-like neighborhood that extends the *lbest* PSO neighborhood to two dimensions. Figure 2.2(c) illustrates a flattened section of the Von Neumann neighborhood. Particles share information with their direct neighbors (above, below, left, and right) based on the grid positions. Kennedy and Mendes found that PSOs using the Von Neumann neighborhood outperformed PSOs using other neighborhoods for a number of problems.

A variety of other neighborhood structures have been investigated. Kennedy and Mendes [64] also presented the idea of random neighborhoods where no fixed neighborhood structure exists beforehand. Mendes *et al.* [75] investigated the performance of three hand-tuned neighborhoods, namely *Square*, *Pyramid*, and *4Cluster*. Mendes *et al.*

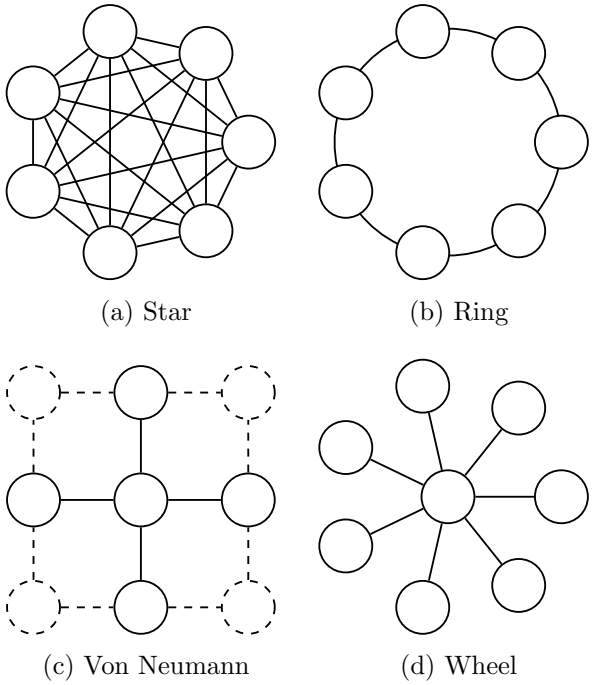


Figure 2.2: PSO neighborhood topologies.

found that the *Pyramid* neighborhood structure performed slightly better in some cases where the star neighborhood structure performed poorly.

While no one neighborhood is superior to all the other, it has been claimed that fully-connected neighborhoods tend to perform better for unimodal problems, while lesser-connected neighborhoods tend the perform better for multimodal problems [61, 64, 75]. An exhaustive comparison between the *gbest* and *lbest* PSOs, carried out by Engelbrecht [33], showed that neither neighborhood outright outperforms the other for any problem class. As with the other PSO parameters, the best performing neighborhood structure was shown to be problem dependent.

2.1.4 Parameter Sensitivity and Convergence

Various studies have found that PSO is sensitive to the control parameter choices, specifically the inertia weight, acceleration coefficients, and velocity clamping [4, 60, 97, 99]. Wrong initialization of these parameters may deter the swarm from converging.

Eberhart and Shi [31] found empirically that a constant inertia weight, w, of 0.7298 and acceleration coefficients, c_1 and c_2, of 1.49618 will lead to convergent behavior. It should be noted that Eberhart and Shi only studied a limited selection of problems and care should be taken when selecting the control parameters for any other problem. Further studies by Clerc [13] and Suganthan [101] showed that non-constant, or adaptive, inertia weight values could also lead to convergent behavior. However, a recent, more exhaustive, study by Harrison *et al.* [49] showed that adaptive strategies, in general, do not perform well. In many cases, adaptive strategies lead to divergent behaviour coupled with excessive invalid particles, and thus infeasible solutions, or lead to prohibitively low particle step sizes caused by rapid convergence.

While the empirical studies show that the choice of inertia weight is extremely important to ensure convergent behavior, Clerc and Kennedy [14] showed that for some problems velocity clamping was still required to prevent "explosive" particle velocities. Clerc and Kennedy did, however, not take into consideration that the inertia weight might be dependent on c_1 and c_2 as explained below.

Various theoretical studies have been conducted with the goal of better understanding the behavior of particle trajectories to derive heuristics to select parameter values that would lead to, or guarantee convergence to an equilibrium state [9, 7, 14, 41, 56, 59, 81, 82, 89, 90, 104, 105, 106, 113, 115].

Cleghorn and Engelbrecht [9] generalized the work done by Van den Bergh and Engelbrecht [106], Van den Bergh [105], and Trelea [104] and found that convergence is guaranteed for parameter values that satisfy the following relation:

$$c_1 + c_2 < 2(1 + w), \quad \text{for } -1 < w < 1,\ c_1 > 0 \text{ and } c_2 > 0 \tag{2.6}$$

Gazi [41] expanded the region derived by Kadirkamanathan *et al.* [59] and found that convergence is guaranteed for parameter values that satisfy the following relation:

$$\begin{aligned} c_1 + c_2 &< \frac{24(1 + w)}{7}, \quad \text{for } -1 < w \leq 0 \\ c_1 + c_2 &< \frac{24(1 - w)^2}{7(1 + w)}, \quad \text{for } 0 < w \leq 1 \end{aligned} \tag{2.7}$$

Poli [89], Poli and Broomhead [90] and Jiang *et al.* [56] independently found that con-

vergence is guaranteed for parameter values that satisfy the following relation:

$$c_1 + c_2 < \frac{24(1-w)^2}{7-5w}, \quad \text{for } -1 \leq w \leq 1 \tag{2.8}$$

Cleghorn [7] and Cleghorn and Engelbrecht [10] developed an objective function specifically designed for the convergence analysis of PSO variants. Using this purpose built objective function, Cleghorn and Engelbrecht [9] showed with the support of empirical evidence that the region defined by Equation (2.8) matched almost perfectly the convergence behavior of a non-simplified *gbest* PSO.

Take note that the experimental work presented in this study does not make use of velocity clamping. Instead, values for the inertia weight, w, and acceleration coefficients, c_1 and c_2 are carefully chosen to avoid "explosive" particle velocities.

2.2 Multi-objective Optimization

Many real-world optimization problems deal with optimizing problems that have more than one objective [20]. The objectives are typically in conflict with one another. Zotes and Peñas [120], for example, modeled the trade-off between fuel consumption, and thus mission cost, and mission duration as a multi-objective problem (MOP). The shorter the mission duration, the higher the fuel consumption and vice-versa. The solution for MOP is a set of optimal trade-off solutions. MOO algorithms can be used to solve MOPs.

Sections 2.2.1 and 2.2.2 formally introduce MOO and the definitions commonly found in the literature [19, 25, 54, 93, 94, 107, 116].

2.2.1 Multi-objective Problem

Let $\mathcal{S} \subseteq \mathbb{R}^{n_x}$ denote the n_x-dimensional search space (also referred to as the decision space), and $\mathcal{F} \subseteq \mathcal{S}$ the feasible space as determined by constraints. If there are no constraints, $\mathcal{F} = \mathcal{S}$. $\mathcal{O}$ denotes the n_m-dimensional objective space. A MOP translates a decision vector, $\mathbf{x} \in \mathcal{F}$, to an objective vector, $\mathbf{q}$, such that $\mathbf{q} \in \mathcal{O}$.

Definition 2.1. ***Decision vector*** *A decision vector, $\mathbf{x} = (x_1, x_2, \ldots, x_{n_x}) \in \mathcal{F}$, is an n_x-dimensional vector representing values chosen for an optimization problem.*

Definition 2.2. ***Objective vector*** *An objective vector,* $\mathbf{q} = (q_1, q_2, \ldots, q_{n_m}) \in \mathcal{O}$*, is an* n_m*-dimensional vector representing possible solutions for an optimization problem.*

Definition 2.3. ***Multi-objective problem*** *Without loss of generalization a MOP,* $f(\mathbf{x})$*, with* n_m *objectives is of the form:*

$$\min f(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_{n_m}(\mathbf{x})) \tag{2.9}$$

with $\mathbf{x} \in \mathcal{F}$*,* $f_m : \mathbb{R}^{n_x} \to \mathbb{R}$ *for all* $m \in [1, n_m]$*, and* $\mathcal{F} \subset \mathbb{R}^{n_x}$ *is the feasible space as determined by constraints.*

The duality principle [23] allows for the generalization of Definition (2.3) to include maximization objectives, $f_m^{'}(\mathbf{x})$, by rewriting them as minimization objectives, $f_m(\mathbf{x})$, where $f_m(\mathbf{x}) = -f_m^{'}(\mathbf{x})$.

Figure 2.3 illustrates a decision space, $\mathcal{F}$, its corresponding objective space, $\mathcal{O}$, along with a decision vector, $\mathbf{x}$, and the corresponding objective vector, $f(\mathbf{x})$.

2.2.2 Pareto-optimality

In order to ensure understanding and consistency when discussing Pareto-optimality, a number of frequently used definitions are listed:

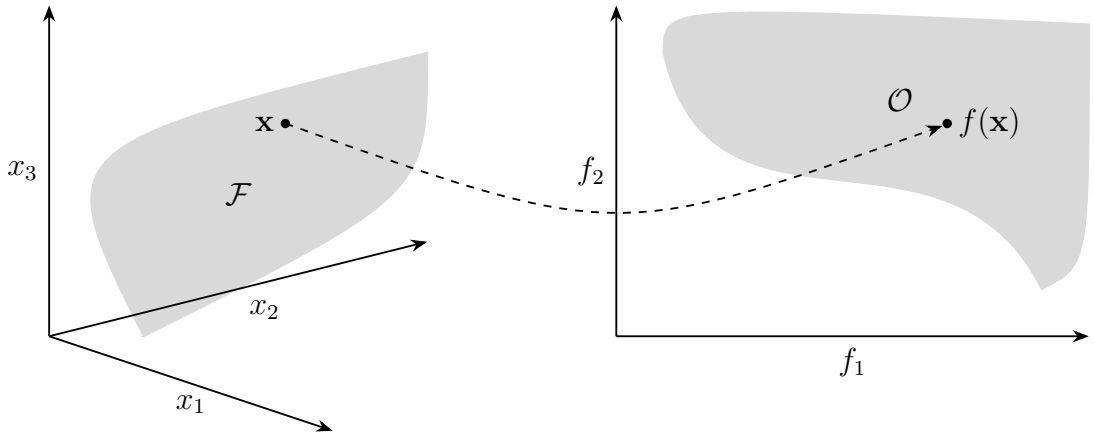


Figure 2.3: Decision and objective space.

Definition 2.4. ***Pareto-dominance*** *A decision vector* $\mathbf{x}_1 \in \mathcal{F}$ *dominates a decision vector* $\mathbf{x}_2 \in \mathcal{F}$ *(denoted by* $\mathbf{x}_1 \prec \mathbf{x}_2$*) if and only if* $f_m(\mathbf{x}_1) \leq f_m(\mathbf{x}_2)\ \forall\ m \in [1, n_m]$ *and* $\exists\ m \in [1, n_m]$ *such that* $f_m(\mathbf{x}_1) < f_m(\mathbf{x}_2)$.

Definition 2.5. ***Weak Pareto-dominance*** *A decision vector* $\mathbf{x}_1 \in \mathcal{F}$ *weakly dominates a decision vector* $\mathbf{x}_2 \in \mathcal{F}$ *(denoted by* $\mathbf{x}_1 \preceq \mathbf{x}_2$*) if and only if* $f_m(\mathbf{x}_1) \leq f_m(\mathbf{x}_2)\ \forall\ m \in [1, n_m]$.

Definition 2.6. ***ε-dominance*** *A decision vector* $\mathbf{x}_1 \in \mathcal{F}$ *ε-dominates a decision vector* $\mathbf{x}_2 \in \mathcal{F}$ *(denoted by* $\mathbf{x}_1 \prec_\varepsilon \mathbf{x}_2$*), for some* $\varepsilon > 0$*, if and only if* $\frac{f_m(\mathbf{x}_1)}{1+\varepsilon} \leq f_m(\mathbf{x}_2)\ \forall\ m \in [1, n_m]$ *and* $\exists\ m \in [1, n_m]$ *such that* $\frac{f_m(\mathbf{x}_1)}{1+\varepsilon} < f_m(\mathbf{x}_2)$.

Definition 2.7. ***Pareto-optimal*** *A decision vector* $\mathbf{x}_1 \in \mathcal{F}$ *is said to be Pareto-optimal if no decision vector* $\mathbf{x}_2 \in S$ *exists such that* $\mathbf{x}_2 \prec \mathbf{x}_1$.

Definition 2.8. ***Pareto-optimal set*** *A set* $\mathcal{P} \subseteq \mathcal{F} \in \mathbb{R}^{n_x}$ *is said to be the Pareto-optimal set (POS) if it contains only Pareto-optimal decision vectors.* $\mathcal{P}$ *is formally defined as* $\mathcal{P} = \{\mathbf{x}_1 \in \mathcal{F} \mid \nexists\ \mathbf{x}_2 \in \mathcal{F} : \mathbf{x}_2 \prec \mathbf{x}_1\}$.

Definition 2.9. ***Pareto-optimal front*** *A set* $Q \subseteq \mathcal{O} \in \mathbb{R}^{n_m}$ *is said to be the POF if it contains only objective vectors for Pareto-optimal decision vectors.* Q *is formally defined as* $Q = \{f(\mathbf{x}) \mid \mathbf{x} \in \mathcal{P}\}$.

Figure 2.4 depicts a Pareto-optimal front along with the corresponding objective space. Next, a number of definitions for specific solutions that are often used when discussing MOO, are listed:

Definition 2.10. ***Ideal Objective Vector*** *The ideal objective vector,* $\mathbf{z}^*$*, is a vector with components consisting of the optimal objective values for each of the* n_m *objective functions.* $\mathbf{z}^*$ *is formally defined as* $\mathbf{z}^* = (f_1^*, f_2^*, \ldots, f_{n_m}^*)$ *where* f_m^* *is the optimal objective value for objective* $m \in [1, n_m]$.

Definition 2.11. ***Utopian Objective Vector*** *The utopian objective vector,* $\mathbf{z}^{**}$*, is a vector with components marginally smaller than that of the ideal objective vector,* $\mathbf{z}^*$*.* $\mathbf{z}^{**}$ *is formally defined as* $\mathbf{z}^{**} = (z_1^* - \varepsilon_1, z_2^* - \varepsilon_2, \ldots, z_{n_m}^* - \varepsilon_{n_m})$ *where* $\varepsilon_m > 0\ \forall\ m \in [1, n_m]$.

Definition 2.12. ***Nadir Objective Vector*** *The nadir objective vector, $\mathbf{z}^{nad}$, is a vector with components consisting of the worst objective values in the Pareto-optimal set, $\mathcal{P}$, for each of the n_m objective functions. $\mathbf{z}^{nad}$ is formally defined as $\mathbf{z}^{nad} = (f_1^{**}, f_2^{**}, \ldots, f_{n_m}^{**})$ where $f_m^{**} = f_m(\mathbf{x}_1)$ with $\mathbf{x}_1 \in \mathcal{P} \mid \nexists\, \mathbf{x}_2 \in \mathcal{P} : f_m(\mathbf{x}_2) > f_m(\mathbf{x}_1)$.*

Figure 2.5 depicts a Pareto-optimal front along with the ideal objective, utopian objective, and nadir objective vectors.

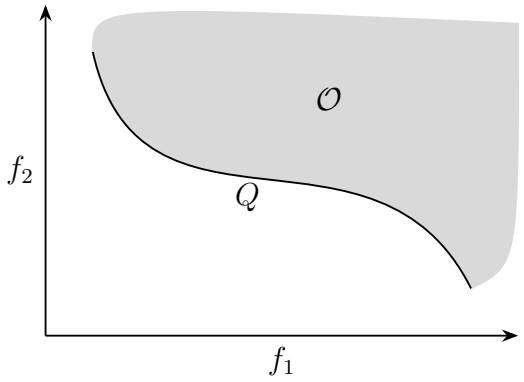


Figure 2.4: Pareto-optimal front.

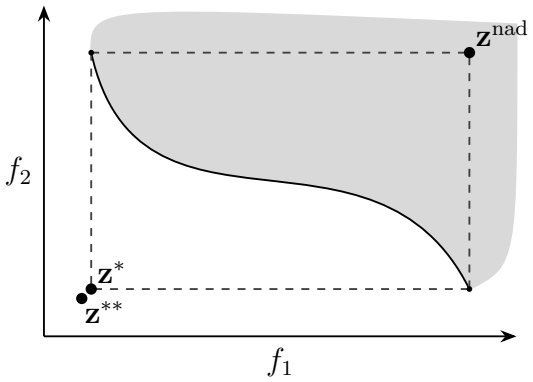


Figure 2.5: Ideal ($\mathbf{z}^*$), Utopian ($\mathbf{z}^{**}$), and Nadir ($\mathbf{z}^{\text{nad}}$) objective vectors.

2.3 Particle Swarm based Multi-objective Optimization

The success of PSO as a single-objective optimizer has motivated research into extensions that enable PSO to solve MOPs. The first such extension was proposed in an unpublished manuscript by Moore and Chapman [77] in 1999. Since this first proposal, much research has been put into developing additional extensions that enable PSO to solve MOPs. By 2006, over twenty-five such extensions have been proposed [17]. This study considers one such extension, proposed by Parsopoulos and Vrahatis [84, 85] in 2002, named VEPSO. For more information on other multi-objective PSO algorithms, the interested reader is referred to [17, 32, 86].

Section 2.3.1 provides the necessary background overview of VEPSO, followed by a discussion of alternative VEPSO knowledge transfer strategies (KTSs) in Section 2.3.2.

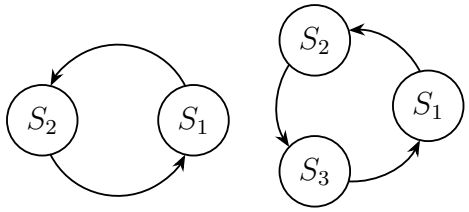
2.3.1 Vector Evaluated Particle Swarm Optimization

In 2002, Parsopoulos and Vrahatis [84, 85] proposed a MOO adaptation of PSO named vector evaluated particle swarm optimization (VEPSO). Parsopoulos and Vrahatis adopted the main ideas of the vector evaluated genetic algorithm (VEGA) by Schaffer [95] to fit the PSO framework.

VEGA constructs the mating pool by selecting individuals according to their fitness for each of the objectives separately. Afterwards, the mating pool is shuffled, and crossover and mutation are applied as per the normal genetic algorithm (GA) to generate the next generation's population.

Based on these ideas, VEPSO uses two subswarms in the case of two-objective MOPs, one for each of the objectives. Each subswarm is evaluated according to one of the objectives. The change in velocities is based on information coming from the other swarm. Specifically, the best particle position for one subswarm is selected as the best performing particle position from the other subswarm. Parsopoulos and Vrahatis [84] found that VEPSO outperformed VEGA for a number of the tested problems.

The selection of the best performing particle position from the other swarm is referred to as the ring KTS [87]. Figure 2.6 illustrates the flow of information for the ring KTS



(a) Two subswarms (b) Three subswarms

Figure 2.6: Ring knowledge transfer strategy

between the subswarms in the case of two- and three-objective MOPs.

The velocity update equation for VEPSO is identical to that of the basic PSO, as defined in equation (2.3), with the exception that the neighborhood best position, $\hat{\mathbf{y}}_i(t)$, also referred to as the KTS guide, is selected as the best performing particle position from the other subswarm. The velocity and position update equations for VEPSO's second subswarm, S_2, are thus defined as follows:

$$S_2.\mathbf{v}_i(t+1) = wS_2.\mathbf{v}_i(t) + c_1\mathbf{r}_1(S_2.\mathbf{y}_i(t) - S_2.\mathbf{x}_i(t)) + c_2\mathbf{r}_2(S_1.\hat{\mathbf{y}}_i(t) - S_2.\mathbf{x}_i(t)) \quad (2.10)$$

$$S_2.\mathbf{x}_i(t+1) = S_2.\mathbf{x}_i(t) + S_2.\mathbf{v}_i(t+1) \quad (2.11)$$

where $S_2.\hat{\mathbf{y}}_i(t)$ is the KTS guide for particle i at iteration t, selected as the best performing particle position from the subswarm S_1.

During each iteration, VEPSO stores the Pareto-optimal solutions in an archive. The archive contains only non-dominated solutions, and once the algorithm stops, the solutions in the archive form the Pareto-optimal set. Parsopoulos and Vrahatis [84, 85] did not provide any detail on how to implement the archive. When Parsopoulos *et al.* [88] introduced a parallel variant of VEPSO, they specified the archive implementation to match that of Jin *et al.* [58]. Algorithm 2 provides pseudo-code for a basic archive maintenance algorithm similar to that of Jin *et al.* [58]. Algorithm 3 presents the VEPSO algorithm.

2.3.2 Knowledge Transfer Strategies

Grobler [45] and Harrison *et al.* [47, 48] evaluated variations to the ring KTS as used in

the basic VEPSO algorithm. Grobler and Engelbrecht [46] and Grobler [45] evaluated the random KTS that randomly selects the swarm from which the best performing particle position is selected. Experimental results showed that the random KTS significantly outperformed the ring KTS along with several other KTS variations also evaluated by Grobler.

Harrison *et al.* [48] proposed KTSs based on parent-centric crossover (PCX) [28]. For the parent-centric crossover archive (PCXA) KTS, the KTS guide is calculated as the offspring of PCX applied to three randomly selected non-dominated solutions from the archive. PCXA is based on the assumption that areas in the decision space around the non-dominated solutions are worth exploring.

Harrison *et al.* [48] found that the PCXA KTS achieved the best spread of solutions as defined by Goh and Tan's [44] solution distribution measure.

The work presented in this thesis makes use of the random and PCXA KTSs.

2.4 Multi-objective Performance Measures

Comparing different optimization techniques empirically requires a means to define the quality of an algorithm's performance. In the case of MOO, the definition of quality is

Algorithm 2 Basic Archive Maintenance

```
1: for each particle i = 1, ..., n_s do
2:   if x_i is not dominated by any solution in the archive
3:       and x_i is not similar to any solutions in the archive then
4:     Remove all archive solutions that are dominated by x_i;
5:     if archive is not full then
6:       Add x_i to the archive;
7:     else
8:       Discard x_i;
9:     end if
10:  end if
11: end for
```

Algorithm 3 Vector Evaluated Particle Swarm Optimization

for each objective $m = 1, ..., n_m$ **do**
  Create and initialize a swarm, S_m, of n_{s_m} particles uniformly within a predefined hypercube of dimension n_x;
  Let f_m be the objective function;
  Let $S_m.\mathbf{y}_i$ represent the personal best position of particle $S_m.\mathbf{x}_i$, initialized to $S_m.\mathbf{x}_i(0)$;
  Let $S_m.\hat{\mathbf{y}}_i$ represent the KTS guide of particle $S_m.\mathbf{x}_i$;
  Initialize $S_m.\mathbf{v}_i(0)$ to $\mathbf{0}$;
end for
Let $t = 0$;
repeat
  for each objective $m = 1, ..., n_m$ **do**
    for each particle $i = 1, ..., S_m.n_s$ **do**
      if $f_m(S_m.\mathbf{x}_i) < f_m(S_m.\mathbf{y}_i)$ **then**
        $S_m.\mathbf{y}_i = S_m.\mathbf{x}_i(t)$;
      end if
      Update the archive with the solution $S_m.\mathbf{x}_i$;
    end for
  end for
  for each objective $m = 1, ..., n_m$ **do**
    for each particle $i = 1, ..., S_m.n_s$ **do**
      Select $S_m.\hat{\mathbf{y}}_i$ using a KTS;
      $S_m.\mathbf{v}_i(t+1) = wS_m.\mathbf{v}_i(t)+c_1\mathbf{r}_1(S_m.\mathbf{y}_i(t)-S_m.\mathbf{x}_i(t))+c_2\mathbf{r}_2(S_m.\hat{\mathbf{y}}_i(t)-S_m.\mathbf{x}_i(t))$;
      $S_m.\mathbf{x}_i(t+1) = S_m.\mathbf{x}_i(t) + S_m.\mathbf{v}_i(t+1)$;
    end for
  end for
  $t = t + 1$;
until stopping condition is true

substantially more complex than for a single objective optimization problem, because the optimization goal itself consists of multiple, often contradictory, objectives.

In order to define quality, the goals for MOO must first be defined. Zitzler *et al.* [118] defined the goals for a MOO algorithm as follows:

1. minimize the distance between the obtained Pareto-optimal front and the true Pareto-optimal front,
2. to find a set of solutions with a good, in most cases uniform, distribution, and
3. maximize the extent of the obtained Pareto-optimal front.

An alternative definition with only two goals was defined by Deb [24]:

1. to find a set of solutions as close as possible to the Pareto-optimal front, and
2. to find a set of solutions as diverse as possible.

Both definitions define a quality POF as having a good spread of solutions that are as close as possible to the true POF. Sections 2.4.1 through 2.4.3 discuss some of the popular MOO measurements, as used throughout this study, that measure closeness to the true POF, diversity, and combinations of both closeness to the true POF and diversity.

2.4.1 Closeness to the Pareto-optimal Front

Generational Distance

Van Veldhuizen and Lamont [108, 109] introduced the generational distance (GD) measure to quantify the distance from the obtained POF to the true POF. The GD measure is calculated as

$$GD = \frac{\sqrt{\sum_{i=1}^{|Q|} d_i^2}}{|Q|} \tag{2.12}$$

where Q is the obtained POF and d_i is the Euclidean distance between the $i'th$ solution in the obtained POF and the nearest solution in the true POF, Q_{true}.

2.4.2 Diversity among the Pareto-optimal Front

Maximum Spread

Zitzler [116] introduced the maximum spread measure to quantify the length of the diagonal of a hyper-box formed by the extreme function values. Tan *et al.* [102] modified the maximum spread measure to normalize the objective function values. The maximum spread measure is calculated as

$$\bar{D} = \sqrt{\frac{1}{n_m}\sum_{m=1}^{n_m}\left(\frac{max_m^* - min_m^*}{max_{true,m} - min_{true,m}}\right)^2} \tag{2.13}$$

with

$$max_m^* = \min\left\{\max_{k=1}^{|Q|}\{q_{k,m}\}, max_{true,m}\right\} \tag{2.14}$$

and

$$min_m^* = \max\left\{\min_{k=1}^{|Q|}\{q_{k,m}\}, min_{true,m}\right\} \tag{2.15}$$

where Q is the obtained POF, $|Q|$ is the cardinality of the set Q, $max_{true,m}$ and $min_{true,m}$ are the maximum and minimum values reached by the true POF for the m'th objective.

Spacing

Schott [96] introduced the spacing measure in 1995 to quantify the diversity of the POF. The spacing measure is formally defined as

$$\bar{S} = \sqrt{\frac{1}{|Q|-1}\sum_{k=1}^{|Q|}\left(\bar{d^*} - d_k^*\right)^2} \tag{2.16}$$

with

$$d_k^* = \min_{\mathbf{q}_l \in Q \wedge l \neq k}\sum_{j=1}^{n_x}|q_{k,j} - q_{l,j}| \tag{2.17}$$

and

$$\bar{d^*} = \sum_{k=1}^{|Q|}\frac{d_k^*}{|Q|} \tag{2.18}$$

where Q is the set of solutions that make up the obtained POF.

When the solutions are near uniformly spread, the resulting spacing, $\bar{S}$, values will be small.

Solution Distribution

Goh and Tan [44] introduced the distribution measure in 2007, based on the spacing measure. The distribution measure is formally defined as

$$D = \frac{1}{|Q|}\sqrt{\frac{1}{|Q|}\sum_{k=1}^{|Q|}\left(d_k - \bar{d}\right)^2} \tag{2.19}$$

with

$$\bar{d} = \frac{1}{|Q|}\sum_{k=1}^{|Q|} d_k \tag{2.20}$$

where d_k is the Euclidean distance in objective space between the k'th solution and its nearest neighbor in Q.

When the solutions are near uniformly spread, the resulting distribution, D, values will be small.

Spread

Deb *et al.* [29] introduced the spread measure to quantify the distance between solutions while also taking the extreme solutions into account. The spread measure is calculated as

$$\Delta = \frac{\sum_{m=1}^{n_m} d_m^e + \sum_{k=1}^{|Q|} |d_k - \bar{d}|}{\sum_{m=1}^{n_m} d_m^e + |Q|\bar{d}} \tag{2.21}$$

with

$$\bar{d} = \frac{1}{|Q|}\sum_{k=1}^{|Q|} d_k \tag{2.22}$$

where d_k can be any distance measure between neighboring solutions, and d_m^e is the distance between the extreme solutions of $|Q|$ and the true POF corresponding to the m'th objective.

An ideal distribution will have $\Delta = 0$ if the distances between solutions are equal and $|Q|$ contains the extreme solutions of the true POF, otherwise $\Delta > 0$.

2.4.3 Closeness and Diversity of the Pareto-optimal Front

Hypervolume

Zitzler and Thiele [117] introduced the hypervolume ($\mathcal{HV}$) measure to quantify the volume, in objective space, covered by the hypercubes, $\mathcal{V}_k$, for each solution $\mathbf{q}_k \in Q$, where $\mathcal{V}_k$ is the hypercube constructed between the reference point $\boldsymbol{\mathcal{W}}$ and the solution $\mathbf{q}_k$. Because the hypercube volumes overlap, the $\mathcal{HV}$ measure is defined as the union of all the hypercubes. The $\mathcal{HV}$ measure is calculated as

$$\mathcal{HV} = \text{volume}(\cup\, \mathcal{V}_k) \;\forall\; \mathbf{q}_k \in Q \tag{2.23}$$

Zitzler and Thiele [117] defined the reference point $\boldsymbol{\mathcal{W}}$ as the zero vector, $\mathbf{0}$.

Zitzler [116] noted that it is stated by Van Veldhuizen [107] that the $\mathcal{HV}$ measure may be misleading if the POF is non-convex. Zitzler concluded that this fact simply indicates that the coverage of the objective space is only one of several possible criteria that should be used to evaluate the quality of the POF. In contrast to Zitzler and Thiele [117], more recent studies all use the nadir vector, $\mathbf{z}^{\text{nad}}$, as the reference point. When using the nadir vector as the reference point, the quality of the POF increases as the $\mathcal{HV}$ increases, whereas when using the zero vector the quality of the POF increases as the $\mathcal{HV}$ decreases.

Zitzler [116] also noted that the $\mathcal{HV}$ values for two sets, A and B, cannot be used to derive whether either set entirely dominates the other. In conclusion, it is found that the $\mathcal{HV}$ measure alone is not enough to fully understand and quantify the POF. It is recommended that additional performance measures should be used in conjunction with the $\mathcal{HV}$.

Calculation of the hypervolume becomes expensive as the number of Pareto-optimal solutions and the number of dimensions increase, so does the complexity of the overlap calculations. Note that faster calculation algorithms do exist [5].

Inverted Generational Distance

Coello Coello and Reyes-Sierra [16, 91] introduced the inverted generational distance

(IGD) measure. The IGD measure is calculated as

$$IGD = \frac{\sqrt{\sum_{k=1}^{|Q_{true}|} d_k^2}}{|Q_{true}|} \tag{2.24}$$

where Q_{true} is the true POF and d_k is the distance between the $k'th$ solution in the true POF and the nearest solution in the known POF, Q.

The quality of the IGD measure depends on the quality of the true POF, Q_{true}. Ishibuchi *et al.* [55] investigated the difficulties in specifying the reference points that make up the true POF. Ishibuchi *et al.* also showed how uniform sampling of reference points from the known true POF leads to counter-intuitive results, highlighting the importance of selecting the right solutions for the set Q_{true}. The work presented in this study made use of the true POF sets from the jMetal framework [80] for all the IGD calculations. The true POF sets in the jMetal framework were mathematically derived using the function definitions.

Attainment Surface

Fonseca and Fleming [35] presented a non-quantitative way to asses the performance of the POF obtained by a MOO algorithm. Given a set of non-dominated solutions, $Q \subseteq \mathcal{O}$, a boundary function that divides the objective space, $\mathcal{O}$, into two regions, i.e. the region weakly dominated by Q and the region not dominated by Q, can be found. Fonseca and Fleming called this boundary function, which can also be seen as the locus of the family of tightest objective vectors known to be attainable, the attainment surface.

Fonseca and Fleming described a method to compute the x%-attainment surface, using randomly generated intersection lines that takes multiple optimization runs into account, where x can be chosen arbitrarily. The x%-attainment surface, in conjunction with statistical tests, can be used to compare POFs obtained from multiple MOO algorithms. Zitzler [116] noted that a drawback of the attainment surface approach is that no clear way is given how to express the quality difference between POFs obtained from two different algorithms. Zitzler did, however, note that Fonseca and Fleming's approach allows for meaningful statistical interpretations and is well suited for visualizing the POF over several runs.

A more thorough treatment of attainment surfaces is given in chapter 5.

2.5 Test Problem Sets

When attempting to better understand the strengths and weaknesses of an algorithm, it is important to have a strong understanding of the problem at hand. This is as true for the field of MOO as it is for any other field. For a MOO algorithm, the performance in terms of the two primary goals of convergence and diversity must be evaluated. For an accurate evaluation, a number of problem characteristics must be covered [24]. Firstly, the problems should present a challenge for convergence. Multimodality, deception, and isolated optima are known problem areas for convergence for single objective problems. Secondly, POFs with convexity or non-convexity, discreteness, and non-uniformity properties provide a challenge for diversity.

The two benchmark problem sets used throughout this study are discussed next. Take note that to avoid confusion the notation and symbols used throughout this section are kept similar to original papers that introduced these benchmark problem sets.

2.5.1 Zitzler-Deb-Thiele's Test Problems

Zitzler *et al.* [118] framed six test problems, referred to as the Zitzler-Deb-Thiele (ZDT) test problems, ZDT1 through ZDT6. The ZDT problems were constructed following the process described by Deb [24]. All six ZDT problems have two objectives defined as

$$\begin{aligned} &minimize \quad f_1(\mathbf{x}) \\ &minimize \quad f_2(\mathbf{x}) = g(\mathbf{x})h(f_1(\mathbf{x}), g(\mathbf{x})) \end{aligned} \tag{2.25}$$

The six ZDT test problems vary in the way that the functions $f_1(\mathbf{x})$, $g(\mathbf{x})$, and $h(\mathbf{x})$ are defined. The POF is formed by $g(\mathbf{x}) = 1$ in all functions, except ZDT5. ZDT5 is a Boolean function defined over bit-strings and is not considered in the work presented in this thesis.

The remaining five ZDT test problems are defined as follows:

ZDT1

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 \\ g(\mathbf{x}) &= 1 + \frac{9}{n-1}\sum_{i=2}^{n} x_i \\ h(f_1, g) &= 1 - \sqrt{\frac{f_1}{g}} \end{aligned} \tag{2.26}$$

where $n = 30$, and $x_i \in [0, 1]$. ZDT1 has a convex POF.

ZDT2

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 \\ g(\mathbf{x}) &= 1 + 9\sum_{i=2}^{n} \frac{x_i}{n-1} \\ h(f_1, g) &= 1 - \left(\frac{f_1}{g}\right)^2 \end{aligned} \tag{2.27}$$

where $n = 30$, and $x_i \in [0, 1]$. ZDT2 is the non-convex, concave POF counterpart to ZDT1.

ZDT3

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 \\ g(\mathbf{x}) &= 1 + 9\sum_{i=2}^{n} \frac{x_i}{n-1} \\ h(f_1, g) &= 1 - \sqrt{\frac{f_1}{g}} - \left(\frac{f_1}{g}\right)\sin(10\pi f_1) \end{aligned} \tag{2.28}$$

where $n = 30$, and $x_i \in [0, 1]$. The sine function in $h(\mathbf{x})$ causes discontinuity in the POF. Take note that the discontinuity is only in the POF. The decision space is not discontinuous. ZDT3 has discrete features; its POF consists of several non-contiguous convex parts.

ZDT4

$$\begin{aligned} f_1(\mathbf{x}) &= x_1 \\ g(\mathbf{x}) &= 1 + 10(n-1) + \sum_{i=2}^{n}(x_i^2 - 10\cos(4\pi x_i)) \\ h(f_1, g) &= 1 - \sqrt{\frac{f_1}{g}} \end{aligned} \tag{2.29}$$

where $n = 10$, $x_1 \in [0, 1]$ and $x_2, \ldots, x_n \in [-5, 5]$. The best local POF is formed with $g(\mathbf{x}) = 1.25$. ZDT4 is multimodal and contains 21^9 local POFs.

ZDT6

$$\begin{aligned} f_1(\mathbf{x}) &= 1 - e^{-4x_1}\sin^6(6\pi x_1) \\ g(\mathbf{x}) &= 1 + 9\left(\frac{\sum_{i=2}^{n} x_i}{n-1}\right)^{\frac{1}{4}} \\ h(f_1, g) &= 1 - \left(\frac{f_1}{g}\right)^2 \end{aligned} \tag{2.30}$$

where $n = 10$, and $x_i \in [0, 1]$. ZDT6 has a non-convex, concave POF presenting two challenges. Firstly, the POF solutions are non-uniformly distributed along the global POF with the POF biased towards solutions where $f_1(\mathbf{x})$ is near one. Secondly, the density of the solutions is lowest near the POF and highest away from the POF.

2.5.2 Walking Fish Group Test Problems

Huband *et al.* [52, 53] proposed a test suite, referred to as the Walking Fish Group (WFG) test problems, that consists of nine scalable MOPs, WFG1 through WFG9. All

the WFG problems are of the following form:

$$\begin{aligned}
given\ \mathbf{z} &= \{z_1, \ldots, z_k, z_{k+1}, \ldots, z_n\} \\
minimize\ f_m(\mathbf{x}) &= Dx_M + S_m h_m(x_1, \ldots, x_{M-1})\ \forall\ m \in [1, M] \\
where\ \mathbf{x} &= \{x_1, \ldots, x_M\} \\
&= \{\max(t^p_M, A_1)(t^p_1 - 0.5) + 0.5, \ldots, \\
&\qquad \max(t^p_M, A_{M-1})(t^p_{M-1} - 0.5) + 0.5, t^p_M\} \\
\mathbf{t}^p &= \{t^p_1, \ldots, t^p_M\} \leftarrowtail \mathbf{t}^{p-1} \leftarrowtail \ldots \leftarrowtail \mathbf{t}^1 \leftarrowtail \mathbf{z}_{[0,1]} \\
\mathbf{z}_{[0,1]} &= \{z_{1,[0,1]}, \ldots, z_{n,[0,1]}\} \\
&= \left\{ \frac{z_1}{z_{1,\max}}, \ldots, \frac{z_n}{z_{n,\max}} \right\}
\end{aligned} \tag{2.31}$$

where

- M is the number of objectives, $\mathbf{x}$ is a set of M underlying parameters, where $\mathbf{x}_M$ is an underlying distance parameter and $\mathbf{x}_{1:M-1}$ are underlying position parameters;
- $\mathbf{z}$ is a set of $k + l = n \leq M$ working parameters, where the first k working parameters are position-related parameters and the last l working parameters are distance-related parameters;
- $D > 0$ is a distance scaling constant;
- $A_{1:M-1} \in 0, 1$ are degeneracy constants, and for each $A_i = 0$ the dimensionality of the POF is reduced by one;
- $h_{1:M}$ are shape functions;
- $S_{1:M} > 0$ are scaling constants; and
- $\mathbf{t}^{1:p}$ are transition vectors where "$\leftarrowtail$" indicates that each transition vector is created from another vector via transformation functions.

The domain of all $z_i \in \mathbf{z}$ is $[0, z_{i,\max}]$ where all $z_{i,\max} > 0$. Note that all $x_i \in \mathbf{x}$ will have domain $[0, 1]$.

Shape functions determine the nature of the POF, and map parameters with domain $[0, 1]$ onto the range $[0, 1]$. The following shape functions are defined:

Linear

$$\begin{aligned}
\text{linear}_1(x_1, \ldots, x_{M-1}) &= \prod_{i=1}^{M-1} x_i \\
\text{linear}_{m=2:M-1}(x_1, \ldots, x_{M-1}) &= (\prod_{i=1}^{M-m} x_i)(1 - x_{M-m+1}) \\
\text{linear}_M(x_1, \ldots, x_{M-1}) &= 1 - x_1
\end{aligned} \tag{2.32}$$

When $h_{1:M} = \text{linear}_m$, the POF is a linear hyperplane, where $\sum_{m=1}^{M} h_m = 1$.

Convex

$$\begin{aligned}
\text{convex}_1(x_1, \ldots, x_{M-1}) &= \prod_{i=1}^{M-1} (1 - \cos(x_i \frac{\pi}{2})) \\
\text{convex}_{m=2:M-1}(x_1, \ldots, x_{M-1}) &= (\prod_{i=1}^{M-m} (1 - \cos(x_i \frac{\pi}{2})))(1 - \sin(x_{M-m+1} \frac{\pi}{2})) \\
\text{convex}_M(x_1, \ldots, x_{M-1}) &= 1 - \sin(x_1 \frac{\pi}{2})
\end{aligned} \tag{2.33}$$

When $h_{1:M} = \text{convex}_m$, the POF is purely convex.

Concave

$$\begin{aligned}
\text{concave}_1(x_1, \ldots, x_{M-1}) &= \prod_{i=1}^{M-1} (1 - \sin(x_i \frac{\pi}{2})) \\
\text{concave}_{m=2:M-1}(x_1, \ldots, x_{M-1}) &= (\prod_{i=1}^{M-m} \sin(x_i \frac{\pi}{2}))(\cos(x_{M-m+1} \frac{\pi}{2})) \\
\text{concave}_M(x_1, \ldots, x_{M-1}) &= \cos(x_1 \frac{\pi}{2})
\end{aligned} \tag{2.34}$$

When $h_{1:M} = \text{concave}_m$, the POF is purely concave, and a region of the hypersphere of radius one is centered at the origin, where $\sum_{m=1}^{M} h_m^2 = 1$.

Mixed convex/concave ($\alpha > 0, A \in \{1, 2, \ldots\}$)

$$\text{mixed}_M(x_1, \ldots, x_{M-1}) = (1 - x_1 - \frac{\cos(2A\pi x_1 + \frac{\pi}{2})}{2A\pi})^{\alpha} \tag{2.35}$$

Equation (2.35) causes the POF to contain both convex and concave segments, the number of which is controlled by A. The overall shape is controlled by α: when $\alpha > 1$, the overall shape is convex, when $\alpha < 1$, the overall shape is concave, and when $\alpha = 1$, the overall shape is linear.

Disconnected ($\alpha, \beta > 0, A \in \{1, 2, \dots\}$)

$$\text{disc}_M(x_1, \dots, x_{M-1}) = 1 - (x_1)^\alpha \cos^2(A(x_1)^\beta \pi) \tag{2.36}$$

Equation (2.36) causes the POF to have disconnected regions, the number of which is controlled by A. The overall shape is controlled by α: when $\alpha > 1$, the overall shape is convex, when $\alpha < 1$, the overall shape is concave, and when $\alpha = 1$, the overall shape is linear. The location of the disconnected regions is controlled by β: larger values of β push the location of the disconnected regions towards larger values of x_1, and vice versa.

Transformation functions map input parameters with domain $[0, 1]$ onto the range $[0, 1]$. For brevity, a weighted product reduction function (analogous to the weighted sum reduction function) has been omitted. The following transformation functions have been defined:

Bias: Polynomial ($\alpha > 0, \alpha \neq 1$)

$$\text{b\_poly}(y, \alpha) = y^\alpha \tag{2.37}$$

When $\alpha > 1$, y is biased towards zero, and when $\alpha < 1$, y is biased towards one.

Bias: Flat Region ($A, B, C \in [0, 1], B < C, B = 0 \Rightarrow A = 0 \wedge C \neq 1, C = 1 \Rightarrow A = 1 \wedge B \neq 0$)

$$\begin{aligned} \text{b\_flat}(y, A, B, C) = A &+ \min(0, \lfloor y - B \rfloor) \frac{A(B - y)}{B} \\ &- \min(0, \lfloor C - y \rfloor) \frac{(1 - A)(y - C)}{1 - C} \end{aligned} \tag{2.38}$$

Values of y between B and C – the area of the flat region – are all mapped to the value of A.

Bias: Parameter Dependent ($A \in (0,1), 0 < B < C$)

$$\begin{aligned} \text{b\_param}(y, \mathbf{y}', A, B, C) &= y^{B+(C-B)v(u(\mathbf{y}'))} \\ v(u(\mathbf{y}')) &= A - (1 - 2u(\mathbf{y}'))|\lfloor 0.5 - u(\mathbf{y}'\rfloor + A| \end{aligned} \tag{2.39}$$

A, B, C, and the secondary parameter vector $\mathbf{y}'$ together determine the degree to which y is biased by being raised to an associated power, values of $u(\mathbf{y}') \in [0, 0.5]$ are mapped linearly onto $[B, B + (C - B)A]$, and values of $u(\mathbf{y}') \in [0.5, 1]$ are mapped linearly onto $[B + (C - B)A, C]$.

Shift: Linear ($A \in (0,1)$)

$$\text{s\_linear}(y, A) = \frac{y - A}{|\lfloor A - y\rfloor + A|} \tag{2.40}$$

Shift: Deceptive ($A \in (0,1), 0 < B \ll 1, 0 < C \ll 1, A - B > 0, A + B < 1$)

$$\begin{aligned} \text{s\_decept}(y, A, B, C) = 1 + (|y - A| - B)\Bigg(&\frac{\lfloor y - A + B\rfloor (1 - C + \frac{A-B}{B})}{A - B} \\ &+ \frac{\lfloor A + B - y\rfloor (1 - C + \frac{1-A-B}{B})}{1 - A - B} + \frac{1}{B}\Bigg) \end{aligned} \tag{2.41}$$

A is the global minimum of the transformation. B is the "aperture" size of the well/basin leading to the global minimum at A, and C is the value of the deceptive minima. There are always two deceptive minima.

Shift: Multimodal ($A \in \{1, 2, \dots\}, B \geq 0, (4A + 2)\pi \geq 4B, C \in (0,1)$

$$\text{s\_decept}(y, A, B, C) = \frac{1 + \cos((4A + 2)\pi(0.5 - \frac{|y-C|}{2(\lfloor C-y\rfloor + C)})) + 4B(\frac{|y-C|}{2(\lfloor C-y\rfloor + C)})^2}{B + 2} \tag{2.42}$$

A controls the number of minima, B controls the magnitude of the "hill sizes" of the multi-modality, and C is the value for which y is mapped to zero. When $B = 0$, $2A + 1$ values of y, one at C, are mapped to zero, and when $B \neq 0$, there are $2A$ local minima, and one global minimum at C. Larger values of A and smaller values of B create more difficult problems.

Reduction: Weighted Sum ($|\mathbf{w}| = |\mathbf{y}|, w_1, \ldots, w_{|y|} > 0$)

$$\text{r\_sum}(\mathbf{y}, \mathbf{w}) = \frac{\sum_{i=1}^{|\mathbf{y}|} w_i y_i}{\sum_{i=1}^{|\mathbf{y}|} w_i} \tag{2.43}$$

The constant weight vector $\mathbf{w}$ forces an algorithm to treat the parameter vector, $\mathbf{y}$, differently.

Reduction: Non-separable ($A \in \{1, \ldots, |\mathbf{y}|\}, |\mathbf{y}| \mod A = 0$)

$$\text{r\_nonsep}(\mathbf{y}, A) = \frac{\sum_{j=1}^{|\mathbf{y}|}(y_j + \sum_{k=0}^{A-2} |y_j - y_1 + (j+k) \mod |\mathbf{y}|)}{\frac{|\mathbf{y}|}{A}\lceil\frac{A}{2}\rceil(1 + 2A - 2\lceil\frac{A}{2}\rceil)} \tag{2.44}$$

A controls the degree of non-separability, noting that

$$\text{r\_nonsep}(\mathbf{y}, 1) = \text{r\_sum}(\mathbf{y}, \mathbf{1})$$

Bias transformations impact the search process by biasing the fitness landscape. Shift transformations move the location of optima. In the absence of any shift, all distance-related parameters would be extremal parameters, with optimal value at zero. Shift transformations can be used to set the location of parameter optima, subject to skewing by bias transformations. Setting the location of parameter optima using shift transformations is useful if medial and extremal parameters are to be avoided. It is recommended that all distance-related parameters be subjected to at least one shift transformation. The deceptive and multimodal shift transformations make the corresponding problem deceptive and multimodal, respectively. The flat region transformation can have a significant impact on the fitness landscape and can also be used to create a many-to-one mapping from the POF to the POS.

To ensure problems are well designed, the following restrictions apply:

Constants

Constants must be fixed values and cannot be tied to the value of any parameter.

Primary Parameters

For any given transition vector, all the parameters of the originating transition vector must be employed exactly once as a primary parameter, counting parameters that appear independently as primary parameters, and in the same order in which the parameters appear in the originating transition vector.

Secondary Parameters

Care must be taken to avoid cyclical dependencies in b_param. If a is a primary parameter of b_param, and b is a secondary parameter, the a depends on b. If b likewise depends on c, the a depends, indirectly, on c. To prevent cyclical dependencies, no two parameters should be dependent on one another. A parameter should also not depend on itself.

Shifts

Parameters should only be subjected to a maximum of one shift transformation.

Reductions

Reduction transformation should belong to transition vectors that are closer to the underlying parameter vector than any shift transformations.

b_flat

When $A = 0$, b_flat should only belong to transition vectors that are further away from the underlying parameter vector than any shift or reduction transformation.

The constant values and domains of the working parameters for the nine WFG test problems are defined as follows:

Constants

$$\begin{aligned} S_{m=1:M} &= 2m \\ A_1 &= 1 \\ A_{2:M-1} &= \begin{cases} 0 & \text{for WFG3} \\ 1 & \text{otherwise} \end{cases} \end{aligned} \tag{2.45}$$

The settings for $S_{1:M}$ ensure that the POFs have dissimilar trade-off magnitudes, and the settings for $A_{1:M-1}$ ensure that the POFs are not degenerate, except in the case of WFG3, which has a one-dimensional POF.

Domains

$$z_{i=1:n,max} = 2i \tag{2.46}$$

The working parameters have domains of dissimilar magnitude.

Finally, the shape and transformations for the nine WFG test problems are defined as follows:

WFG1

$$\begin{aligned} \text{Shape} \qquad h_{m=1:M-1} &= \text{convex}_m \\ h_M &= \text{mixed}_M, \text{ with } \alpha = 1 \text{ and } A = 5 \\ \mathbf{t^1} \qquad t^1_{i=1:k} &= y_i \\ t^1_{i=k+1:n} &= \text{s\_linear}(y_i, 0.35) \\ \mathbf{t^2} \qquad t^2_{i=1:k} &= y_i \\ t^2_{i=k+1:n} &= \text{b\_flat}(y_i, 0.8, 0.75, 0.85) \\ \mathbf{t^3} \qquad t^3_{i=1:n} &= \text{b\_poly}(y_i, 0.02) \\ \mathbf{t^4} \qquad t^4_{i=1:M-1} &= \text{r\_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), \\ &\qquad (2((i-)k/(M-1)+1), \ldots, 2ik/(M-1))) \\ t^4_M &= \text{r\_sum}((y_{k+1}, \ldots, y_n), (2(k+1), \ldots, 2n)) \end{aligned} \tag{2.47}$$

WFG2

$$
\begin{aligned}
\text{Shape} \qquad & h_{m=1:M-1} = \text{convex}_m \\
& h_M = \text{disc}_M, \text{ with } \alpha = \beta = 1 \text{ and } A = 5 \\
\mathbf{t^1} \qquad & t^1_{i=1:k} = y_i \\
& t^1_{i=k+1:n} = \text{s\_linear}(y_i, 0.35) \\
\mathbf{t^2} \qquad & t^2_{i=1:k} = y_i \\
& t^2_{i=k+1:n} = \text{r\_nonsep}((y_{k+2(i-k)-1}, y_{k+2(i-k)}), 2) \\
\mathbf{t^3} \qquad & t^3_{i=1:M-1} = \text{r\_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), (1, \ldots, 1)) \\
& t^3_M = \text{r\_sum}((y_{k+1}, \ldots, y_{k+l/2}), (1, \ldots, 1))
\end{aligned} \tag{2.48}
$$

WFG3

$$
\begin{aligned}
\text{Shape} \qquad & h_{m=1:M} = \text{linear}_m \\
\mathbf{t^1} \qquad & t^1_{i=1:k} = y_i \\
& t^1_{i=k+1:n} = \text{s\_linear}(y_i, 0.35) \\
\mathbf{t^2} \qquad & t^2_{i=1:k} = y_i \\
& t^2_{i=k+1:n} = \text{r\_nonsep}((y_{k+2(i-k)-1}, y_{k+2(i-k)}), 2) \\
\mathbf{t^3} \qquad & t^3_{i=1:M-1} = \text{r\_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), (1, \ldots, 1)) \\
& t^3_M = \text{r\_sum}((y_{k+1}, \ldots, y_{k+l/2}), (1, \ldots, 1))
\end{aligned} \tag{2.49}
$$

WFG4

$$
\begin{aligned}
\text{Shape} \qquad & h_{m=1:M} = \text{concave}_m \\
\mathbf{t^1} \qquad & t^1_{i=1:n} = \text{s\_multi}(y_i, 30, 10, 0.35) \\
\mathbf{t^2} \qquad & t^2_{i=1:M-1} = \text{r\_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), (1, \ldots, 1)) \\
& t^2_M = \text{r\_sum}((y_{k+1}, \ldots, y_n), (1, \ldots, 1))
\end{aligned} \tag{2.50}
$$

WFG5

$$\begin{aligned}
\text{Shape} \qquad & h_{m=1:M} = \text{concave}_m \\
\mathbf{t^1} \qquad & t^1_{i=1:n} = \text{s_decept}(y_i, 0.35, 0.001, 0.05) \\
\mathbf{t^2} \qquad & t^2_{i=1:M-1} = \text{r_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), (1, \ldots, 1)) \\
& t^2_M = \text{r_sum}((y_{k+1}, \ldots, y_n), (1, \ldots, 1))
\end{aligned} \tag{2.51}$$

WFG6

$$\begin{aligned}
\text{Shape} \qquad & h_{m=1:M} = \text{concave}_m \\
\mathbf{t^1} \qquad & t^1_{i=1:k} = y_i \\
& t^1_{i=k+1:n} = \text{s_linear}(y_i, 0.35) \\
\mathbf{t^2} \qquad & t^2_{i=1:M-1} = \text{r_nonsep}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), k/(M-1)) \\
& t^2_M = \text{r_nonsep}((y_{k+1}, \ldots, y_n), l)
\end{aligned} \tag{2.52}$$

WFG7

$$\begin{aligned}
\text{Shape} \qquad & h_{m=1:M} = \text{concave}_m \\
\mathbf{t^1} \qquad & t^1_{i=1:k} = \text{b_param}(y_i, \text{r_sum}((y_{i+1}, \ldots, y_n), (1, \ldots, 1)), \frac{0.98}{49.98}, 0.02, 50) \\
& t^1_{i=k+1:n} = y_i \\
\mathbf{t^2} \qquad & t^2_{i=1:k} = y_i \\
& t^2_{i=k+1:n} = \text{s_linear}(y_i, 0.35) \\
\mathbf{t^3} \qquad & t^3_{i=1:M-1} = \text{r_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), (1, \ldots, 1)) \\
& t^3_M = \text{r_sum}((y_{k+1}, \ldots, y_n), (1, \ldots, 1))
\end{aligned} \tag{2.53}$$

WFG8

$$
\begin{aligned}
\text{Shape} \quad & h_{m=1:M} = \text{concave}_m \\
\mathbf{t^1} \quad & t^1_{i=1:k} = y_i \\
& t^1_{i=k+1:n} = \text{b\_param}(y_i, \text{r\_sum}((y_1, \ldots, y_{i-1}), (1, \ldots, 1)), \frac{0.98}{49.98}, 0.02, 50) \\
\mathbf{t^2} \quad & t^2_{i=1:k} = y_i \\
& t^2_{i=k+1:n} = \text{s\_linear}(y_i, 0.35) \\
\mathbf{t^3} \quad & t^3_{i=1:M-1} = \text{r\_sum}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), (1, \ldots, 1)) \\
& t^3_M = \text{r\_sum}((y_{k+1}, \ldots, y_n), (1, \ldots, 1))
\end{aligned} \tag{2.54}
$$

WFG9

$$
\begin{aligned}
\text{Shape} \quad & h_{m=1:M} = \text{concave}_m \\
\mathbf{t^1} \quad & t^1_{i=1:n-1} = \text{b\_param}(y_i, \text{r\_sum}((y_{i+1}, \ldots, y_n), (1, \ldots, 1)), \frac{0.98}{49.98}, 0.02, 50) \\
& t^1_n = y_n \\
\mathbf{t^2} \quad & t^2_{i=1:k} = \text{s\_decep}(y_i, 0.35, 0.001, 0.05) \\
& t^2_{i=k+1:n} = \text{s\_multi}(y_i, 30, 95, 0.35) \\
\mathbf{t^3} \quad & t^3_{i=1:M-1} = \text{r\_nonsep}((y_{(i-1)k/(M-1)+1}, \ldots, y_{ik/(M-1)}), k/(M-1)) \\
& t^3_M = \text{r\_nonsep}((y_{k+1}, \ldots, y_n), l)
\end{aligned} \tag{2.55}
$$

2.6 Summary

The objective of this chapter was to provide background information on all the various computational intelligence techniques, performance measures, and benchmark suites employed throughout this study. Specific focus was placed on particle swarm optimization, multi-objective optimization, multi-objective particle swarm optimization, multi-objective measurements, and multi-objective benchmark suites. Particle swarm optimization was presented along with the various velocity update models, neighborhood structures, and convergence and stability criteria were discussed. Definitions that are commonly found in multi-objective optimization literature were given and explained.

Vector evaluated particle swarm optimization was discussed as an example of a multi-objective particle swarm optimization algorithm. A discussion on multi-objective performance measures was given. Finally, a detailed overview of two multi-objective problem toolkits and their associated test problems, as used throughout this study, was given.

The next chapter investigates the exploration behavior of the vector evaluated particle swarm optimization algorithm using the aforementioned test sets.

Chapter 3

Vector Evaluated Particle Swarm Optimization Exploration Behaviour

"The eye sees only what the mind is prepared to comprehend."

Robertson Davies (1913 - 1995)

The previous chapter presented VEPSO, a PSO variant that deals with MOO. Previous studies by Fieldsend [34] and Matthysen *et al.* [74] have shown that VEPSO suffers from a stagnation problem. In order to better understand the VEPSO search process and why the search stagnates, this chapter presents an explorative analysis of VEPSO in low-dimensional objective space. A new candidate solution visualization approach is introduced to visually analyze the search process of VEPSO. The newly introduced visualization approach can also be applied to other MOO algorithms. The main objective of the analysis is to better understand why VEPSO stagnates and fails to find better solutions as the number of iterations increases. It is hypothesized that the stagnation and poor performance can be attributed to a lack of exploitation. In order to test the lack of exploitation hypothesis, an archive-guided PSO algorithm is developed and compared against VEPSO. A number of new quantitative measurements are introduced to assist with validating the lack of exploitation hypothesis.

Section 3.1 presents and discusses the POFs obtained from running the VEPSO algorithm. The behavior of the candidate solutions, the solutions in objective space that

each particle's current position represents, is investigated. Section 3.2 presents an analysis of candidate solution dispersion in order to identify why the VEPSO performs poorly. Section 3.3 introduces and presents an analysis of an archive-guided PSO algorithm that attempts to address the shortcomings of the VEPSO algorithm. Section 3.4 presents an analysis of the movement diversity of the candidate solution using the newly introduced candidate solution movement diversity measure. Finally, Section 3.5 presents a summary of the findings of this chapter.

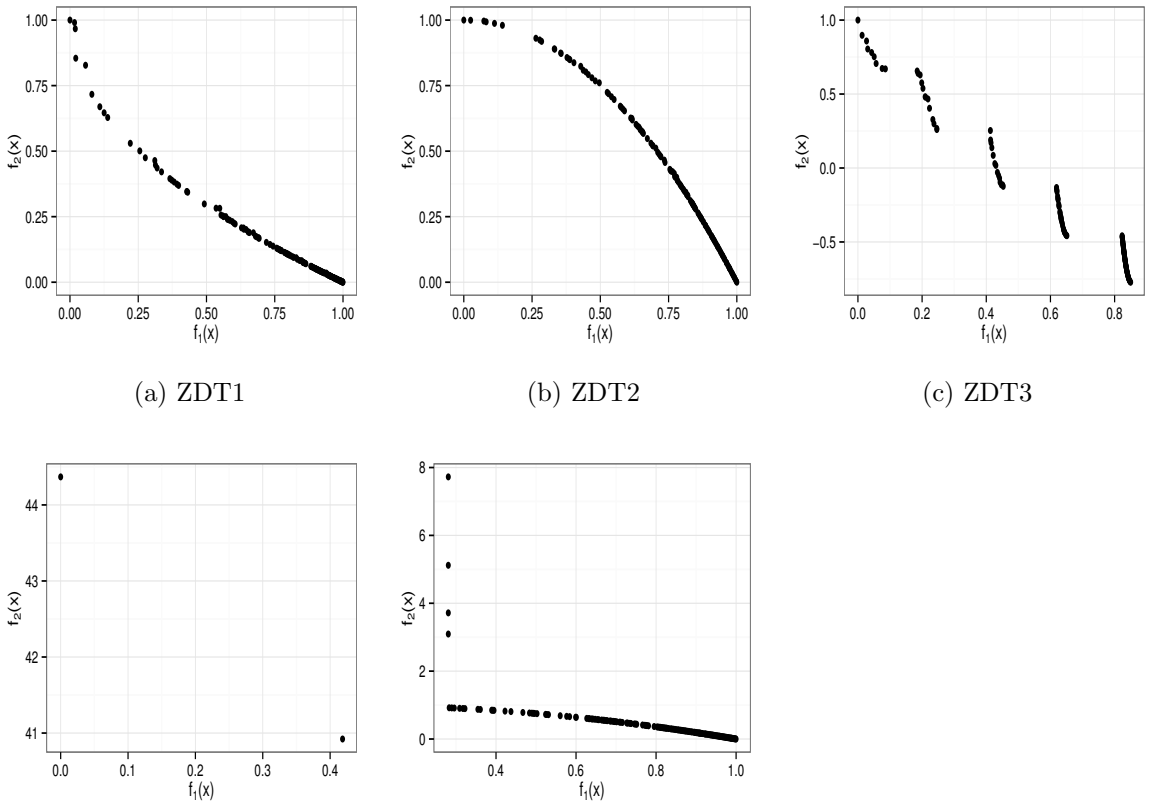
3.1 Explorative Analysis

The analysis presented in this section made use of the VEPSO algorithm using a ring KTS with a maximum archive size of 150 with distance based pruning [2]. The results presented were taken over 30 independent runs of 2000 iterations. The inertia weight, w, was set to 0.729844, and the acceleration constants, c_1 and c_2, were set to 1.49618.

Figures 3.1(a) through 3.1(e) show the POF for the ZDT1 through ZDT6 problems and Figures 3.2(a) through 3.2(i) show the POF for the WFG1 through WFG9 problems. Visual inspection of the POFs has shown that the VEPSO algorithm managed to find a close approximation to the POF for the majority of test problems. However, in many of the cases, the POF is not smooth and have a poor spread. Additionally, for test problem ZDT4 the POF is not found. Increasing the number of iterations did not improve the found POFs, and it is concluded, as with previous studies, that the VEPSO algorithm stagnates [34, 74].

In order to better understand why the VEPSO stagnates, the behavior of the candidate solutions represented by the objective vectors was tracked and plotted. Figures 3.3(a) through 3.3(j) show the resulting plots for ZDT1 through ZDT6 and figures 3.4(a) through 3.4(r) show the resulting plots for WFG1 through WFG9. For each problem, the candidate solution plots for each of the two swarms, representing the two objectives, are presented. Visual inspection of the figures shows that the particles continue to move around the objective space without converging towards the POF. The particle movement almost seems random.

As stated in the previous chapter, Deb [25] defined the two distinct goals for a MOO



(a) ZDT1 (b) ZDT2 (c) ZDT3

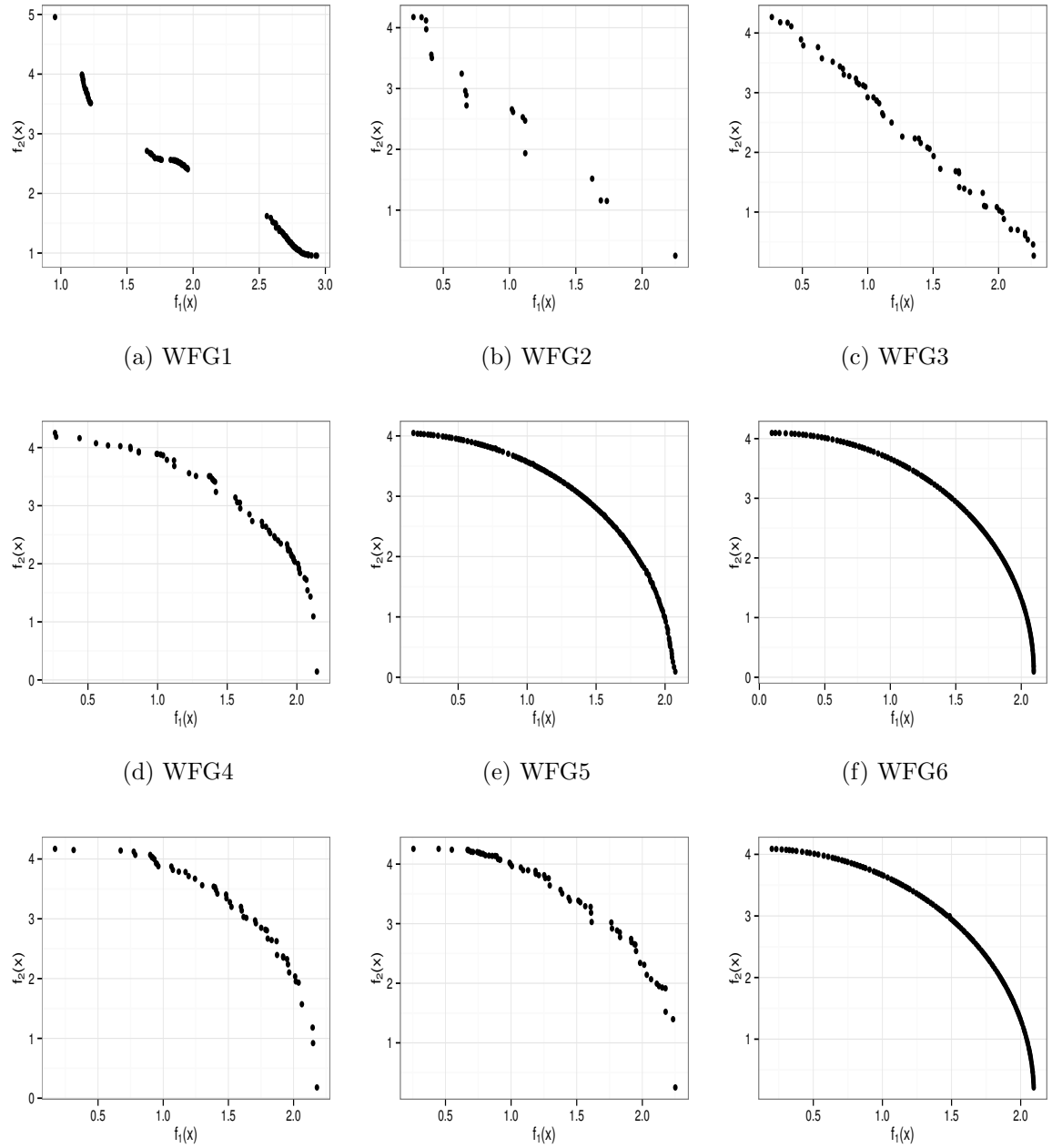
(d) ZDT4 (e) ZDT6

Figure 3.1: VEPSO calculated POFs for ZDT problems

algorithm as follows:

1. discover solutions as close to the Pareto-optimal solutions as possible, and
2. find solutions as diverse as possible in the obtained non-dominated front.

In contrast to these goals, the candidate solution plots show that the particles are not moving towards or spreading along the POF.

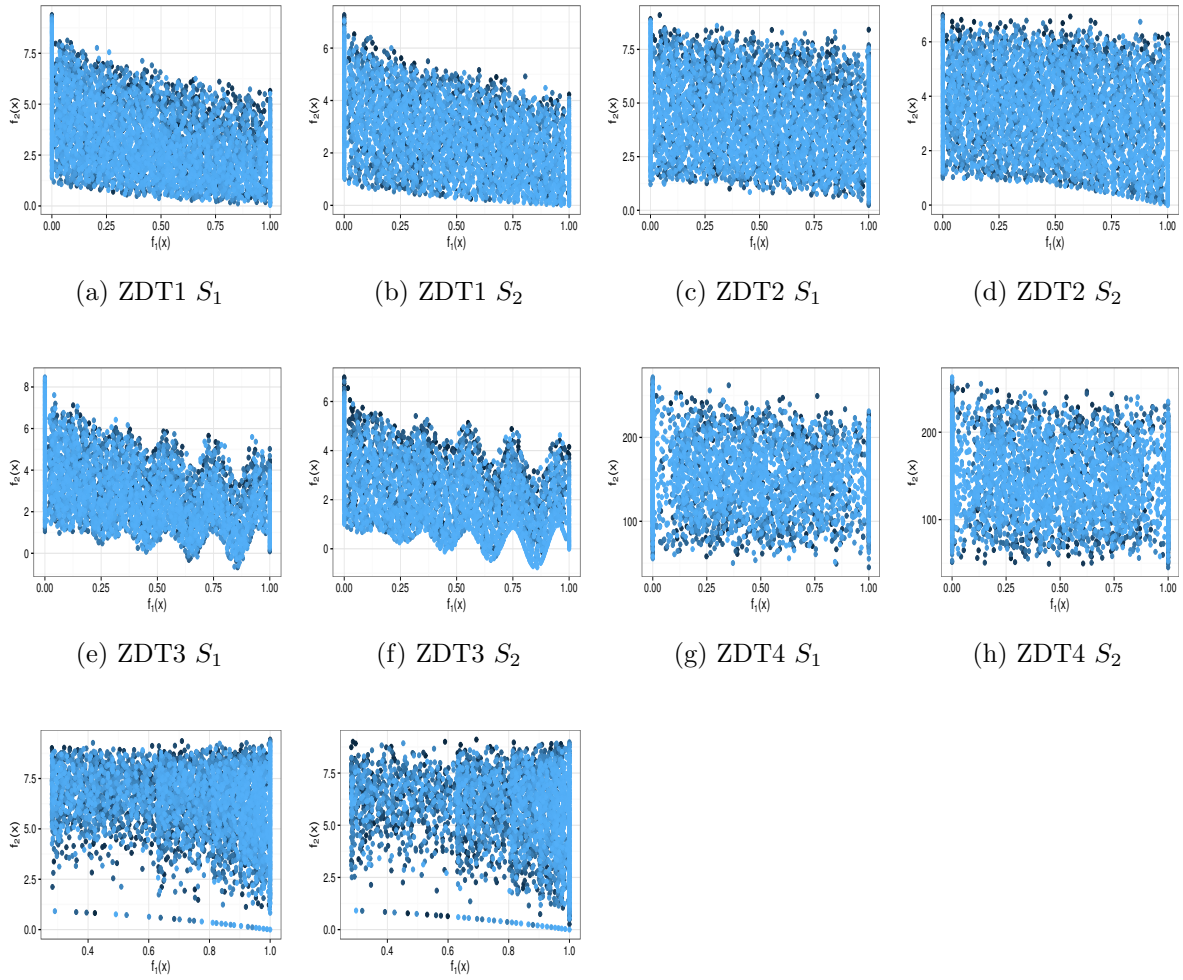


(a) WFG1 (b) WFG2 (c) WFG3

(d) WFG4 (e) WFG5 (f) WFG6

(g) WFG7 (h) WFG8 (i) WFG9

Figure 3.2: VEPSO calculated POFs for WFG problems



(a) ZDT1 S_1 (b) ZDT1 S_2 (c) ZDT2 S_1 (d) ZDT2 S_2

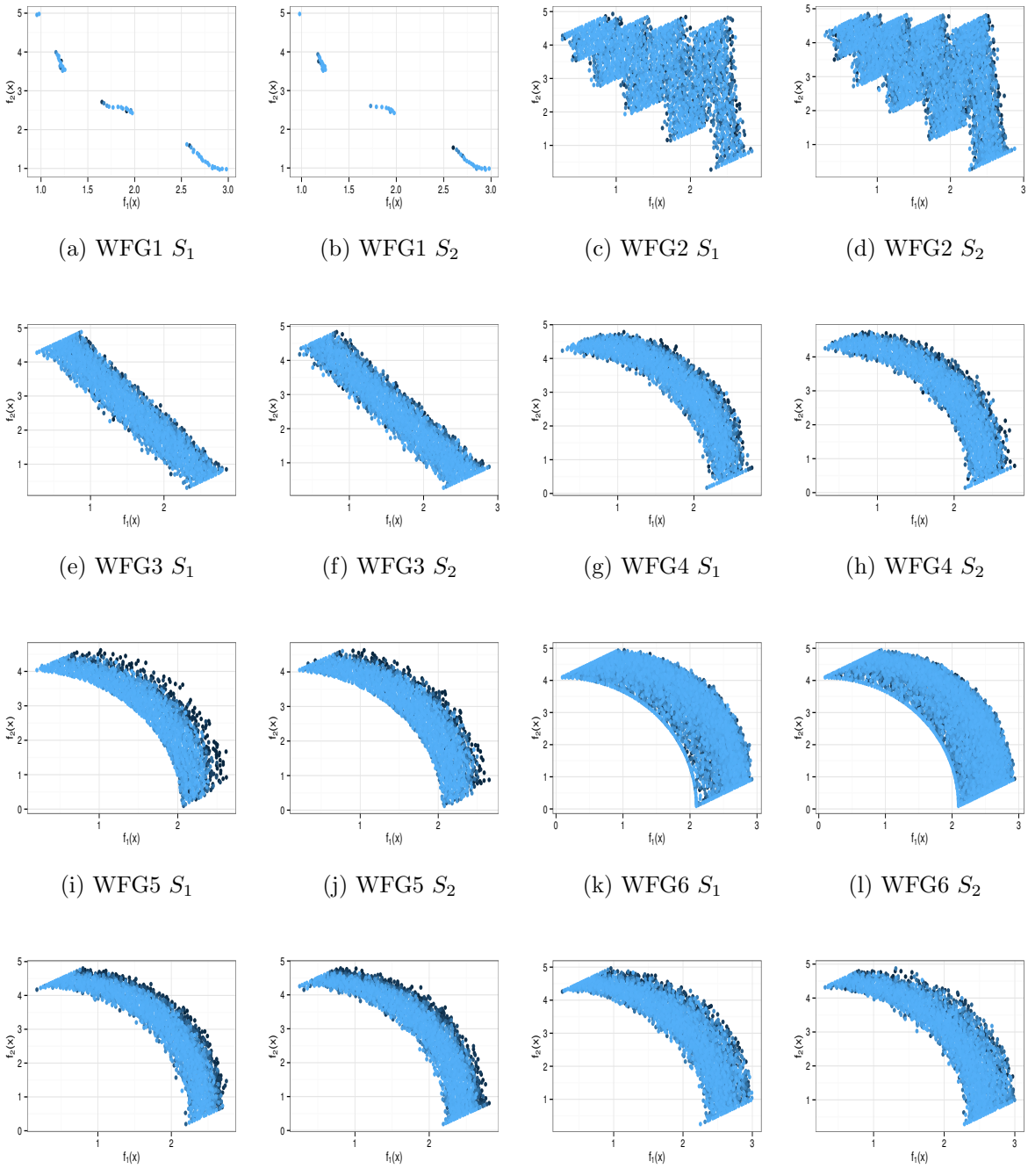
(e) ZDT3 S_1 (f) ZDT3 S_2 (g) ZDT4 S_1 (h) ZDT4 S_2

(i) ZDT6 S_1 (j) ZDT6 S_2

Figure 3.3: VEPSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations) for ZDT1 through ZDT6

3.2 Candidate Solution Dispersion

To further analyze the behavior of the particles, the candidate solution dispersion is measured quantitatively using three new measurements. These three measurements, when interpreted together, show whether the particles are exploiting by moving closer to and spreading along the POF, or exploring by moving further away from the POF.



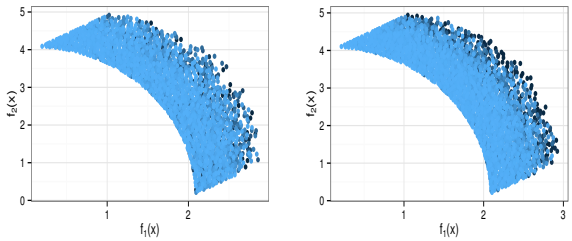
(a) WFG1 S_1 (b) WFG1 S_2 (c) WFG2 S_1 (d) WFG2 S_2

(e) WFG3 S_1 (f) WFG3 S_2 (g) WFG4 S_1 (h) WFG4 S_2

(i) WFG5 S_1 (j) WFG5 S_2 (k) WFG6 S_1 (l) WFG6 S_2

(m) WFG7 S_1 (n) WFG7 S_2 (o) WFG8 S_1 (p) WFG8 S_2

Figure 3.4: VEPSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations) for WFG1 through WFG8



(q) WFG9 S_1 (r) WFG9 S_2

Figure 3.4: VEPSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations) for WFG9

3.2.1 Measuring Dispersion

The first measurement tracks whether the candidate solutions are moving towards the POF, by measuring the average distance between each candidate solution and the point $[0, 0]$. The first measurement is formally defined as:

$$\mathcal{D}_l = \frac{1}{n_o} \sum_{k=1}^{n_o} |\mathbf{q}_k| \tag{3.1}$$

where $\mathbf{q}_k$ is candidate solution k with length $|\mathbf{q}_k|$. n_o is the number of candidate solutions. Large values for $\mathcal{D}_l$ indicate that the candidate solutions are far away from the POF, exploring more, while small values indicate that the candidate solutions are closer to the POF, exploiting more. A plot of $\mathcal{D}_l$ against the number of iterations shows whether the particles are moving closer to or further away from the POF. For algorithms that explore initially and exploit later, the expected result would be a decreasing value for $\mathcal{D}_l$ as the iterations increases. This measure is not intended to measure how closely the POF is being approximated, but rather only to give a general idea of the candidate solutions' movement in the objective space.

The second measurement quantifies the spread of the candidate solutions by measuring the average angle between lines drawn from the candidate solutions to the point $[0, 0]$. The second measurement is formally defined as:

$$\mathcal{D}_\theta = \frac{1}{n_o - 1} \sum_{k=1}^{n_o - 1} \alpha_k \tag{3.2}$$

where α_k is the angle between the k'th and $k+1$'th lines. Large values for $\mathcal{D}_\theta$ indicate that the candidate solutions are spread out more, covering a larger part of the objective space, presumably along or parallel to the POF. Small values indicate that the candidate solutions are less spread out.

The third measurement quantifies how even the spread of the candidate solutions are by measuring the standard deviation between the angles obtained from measurement two. The third measurement is formally defined as:

$$\mathcal{D}_\sigma = \sqrt{\frac{1}{n_o - 2} \sum_{k=1}^{n_o-1} (\alpha_k - \mathcal{D}_\theta)^2} \tag{3.3}$$

Small values for $\mathcal{D}_\sigma$ indicate that the candidate solutions are spread evenly, presumably along or parallel to the POF. Large values for $\mathcal{D}_\sigma$ indicate that the candidate solutions are not spread evenly and could be scattered in clusters. Both $\mathcal{D}_\theta$ and $\mathcal{D}_\sigma$ need to be viewed together to get an understanding of the candidate solutions' spread. Large values for $\mathcal{D}_\theta$ and small values for $\mathcal{D}_\sigma$ would be ideal, because it indicates a large even spread. Small values for $\mathcal{D}_\theta$ or large values for $\mathcal{D}_\sigma$ indicates a less optimal spread of the candidate solutions.

Note that the three new measurements differ from the existing measurements, such as generational distance [107] and spread [27], in that the existing measurements evaluate only the non-dominated solutions and not the candidate solutions of the current particle positions. This chapter focuses on exploring the behavior of the current candidate solutions and not the non-dominated solutions that have already been found.

3.2.2 Dispersion Analysis

The dispersion measurement results for VEPSO are presented in figures 3.5(a) through 3.5(ap). For each problem, the $\mathcal{D}_l$, $\mathcal{D}_\theta$ and $\mathcal{D}_\sigma$ values are plotted for iterations 1 through 500. For ZDT1 to ZDT4 and ZDT6, the POF lies in the area where $f_1(\mathbf{x}) \leq 1$ and $f_2(\mathbf{x}) \leq 1$. Based on where the POF lies, it can be concluded that, if the particles are exploring close to the POF, the $\mathcal{D}_l$ value should be close to or approaching 1. The figures show $\mathcal{D}_l$ values that are much larger in the majority of swarms. Seven of the 12 ZDT swarms had $\mathcal{D}_l$ values exceeding 4. For ZDT4 observe that the $\mathcal{D}_l$ values exceed 150

indicating extremely poor performance. The resulting POF also confirms this. After around iteration 50 little change in the $\mathcal{D}_l$ values can be noted for all the ZDT problems. The best $\mathcal{D}_l$ values were around 2 for ZDT1, ZDT2, and ZDT3.

For WFG1 through WFG9, the POF lies in the area where $f_1(\mathbf{x})$ is close to 4 for $f_2(\mathbf{x}) = 0$ and $f_2(\mathbf{x})$ is close to 2 for $f_1(\mathbf{x}) = 0$. Well-performing values for $\mathcal{D}_l$ should thus approach 2.5. Similar to the ZDT problems, after around iteration 50 no further improvement in $\mathcal{D}_l$ can be noted for any of the WFG problems except for WFG9. In the case of WFG9, improvement in $\mathcal{D}_l$ can be noted up to around iteration 300.

Increasing the number of iterations also led to no notable improvement in $\mathcal{D}_l$ values, confirming that the algorithm stagnates. The $\mathcal{D}_\theta$ and $\mathcal{D}_\sigma$ measurements showed no notable change as the number of iterations increased. The results indicate that the particles are not exploiting or refining the already found POF as the number of iterations increases. The exploration behavior of the VEPSO remains constant after around iteration 50 and does not focus on more exploitation.

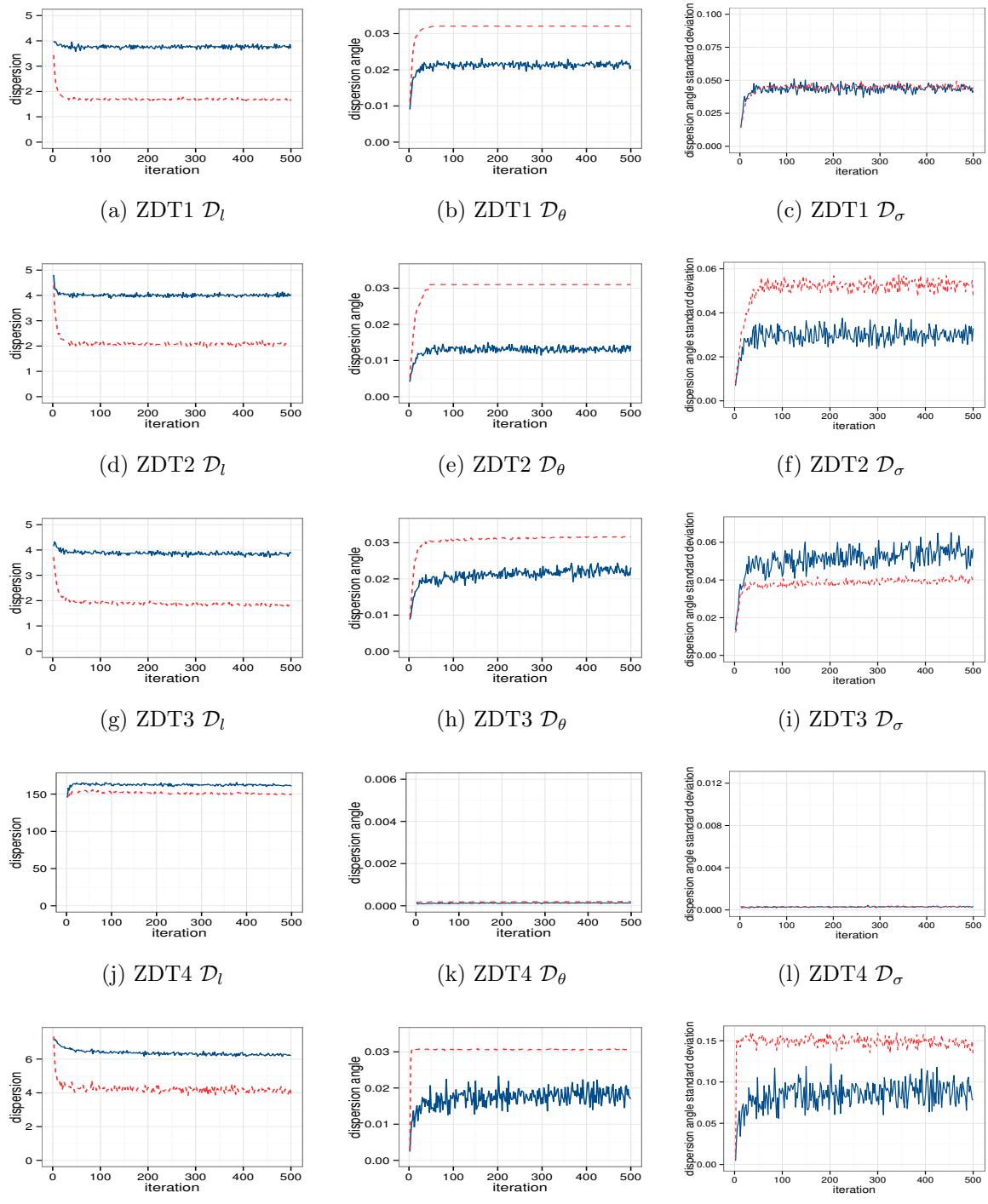
3.3 Increasing Exploitation

In this section, the archive-guided PSO is introduced in Section 3.3.1 to address the issues with VEPSO identified in the previous section. The exploration behavior of the newly introduced archive-guided PSO is then compared against VEPSO in Section 3.3.2.

3.3.1 Archive-guided Particle Swarm Optimization

The archive-guided PSO increases the pull towards the POF by adding an archive guide term to the velocity update equation. The archive guide is randomly weighted along with the global guide such that the social influence of the velocity update remains proportionally weighted against the inertia and cognitive terms. This prevents the social term from overwhelming the search process. The archive-guided PSO velocity update equation is formally defined as follows:

$$\begin{aligned}\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + \lambda_i c_2\mathbf{r}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t)) \\ + (1-\lambda_i)c_3\mathbf{r}_3(\hat{\mathbf{a}}_i(t) - \mathbf{x}_i(t))\end{aligned} \tag{3.4}$$



(a) ZDT1 $\mathcal{D}_l$ (b) ZDT1 $\mathcal{D}_\theta$ (c) ZDT1 $\mathcal{D}_\sigma$

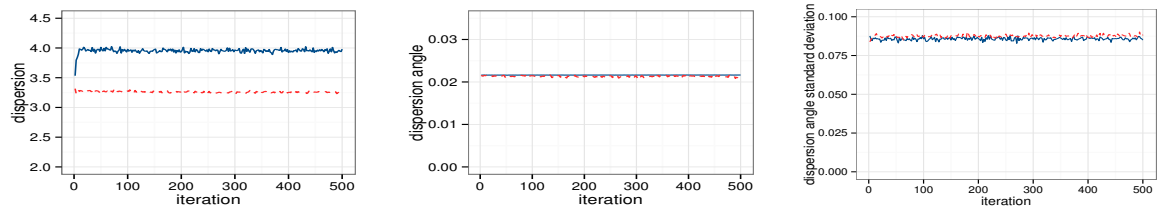
(d) ZDT2 $\mathcal{D}_l$ (e) ZDT2 $\mathcal{D}_\theta$ (f) ZDT2 $\mathcal{D}_\sigma$

(g) ZDT3 $\mathcal{D}_l$ (h) ZDT3 $\mathcal{D}_\theta$ (i) ZDT3 $\mathcal{D}_\sigma$

(j) ZDT4 $\mathcal{D}_l$ (k) ZDT4 $\mathcal{D}_\theta$ (l) ZDT4 $\mathcal{D}_\sigma$

(m) ZDT6 $\mathcal{D}_l$ (n) ZDT6 $\mathcal{D}_\theta$ (o) ZDT6 $\mathcal{D}_\sigma$

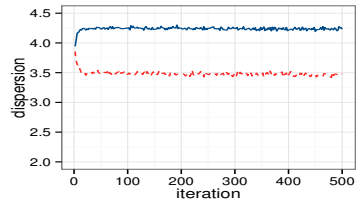
Figure 3.5: VEPSO dispersion results for ZDT and WFG problems (solid dark blue indicates values for S_1 and dashed light red indicates values for S_2)



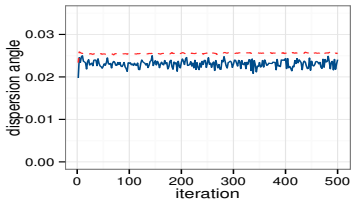
(p) WFG1 $\mathcal{D}_l$

(q) WFG1 $\mathcal{D}_\theta$

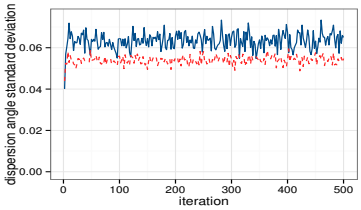
(r) WFG1 $\mathcal{D}_\sigma$



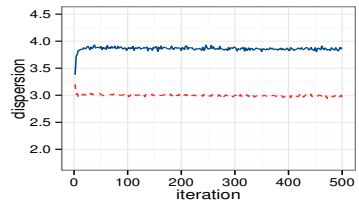
(s) WFG2 $\mathcal{D}_l$



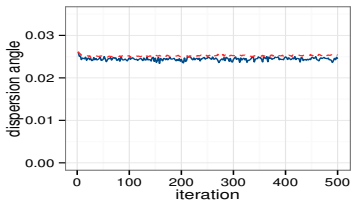
(t) WFG2 $\mathcal{D}_\theta$



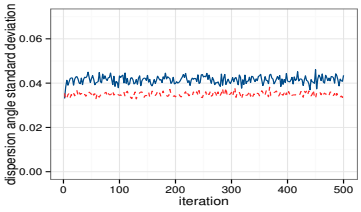
(u) WFG2 $\mathcal{D}_\sigma$



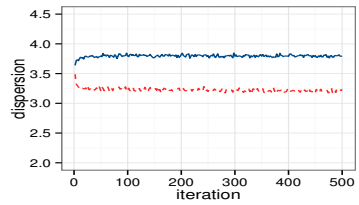
(v) WFG3 $\mathcal{D}_l$



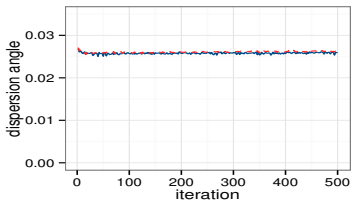
(w) WFG3 $\mathcal{D}_\theta$



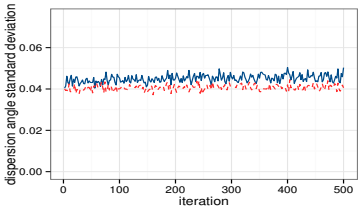
(x) WFG3 $\mathcal{D}_\sigma$



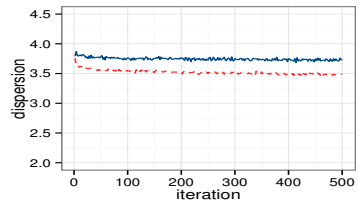
(y) WFG4 $\mathcal{D}_l$



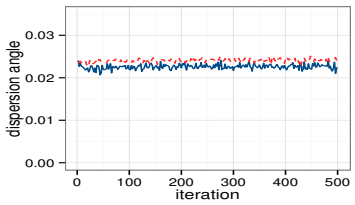
(z) WFG4 $\mathcal{D}_\theta$



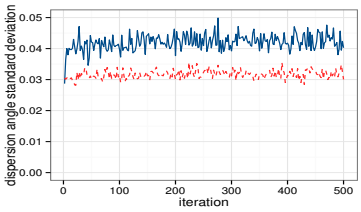
(aa) WFG4 $\mathcal{D}_\sigma$



(ab) WFG5 $\mathcal{D}_l$

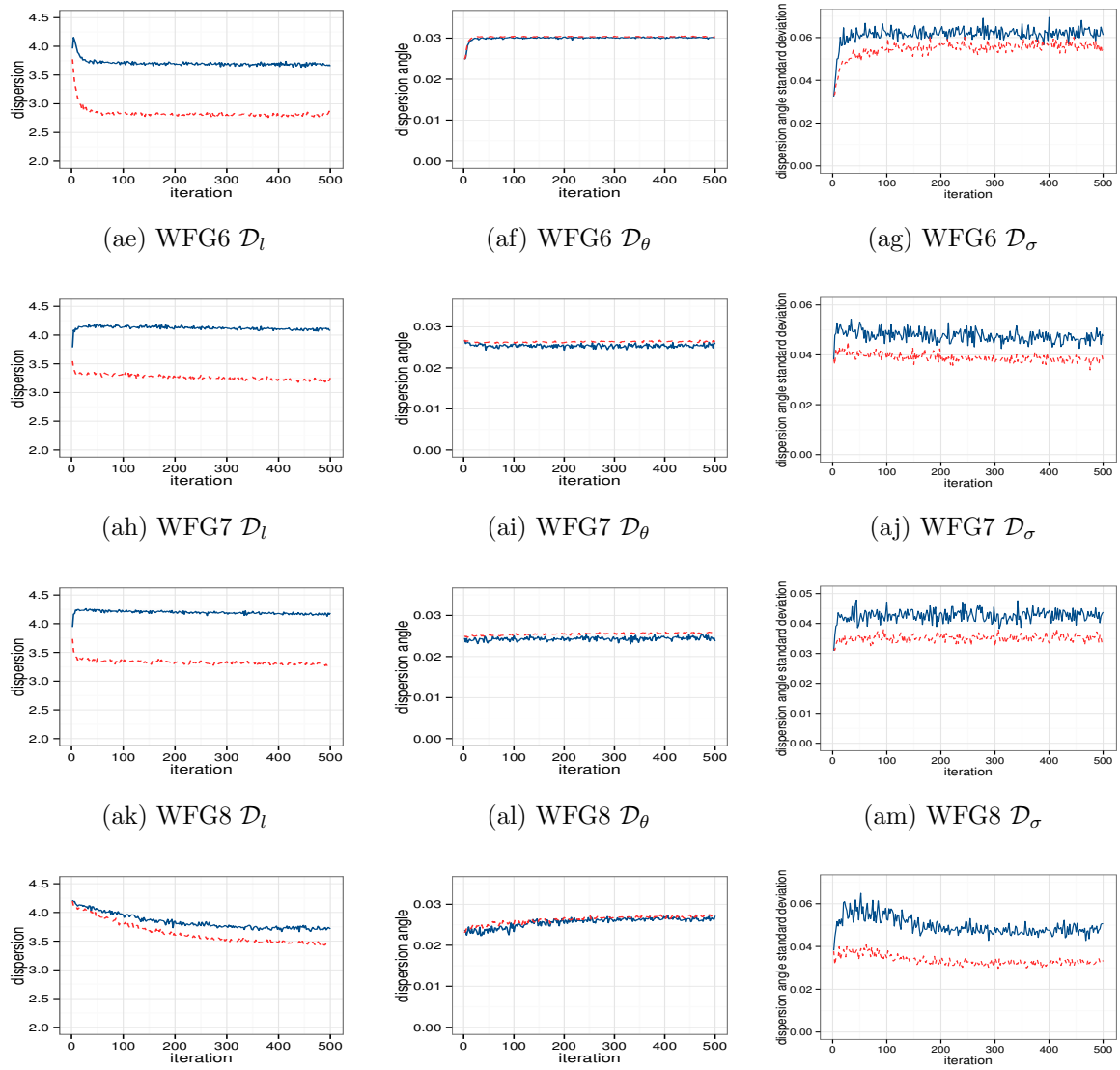


(ac) WFG5 $\mathcal{D}_\theta$



(ad) WFG5 $\mathcal{D}_\sigma$

Figure 3.5: VEPSO dispersion results for ZDT and WFG problems (solid dark blue indicates values for S_1 and dashed light red indicates values for S_2)



(ae) WFG6 $\mathcal{D}_l$ (af) WFG6 $\mathcal{D}_\theta$ (ag) WFG6 $\mathcal{D}_\sigma$

(ah) WFG7 $\mathcal{D}_l$ (ai) WFG7 $\mathcal{D}_\theta$ (aj) WFG7 $\mathcal{D}_\sigma$

(ak) WFG8 $\mathcal{D}_l$ (al) WFG8 $\mathcal{D}_\theta$ (am) WFG8 $\mathcal{D}_\sigma$

(an) WFG9 $\mathcal{D}_l$ (ao) WFG9 $\mathcal{D}_\theta$ (ap) WFG9 $\mathcal{D}_\sigma$

Figure 3.5: VEPSO dispersion results for ZDT and WFG problems (solid dark blue indicates values for S_1 and dashed light red indicates values for S_2)

where λ_i is a constant, random number, uniformly sampled from 0 to 1, $c_3 > 0$ is an acceleration constant, and $\mathbf{r}_3$ is a random vector with components uniformly sampled from 0 to 1, $\hat{\mathbf{a}}_i(t)$ is a randomly selected solution from the archive for particle i at iteration t. The archive term now allows for information exchange between the swarms

as both swarms submit and select from the archive. For the archive-guided PSO, $\hat{\mathbf{y}}_i(t)$ is the global best particle of the current swarm.

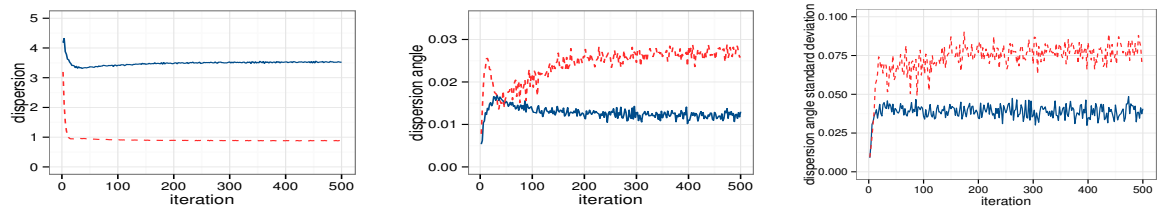
The addition of the archive term forces the information exchange to only exchange particle positions that improve one or more objectives, thus pulling the particles closer to the POF. On the other hand, the VEPSO random KTS allows the exchange of particle positions that attract particles away from the POF. The only requirement for a particle to be selected by the random KTS is that the particle must be well performing relative to the local objective of the randomly selected swarm.

3.3.2 Comparative Analysis

The archive-guided PSO algorithm was executed on the same ZDT and WFG test sets. The dispersion measurement results are presented in figures 3.6(a) through 3.6(ap). The results show a notable improvement in both the $\mathcal{D}_l$ and $\mathcal{D}_\theta \pm \mathcal{D}_\sigma$ measurements for a number of problems. For ZDT1 to ZDT3 and ZDT6 at least one of the swarms have a near ideal $\mathcal{D}_l$ value of 1. Also, note that the dispersion angle kept on improving for ZDT1 and ZDT2 up to around iteration 250. This indicates that the particles were more spread out along or parallel to the POF. The most promising result was noted for ZDT4: $\mathcal{D}_l$ kept on decreasing while $\mathcal{D}_\theta$ showed a corresponding increase. Take note that the $\mathcal{D}_\theta$ values were still considerably lower than any of the other results.

For the WFG test set the number of problems with $\mathcal{D}_l$ values lower than 3 increased to 7 from only one previously. It should also be noted that the results show that, typically, one of the swarms per problem performed much better while the other seemed to maintain poorer $\mathcal{D}_l$ values. Increasing the number of iterations did not improve the results.

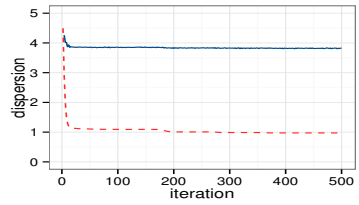
The candidate solution dispersion measurements indicate that the archive-guided PSO should perform better than the VEPSO as the search focused more on areas in the objective space close to the POF. To confirm this, the POFs were plotted. Figures 3.7(a) through 3.7(e) show the POF for the ZDT1 through ZDT6 problems and figures 3.8(a) through 3.8(i) show the POF for the WFG1 through WFG9 problems. It should be noted that archive-guided PSO failed to find a well-formed POF for ZDT4 in 12 of the 30 samples. A sample where a well-formed POF was found is shown here. It should



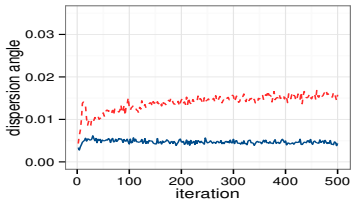
(a) ZDT1 $\mathcal{D}_l$

(b) ZDT1 $\mathcal{D}_\theta$

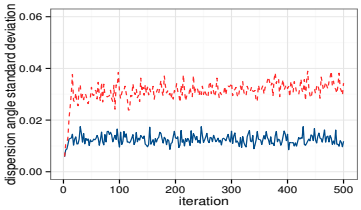
(c) ZDT1 $\mathcal{D}_\sigma$



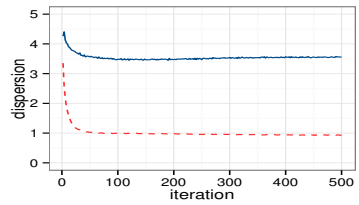
(d) ZDT2 $\mathcal{D}_l$



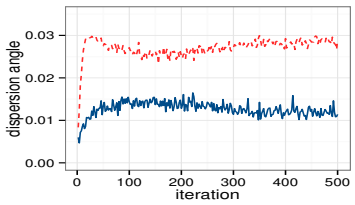
(e) ZDT2 $\mathcal{D}_\theta$



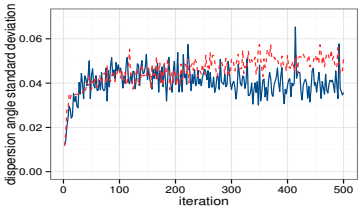
(f) ZDT2 $\mathcal{D}_\sigma$



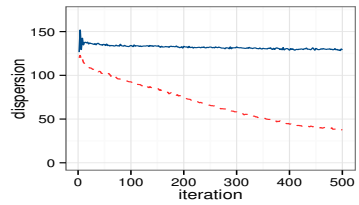
(g) ZDT3 $\mathcal{D}_l$



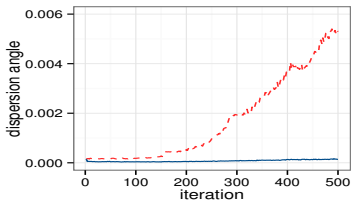
(h) ZDT3 $\mathcal{D}_\theta$



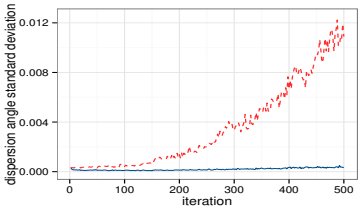
(i) ZDT3 $\mathcal{D}_\sigma$



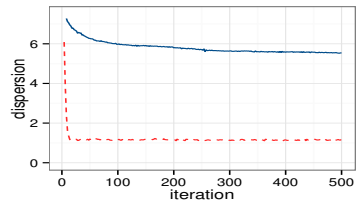
(j) ZDT4 $\mathcal{D}_l$



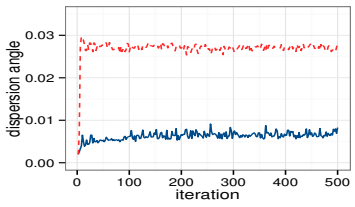
(k) ZDT4 $\mathcal{D}_\theta$



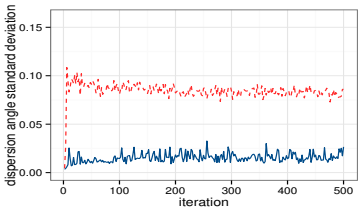
(l) ZDT4 $\mathcal{D}_\sigma$



(m) ZDT6 $\mathcal{D}_l$

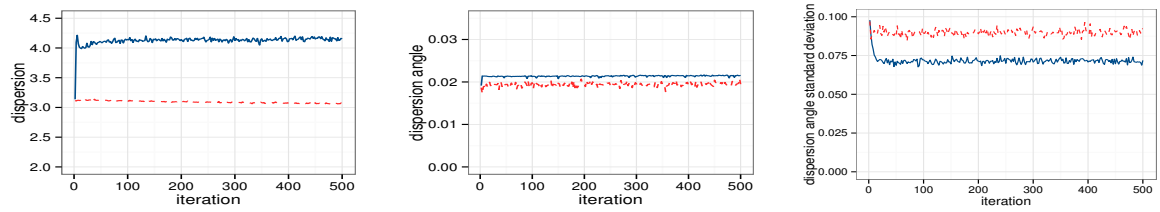


(n) ZDT6 $\mathcal{D}_\theta$

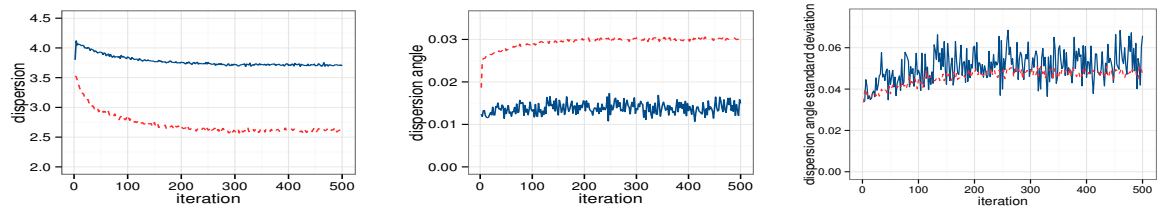


(o) ZDT6 $\mathcal{D}_\sigma$

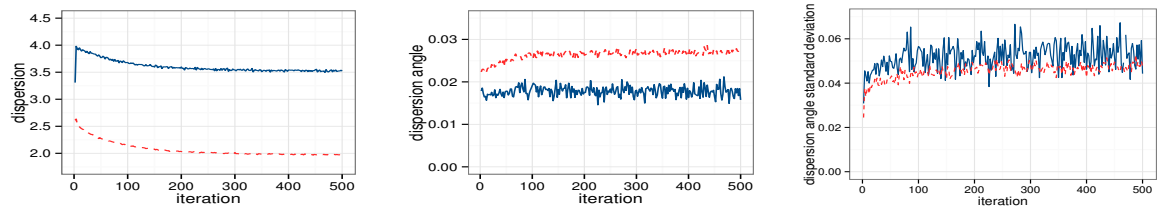
Figure 3.6: Archive-guided PSO dispersion results for ZDT and WFG problems (solid dark blue indicates values for S_1 and dashed light red indicates values for S_2)



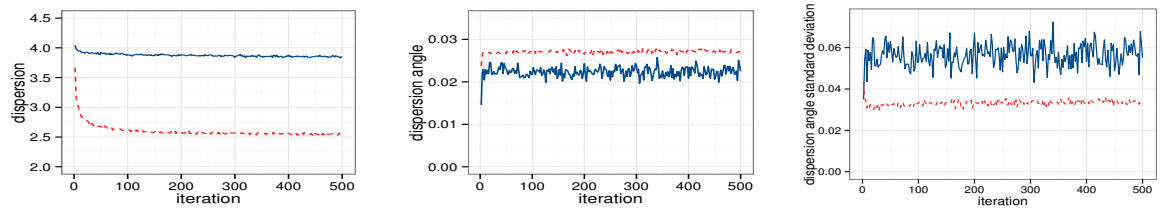
(p) WFG1 $\mathcal{D}_l$ (q) WFG1 $\mathcal{D}_\theta$ (r) WFG1 $\mathcal{D}_\sigma$



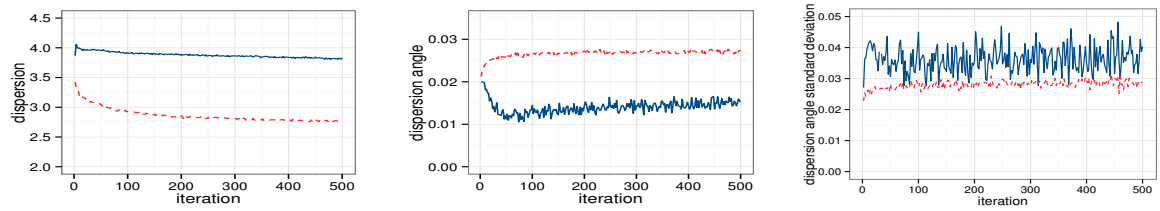
(s) WFG2 $\mathcal{D}_l$ (t) WFG2 $\mathcal{D}_\theta$ (u) WFG2 $\mathcal{D}_\sigma$



(v) WFG3 $\mathcal{D}_l$ (w) WFG3 $\mathcal{D}_\theta$ (x) WFG3 $\mathcal{D}_\sigma$

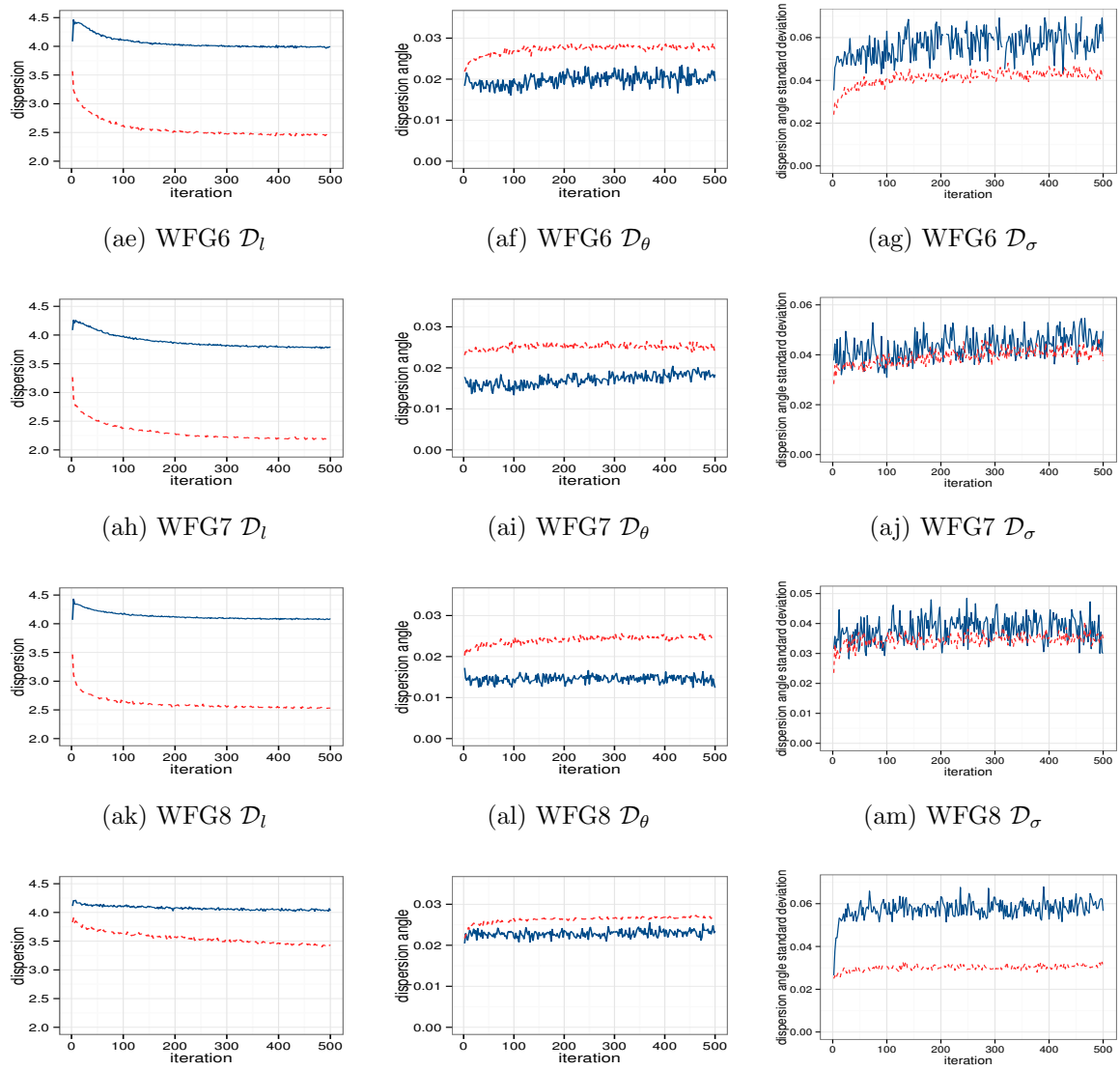


(y) WFG4 $\mathcal{D}_l$ (z) WFG4 $\mathcal{D}_\theta$ (aa) WFG4 $\mathcal{D}_\sigma$



(ab) WFG5 $\mathcal{D}_l$ (ac) WFG5 $\mathcal{D}_\theta$ (ad) WFG5 $\mathcal{D}_\sigma$

Figure 3.6: Archive-guided PSO dispersion results for ZDT and WFG problems (solid dark blue indicates values for S_1 and dashed light red indicates values for S_2)



(ae) WFG6 $\mathcal{D}_l$ (af) WFG6 $\mathcal{D}_\theta$ (ag) WFG6 $\mathcal{D}_\sigma$

(ah) WFG7 $\mathcal{D}_l$ (ai) WFG7 $\mathcal{D}_\theta$ (aj) WFG7 $\mathcal{D}_\sigma$

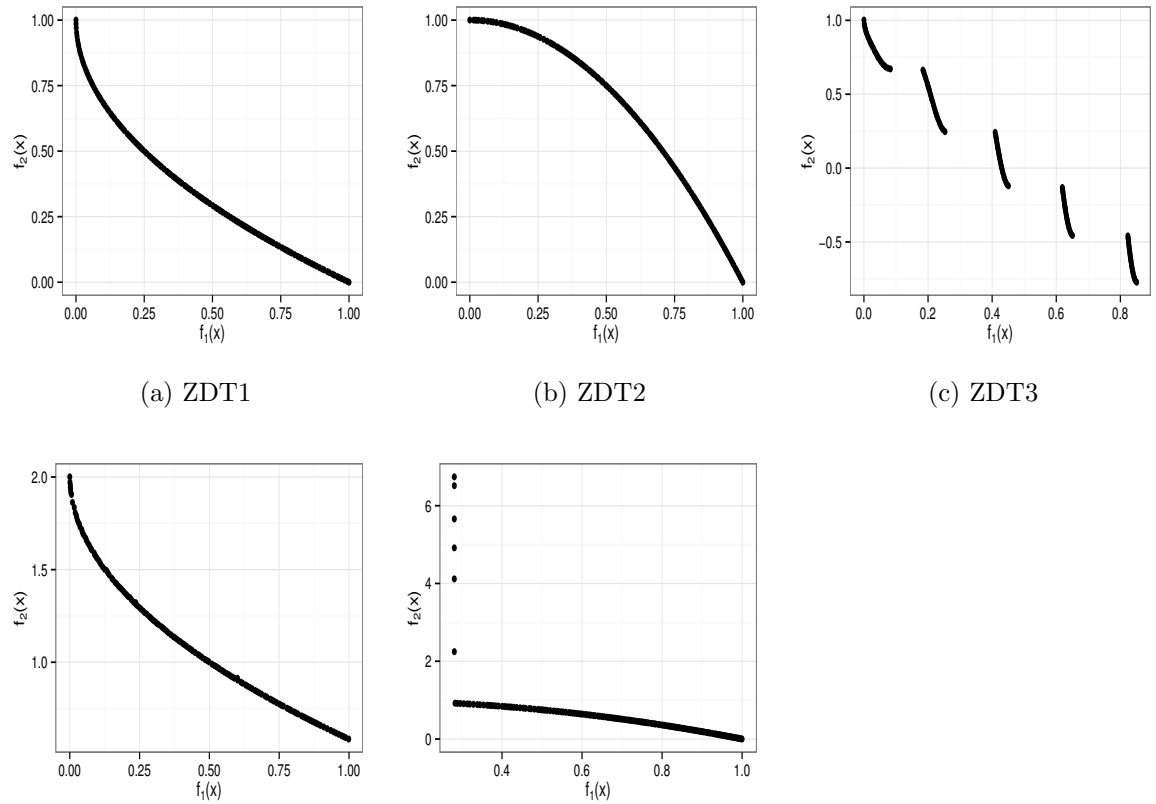
(ak) WFG8 $\mathcal{D}_l$ (al) WFG8 $\mathcal{D}_\theta$ (am) WFG8 $\mathcal{D}_\sigma$

(an) WFG9 $\mathcal{D}_l$ (ao) WFG9 $\mathcal{D}_\theta$ (ap) WFG9 $\mathcal{D}_\sigma$

Figure 3.6: Archive-guided PSO dispersion results for ZDT and WFG problems (solid dark blue indicates values for S_1 and dashed light red indicates values for S_2)

also be noted that VEPSO failed to find the POF for ZDT4 in all 30 samples.

Visual inspection of the POFs reveals that the POFs as found by the archive-guided PSO are much smoother for the majority of problems. WFG6 is a notable exception where the POF was slightly more jagged than the one found by VEPSO. The density



(a) ZDT1 (b) ZDT2 (c) ZDT3

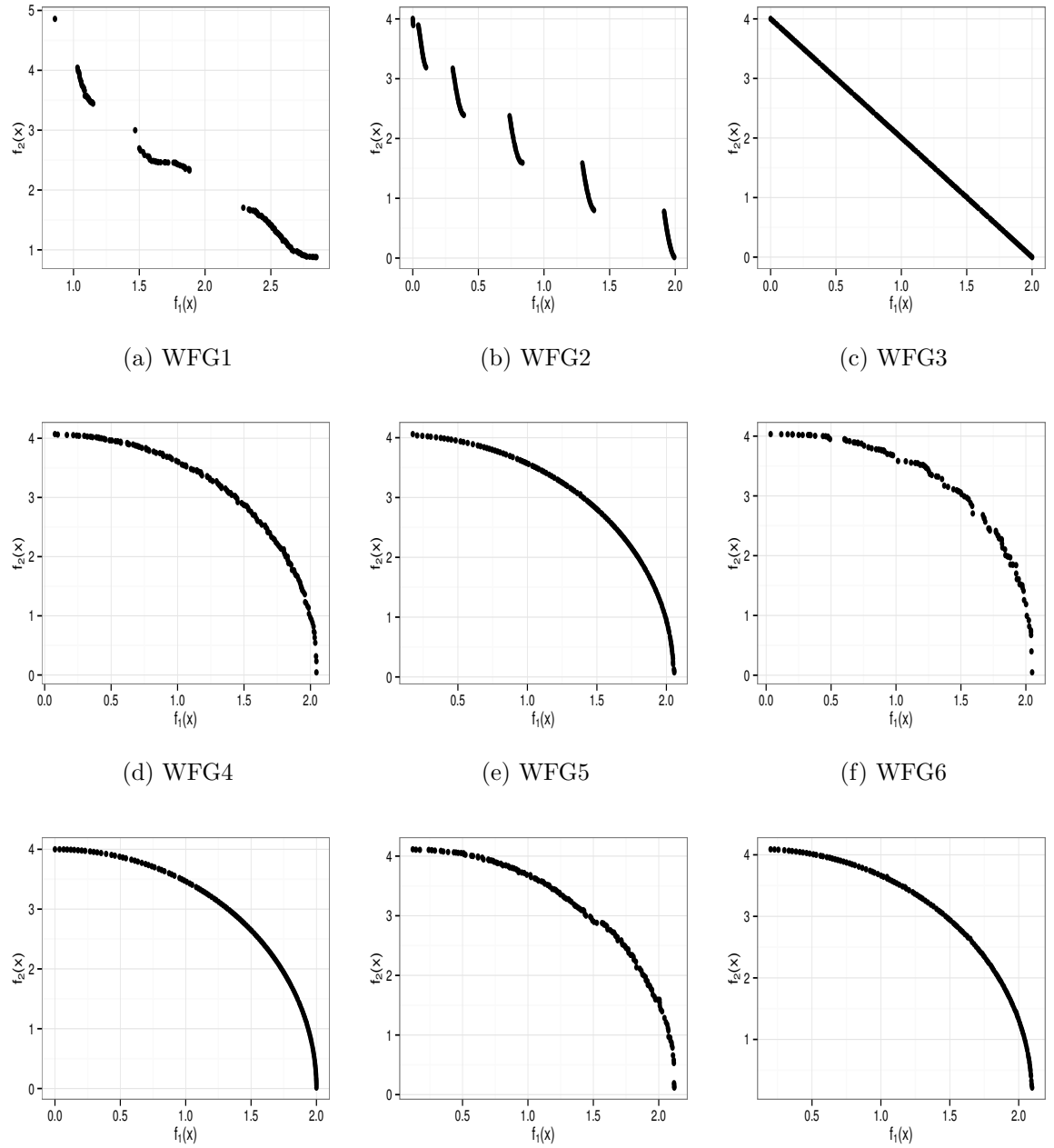
(d) ZDT4 (e) ZDT6

Figure 3.7: Archive-guided PSO calculated POFs for ZDT problems

of the POF is also much more consistent with archive-guided PSO than with VEPSO. WFG2, WFG3, WFG4, WFG7, and WFG8 are good examples where the found POF is much denser and smoother than the one found by VEPSO.

In order to understand why archive-guided PSO outperformed VEPSO, the candidate solution movement over the iterations are plotted. Figures 3.9(a) through 3.9(j) show the resulting plots for ZDT1 through ZDT6 and figures 3.10(a) through 3.10(r) show the resulting plots for WFG1 through WFG9.

Visual inspection shows that, for ZDT1, ZDT2, ZDT3, ZDT4, WFG2, WFG3, WFG5 and WFG7, one of the two swarms are much more focused on refining the POF. This correlates with our candidate solution dispersion measurements that indicated one of the

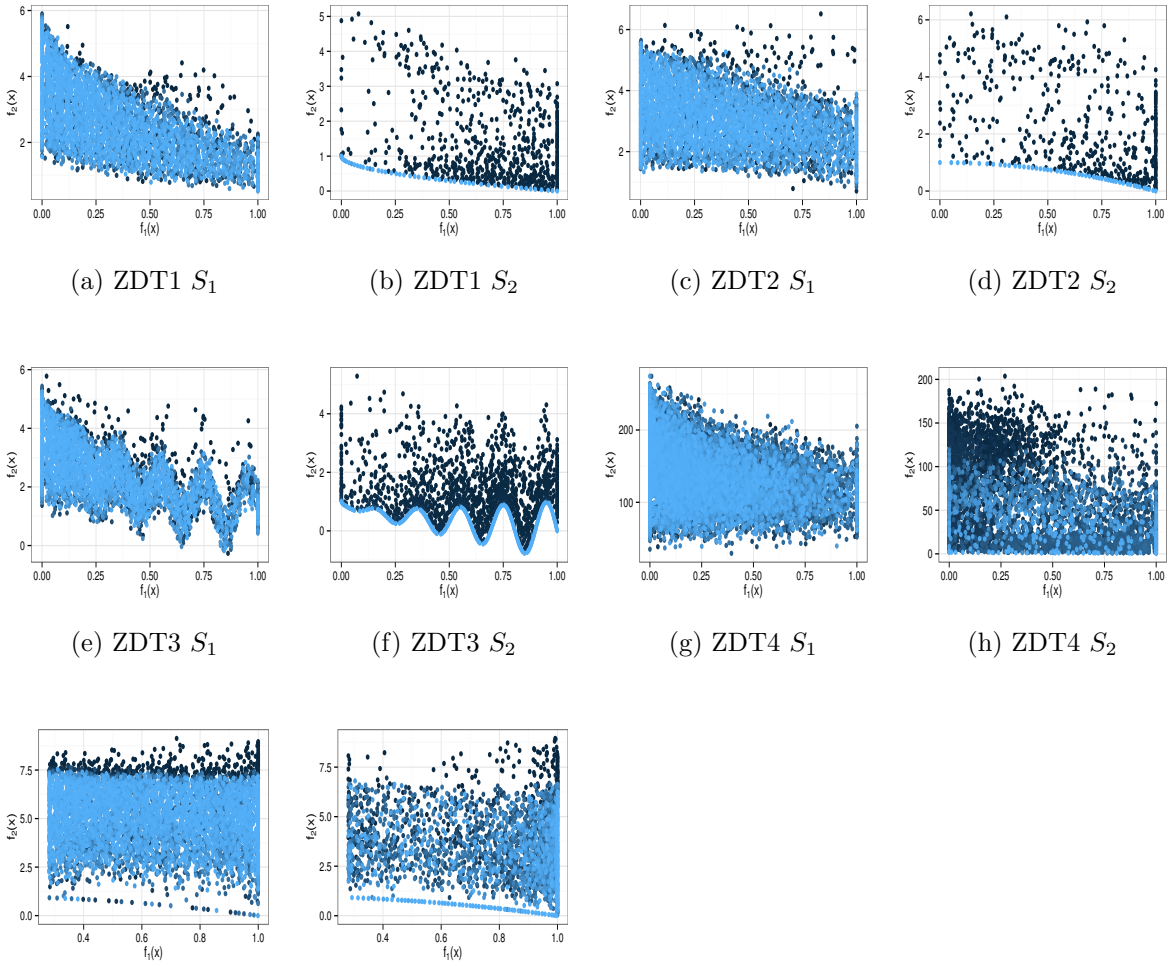


(a) WFG1 (b) WFG2 (c) WFG3

(d) WFG4 (e) WFG5 (f) WFG6

(g) WFG7 (h) WFG8 (i) WFG9

Figure 3.8: Archive-guided PSO calculated POFs for WFG problems



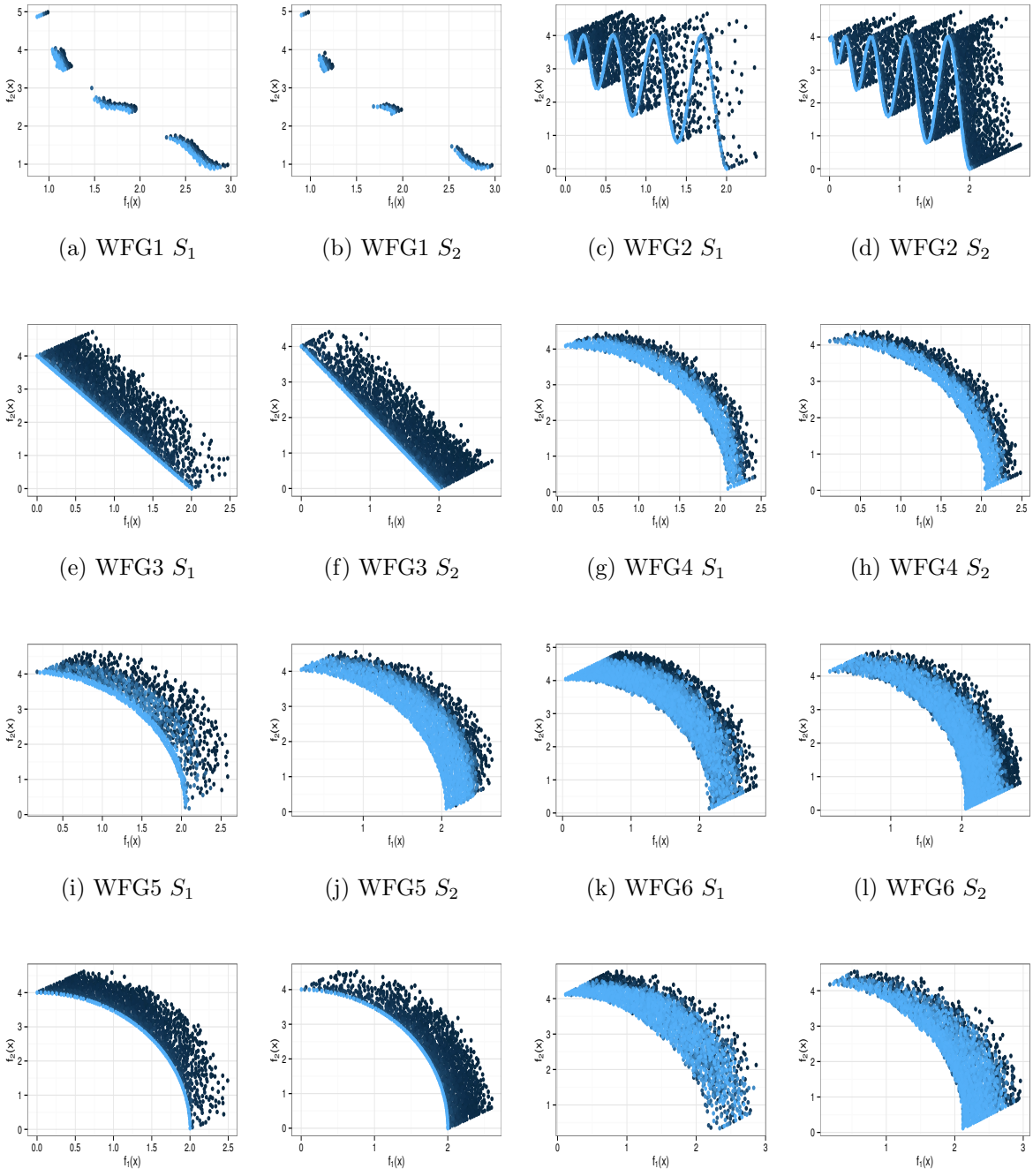
(a) ZDT1 S_1 (b) ZDT1 S_2 (c) ZDT2 S_1 (d) ZDT2 S_2

(e) ZDT3 S_1 (f) ZDT3 S_2 (g) ZDT4 S_1 (h) ZDT4 S_2

(i) ZDT6 S_1 (j) ZDT6 S_2

Figure 3.9: Archive-guided PSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations)

swarms were more focused on exploitation. In the case of WFG2, a sinus like wave pattern can be seen. The lower left edges of this pattern make up the POF. It is interesting to note that the search kept on exploring the dominated region between the POF sections for WFG2. WFG8 and WFG9 seemed to have proved most challenging as the search kept exploring and did not seem to converge towards the POF. The quality and density of the POFs found by the archive-guided PSO can be attributed to the fact that the



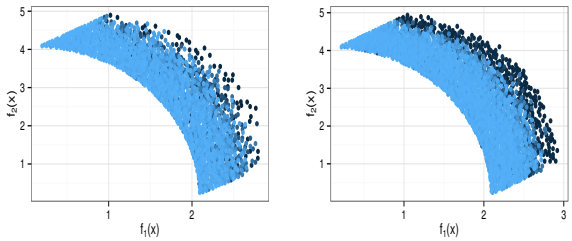
(a) WFG1 S_1 (b) WFG1 S_2 (c) WFG2 S_1 (d) WFG2 S_2

(e) WFG3 S_1 (f) WFG3 S_2 (g) WFG4 S_1 (h) WFG4 S_2

(i) WFG5 S_1 (j) WFG5 S_2 (k) WFG6 S_1 (l) WFG6 S_2

(m) WFG7 S_1 (n) WFG7 S_2 (o) WFG8 S_1 (p) WFG8 S_2

Figure 3.10: Archive-guided PSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations)



(q) WFG9 S_1 (r) WFG9 S_2

Figure 3.10: Archive-guided PSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations)

algorithm focuses much more on refining the POF than VEPSO.

3.4 Candidate Solution Movement Diversity

The candidate solution plots in the previous section showed that, for a number of swarms for both VEPSO and archive-guided PSO, the particles kept on exploring dominated regions of the objective space. The particles are clearly exploring more of the objective space at higher iterations than would have been desired. However, the plots did not show how large a region each of the particles explored. In this section, a new movement diversity measurement is introduced to understand more about the behavior of the individual particles. Section 3.4.1 describes the newly introduced movement diversity measure, followed by an analysis of the movement diversity results in Section 3.4.2.

3.4.1 Measuring Movement Diversity

The movement diversity measurement quantifies the degree of dispersion of a particle over a set number of iterations. The movement diversity measurement is based on the swarm diversity measurement [69, 111]. Instead of calculating the diversity over the position of the particles in the swarm, the movement diversity is calculated using the positions each particle occupied over the last n_t iterations. For this study, n_t was chosen

to be 20. The movement diversity measure is formally defined as:

$$\Delta d(S,t) = \frac{1}{n_s}\sum_{i=1}^{n_s}\frac{1}{n_t}\sum_{\Delta t=0}^{n_t-1}\sqrt{\sum_{j=1}^{n_x}(x_{ij}(t-\Delta t)-\bar{x}_{ij}(t))^2} \tag{3.5}$$

with

$$\bar{x}_{ij}(t) = \frac{1}{n_t}\sum_{\Delta t=0}^{n_t-1} x_{ij}(t-\Delta t) \tag{3.6}$$

where S is the swarm, t is the iteration, n_s is the number of particles in the swarm S, n_t is the number of iterations over which the movement diversity is calculated, n_x is the number of components of particle position $\mathbf{x}_i$, $x_{ij}(t)$ is the j'th component of the i'th particle position, and $\bar{x}_{ij}(t)$ is the average of the j'th component of the i'th particle position over the last n_t iterations.

The larger the movement diversity, the larger the region of the decision space that has been explored. The smaller the movement diversity, the smaller the region of the decision space that has been explored, indicating more exploitation. Ideally, the movement diversity should be large during the initial phase of the search, exploring more of the decision space, and smaller during the latter phase, exploiting more of the already found solutions.

Unlike the $\mathcal{D}_l$, $\mathcal{D}_\theta$ and $\mathcal{D}_\sigma$ candidate solution dispersion measurements, the movement diversity focuses on particle behavior in the decision space and not in the objective space. The movement diversity focuses exclusively on the movement, or path, of the particle, over n_t iterations. It is not a measure of how much the particles in the swarm are clustered or spread out.

3.4.2 Movement Diversity Analysis

Figures 3.11(a) through 3.11(e) show the movement diversity for ZDT1 through ZDT6. Figures 3.12(a) through 3.12(i) show the movement diversity for WFG1 through WFG9. Both VEPSO and archive-guided PSO swarms S_1 and S_2 are shown. Note that the remainder of this thesis uses the notation, algorithm:S_m, to refer to the swarm S_m for the corresponding algorithm.

For all the ZDT problems, the archive-guided PSO:S_2 swarm had the lowest movement diversity, and the archive-guided PSO:S_1 swarm maintained the second lowest

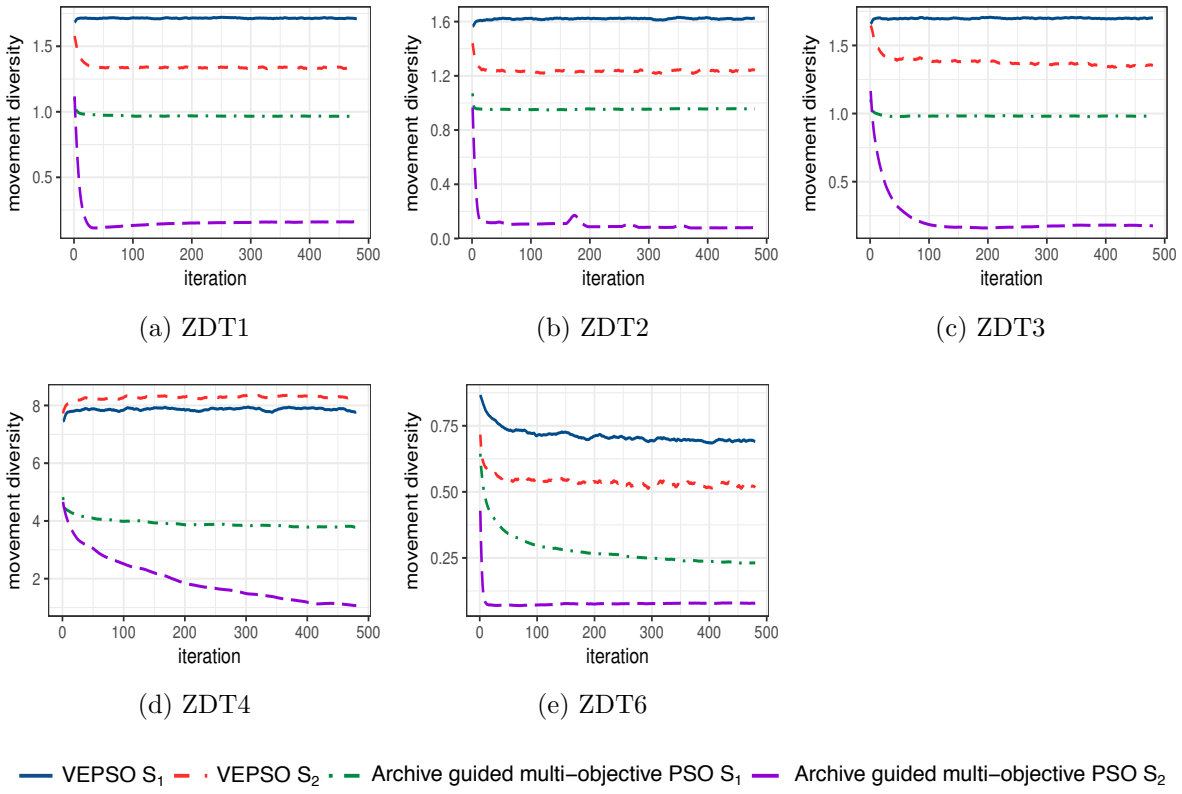


Figure 3.11: Movement diversity for VEPSO and archive-guided PSO

diversity. Archive-guided PSO:S_2 maintained a significantly lower movement diversity than VEPSO:S_2. A comparison of the movement diversity results with the candidate solution plots shows that the low movement diversity is due to the fact that the algorithm has found the POF and is only focusing on minor improvements to the already found front. This clearly shows that archive-guided PSO:S_2 is exploiting much more than the corresponding VEPSO swarm. It can be noted that the archive-guided PSO:S_2 swarm movement diversity did not reach zero and the particles kept on moving about towards the end of the search.

Note that both VEPSO and archive-guided PSO keep track of the nondominated solutions by storing them in an archive [2, 68]. When a new solution is added to the archive, all the solutions that are dominated by the new solution are removed from the archive. At any time, the archive contains the solutions that represent the best POF

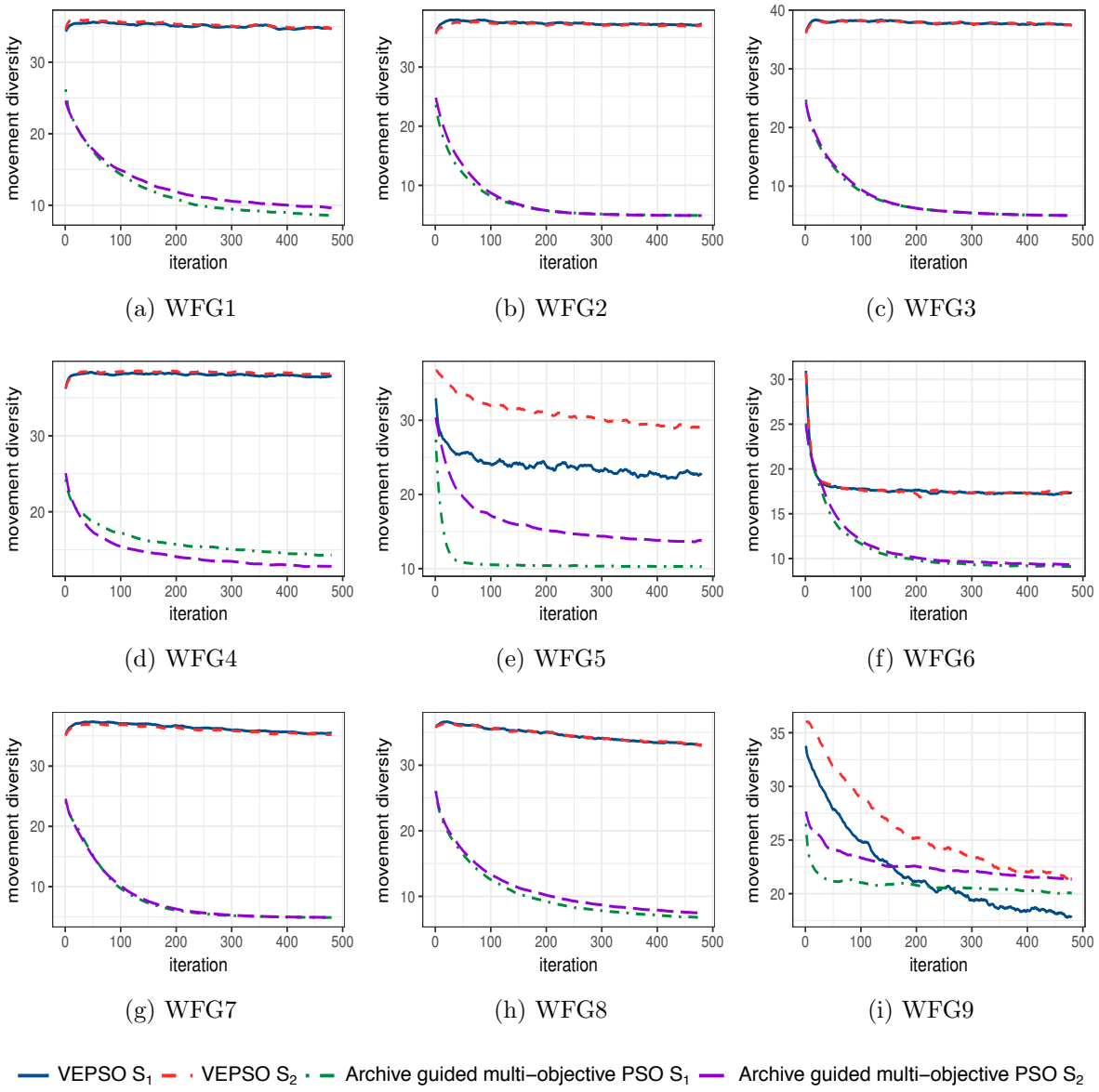


Figure 3.12: Movement diversity for VEPSO and archive-guided PSO

found thus far.

The continued movement of the archive-guided PSO:S_2 swarm can be attributed to the use of the randomly selected archive position in the velocity update equation of the archive-guided PSO. Effectively, the particles are moving on the POF attempting to

improve the POF. The large difference between the archive-guided PSO:S_1 and archive-guided PSO:S_2 movement diversities may be attributed to the construction of the ZDT problems. The ZDT function $f_1(\mathbf{x})$ is always defined as $f_1(\mathbf{x}) = x_1$ where $x_1 \in (0, 1)$. The values for $x_{2..n_x}$, where n_x is the number of dimensions of $\mathbf{x}$, is irrelevant for function $f_1(\mathbf{x})$. It is hypothesized that this causes a situation in swarm S_1 where x_1 is quickly optimized to a value of 0 while the $x_{2..n_x}$ values retain a higher diversity of non-zero values. These non-zero values for $x_{2..n_x}$ lead to worse $f_2(\mathbf{x})$ values that can only be improved through the archive guide, because both the personal best and local best positions consider only the $f_1(\mathbf{x})$ function value. Similarly, this could explain why S_2 manages to find the POF, seemingly so much easier when looking at the candidate solution plots because function $f_2(\mathbf{x})$ takes all the components of $\mathbf{x}$ into account. While the archive-guided PSO results show more clearly that there is a potential problem, the VEPSO algorithm may also be susceptible to this problem. A more thorough investigation will need to be conducted to confirm this hypothesis.

A similar trend can be noted for the movement diversity of S_1 and S_2 for the WFG problems. Only WFG5 and WFG9 resulted in non-similar trends for the movement diversity. The archive-guided PSO candidate solution plots for WFG5 show that the archive-guided PSO:S_1 particles were exploring closer to the POF than the archive-guided PSO:S_2 particles. The lower movement diversity for archive-guided PSO:S_1 can be attributed to the fact that the particles were busy exploiting more whereas the archive-guided PSO:S_2 particles were still exploring more. Similarly, the VEPSO:S_1 are exploring a slightly smaller area than VEPSO:S_2 when looking at the candidate solution plots and this is reinforced by the resulting movement diversity plot. Note, however, that for WFG5 both VEPSO and archive-guided PSO maintained a fairly high movement diversity indicating a high level of exploration were still taking place.

Similar to WFG5, for WFG9 the movement diversity for both VEPSO and archive-guided PSO swarms indicate that a high level of exploration were still taking place. From the candidate solution plot for archive-guided PSO:S_2 the expectation is set that archive-guided PSO:S_2 would have the lowest movement diversity because the candidate solution plot indicate the swarm is exploiting slightly more than the other swarms for WFG9. This is however not the case, and VEPSO:S_1 maintained a lower movement diversity.

This may be due to the complexity of the WFG9 problem. Further investigation is required to confirm this.

For WFG2, WFG3, and WFG7 the movement diversity shows that the archive-guided PSO particles are moving significantly less than the corresponding VEPSO particles. The candidate solution plots confirm that this is again due to exploitation of the already found VEPSO.

In general, the movement diversity measurement results show that the VEPSO particles, for both swarms, were exploring much more than the corresponding archive-guided PSO particles. The lower movement diversity indicates that the algorithm focuses more on exploitation of the already found well-performing areas of the decision space. However, there are exceptions where a lower movement diversity was noted while the candidate solution plot indicated that the particles were still exploring far away from the VEPSO.

3.5 Summary

This chapter presented an explorative analysis in low-dimensional objective space of the VEPSO algorithm's search behavior. The use of two objective problems allowed for the visualization of the candidate solutions in the objective space. The visualization of the candidate solutions aided in investigating the exploration-exploitation behavior of VEPSO. The ZDT and WFG test sets were used as benchmark problems for this analysis. The obtained POFs for each of the problems were plotted. In many cases the POFs were jagged, and the solutions were not evenly spread along the POF.

A hypothesis for VEPSO's poor performance was presented. In order to investigate further, the movement of the candidate solutions was plotted. The plots showed that VEPSO continues to explore and does not focus on refining (or exploiting) the already found POF, confirming the hypothesis. Three new measurements were introduced to quantify the dispersion of the particles in the objective space. These measurements focus on measuring the candidate solutions rather than the Pareto optimal solutions.

The archive-guided PSO was introduced to address the lack of POF refinement that was noted in the VEPSO results. The archive-guided PSO was shown to spend more time refining the POF than VEPSO. The resulting POFs reflected the performance im-

provement when compared to VEPSO. The candidate solution movement plots reflected the fact that more time was spent refining solutions near or on the POF.

A new quantitive measurement was introduced to measure the particle movement diversity. The movement diversity measurement reveals the extent of areas in the decision space that particles are exploring. Larger movement diversities indicate that the particles are focusing more on exploration and less on exploitation. The movement diversity was calculated for VEPSO and archive-guided PSO for all the ZDT and WFG problems. The results confirmed that the archive-guided PSO focuses more on exploitation when compared to VEPSO. Additionally, a potential weakness in the archive-guided PSO algorithm was noted for the ZDT problems where swarm S_1 tended to perform poorly when compared to swarm S_2. Further investigation needs to be conducted to determine if there is truly a problem and if the VEPSO algorithm may be susceptible to a similar problem.

Non-zero movement diversities were noted for all the problems. In the case of the archive-guided PSO, this can be attributed to the use of a randomly selected archive guide in the velocity update equation.

In general, it was shown that the archive-guided PSO algorithm outperformed the VEPSO algorithm due to the increased exploitation.

The next chapter presents an investigation of the effect that various archive implementations have on VEPSO's performance.

Chapter 4

Vector Evaluated Particle Swarm Optimization Archive Management

"A retentive memory may be a good thing, but the ability to forget is the true token of greatness."

Elbert Hubbard (1856 - 1915)

The previous chapter investigated the exploration behavior of VEPSO. The exploration behavior analysis showed that VEPSO does not exploit the known well-performing regions of the search space. The lack of exploitation, in turn, leads to poor quality POFs. The POF is constructed using the non-dominated solutions found in VEPSO's archive. When introducing VEPSO, Parsopoulos and Vrahatis [84, 85] did not describe the management of the VEPSO archive in detail. This chapter presents an analysis of the effect of various archive management strategies (AMSs) on the performance of VEPSO. The main objective is to identify the influence of various AMSs on the resulting POFs. To achieve this objective, various AMSs are discussed followed by a POF diversity and performance analysis. A weakness in two of the diversity measures used during the POF diversity analysis is identified and analyzed.

4.1 Introduction

Various studies have been conducted to analyze the performance of the VEPSO algorithm [47, 74]. These studies use quantitative performance measures to compare the performance of the VEPSO algorithm against other well-known MOO algorithms. Helbig and Engelbrecht [50] investigated the effect that the AMS has on performance in the context of dynamic environments using the dynamic vector evaluated particle swarm optimization (DVEPSO) algorithm. While studying the performance of a newly introduced KTS, Harrison *et al.* [47] compared the VEPSO algorithm against the optimized multi-objective particle swarm optimization (OMOPSO) [92] and speed-constrained multi-objective particle swarm optimization (SMPSO) [79] algorithms. In contrast to OMOPSO and SMPSO, VEPSO's AMS is not well defined.

This chapter presents and evaluates eight different bounded AMSs. One of the evaluated AMSs, the hypersurface contribution AMS, is also introduced in this chapter. In addition to the different AMSs, the effect that the size limit of the archive has on the POF's diversity is also evaluated. A critical weakness in Schott's [96] spacing and Goh and Tan's [44] distribution measures are identified and analyzed. The crowding distribution measure is introduced to address the weakness of the aforementioned diversity measures. A performance evaluation, using the IGD measure to compare the various AMSs, is used to identify the best performing AMS for VEPSO.

The remainder of this chapter is structured as follows: Section 4.2 provides a more detailed overview of archive management and the various AMSs used throughout this study. A POF diversity analysis is presented in Section 4.3. The diversity measure results contained some discrepancies. Section 4.4 presents and validates a hypothesis as to why some of the measures may yield misleading results. Section 4.5 presents a performance analysis of the different AMSs and their influence on the resulting POFs. Finally, Section 4.6 provides a summary of the findings of this chapter.

4.2 Archives

While several different AMSs have been developed for MOO algorithms [3, 47, 79, 92], it is well known that more research in archive management is needed to further the

performance of multi-objective optimization algorithms [3, 110].

This section discusses the basics of archive management in Section 4.2.1 and the various deletion approaches used by bounded AMSs in Section 4.2.2.

4.2.1 Archive Management

Archives store non-dominated solutions found during the search process. Typically, an optimization algorithm will submit solutions to be stored in the archive at the end of each iteration. A new solution is only admitted into the archive if none of the existing solutions in the archive dominate the new solution. When a new solution is admitted, all the solutions in the archive that are dominated by the new solution are removed. Archives can be bounded or unbounded.

Bounded archives limit the number of solutions that can be kept in the archive, $\mathcal{A}$. The archive limit can be preset or dynamically adapted [72]. Once the archive size limit is reached, new solutions can only be admitted if existing solutions in the archive are removed. If no solutions in the archive are dominated by the new solution, the solution can either not be admitted [58], or a deletion approach (also known as a removal function or pruning method) can be used to remove a solution from the archive [3, 68].

Algorithm 4 provides pseudo-code for a bounded AMS with a deletion approach to maintain the archive size.

4.2.2 Deletion Approaches

The experimental work presented in this chapter made use of a bounded AMS with the following eight deletion approaches:

Crowding Distance

First introduced by Deb *et al.* [29], crowding distance was introduced to estimate the density of solutions surrounding a particular solution. The crowding distance of a Pareto-optimal solution is calculated as the average distance between the two points on either side of the specified point along each of the objectives. When the archive is full, the solution with the smallest crowding distance is removed from the archive.

Algorithm 4 Bounded archive management strategy with a deletion approach

```
for each particle i = 1, ..., n_s do
  if x_i is not dominated by any solution in the archive
      and x_i is not similar to any solutions in the archive then
    if x_i dominates any solution a in the archive then
      Remove all archive solutions that is dominated by x_i;
      Add x_i to the archive;
    else
      Add x_i to the archive;
      if archive size exceeds archive size limit then
        Select a solution a from the archive using a deletion approach;
        Remove solution a from the archive;
      end if
    end if
  end if
end for
```

It should be noted that various enhancements to the crowding distance have been proposed to improve the accuracy of the solution density estimation when more than two objectives are used [38, 70, 103].

Distance

Bart-Beielstein *et al.* [3] proposed a relative distance based measure. The relative distance is defined as the sum of the Euclidean distance between the specified solution and all other solutions in the archive. The distance deletion fitness is defined as:

$$f_{del,u} = \sum_{l=1,l\neq u}^{|\mathcal{A}|} \left(\sqrt{\sum_{m=1}^{n_m} \left(\frac{a_{um} - a_{lm}}{\max(a_m) - \min(a_m)} \right)^2} \right)^{-1} \tag{4.1}$$

with $\mathbf{a} \in \mathcal{A}$. When the archive is full, the solution with the highest deletion fitness, $f_{del,u}$, is removed from the archive.

Nearest Neighbor

Harrison *et al.* [47] used a nearest neighbor deletion approach. The nearest neighbor is calculated similarly to the distance approach. However, instead of computing the distance between each pair of solutions in the archive, only the n_n-closest neighboring solutions, calculated using the Euclidean distance, are used in the calculation.

Random

The random deletion approach removes a randomly selected solution from the archive.

Hypervolume Contribution

Zitzler and Thiele [117] proposed a measure that, in the two-dimensional case, calculates the area of the rectangle defined by the points $(0, 0)$ and $\mathbf{a} = (a_1, a_2)$. The proposed measure came to be known as the hypervolume measure.

Jiang and Cai [57] introduced an archive based on the idea of the minimum reduction in hypervolume. Each solution in the archive is ranked according to the amount that the solution contributes to the hypervolume measure. When the archive is full, the solution that contributes the least to the hypervolume is removed from the archive.

Calculating the hypervolume is computationally expensive [40]. Bader and Zitzler [1] proposed a Monte Carlo simulation based approach for efficient estimation of the hypervolume measure.

Hypersurface Contribution

Based on the idea of hypervolume contribution, a hypersurface contribution based archive is proposed. The hypersurface contribution for a solution in the archive $\mathbf{a}$ is calculated as the hypersurface area contributed to the POF by the solution. The extreme solutions $\mathbf{a}_1$ and $\mathbf{a}_4$ have hypersurface contributions set to ∞. Figure 4.1 depicts the hypersurface contribution calculation $|a_{12} - a_{22}| + |a_{31} - a_{21}|$ for a non-dominated solution $\mathbf{a}_2$ for the two objectives case.

Similar to the hypervolume contribution archive, when the hypersurface archive is full, the solution that contributes the least to the hypersurface is removed from the

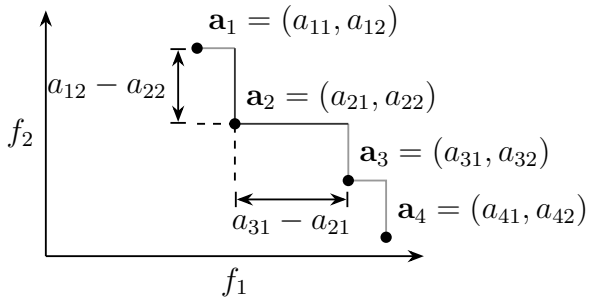


Figure 4.1: Hypersurface contribution calculation

archive.

ϵ-dominance

Sierra and Coello Coello [92] introduced the OMOPSO algorithm using an ϵ-dominance based archive to store the non-dominated solutions.

Laumanns [71] termed ϵ-dominance as defined in definition 2.6 as multiplicative ϵ-dominance. An alternative additive definition defines the ϵ-dominance relation as $\epsilon_m + f_m(\mathbf{x}_1) < f_m(\mathbf{x}_2)$.

Mostaghim and Teich [78] found that ϵ-dominance based archives in multi-objective PSO outperformed other archiving techniques in some cases.

Note that the ϵ-dominance based archive, in contrast to the other archives presented here, do not impose a limit on the number of entries. The ϵ_m parameters must be tweaked until the algorithm generates the desired number of solutions.

Adaptive Grid

Coello Coello and Lechuga [18] proposed subdivision of the objective space into hypercubes. Between 30 and 50 divisions were proposed as being optimal. Bart-Beielstein *et al.* [3] proposed an adaptive grid variant based on the idea proposed by Coello Coello and Lechuga. The adaptive grid proposed by Bart-Beielstein *et al.* resizes the hypercubes every iteration. The edge lengths of the hypercube are determined as:

$$\check{e}_m = \frac{\ddot{c}\,(\max(a_m) - \min(a_m))}{|\mathcal{A}|} \tag{4.2}$$

with $\mathbf{a} \in \mathcal{A}$, where $m \in [1, n_m]$ is the dimension of the hypercube edge and $\ddot{c} \in [0, 1]$ is a selection pressure. Bart-Beielstein *et al.* set $\ddot{c}$ to 0.5. The hypercube grid coordinate, $\check{\mathbf{c}}$, for an archive solution $\mathbf{a}$ is calculated as:

$$\check{c}_m = \left\lfloor \frac{a_m - \min(a_m)}{\check{e}_m} \right\rfloor \tag{4.3}$$

The deletion fitness is defined as:

$$f_{del,u} = |H|^2 \tag{4.4}$$

where $|H|$ is the number of solutions with the same hypercube grid coordinate. When the archive is full, a solution with the highest deletion fitness, $f_{del,u}$, is randomly selected and removed from the archive. The number of divisions is defined as $\left\lfloor \frac{n_s}{\ddot{c}} \right\rfloor$. For a swarm with 50 particles with $\ddot{c} = 0.5$ this results in $\left\lfloor \frac{50}{0.5} \right\rfloor = 100$ hypercube divisions per dimension.

4.3 Archive's Influence on Diversity

This section presents an evaluation of the impact that the archive size limit and choice of AMS have on VEPSO's resulting POF's diversity. A discussion on how to measure the influence on diversity is given in Section 4.3.1, followed by a diversity analysis in Section 4.3.2.

4.3.1 Measuring Pareto-optimal Front Diversity

The analysis presented in this section made use of the four POF diversity measures presented in Section 2.4.2, namely maximum spread by Zitzler [116], distribution by Goh and Tan [44], spacing by Schott [96], and spread by Deb *et al.* [29].

The diversity was measured for VEPSO (Random) and VEPSO (PCXA) with archive size limits, 50, 150, and 500, in combination with the AMSs presented in the previous section. Each archive size limit and deletion approach combination were tested using the ZDT test set. The results presented were taken over 30 independent runs of 2000 iterations of each algorithm for each problem. The inertia weight, w, was set to 0.729844, and the acceleration constants, c_1 and c_2, were set to 1.49618, n_n was set to 2 for the nearest neighbor deletion approach.

4.3.2 Pareto-optimal Front Diversity Analysis

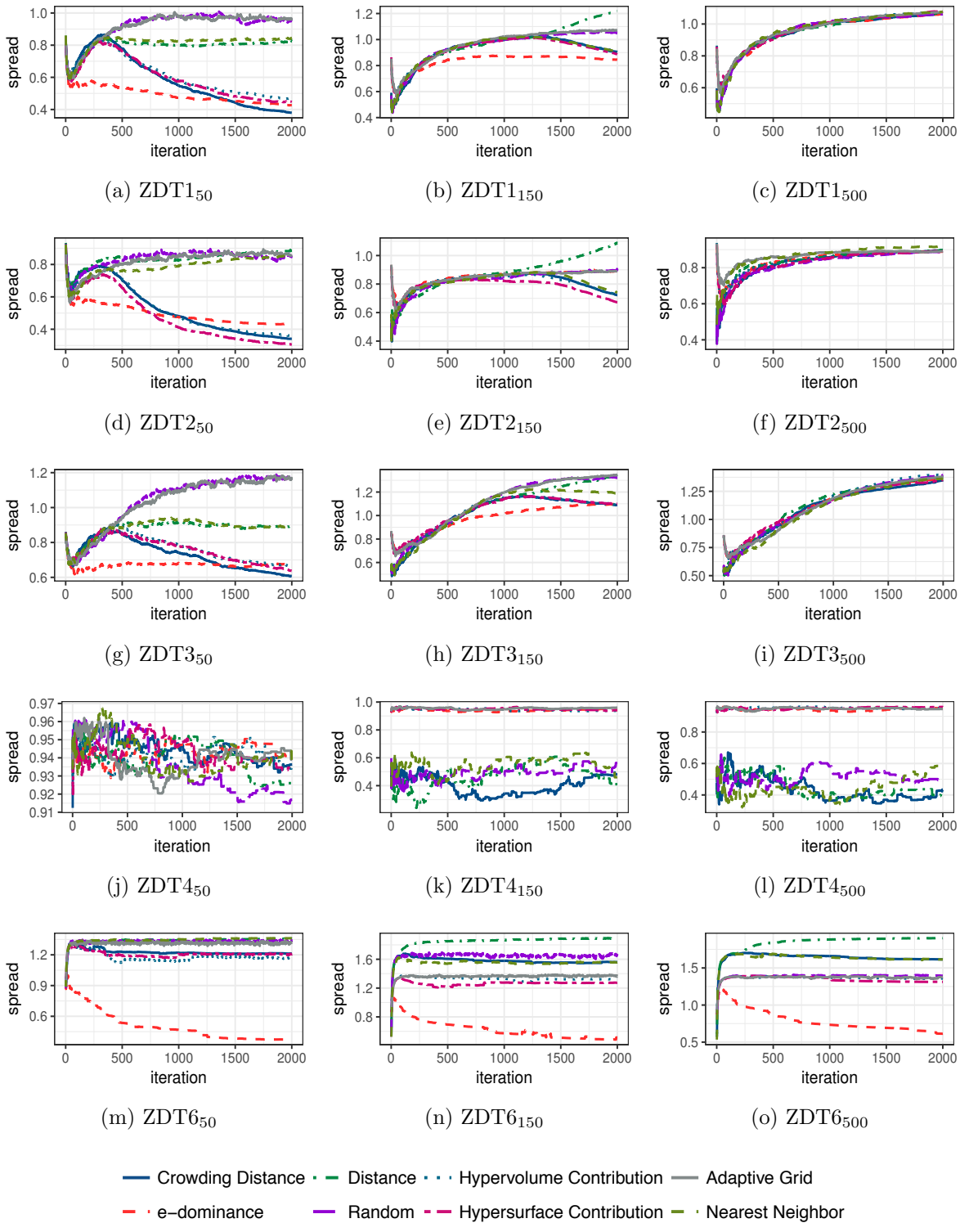
Figures 4.2 through 4.9 depict the measurement values for the four measures for ZDT1 through ZDT6 over 2000 iterations for archive sizes 50, 150, and 500. The next four subsections present an analysis for each of the measurements followed by a summary of the analysis.

Spread – Δ

Figures 4.2 and 4.3 depict the Δ (spread) values for the VEPSO (Random) and VEPSO (PCXA) algorithms, respectively. Results for each of the eight AMSs are shown. From figures 4.2(a), 4.2(d), and 4.2(g), the effect of the archive size is clearly visible. Seven of the eight AMSs for VEPSO (Random) had similar Δ values up to the point where the archive size was reached. The ϵ-dominance AMS, which does not have a fixed size limit, was the exception. Figures 4.10(a) through 4.10(c) depict the number of solutions in the archive corresponding to the Δ values in figures 4.2(a) through 4.2(c). Once the archive size was reached, the Δ decreased only in the cases where the crowding distance, hypervolume contribution, and hypersurface contribution AMSs were used.

For archive size 50, the random and adaptive grid AMSs performed worse than all the other approaches. For ZDT1, ZDT2, and ZDT6 with an archive size of 150, and ZDT with an archive size of 500, the distance AMS led to worse Δ values than even the random AMS once the archive size limit was reached. In most cases, the Δ values for the ϵ-dominance AMS tended to improve faster than the other AMSs but then stagnates consistently at a level worse than the crowding distance AMS. While the ϵ-dominance AMS results do look promising, this was due to the lack of a fixed archive size limit. The ϵ_m parameter also proved to be extremely sensitive to the performance tuning. Values for ϵ_m, which would consistently, over multiple algorithm runs, lead to specified archive sizes, could not be found for certain problems such as ZDT6.

In contrast to VEPSO (Random), the resulting Δ values for VEPSO (PCXA) are much more consistent for the different archive sizes. This can be attributed to the VEPSO (PCXA) algorithm reaching the archive size much faster than the VEPSO (Random) algorithm, as can be noted in figures 4.11(a) through 4.11(c). The speed at which the archive size was reached, reduced the number of iterations where the Δ values were



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

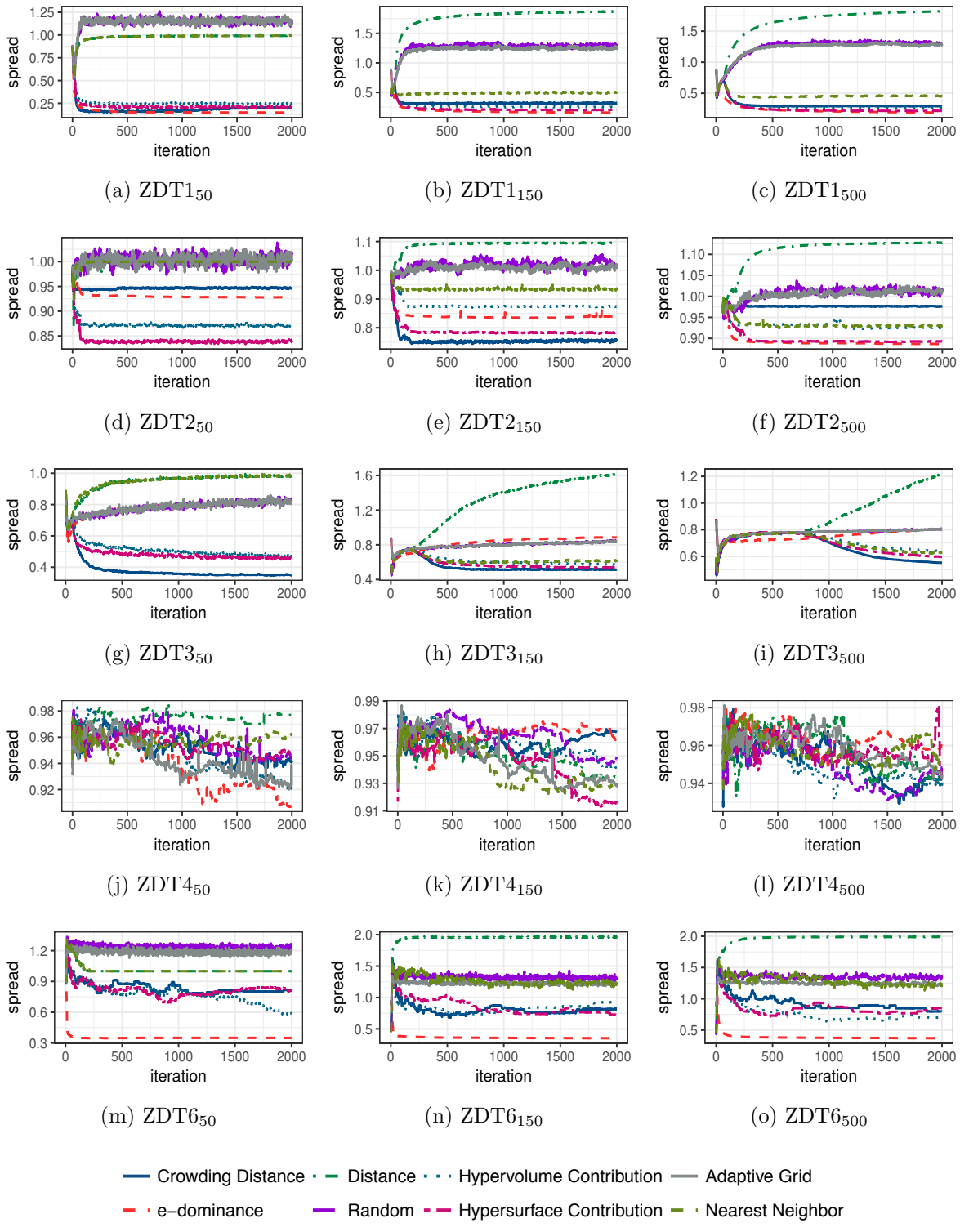
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.2: VEPSO (Random) spread, Δ, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

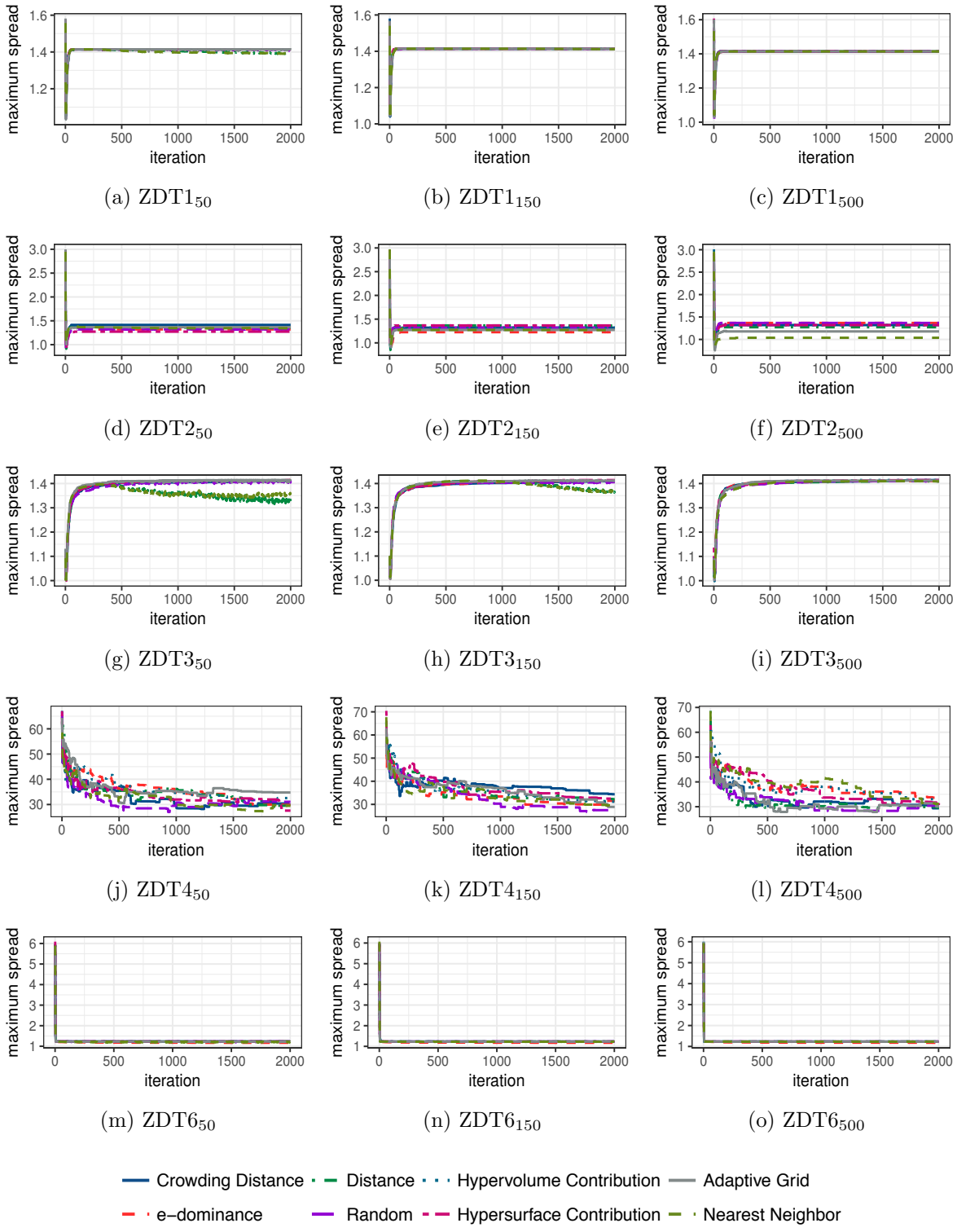
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.3: VEPSO (PCXA) spread, Δ, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

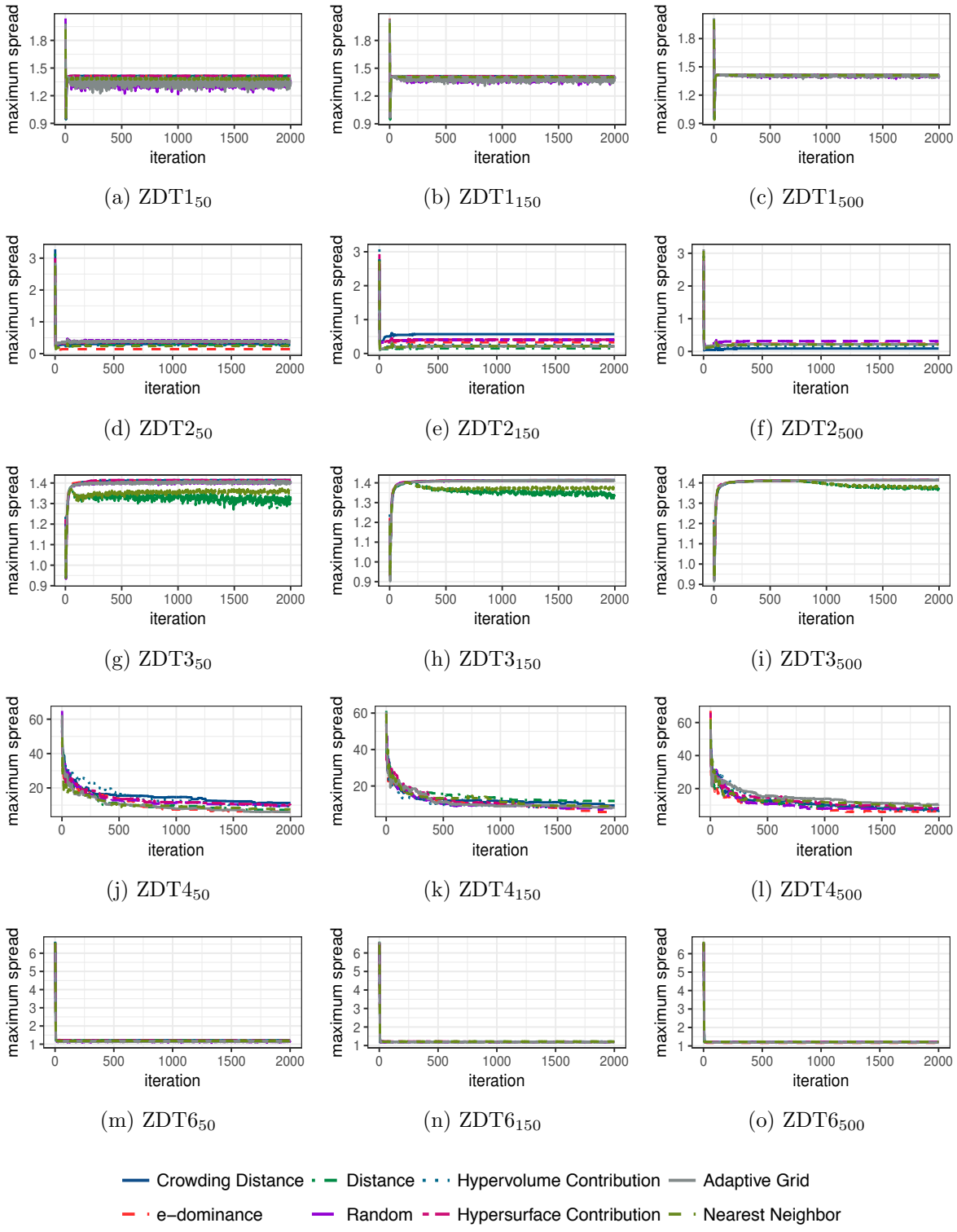
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.4: VEPSO (Random) maximum spread, $\bar{D}$, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

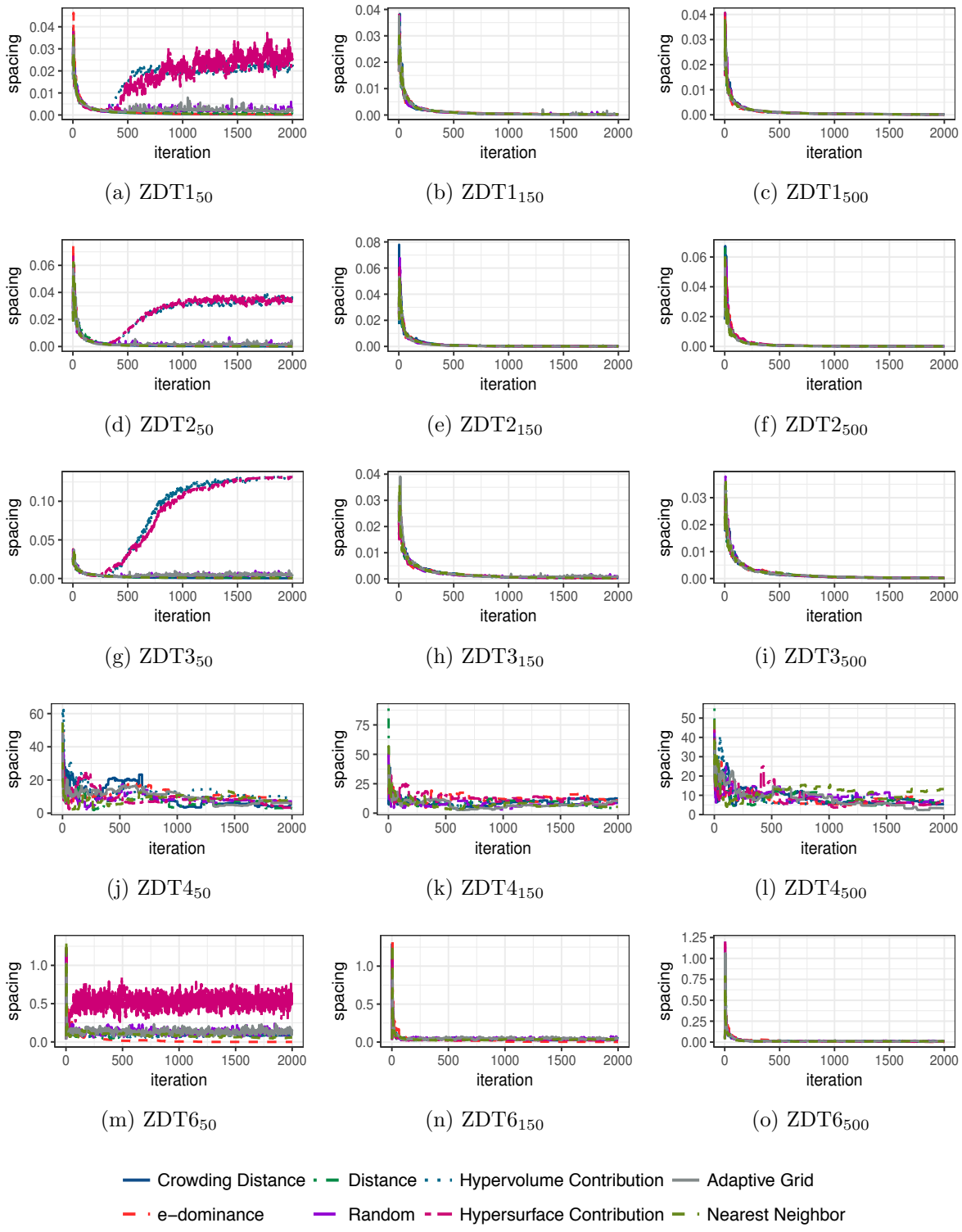
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.5: VEPSO (PCXA) maximum spread, $\bar{D}$, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

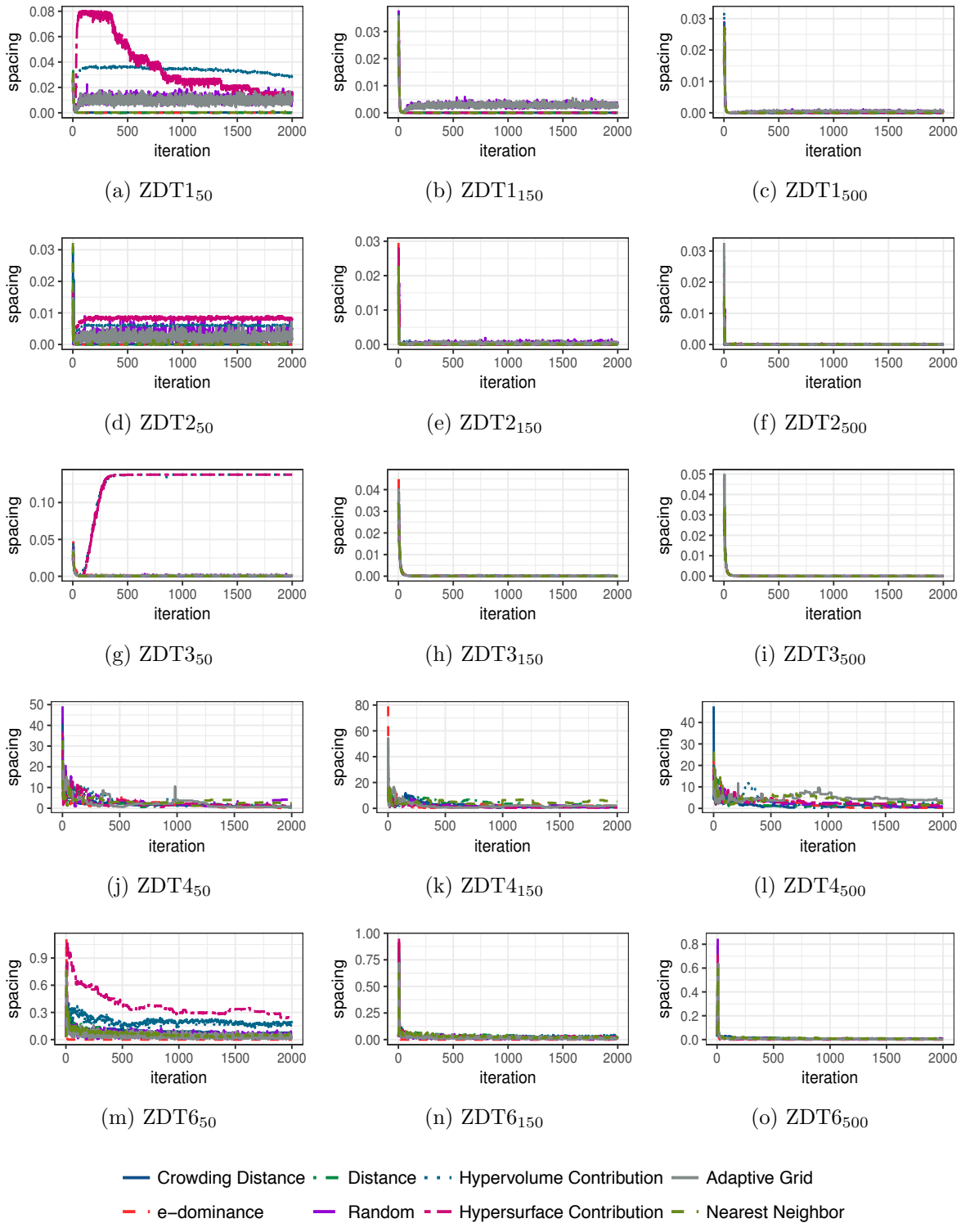
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.6: VEPSO (Random) spacing, $\bar{S}$, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

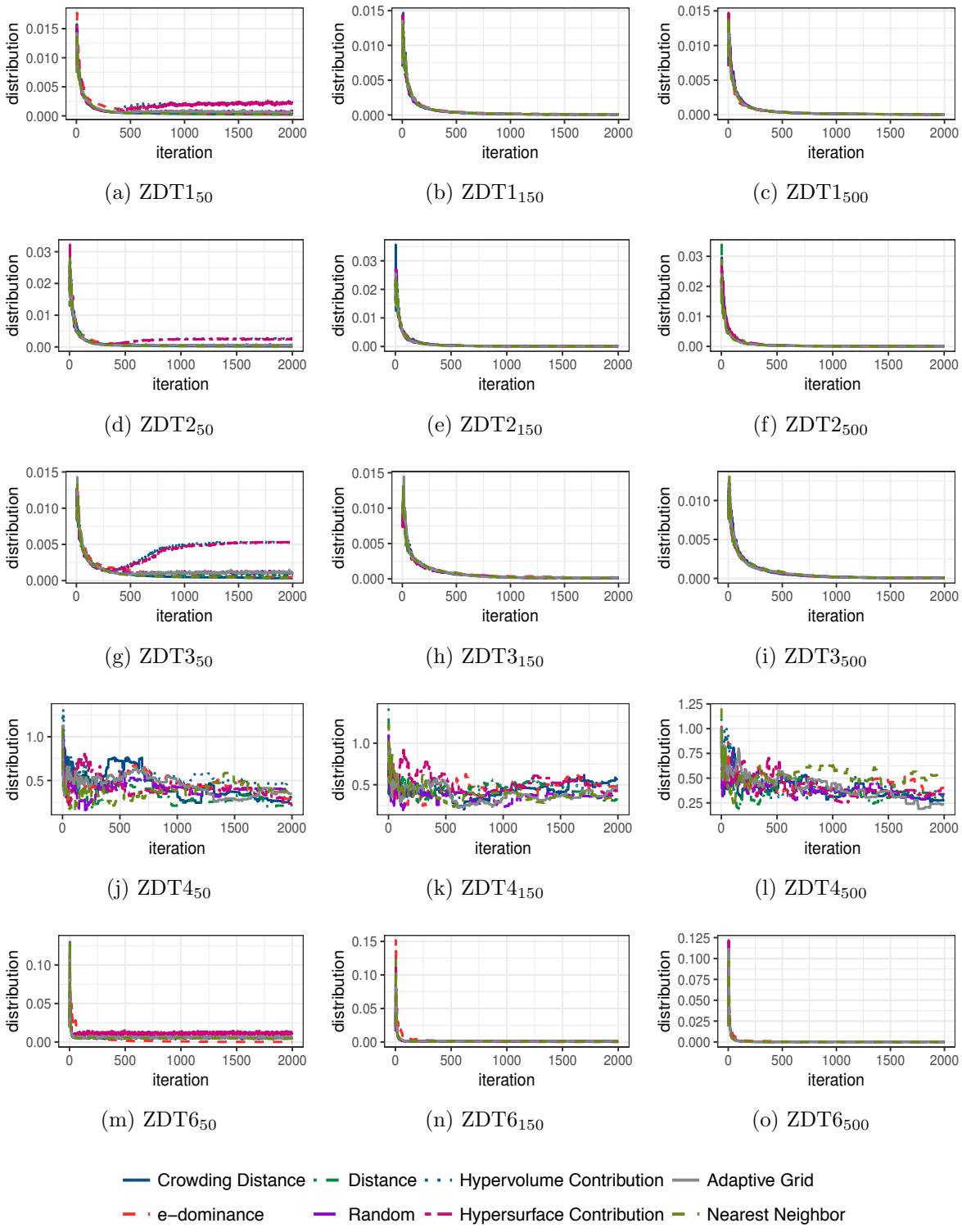
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.7: VEPSO (PCXA) spacing, $\bar{S}$, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

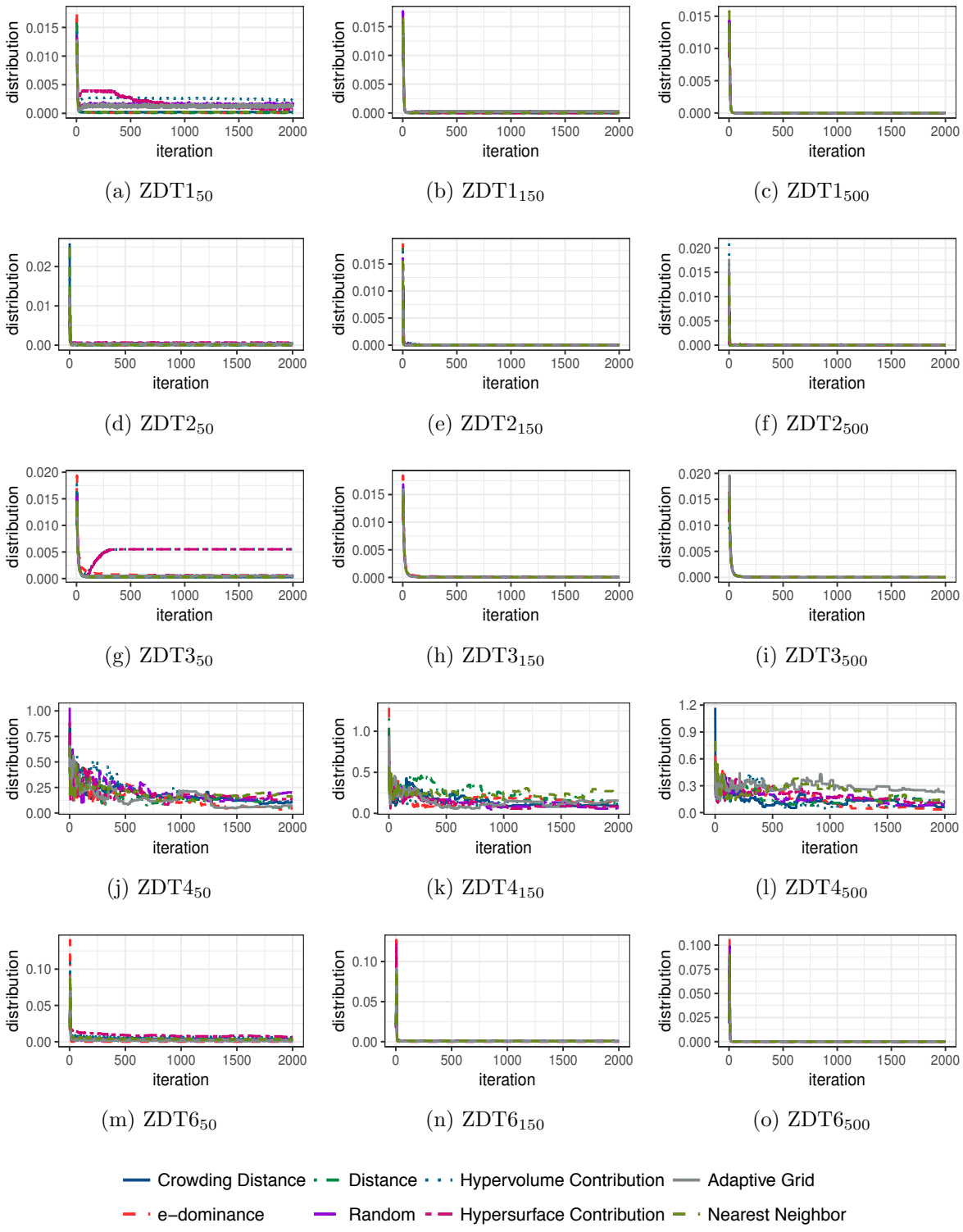
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.8: VEPSO (Random) distribution, D, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

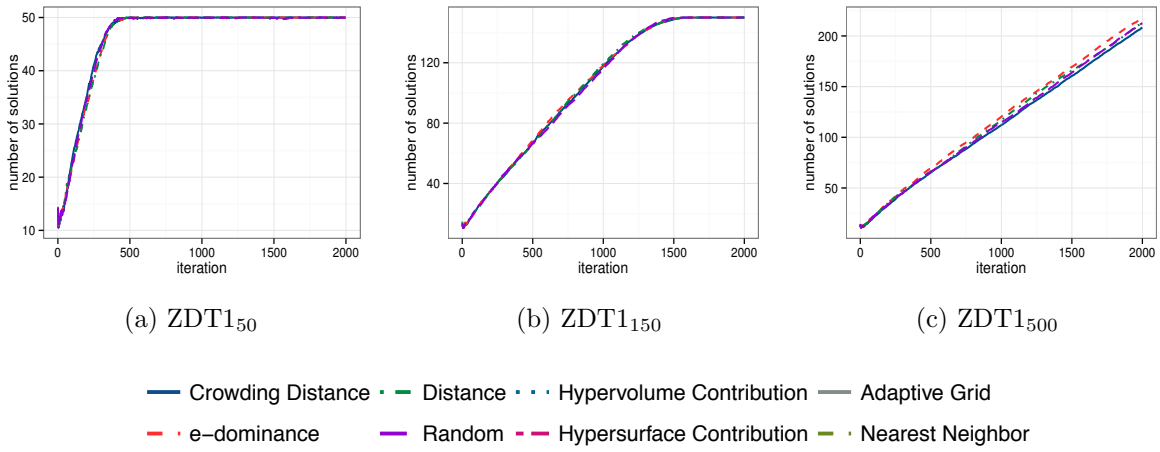
(d) $ZDT2_{50}$ (e) $ZDT2_{150}$ (f) $ZDT2_{500}$

(g) $ZDT3_{50}$ (h) $ZDT3_{150}$ (i) $ZDT3_{500}$

(j) $ZDT4_{50}$ (k) $ZDT4_{150}$ (l) $ZDT4_{500}$

(m) $ZDT6_{50}$ (n) $ZDT6_{150}$ (o) $ZDT6_{500}$

Figure 4.9: VEPSO (PCXA) distribution, D, over archive sizes 50, 150 and 500 for ZDT1 through ZDT6



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

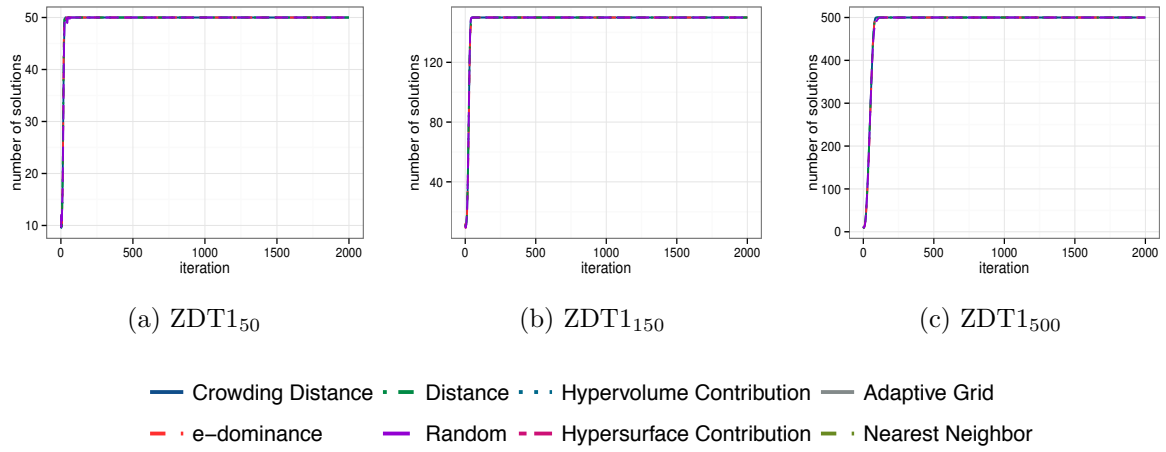
Figure 4.10: VEPSO (Random) ZDT1 number of solutions over archive sizes 50, 150 and 500

similar for the eight AMSs. Similar to VEPSO (Random), the ϵ-dominance AMS was an exception to this pattern for ZDT6. Once the archive size was reached, the crowding distance, hypervolume contribution, hypersurface contribution, or ϵ-dominance AMS performed best. For all the problems, except ZDT4, with archive sizes 150 and 500, the distance AMS performed worse than even the random AMS in terms of the Δ. The adaptive grid AMS performed on-par with the random AMS. For archive size 50, the nearest neighbor AMS performed equally to the distance AMS, including achieving the worst performance on ZDT3.

For ZDT4 the Δ was erratic for both VEPSO (Random) and VEPSO (PCXA) for all the AMSs over all three archive sizes. The Δ did not converge to any value.

For ZDT6 the archive size was reached much faster than in the case of ZDT1, ZDT2, and ZDT3. Similar behavior for the Δ can be noted once the archive size was reached.

Overall it is shown that Δ is extremely sensitive to the choice of the AMS in cases where the archive size is reached. For smaller archives, the archive size is reached faster and the effect on the Δ is thus larger in these cases. The crowding distance, hypervolume contribution, and hypersurface AMSs achieved better Δ values when compared to the nearest neighbor, distance, adaptive grid, and random AMSs. The ϵ-dominance AMS performed well when the ϵ_m parameter value is well-tuned.



(a) $ZDT1_{50}$ (b) $ZDT1_{150}$ (c) $ZDT1_{500}$

Figure 4.11: VEPSO (PCXA) ZDT1 number of solutions over archive sizes 50, 150 and 500

Maximum Spread – $\bar{D}$

$\bar{D}$ proved to be a much more consistent than Δ when comparing different archive sizes and AMSs. For VEPSO (Random) ZDT1, there was no notable difference in the $\bar{D}$ value after around 100 iterations, with only the nearest neighbor and distance AMSs showing a slightly improved $\bar{D}$ with an archive size of 50. The distance and nearest neighbor AMSs showed the most notable variance in the $\bar{D}$ value for ZDT3 over the archive sizes. Figures 4.4(g) through 4.4(i) depict the $\bar{D}$ for ZDT3 with archive sizes 50, 150 and 500. For smaller archive sizes, the distance and nearest neighbor AMSs achieved slightly lower $\bar{D}$ values. For ZDT4, the $\bar{D}$ value is somewhat erratic, with a general decrease up to around iteration 1000.

For VEPSO (PCXA) the $\bar{D}$ value behavior was similar to VEPSO (Random) with the random AMS $\bar{D}$ values being slightly erratic. The performance in the case of ZDT4 was notably less erratic. ZDT3 showed a similar pattern with the distance and nearest neighbor AMSs showing a slight improvement in $\bar{D}$ for smaller archive sizes.

Overall, the $\bar{D}$ value provided more consistent results over the eight AMSs and varying archive sizes. The $\bar{D}$ value showed sensitivity on ZDT3 to the archive size when using the distance or nearest neighbor AMSs. No discernible pattern could be seen between the $\bar{D}$ value and when the archive size was reached.

Spacing – $\bar{S}$

For VEPSO (Random) on ZDT1 though ZDT6, the $\bar{S}$ value showed little to no sensitivity towards the choice of AMS for archive sizes larger than 50. For ZDT1 through ZDT3 with archive size 50, the hypervolume contribution and hypersurface contribution AMSs maintained a higher $\bar{S}$ value than the other AMSs once the archive size was reached. For ZDT6 with archive size 50, only the hypersurface contribution AMS maintained a higher $\bar{S}$ value than the other AMSs. Again, somewhat erratic $\bar{S}$ values can be noted for ZDT4.

For VEPSO (PCXA) on ZDT1 and ZDT2, the $\bar{S}$ value showed extreme sensitivity when using the hypersurface contribution and hypervolume contribution AMSs with an archive size of 50. Figure 4.15(a) depicts the $\bar{S}$ value for ZDT1. Note the high variation in $\bar{S}$ values. In contrast to the Δ measure shown earlier, the $\bar{S}$ measure achieved lower values for the larger archive sizes. Similarly, for both VEPSO (Random) and VEPSO (PCXA), smaller $\bar{S}$ values were achieved for larger archive sizes on ZDT6. For VEPSO (PCXA) on ZDT1 through ZDT3, and ZDT6, the $\bar{S}$ measure did not show sensitivity towards the choice of crowding distance, ϵ-dominance, distance and nearest neighbor AMSs for a fixed archive size. A slight sensitivity towards the adaptive grid AMS can be noted.

Overall, it can be noted that $\bar{S}$ showed little to no sensitivity towards the choice of AMS for all archive sizes larger than 50.

Distribution – D

For VEPSO (Random), the D values showed sensitivity only towards the AMS for ZDT1 through ZDT3 with archive size 50 where the hypervolume contribution and hypersurface contribution AMSs achieved higher D values. D values for ZDT4 were, similar to the other measures, erratic.

A similar pattern can be noted for VEPSO (PCXA) as for VEPSO (Random), with only ZDT1 and ZDT3 showing sensitivity towards the hypervolume contribution and hypersurface contribution AMSs.

Overall, it can be noted that D had the least sensitivity to the archive size and choice of AMS between all the tested measures.

Summary

The experimental results showed that all the tested measures exhibited sensitivity towards the choice of AMS and archive size in at least some cases. The results indicate that Δ had the highest resolution of all the measures, but also had the highest sensitivity towards the choice of AMS and archive size. D was the least sensitive and can also be reasoned to have had the lowest resolution of all the measures. $\bar{S}$ had the highest sensitivity towards the hypersurface contribution and hypervolume contribution AMSs. $\bar{D}$ proved to be the most consistent measurement as it only measures the extent of the POF and not the diversity of the solutions that make up the POF. Due to the use of solutions from the archive, VEPSO (PCXA) was influenced more than VEPSO (Random) by the choice of AMS. Overall, the crowding distance AMS generally yielded good results for all the POF diversity measures.

The objective of this Section was to evaluate the hypothesis that the diversity is influenced by the choice of AMS. The experimental results using the ZDT problems with different AMSs confirmed this hypothesis. From the experimental results it can be concluded that care must be taken when choosing the archive size and AMS as the impact on the POF diversity can be notable. Comparing diversity measures between algorithms should not be done without taking the archive size into account.

An investigation into why the $\bar{S}$ and D measures displayed less sensitivity to changes in the archive size and AMS is presented in the next section.

4.4 Misleading Pareto-optimal Front Diversity Measures

The previous section showed that the distribution, D, and spacing, $\bar{S}$, measures displayed the least sensitivity towards both the archive size and AMS. A more detailed analysis of the distribution, D, and spacing, $\bar{S}$, calculations reveal a potential weakness that could explain the lack of sensitivity. Section 4.4.1 presents the pairwise hypothesis as the cause of the lack of sensitivity. Section 4.4.2 introduces the crowding distribution measure to address the pairwise weakness, followed by an analysis that validates the

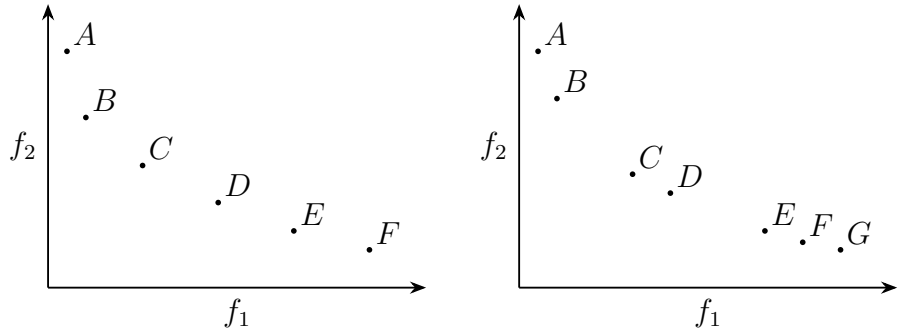
pairwise hypothesis in Section 4.4.3.

4.4.1 Pairwise Hypothesis

As presented in section 2.4.2, distribution calculates d_k as the Euclidean distance in objective space between solution k and its nearest neighbor in Q. Similarly, spacing calculates d_k as the Manhattan distance between solution k and its nearest neighbor in Q. In each case, the distance between solution k and its nearest neighbor in Q is used in the diversity calculation.

Using the nearest neighbor solution in objective space creates a pairwise combination problem where two solutions select each other during the diversity calculation. Figure 4.12 depicts two example POFs for discussion purposes. Figure 4.12(a) illustrates a well-spread POF with a nearly equal distribution of solutions. Potential nearest neighbor pairings for the first example are (A, B), (B, C), (D, E), and (E, F). Note that the (C, D) pairing is not used in any calculation as it is not the nearest neighbor pairing. In the second example illustrated in figure 4.12(b), the solutions that make up the POF are clustered. In this case, the nearest neighbor pairings would be (A, B), (C, D), (E, F), and (F, G). The diversity calculations will not take the larger distances between B and C or D and E into consideration, which in turn, would lead to misleading diversity measurement values. As long as the distances between the solutions that make up the nearest neighbor pairings are close to equal, the spacing and distribution measurement values would indicate a good diversity.

Adapting the distribution and spacing calculations to take more than one nearest neighbor into account would not solve the problem. In figure 4.12(b) a grouping of three solutions, (E, F, G), can easily be formed without taking the distance between D and E into account. Similarly, in actual POFs, solutions can be grouped into groups of three or more solutions that would still be susceptible to the same problem as the pairwise groupings; that is, not to take all the distances between solutions into account.



(a) Nearly equal distribution (b) Clustered distribution

Figure 4.12: POF solution distribution examples

4.4.2 Crowding Distance based Distribution

In order to verify the hypothesis that the nearest neighbor pairwise groupings affect the resulting distribution and spacing values, a new diversity measure is introduced. The new diversity measure is based on crowding distance, and avoids the pairwise problem by using sorted sets to determine the next neighbor for the distance calculation. The crowding distribution is defined as:

$$C = \frac{1}{|Q| - 1} \sum_{k=1}^{|Q|} |\bar{d} - d_k| \tag{4.5}$$

with

$$\bar{d} = \frac{1}{|Q|} \sum_{k=1}^{|Q|} d_k \tag{4.6}$$

and

$$d_k = \sum_{j=1}^{n_m} \left(d^*_{k,j} - q_{k,j} \right) \tag{4.7}$$

where

$$d^*_{k,m} = \begin{cases} \min\{q_{u,m}\} & \text{if } \mathbf{q}_u \in Q : q_{u,m} > q_{k,m} \\ 0 & \text{otherwise} \end{cases} \tag{4.8}$$

and n_m is the number of objectives, d_k is calculated for solution $\mathbf{q}_k \in Q \subseteq \mathbb{R}^{n_m}$, similar to the crowding distance, as the sum of the differences between $q_{k,m}$ and $q_{k+1,m}$; $q_{k+1,m}$

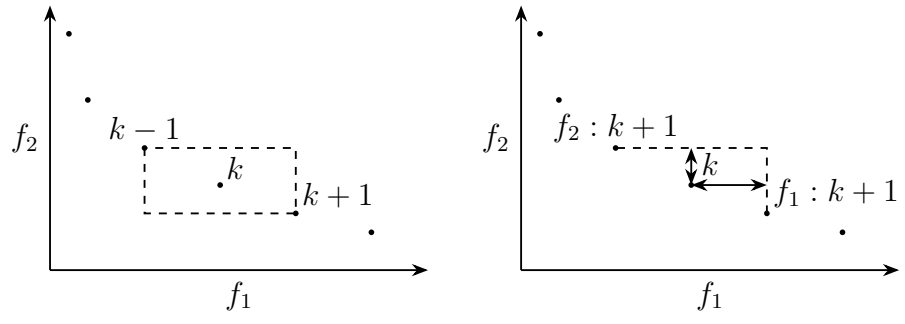
is the next solution in the set Q ordered by the m'th objective. Figure 4.13 illustrates the crowding distance hypercube and the distances used by the crowding distribution calculation; $f_m : k+1$ denotes the next solution in the set Q ordered by the m'th objective.

The crowding distribution is not subject to the pairwise problem due to the use of the crowding distance based calculation of d_k.

It should be noted that the distance calculations used in the distribution and spacing calculations cannot be changed to avoid the pairwise problem. Selecting the neighboring solution from a set sorted by objective function will not scale to more than two objectives. On the other hand, the proposed crowding distribution measure can be used for problems with more than two objectives.

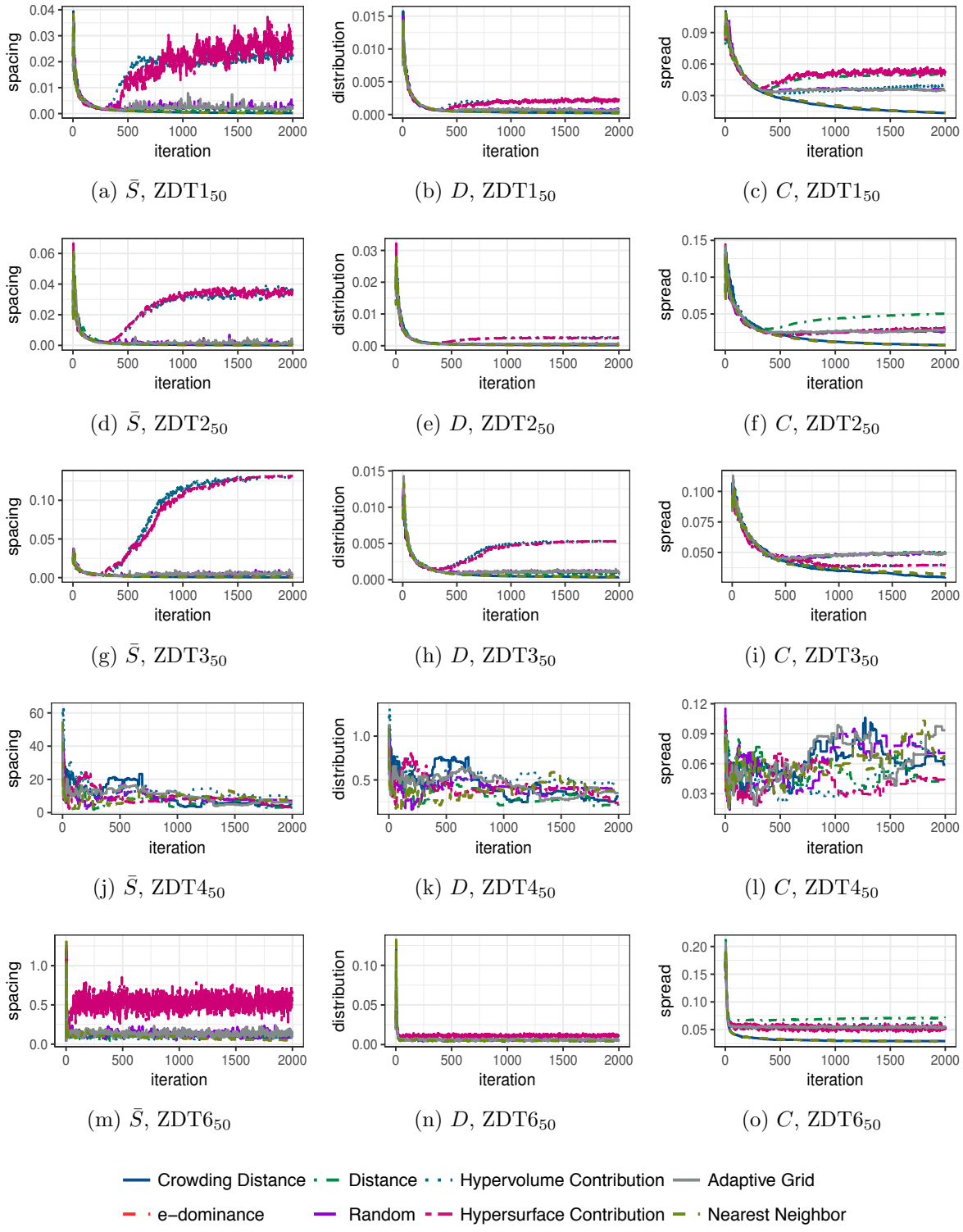
4.4.3 Analysis

Figures 4.14(a) through 4.26(e) depict the measurement values for the three measures, the number of solutions in the archive for ZDT1 through ZDT6 over 2000 iterations as well as a selection of the resulting POFs. For VEPSO (Random) on ZDT1, ZDT2, and ZDT3, the crowding distribution, C, measurement values stopped decreasing around iteration 400 for the distance, random, and adaptive grid AMSs. The discontinuation in the decrease of the measurement values for C deviates from the measurement values for distribution, D, and spacing, $\bar{S}$, where a continued decrease can be noted. From



(a) Crowding distance (b) Crowding distribution

Figure 4.13: Crowding distance and distribution calculation hypercubes



(a) $\bar{S}$, ZDT1_{50} (b) D, ZDT1_{50} (c) C, ZDT1_{50}

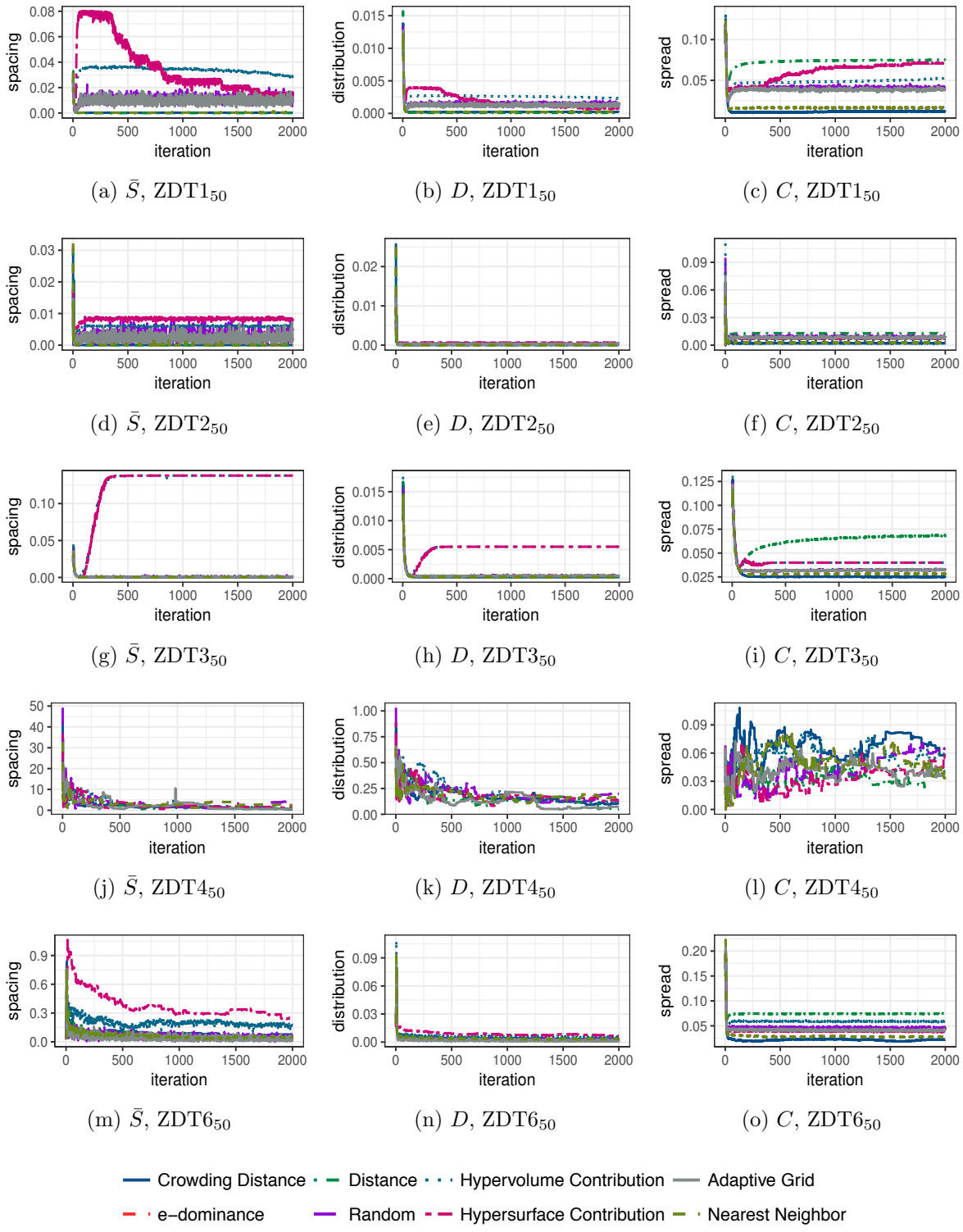
(d) $\bar{S}$, ZDT2_{50} (e) D, ZDT2_{50} (f) C, ZDT2_{50}

(g) $\bar{S}$, ZDT3_{50} (h) D, ZDT3_{50} (i) C, ZDT3_{50}

(j) $\bar{S}$, ZDT4_{50} (k) D, ZDT4_{50} (l) C, ZDT4_{50}

(m) $\bar{S}$, ZDT6_{50} (n) D, ZDT6_{50} (o) C, ZDT6_{50}

Figure 4.14: VEPSO (Random) spacing, $\bar{S}$, distribution, D, crowding distribution, C, with archive size 50 for ZDT1 through ZDT6



(a) $\bar{S}$, ZDT1_{50} (b) D, ZDT1_{50} (c) C, ZDT1_{50}

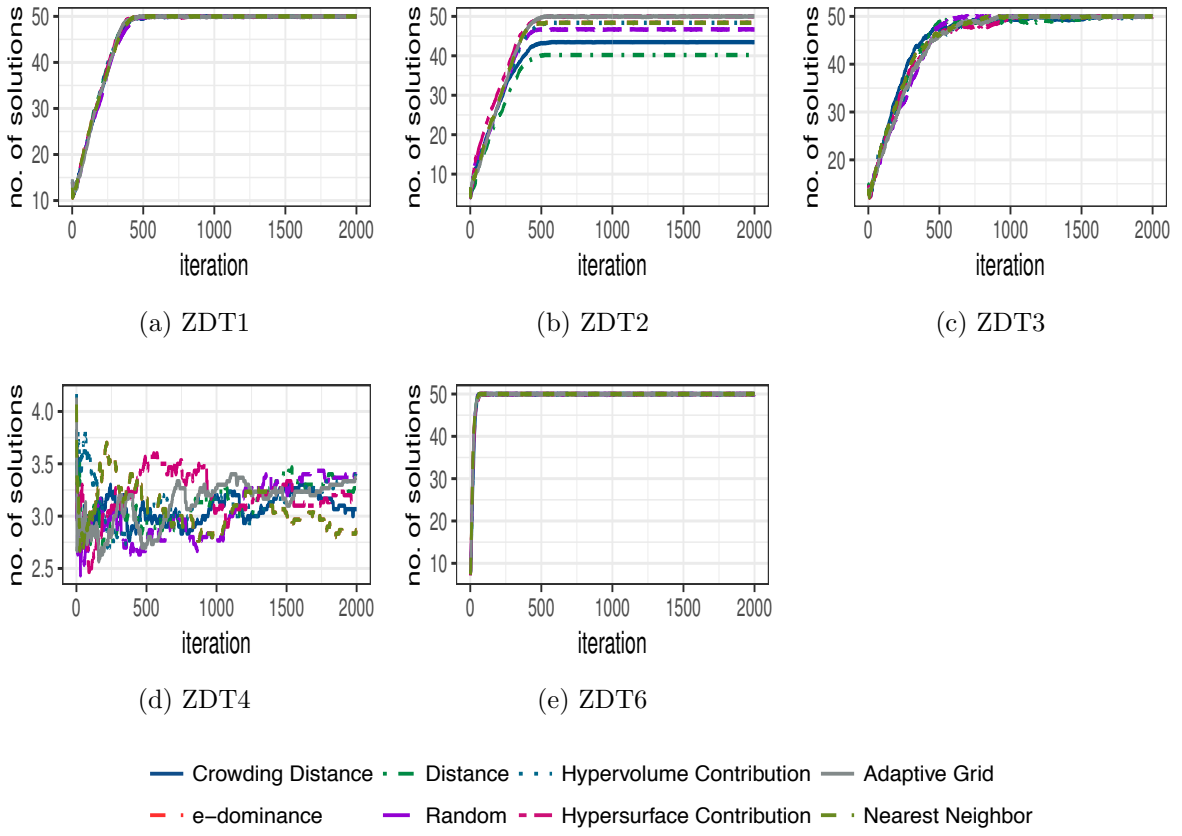
(d) $\bar{S}$, ZDT2_{50} (e) D, ZDT2_{50} (f) C, ZDT2_{50}

(g) $\bar{S}$, ZDT3_{50} (h) D, ZDT3_{50} (i) C, ZDT3_{50}

(j) $\bar{S}$, ZDT4_{50} (k) D, ZDT4_{50} (l) C, ZDT4_{50}

(m) $\bar{S}$, ZDT6_{50} (n) D, ZDT6_{50} (o) C, ZDT6_{50}

Figure 4.15: VEPSO (PCXA) spacing, $\bar{S}$, distribution, D, crowding distribution, C, with archive size 50 for ZDT1 through ZDT6



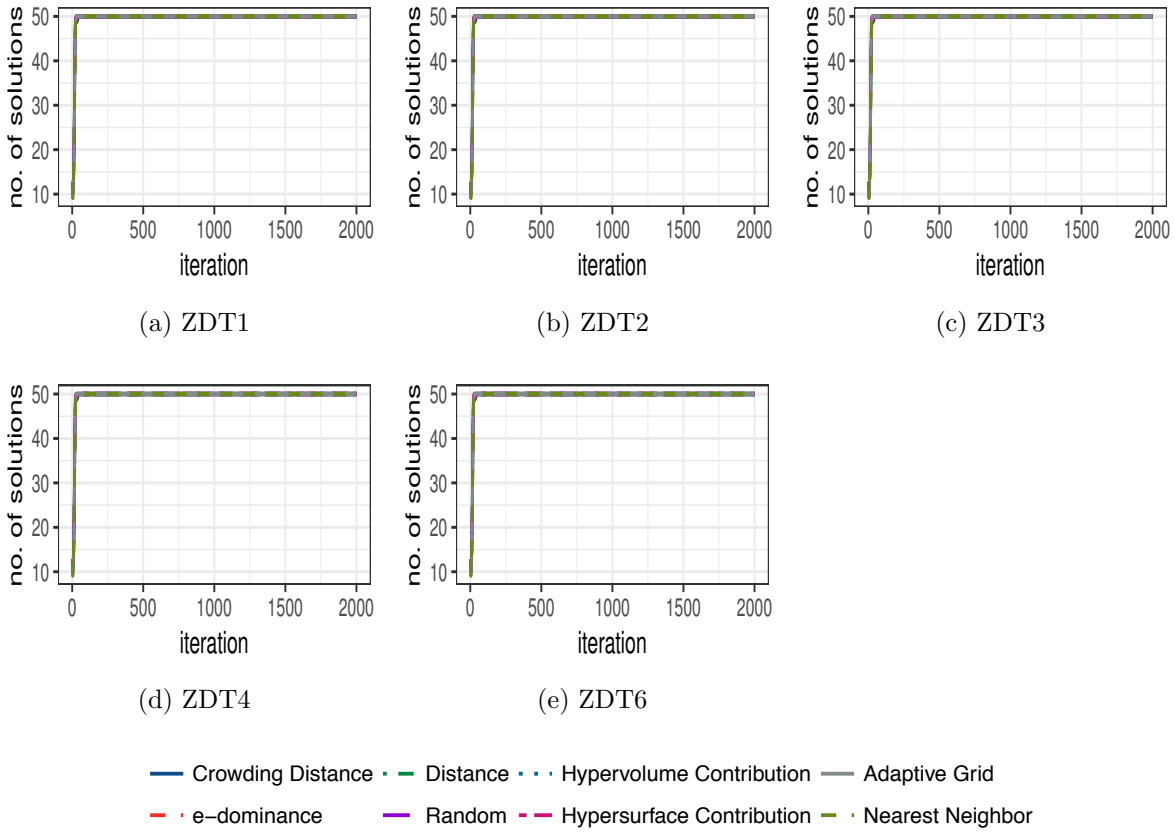
(a) ZDT1 (b) ZDT2 (c) ZDT3

(d) ZDT4 (e) ZDT6

Figure 4.16: VEPSO (Random) number of solutions

figures 4.16(a) through 4.16(c) it can be noted that the archive limit is reached around iteration 400. This corresponds to where the C measurement values start to deviate from the D and $\bar{S}$ measurement values. Figures 4.18(a) through 4.19(e) show the POFs obtained for iterations 100, 250, 500, 750, and 1000 for the VEPSO (Random) algorithm with the distance and nearest neighbor AMSs. A large gap in the POF in figure 4.18(e) can be noted. The nearest neighbor AMS POF, in turn, shows a much better spread of solutions. The POFs clearly confirm that the C measurement values represent the actual spread of solutions whereas the D and $\bar{S}$ measurement values give a misleading indication of the actual spread of solutions.

The POFs shown in figures 4.20(a) through 4.21(e) confirm that a similar misleading indication between the actual spread and the D and $\bar{S}$ measurement values exists. In



(a) ZDT1 (b) ZDT2 (c) ZDT3

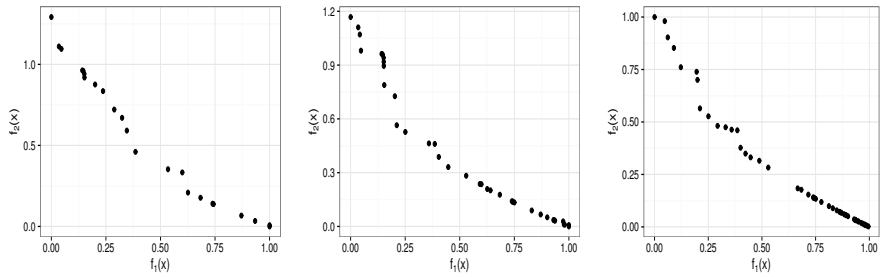
(d) ZDT4 (e) ZDT6

Figure 4.17: VEPSO (PCXA) number of solutions

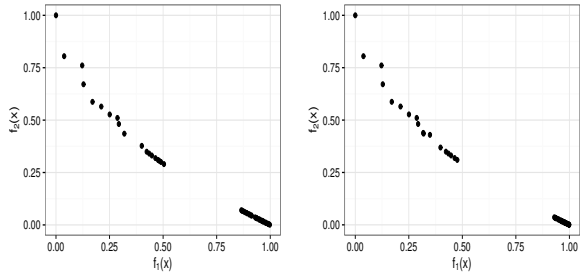
this case, a degradation in the visible diversity of the solutions can be noted between the POF for iteration 500 shown in figure 4.21(c) and the POF at iteration 1000 shown in figure 4.21(e).

The ZDT4 results were erratic for all three diversity measures. It should also be noted that none of the ZDT4 algorithms reached the archive size limit. The ZDT6 C results show that the distance AMS had the highest diversity and the crowding and nearest neighbor AMSs had the lowest diversity. The D and $\bar{S}$ results did not reveal the same trend. The POFs shown in figures 4.22(a) through 4.23(e) confirm that the D and $\bar{S}$ measurement values are misleading and that there is a notable difference in the spread of solutions as indicated by the C measurement values.

For VEPSO (PCXA) on ZDT1, the distance AMS had higher C values than that

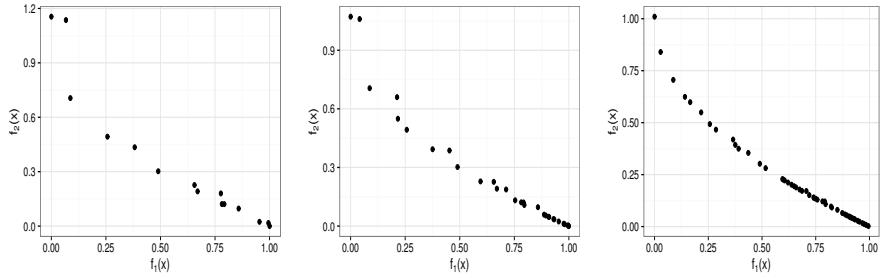


(a) $t = 100$ (b) $t = 250$ (c) $t = 500$

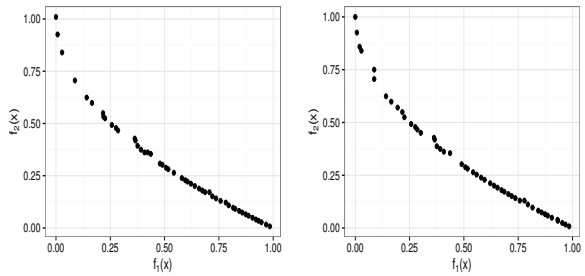


(d) $t = 750$ (e) $t = 1000$

Figure 4.18: VEPSO (Random) with distance AMS POF for ZDT1

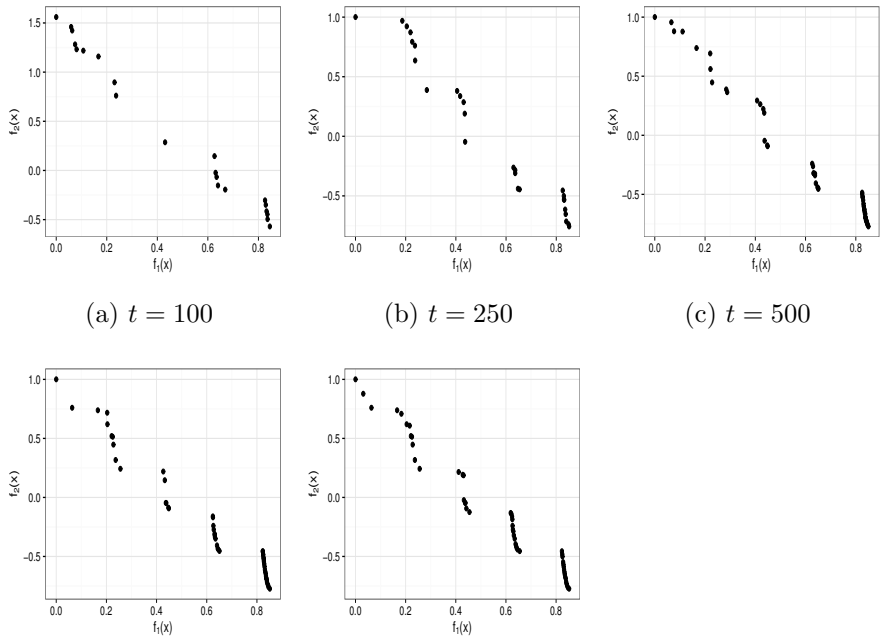


(a) $t = 100$ (b) $t = 250$ (c) $t = 500$



(d) $t = 750$ (e) $t = 1000$

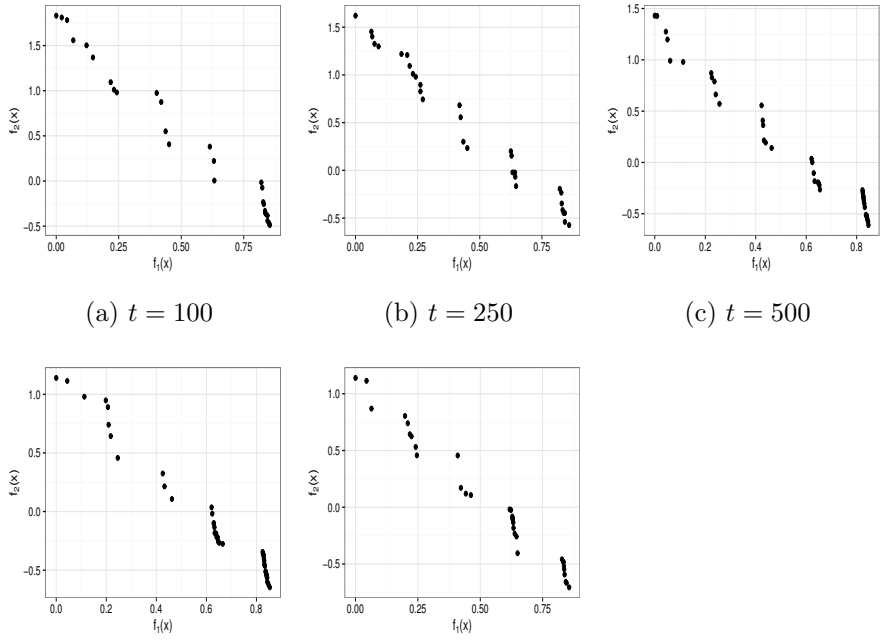
Figure 4.19: VEPSO (Random) with nearest neighbor AMS POF for ZDT1



(a) $t = 100$ (b) $t = 250$ (c) $t = 500$

(d) $t = 750$ (e) $t = 1000$

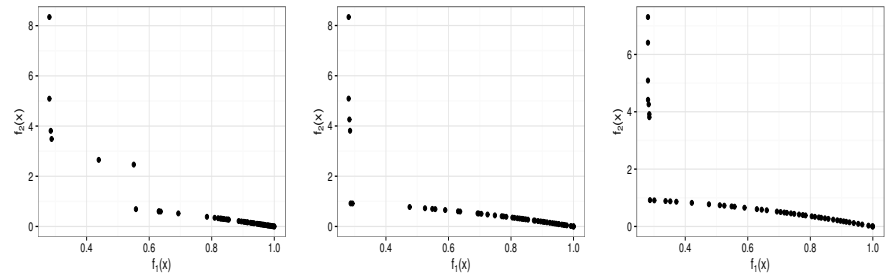
Figure 4.20: VEPSO (Random) with crowding distance AMS POF for ZDT3



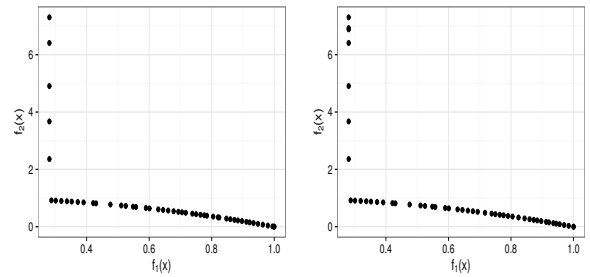
(a) $t = 100$ (b) $t = 250$ (c) $t = 500$

(d) $t = 750$ (e) $t = 1000$

Figure 4.21: VEPSO (Random) with random AMS POF for ZDT3

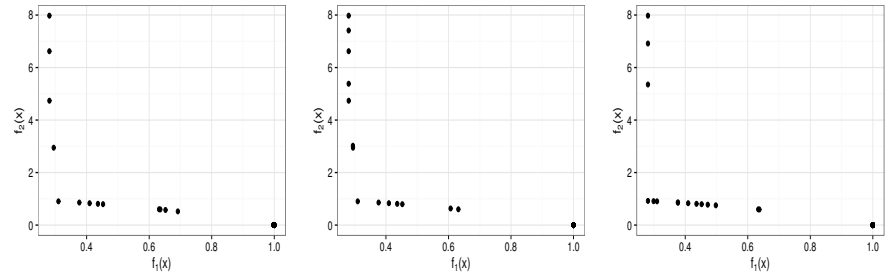


(a) $t = 100$ (b) $t = 250$ (c) $t = 500$

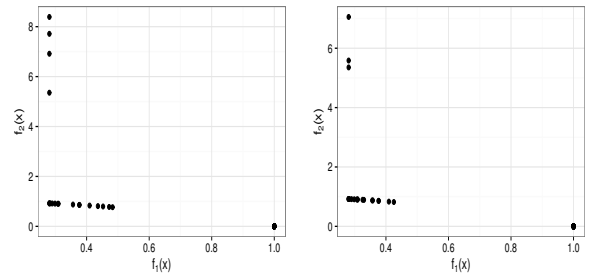


(d) $t = 750$ (e) $t = 1000$

Figure 4.22: VEPSO (Random) with crowding distance AMS POF for ZDT6



(a) $t = 100$ (b) $t = 250$ (c) $t = 500$



(d) $t = 750$ (e) $t = 1000$

Figure 4.23: VEPSO (Random) with distance AMS POF for ZDT6

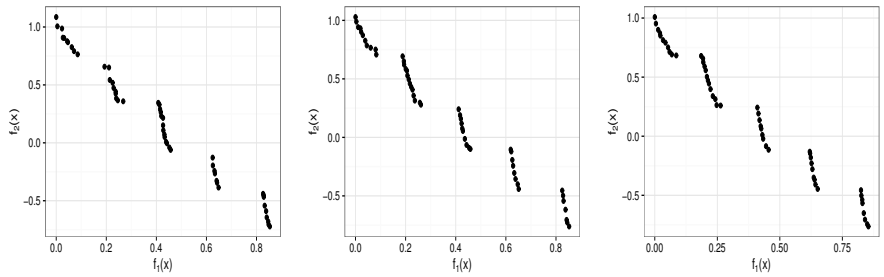
of the other archive management strategies. Again, this pattern does not match the D results where the hypersurface contribution and hypervolume contribution AMSs had higher diversities. For ZDT1, the hypersurface contribution D and $\bar{S}$ measurement values start high and then gradually decreases, whereas the C measurement values start lower and then gradually increases. For ZDT1 the random and adaptive grid AMSs had an unstable $\bar{S}$ value as visible in figure 4.15(a). Similar to the ZDT1 result, the $\bar{S}$ values for ZDT2 were unstable for the random and adaptive grid AMSs. No notable difference in D values could be noted for the eight AMSs. The distance AMS had slightly higher C values.

For ZDT3 the C measurement values again show varying performance for the eight AMSs whereas the $\bar{S}$ and D measurement values showed only a varying performance for the hypersurface contribution and hypervolume AMSs. Figures 4.24(a) through 4.26(e) clearly show that the AMSs achieved notably different spreads as reflected in the C measurement values. Figures 4.25(e) is an excellent example of the pairwise grouping problem. Clusters of solutions are clearly visible, and the corresponding $\bar{S}$ and D measurement values give no indication of the degraded spread of solutions.

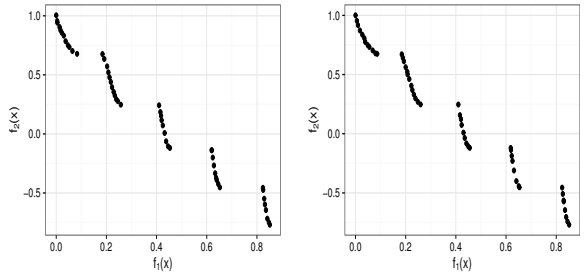
Similar to the VEPSO (Random) results for ZDT4, the diversity measures were erratic. For ZDT6, in contrast to the D and $\bar{S}$ results, the C values showed notable differences in the results achieved by the AMSs. Only the hypersurface contribution and the hypervolume contribution showed a difference in the $\bar{S}$ measurement values.

Overall, the results indicate that the distribution, D, and spacing, $\bar{S}$, measurement values were misleading whereas the crowding distribution, C, measurement values gave a more accurate indication of the actual spread of solutions on the POF. The results confirm the hypothesis that the distribution, D, and spacing, $\bar{S}$, measures are susceptible to a pairwise grouping problem where larger distances in the actual spread of solutions are ignored. This, in turn, leads to misleading measurement values.

It should also be noted that the crowding distribution measurement calculation uses the same crowding distance calculation as the crowding distance AMS. The use of the same crowding distance calculation could lead to a selection bias problem where the crowding distance AMS could wrongly achieve slightly better results than other diversity measures. This bias should not affect the findings in this section as the objective of this

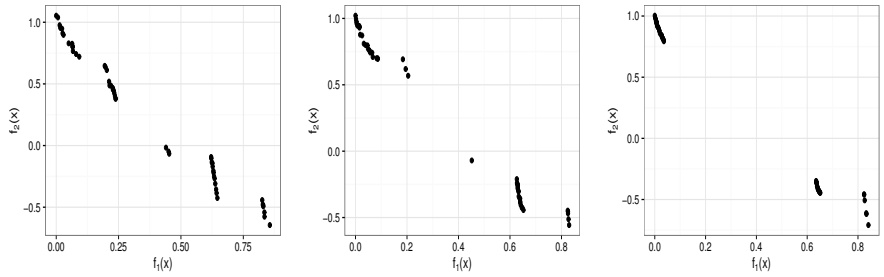


(a) $t = 100$ (b) $t = 250$ (c) $t = 500$

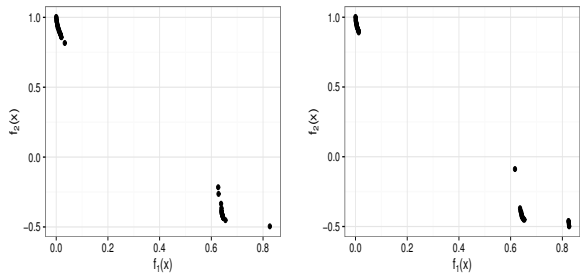


(d) $t = 750$ (e) $t = 1000$

Figure 4.24: VEPSO (PCXA) with crowding distance AMS POF for ZDT3

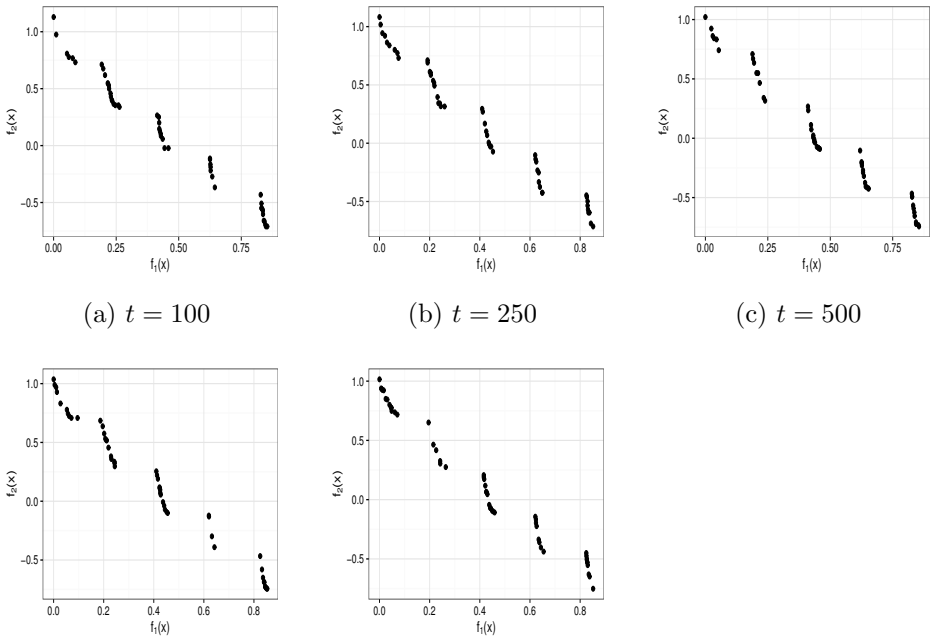


(a) $t = 100$ (b) $t = 250$ (c) $t = 500$



(d) $t = 750$ (e) $t = 1000$

Figure 4.25: VEPSO (PCXA) with distance AMS POF for ZDT3



(a) $t = 100$ (b) $t = 250$ (c) $t = 500$

(d) $t = 750$ (e) $t = 1000$

Figure 4.26: VEPSO (PCXA) with random AMS POF for ZDT3

section was to show that the distribution and spacing measurement calculations are insensitive and misleading due to the pairwise grouping problem.

4.5 Archive Management Strategy's Influence on Performance

This section presents an evaluation of the impact that the choice of AMS has on VEPSO's performance. A discussion on how to measure the influence on performance is given in Section 4.5.1, followed by a performance analysis in Section 4.5.2.

4.5.1 Measuring Performance

The results for the VEPSO (Random) and VEPSO (PCXA) algorithms are presented in this section. The performance was measured using the well-known IGD measurement, as

described in section 2.4.3. The true POF sets from the jMetal framework [80] were used for all the IGD calculations. Each KTS was evaluated using the eight AMSs presented earlier in this chapter. The results presented were taken over 30 independent runs of 2000 iterations of each algorithm using the ZDT and 2-objective WFG test sets. The inertia weight, w was set to 0.729844, and the acceleration constants, c_1 and c_2, were set to 1.49618, n_n was set to 2 for the nearest neighbor AMS.

4.5.2 Performance Analysis

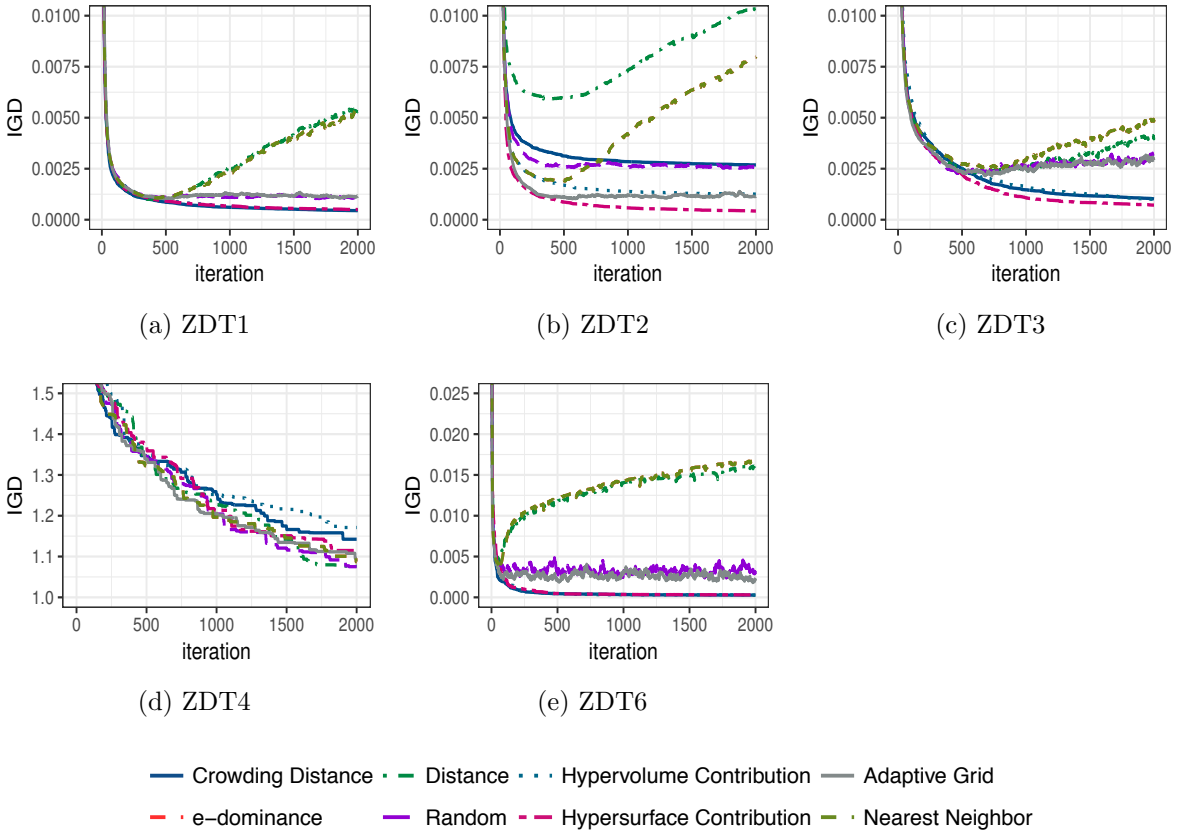
The next two subsections present an analysis of the performance for each of the VEPSO KTSs followed by a summary of the analysis.

VEPSO (Random)

Figures 4.27(a) through 4.27(e) and 4.28(a) through 4.28(i) depict the average IGD measurement values for ZDT1 through ZDT6 and WFG1 through WFG9 for the VEPSO with a random KTS. For 12 of the 14 problems, the nearest neighbor and distance AMSs performed notably worse than all the other AMSs. It should be noted that, in nine cases, the random AMS performed on-par with the other AMSs. In the remaining cases, the random AMS performed on-par with the adaptive grid AMS. For 12 of the cases, the IGD measurement values for the nearest neighbor and distance AMSs increase after an initial decrease, indicating that the POF approximation degrades the longer the algorithm runs. For ZDT4 the hypervolume contribution and crowding distance AMSs performed notably worse than all other AMSs.

A statistical analysis of the performance of the various AMSs is presented in Table 4.1. IGD measurement values for each of the AMSs were compared with all the other AMSs using the Mann-Whitney U [43] test with a confidence level of 95%. If a statistically significant difference was detected a win was recorded for the AMS and a loss was recorded for the other AMS. The difference between the wins and losses were computed and is also listed. A ranking was assigned according to the win-loss difference [48]. The overall wins, losses, difference, and rank across all the problems are shown.

The results indicate that the crowding distance AMS performed overall best, followed by the ϵ-dominance and hypersurface contribution AMSs. The crowding distance AMS



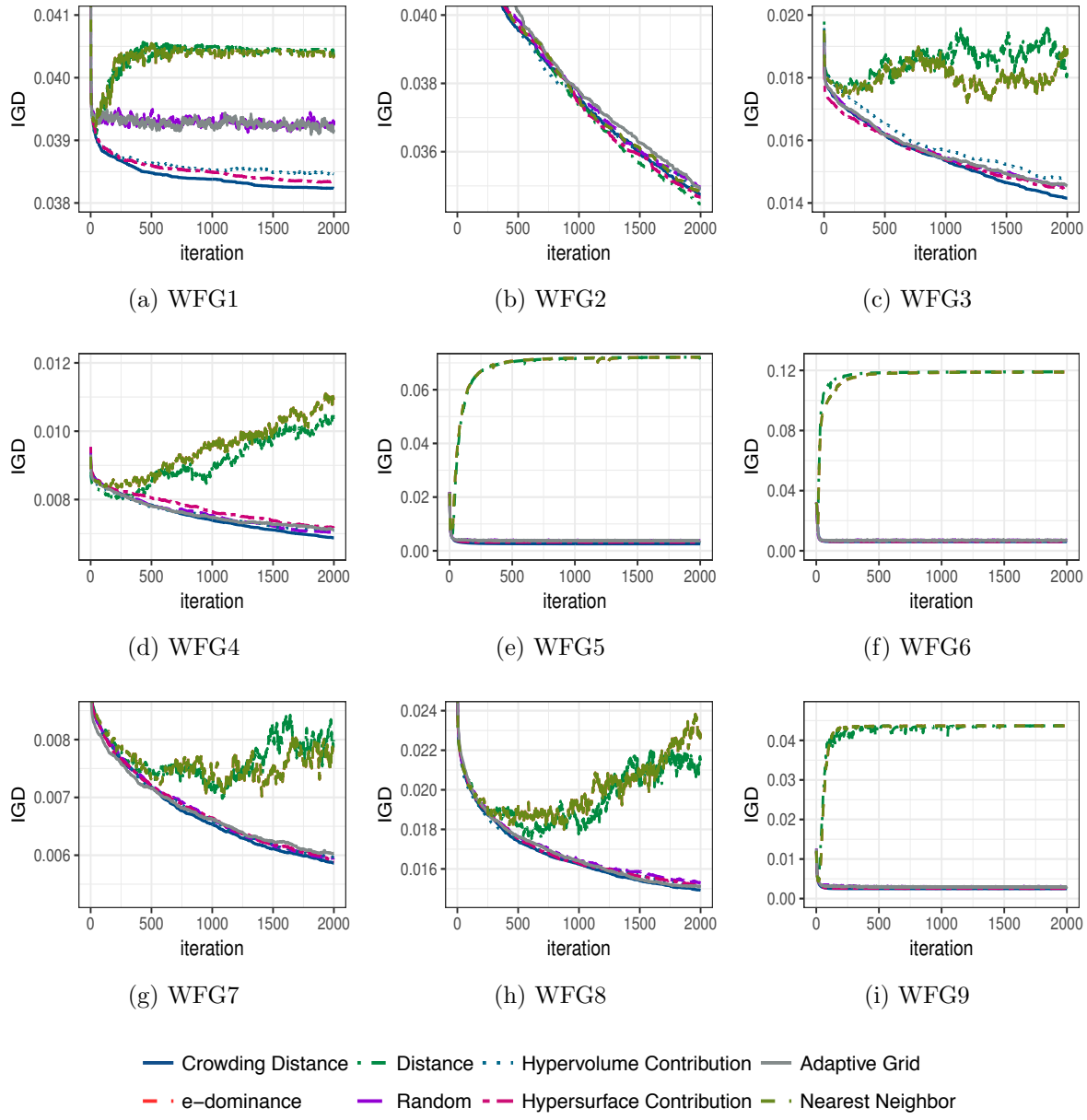
(a) ZDT1 (b) ZDT2 (c) ZDT3

(d) ZDT4 (e) ZDT6

Figure 4.27: VEPSO (Random) Inverted Generational Distance for ZDT1 through ZDT6

recorded only three losses compared to 70 wins. The ϵ-dominance AMS ranked first for seven of the 14 problems including for ZDT4 where the crowding distance AMS ranked second to last. The ϵ-dominance AMS however recorded nine losses compared to 67 wins, ranking second overall. The distance and nearest neighbor AMSs performed statistically the worst. The distance AMS recorded 75 losses compared to a single win, while the nearest neighbor AMS recorded 72 losses with only four wins.

It can also be noted that overall, the results for the ZDT and WFG functions tend to correspond with one another.



(a) WFG1 (b) WFG2 (c) WFG3

(d) WFG4 (e) WFG5 (f) WFG6

(g) WFG7 (h) WFG8 (i) WFG9

Figure 4.28: VEPSO (Random) Inverted Generational Distance for WFG1 through WFG9

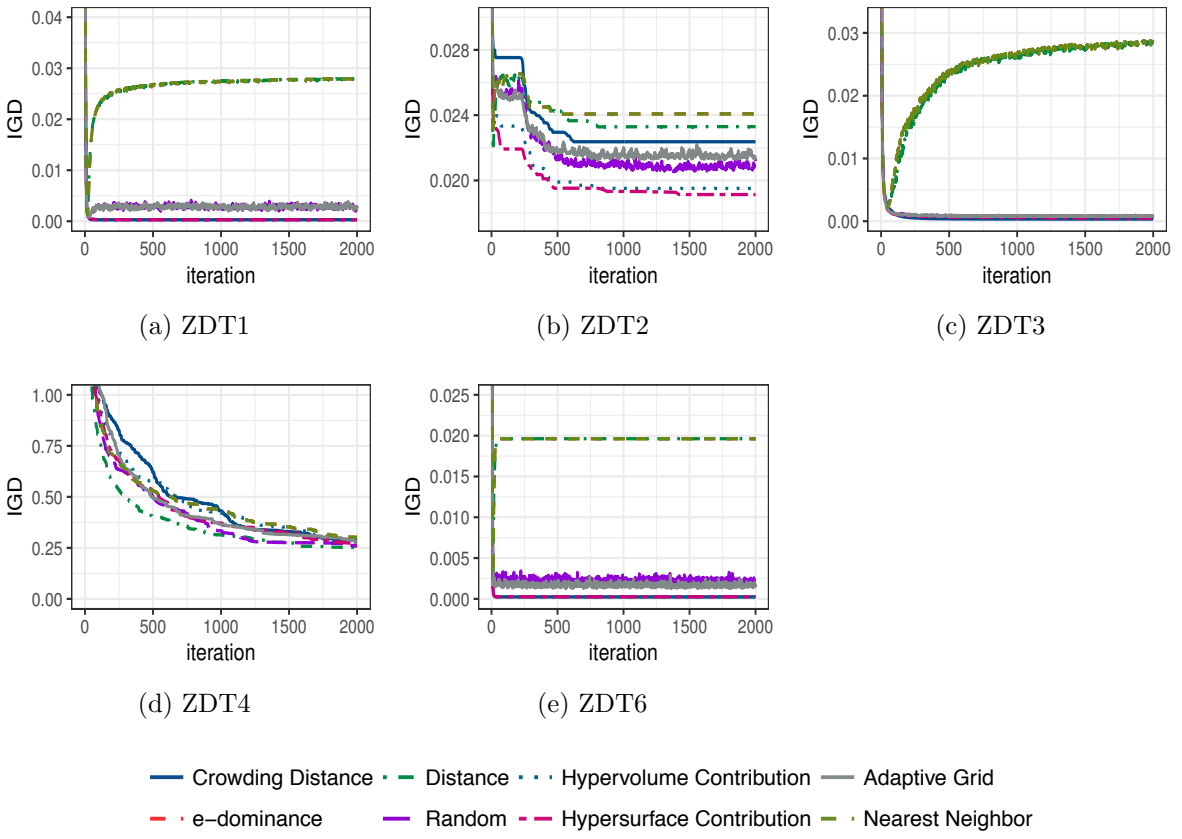
Table 4.1: VEPSO (Random) archive management strategy comparison

Archive	Result	ZDT Function					WFG Function									Overall
		1	**2**	**3**	**4**	**6**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
Crowding Distance	Wins	6	6	5	0	5	7	0	5	6	7	7	5	4	7	70
	Losses	0	0	0	0	1	0	0	0	1	0	0	0	1	0	3
	Difference	6	6	5	0	4	7	0	5	5	7	7	5	3	7	67
	Rank	**1**	**1**	**1**	6	2	**1**	**1**	2	2	**1**	**1**	**1**	2	**1**	**1**
Hypersurface Contribution	Wins	4	6	5	0	4	5	0	3	2	4	6	3	2	4	48
	Losses	1	0	0	0	3	2	0	1	3	2	1	0	2	2	17
	Difference	3	6	5	0	1	3	0	2	-1	2	5	3	0	2	31
	Rank	3	**1**	**1**	6	4	3	**1**	3	6	3	2	3	5	3	3
Hypervolume Contribution	Wins	4	5	4	0	5	4	0	2	2	4	5	2	2	4	43
	Losses	1	2	2	5	1	3	0	3	2	2	2	2	1	2	28
	Difference	3	3	2	-5	4	1	0	-1	0	2	3	0	1	2	15
	Rank	3	3	4	8	2	4	**1**	6	4	3	3	4	3	3	4
Nearest Neighbor	Wins	0	0	0	1	0	1	0	0	0	0	1	0	0	1	4
	Losses	6	6	6	0	6	6	0	6	6	6	6	6	6	6	72
	Difference	-6	-6	-6	1	-6	-5	0	-6	-6	-6	-5	-6	-6	-5	-68
	Rank	7	7	7	**1**	7	7	**1**	7	7	7	7	7	7	7	7
Distance	Wins	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1
	Losses	6	6	6	0	6	7	0	6	6	6	7	6	6	7	75
	Difference	-6	-6	-6	1	-6	-7	0	-6	-6	-6	-7	-6	-6	-7	-74
	Rank	7	7	7	**1**	7	8	**1**	7	7	7	8	7	7	8	8
ϵ-dominance	Wins	4	4	4	1	7	6	0	6	7	6	4	5	7	6	67
	Losses	0	3	0	0	0	1	0	0	0	1	3	0	0	1	9
	Difference	4	1	4	1	7	5	0	6	7	5	1	5	7	5	58
	Rank	2	4	3	**1**	**1**	2	**1**	**1**	**1**	2	4	**1**	**1**	2	2
Adaptive Grid ($c = 0.5$)	Wins	2	2	2	1	2	3	0	2	2	2	2	2	2	2	26
	Losses	4	4	4	0	4	4	0	2	2	4	4	3	1	4	40
	Difference	-2	-2	-2	1	-2	-1	0	0	0	-2	-2	-1	1	-2	-14
	Rank	5	5	5	**1**	5	5	**1**	4	4	5	5	6	3	5	5
Random	Wins	2	2	2	1	2	2	0	2	3	2	2	2	2	2	26
	Losses	4	4	4	0	4	5	0	2	2	4	4	2	2	4	41
	Difference	-2	-2	-2	1	-2	-3	0	0	1	-2	-2	0	0	-2	-15
	Rank	5	5	5	**1**	5	6	**1**	4	3	5	5	4	5	5	5

VEPSO (PCXA)

Figures 4.29(a) through 4.29(e) and 4.30(a) through 4.30(i) depict the average IGD measurement values for ZDT1 through ZDT6 and WFG1 through WFG9 for the VEPSO (PCXA). For 12 of the 14 problems, the IGD measurement values showed notable worse performance for the distance and nearest neighbor AMSs in comparison with all the other AMSs. For 11 of the problems, the IGD measurement value increased after a rapid initial decrease for the distance and nearest neighbor AMSs. For ZDT1 and ZDT6 the adaptive grid and random AMSs performed notably worse than all the other AMSs, excluding the distance and nearest neighbor AMSs which performed overall the worst.

Table 4.2 lists the statistical analysis results. Similar to the VEPSO (Random), the



(a) ZDT1 (b) ZDT2 (c) ZDT3

(d) ZDT4 (e) ZDT6

Figure 4.29: VEPSO (PCXA) Inverted Generational Distance for ZDT1 through ZDT6

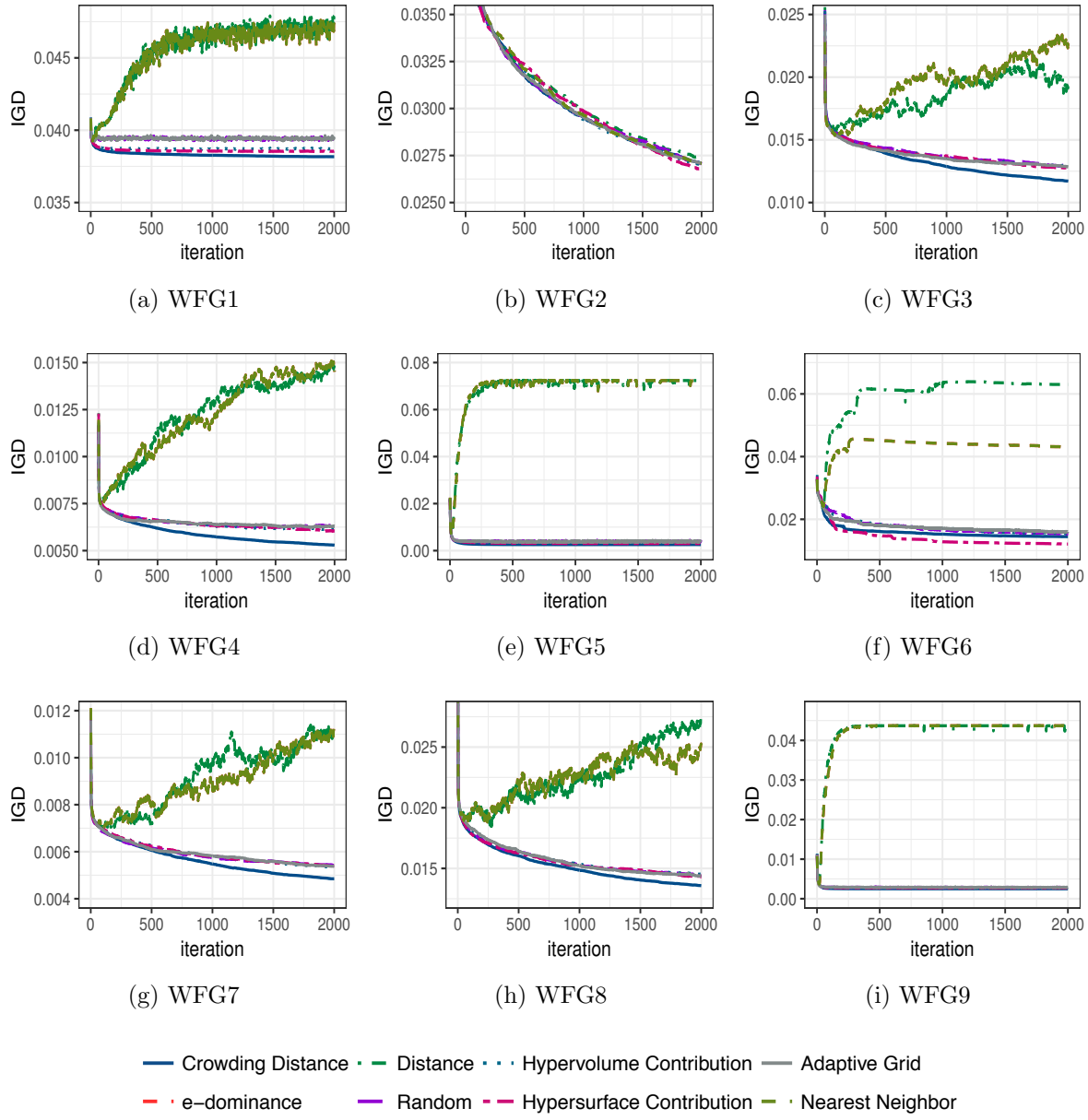


Figure 4.30: VEPSO (PCXA) Inverted Generational Distance for WFG1 through WFG9

statistical analysis showed that the crowding distance AMS performs best, followed by the ϵ-dominance and hypersurface AMSs. The ϵ-dominance AMS ranked first for nine of the 14 problems, three more than the crowding distance AMS. However, the ϵ-dominance

Table 4.2: VEPSO (PCXA) archive management strategy comparison

Archive	Result	ZDT Function					WFG Function									Overall
		1	**2**	**3**	**4**	**6**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
Crowding Distance	Wins	5	1	7	0	5	6	0	6	6	7	2	6	6	7	64
	Losses	2	1	0	0	2	0	0	1	1	0	0	1	1	0	9
	Difference	3	0	7	0	3	6	0	5	5	7	2	5	5	7	55
	Rank	3	5	**1**	**1**	3	**1**	**1**	2	2	**1**	2	2	2	**1**	**1**
Hypersurface Contribution	Wins	6	1	5	0	4	5	0	2	5	5	6	2	2	5	48
	Losses	1	0	1	0	3	2	0	2	2	2	0	2	2	2	19
	Difference	5	1	4	0	1	3	0	0	3	3	6	0	0	3	29
	Rank	2	2	2	**1**	4	3	**1**	3	3	3	**1**	3	3	3	3
Hypervolume Contribution	Wins	4	1	5	0	6	4	0	2	3	4	2	2	2	4	39
	Losses	3	0	1	0	0	3	0	2	3	3	1	2	2	3	23
	Difference	1	1	4	0	6	1	0	0	0	1	1	0	0	1	16
	Rank	4	2	2	**1**	**1**	4	**1**	3	4	4	4	3	3	4	4
Nearest Neighbor	Wins	0	1	0	0	1	1	0	0	0	0	1	0	0	1	5
	Losses	6	1	7	0	6	6	0	7	6	6	6	6	6	6	69
	Difference	-6	0	-7	0	-5	-5	0	-7	-6	-6	-5	-6	-6	-5	-64
	Rank	7	5	8	**1**	7	7	**1**	8	7	7	7	7	7	7	8
Distance	Wins	0	4	1	0	0	0	0	1	0	0	0	0	0	0	6
	Losses	6	0	6	0	7	7	0	6	6	6	7	6	6	7	70
	Difference	-6	4	-5	0	-7	-7	0	-5	-6	-6	-7	-6	-6	-7	-64
	Rank	7	**1**	7	**1**	8	8	**1**	7	7	7	8	7	7	8	7
ϵ-dominance	Wins	7	0	2	0	6	6	0	7	7	6	3	7	7	6	64
	Losses	0	7	5	0	0	0	0	0	0	1	1	0	0	1	15
	Difference	7	-7	-3	0	6	6	0	7	7	5	2	7	7	5	49
	Rank	**1**	8	6	**1**	**1**	**1**	**1**	**1**	**1**	2	2	**1**	**1**	2	2
Adaptive Grid ($c = 0.5$)	Wins	2	1	3	0	2	2	0	2	2	2	2	2	2	2	24
	Losses	4	1	3	0	4	4	0	2	4	4	2	2	2	4	36
	Difference	-2	0	0	0	-2	-2	0	0	-2	-2	0	0	0	-2	-12
	Rank	5	5	4	**1**	5	5	**1**	3	6	5	6	3	3	5	6
Random	Wins	2	1	3	0	2	2	0	2	2	2	2	2	2	2	24
	Losses	4	0	3	0	4	4	0	2	3	4	1	2	2	4	33
	Difference	-2	1	0	0	-2	-2	0	0	-1	-2	1	0	0	-2	-9
	Rank	5	2	4	**1**	5	5	**1**	3	5	5	4	3	3	5	5

AMS did have six more losses than the crowding distance AMS. The ϵ-dominance AMS was also ranked eight for ZDT2 and sixth for ZDT3, a trend that was not visible for any of the WFG problems. The distance and nearest neighbor AMSs again performed

the worst. The difference between the wins and losses for both the distance and nearest neighbor AMSs was -64. The distance AMS did however record one more win for a total of six wins, thus ranking above the nearest neighbor AMS. It is noteworthy that, overall, the adaptive grid AMS ranked sixth, while random AMS ranked fifth, both with 24 wins. The results indicate that the adaptive grid AMS overall performed no better than the random AMS when compared with the other AMSs over all the problems.

Summary

Overall, for both the VEPSO (Random) and VEPSO (PCXA), the crowding distance AMS performed best. The proposed hypersurface contribution AMS showed promise and was only outperformed by the crowding distance and ϵ-dominance AMSs. The hypersurface contribution AMS also outperformed the hypervolume contribution AMS. Further research is required to establish how the various AMSs' performance scale to more objectives. It should be noted that the computational complexity of the hypervolume contribution and hypersurface contribution AMS is notably higher than that of the crowding distance AMS.

The ϵ-dominance AMS ranked second overall; only the crowding distance AMS outperformed the ϵ-dominance AMS. For each of the problems, the ϵ_m value needed to be tuned to store around 50 solutions. This tweaking process can be tedious and requires prior exposure to the problem, making the ϵ-dominance AMS less ideal to use in many situations.

It is noteworthy that, overall, the ϵ-dominance AMS outperformed the adaptive grid AMS. Similar to the adaptive grid AMS, the ϵ-dominance AMS limits the number of solutions using a hypercube approach. In the case of the adaptive grid AMS, the hypercube grid is fixed, whereas in the ϵ-dominance AMS, the hypercube is relative to the objective vector position. In contrast to the adaptive grid AMS, the ϵ-dominance AMS limits the number of solutions within the hypercube to a single solution. Further tuning of the selection pressure parameter, $\ddot{c}$, may increase the performance of the adaptive grid AMS. Similar to the tuning of the ϵ_m parameter, tuning the selection pressure, $\ddot{c}$, is not desirable.

Overall, the results indicated that the distance and nearest neighbor AMSs performed

worse than all other AMSs, including the random AMS. Based on the results presented in this section, it is never advisable to use the distance or nearest neighbor AMSs.

4.6 Summary

This chapter presented an investigation of the effect of various AMSs on the performance of the VEPSO algorithm. The objective was to identify if VEPSO is sensitive towards the choice of AMS and if so, to determine the impact on VEPSO's performance.

Seven different bounded AMSs were reviewed. An eighth AMS, the hypersurface contribution AMS, based on the hypervolume contribution AMS, was introduced.

An analysis of the effect of different AMSs on the diversity of the resulting POF using four diversity measures was presented. The diversity analysis showed that VEPSO is sensitive towards the choice of AMS. A discrepancy was noted in the diversity measurement results. Further investigation revealed that the discrepancy is due to a pairwise grouping weakness that exists in the distribution and spacing measurement calculations. A crowding distribution measurement was introduced to address and confirm the weakness and show how the weakness could be avoided.

Finally, a performance analysis using the well known IGD measure was conducted. The results reaffirmed the diversity analysis results that VEPSO is sensitive towards the choice of AMS. Additionally, the results indicated that the crowding distance AMS was the overall best performing AMS. The newly introduced hypersurface AMS showed promise, ranking third behind the ϵ-dominance, outperforming all the other AMSs. The results also indicated that the distance and nearest neighbor AMSs performed worse than all other AMSs and should thus be avoided. It can be noted that even the random AMS outperformed the distance and nearest neighbor AMSs.

This chapter also showed how little information about a MOO algorithm's performance is revealed by the measurements. In the next chapter, a new comparative measurement, the porcupine measure, is introduced.

Chapter 5

Porcupine Measure

"The empiricist... thinks he believes only what he sees, but he is much better at believing than at seeing."

George Santayana (1865 - 1952)

The previous chapter investigated the effect that the choice of archive implementation has on the performance of VEPSO. The findings highlighted the need for more quantitative performance measures to analyze a MOO algorithm's performance. The main objective of this chapter is to introduce a newly developed performance measure, named the porcupine measure, to quantitatively compare POFs. The new measure is based on attainment surfaces, introduced by Fonseca and Fleming [35] over twenty years ago.

Despite being introduced over twenty years ago, Fonseca and Fleming's attainment surfaces have not been widely used. This chapter also investigates some of the shortcomings that may have led to the lack of adoption of attainment surface based performance measures. The quantitative measure based on attainment surfaces, introduced by Knowles and Corne [67], is analyzed. Improvements to Knowles and Corne's approach for two objective Pareto-optimal front (POF) comparisons are also proposed.

5.1 Introduction

Attainment surfaces provide researchers in MOO with a means to accurately visualize the region dominated by a POF. In many studies, POFs are shown by joining the non-dominated solutions using a curve. Fonseca and Fleming reasoned that using a curve to join the solutions is incorrect. The use of a curve creates a false impression that intermediate solutions exist between any two non-dominated solutions. In reality, there is no guarantee that any intermediate solutions exist. Fonseca and Fleming suggested that instead of a curve, the non-dominated solutions can be used to create an envelope that separates the dominated and non-dominated space. The envelope formed by the non-dominated solutions is referred to as an attainment surface.

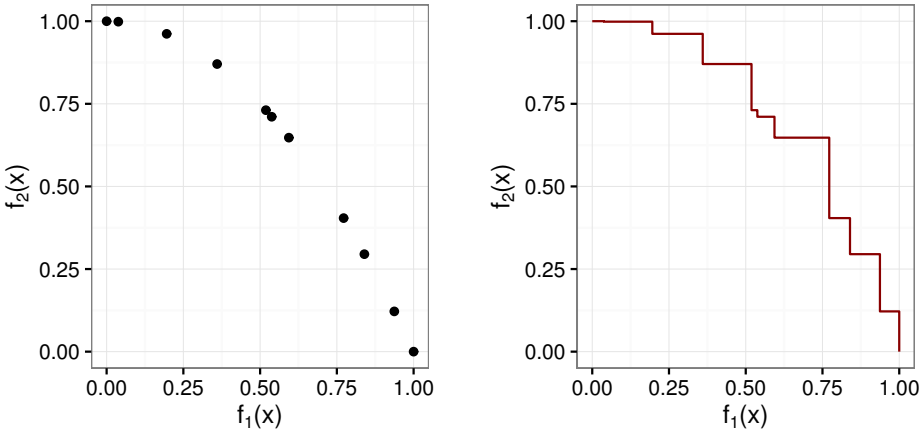
The remainder of this chapter is organized as follows. Section 5.2 presents the background and related work. Two-dimensional attainment surfaces are introduced in Section 5.3, followed by the generalization to n_m-dimensions in Section 5.4. Finally, the findings of this chapter are summarised in Section 5.5.

5.2 Background and Related Work

Fonseca and Fleming [35] suggested that the non-dominated solutions that make up the POF be used to construct an attainment surface. The attainment surface's envelope is defined as the boundary in the objective space that separates those points which are dominated by, or equal to at least one of the non-dominated solutions that make up the POF, from those points which no non-dominated solution dominates or equals. Figure 5.1 depicts an attainment surface and the corresponding POF.

The attainment surface envelope is identical to the envelope used during the calculation of the hypervolume metric [107, 117]. In contrast to the hypervolume calculation, in the case of an attainment surface, the envelope is, however, not used directly in the calculation of a performance metric. Instead, the attainment surface can be used to visually compare algorithms' performance by plotting the attainment surfaces for both algorithms.

For stochastic algorithms, variations in the performance over multiple runs (also referred to as samples) are expected. Fonseca and Fleming [35] described a procedure



(a) Pareto optimal front.

(b) Attainment surface.

Figure 5.1: Example Pareto optimal front and attainment surface.

to generate an attainment surface that represents a given algorithm's performance over multiple independent runs.

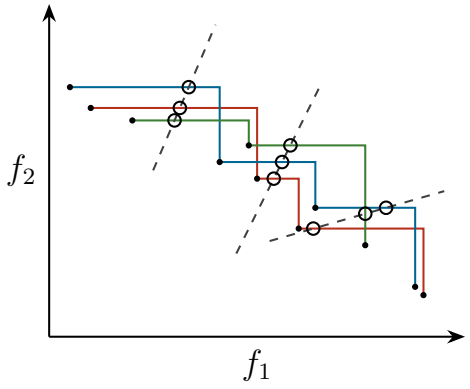
The attainment surface for multiple independent runs is computed by first determining the attainment surface for each run's POF. Next, a number of random imaginary lines is chosen pointing in the direction of improvement for all the objectives. For each line, the intersection points with each of the lines and the attainment surfaces are calculated. Figures 5.2(a) and 5.2(b) depict three attainment surfaces with intersection lines and intersection points.

For each line, the intersection points can be seen as a sample distribution that is uni-dimensional and can thus be strictly ordered. By calculating the median for each of the sample distributions, the objective vectors that are likely to be attained in exactly 50% of the runs can be identified. The envelope formed by the median points is known as the 50% grand attainment surface. Similar to how the median is used to construct the 50% grand attainment surface, the lower and upper quantiles ($25th$ and $75th$ percentiles) are used to construct the 25% and 75% grand attainment surfaces.

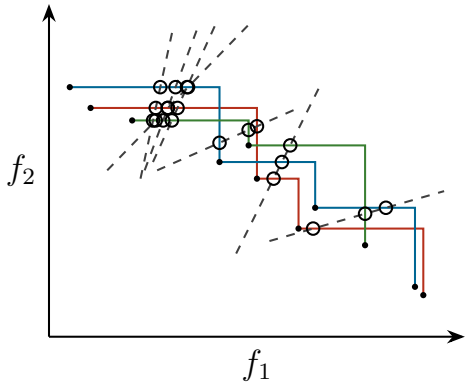
The sample distribution approach can also be used to compare performance between algorithms: In order to compare two algorithms, two sample distributions, one for each of the algorithms, are calculated per intersection line. Standard non-parametric statistical

test procedures can then be used to determine if there is a statistically significant difference between the two sample distributions. Using the statistical test results, a combined grand attainment surface, as depicted in figure 5.2(c), can be constructed showing the regions where each of the algorithms outperforms the other. Fonseca and Fleming [35] suggested that suitable test procedures include the median test, its extensions to other quantiles, and tests of the Kolmogorov-Smirnov type [43].

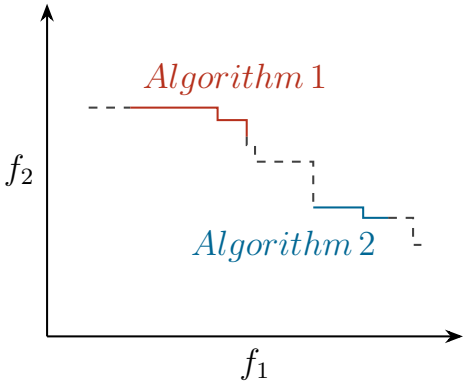
Knowles and Corne [67] extended the work done by Fonseca and Fleming and used attainment surfaces to quantify the performance of their Pareto Archives Evolution Strategy (PAES) algorithm. Knowles and Corne identified four variables in the approach



(a) Example attainment surfaces with intersection lines



(b) Example attainment surfaces with unequally spread intersection lines



(c) Grand attainment surface

Figure 5.2: Attainment surfaces.

proposed by Fonseca and Fleming, namely:

- how many comparison lines should be used,
- where the comparison lines should go,
- which statistical test should be used to compare the univariate distribution,
- and in what form should the results be presented.

Knowles and Corne found that their experimental results indicated that at least 1000 lines should be used. In order to generate the intersection lines, the *minimum* and *maximum* values for each objective over the non-dominated solutions were found. The objective values were then normalized according to the *minimum* and *maximum* values into the range $[0, 1]$. Intersection lines were then generated as equally spread lines from the origin rotated from $(0, 1)$ to $(1, 0)$, effectively rotating $90°$ covering the complete POF.

For n_m-objective function MOPs, where $n_m > 2$, Knowles and Corne suggested using a grid-based approach where points are spread equally on the n_m, $(n_m - 1)$-dimensional hyperplanes. Each hyperplane corresponds to an objective value fixed at the value 1.0. The intersection lines are drawn from the origin to these equally spread out points. In the case of 3 objective function MOPs, a 6×6 grid would result in 108, $(3 \times 6 \times 6)$ points and thus 108 intersection lines. Similarly, using a 16×16 grid on a n_m-objective function MOP would result in 768 intersection lines, and so forth.

For statistical significance testing, Knowles and Corne used the Mann-Whitney U test [43] with a 95% confidence level.

Finally, Knowles and Corne found that a convenient way to report the comparison results were to use simple value pairs $[a, b]$, hereafter referred to as the *KC measure*, where a gives the percentage of space for which algorithm A was found to be statistically superior to algorithm B and b gives the percentage where algorithm B was found to be statistically superior to algorithm A. Note that $100 - (a + b)$ gives the percentage where neither algorithm was found to be statistically superior to the other.

Knowles and Corne [67] generalized the definition of the comparison to compare more than two algorithms. For n_h algorithms, the above comparison is done for all $\binom{n_h}{2}$

algorithm pairs. For each algorithm, h, two percentages are reported: a_h, the region where algorithm h was not beaten by any other algorithm, and b_h, the region where algorithm h beats all the other $(n_h - 1)$ algorithms. Note that $a_h \geq b_h$ as the region described by b_h is contained in the region described by a_h.

Knowles [66] found that visualizing attainment surfaces in three-dimensions was difficult due to the intersection lines not being evenly spread. As an alternative, Knowles presented an algorithm, inspired by the work done by Smith *et al.* [100], to visually draw summary attainment surfaces using axis-aligned lines. The algorithm was found to be particularly well suited for drawing three-dimensional attainment surfaces.

Da Fonseca *et al.* [22] continued work on attainment surfaces by introducing the empirical attainment function (EAF). The EAF is a mean-like, first-order moment measure of the solutions found by a multi-objective optimizer. The EAF allows for intuitive visual comparisons between bi-objective optimization algorithms by plotting the solution probabilities as a heat map [73]. Fonseca *et al.* [36] studied the use of the second-order EAF, which allows for the pairwise relationship between random Pareto-set approximations, to be studied.

It should be noted that calculation of the EAF for three or more dimensions is not trivial [37]. Efficient algorithms to calculate the EAF for two and three dimensions have been proposed [37].

5.3 2-dimensional Attainment Surface

This section presents a discussion of 2-dimensional attainment surfaces and how they are used to calculate the KC measure. A bias in the KC measure calculation is identified and analyzed. Section 5.3.1 presents a discussion of the intersection line generation approach used by Knowles and Corne. A new attainment surface shaped intersection lines (ASSIL) generation approach is presented in Section 5.3.2, followed by a more scalable approach named weighted attainment surface shaped intersection lines (WASSIL) in Section 5.3.3.

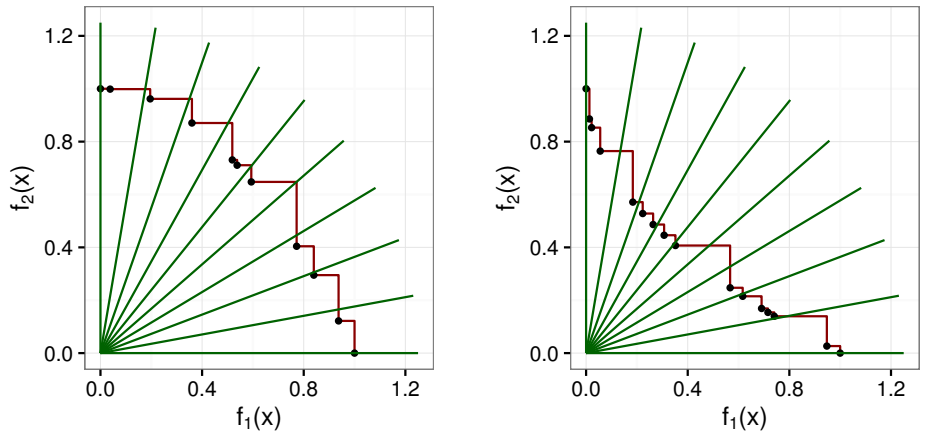
5.3.1 Generating Intersection Lines

The attainment surface calculation approach presented by Fonseca and Fleming [35] did not describe in detail how to generate the intersection lines. Instead, it was only stated that a random number of intersection lines, each pointing in the direction of improvement for all the objectives, should be used. This approach worked well to construct a visualization of the attainment surface.

When Knowles and Corne [67] extended the intersection line approach to build a quantitative comparison measure, they needed the lines to be equally spread. If the lines were not equally spread, as depicted in figure 5.2(b), certain regions of the attainment surface would contribute more than others, leading to misleading results.

Figure 5.3 depicts two example attainment surfaces with rotation-based intersection lines. Figure 5.3(a) depicts a *concave* attainment surface. Visually, the rotation-based intersection lines look to be equally spread. Figure 5.3(b), however, depicts a *convex* attainment surface. Visually, the regions between the intersection lines are larger closer to the objective axes. Clearly, the rotation-based intersection lines are not equally spaced for *convex* shaped fronts.

An evaluation of the rotation-based and random intersection line approaches is pre-



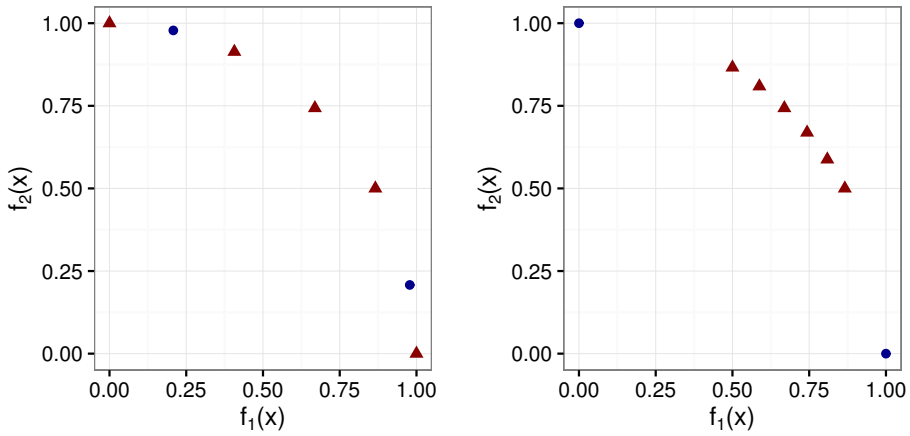
(a) Concave POF (b) Convex POF

Figure 5.3: Attainment surfaces with rotation-based intersection lines.

sented in this section using six artificially generated POF test cases based on the ones used by Knowles and Corne [67]. Figure 5.4 depicts the six artificially generated POF test cases. Each of these artificially generated POF test cases was tested using six POF shape geometries, namely concave, convex, line, mixed, and disconnected. Figure 5.5 depicts the five POF shape geometries. Figures 5.6 through 5.11 depict the POFs with the corresponding attainment surfaces used for the evaluation.

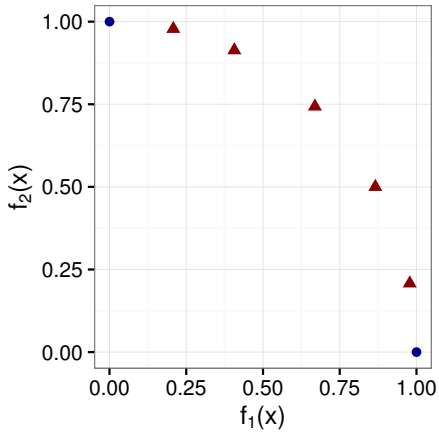
For each of the artificially generated POFs, the true KC measure is known. Table 5.2 summarises the true KC measure and the KC measure with rotation-based and random intersection lines. For each of the approaches, 1000 intersection lines were used for the calculation. Results with a deviation from the true KC measure greater than 5%, indicating less accurate and thus less desirable results, are shown in bold. For POF test cases 1 through 3, only two of the 15 measurements using the random intersection line generation approach had a deviation less than 5%. Overall, 50% of the measurements using the random intersection line generation approach had a deviation greater than 5%. This confirms that the random intersection line generation approach is not well suited for the KC measure calculation.

The rotation-based intersection line generation approach presented by Knowles and Corne fared better than the random intersection line generation approach. Only seven of the 30 measurements using the rotation-based intersection line generation approach had a deviation greater than 5%. Case 1 with a convex POF fared worse with a deviation of almost 15% for both algorithms. Four of the five case 2 measurements using the rotation-based intersection line generation approach had a deviation greater than 5%. For the remaining case 2 measurement, a deviation of at least 3% is noted. The results indicate the rotation-based intersection line generation approach outperformed, with respect to accuracy, the random intersection line generation approach. However, the results also indicate that the rotation-based intersection line generation approach too is not well suited for the KC measure calculation and that the results vary based on the POF shape and the spread of the solutions.

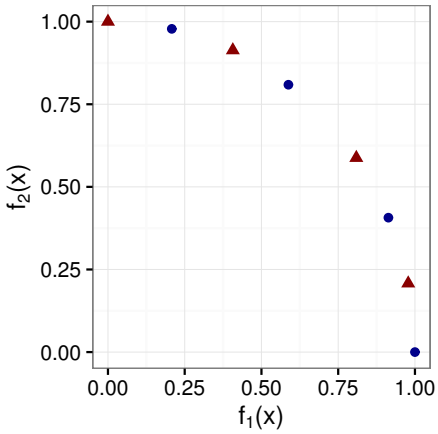


(a) Case 1

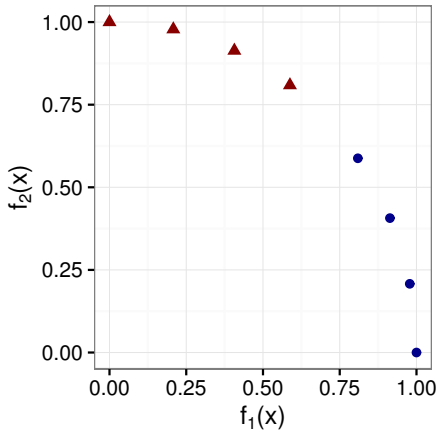
(b) Case 2



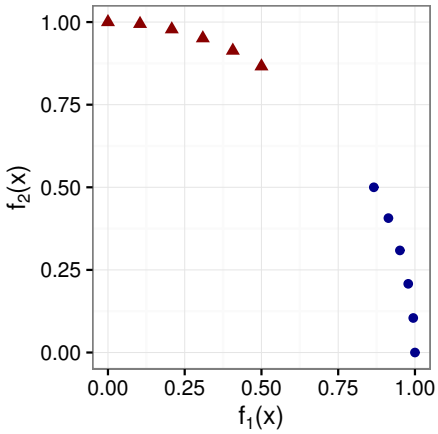
(c) Case 3



(d) Case 4

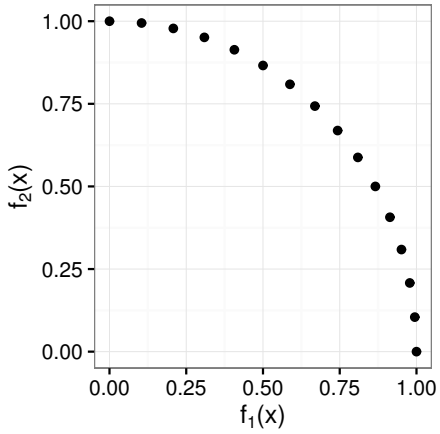


(e) Case 5

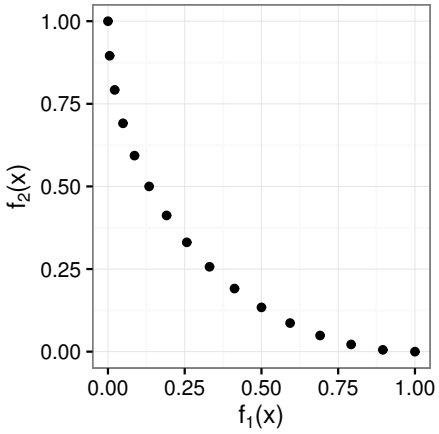


(f) Case 6

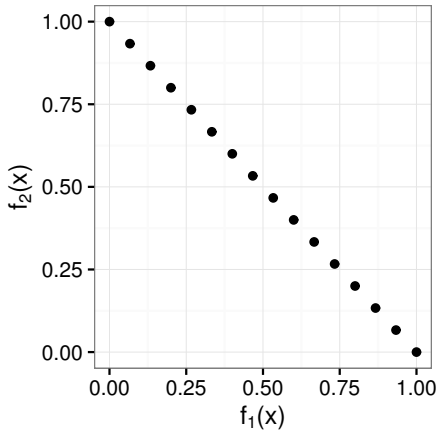
Figure 5.4: Test case Pareto-optimal fronts. Dots represent algorithm A and triangles represent algorithm B.



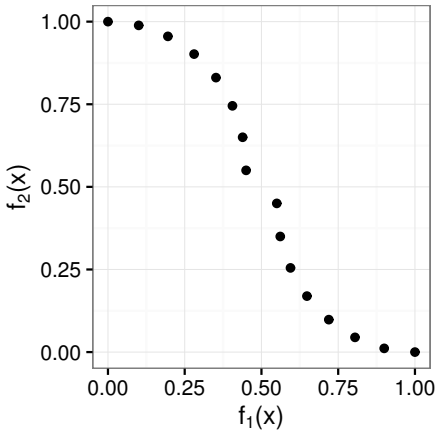
(a) Concave



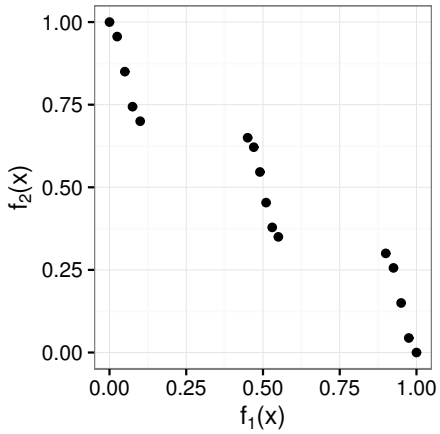
(b) Convex



(c) Line

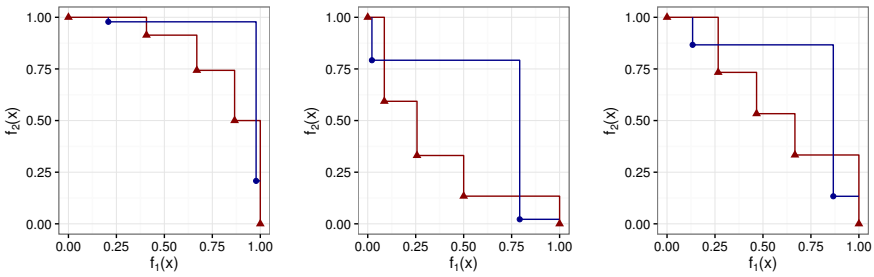


(d) Mixed

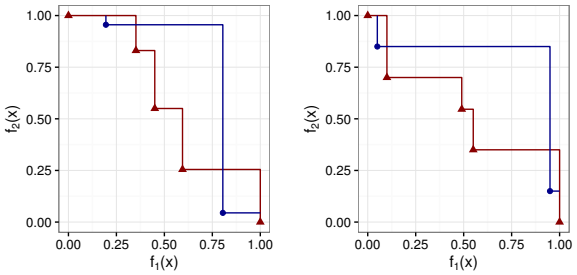


(e) Disconnected

Figure 5.5: Test case Pareto-optimal front geometries.

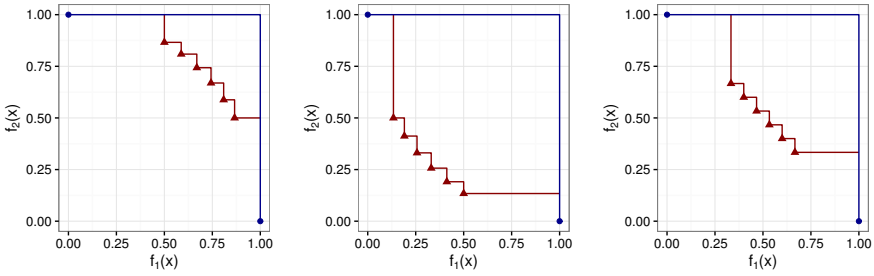


(a) Case 1: Concave (b) Case 1: Convex (c) Case 1: Line

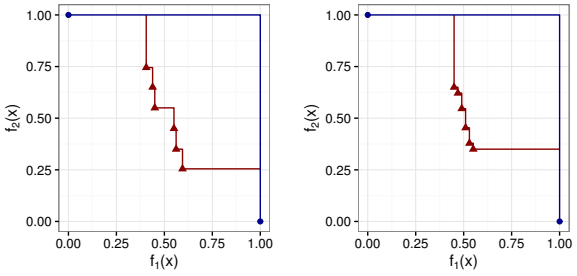


(d) Case 1: Mixed (e) Case 1: Disconnected

Figure 5.6: Pareto-optimal fronts and attainment surfaces for case 1 with various geometries. Dots represent algorithm A and triangles represent algorithm B.

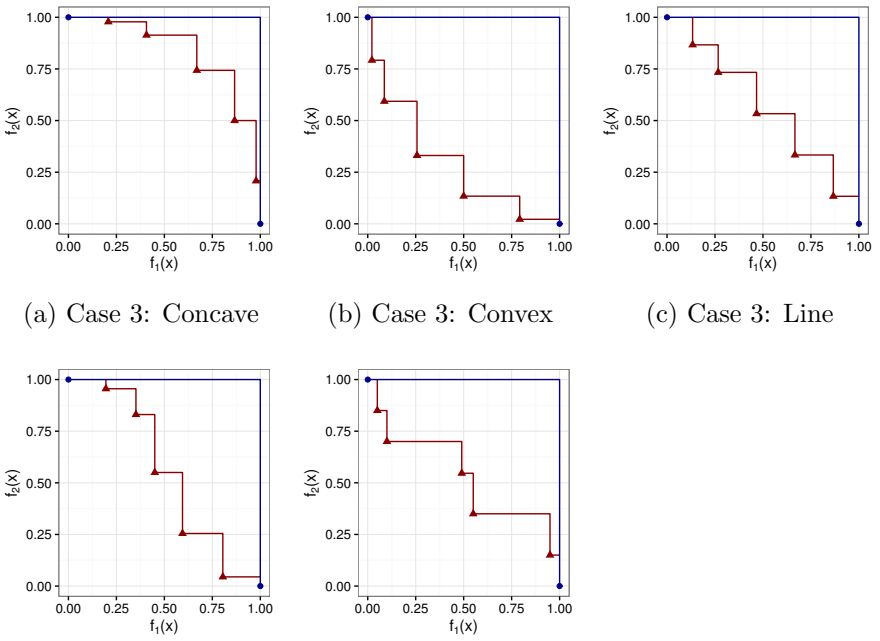


(a) Case 2: Concave (b) Case 2: Convex (c) Case 2: Line



(d) Case 2: Mixed (e) Case 2: Disconnected

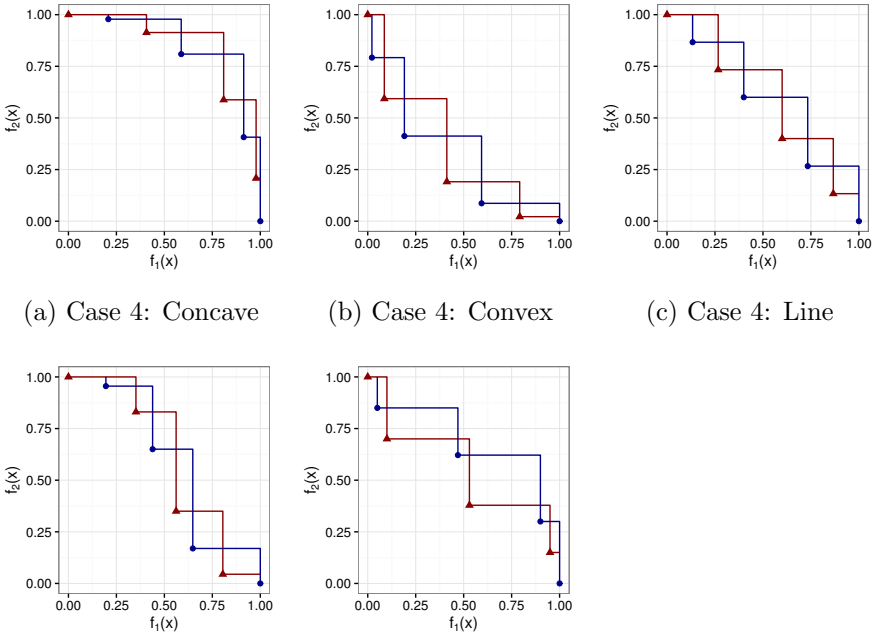
Figure 5.7: Pareto-optimal fronts and attainment surfaces for case 2 with various geometries. Dots represent algorithm A and triangles represent algorithm B.



(a) Case 3: Concave (b) Case 3: Convex (c) Case 3: Line

(d) Case 3: Mixed (e) Case 3: Disconnected

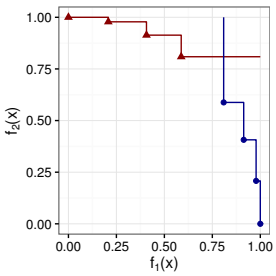
Figure 5.8: Pareto-optimal fronts and attainment surfaces for case 3 with various geometries. Dots represent algorithm A and triangles represent algorithm B.



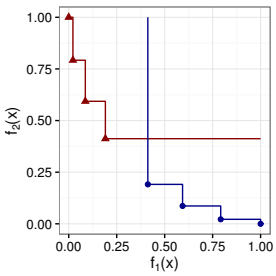
(a) Case 4: Concave (b) Case 4: Convex (c) Case 4: Line

(d) Case 4: Mixed (e) Case 4: Disconnected

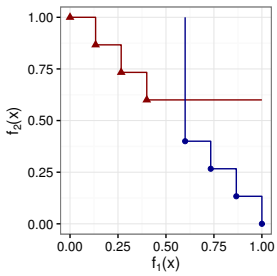
Figure 5.9: Pareto-optimal fronts and attainment surfaces for case 4 with various geometries. Dots represent algorithm A and triangles represent algorithm B.



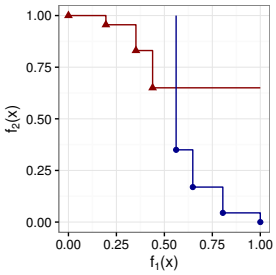
(a) Case 5: Concave



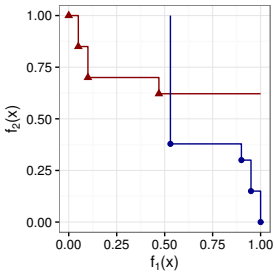
(b) Case 5: Convex



(c) Case 5: Line

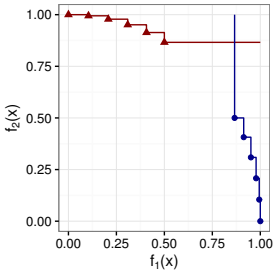


(d) Case 5: Mixed

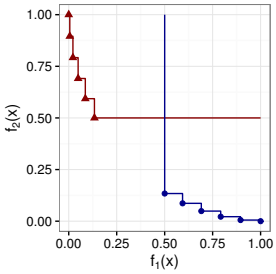


(e) Case 5: Disconnected

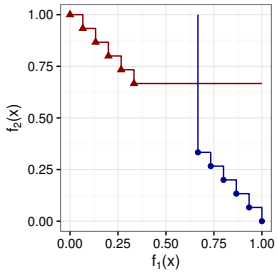
Figure 5.10: Pareto-optimal fronts and attainment surfaces for case 5 with various geometries. Dots represent algorithm A and triangles represent algorithm B.



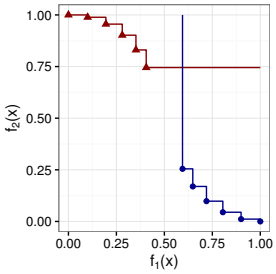
(a) Case 6: Concave



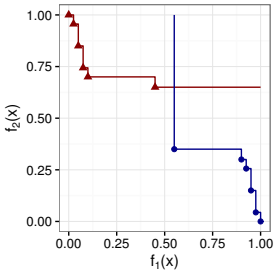
(b) Case 6: Convex



(c) Case 6: Line



(d) Case 6: Mixed



(e) Case 6: Disconnected

Figure 5.11: Pareto-optimal fronts and attainment surfaces for case 6 with various geometries. Dots represent algorithm A and triangles represent algorithm B.

Table 5.1: KC measure with rotation-based and random intersection lines result comparison

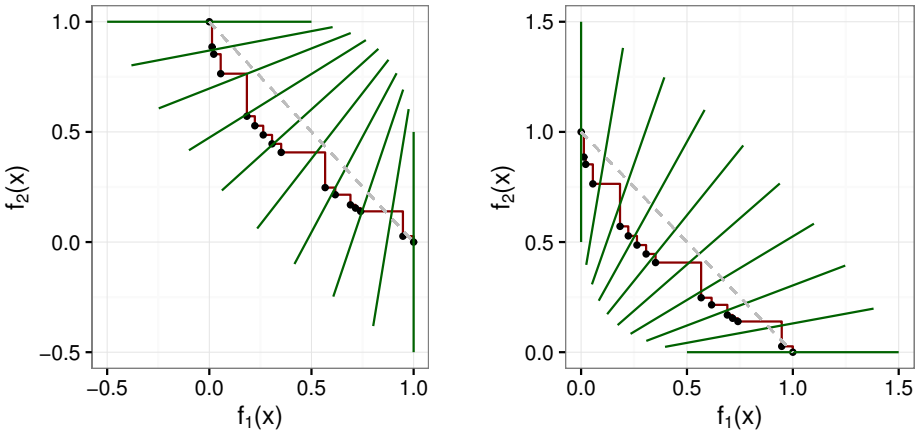
Case	Geometry	True	Intersection Line Generation Approach	
			Rotation-based	Random
Case 1	Concave	[73.27, 26.73]	[71.00, 29.00]	[77.10, 22.90]
	Convex	[70.37, 29.63]	[**85.10**, **14.90**]	[**84.40**, **15.60**]
	Line	[70.00, 30.00]	[74.60, 25.40]	[**77.80**, **22.20**]
	Mixed	[69.67, 30.33]	[73.20, 26.80]	[**78.50**, **21.50**]
	Disconnected	[77.50, 22.50]	[**82.70**, **17.30**]	[**86.40**, **13.60**]
Case 2	Concave	[50.00, 50.00]	[**41.00**, **59.00**]	[**64.60**, **35.40**]
	Convex	[86.60, 13.40]	[83.00, 17.00]	[**96.40**, **3.60**]
	Line	[66.67, 33.33]	[**59.00**, **41.00**]	[**82.20**, **17.80**]
	Mixed	[66.99, 33.01]	[**59.60**, **40.40**]	[**81.80**, **18.20**]
	Disconnected	[60.00, 40.00]	[**51.60**, **48.40**]	[**73.80**, **26.20**]
Case 3	Concave	[79.21, 20.79]	[**73.80**, **26.20**]	[**92.10**, **7.90**]
	Convex	[97.81, 2.19]	[97.20, 2.80]	[99.90, 0.10]
	Line	[86.67, 13.33]	[83.00, 17.00]	[**97.30**, **2.70**]
	Mixed	[88.01, 11.99]	[84.80, 15.20]	[**96.40**, **3.60**]
	Disconnected	[90.00, 10.00]	[87.30, 12.70]	[**97.30**, **2.70**]
Case 4	Concave	[50.00, 50.00]	[50.00, 50.00]	[50.90, 49.10]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[52.30, 47.70]
	Line	[50.00, 50.00]	[50.00, 50.00]	[49.10, 50.90]
	Mixed	[55.71, 44.29]	[54.30, 45.70]	[54.00, 46.00]
	Disconnected	[69.14, 30.86]	[73.00, 27.00]	[**78.70**, **21.30**]
Case 5	Concave	[50.00, 50.00]	[50.00, 50.00]	[49.60, 50.40]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[49.60, 50.40]
	Mixed	[45.56, 54.44]	[45.30, 54.70]	[40.70, 59.30]
	Disconnected	[45.43, 54.57]	[45.00, 55.00]	[41.30, 58.70]
Case 6	Concave	[50.00, 50.00]	[50.00, 50.00]	[52.40, 47.60]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[47.80, 52.20]
	Line	[50.00, 50.00]	[50.00, 50.00]	[48.00, 52.00]
	Mixed	[42.47, 57.53]	[42.90, 57.10]	[38.20, 61.80]
	Disconnected	[45.00, 55.00]	[44.70, 55.30]	[43.50, 56.50]

5.3.2 2-dimensional Attainment Surface Shaped Intersection Lines

In order to address the unequal spacing of the rotation-based intersection lines, a new approach to place the intersection lines are proposed. To compensate for the shape of the front, the intersection lines can be positioned on a line, pointing either *inward* or *outward*, based on the shape of the attainment surfaces being compared. The line connects the extreme values of the attainment surfaces. Figure 5.12 depicts the *inward* and *outward* intersection line approaches for a *convex* shaped front. The regions are clearly more equally spread for the *inward* intersection line approach.

However, the direction of the intersection lines is less desirable for comparison purposes. At the edges, the intersection lines are parallel with the opposite objective's axis. Intuitively, it is more desirable that the intersection lines should be parallel to the closest objective's axis. Another disadvantage of the inward and outward approaches is that the approach to be selected depends on the shape of the front, which is typically unknown. For attainment surfaces that are not fully *convex* or *concave* neither approach is suitable.

An alternative approach referred to as ASSIL, is to generate the intersection lines



(a) Inward (b) Outward

Figure 5.12: Attainment surfaces with outward/inward intersection lines.

along the shape of the attainment surface. In order to equally spread the intersection lines, the Manhattan distance is used to calculate equal spacings for the intersection lines along the attainment surface. Figure 5.13 depicts the Manhattan distance calculation between two points on the POF, $\mathbf{q}_3$ and $\mathbf{q}_1$ in this case. ASSIL is generated as described in Algorithm 5.

Algorithm 5 Attainment surface shaped intersection line generation

Let $Q = \{\mathbf{q}_k : k \in \{1, 2, ..., n_q\}\}$ be the optimal POF with n_q solutions and $\mathbf{q}_k = (q_{k1}, q_{k2})$;
Let $k = 1$;
Let n_l be the desired number of intersection lines;
for $u_l = 1..n_l$ **do**
  Let $d = (u_l - 1) \times \frac{(q_{n_q 1} - q_{11}) + (q_{12} - q_{n_q 2})}{n_l - 1}$;
  while $d < (q_{k1} - q_{11}) + (q_{12} - q_{k2})$ **do**
    $k = k + 1$;
  end while
  if $d = (q_{k1} - q_{11}) + (q_{12} - q_{k2})$ **then**
    Let $\hat{\mathbf{q}} = \mathbf{q}_k$;
  else if $d \leq (q_{(k+1)1} - q_{11}) + (q_{12} - q_{k2})$ **then**
    Let $\hat{\mathbf{q}} = (q_{11} + (d - (q_{12} - q_{k2})), q_{k2})$;
  else
    Let $\hat{\mathbf{q}} = (q_{(k+1)1}, q_{12} - (d - (q_{(k+1)1} - q_{11})))$;
  end if
  Let $\theta = \frac{d}{(q_{n_q 1} - q_{11}) + (q_{12} - q_{n_q 2})} \times \frac{\pi}{2}$;
  // Finally, draw the generated intersection line
  drawIntersectionLine($from = (\hat{q}_1 - sin\theta, \hat{q}_2 - cos\theta)$, $to = (\hat{q}_1 + sin\theta, \hat{q}_2 + cos\theta)$);
end for

Figure 5.14 depicts the ASSIL approach. Generating the intersection lines along the shape of the attainment surface allows for an equal spacing of the intersection lines independent of the shape of the front. For all shapes that the attainment surface can assume, be it *convex*, *concave* or mixed, the intersection lines are equally spread out.

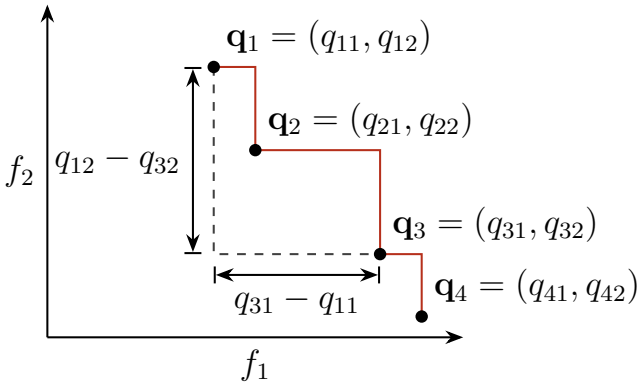
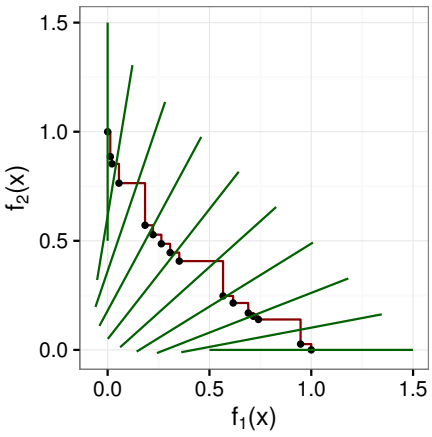
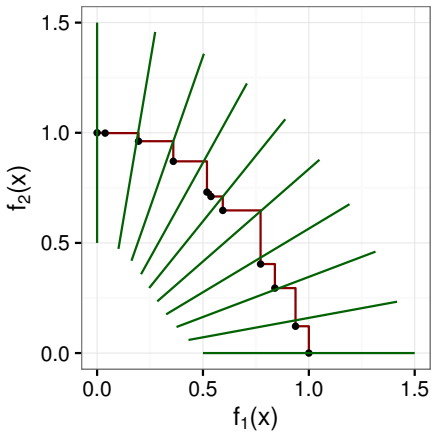


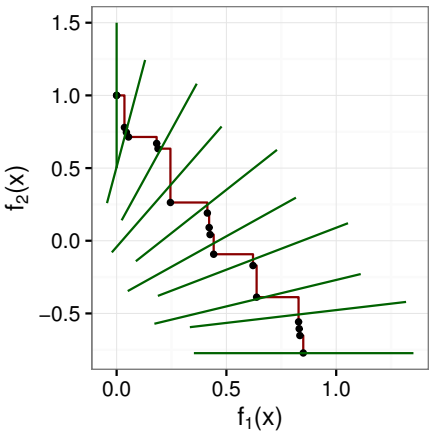
Figure 5.13: Attainment surface with Manhattan distance calculations.



(a) Convex POF



(b) Concave POF



(c) Disconnected POF

Figure 5.14: Attainment surfaces with unbiased ASSIL.

The proposed KC measure using ASSIL is compared against the KC measure using rotation-based and randomly generated intersection lines. The KC measure is calculated as shown in Algorithm 6.

Algorithm 6 KC measure

Let $total = 0$;
Let $wins_A = 0$;
Let $wins_B = 0$;
for each intersection line l **do**
    Let O be the strict ordering of the intersection points for algorithms A and B on intersection line l;
    Let $O_A \subset O$ be the ordering of the intersection points for algorithm A on intersection line l;
    Let $O_B \subset O$ be the ordering of the intersection points for algorithm B on intersection line l;
    if O_A is statistically significantly less than O_B **then**
        $wins_A = wins_A + 1$;
    else if O_B is statistically significantly less than O_A **then**
        $wins_B = wins_B + 1$;
    end if
    $total = total + 1$;
end for
Return $[\frac{100\, wins_A}{total}, \frac{100\, wins_B}{total}]$;

Table 5.2 summarises the true KC measure, the KC measure with rotation-based and random intersection lines, and the KC measure with ASSIL results. For each of the approaches, 1000 intersection lines were used for the calculation.

As expected, the ASSIL generation approach produced results much closer to the true KC measure: The closer the POFs being compared are to the true POF, the more accurate the comparison using the ASSIL generation approach becomes.

Tables 5.3 and 5.4 present a comparison of the varying results obtained from using the various intersection line generation approaches. The tables show comparisons using

Table 5.2: KC measure with ASSIL result comparison

Case	Geometry	True	Intersection Line Generation Approach		
			Rotation-based	Random	ASSIL
Case 1	Concave	[73.27, 26.73]	[71.00, 29.00]	[76.80, 23.20]	[73.20, 26.80]
	Convex	[70.37, 29.63]	[**85.10**, **14.90**]	[**85.60**, **14.40**]	[70.30, 29.70]
	Line	[70.00, 30.00]	[74.60, 25.40]	[**79.30**, **20.70**]	[70.00, 30.00]
	Mixed	[69.67, 30.33]	[73.20, 26.80]	[**82.10**, **17.90**]	[69.80, 30.20]
	Disconnected	[77.50, 22.50]	[**82.70**, **17.30**]	[**86.90**, **13.10**]	[77.50, 22.50]
Case 2	Concave	[50.00, 50.00]	[**41.00**, **59.00**]	[**67.60**, **32.40**]	[50.00, 50.00]
	Convex	[86.60, 13.40]	[83.00, 17.00]	[**96.40**, **3.60**]	[86.60, 13.40]
	Line	[66.67, 33.33]	[**59.00**, **41.00**]	[**81.30**, **18.70**]	[66.60, 33.40]
	Mixed	[66.99, 33.01]	[**59.60**, **40.40**]	[**81.60**, **18.40**]	[66.90, 33.10]
	Disconnected	[60.00, 40.00]	[**51.60**, **48.40**]	[**75.80**, **24.20**]	[60.00, 40.00]
Case 3	Concave	[79.21, 20.79]	[**73.80**, **26.20**]	[**92.60**, **7.40**]	[79.20, 20.80]
	Convex	[97.81, 2.19]	[97.20, 2.80]	[99.90, 0.10]	[97.80, 2.20]
	Line	[86.67, 13.33]	[83.00, 17.00]	[**96.50**, **3.50**]	[86.60, 13.40]
	Mixed	[88.01, 11.99]	[84.80, 15.20]	[**95.60**, **4.40**]	[87.90, 12.10]
	Disconnected	[90.00, 10.00]	[87.30, 12.70]	[**97.60**, **2.40**]	[90.00, 10.00]
Case 4	Concave	[50.00, 50.00]	[50.00, 50.00]	[49.00, 51.00]	[50.00, 50.00]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[50.60, 49.40]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[54.00, 46.00]	[50.00, 49.90]
	Mixed	[55.71, 44.29]	[54.30, 45.70]	[**50.70**, **49.30**]	[55.70, 44.30]
	Disconnected	[69.14, 30.86]	[73.00, 27.00]	[**77.80**, **22.20**]	[69.10, 30.90]
Case 5	Concave	[50.00, 50.00]	[50.00, 50.00]	[49.50, 50.50]	[50.00, 50.00]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[50.40, 49.60]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[49.20, 50.80]	[50.00, 50.00]
	Mixed	[45.56, 54.44]	[45.30, 54.70]	[43.40, 56.60]	[45.60, 54.40]
	Disconnected	[45.43, 54.57]	[45.00, 55.00]	[45.40, 54.60]	[45.40, 54.60]
Case 6	Concave	[50.00, 50.00]	[50.00, 50.00]	[48.00, 52.00]	[50.00, 50.00]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[49.50, 50.50]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[49.30, 50.70]	[50.00, 50.00]
	Mixed	[42.47, 57.53]	[42.90, 57.10]	[37.70, 62.30]	[42.50, 57.50]
	Disconnected	[45.00, 55.00]	[44.70, 55.30]	[44.10, 55.90]	[45.00, 55.00]

Table 5.3: Intersection line comparison between VEPSO ($\mathcal{V}$), SMPSO ($\mathcal{S}$), and OMOPSO ($\mathcal{O}$)

Problem	Intersections	$\mathcal{V}$ vs $\mathcal{O}$	$\mathcal{V}$ vs $\mathcal{S}$	$\mathcal{S}$ vs $\mathcal{O}$
ZDT1	Rotational	[**14.9**, **66.4**]	[**6.1**, **80.6**]	[**79.4**, 1.1]
	Inward	[25.6, 55.5]	[13.8, **65.4**]	[70.2, 3.9]
	Outward	[19.8, 62.7]	[8.7, 75.6]	[77.3, 1.6]
	ASSIL	[22.5, 60.1]	[11.4, 72.2]	[72.5, 4.3]
ZDT2	Rotational	[8.4, 64.3]	[2.9, 73.6]	[58.9, 3.9]
	Inward	[8.0, 63.4]	[2.0, 74.3]	[56.3, 0.9]
	Outward	[10.4, 62.7]	[4.2, 70.5]	[60.1, 5.7]
	ASSIL	[9.0, 63.9]	[3.4, 72.6]	[60.0, 3.8]
ZDT3	Rotational	[13.4, **77.9**]	[3.9, 90.8]	[**81.4**, 7.3]
	Inward	[**7.1**, **61.6**]	[2.1, **83.2**]	[72.4, 10.5]
	Outward	[16.4, 73.2]	[4.9, 87.7]	[78.4, 8.7]
	ASSIL	[12.5, 72.9]	[3.2, 89.3]	[74.8, 9.5]
ZDT4	Rotational	[0.0, 99.9]	[0.0, 99.9]	[**99.9**, 0.0]
	Inward	[0.0, **87.9**]	[0.0, **88.1**]	[**80.9**, 0.0]
	Outward	[0.0, 100.0]	[0.0, 100.0]	[**100.0**, 0.0]
	ASSIL	[0.0, 100.0]	[0.0, 97.7]	[93.7, 0.0]
ZDT6	Rotational	[**40.1**, **13.3**]	[15.2, **25.9**]	[**58.9**, 11.1]
	Inward	[**64.6**, 2.8]	[15.6, **35.3**]	[**73.3**, **0.7**]
	Outward	[**50.6**, 7.8]	[6.1, **52.9**]	[**64.5**, **5.0**]
	ASSIL	[59.4, 4.3]	[10.9, 41.8]	[52.3, 14.1]

the VEPSO, OMOPSO and SMPSO algorithms using the ZDT and 2-objective WFG test sets.

Variations in the results between the different intersection line generation approaches can be seen. ZDT1, ZDT3, ZDT4, and ZDT6 all show variations in the results greater than 5% for each of the comparisons. WFG1, WFG2, WFG5, WFG6, and WFG9 all show variations in the results greater than 5% for at least one of the comparisons. The variations are indicative of the bias towards certain attainment surface shapes shown by the various intersection line generation approaches. The results indicate that the KC measure with ASSIL is well suited for POF comparisons.

Table 5.4: Intersection line comparison between VEPSO ($\mathcal{V}$), SMPSO ($\mathcal{S}$), and OMOPSO ($\mathcal{O}$)

Problem	Intersections	$\mathcal{V}$ vs $\mathcal{O}$	$\mathcal{V}$ vs $\mathcal{S}$	$\mathcal{S}$ vs $\mathcal{O}$
WFG1	Rotational	[0.0, 99.9]	[0.2, 96.9]	[28.6, 65.8]
	Inward	[0.0, 100.0]	[0.2, 97.6]	[**38.4**, **55.6**]
	Outward	[0.0, 99.9]	[0.2, 97.5]	[26.6, 68.0]
	ASSIL	[0.0, 99.9]	[0.2, 97.5]	[27.2, 67.2]
WFG2	Rotational	[0.0, 99.9]	[0.0, 99.9]	[0.0, **65.5**]
	Inward	[0.0, 100.0]	[0.0, 100.0]	[0.0, **86.6**]
	Outward	[0.0, 99.9]	[0.0, 99.9]	[0.0, **63.9**]
	ASSIL	[0.0, 99.9]	[0.0, 99.9]	[0.0, 73.1]
WFG3	Rotational	[0.0, 99.9]	[0.0, 99.9]	[0.0, 88.8]
	Inward	[0.0, 100.0]	[0.0, 100.0]	[0.0, 87.3]
	Outward	[0.0, 99.9]	[0.0, 99.9]	[0.0, 89.2]
	ASSIL	[0.0, 99.9]	[0.0, 99.9]	[0.0, 88.8]
WFG4	Rotational	[0.0, 99.9]	[0.0, 99.9]	[99.9, 0.0]
	Inward	[0.0, 100.0]	[0.0, 100.0]	[100.0, 0.0]
	Outward	[0.0, 99.9]	[0.0, 99.9]	[99.9, 0.0]
	ASSIL	[0.0, 99.9]	[0.0, 99.9]	[99.9, 0.0]
WFG5	Rotational	[16.9, 20.0]	[0.1, 62.7]	[52.4, 2.0]
	Inward	[**10.1**, 21.0]	[0.2, 59.8]	[**39.9**, 0.5]
	Outward	[18.6, 23.4]	[0.2, 64.4]	[55.0, 2.9]
	ASSIL	[16.2, 19.1]	[0.2, 61.1]	[50.3, 1.7]
WFG6	Rotational	[25.6, 13.5]	[0.0, 99.9]	[99.9, 0.0]
	Inward	[**9.9**, 13.5]	[0.0, 100.0]	[100.0, 0.0]
	Outward	[**29.6**, 13.7]	[0.0, 99.9]	[99.9, 0.0]
	ASSIL	[23.3, 13.7]	[0.0, 99.9]	[99.9, 0.0]
WFG7	Rotational	[0.0, 99.9]	[0.0, 99.9]	[0.0, 92.7]
	Inward	[0.0, 100.0]	[0.0, 100.0]	[0.0, 95.4]
	Outward	[0.0, 99.9]	[0.0, 99.9]	[0.0, 92.6]
	ASSIL	[0.0, 99.9]	[0.0, 99.9]	[0.0, 93.2]
WFG8	Rotational	[0.0, 99.9]	[0.0, 99.9]	[0.0, 94.6]
	Inward	[0.0, 100.0]	[0.0, 100.0]	[0.0, 95.8]
	Outward	[0.0, 99.9]	[0.0, 99.9]	[0.0, 94.4]
	ASSIL	[0.0, 99.9]	[0.0, 99.9]	[0.0, 95.1]
WFG9	Rotational	[8.0, 30.3]	[0.0, 99.9]	[99.9, 0.0]
	Inward	[2.6, **41.1**]	[0.0, 100.0]	[100.0, 0.0]
	Outward	[8.8, 27.9]	[0.0, 99.9]	[99.9, 0.0]
	ASSIL	[7.1, 31.5]	[0.0, 99.9]	[99.9, 0.0]

5.3.3 Weighted 2-dimensional Attainment Surface Shaped Intersection Lines

As an alternative to the equally spread intersection lines used by ASSIL, intersection lines can be generated along the shape of the POF, at least one intersection line per attainment surface line segment. Because the attainment surface segments are not all of equal lengths, a weight is associated with each intersection line to balance the KC measure result. The WASSIL generation algorithm is given in Algorithm 7.

Algorithm 7 Weighted attainment surface shaped intersection line generation

1: Let $Q = \{\mathbf{q_k} : k \in \{1, 2, ..., n_q\}\}$ be the optimal POF with n_q solutions and $\mathbf{q_k} = (q_{k1}, q_{k2})$;
2: **for** each attainment line segment s_l **do**
3: Let $w_l = length(s_l)$;
// If w_l exceeds a predefined threshold, s_l can be subdivided to increase the
// measurement's accuracy
4: Let $\hat{\mathbf{c}}$ be the center of s_l;
5: Let $\theta = \frac{\pi(c_1 + (\max(q_{k2}) - c_2))}{2d}$ with $k \in \{1, 2, ..., n_q\}$;
6: Let $\hat{\mathbf{p}} = (\sin\theta, \cos\theta)$;
// Finally, draw the generated intersection line from ($\hat{\mathbf{c}} - \hat{\mathbf{p}}$) to ($\hat{\mathbf{c}} + \hat{\mathbf{p}}$)
7: drawIntersectionLineWithWeight($from = \hat{\mathbf{c}} - \hat{\mathbf{p}}$, $to = \hat{\mathbf{c}} + \hat{\mathbf{p}}$, $weight = w_l$);
8: **end for**

Figure 5.15 depicts a convex POF with WASSIL generated intersection lines. The figure clearly shows that the intersection lines are positioned along the attainment surface, and due to the positioning, the lines are angled slightly differently from the intersection lines in figure 5.3(b). The WASSIL algorithm should, for the test cases, result in a weighted KC measure result that matches the true KC measure result.

Note that the weighted KC measure is calculated as shown in Algorithm 8.

Algorithm 8 Weight adjusted KC measure

Let $w_{total} = 0$;
Let $w_A = 0$;
Let $w_B = 0$;
for each intersection line l **do**
    Let w_l be the weight associated with l;
    Let O be the strict ordering of the intersection points for algorithms A and B on intersection line l;
    Let $O_A \subset O$ be the ordering of the intersection points for algorithm A on intersection line l;
    Let $O_B \subset O$ be the ordering of the intersection points for algorithm B on intersection line l;
    if O_A is statistically significantly less than O_B **then**
        $w_A = w_A + w_l$;
    else if O_B is statistically significantly less than O_A **then**
        $w_B = w_B + w_l$;
    end if
    $w_{total} = w_{total} + w_l$;
end for
Return $[\frac{100\,w_A}{w_{total}}, \frac{100\,w_B}{w_{total}}]$;

Table 5.5 summarises the true KC measure, the KC measure with rotation-based and random intersection lines, the KC measure with ASSIL, and the KC measure with WASSIL results. For each of the approaches, 1000 intersection lines were used for the calculation.

Results with a deviation from the true KC measure greater than 5%, indicating less accurate and thus less desirable results, are shown in bold. For POF test cases 1 through 3 only two of the 15 measurements using the random intersection line generation approach had a deviation less than 5%. Overall, 50% of the measurements using the random intersection line generation approach had a deviation greater than 5%. This confirms that the random intersection line generation approach is not well suited for the KC

Table 5.5: KC measure with WASSIL result comparison

Case	Geometry	True	Intersection Line Generation Approach			
			Rotation-based	Random	ASSIL	WASSIL
Case 1	Concave	[73.27, 26.73]	[71.00, 29.00]	[**79.90**, **20.10**]	[73.20, 26.80]	[73.27, 26.73]
	Convex	[70.37, 29.63]	[**85.10**, **14.90**]	[**87.40**, **12.60**]	[70.30, 29.70]	[70.37, 29.63]
	Line	[70.00, 30.00]	[74.60, 25.40]	[**78.80**, **21.20**]	[70.00, 30.00]	[70.00, 30.00]
	Mixed	[69.67, 30.33]	[73.20, 26.80]	[**78.30**, **21.70**]	[69.80, 30.20]	[69.67, 30.33]
	Disconnected	[77.50, 22.50]	[**82.70**, **17.30**]	[**87.60**, **12.40**]	[77.50, 22.50]	[77.50, 22.50]
Case 2	Concave	[50.00, 50.00]	[**41.00**, **59.00**]	[**67.00**, **33.00**]	[50.00, 50.00]	[50.00, 50.00]
	Convex	[86.60, 13.40]	[83.00, 17.00]	[**97.20**, **2.80**]	[86.60, 13.40]	[86.60, 13.40]
	Line	[66.67, 33.33]	[**59.00**, **41.00**]	[**85.60**, **14.40**]	[66.60, 33.40]	[66.67, 33.33]
	Mixed	[66.99, 33.01]	[**59.60**, **40.40**]	[**82.10**, **17.90**]	[66.90, 33.10]	[66.99, 33.01]
	Disconnected	[60.00, 40.00]	[**51.60**, **48.40**]	[**77.40**, **22.60**]	[60.00, 40.00]	[60.00, 40.00]
Case 3	Concave	[79.21, 20.79]	[**73.80**, **26.20**]	[**93.20**, **6.80**]	[79.20, 20.80]	[79.21, 20.79]
	Convex	[97.81, 2.19]	[97.20, 2.80]	[100.00, 0.00]	[97.80, 2.20]	[97.81, 2.19]
	Line	[86.67, 13.33]	[83.00, 17.00]	[**96.60**, **3.40**]	[86.60, 13.40]	[86.67, 13.33]
	Mixed	[88.01, 11.99]	[84.80, 15.20]	[**95.70**, **4.30**]	[87.90, 12.10]	[88.01, 11.99]
	Disconnected	[90.00, 10.00]	[87.30, 12.70]	[**97.50**, **2.50**]	[90.00, 10.00]	[90.00, 10.00]
Case 4	Concave	[50.00, 50.00]	[50.00, 50.00]	[49.40, 50.60]	[50.00, 50.00]	[50.00, 50.00]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[48.70, 51.30]	[50.00, 50.00]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[52.30, 47.70]	[50.00, 49.90]	[50.00, 50.00]
	Mixed	[55.71, 44.29]	[54.30, 45.70]	[51.00, 49.00]	[55.70, 44.30]	[55.71, 44.29]
	Disconnected	[69.14, 30.86]	[73.00, 27.00]	[**76.70**, **23.30**]	[69.10, 30.90]	[69.14, 30.86]
Case 5	Concave	[50.00, 50.00]	[50.00, 50.00]	[48.90, 51.10]	[50.00, 50.00]	[50.00, 50.00]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[47.80, 52.20]	[50.00, 50.00]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[51.00, 49.00]	[50.00, 50.00]	[50.00, 50.00]
	Mixed	[45.56, 54.44]	[45.30, 54.70]	[43.40, 56.60]	[45.60, 54.40]	[45.56, 54.44]
	Disconnected	[45.43, 54.57]	[45.00, 55.00]	[40.60, 59.40]	[45.40, 54.60]	[45.43, 54.57]
Case 6	Concave	[50.00, 50.00]	[50.00, 50.00]	[51.70, 48.30]	[50.00, 50.00]	[50.00, 50.00]
	Convex	[50.00, 50.00]	[50.00, 50.00]	[48.60, 51.40]	[50.00, 50.00]	[50.00, 50.00]
	Line	[50.00, 50.00]	[50.00, 50.00]	[51.70, 48.30]	[50.00, 50.00]	[50.00, 50.00]
	Mixed	[42.47, 57.53]	[42.90, 57.10]	[38.50, 61.50]	[42.50, 57.50]	[42.47, 57.53]
	Disconnected	[45.00, 55.00]	[44.70, 55.30]	[42.60, 57.40]	[45.00, 55.00]	[45.00, 55.00]

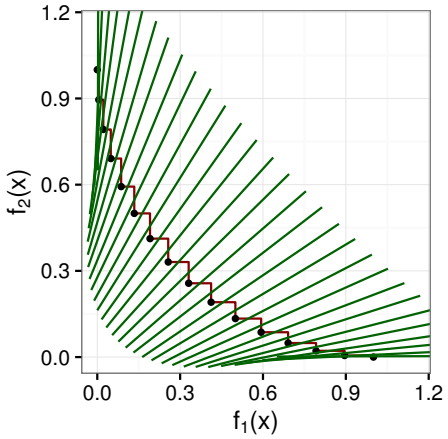


Figure 5.15: Convex POF and attainment surface with WASSILs.

measure calculation.

The rotation-based intersection line generation approach presented by Knowles and Corne fared better than the random intersection line generation approach. Only seven of the 30 measurements using the rotation-based intersection line generation approach had a deviation greater than 5%. Case 1 with a convex POF fared worse with a deviation of almost 15% for both algorithms. Four of the five case 2 measurements using the rotation-based intersection line generation approach had a deviation greater than 5%. For the remaining case 2 measurement, a deviation of at least 3% can be noted. While the rotation-based intersection line generation approach outperformed, with respect to accuracy, the random intersection line generation approach, the results indicate that the rotation-based intersection line generation approach to is not well suited for the KC measure calculation and that the results vary based on the POF shape and the spread of the solutions.

As expected, the WASSIL generation approach produced the same results as the actual KC measure: The closer the POFs being compared are to the true POF, the more accurate the comparison using the WASSIL generation approach becomes.

5.4 n_m-dimensional Attainment Surface

For n_m-dimensional problems, Knowles and Corne [67] recommended using a grid-based intersection line generation approach as explained in section 5.2. Similar to the rotational approach for 2-dimensional problems, the grid-based approach will lead to unbalanced intersection lines when measuring irregularly shaped POFs. Figure 5.16 shows an example of an irregularly shaped 3-dimensional attainment surface.

Section 5.4.1 discusses the challenges that need to be addressed in order to generate intersection lines for n_m-dimensional attainment surfaces. Section 5.4.2 and 5.4.3 introduces two algorithms to generate n_m-dimensional attainment surface intersection lines. Finally, Section 5.4.4 presents a stability analysis of the two proposed algorithms.

5.4.1 Generalizing attainment surface intersection line generation to n_m-dimensions

For 2-dimensional problems, the ASSIL approach generates balanced intersection lines. Intuitively, generalization of the ASSIL approach for n_m-dimensions requires the calculation of equally spread points over the n_m-dimensional attainment surface.

Calculating equally spread points requires that the surface is divided into equally sized $n_m - 1$ dimensional hypercubes. For the 3-dimensional case, this requires that the

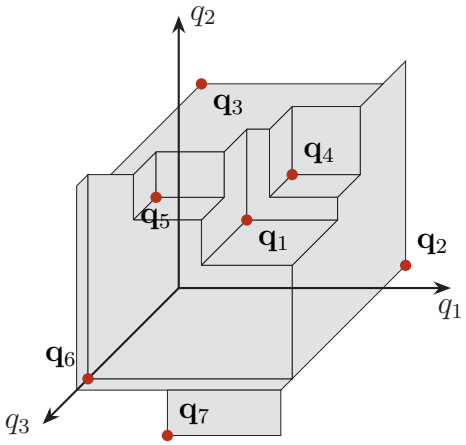


Figure 5.16: 3-dimensional attainment surface.

attainment surface is divided into equally sized squares. The intersection vectors are positioned from the middle of each square. The length of the square's edges is set to the greatest common divider of the lengths of the edges that make up the attainment surface. Even for simple cases, this will lead to an excessive number of squares. The more squares, the higher the computational cost of the measure.

In order to lower the computational cost of the measure, the number of squares needs to be reduced. Because the square edge lengths are based on the greatest common divider, there is no way to reduce the number of squares as long as the squares must be equal in size. If the square sizes differ in size, the resulting KC measure will become biased to areas with smaller squares. Areas with smaller squares will carry more weight in the overall KC measure calculation as there will be more of them.

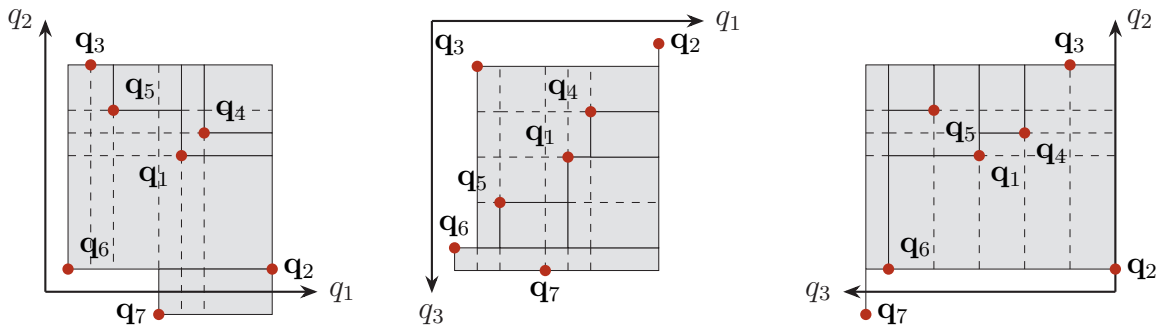
In contrast to the ASSIL approach, the WASSIL approach does not require the intersection lines to be equally spread, instead only a weight factor must be known for each intersection line. For the 3-dimensional case, the weight factor for each intersection line is calculated as the area of the squares that make up the attainment surface. The weight factor for each square is thus calculated by multiplying the square's edge, or rather side, lengths. The weight factor can also be based on the size of rectangles, instead of squares, in the 3-dimensional case (hyper-rectangles in the n_m-dimensional case).

The next section introduces an algorithm to generate weighted attainment surface intersection lines for n_m-dimensional attainment surfaces.

5.4.2 Porcupine Measure (Naive Implementation)

This section presents the naive implementation for the n_m-dimensional attainment surface based quantitative measurement, named the porcupine measure. The naive implementation uses each of the n_m-dimensional values from each Pareto-optimal point to subdivide the attainment surface in each of the dimensions. Figure 5.17 depicts an example of the subdivision approach for each of the three dimensions. Figure 5.18 depicts the attainment surface, in 3-dimensional space, with the subdivisions visible.

In addition to the calculation of the hyper-rectangles, the center point, and intersection vector needs to be calculated. The naive implementation of the porcupine measure is summarized in Algorithm 9.



(a) Front (b) Top (c) Side

Figure 5.17: Top, front, side view of attainment surface with naive subdivisions.

Algorithm 9 Porcupine measure (naive implementation) – Part 1 of 2

1: Let $Q = \{\mathbf{q}_k : k \in \{1, 2, ..., n_q\}\}$ be the POF that corresponds with the attainment surface;
2: **for** each of the m objective space basis vectors, $\mathbf{e}_m$, with $m \in \{1, 2, ..., n_m\}$ **do**
3: Project the attainment surface parallel to the basis vector, $\mathbf{e}_m$, onto the orthogonal $(n_m - 1)$-dimensional subspace;
4: Subdivide the projected attainment surface in each of its $(n_m - 1)$-dimensions at every Pareto-optimal point $q_k \in Q$ into hyper-rectangles (figure 5.17 depicts the hyper-rectangle subdivisions for the three 2-dimensional projected attainment surfaces of a 3-dimensional attainment surface);
5: **for** each hyper-rectangle **do**
6: Let $\hat{\mathbf{c}}$ be the center point of the hyper-rectangle;
7: Let w_h be the weight of the hyper-rectangle, equal to the product of the hyper-rectangle edge lengths;
8: **for** each dimension m **do**
9: Let min_m be the smallest of the m'th dimensional values of all the Pareto-optimal points where at least one other dimensional value is less or equal to that of the corresponding center vector's dimensional value;
10: Let max_m be the largest of the m'th dimensional values of all the Pareto-optimal points;
11: **end for**

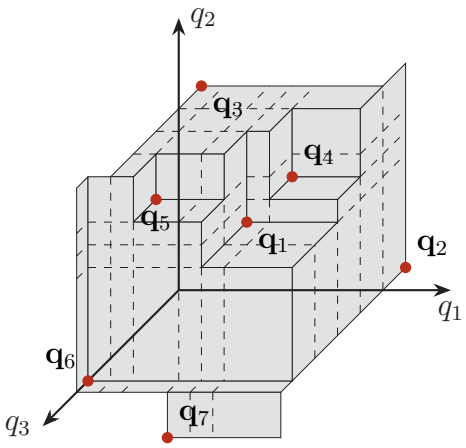


Figure 5.18: 3-dimensional attainment surface with naive subdivisions.

Algorithm 9 Porcupine measure (naive implementation) – Part 2 of 2

Let $\hat{\mathbf{p}}$ be the intersection vector, calculated as $\hat{p}_m = \frac{\hat{c}_m - min_m}{max_m - min_m}$, where $\hat{p}_m$ is the m'th component of the vector $\hat{\mathbf{p}}$;

drawIntersectionLineWithWeight($from = \hat{\mathbf{c}} - \hat{\mathbf{p}}$, $to = \hat{\mathbf{c}} + \hat{\mathbf{p}}$, $weight = w_h$);

end for

end for

Using the intersection lines generated by the above algorithm, two algorithms can now be compared using a nonparametric statistical test, such as the Mann-Whitney U test [43]. The porcupine measure is defined, similar to the weighted KC measure, as the weighted sum of the intersection lines where a statistically significant difference exists over the sum of all the weights (i.e. the percentage of the surface area of the attainment surface, as determined by the weights, where one algorithm statistically performs superior to another).

Figure 5.19 depicts an attainment surface with subdivisions and intersection vectors generated using the naive approach. The porcupine measure's name is derived from the fact that the intersection vectors resemble the spikes of a porcupine.

The naive implementation presented in this section is easy to implement but results in more subdivisions than necessary.

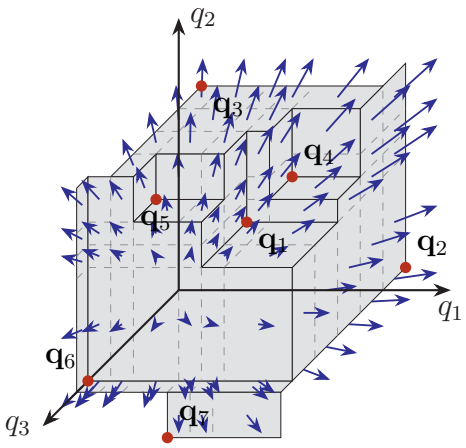


Figure 5.19: 3-dimensional attainment surface with naive subdivisions and intersection vectors.

5.4.3 Porcupine Measure (Optimized Implementation)

The large number of subdivisions that result from using the naive implementation of the porcupine measure creates a computational complexity problem when performing the statistical calculations required by the porcupine measure. To reduce the computational cost of the porcupine measure, the naive implementation can be optimized by subdividing the attainment surface only as necessary to accommodate the shape of the attainment surface. Figure 5.20 depicts an attainment surface with the subdivision lines (dashed) as generated by the optimized implementation.

Note that the algorithm yields the minimum number of subdivisions such that the results are independent of the dimension ordering of the Pareto-optimal points. This is per design to allow for reproducibility and increased stability of the results.

The optimized implementation of the porcupine measure is summarized in Algorithm 10.

Algorithm 10 Porcupine measure (optimized implementation) – Part 1 of 3

1: Let $Q = \{\mathbf{q}_k : k \in \{1, 2, ..., n_q\}\}$ be the POF that corresponds with the attainment surface;
2: Let $\mathbf{q}_{max}$ be the *nadir* vector;
3: **for** each of the m objective space basis vectors, $\mathbf{e}_m$, with $m \in \{1, 2, ..., n_m\}$, **do**

Algorithm 10 Porcupine measure (optimized implementation) – Part 2 of 3

4: Project the attainment surface parallel to the basis vector, $\mathbf{e}_m$, onto the orthogonal $(n_m - 1)$-dimensional subspace
5: **for** each of the Pareto-optimal points, $\mathbf{q}_k \in Q$ **do**
6: Let $\hat{\mathbf{q}}$ be the vector with dimensional values set to the minimum dimensional values that are dominated or the corresponding q_{max} value if no minimum exist;
7: Let $\hat{Q}$ be the set of non-dominated points that dominate $\hat{\mathbf{q}}$ but not $\mathbf{q}_k$;
8: **for** each dimension m **do**
9: Let Q_m be the set of all the m'th dimensional values of the points in $\hat{Q}$ that fall inside the range $[q_{km}, \hat{q}_m]$;
10: **end for**
11: Let Q_{min} be the set of all the minimum points that will affect the calculation of $\mathbf{p}$;
12: Let Q_{max} be the set of all the maximum points that will affect the calculation of $\mathbf{p}$;
13: Add all dimensional values of the points in Q_{min} and Q_{max} that fall inside the range $[q_{km}, \hat{q}_m]$ to the corresponding Q_m set;
14: Add additional values to the Q_m sets to limit the hyper-rectangle edge lengths to Δ_{max};
15: Subdivide the projected attainment surface into hyper-rectangles in each its m dimensions for every value in the Q_m sets;
16: **for** each hyper-rectangle **do**
17: Let $\hat{\mathbf{c}}$ be the center point of the hyper-rectangle;
18: Let w_h be the weight of the hyper-rectangle, equal to the product of the hyper-rectangle edge lengths;
19: **for** each dimension m **do**
20: Let min_m be the smallest of the m'th dimensional values of all the Pareto-optimal points where at least one other dimensional value is less or equal to that of the corresponding center vector's dimensional value;
21: Let max_m is the largest of the m'th dimensional values of all the Pareto-optimal points;
22: **end for**

Algorithm 10 Porcupine measure (optimized implementation) – Part 3 of 3

Let $\hat{\mathbf{p}}$ be the intersection vector, calculated as $\hat{p}_m = \frac{c_m - min_m}{max_m - min_m}$, where $\hat{p}_m$ is the m'th component of the vector $\hat{\mathbf{p}}$;
drawIntersectionLineWithWeight($from = \hat{\mathbf{c}} - \hat{\mathbf{p}}$, $to = \hat{\mathbf{c}} + \hat{\mathbf{p}}$, $weight = w_h$);
end for
end for
end for

Similar to the naive implementation, the porcupine measure is defined as the weighted sum of the intersection lines where a statistically significant difference exists over the sum of all the weights (i.e. the percentage of the surface area of the attainment surface, as determined by the weights, where one algorithm statistically performs superior to another).

Figure 5.21 depicts an attainment surface with subdivisions and intersection vectors generated using the optimized implementation. As can be seen in the figure, the optimized implementation resulted in notably fewer subdivisions and intersection vectors. The lower number of intersection vectors considerably reduces the computational complexity of the measure.

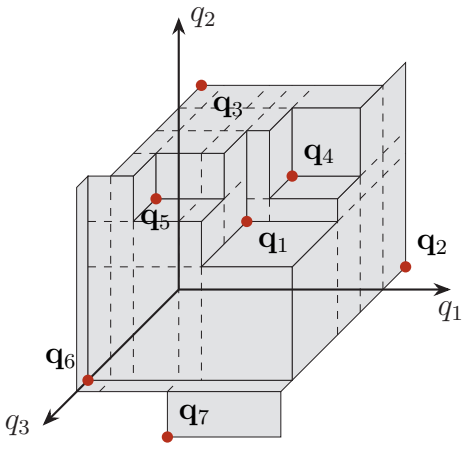


Figure 5.20: 3-dimensional attainment surface with optimized subdivisions.

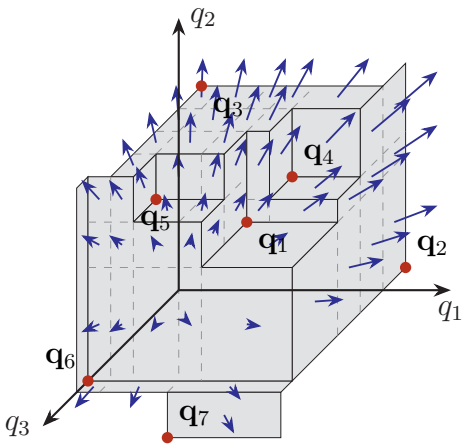


Figure 5.21: 3-dimensional attainment surface with optimized subdivisions and intersection vectors.

5.4.4 Stability Analysis

In order to determine the stability of the optimized implementation compared to the naive implementation, 30 independent runs of each measure were executed. For each run, the porcupine measure was calculated using the POFs as calculated by 30 independent runs of each of the algorithms being compared. A total of 30×30, or 900, runs were executed for each algorithm being compared.

Table 5.6 lists the mean, standard deviation (σ), minimum, and maximum for the naive and optimized implementations of the porcupine measure. The maximum edge length for the optimized implementations of the porcupine measure was set to 0.1.

Experimental results showed that a maximum edge length of 0.1 for the optimized implementation yielded best results when compared with the naive implementation. Increasing the maximum edge length decreased the statistical significance of the optimized implementation of the porcupine measure's results when compared with the naive implementation. Decreasing the maximum edge length did not affect the statistical significance of the results.

Statistical testing was performed to determine if there were any statistically significant differences between the naive and optimized implementation's results. The Mann-Whitney U test was used with a confidence level of 95%.

The purpose of the statistical testing was to determine if there was any information loss due to using the optimized implementation when compared to the naive implementation. The results indicated that for 52 out of the 54, or 96%, of the measurements there were no statistically significant difference. Only for two measurements, namely OMOPSO in the WFG3 OMOPSO vs. VEPSO comparison and the SMPSO in the WFG3 VEPSO vs. SMPSO comparison, a statistically significant difference could be noted. In spite of the statistically significant difference, the ranking of the algorithms in the result did not change. Based on the results it can, therefore, be concluded that the optimized implementation yields the same results as the naive implementation and no information was lost. As no information was lost when using the optimized implementation, it can be concluded that the optimized implementation can be used when conducting comparisons.

The mean standard deviation of the optimized implementation's results was 2.743, and the maximum standard deviation was 7.022. The conclusion that can be drawn from the data is that the optimized implementation of the porcupine measure is very robust as measurement values for each of the samples were close to the average.

For the experimentation done for this study, the runtime of the optimized implementation was notably faster than that of the naive implementation. A difference in the orders of a few magnitudes was noticeable.

The computational complexity of the naive implementation is directly proportional to the size of the Q_m sets. It should be noted that, for the tested algorithms with a POF size of 50 points, tested over 30 samples, the optimal POF had a typical size of ± 1250 points. The size of the Q_m sets were thus ± 1250 points. For three dimensions, this results in a minimum computational complexity of at least 1250^3 or rather $1,953,125,000$ (almost two billion). The optimized implementation resulted in a much reduced computational complexity because only the necessary subdivisions were made. The size of the Q_m sets were much smaller. For the three-dimensional case, the maximum edge length leads to a minimum complexity of at least $(\frac{1}{0.1})^3$, or rather 1000 times lower than that of the naive version.

Table 5.6: Naive vs Optimized Porcupine measure (3-objective WFG problem set)

Problem	Algorithm	Naive				Optimized			
		Mean	**σ**	**Min**	**Max**	**Mean**	**σ**	**Min**	**Max**
WFG1	OMOPSO	3.054	2.178	0.2596	9.4	3.092	2.245	0.5106	9.728
	SMPSO	32.34	6.054	17.29	43.92	32.38	6.132	16.57	42.92
	OMOPSO	1.308	1.11	0.2418	4.886	1.32	1.153	0.2445	5.071
	VEPSO	43.55	4.06	30.43	52.71	43.16	4.031	32.27	52.73
	VEPSO	19.62	6.619	8.857	47.63	18.73	4.149	8.679	28.34
	SMPSO	6.631	2.949	0.8017	17.61	6.701	2.782	3.491	17.05
WFG2	OMOPSO	49.88	6.116	32.83	60.57	49.29	6.517	30.85	60.54
	SMPSO	0.01869	0.06869	0.0	0.3711	0.01736	0.06466	0.0	0.3442
	OMOPSO	59.43	4.403	47.63	67.35	59.64	4.397	47.61	66.87
	VEPSO	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	VEPSO	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	SMPSO	59.76	4.612	46.3	66.91	59.83	4.716	46.26	67.61
WFG3	OMOPSO	26.98	4.088	18.44	37.32	28.99	4.524	20.43	43.72
	SMPSO	3.099	2.262	0.1569	8.927	3.078	2.097	0.2126	7.497
	OMOPSO	**62.55**	**3.341**	**55.16**	**68.42**	**66.07**	**3.722**	**58.36**	**73.14**
	VEPSO	0.5199	0.1456	0.2931	0.8233	0.5875	0.1695	0.2527	0.8897
	VEPSO	1.177	0.48	0.2131	2.315	1.243	0.5278	0.2095	2.5
	SMPSO	**59.17**	**3.402**	**52.04**	**64.47**	**62.57**	**3.668**	**55.66**	**69.12**
WFG4	OMOPSO	26.78	2.433	22.7	30.65	26.37	2.919	20.3	31.05
	SMPSO	9.348	1.915	5.002	13.57	9.306	1.989	5.122	13.62
	OMOPSO	63.0	4.539	55.21	75.74	63.82	4.798	55.75	77.11
	VEPSO	0.02337	0.03589	0.0	0.1569	0.02468	0.04244	0.0	0.1929
	VEPSO	0.02615	0.09773	0.0	0.533	0.02406	0.08814	0.0	0.4763
	SMPSO	71.11	3.919	62.78	80.38	71.92	4.072	63.2	82.07
WFG5	OMOPSO	25.67	4.393	17.47	34.0	25.79	4.472	17.55	34.4
	SMPSO	17.61	3.64	11.16	27.63	17.65	3.824	10.68	27.83
	OMOPSO	26.21	2.314	22.2	30.72	26.31	2.432	21.48	31.12
	VEPSO	17.94	2.65	10.6	22.42	17.54	2.691	10.56	21.72
	VEPSO	25.13	4.301	14.71	33.27	25.12	4.463	13.9	33.76
	SMPSO	5.143	1.277	2.736	8.069	5.187	1.338	2.539	8.049
WFG6	OMOPSO	10.21	2.398	6.651	15.76	10.55	2.356	7.035	15.89
	SMPSO	45.3	3.61	37.82	53.57	45.62	3.754	38.12	54.17
	OMOPSO	11.93	4.704	6.098	22.65	12.29	4.854	5.991	23.65
	VEPSO	16.05	5.158	5.421	25.88	16.53	5.414	5.566	26.5
	VEPSO	1.02	0.8952	0.01929	3.795	1.046	0.9204	0.007599	3.983
	SMPSO	20.61	6.907	8.904	37.24	20.94	7.022	8.945	37.69
WFG7	OMOPSO	38.86	3.64	31.77	44.27	39.28	3.516	32.12	44.38
	SMPSO	3.892	0.9721	1.684	5.522	3.888	1.018	1.641	5.507
	OMOPSO	74.41	2.681	67.75	78.68	74.72	2.743	67.8	79.15
	VEPSO	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
	VEPSO	0.001361	0.007456	0.0	0.04084	9.506e-05	0.0005206	0.0	0.002852
	SMPSO	73.39	2.598	68.5	77.94	73.69	2.682	68.92	79.15
WFG8	OMOPSO	41.21	4.789	31.08	49.28	41.55	4.954	30.95	49.24
	SMPSO	3.049	0.948	1.443	5.066	3.106	0.9568	1.54	5.088
	OMOPSO	70.18	3.918	63.04	77.36	70.49	3.847	63.4	77.46
	VEPSO	6.884e-05	0.0002857	0.0	0.001467	0.0	0.0	0.0	0.0
	VEPSO	0.1489	0.2362	0.0	1.023	0.1458	0.223	0.0	0.869
	SMPSO	62.88	3.841	56.14	72.58	63.15	3.861	56.65	73.03
WFG9	OMOPSO	15.89	2.462	11.12	21.29	15.87	2.47	11.48	21.66
	SMPSO	37.58	3.03	32.14	43.94	37.68	3.048	31.95	44.27
	OMOPSO	20.7	2.526	16.86	25.28	20.61	2.515	16.9	26.05
	VEPSO	23.59	1.801	19.21	26.88	24.0	1.795	19.61	27.51
	VEPSO	1.206	0.7576	0.2238	3.162	1.219	0.8411	0.2526	3.605
	SMPSO	21.91	5.093	12.24	31.96	21.8	5.307	11.12	32.25

5.5 Summary

This chapter investigated shortcomings that may have led to the lack of adoption of attainment surface based quantitative measurements for MOO. It was shown that the quantitative measure proposed by Knowles and Corne could result in misleading results for convex POFs when using rotational intersection lines. The ASSIL and WASSIL generation approaches were proposed. The ASSIL and WASSIL generation approaches were shown not to be biased against any attainment surface shape.

An algorithm for an n_m-dimensional attainment surface based quantitative measure, named the porcupine measure, was presented. Additionally, a computationally optimized implementation of the porcupine measure was introduced and analyzed. The results indicated that the optimized implementation performed as well as the naive implementation.

The porcupine measure allows for a quantitative comparison between n_m-dimensional POFs through the use of attainment surfaces. The porcupine measure provides additional information on an algorithm's performance when compared to another algorithm, which was previously not quantifiable.

Chapter 6

Multi-guided Particle Swarm Optimization

"No great discovery was ever made without a bold guess."

Isaac Newton (1643 - 1727)

This chapter presents a proposal for a new PSO based MOO algorithm, named MGPSO. The exploration behavior of MGPSO is analyzed using the candidate solution visualization technique presented in Section 3.1. A thorough performance analysis using the IGD measure along with the ZDT and WFG test sets covering both 2 and 3-objective problems is presented. A comparative analysis using the attainment surface inspired porcupine measure, introduced in Section 5.4, is also presented.

Section 6.1 introduces MGPSO. Section 6.2 describes the experimental procedure used throughout this chapter. A performance analysis is presented in Section 6.3 followed by a comparative analysis of MGPSO versus the *state of the art* MOO algorithms in Section 6.4. Finally, the conclusions are given in Section 6.5.

6.1 Multi-guided Particle Swarm Optimizer

MGPSO is loosely based on VEPSO and attempts to address the shortcomings of the VEPSO algorithm that have been identified throughout this thesis. Similar to VEPSO,

MGPSO is a multi-swarm algorithm where each objective is represented by a subswarm. Particles in each subswarm are evaluated using the corresponding objective function. The personal best and neighborhood best particles are also updated using the corresponding objective function fitness value.

In contrast to VEPSO, MGPSO does not make use of a KTS. Instead, MGPSO retains the neighborhood guide and adds an additional archive guide term. The addition of the archive guide term was inspired by the results from the explorative study of VEPSO in Section 3.3. The explorative study showed that the addition of an archive term might lead to good results. The MGPSO velocity update equation is formally defined as:

$$\begin{aligned}\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\mathbf{r}_1(\mathbf{y}_i(t) - \mathbf{x}_i(t)) + \lambda_i c_2\mathbf{r}_2(\hat{\mathbf{y}}_i(t) - \mathbf{x}_i(t)) \\ + (1-\lambda_i)c_3\mathbf{r}_3(\hat{\mathbf{a}}_i(t) - \mathbf{x}_i(t))\end{aligned} \tag{6.1}$$

where $\mathbf{v}_i(t)$ is the velocity of particle i at iteration t, w is the inertia weight, c_1, c_2 and c_3 are the acceleration coefficients, $\mathbf{r}_1$, $\mathbf{r}_2$, and $\mathbf{r}_3$ are random vectors with components sampled uniformly from $(0,1)$, $\mathbf{x}_i(t)$ is the position of particle i at iteration t, $\mathbf{y}_i(t)$ is the personal best particle position of particle i at iteration t, $\mathbf{y}_i(t)$ is the neighborhood guide for particle i at iteration t, $\hat{\mathbf{a}}_i(t)$ is the archive guide for particle i at iteration t, and λ_i is the exploitation tradeoff coefficient for particle i. λ_i is initialized as a random constant sampled uniformly from $(0,1)$. λ_i controls the amount of influence that the archive guide has on the particle's velocity and thus the amount of exploitation of the already found POF. Smaller λ_i values increase the influence of the archive guide while simultaneously decreasing the influence of the neighborhood guide. Larger λ_i values decrease the influence of the archive guide while simultaneously increasing the influence of the neighborhood guide. As both the neighborhood and archive guide represent social behavior, λ_i maintains the balance between the social and cognitive influences on the particle's velocity.

Inspired by the findings in Section 4.5, the MGPSO archive is a bounded archive using the crowding distance AMS. For easier comparison with other algorithms, the default archive size is set equal to the total number of particles in the subswarms.

The archive guide, $\hat{\mathbf{a}}_i(t)$, is selected, from a competition pool, as the solution with largest corresponding crowding distance in the archive. The competition pool is con-

structed by randomly selecting a predefined number of solutions from the archive. Empirical experimentation showed that tournament sizes of 2 and 3 yielded good results. By using crowding distance as part of the archive guide selection process, the MGPSO is guided to focus more on sparsely populated areas of the objective space.

MGPSO differs from the archive-guided PSO introduced in Section 3.3 by specifying the archive management strategy, as well as using crowding distance based tournament selection to select the archive guide, $\hat{\mathbf{a}}_i(t)$.

The MGPSO algorithm is formally defined in Algorithm 11.

Algorithm 11 Multi-guided Particle Swarm Optimization – Part 1 of 2

for each objective $m = 1, ..., n_m$ **do**
  Create and initialize a swarm, S_m, of n_{s_m} particles uniformly within a predefined hypercube of dimension n_x;
  Let f_m be the objective function;
  Let $S_m.\mathbf{y}_i$ represent the personal best position of particle $S_m.\mathbf{x}_i$, initialized to $S_m.\mathbf{x}_i(0)$;
  Let $S_m.\hat{\mathbf{y}}_i$ represent the neighborhood best position of particle $S_m.\mathbf{x}_i$, initialized to $S_m.\mathbf{x}_i(0)$;
  Initialize $S_m.\mathbf{v}_i(0)$ to $\mathbf{0}$;
  Initialize $S_m.\lambda_i \sim U(0, 1)$;
end for
Let $t = 0$;
repeat
  for each objective $m = 1, ..., n_m$ **do**
    for each particle $i = 1, ..., S_m.n_s$ **do**
      if $f_m(S_m.\mathbf{x}_i) < f_m(S_m.\mathbf{y}_i)$ **then**
        $S_m.\mathbf{y}_i = S_m.\mathbf{x}_i(t)$;
      end if
      for particles $\hat{\imath}$ with particle i in their neighborhood **do**
        if $f_m(S_m.\mathbf{y}_i) < f_m(S_m.\hat{\mathbf{y}}_{\hat{\imath}})$ **then**
          $S_m.\hat{\mathbf{y}}_{\hat{\imath}} = S_m.\mathbf{y}_i$;
        end if

Algorithm 11 Multi-guided Particle Swarm Optimization – Part 2 of 2

end for
Update the archive with the solution $S_m.\mathbf{x}_i$;
end for
end for
for each objective $m = 1, ..., n_m$ **do**
for each particle $i = 1, ..., S_m.n_s$ **do**
Select a solution, $S_m.\hat{\mathbf{a}}_i(t)$, from the archive using tournament selection;
$$S_m.\mathbf{v}_i(t+1) = wS_m.\mathbf{v}_i(t) + c_1\mathbf{r}_1(S_m.\mathbf{y}_i(t) - S_m.\mathbf{x}_i(t)) + S_m.\lambda_i c_2\mathbf{r}_2(S_m.\hat{\mathbf{y}}_i(t) - S_m.\mathbf{x}_i(t)) + (1 - S_m.\lambda_i)c_3\mathbf{r}_3(S_m.\hat{\mathbf{a}}_i(t) - S_m.\mathbf{x}_i(t)));$$
$$S_m.\mathbf{x}_i(t+1) = S_m.\mathbf{x}_i(t) + S_m.\mathbf{v}_i(t+1);$$
end for
end for
$t = t + 1$;
until stopping condition is true

6.2 Experimental Procedure

MGPSO was executed with 50 particles, divided between the subswarms, for each of the ZDT, 2-objective WFG, and 3-objective WFG problems. Optimized values for the number of particles per subswarm, the inertia weight, w, the acceleration coefficients, c_1, c_2, and c_3, the number of particles per subswarm, and the tournament size were used. An analysis of the influence these parameters have on the performance of MGPSO, the parameter optimization procedure, and the optimized parameters used throughout this chapter is presented in chapter 7.

The MGPSO algorithm was implemented and executed using the CIlib framework[1] [15, 83]. The results presented were taken over 30 independent runs of 2000 iterations for each algorithm for each problem.

[1] https://cirg-up.github.io/cilib/

6.3 Exploration and Performance Analysis

POFs obtained through the MGPSO are shown in figures 6.1(a) through 6.1(e) for the ZDT problems, figures 6.2(a) through 6.2(i) for the 2-objective WFG problems, and figures 6.3(a) through 6.3(i) for the 3-objective WFG problems. The figures show that MGPSO managed to obtain a close approximation of the true POFs for almost all of the test problems. The POF for ZDT4 shows that MGPSO was not able to obtain a close approximation and the problem was particularly challenging for MGPSO to solve. The ZDT4 result indicates that the MGPSO is susceptible to getting stuck in local optima under certain conditions. Further work is required to fully analyze MGPSO's susceptibility to local optima.

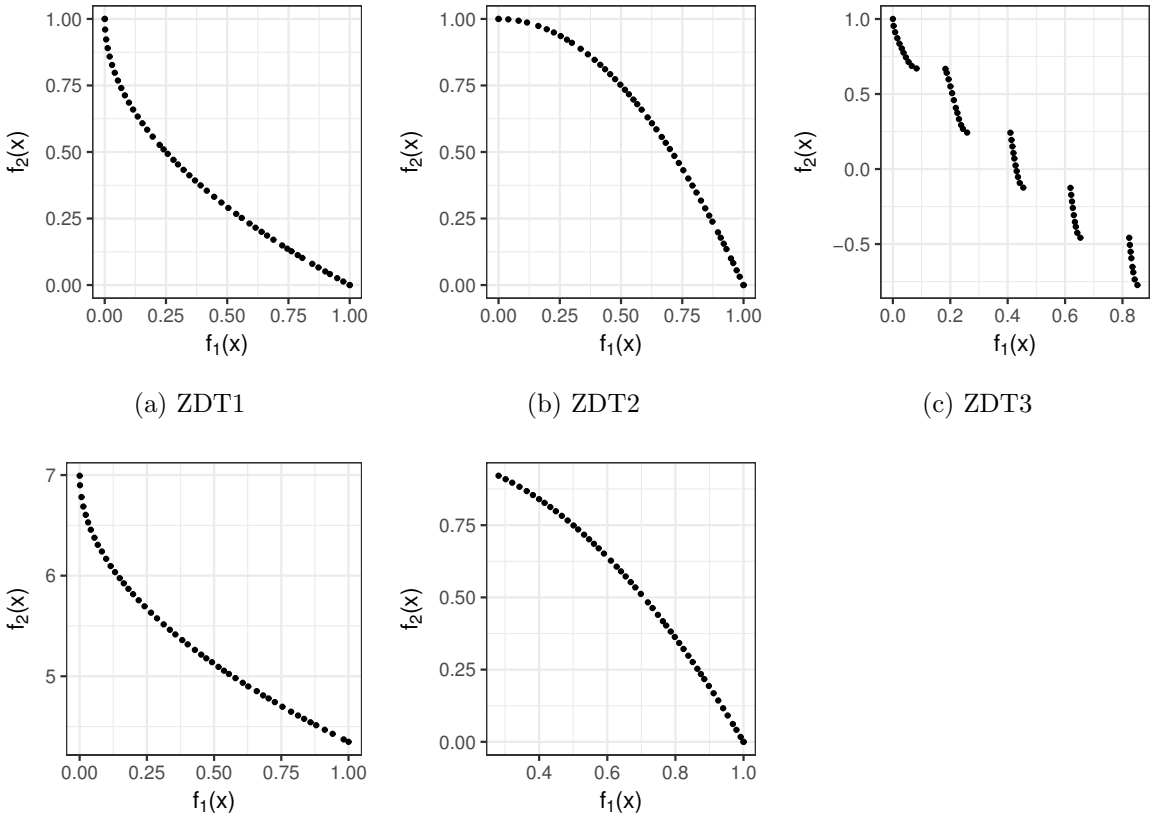
Section 6.3.1 presents a discussion of the MGPSO's exploration behavior, followed by a performance analysis in Section 6.3.2. A comparison of the obtained POFs is given in Section 6.3.3. Finally, a summary of the findings is given in Section 6.3.4.

6.3.1 Exploration Behavior

In order to study the exploration behavior of MGPSO, the candidate solutions represented by the particle positions were tracked and plotted. For each of the 2-objective problems, two candidate solution plots were generated, one for each of the sub-swarms. Figures 6.4(a) through 6.4(j) show the candidate solution plots for the ZDT problems and figures 6.5(a) through 6.5(r) show the candidate solution plots for the WFG problems.

The candidate solution plots for the majority of the problems show that at least one swarm in the case of the ZDT problems and both swarms in the case of the WFG problems tend to explore the region close to the true POF. MGPSO's search behavior is much more focused on exploiting the well-performing region close to the POF when compared to VEPSO's more exploration focused behavior. The exploration first, followed by exploitation, pattern clearly aligns with the two goals for a MOO algorithm as defined by Deb [25].

For S_2 on the ZDT1 problem, denoted ZDT1:S_2, a second line of solutions near the POF is also clearly visible. This line represents a local minimum POF. Similar local minima lines that are visible for ZDT2:S_2 and ZDT4:S_2.

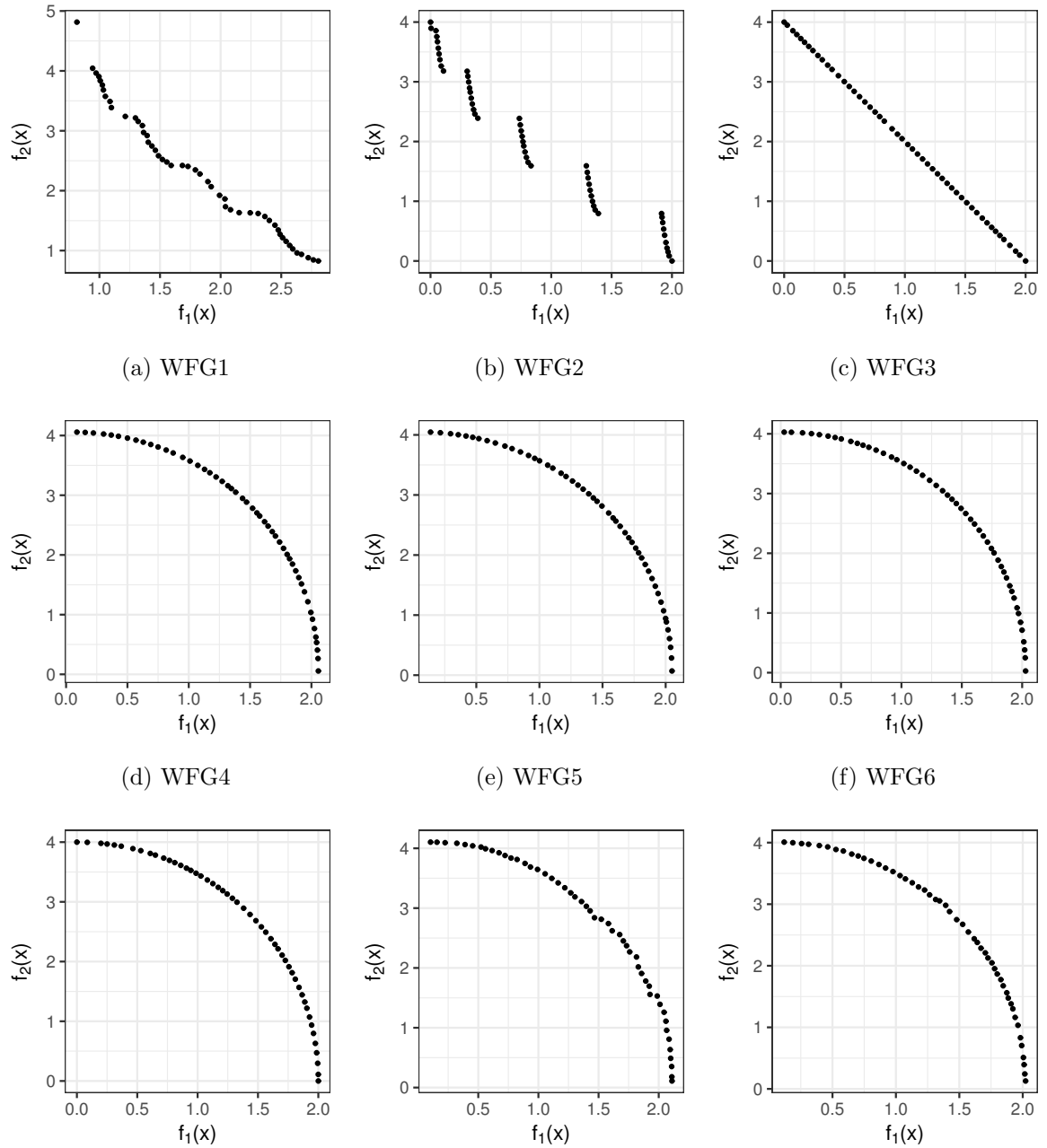


(a) ZDT1 (b) ZDT2 (c) ZDT3

(d) ZDT4 (e) ZDT6

Figure 6.1: MGPSO calculated POFs for ZDT problems

For the majority of the WFG problems, the search is focused in the region close to the POF. A notable exception is WFG5:S_2, where the swarm size for S_2 was zero as a result of the optimized values from Chapter 7. That is, there were no particles in S_2. The particles in S_1 were enough to find the POF as reflected by the candidate solution plot for WFG5:S_1 where a well-defined POF can be seen. For WFG9:S_1, the search seems to be actively exploring both an area close to the POF as well as the region making up the far end away from the POF. In contrast to WFG9:S_1, for WFG9:S_2 the search is much more focused on the well-performing POF region.

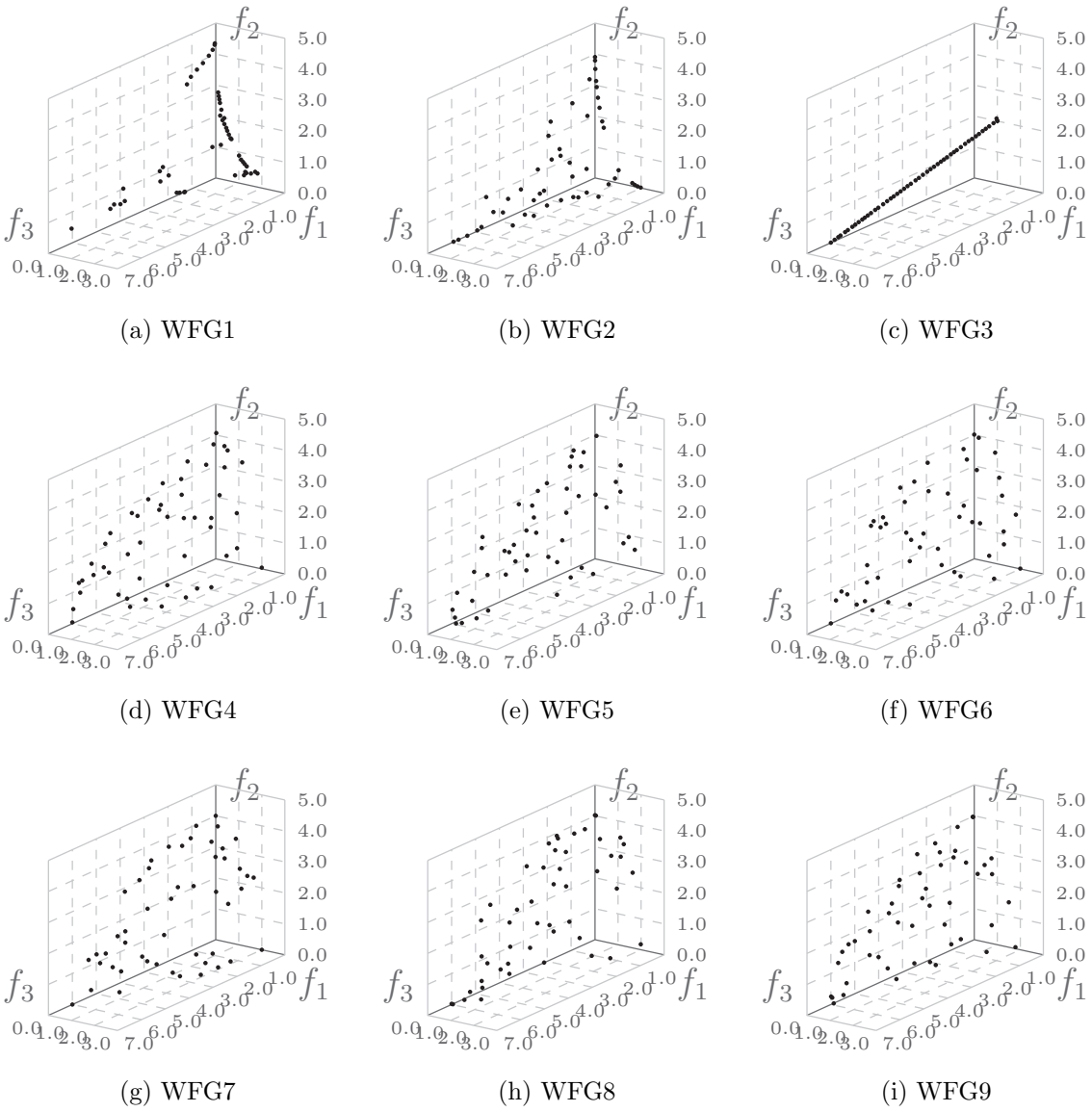


(a) WFG1 (b) WFG2 (c) WFG3

(d) WFG4 (e) WFG5 (f) WFG6

(g) WFG7 (h) WFG8 (i) WFG9

Figure 6.2: MGPSO calculated POFs for 2-objective WFG problems



(a) WFG1 (b) WFG2 (c) WFG3

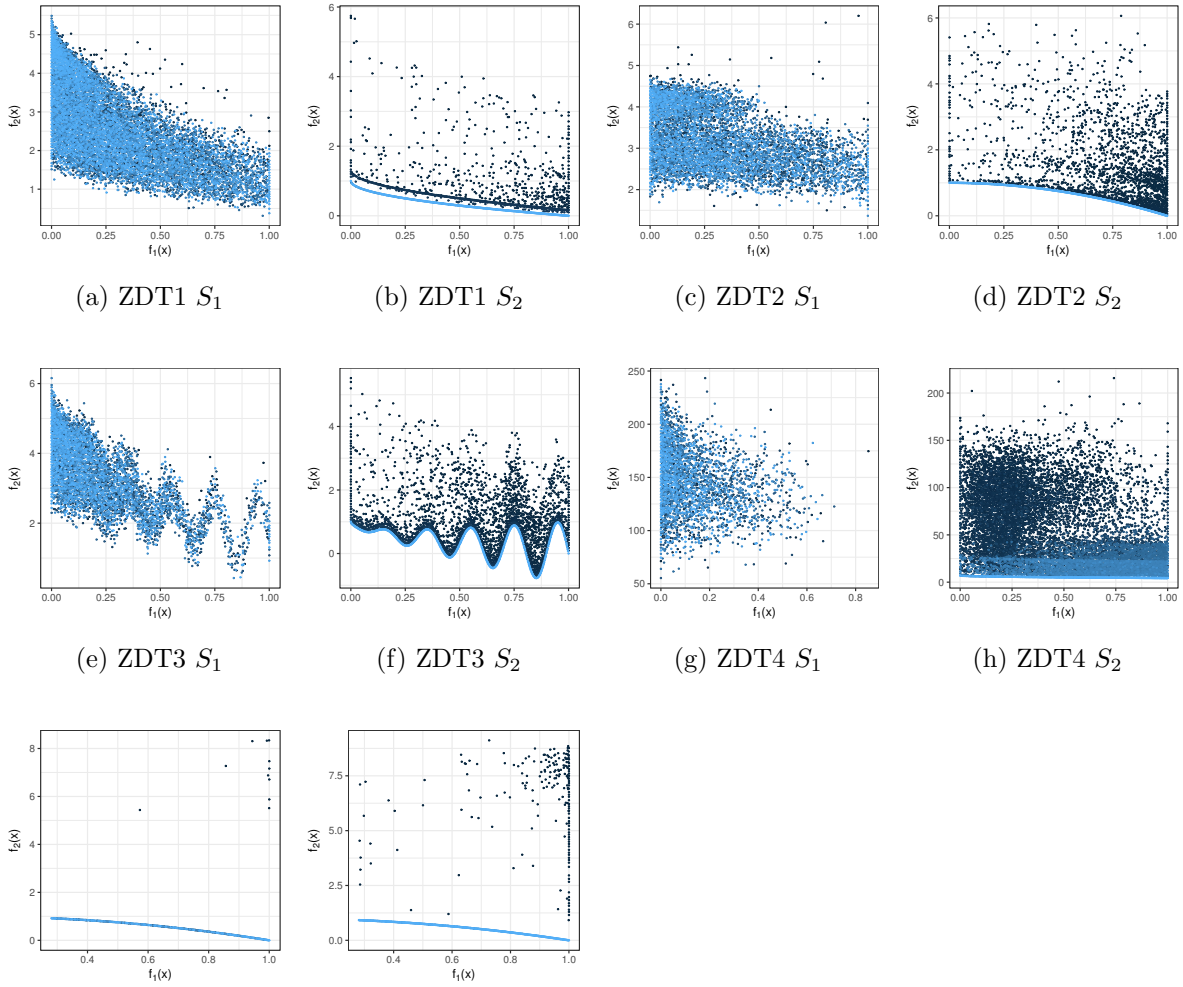
(d) WFG4 (e) WFG5 (f) WFG6

(g) WFG7 (h) WFG8 (i) WFG9

Figure 6.3: MGPSO calculated POFs for 3-objective WFG problems

6.3.2 Performance Analysis

While the candidate solution plots show the exploration behavior, no information regarding the quality of the found solutions are given. In order to measure the quality of the solutions quantitatively, the IGD was calculated for each of the problems. The true



(a) ZDT1 S_1 (b) ZDT1 S_2 (c) ZDT2 S_1 (d) ZDT2 S_2

(e) ZDT3 S_1 (f) ZDT3 S_2 (g) ZDT4 S_1 (h) ZDT4 S_2

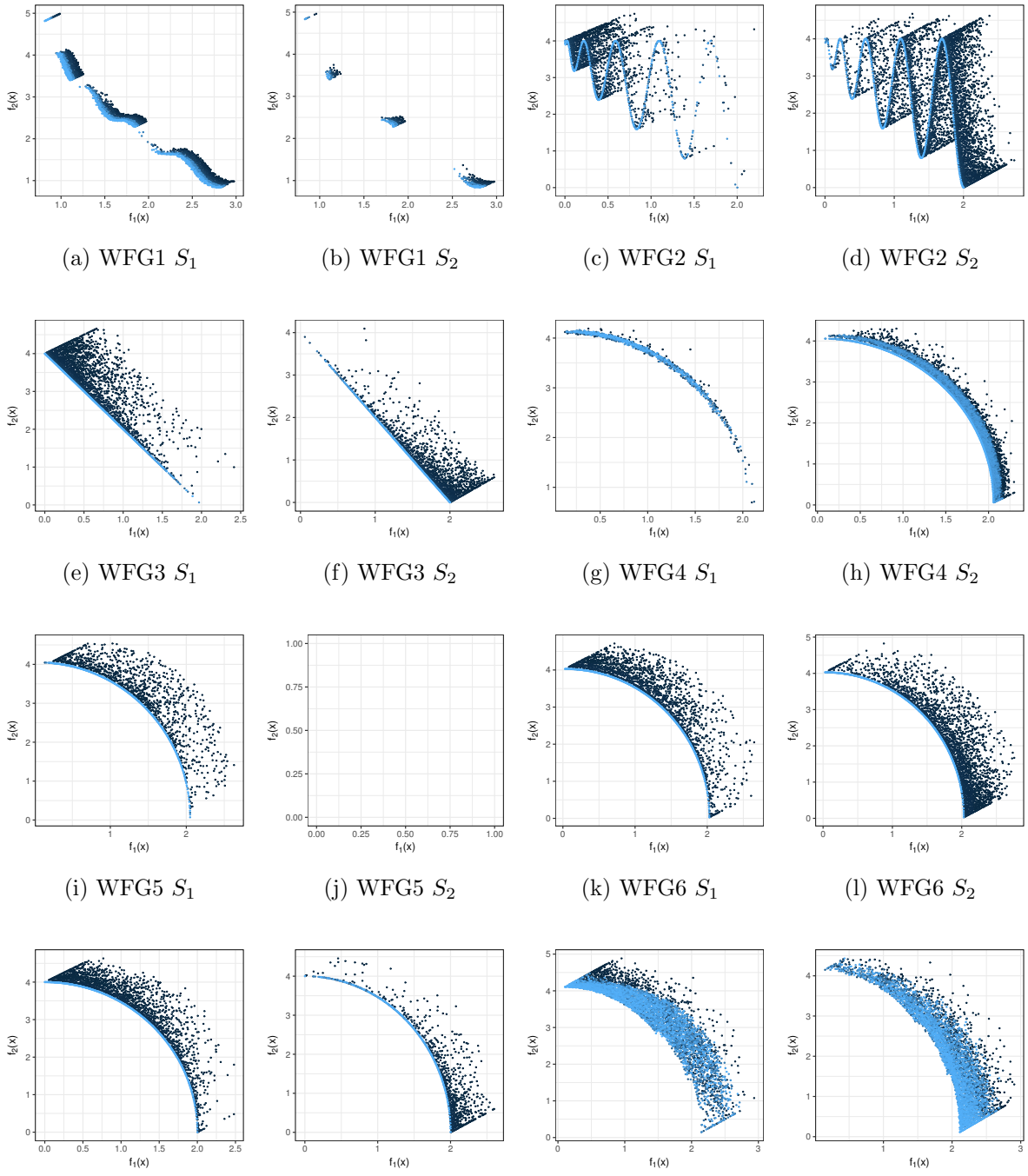
(i) ZDT6 S_1 (j) ZDT6 S_2

Figure 6.4: MGPSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations)

POF sets from the jMetal framework [80] were used for all the IGD calculations.

For comparison purposes, the IGD values for VEPSO with the random KTS, VEPSO with PCXA KTS, SMPSO [79] and OMOPSO [92] are also given. SMPSO and OMOPSO have been shown to perform well for a variety of MOPs. A summary of the control parameter values for SMPSO and OMOPSO is provided in Appendix C.

Figures 6.6(a) through 6.7(i) depict the mean IGD for each of the ZDT and 2-objective



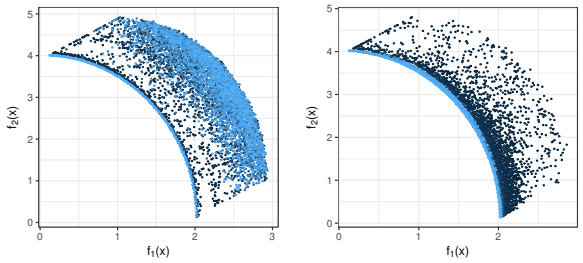
(a) WFG1 S_1 (b) WFG1 S_2 (c) WFG2 S_1 (d) WFG2 S_2

(e) WFG3 S_1 (f) WFG3 S_2 (g) WFG4 S_1 (h) WFG4 S_2

(i) WFG5 S_1 (j) WFG5 S_2 (k) WFG6 S_1 (l) WFG6 S_2

(m) WFG7 S_1 (n) WFG7 S_2 (o) WFG8 S_1 (p) WFG8 S_2

Figure 6.5: MGPSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations)



(q) WFG9 S_1 (r) WFG9 S_2

Figure 6.5: MGPSO swarms S_1 and S_2 candidate solutions for iterations 1-2000 (darker color represents lower iterations, lighter color indicates higher iterations)

WFG problems. Figures 6.8(a) through 6.8(i) depict the mean IGD for each of the 3-objective WFG problems.

The figures indicate that MGPSO managed to successfully explore and exploit the well-performing regions of the search space. The figures additionally show that MGPSO outperformed VEPSO for the majority of the test problems, especially when considering the 3-objective test problems. Only SMPSO performed better than MGPSO in a few of the 2-objective problems.

For the majority of the problems, the best-performing IGD measurement values were achieved in less than 500 iterations. WFG1 is a notable exception where the IGD measurement value continued to decrease, albeit very slowly, all the way up to iteration 2000.

For each of the test sets a win, loss, difference, ranking was calculated to statistically rank the performance of MGPSO. IGD measurement values for each algorithm were compared with all the other algorithms using the Mann-Whitney U [43] test with a confidence level of 95%. If a statistically significant difference was detected, a win was recorded for the algorithm, and a loss was recorded for the other algorithm. The difference between the wins and losses were then computed and listed in the table.

Table 6.1 lists the ranking for the ZDT test set, table 6.2 lists the ranking for the 2-objective WFG test set, and table 6.3 lists the ranking for the 3-objective WFG test set.

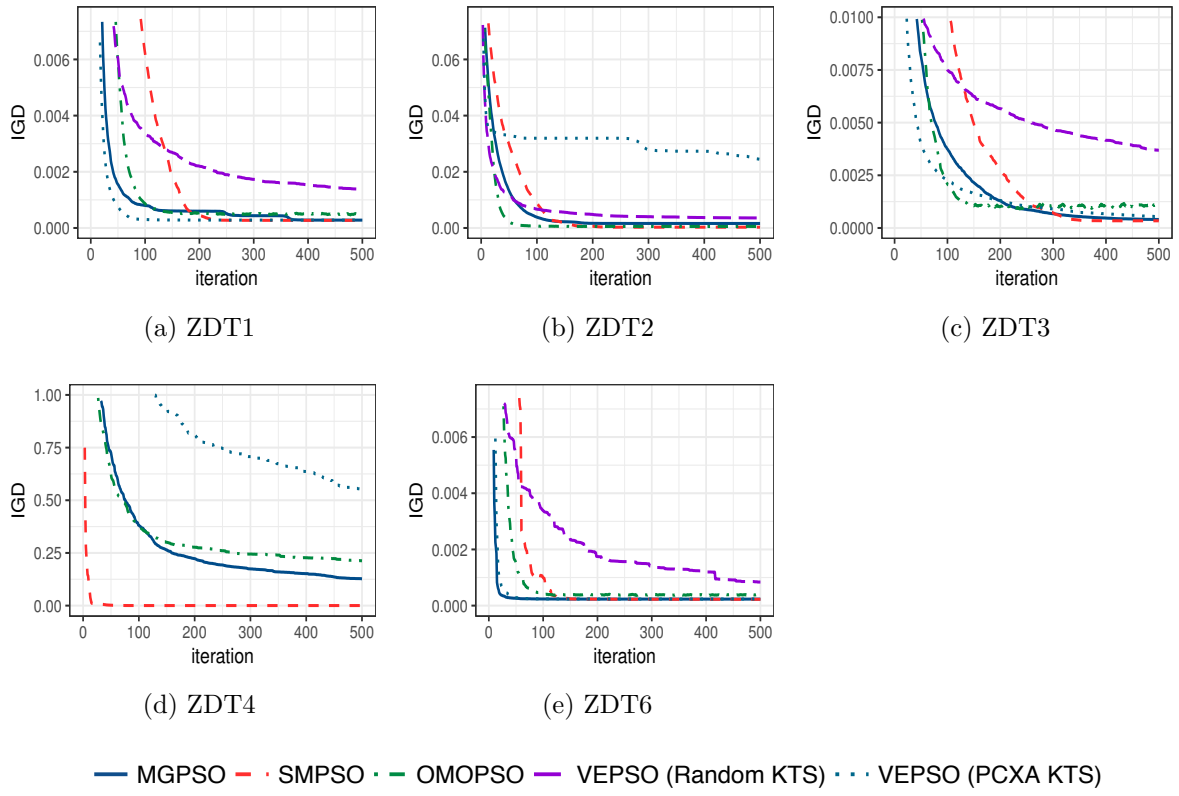


Figure 6.6: PSO inverted generational distance for ZDT problems

For the ZDT test set, MGPSO was ranked second while SMPSO was ranked first. For each of the ZDT problems, MGPSO recorded only a single loss, indicating that only SMPSO, that recorded no losses, won against it. VEPSO with the random KTS performed worst, followed by OMOPSO.

For the 2-objective WFG test set, MGPSO was ranked first followed by SMPSO. For six of the 2-objective WFG problems, MGPSO ranked first. For the remaining three problems, MGPSO ranked second twice and third once. For all three problems where MGPSO did not rank first, SMPSO did rank first. For the 2-objective WFG4 problem, OMOPSO also beat MGPSO. VEPSO with the random KTS performed worst, followed by VEPSO with the PCXA KTS.

Finally, for the 3-objective WFG test set, MGPSO ranked not only first overall, but also ranked first for each individual problem. Only for the 3-objective WFG6 problem

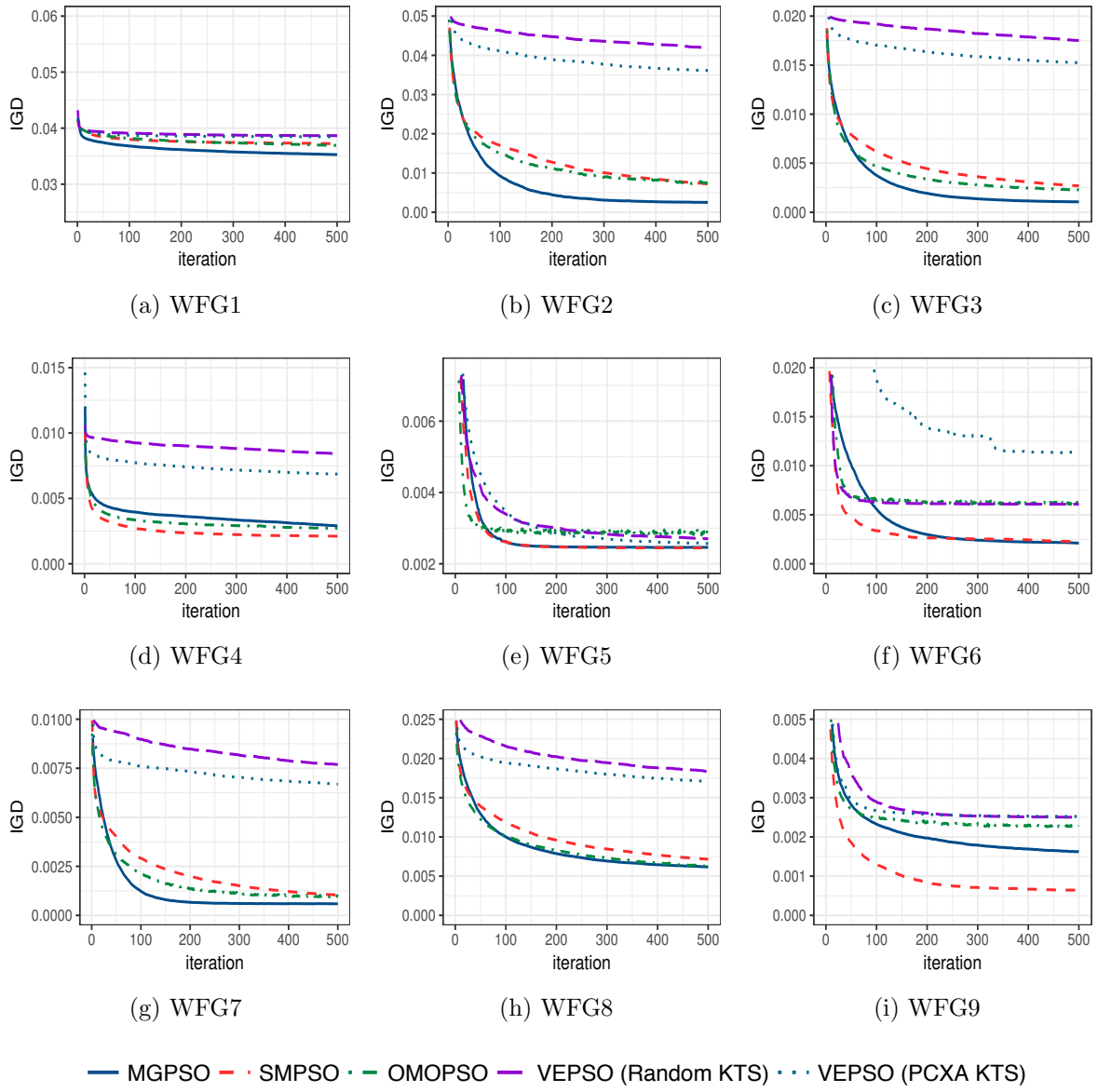


Figure 6.7: PSO inverted generational distance for 2-objective WFG problems

did SMPSO rank jointly first with MGPSO. Similar to the ZDT and 2-objective WFG results, SMPSO ranked second overall for the 3-objective WFG test set. VEPSO with the PCXA KTS performed worst, followed by VEPSO with the random KTS.

Overall, the IGD statistical analysis presented here clearly shows that MGPSO per-

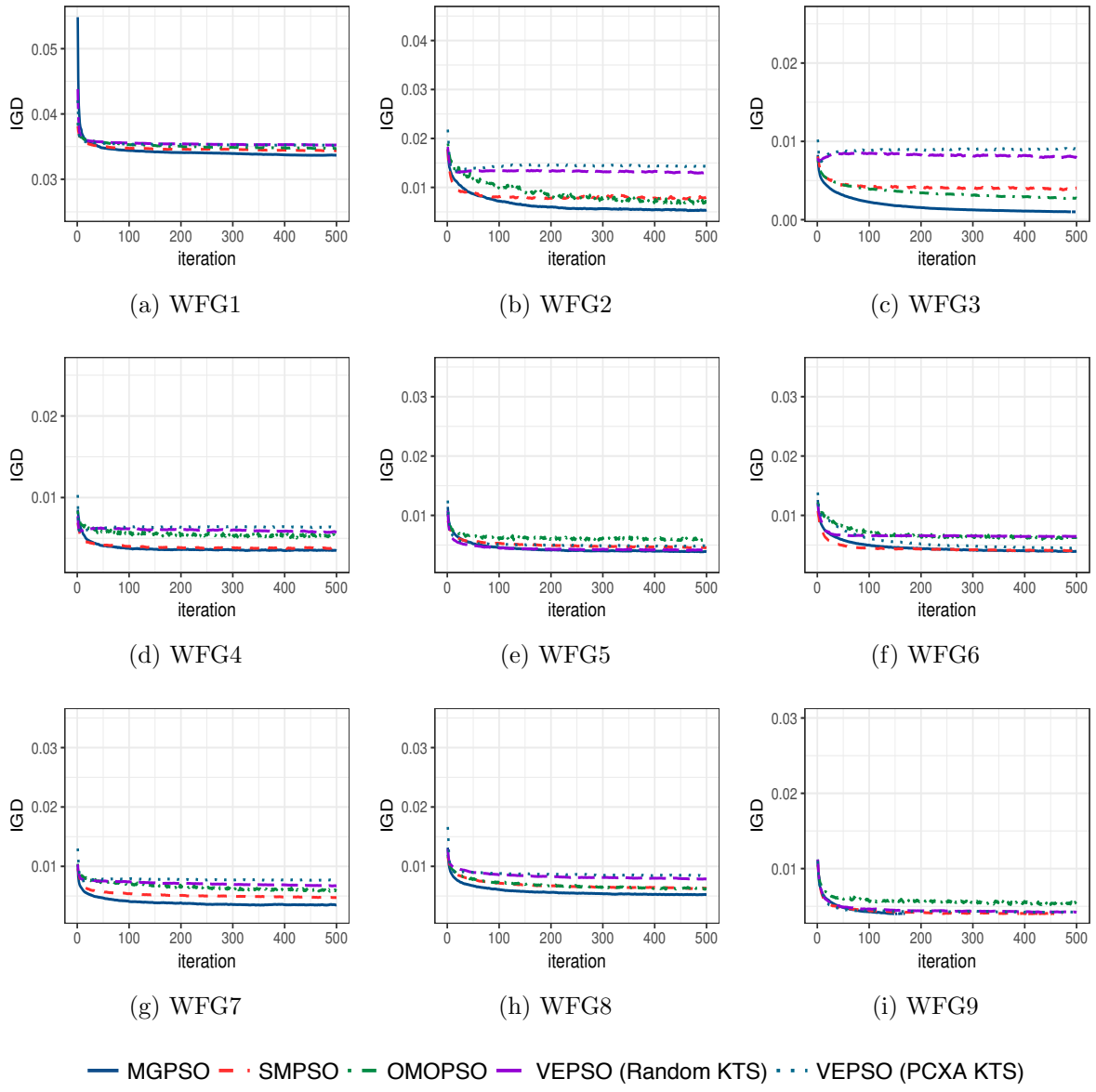


Figure 6.8: PSO inverted generational distance for 3-objective WFG problems

formed on par with leading PSO based multi-objective optimizers. MGPSO's lowest rank was 3 out of 5 for only one of the 23 problems. MGPSO was ranked first for 15 of the 23 tested problems. Overall, both of the tested VEPSO variants performed poorly, in terms of IGD, when compared to the other algorithms.

Table 6.1: Multi-objective PSO algorithms inverted generational distance ranking for ZDT1 through ZDT6

Algorithm	Result	ZDT Function					Overall
		1	**2**	**3**	**4**	**6**	
MGPSO	Wins	2	3	3	3	3	14
	Losses	1	1	1	1	1	5
	Difference	1	2	2	2	2	9
	Rank	2	2	2	2	2	2
SMPSO	Wins	4	4	4	4	4	20
	Losses	0	0	0	0	0	0
	Difference	4	4	4	4	4	20
	Rank	**1**	**1**	**1**	**1**	**1**	**1**
OMOPSO	Wins	1	2	1	2	0	6
	Losses	3	2	3	2	3	13
	Difference	-2	0	-2	0	-3	-7
	Rank	4	3	4	3	4	4
$\text{VEPSO}_{\text{Random}}$	Wins	0	1	0	0	0	1
	Losses	4	3	4	4	3	18
	Difference	-4	-2	-4	-4	-3	-17
	Rank	5	4	5	5	4	5
$\text{VEPSO}_{\text{PCXA}}$	Wins	2	0	2	1	2	7
	Losses	1	4	2	3	2	12
	Difference	1	-4	0	-2	0	-5
	Rank	2	5	3	4	3	3

6.3.3 Pareto-optimal Front Comparison

In addition to quantifying the performance of MGPSO using IGD, the porcupine measure introduced in the previous chapter was also used. Results for the porcupine measure is reported using a value pair $[a, b]$, where a indicates the percentage of the attainment surface where algorithm A performed statistically significantly better than algorithm B, and b indicates the percentage where algorithm B performed statistically significantly better than algorithm A.

Table 6.4 presents the porcupine measure for comparisons between MGPSO and other PSO-based MOO algorithms. Percentages that exceed those of the algorithm being compared against by more than 5% are shown in bold. For ease of reading, the red bolded values indicate that the attainment surface found by MGPSO outperformed the

Table 6.2: Multi-objective PSO algorithms inverted generational distance ranking for 2-objective WFG1 through WFG9

Algorithm	Result	2-objective WFG Function									Overall
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
MGPSO	Wins	4	4	4	2	3	3	4	3	3	30
	Losses	0	0	0	2	1	0	0	0	1	4
	Difference	4	4	4	0	2	3	4	3	2	26
	Rank	**1**	**1**	**1**	3	2	**1**	**1**	**1**	2	**1**
SMPSO	Wins	2	2	2	4	4	3	2	2	4	25
	Losses	2	1	2	0	0	0	2	2	0	9
	Difference	0	1	0	4	4	3	0	0	4	16
	Rank	3	2	3	**1**	**1**	**1**	3	3	**1**	2
OMOPSO	Wins	3	2	3	3	0	0	3	3	0	17
	Losses	1	1	1	1	4	3	1	0	4	16
	Difference	2	1	2	2	-4	-3	2	3	-4	1
	Rank	2	2	2	2	5	4	2	**1**	5	3
$VEPSO_{Random}$	Wins	0	0	0	0	1	2	0	0	2	5
	Losses	4	4	4	4	3	2	4	4	2	31
	Difference	-4	-4	-4	-4	-2	0	-4	-4	0	-26
	Rank	5	5	5	5	4	3	5	5	3	5
$VEPSO_{PCXA}$	Wins	1	1	1	1	2	0	1	1	1	9
	Losses	3	3	3	3	2	3	3	3	3	26
	Difference	-2	-2	-2	-2	0	-3	-2	-2	-2	-17
	Rank	4	4	4	4	3	4	4	4	4	4

attainment surface of the algorithm being compared against by more than 5%, while blue bolded values indicate that the attainment surface found by MGPSO was outperformed by the attainment surface of the algorithm being compared to by more than 5%.

The porcupine measure results show that MGPSO performed favorably when compared against the PSO-based MOO algorithms using the ZDT, 2-objective WFG, and 3-objective WFG test sets. For 78 of the 92 comparisons, MGPSO performed statistically significantly better than the algorithm being compared against. For 42 of the comparisons, MGPSO performed statistically significantly better than the algorithm being compared against by more than 90%. MGPSO only performed worse by more than 5% for 10, or less than 11%, of the comparisons. For four of the comparisons, neither algorithm outperformed the other by more than 5%.

Table 6.3: Multi-objective PSO algorithms inverted generational distance ranking for 3-objective WFG1 through WFG9

Algorithm	Result	3-objective WFG Function									Overall
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
MGPSO	Wins	4	4	4	4	4	3	4	4	4	35
	Losses	0	0	0	0	0	0	0	0	0	0
	Difference	4	4	4	4	4	3	4	4	4	35
	Rank	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
SMPSO	Wins	3	2	2	3	2	3	3	2	3	23
	Losses	1	2	2	1	2	0	1	2	1	12
	Difference	2	0	0	2	0	3	2	0	2	11
	Rank	2	3	3	2	3	**1**	2	3	2	2
OMOPSO	Wins	2	3	3	2	0	0	2	3	0	15
	Losses	2	1	1	2	4	3	2	1	4	20
	Difference	0	2	2	0	-4	-3	0	2	-4	-5
	Rank	3	2	2	3	5	4	3	2	5	3
$VEPSO_{Random}$	Wins	0	1	1	1	3	0	1	1	1	9
	Losses	3	3	3	3	1	3	3	3	2	24
	Difference	-3	-2	-2	-2	2	-3	-2	-2	-1	-15
	Rank	4	4	4	4	2	4	4	4	3	4
$VEPSO_{PCXA}$	Wins	0	0	0	0	1	2	0	0	1	4
	Losses	3	4	4	4	3	2	4	4	2	30
	Difference	-3	-4	-4	-4	-2	0	-4	-4	-1	-26
	Rank	4	5	5	5	4	3	5	5	3	5

It can be noted that MGPSO was only significantly outperformed, where the algorithm that is compared against performed better for more than 50% of the POF, only in 4 of the comparisons against SMPSO on ZDT3, ZDT4, the 2-objective WFG4, and WFG9 problems and only once against OMOPSO on ZDT3. SMPSO and OMOPSO managed to outperform MGPSO only for 10 of the 92 comparisons.

Except for the comparison against SMPSO for the ZDT problems, the results overwhelmingly show that MGPSO outperformed the other PSO-based MOO algorithms. For the comparison against SMPSO for the ZDT problems, only on ZDT6 did MGPSO outperform SMPSO. This result corresponds with the findings of the IGD statistical analysis.

Table 6.4: Multi-objective PSO algorithms porcupine measure

Problem	Objectives	MGPSO vs			
		SMPSO	OMOPSO	$\text{VEPSO}_{\text{RND}}$	$\text{VEPSO}_{\text{PCXA}}$
ZDT1	2	[8.3, 9.2]	[**71.4**, 3.5]	[**85.6**, 3.3]	[7.2, 8.9]
ZDT2	2	[4.0, **10.1**]	[**52.8**, 5.0]	[**92.9**, 1.1]	[**100.0**, 0.0]
ZDT3	2	[2.3, **85.9**]	[37.6, **56.7**]	[**96.3**, 0.0]	[**86.3**, 0.0]
ZDT4	2	[0.0, **87.7**]	[**67.9**, 0.0]	[**100.0**, 0.0]	[**100.0**, 0.0]
ZDT6	2	[**31.6**, 4.6]	[**48.7**, 15.1]	[**45.5**, 6.9]	[5.7, 6.9]
WFG1	2	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG2	2	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG3	2	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG4	2	[0.0, **78.3**]	[28.7, **34.7**]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG5	2	[5.1, **11.0**]	[**45.7**, 4.2]	[**55.4**, 1.4]	[**37.0**, 0.1]
WFG6	2	[**30.1**, 2.4]	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG7	2	[**99.9**, 0.0]	[**99.8**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG8	2	[**77.2**, 0.6]	[**43.5**, 33.3]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG9	2	[0.0, **99.9**]	[**89.1**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG1	3	[**78.0**, 4.9]	[**83.0**, 2.7]	[**90.2**, 0.8]	[**80.6**, 0.9]
WFG2	3	[**81.2**, 0.2]	[16.7, **27.7**]	[**100.0**, 0.0]	[**100.0**, 0.0]
WFG3	3	[**89.1**, 8.5]	[**81.4**, 14.1]	[**92.4**, 6.5]	[**91.1**, 7.9]
WFG4	3	[**58.6**, 11.4]	[**62.7**, 15.1]	[**100.0**, 0.0]	[**100.0**, 0.0]
WFG5	3	[**71.4**, 11.1]	[41.5, 38.5]	[**44.3**, 12.4]	[**83.1**, 0.0]
WFG6	3	[**19.4**, 11.3]	[**59.7**, 13.9]	[**84.8**, 0.0]	[**72.9**, 0.0]
WFG7	3	[**99.9**, 0.0]	[**81.7**, 1.3]	[**100.0**, 0.0]	[**100.0**, 0.0]
WFG8	3	[**86.4**, 0.0]	[**65.7**, 15.2]	[**99.9**, 0.0]	[**100.0**, 0.0]
WFG9	3	[4.9, **17.8**]	[**39.6**, 31.8]	[**49.9**, 7.9]	[**60.7**, 3.9]

6.3.4 Summary

The results presented in this section showed that MGPSO performed well when compared against PSO-based MOO algorithms for 2 and 3-objective problems. MGPSO also consistently outperformed OMOPSO, VEPSO with random KTS, and VEPSO with PCXA KTS for the majority of the tested problems.

The results also show that MGPSO is scalable to 3-objectives without sacrificing performance when compared to the other PSO-based MOO algorithms. The IGD results showed that MGPSO outperformed all the PSO-based MOO algorithms for the 3-objective WFG problems. The porcupine measure results show that, for only two of the 3-objective WFG problems, did MGPSO not outperform the algorithms being

compared against.

Overall, the results show that MGPSO is highly competitive when compared against PSO-based MOO algorithms.

6.4 Comparative Analysis

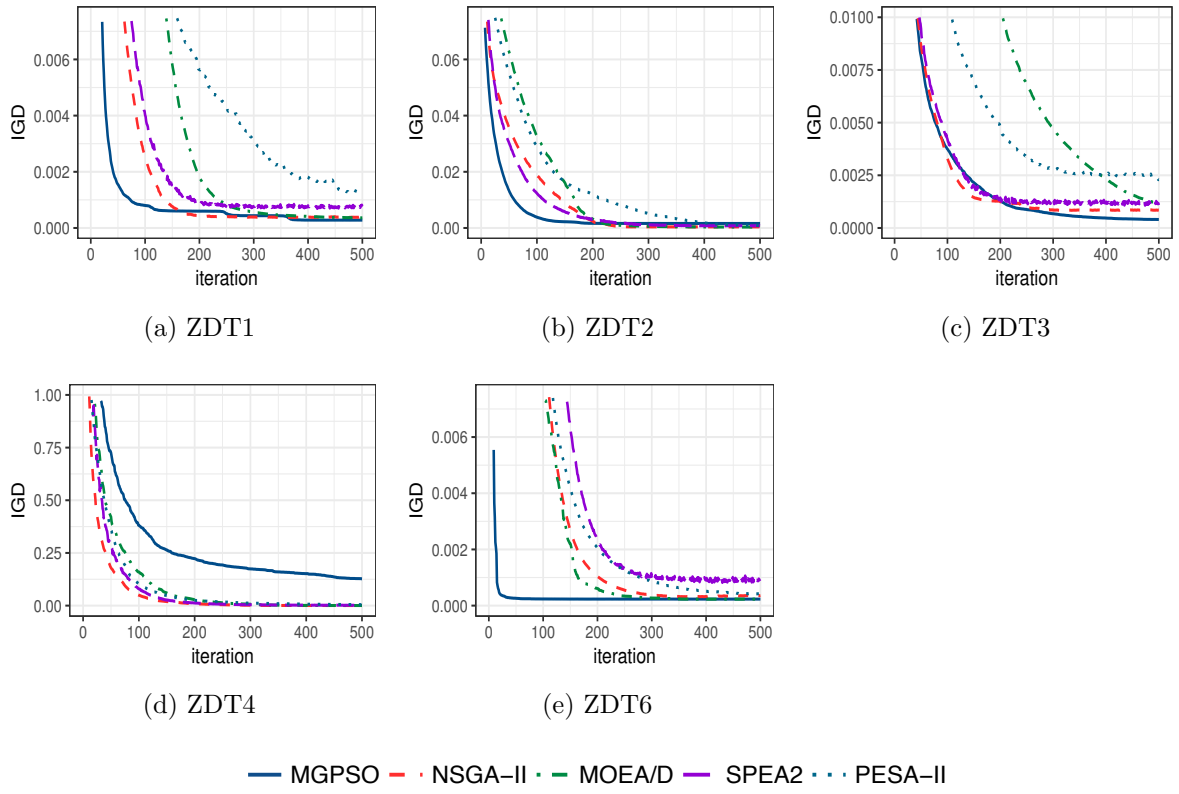
In order to rank MGPSO's performance for solving MOO problems, a comparison against the *state of the art* MOO algorithms was carried out. The non-dominated sorting genetic algorithm II (NSGA II) by Deb *et al.* [26, 29], multi-objective evolutionary algorithm based on decomposition (MOEA/D) by Zhang and Li [114], strength Pareto evolutionary algorithm 2 (SPEA2) by Zitzler *et al.* [119], and Pareto envelope-based selection algorithm II (PESA-II) by Corne *et al.* [21] algorithms were selected for the purposes of the empirical comparison. NSGA II, MOEA/D, SPEA2 and PESA-II have been shown to perform well for a variety of MOPs. A summary of the control parameter values for the algorithms being compared against is provided in Appendix C.

6.4.1 Performance Analysis

Figures 6.9(a) through 6.9(e), figures 6.10(a) through 6.10(i), and figures 6.11(a) through 6.11(i) respectively depict the IGD for the ZDT, 2-objective WFG, and 3-objective WFG test sets.

The figures show that MGPSO performed favorably against the leading MOO algorithms. For ZDT1, ZDT2, and ZDT6, the IGD for MGPSO decreased significantly faster than that of the other algorithms. However, for ZDT2, the IGD leveled out significantly higher than that of the other algorithms. For the 2-objective WFG2, MGPSO significantly outperformed all the other algorithms. For ZDT4, MGPSO performed significantly worse than the other algorithms; the obtained POF also shows the weak performance.

For both the 2-objective and 3-objective WFG1 problems, NSGA II performed notably better than any of the other MOO algorithms, including MGPSO. For the 2-objective WFG2 and WFG6 problems, MGPSO outperformed all the other MOO algorithms. MGPSO performed generally well on the 3-objective WFG with PESA-II and



(a) ZDT1 (b) ZDT2 (c) ZDT3

(d) ZDT4 (e) ZDT6

Figure 6.9: MOO inverted generational distance for ZDT problems

MOEA/D generally performing poor compared to the other MOO algorithms.

Tables 6.5, 6.6, and 6.7 list the wins, losses, difference, and ranking for the ZDT, 2-objective WFG, and 3-objective WFG test sets respectively. In all three cases, MGPSO ranked first overall.

For the ZDT test set, MGPSO ranked first for all the problems except ZDT4 where MGPSO ranked last and NSGA II first. MOEA/D and NSGA II ranked second and third, respectively, for the ZDT problems. MOEA/D ranked jointly first for ZDT4 and ZDT6. MOEA/D and NSGA II both achieved 12 wins behind MGPSO's 15 wins. MGPSO recorded losses for only ZDT4.

For the 2-objective WFG test set, MGPSO ranked first for six of the nine problems. MGPSO ranked second for WFG1 and WFG8, and fourth for WFG4. NSGA II ranked second overall and ranked jointly first for WFG1, WFG4, and WFG8. SPEA2 performed

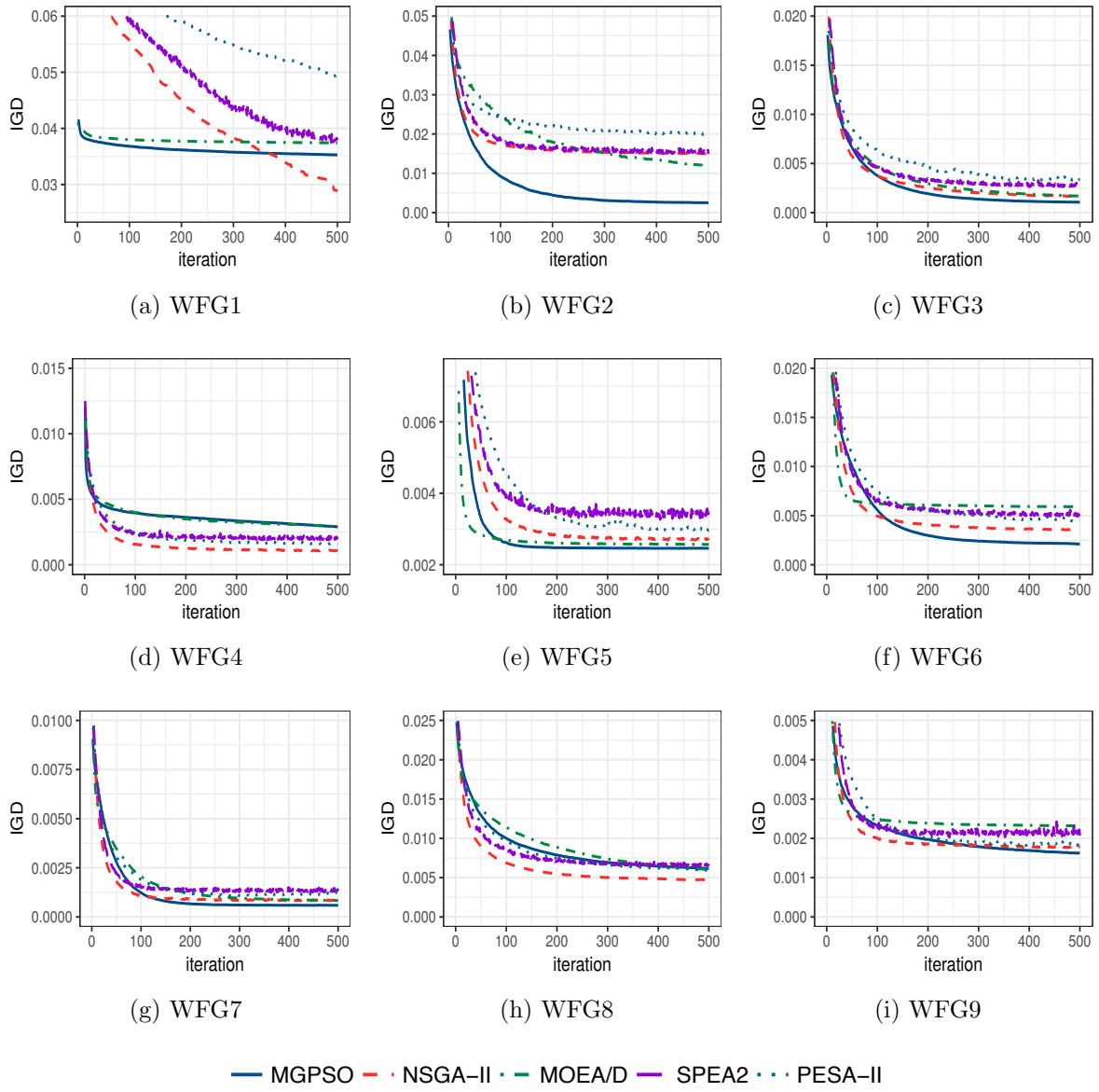


Figure 6.10: MOO inverted generational distance for 2-objective WFG problems

the worst, followed by PESA-II in fourth place.

For the 3-objective WFG2 test set, MGPSO performed even better than on the 3-objective problems. MGPSO ranked first for all but two of the problems. For the remaining two problems, WFG1 and WFG8, MGPSO ranked second. MGPSO recorded

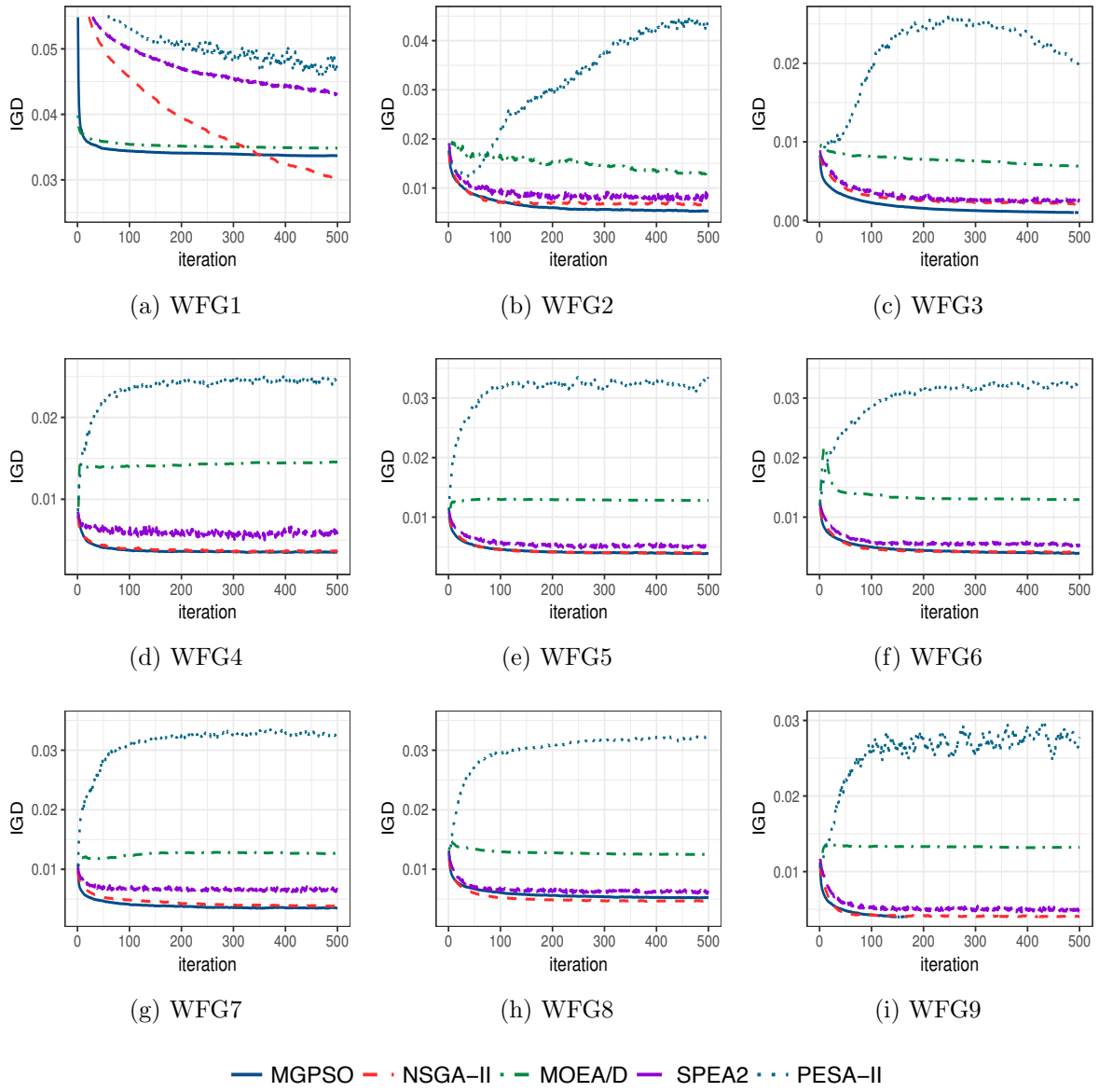


Figure 6.11: MOO inverted generational distance for 3-objective WFG problems

34 wins and only two losses, followed by NSGA II with 29 wins and seven losses. NSGA II ranked jointly first for WFG1 and WFG8 and second for the remaining seven problems. PESA-II ranked last, followed by MOEA/D in fourth place.

Table 6.5: State of the Art MOO algorithms inverted generational distance ranking for ZDT1 through ZDT6

Algorithm	Result	ZDT Function					Overall
		1	**2**	**3**	**4**	**6**	
MGPSO	Wins	4	4	4	0	3	15
	Losses	0	0	0	4	0	4
	Difference	4	4	4	-4	3	11
	Rank	**1**	**1**	**1**	5	**1**	**1**
NSGA II	Wins	2	2	3	3	2	12
	Losses	2	2	1	0	2	7
	Difference	0	0	2	3	0	5
	Rank	3	3	2	**1**	3	3
MOEA/D	Wins	3	3	0	3	3	12
	Losses	1	1	3	0	0	5
	Difference	2	2	-3	3	3	7
	Rank	2	2	5	**1**	**1**	2
SPEA2	Wins	0	0	1	2	0	3
	Losses	3	4	2	2	4	15
	Difference	-3	-4	-1	0	-4	-12
	Rank	4	5	3	3	5	4
PESA-II	Wins	0	1	0	1	1	3
	Losses	3	3	2	3	3	14
	Difference	-3	-2	-2	-2	-2	-11
	Rank	4	4	4	4	4	4

6.4.2 Pareto-optimal Front Comparison

Table 6.8 depicts the porcupine measure results for MGPSO compared against the *state of the art* MOO algorithms.

The porcupine measure results again show that MGPSO performed favorably when compared against the *state of the art* MOO algorithms using the ZDT, 2-objective WFG, and 3-objective WFG test sets. For 66 of the 92 comparisons, MGPSO performed statistically significantly better than the algorithm being compared against. For 30 of the comparisons, MGPSO performed statistically significantly better than the algorithm being compared against by more than 90%. MGPSO only performed worse by more than 5% for 21, or less than 23%, of the comparisons. For five of the comparisons, neither algorithm outperformed the other by more than 5%.

Table 6.6: State of the Art MOO algorithms inverted generational distance ranking for 2-objective WFG1 through WFG9

Algorithm	Result	2-objective WFG Function									Overall
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
MGPSO	Wins	3	4	4	0	4	4	4	1	3	27
	Losses	1	0	0	3	0	0	0	1	0	5
	Difference	2	4	4	-3	4	4	4	0	3	22
	Rank	2	**1**	**1**	4	**1**	**1**	**1**	2	**1**	**1**
NSGA II	Wins	4	2	2	4	2	3	2	4	2	25
	Losses	0	2	1	0	2	1	1	0	0	7
	Difference	4	0	1	4	0	2	1	4	2	18
	Rank	**1**	3	2	**1**	3	2	2	**1**	2	2
MOEA/D	Wins	1	2	2	0	3	0	2	1	0	11
	Losses	2	1	1	3	1	4	1	1	1	15
	Difference	-1	1	1	-3	2	-4	1	0	-1	-4
	Rank	3	2	2	4	2	5	2	2	3	3
SPEA2	Wins	1	1	1	2	0	1	0	0	0	6
	Losses	2	2	3	2	4	3	4	4	3	27
	Difference	-1	-1	-2	0	-4	-2	-4	-4	-3	-21
	Rank	3	4	4	3	5	4	5	5	5	5
PESA-II	Wins	0	0	0	3	1	2	1	1	1	9
	Losses	4	4	4	1	3	2	3	1	2	24
	Difference	-4	-4	-4	2	-2	0	-2	0	-1	-15
	Rank	5	5	5	2	4	3	4	2	3	4

NSGA II, MOEA/D, and PESA-II significantly outperformed MGPSO for ZDT4. MGPSO was also outperformed by NSGA II for ZDT3, and slightly (6.2%) by MOEA/D for ZDT6. MGPSO outperformed SPEA2 for all the ZDT problems. MGPSO was significantly outperformed by NSGA II, SPEA2, and PESA-II for the 2-objective WFG4 problem. MOEA/D only managed to slightly (8.5%) outperform MGPSO for the 2-objective WFG4 problem. MGPSO was also significantly outperformed by NSGA II and SPEA2 for the 2-objective WFG1 problem and by NSGA II and PESA-II for the 2-objective WFG8 problem. For the 3-objective WFG problems, NSGA II managed to outperform MGPSO for WFG1, WFG2, WFG4, WFG5, and WFG8. In addition to NSGA II, only SPEA2 for WFG2 managed to outperform MGPSO for the 3-objective WFG problems.

Table 6.7: State of the Art MOO algorithms inverted generational distance ranking for 3-objective WFG1 through WFG9

Algorithm	Result	3-objective WFG Function									Overall
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	
MGPSO	Wins	3	4	4	4	4	4	4	3	4	34
	Losses	1	0	0	0	0	0	0	1	0	2
	Difference	2	4	4	4	4	4	4	2	4	32
	Rank	2	**1**	**1**	**1**	**1**	**1**	**1**	2	**1**	**1**
NSGA II	Wins	4	3	3	3	3	3	3	4	3	29
	Losses	0	1	1	1	1	1	1	0	1	7
	Difference	4	2	2	2	2	2	2	4	2	22
	Rank	**1**	2	2	2	2	2	2	**1**	2	2
MOEA/D	Wins	2	1	1	1	1	1	1	1	1	10
	Losses	2	3	3	3	3	3	3	3	3	26
	Difference	0	-2	-2	-2	-2	-2	-2	-2	-2	-16
	Rank	3	4	4	4	4	4	4	4	4	4
SPEA2	Wins	1	2	2	2	2	2	2	2	2	17
	Losses	3	2	2	2	2	2	2	2	2	19
	Difference	-2	0	0	0	0	0	0	0	0	-2
	Rank	4	3	3	3	3	3	3	3	3	3
PESA-II	Wins	0	0	0	0	0	0	0	0	0	0
	Losses	4	4	4	4	4	4	4	4	4	36
	Difference	-4	-4	-4	-4	-4	-4	-4	-4	-4	-36
	Rank	5	5	5	5	5	5	5	5	5	5

With the exception of the comparison against NSGA II for the 3-objective WFG problems, the results overwhelmingly show that MGPSO outperformed the other MOO algorithms. For the 3-objective WFG problems, MGPSO outperformed NSGA II for only WFG3 and WFG7. However, the IGD statistical analysis and IGD graphs indicated that MGPSO outperformed NSGA II, at least, in terms of the IGD measure. On the other hand, the porcupine measure indicated that there are areas of the attainment surface where NSGA II outperformed MGPSO, at least, in terms of the quality of the solutions found.

Table 6.8: State of the Art MOO algorithms porcupine measure

Problem	Objectives	MGPSO vs			
		NSGAII	MOEA/D	SPEA2	PESA-II
ZDT1	2	[**49.8**, 1.6]	[**76.4**, 3.6]	[**97.9**, 0.0]	[**86.1**, 0.0]
ZDT2	2	[**34.5**, 1.8]	[**67.0**, 7.4]	[**98.8**, 0.0]	[**66.8**, 0.0]
ZDT3	2	[0.6, **77.6**]	[**100.0**, 0.0]	[**44.3**, 1.2]	[29.0, 27.3]
ZDT4	2	[15.8, **75.3**]	[0.0, **82.9**]	[**45.3**, 29.7]	[32.9, **63.2**]
ZDT6	2	[**80.7**, 0.4]	[37.9, **44.1**]	[**99.7**, 0.1]	[**99.9**, 0.0]
WFG1	2	[17.3, **78.8**]	[**92.8**, 2.0]	[25.0, **69.7**]	[46.4, 50.9]
WFG2	2	[**99.2**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG3	2	[**99.9**, 0.0]	[**90.7**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG4	2	[0.0, **99.9**]	[20.0, **28.5**]	[1.3, **97.4**]	[4.5, **92.7**]
WFG5	2	[**43.8**, 4.7]	[**42.1**, 36.2]	[**97.6**, 0.0]	[**89.5**, 0.0]
WFG6	2	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG7	2	[**95.1**, 0.0]	[**81.5**, 2.9]	[**99.9**, 0.0]	[**99.9**, 0.0]
WFG8	2	[0.0, **99.7**]	[28.0, **45.6**]	[**35.6**, 7.3]	[17.3, **70.6**]
WFG9	2	[0.0, **7.0**]	[**99.7**, 0.0]	[**29.4**, 0.1]	[**9.0**, 0.0]
WFG1	3	[5.5, **89.8**]	[**86.9**, 9.2]	[**64.4**, 25.2]	[**68.7**, 25.5]
WFG2	3	[29.2, **50.8**]	[**60.0**, 20.7]	[28.5, **55.0**]	[**89.2**, 1.4]
WFG3	3	[**57.6**, 22.5]	[**84.2**, 15.0]	[**71.3**, 13.7]	[**56.7**, 31.1]
WFG4	3	[2.7, **78.3**]	[**64.8**, 21.7]	[**50.4**, 3.9]	[**90.7**, 5.7]
WFG5	3	[6.9, **55.7**]	[**50.9**, 38.8]	[**42.3**, 15.1]	[**90.7**, 3.6]
WFG6	3	[10.0, 13.7]	[**69.3**, 9.2]	[**93.9**, 0.0]	[**94.1**, 3.1]
WFG7	3	[**26.8**, 5.7]	[**71.3**, 12.4]	[**100.0**, 0.0]	[**90.4**, 6.5]
WFG8	3	[0.09, **82.5**]	[**69.6**, 22.0]	[37.8, 38.8]	[**94.4**, 2.5]
WFG9	3	[18.0, 16.0]	[**54.0**, 34.9]	[**40.5**, 7.0]	[**85.6**, 7.9]

6.4.3 Summary

The results presented in this section showed that MGPSO performed well when compared against the *state of the art* MOO algorithms for 2 and 3-objective problems. MGPSO also consistently outperformed MOEA/D, SPEA2, and PESA-II for the majority of the tested problems.

The results also show that MGPSO is scalable to 3-objectives without sacrificing performance when compared to the other *state of the art* MOO algorithms. The IGD results showed that MGPSO ranked overall first for the ZDT, the 2-objective WFG, and the 3-objective WFG test sets. The porcupine measure results showed that, while MGPSO outperformed NSGA II in terms of the IGD measure, NSGA II found higher

quality solutions for the majority of the 3-objective WFG problems for larger percentages of the attainment surface than MGPSO did.

Overall, the results show that MGPSO is highly competitive when compared to the *state of the art* MOO algorithms.

It should be noted that the only conclusion that can be drawn from the results presented in this chapter is that MGPSO is **highly competitive**. An absolute finding that MGPSO outperforms the other MOO algorithms can not be made due to the difficulties in conducting a *fair* comparison. The results presented in this chapter was obtained by running the competing algorithms with recommended parameter values as provided in the literature with a fixed population size for a fixed number of iterations. If a fixed function evaluation budget is used, varying population sizes may yield different results for the different algorithms. The objective of this section was to determine if MGPSO performs on-par with the *state of the art* MOO algorithms, and this objective has been met through the presented analysis and results.

6.5 Summary

This chapter introduced MGPSO, a multi-objective PSO algorithm. MGPSO was compared against current MOO PSOs and the *state of the art* MOO algorithms. IGD and the porcupine measure were used to evaluate MGPSO's performance.

The results indicated that MGPSO performed on-par with the *state of the art* MOO algorithms, and in many cases outperformed the algorithms against which it was compared. MGPSO was shown to perform well on both 2- and 3-objective problems using the ZDT and WFG test sets. MGPSO was ranked overall first in five of the six IGD-based statistical analysis results. The porcupine measure also showed that MGPSO tended to outperform the competing algorithms and managed to find higher quality solutions for the majority of the POFs that make up the attainment surfaces for the tested problems.

Overall, MGPSO succeeded in solving the MOO test problems evaluated in this study and was shown to be highly competitive compared to the other PSO-based and *state of the art* MOO algorithms. MGPSO should be considered a good candidate algorithm for solving MOO algorithms.

Chapter 7

Multi-guided Particle Swarm Optimization Parameter Sensitivity Analysis

"The skeptic does not mean him who doubts, but him who investigates or researches, as opposed to him who asserts and thinks that he has found."

Miguel de Unamuno (1864 - 1936)

The previous chapter introduced the MGPSO and evaluated its performance. The performance evaluation made use of optimized control parameter values. The main objective of this chapter is to discuss the process used to obtain the optimized MGPSO control parameter values and to perform a theoretical stability analysis of the MGPSO control parameter values. A summary of the MGPSO's optimized control parameter values for the ZDT, the 2-objective WFG, and the 3-objective WFG is also given.

The parameter optimization process presented in this chapter makes use of the parameter space exploration technique using parallel coordinates described by Franken [39]. The optimized control parameter values obtained using the technique presented in this chapter was used to obtain the MGPSO's experimental results shown in the previous chapter.

Section 7.1 introduces parameter optimization, followed by a discussion of the MGPSO's control parameter optimization process in Section 7.2. The control parameter

parallel coordinate plots and control parameter optimization results are given in Section 7.3. Section 7.4 presents a theoretical derivation of the order-1 and order-2 stable regions for the MGPSO algorithm. Finally, a summary of the findings is given in Section 7.5.

7.1 Introduction

Over the years researchers have applied various approaches to optimize an algorithm's control parameter values [39]. The methods applied by researchers to select optimal control parameter values include, but is not limited to:

- *One-factor-at-a-time design* allows each control parameter value to be optimized in isolation by fixing the values of all the other control parameters. While efficient in terms of computations required, an obvious drawback to the one-factor-at-a-time design approach is that interdependencies between the control parameters are not taken into account.

- *Factorial design* improves on the one-factor-at-a-time design approach by capturing the control parameter interdependencies by evaluating all possible control parameter value combinations. The computational cost associated with the factorial design approach makes it unsuitable for all but the most trivial cases.

- *Random sampling* improves on the factorial design approach by sampling random control parameter value combinations instead of evaluating all possible combinations. For each control parameter, only a lower and upper boundary is required for the sampling process. While random sampling improves the computational cost, the nature of random number generators often leads to the exploration of only a small region of the parameter space.

- *Low-discrepancy sequences* improve on the random sampling approach by sampling parameter value combinations using low-discrepancy sequences (also known as quasi-random numbers). Low-discrepancy sequences seem random but are in reality much more uniformly spread out over the parameter space. The primary objective of a finite low-discrepancy sequence is to attempt to fill the unit hypercube as uniformly as possible. Various low-discrepancy sequences generators

have been proposed, namely Van der Corput sequences, Halton sequences, Faure sequences, Sobol sequences and Niederreiter sequences [42]. Low-discrepancy sequence approaches reduce the computational cost while still allowing exploration of large areas of the parameter space.

Procedures, such as F-Race, for automated selection and evaluation of control parameter value combinations have also been proposed [6, 65]. These procedures often rely on statistical significance tests in an attempt to optimize the control parameter values within a fixed computational budget.

7.2 Analyzing Control Parameter Sensitivity

Franken [39] presented a visual analysis technique to explore the control parameter space of an algorithm. Franken proposed plotting the results for all the evaluated control parameter value combinations on a parallel coordinate plot. Each control parameter value combination is represented by a single pattern on the parallel coordinate plot. The color for each pattern is determined according to how well the control parameter value combination being represented, performed. By adjusting the colors and highlighting the region of the well-performing patterns a researcher can visually explore the parameter space.

In the case of MGPSO, the following control parameter values must be optimized:

- Number of particles per subswarm $S_m.n_s$ (denoted $|S_m|$) for $m = \{1, 2, ..., n_m\}$,
- the tournament selection competition pool size (T),
- inertia weight (w), and
- the three acceleration coefficients (c_1, c_2, c_3).

In order to optimize the control parameter values, values were sampled for each parameter from the following predefined parameter value domains:

- $|S_m| \in \{0, 1, \ldots, 50\}$ with $m \in \{1, \ldots, n_m\}$ such that $\sum |S_m| = 50$,
- $T \in \{2, 3\}$,

- $w \in \{0.05, 0.075, 0.1, \ldots, 0.95\}$, and
- $c_1, c_2, c_3 \in \{0.50, 0.55, 0.6, \ldots, 1.9\}$.

Experimentation showed that these control parameter values could lead to well-performing results and, as such, is well suited for the control parameter sensitivity analysis approach described in this chapter. For each control parameter value combination, the IGD was calculated and plotted onto the parallel coordinate plot. As many control parameter value combinations as possible should be evaluated to identify the well-performing areas of the parameter space. For this study, over 2500 control parameter value combinations were evaluated per problem.

The control parameter sensitivity analysis should be repeated for each of the problems being optimized. In the case of this study, the control parameter sensitivity analysis was performed 23 times, once for each problem in the ZDT, 2-objective WFG, and 3-objective WFG test sets. The true POF sets from the jMetal framework [80] were used for all the IGD calculations.

Take note that Franken's analysis technique requires an absolute measurement value to optimize, as such the porcupine measure is not a suitable measure due to its comparative and multi-output nature.

7.3 Parameter Sensitivity Analysis

Figures 7.1 through 7.5, 7.6 through 7.14, and 7.15 through 7.23 present the parallel coordinate plots for the ZDT, 2-objective WFG, and 3-objective WFG test sets, respectively. The top 2.5% well-performing control parameter value combinations, shown as solid black lines, are highlighted in green. The top 10 well-performing control parameter value combinations are shown as dashed orange lines, and the best performing control parameter value combination is shown as a solid red line.

For ZDT1 through ZDT6 a clear pattern is visible where well-performing control parameter value combinations have more particles in subswarm S_2 than in subswarm S_1. Both tournament pool sizes $T = 2$ and $T = 3$ performed well. The inertia weight, w, did not show much sensitivity; however, the top 10 values seemed to cluster for at least three

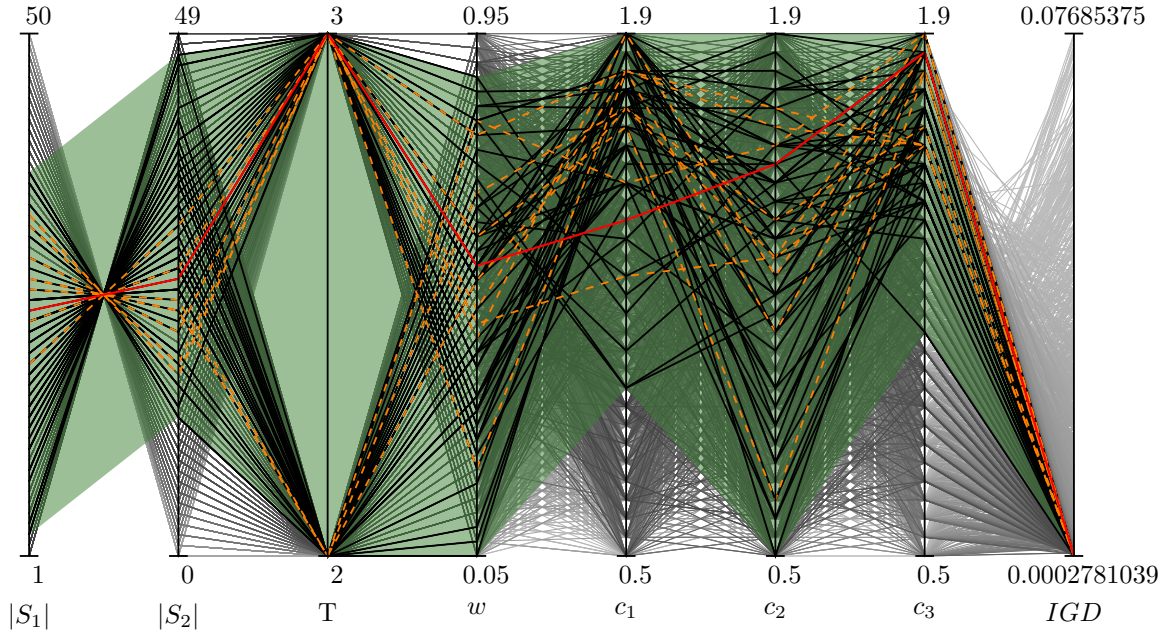


Figure 7.1: MGPSO control parameter value parallel coordinate plot for ZDT1

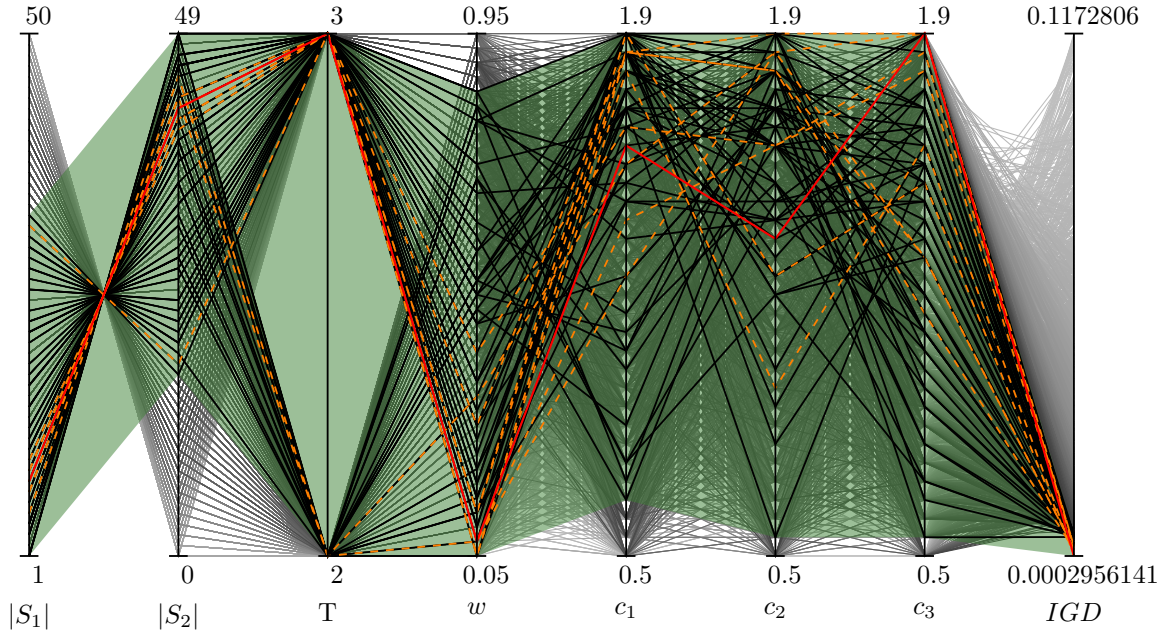


Figure 7.2: MGPSO control parameter value parallel coordinate plot for ZDT2

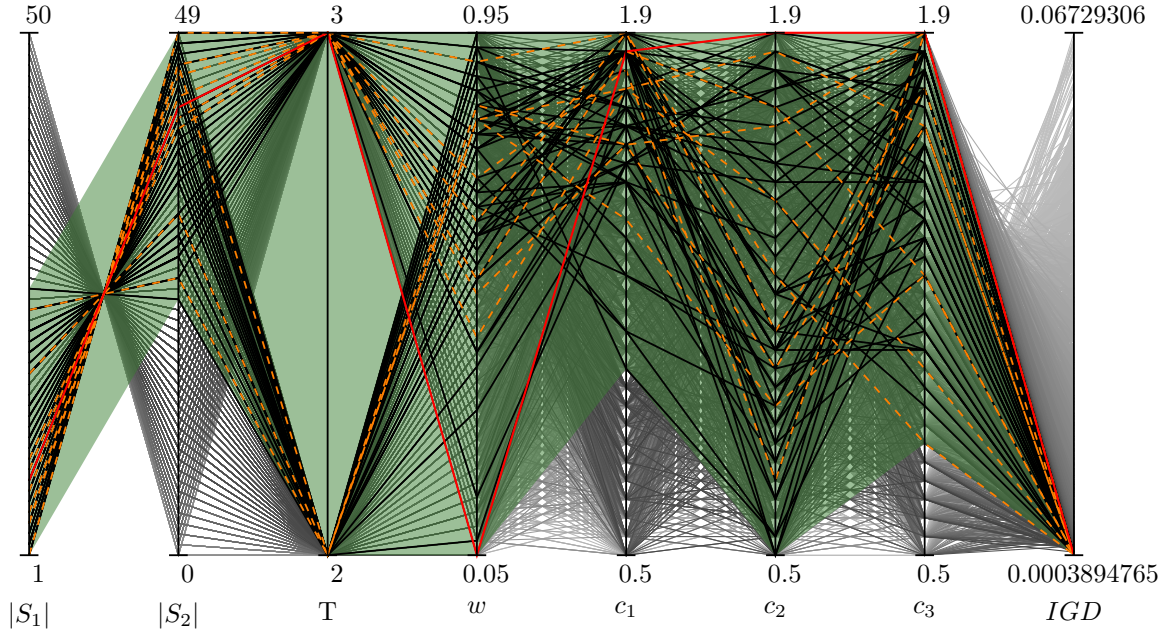


Figure 7.3: MGPSO control parameter value parallel coordinate plot for ZDT3

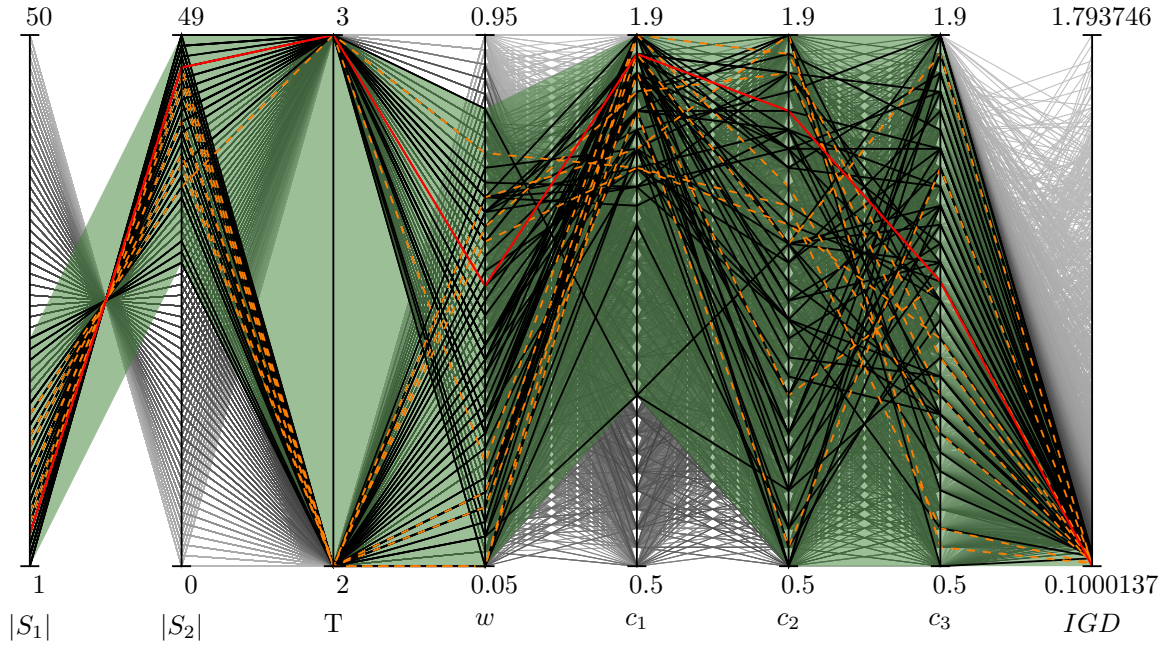


Figure 7.4: MGPSO control parameter value parallel coordinate plot for ZDT4

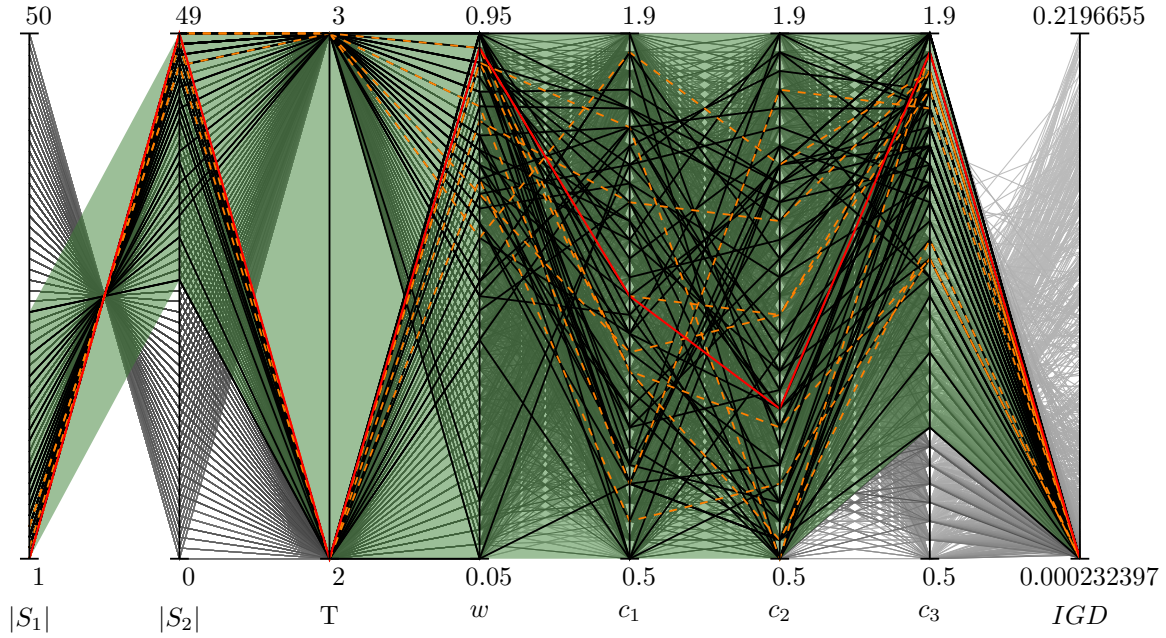


Figure 7.5: MGPSO control parameter value parallel coordinate plot for ZDT6

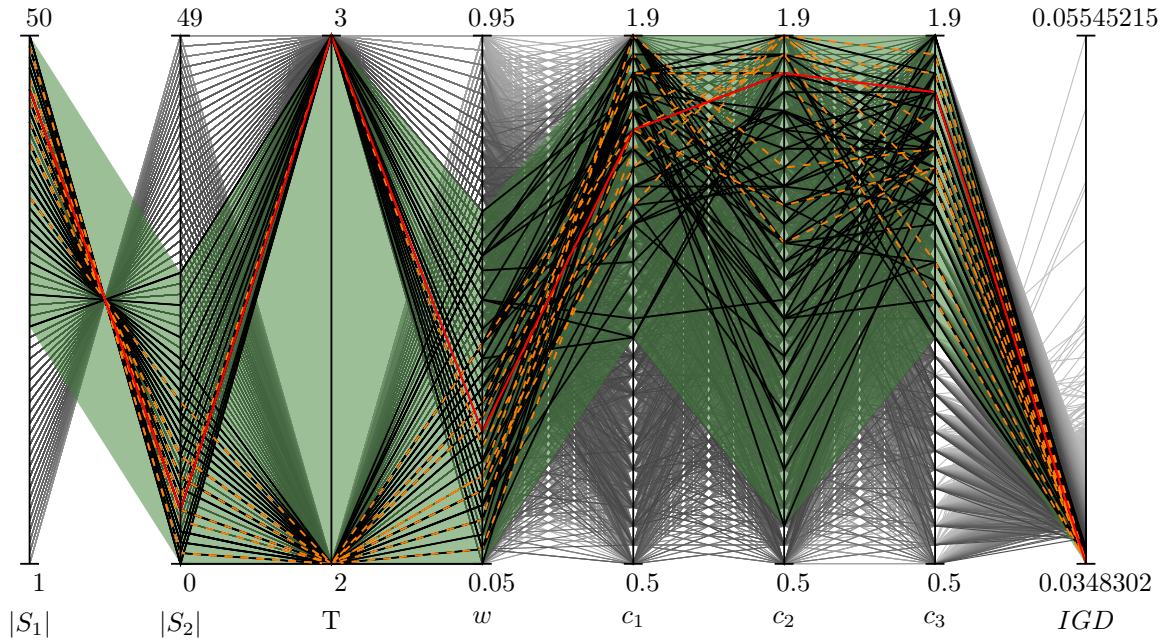


Figure 7.6: MGPSO control parameter value parallel coordinate plot for WFG1 (2 objective)

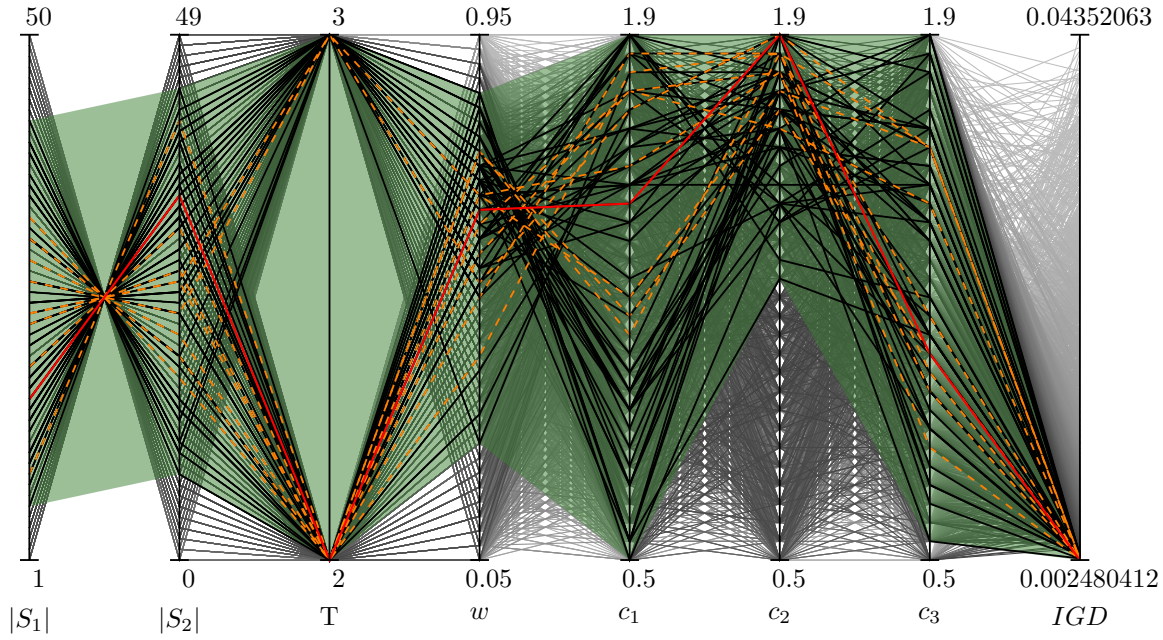


Figure 7.7: MGPSO control parameter value parallel coordinate plot for WFG2 (2 objective)

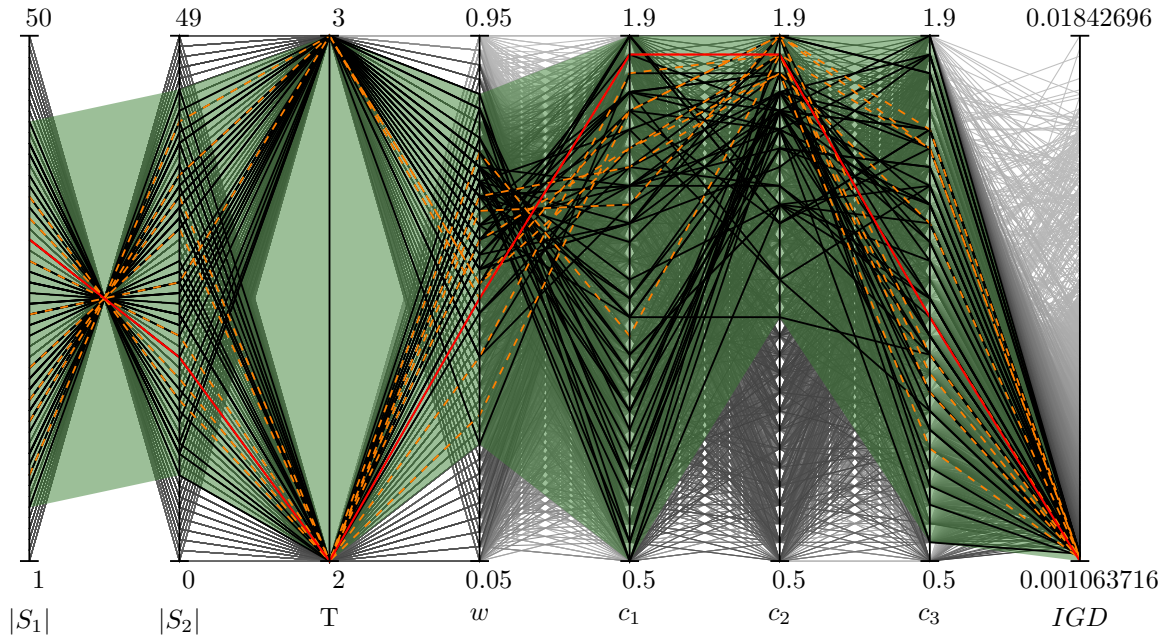


Figure 7.8: MGPSO control parameter value parallel coordinate plot for WFG3 (2 objective)

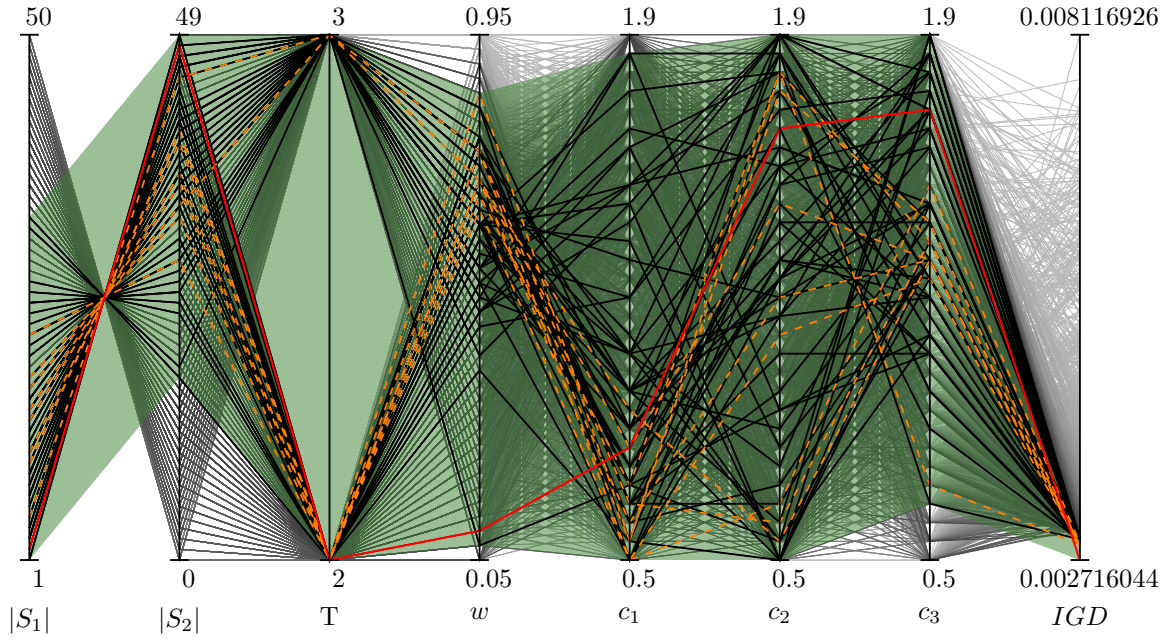


Figure 7.9: MGPSO control parameter value parallel coordinate plot for WFG4 (2 objective)

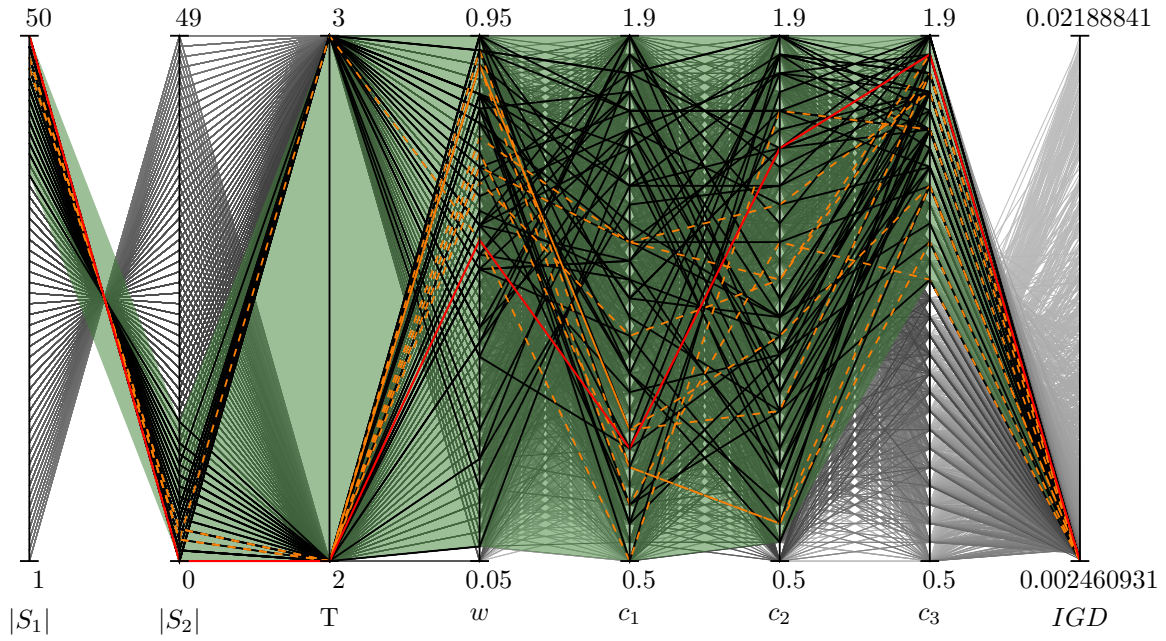


Figure 7.10: MGPSO control parameter value parallel coordinate plot for WFG5 (2 objective)

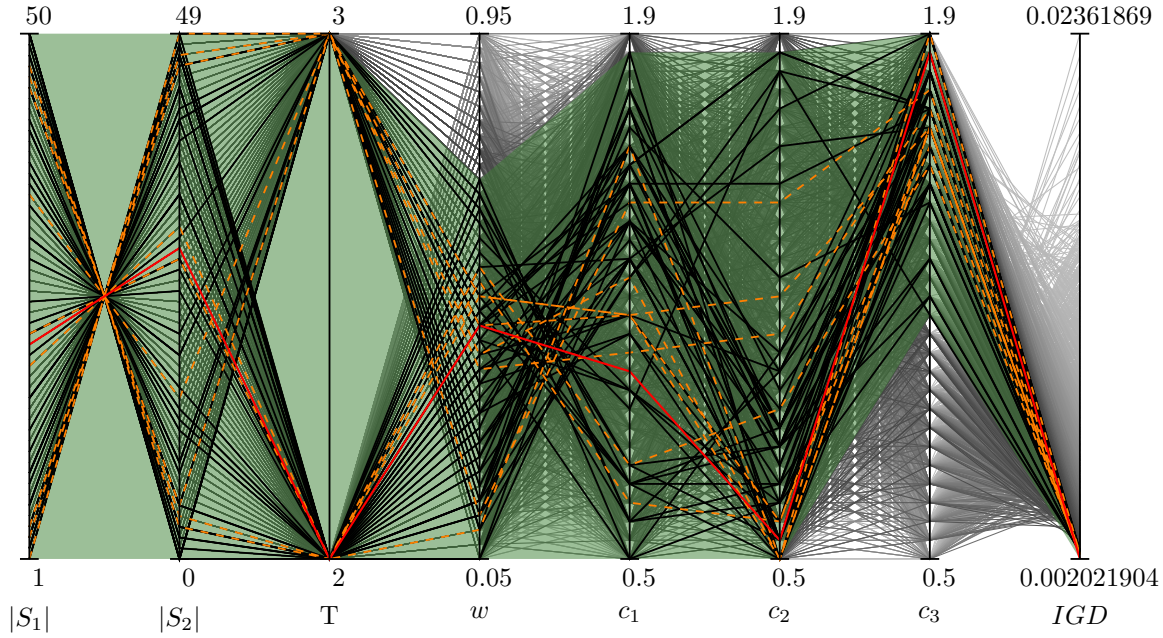


Figure 7.11: MGPSO control parameter value parallel coordinate plot for WFG6 (2 objective)

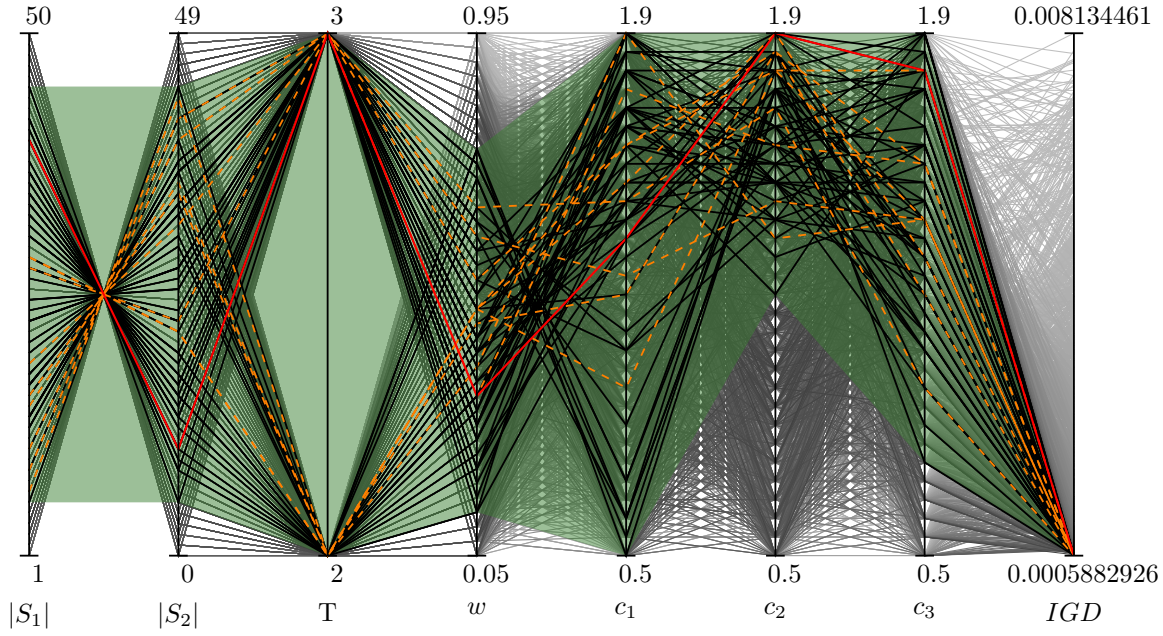


Figure 7.12: MGPSO control parameter value parallel coordinate plot for WFG7 (2 objective)

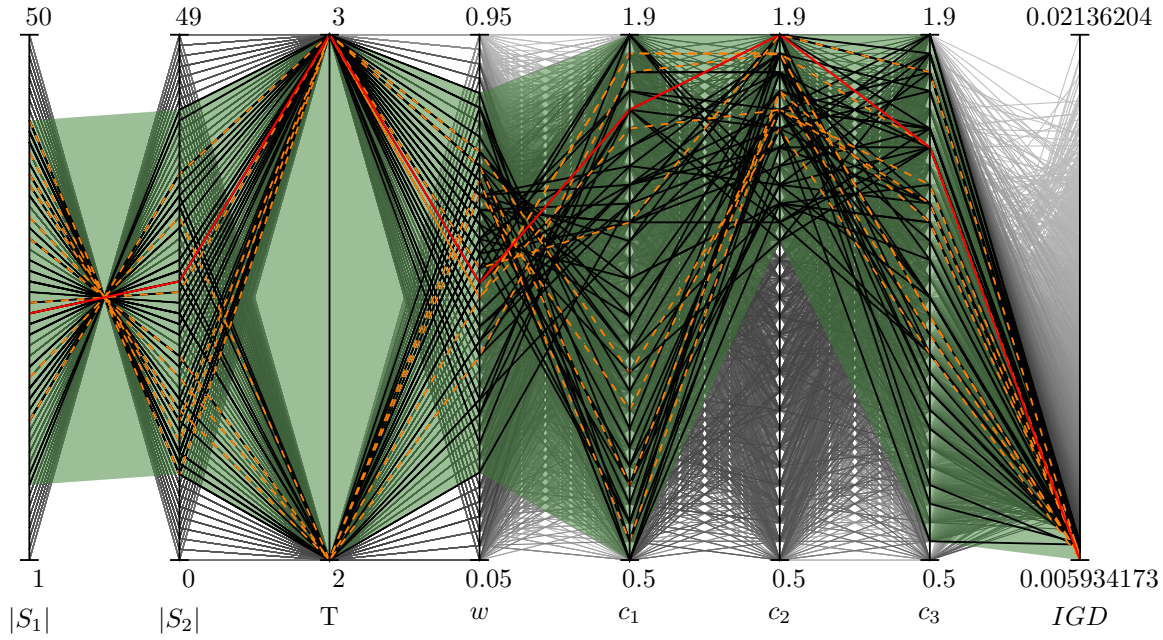


Figure 7.13: MGPSO control parameter value parallel coordinate plot for WFG8 (2 objective)

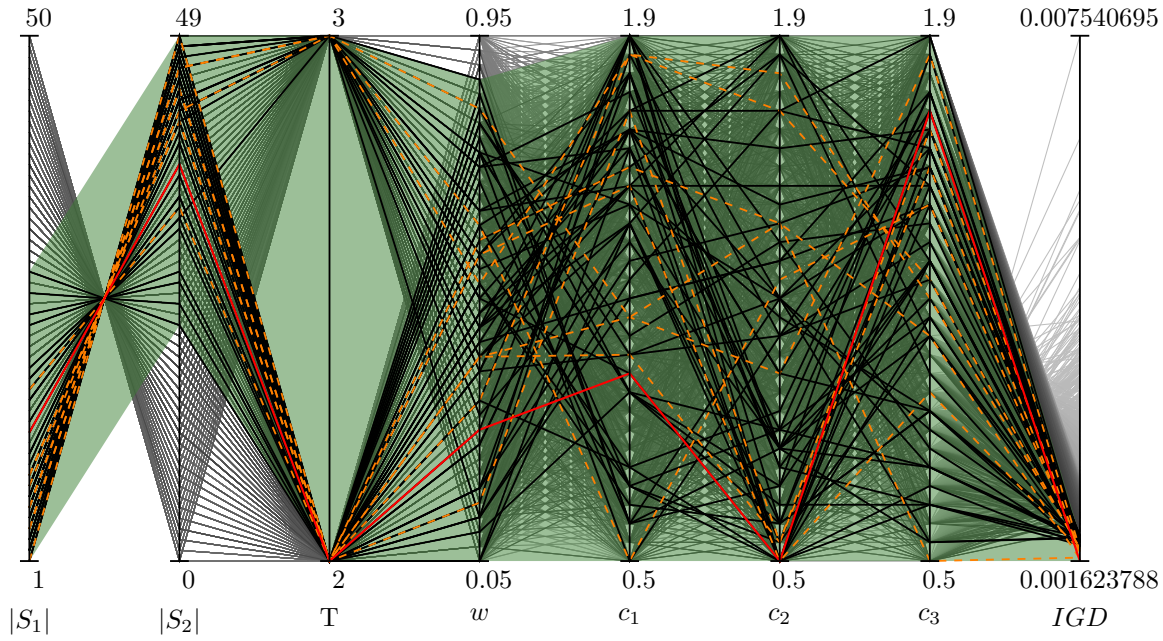


Figure 7.14: MGPSO control parameter value parallel coordinate plot for WFG9 (2 objective)

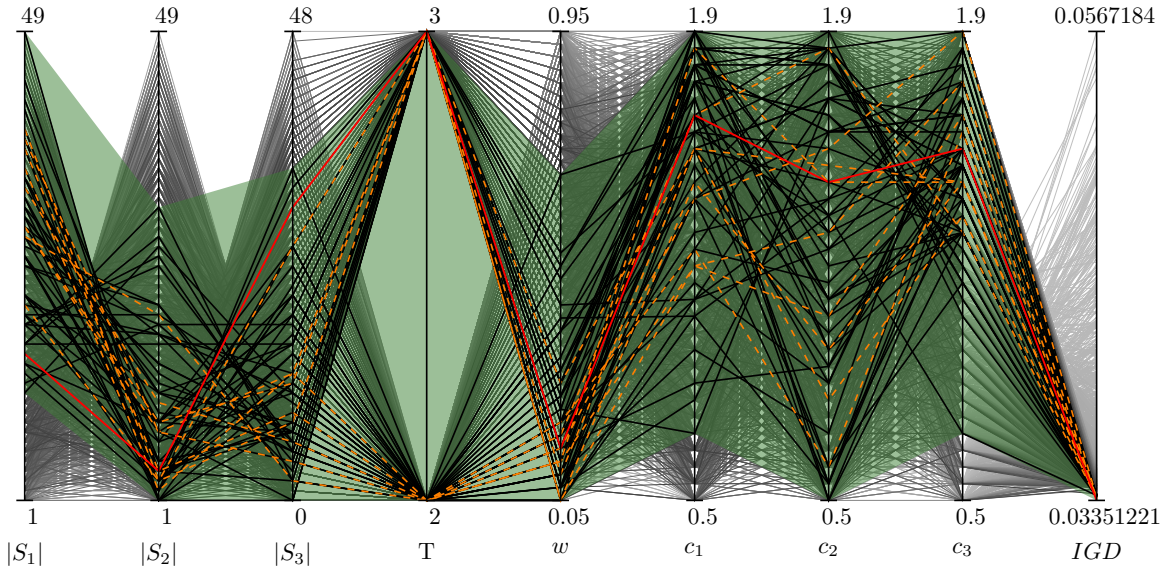


Figure 7.15: MGPSO control parameter value parallel coordinate plot for WFG1 (3 objective)

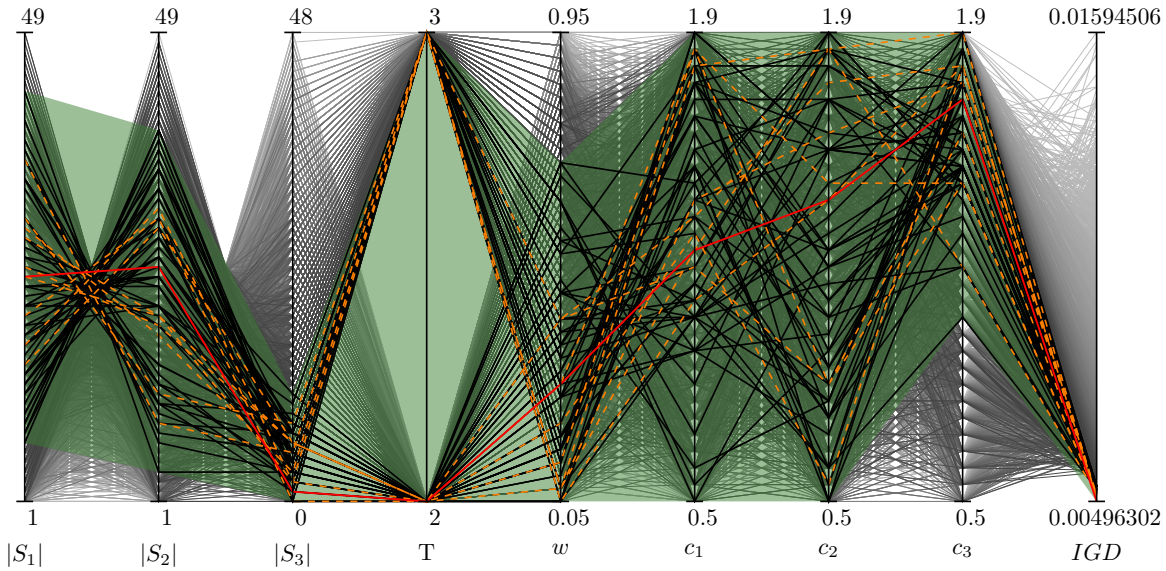


Figure 7.16: MGPSO control parameter value parallel coordinate plot for WFG2 (3 objective)

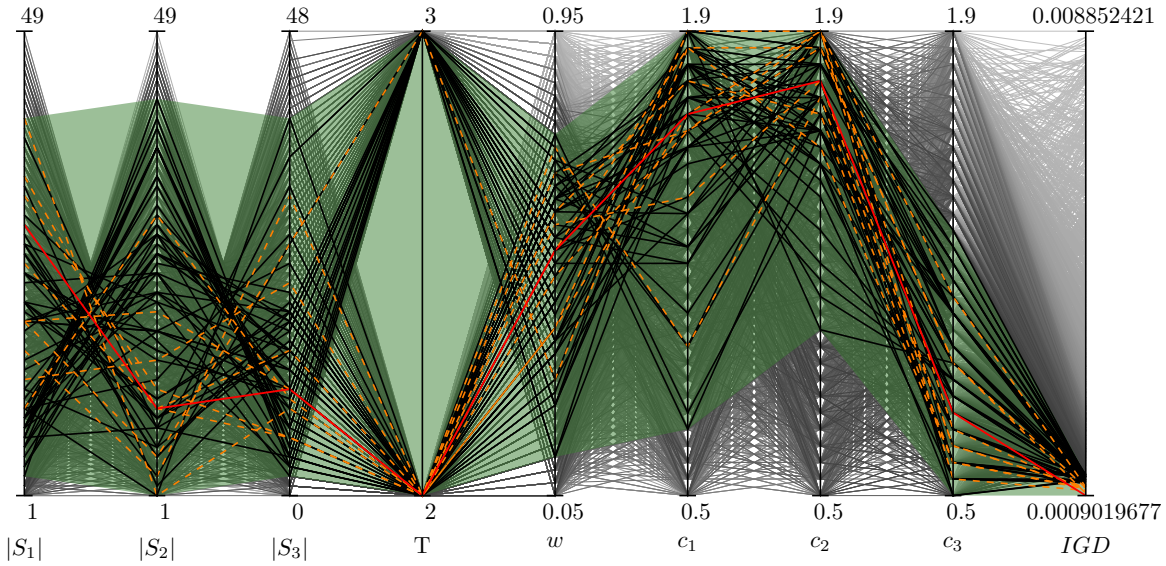


Figure 7.17: MGPSO control parameter value parallel coordinate plot for WFG3 (3 objective)

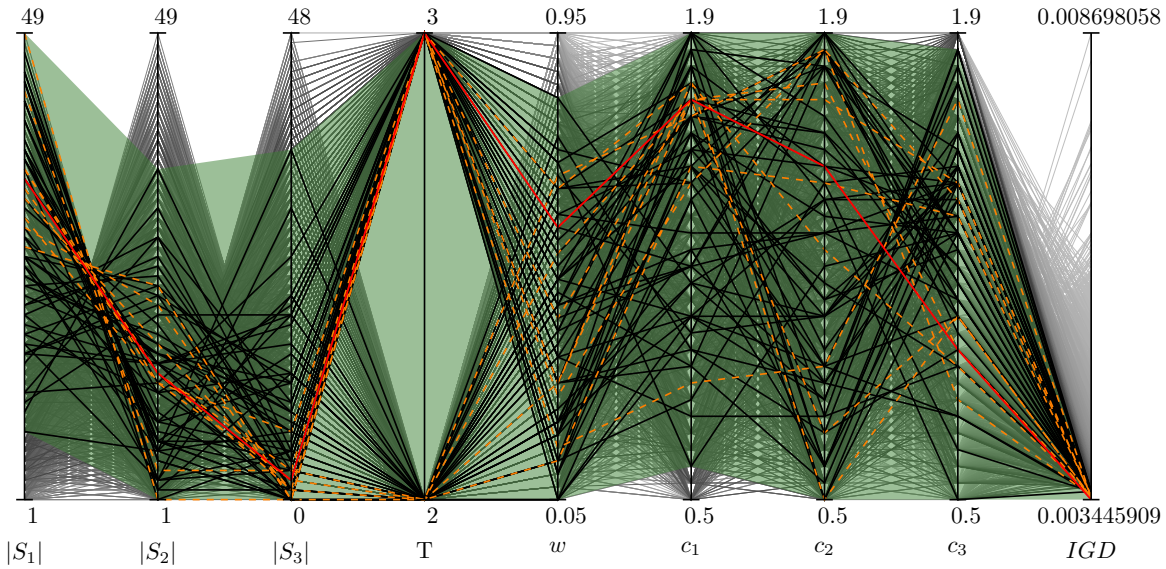


Figure 7.18: MGPSO control parameter value parallel coordinate plot for WFG4 (3 objective)

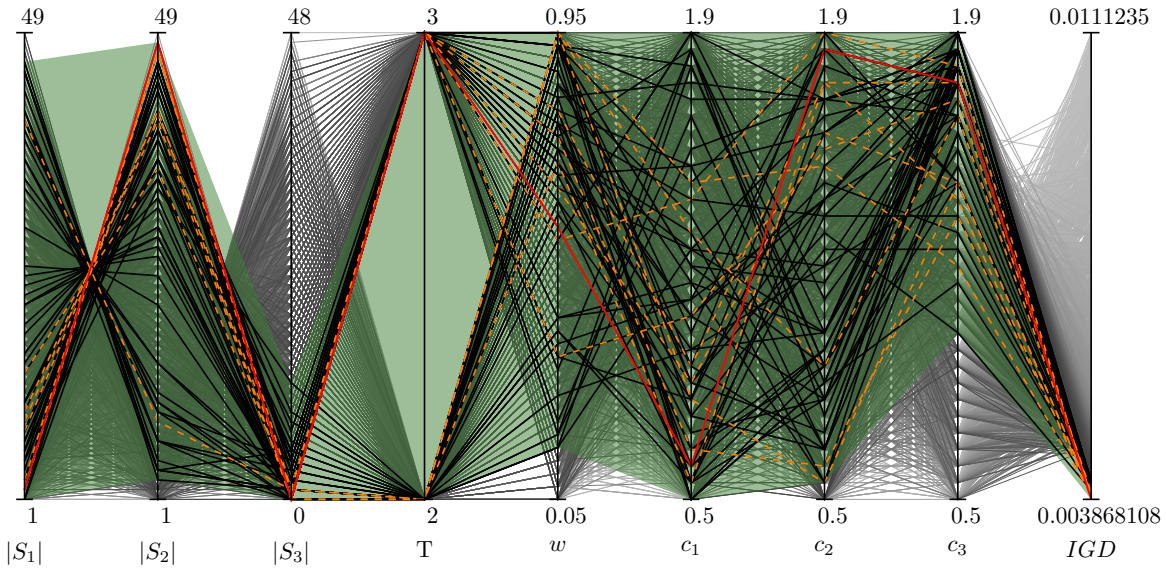


Figure 7.19: MGPSO control parameter value parallel coordinate plot for WFG5 (3 objective)

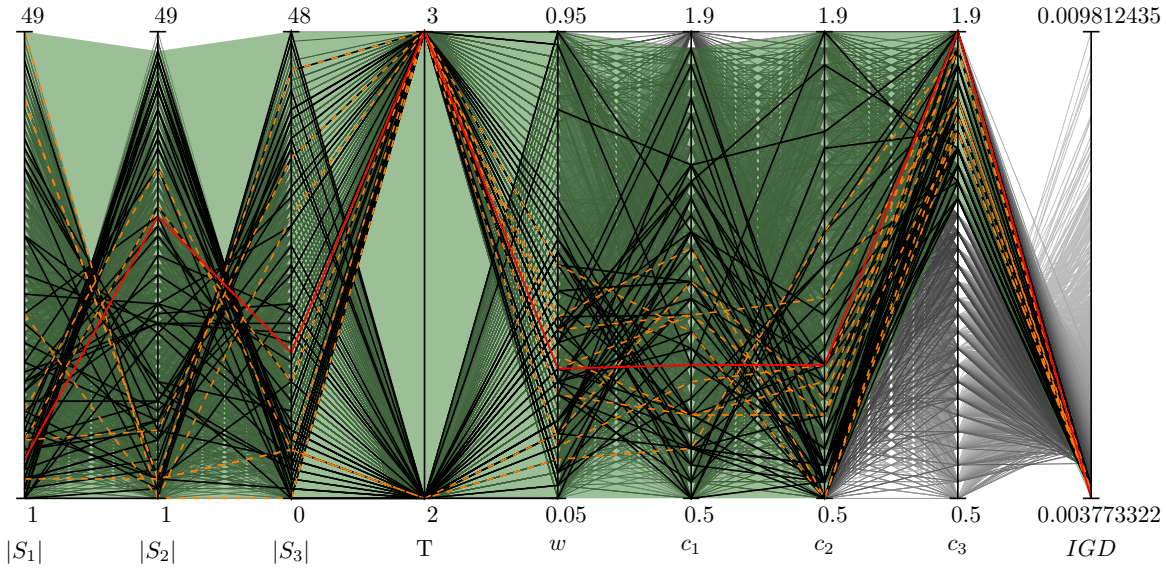


Figure 7.20: MGPSO control parameter value parallel coordinate plot for WFG6 (3 objective)

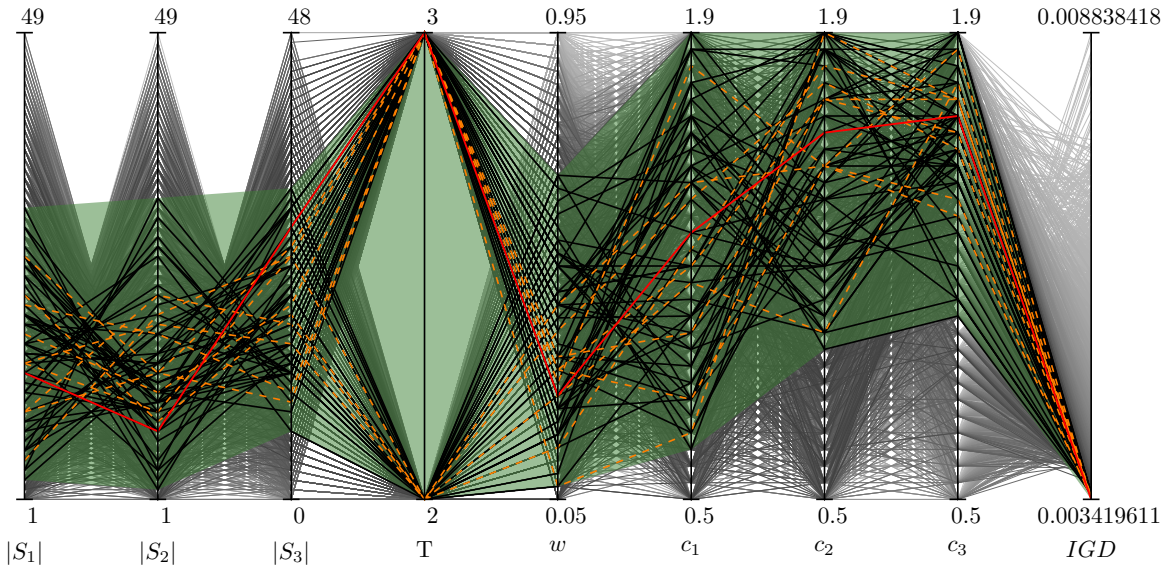


Figure 7.21: MGPSO control parameter value parallel coordinate plot for WFG7 (3 objective)

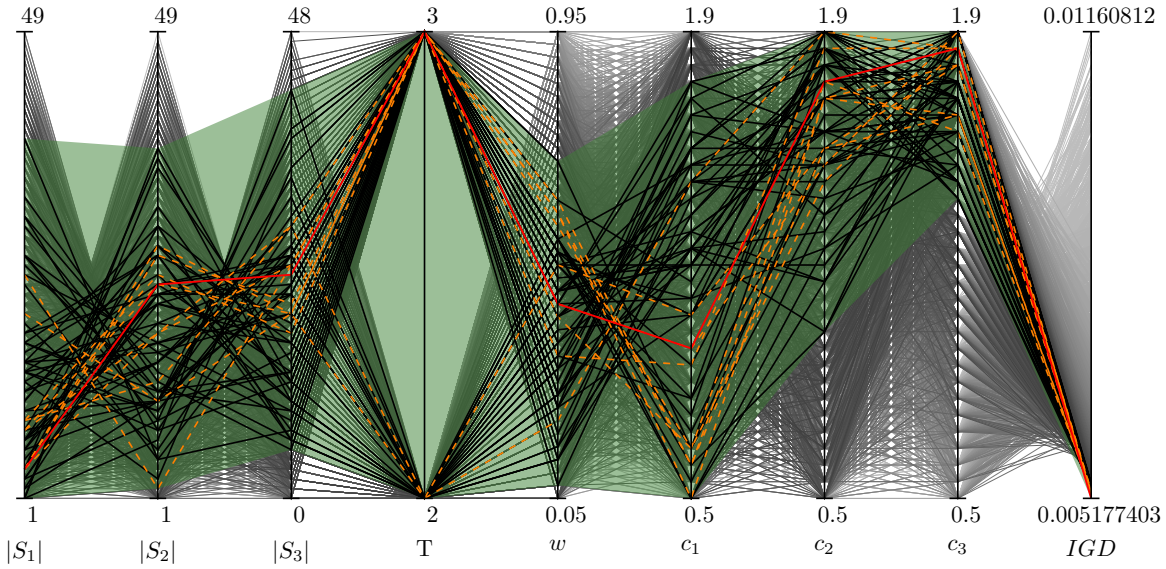


Figure 7.22: MGPSO control parameter value parallel coordinate plot for WFG8 (3 objective)

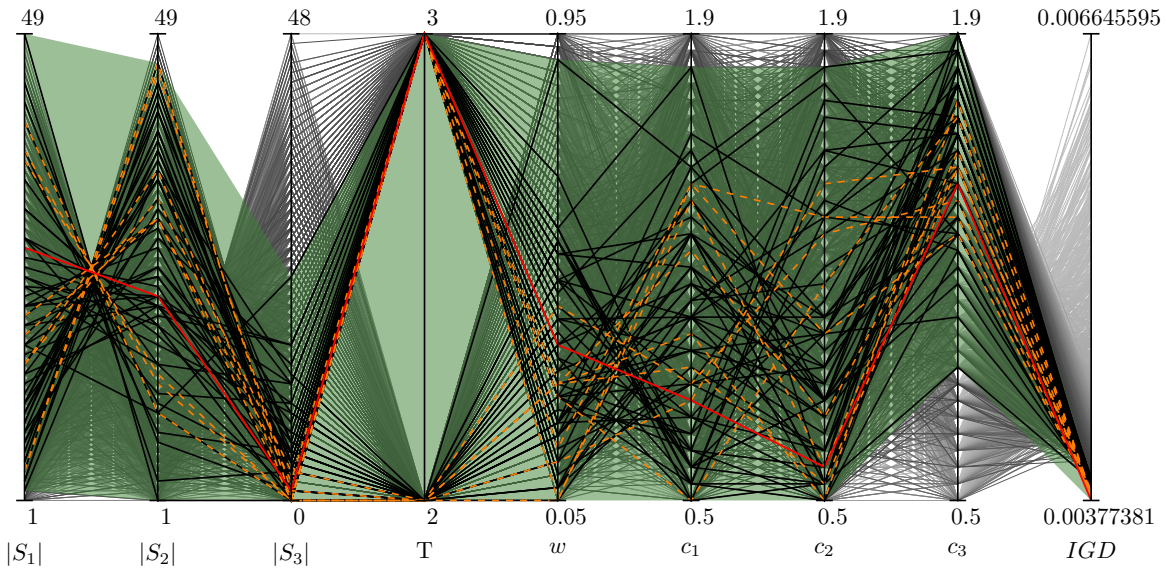


Figure 7.23: MGPSO control parameter value parallel coordinate plot for WFG9 (3 objective)

of the five ZDT problems. At least one of the acceleration coefficients, c_1, c_2, or c_3, showed sensitivity for each of the ZDT problems. c_1 showed high sensitivity preferring larger values for ZDT1, ZDT2, ZDT3, and ZDT4, while c_3 showed high sensitivity preferring larger values for ZDT1, ZDT2, and ZDT6. Larger c_3 values indicate the search process prefers a larger influence by the archive guide. The requirement for a larger influence by the archive guide is most likely a necessity arising from the fact that subswarm S_2 is much larger than subswarm S_1. Also keep in mind that for all the ZDT problems, $f_1(\mathbf{x})$ makes use of only x_1 and is thus much more trivial to solve than $f_2(\mathbf{x})$. Through the archive guide, the MGPSO is thus facilitating the search to optimize both objectives while allowing more particles to optimize $f_2(\mathbf{x})$ directly.

For the 2-objective WFG problems, no general pattern regarding subswarm sizes can be seen. For WFG1 and WFG5 smaller sizes for the S_2 subswarm and larger size for the S_1 subswarm are preferred. For WFG4 and WFG9, the opposite, namely smaller sizes for the S_1 subswarm and larger sizes for the S_2 subswarm are preferred. It should be noted that, in the case of WFG5, extreme sensitivity towards larger sizes for the S_1 subswarm can be noted. Again, little sensitivity towards the tournament pool size can be noted. When considering only the top 10 control parameter value combinations, the

inertia weight, w, showed sensitivity towards larger values for WFG2, WFG3, WFG4, WFG5, and WFG8. For WFG1 and WFG6 smaller w values are preferred. Sensitivity towards larger c_1 values can be noted for WFG1. For c_2, larger values are preferred for WFG2, WFG3, WFG7, and WFG8. For c_3, larger values are preferred for WFG1, WFG5, WFG6, and WFG7. Only WFG4 and WFG8 showed little to no sensitivity towards any of the c_1, c_2, or c_3 values. However, it can be noted that in the case of WFG4 and WFG8, control parameter combinations with small c_2 values had corresponding large c_3 values and vice versa.

For the 3-objective WFG problems, again, no general pattern regarding subswarm sizes can be seen. Smaller subswarm sizes for S_3 are preferred for WFG2, WFG4, WFG5, and WFG9. Little to no sensitivity towards the tournament pool size can be noted, although the best performing control parameter value combination typically made use of a tournament size of 3. Smaller values for the inertia weight, w, are preferred for WFG1, WFG2, WFG6, WFG7, and WFG9. For each problem, at least one of c_1, c_2, or c_3 showed sensitivity, with c_3 being the most sensitive on WFG2, WFG5, WFG6, WFG7, WFG8, and WFG9.

Table 7.1 lists the optimized MGPSO control parameter values for each of the ZDT, 2-objective WFG and 3-objective WFG problems.

It is noteworthy that, for the 2-objective WFG5 and 3-objective WFG4 and WFG5 problems, at least one subswarm size is zero. That is, there is no particle in that subswarm. Effectively, this indicates that MGPSO is able to find well-performing solutions for the objective that is represented by that subswarm without directly evaluating particle positions against that objective. It is hypothesized that indirect optimization still takes place, because the archive takes all the objectives into account, noting that the archive can contain only non-dominated solutions. The crowding distance selection approach to select archive guides results in guiding particles in the remaining subswarm to indirectly optimize the objective without a subswarm. Note that cases with a very small number of particles for one of the subswarms typically have a larger c_3 value. This correlates with the hypothesis that indirect optimization of the objective corresponding to the swarm with the fewer number of particles takes place. Also, note that the inverse is not true, large c_3 values do not indicate that there is a small number of particles

Table 7.1: Optimized MGPSO parameters

Problem	**Objectives**	$\|S_1\|$	$\|S_2\|$	$\|S_3\|$	T	w	c_1	c_2	c_3
ZDT1	2	33	17		3	0.475	1.80	1.10	1.80
ZDT2	2	8	42		3	0.075	1.60	1.35	1.90
ZDT3	2	8	42		3	0.050	1.85	1.90	1.90
ZDT4	2	5	45		2	0.175	1.85	1.35	1.85
ZDT6	2	1	49		3	0.600	1.85	1.55	1.80
WFG1	2	45	5		3	0.275	1.65	1.80	1.75
WFG2	2	24	26		2	0.750	1.15	1.70	1.05
WFG3	2	31	19		2	0.600	1.60	1.85	0.95
WFG4	2	2	48		2	0.100	0.80	1.65	1.70
WFG5	2	50	0		2	0.600	0.80	1.60	1.85
WFG6	2	19	31		2	0.525	0.65	0.60	1.65
WFG7	2	29	21		2	0.450	1.20	1.85	1.55
WFG8	2	37	13		3	0.750	1.00	1.65	1.05
WFG9	2	13	37		2	0.275	1.00	0.50	1.70
WFG1	3	37	4	9	2	0.125	1.20	1.30	1.75
WFG2	3	24	25	1	2	0.275	1.25	1.40	1.70
WFG3	3	29	10	11	2	0.525	1.65	1.75	0.75
WFG4	3	29	21	0	2	0.275	1.75	0.50	1.05
WFG5	3	2	48	0	3	0.575	0.60	1.85	1.75
WFG6	3	5	30	15	3	0.300	0.90	0.90	1.90
WFG7	3	10	22	18	2	0.425	1.45	1.50	1.40
WFG8	3	4	23	23	3	0.425	0.95	1.75	1.85
WFG9	3	4	45	1	2	0.275	1.25	0.75	1.50

in one of the subswarms. The generally large c_3 values indicate that the archive term contributes to the success of the MGPSO and validates its addition.

Also note that for three of the problems, one of the subswarms sizes is 1. Normally, this would indicate that the particle effectively performs hill-climbing. However, in the case of the MGPSO, the archive guide would still guide the particle towards other solutions through the crowding distance-based archive guide selection mechanism.

7.4 Stability Criteria

This section presents a theoretical derivation of the order-1 and order-2 stable regions for the MGPSO algorithm.

In order the derive order-1 and order-2 stable regions for the MGPSO the following general theorem of Cleghorn and Engelbrecht [11] is used:

Theorem 7.1. *The following properties hold for all PSO variants of the form:*

$$x_k(t+1) = x_k(t)\alpha + x_k(t-1)\beta + \gamma_t \tag{7.1}$$

where k indicates the vector component, α and β are well defined random variables, and (γ_t) is a sequence of well defined random variables. In the context of this work, a random variable is said to be well defined if it has an expectation and a variance.

1. *Assuming $\boldsymbol{i}_t$ converges, particle positions are order-1 stable for every initial condition if and only if $\rho(\mathbf{A}) < 1$, where*

$$\mathbf{A} = \begin{bmatrix} E[\alpha] & E[\beta] \\ 1 & 0 \end{bmatrix} \text{ and } \boldsymbol{i}_t = \begin{bmatrix} E[\gamma_t] \\ 0 \end{bmatrix} \tag{7.2}$$

2. *The particle positions are order-2 stable if $\rho(\boldsymbol{B}) < 1$ and $(\boldsymbol{j}_t)$ converges, where*

$$\boldsymbol{B} = \begin{bmatrix} E[\alpha] & E[\beta] & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & E[\alpha^2] & E[\beta^2] & 2E[\alpha\beta] \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & E[\alpha] & 0 & E[\beta] \end{bmatrix}$$

and

$$\boldsymbol{j}_t = \begin{bmatrix} E[\gamma_t] \\ 0 \\ E[\gamma_t^2] \\ 0 \\ 0 \end{bmatrix} \tag{7.3}$$

under the assumption that the limits of $(E[\gamma_t\alpha])$ and $(E[\gamma_t\beta])$ exist.

3. *Assuming that $x(t)$ is order-1 stable, then the following is a necessary condition for order-2 stability:*

$$1 - E[\alpha] - E[\beta] \neq 0 \tag{7.4}$$

$$1 - E\left[\alpha^2\right] - E\left[\beta^2\right] - \left(\frac{2E[\alpha\beta]\,E[\alpha]}{1 - E[\beta]}\right) > 0 \tag{7.5}$$

4. *The convergence of $E[\gamma_t]$ is a necessary condition for order-1 stability, and the convergence of both $E[\gamma_t]$ and $E[\gamma_t^2]$ is a necessary condition for order-2 stability.*

The MGPSO's update equation (6.1), can be put into the form of equation (7.1) by setting

$$\begin{aligned}
\alpha &= (1 + w) - c_1 r_1 - \lambda c_2 r_2 - (1 - \lambda) c_3 r_3 \\
\beta &= -w \\
\gamma_t &= c_1 r_1 y(t) + \lambda c_2 r_2 \hat{y}(t) + (1 - \lambda) c_3 r_3 \hat{a}(t)
\end{aligned}$$

In order to utilize theorem 7.1, the following modeling assumption is used:

Definition 7.1. ***Non-stagnant distribution assumption****:*
It is assumed that $\hat{\boldsymbol{y}}_i(t)$, $\boldsymbol{y}_i(t)$, and $\hat{\boldsymbol{a}}_i(t)$ are random variables sampled from a time dependent distribution, such that $\hat{\boldsymbol{y}}_i(t)$, $\boldsymbol{y}_i(t)$, and $\hat{\boldsymbol{a}}_i(t)$ have well defined expectations and variances for each t and that $\lim_{t\to\infty} E[\hat{\boldsymbol{y}}_i(t)]$, $\lim_{t\to\infty} E[\boldsymbol{y}_i(t)]$, $\lim_{t\to\infty} E[\hat{\boldsymbol{a}}_i(t)]$, $\lim_{t\to\infty} V[\hat{\boldsymbol{y}}_i(t)]$, $\lim_{t\to\infty} V[\boldsymbol{y}_i(t)]$ and $\lim_{t\to\infty} V[\hat{\boldsymbol{a}}_i(t)]$ exist.

It is clear from part 4 of theorem 7.1 that the non-stagnant distribution assumption is a necessary condition for order-1 and order-2 stability. In order to obtain the criteria for order-1 stability, part 1 of theorem 7.1 is used. Specifically, the following expectations are needed

$$\begin{aligned}
E[\alpha] &= (1 + w) - \frac{c_1}{2} - \frac{\lambda c_2}{2} - \frac{(1 - \lambda) c_3}{2} \\
E[\beta] &= -w \\
E[\gamma_t] &= \frac{1}{2}\left(c_1 E[y(t)] + \lambda c_2 E[\hat{y}(t)] + (1 - \lambda) c_3 E[\hat{a}(t)]\right).
\end{aligned}$$

Given the non-stagnant distribution assumption, it follows by the sum of convergent sequences that $E[\gamma_t]$ converges, and therefore $\mathbf{i}_t$ converges. The criteria for order-1 stability is determined by coefficients that satisfy $\rho(\mathbf{A}) < 1$. After some algebraic manipulation, the following criteria for order-1 stability is obtained:

$$|w| < 1 \text{ and } 0 < c_1 + \lambda c_2 + (1-\lambda)c_3 < 4(w+1), \tag{7.6}$$

or in the case of $c = c_1 = c_2 = c_3$,

$$|w| < 1 \text{ and } 0 < 2c < 4(w+1). \tag{7.7}$$

In order to obtain the conditions necessary for order-2 stability, part 3 of theorem 7.1 is used. The calculation of additional expected values is needed. In order to calculate $E[\alpha^2]$, α^2 is first calculated as

$$\begin{aligned}
&\alpha^2 \\
&= ((1+w) - cr_1 - \lambda cr_2 - (1-\lambda)cr_3)^2 \\
&= (1+w)^2 - c_1r_1(1+w) - \lambda c_2r_2(1+w) - (1+w)(1-\lambda)c_3r_3 \\
&- c_1r_1(1+w) + c_1^2r_1^2 + \lambda c_1c_2r_1r_2 + (1-\lambda)c_1c_3r_1r_3 \\
&- \lambda c_2r_2(1+w) + \lambda c_1c_2r_1r_2 + \lambda^2c_2^2r_2^2 + \lambda(1-\lambda)c_2c_3r_2r_3 \\
&- (1+w)(1-\lambda)c_3r_3 + (1-\lambda)c_1c_3r_1r_3 + \\
&+ \lambda(1-\lambda)c_2c_3r_2r_3 + (1-\lambda)^2c_3^2r_3^2
\end{aligned} \tag{7.8}$$

Application of the expectation operator leads to

$$\begin{aligned}
&E[\alpha^2] \\
&= (1+w)^2 - \frac{c_1}{2}(1+w) - \lambda\frac{c_2}{2}(1+w) - (1+w)(1-\lambda)\frac{c_3}{2} \\
&- \frac{c_1}{2}(1+w) + \frac{c_1^2}{3} + \lambda\frac{c_1c_2}{4} + (1-\lambda)\frac{c_1c_3}{4} \\
&- \lambda\frac{c_2}{2}(1+w) + \lambda\frac{c_1c_2}{4} + \lambda^2\frac{c_2^2}{3} + \lambda(1-\lambda)\frac{c_2c_3}{4} \\
&- (1+w)(1-\lambda)\frac{c_3}{2} + (1-\lambda)\frac{c_1c_3}{4} + \lambda(1-\lambda)\frac{c_2c_3}{4} \\
&+ (1-\lambda)^2\frac{c_3^2}{3}
\end{aligned} \tag{7.9}$$

Let $c = c_1 = c_2 = c_3$, then after some algebraic manipulation, equation (7.9) becomes

$$\begin{aligned}
&E[\alpha^2] \\
&= (1+w)^2 - c(1+w) - \lambda c(1+w) - (1+w)(1-\lambda)c \\
&+ c^2\left(\frac{1}{3} + \frac{\lambda}{2} + \frac{1-\lambda}{2} + \frac{\lambda^2}{3} + \frac{\lambda(1-\lambda)}{2} + \frac{(1-\lambda)^2}{3}\right) \\
&= (1+w)((1+w) - 2c) + \frac{c^2}{6}\left(\lambda^2 - \lambda + 7\right)
\end{aligned}$$

The following expectations are also needed:

$$\begin{aligned}
E[\alpha\beta] &= -wE[\alpha] = -w((1+w) - c) \\
E[\beta^2] &= w^2
\end{aligned} \tag{7.10}$$

In order to obtain the conditions necessary for order-2 stability, first consider the condition of equation (7.4) in part 3 of theorem 7.1:

$$\begin{aligned}
1 + E[\alpha] + E[\beta] &\neq 0 \\
c_1 + \lambda c_2 + (1-\lambda)c_3 &\neq 0
\end{aligned} \tag{7.11}$$

or if $c = c_1 = c_2 = c_3$, simply $c \neq 0$.

Now consider the condition of equation (7.5) in part 3 of theorem 7.1:

$$\begin{aligned}
1 - E[\alpha^2] - E[\beta^2] - \left(\frac{2E[\alpha\beta]E[\alpha]}{1 - E[\beta]}\right) &> 0 \\
\implies 2c - 2wc + \left(\frac{2wc^2}{(1+w)}\right) - \frac{c^2}{6}\left(\lambda^2 - \lambda + 7\right) &> 0
\end{aligned}$$

Solving the quadric form equal to 0 leads to

$$c < \frac{12(1-w^2)}{(\lambda^2 - \lambda + 7)(w+1) - 12w} \tag{7.12}$$

Merging the conditions for order-2 in equations (7.12) and (7.11) with the conditions for order-1 stability of equation (7.7) leads to the following criteria for order-1 and order-2 stability:

$$0 < c < \frac{12(1-w^2)}{(\lambda^2 - \lambda + 7)(w+1) - 12w}, \quad |w| < 1 \tag{7.13}$$

This merger is possible because the region defined by equation (7.12) is a subset of the region defined by equation (7.7). It should be noted that the conditions derived for

order-2 stability are only the necessary conditions. To verify that they are sufficient, part 2 of theorem 7.1 is used. Given the complexity of symbolically solving $\rho(\boldsymbol{B}) < 1$, an empirical approach is utilized in line with that used by Cleghorn and Engelbrecht [11]. The experimental procedure is as follows: 10^9 random combinations of the form $\{w, c, \lambda\}$ were constructed such that equations (7.11) and (7.12) were satisfied. It was then tested whether or not $\rho(\boldsymbol{B}) < 1$. It was found that in 100% of the cases, if equations (7.11) and (7.12) were satisfied, then the condition $\rho(\boldsymbol{B}) < 1$ held. This provides strong evidence that the conditions of equation (7.13) are both necessary and sufficient for order-1 and order-2 stability.

7.5 Summary

This chapter presented a discussion of the parameter optimization technique applied to find best values for the control parameters of the MGPSO. Correlation between the extremely small subswarm sizes and large c_3 values was noted. This indicated that MGPSO has the ability to indirectly optimize an objective function by increasing the weight of the contribution of the archive guide term in the MGPSO velocity update equation. Because the archive only contains non-dominated solutions as determined using all the objective functions, the effect of all the objective functions is still considered.

For the majority of the problems evaluated, a large c_3 value was found to be optimal. This again emphasizes that the addition of the MGPSO's archive term plays a large role in guiding the particles to well-performing regions of the search space.

Finally, a theoretical derivation of the order-1 and order-2 stable regions for the MGPSO algorithm was presented. Selecting control parameter values that fall within the derived stability criterion will ensure stability and avoid explosive particle velocities.

Chapter 8

Findings and Conclusions

"He who knows all the answers has not been asked all the questions."

Confucius (551 - 479 BC)

This chapter summarises the major findings of the work done for this study, summarises the conclusions drawn from the study, and provides a number of suggestions for future work that can be pursued as a result of this work.

Section 8.1 presents the summary of findings and conclusions, followed by suggestions for future work in Section 8.2.

8.1 Summary of Findings and Conclusions

This study aimed to develop a multi-objective particle swarm-based algorithm to solve multi-objective optimization (MOO) problems. An investigation into the exploration behavior of the vector evaluated particle swarm optimization (VEPSO) algorithm, and the effect that the archive management strategy has on VEPSO's performance led to the development of the multi-guided particle swarm optimization (MGPSO) algorithm.

Background was provided on particle swarm optimization (PSO), MOO, multi-objective PSO, and MOO test sets to aid in the development of a new multi-objective particle swarm-based algorithm. The test sets presented were used throughout this study to benchmark the performance of various algorithms and variations of the algorithms.

An explorative investigation into the exploration behavior of the VEPSO algorithm was presented. A novel visualization of the candidate solutions representing the current particle positions was developed. The candidate solution plots showed that the VEPSO particles continue to explore the objective space even after the well-performing region close to the Pareto-optimal front (POF) has been discovered. To aid in understanding the VEPSO algorithm's exploration behavior, new quantitative measures were developed to quantify the particle position dispersion. The dispersion measurements indicated that the VEPSO particles were not exploiting the already found well-performing regions close to the POF. A new quantitative measure to quantify particle movement diversity was developed. The particle movement diversity measure showed that the VEPSO particle's continue to move and do not converge in any meaningful way.

To further understand VEPSO's overall performance, VEPSO's archive management was investigated. The original VEPSO proposal did not specify how the archive should be managed, leaving the implementation detail to the algorithm's user. A thorough investigation into the influence that various archive management strategies has on VEPSO's overall performance, was carried out. A new hypersurface contribution archive was proposed. Early results indicated that the hypersurface archive shows much promise that merits further investigation. The archive management analysis showed that the choice of archive implementation greatly influences the POF's diversity as well as the overall performance as measured using the inverted generational distance (IGD) measure. Overall, the bounded archive with the crowding distance deletion approach outperformed the competing approaches. The diversity analysis further showed that two existing POF diversity measures, namely spacing, and distribution, could give misleading results due to a pairwise grouping problem. A new crowding distance based distribution measure, named crowding distribution, was proposed to address the pairwise grouping problem.

In order to better quantify the performance of two MOO algorithms being compared, the porcupine measure was developed. The porcupine measure is based on Knowles and Corne's work to quantify the results for Fonseca and Fleming's grand attainment surface. Knowles and Corne's measure, named the KC measure, is dependent on the generation of intersection lines to statistically compare the attainment surfaces for two or more algorithms. The accuracy of Knowles and Corne's proposal is shown to be dependent on

the shape of the POF. Two alternative intersection line generation approaches, named attainment surface shaped intersection lines (ASSIL) and weighted attainment surface shaped intersection lines (WASSIL), were proposed. Using a number of artificial POFs with known true KC measure values, both approaches were shown to lead to more accurate KC measure values. The WASSIL approach was further extended for n-dimensional attainment surface comparisons, and named the porcupine measure. A more computationally optimized version of the porcupine measure was proposed and shown to perform on-par with the more computationally intensive version.

The lessons learned from the VEPSO exploration behavior and archive management studies were combined to develop the MGPSO. A thorough comparison of MGPSO's performance against current MOO PSO algorithms, as well as the *state of the art* MOO algorithms, was presented. Both IGD and the newly introduced porcupine measure was used as performance measures. The results indicate that MGPSO is highly competitive and performs on-par, or exceeds the performance of the competing algorithms.

The MGPSO control parameter values were optimized using a parallel coordinate visualization technique. The parallel coordinate plots allow for the parameter space to be visually explored. Control parameter inter-dependencies were discussed. The optimized parameters were used to obtain all the experimental results given in this study.

With optimized parameter values, MGPSO was shown to outperform all the competing algorithms with statistical significance for five of the six ranking tables as presented in Sections 6.3.2 and 6.4.1. The results overwhelmingly support the conclusion that MGPSO is well suited to optimize multi-objective problems (MOPs) and very competitive with two or three objective functions.

Finally, a theoretical stability analysis was conducted to derive the order-1 and order-2 stable regions for the MGPSO control parameter values.

8.2 Future Work

Throughout this study, several new ideas for future research have been identified. A summary of each of these ideas is given below.

Candidate Solution Visualization Enhancements

The candidate solution plots as presented in this study greatly aided in the understanding of the VEPSO algorithm's exploration behavior. The approach, as presented in this study, is however limited to two dimensions corresponding to two objectives. The candidate solution visualization would be able to assist in researching the behavior of more algorithms if the visualization can be extended to cater for more dimensions.

One possible way to deal with additional dimensions would be to project the multi-dimensional candidate solution onto a two-dimensional plot. Multiple projections can be made with each showing candidate solutions for a different "range" of values for the dimensions that are reduced by the projection. For example, consider a 3-objective problem with candidate solutions $\mathbf{q}_k \in Q$ with $\mathbf{q}_k = (q_{k1}, q_{k2}, q_{k3})$. For each candidate solution $\mathbf{q}_k$, q_{k1} and q_{k2} can be projected directly on a 2-dimensional plot. To plot values for the remaining dimension q_{k3}, multiple plots need to be generated. For each plot, candidate solutions with q_{k3} values that fall within a specified range get plotted. The ranges can be configured as desired. For example, ranges for q_{k3} can be configured as follows: plot 1 contains all the projected candidate solutions with $q_{k3} \in [0, 0.1)$, plot 2 contains all the projected candidate solutions with $q_{k3} \in [0.1, 0.2)$, and so forth.

Another enhancement would be to use a heatmap of sorts, instead of using a plain scatterplot, where each grid block is colored based on the iteration of the last candidate solution that fell inside that block. This approach would still allow a researcher to visually see the area, in objective space, being explored in later iterations, albeit in a controllable resolution. Following this approach, the resolution of the plots could be lowered enough to interpret many more plots together.

Archives in Higher Dimensions

The experimental results presented in this study highlighted the importance that the archive implementation has on the overall performance of an algorithm. The eight bounded archive implementations presented in this study were evaluated in low-dimensional objective space. That is, only two objectives were used during the evaluation.

A thorough investigation into the scalability of the various archive implementations must be conducted to determine which archive implementation is best for many-objective

optimization problems. In addition to performance, computational complexity must also be evaluated during such a study. The hypervolume contribution archive may perform well in higher dimensions, but the improved performance may come with an unacceptable computational and complexity cost.

Hypersurface Contribution Archive Scalability

The hypersurface contribution archive proposed in this study showed great promise in the archive implementation evaluation presented. A scalability study must be conducted to evaluate the effect of more objectives on the hypersurface contribution archive's overall performance.

Simplification of the Porcupine Measure Implementation

The porcupine measure assists in the comparison of multiple POFs by adding a much-needed quantitative measurement value. As seen in the MGPSO performance analysis, the porcupine measure adds invaluable information about an algorithm's performance when the number of objectives increases beyond two, where even basic POF comparisons become too complex.

While the pseudo-code implementation of the porcupine measure is simple enough, the practical implementation is still fairly complex. Additional work needs to be done to further simplify the porcupine measure's implementation. The simpler the measure's implementation, the higher the probability that the measure will become commonly used and accepted.

Improved λ_i Initialization Approaches

The randomly initialized λ_i component forms a critical part of the MGPSO's velocity update equation. The λ_i component controls the weighted contribution trade-off between the neighborhood guide and the archive guide. While a random initialization strategy was shown to lead to good performance, a more thorough study on initialization strategies for λ_i should be conducted.

Sampling values from a non-uniform distribution, using, for example, a quasi-random low-discrepancy sequence, such as a Sobol sequence, should be investigated with the pur-

pose of optimizing the MGPSO's performance through the assignment of more *optimal* λ_i values.

Variable λ_i values that change over the number of iterations should also be investigated. Varying λ_i's value could allow for the MGPSO's behavior to switch from a more neighborhood, or local objective, focus to a more archive, or multi-objective, focus over time.

Subswarm Specific Parameter Values

The MGPSO algorithm as presented in this study separates the particles into subswarms, one per objective function. Because each subswarm focuses on optimizing a different objective, the particles in each subswarm can be fine-tuned to optimize their assigned objective function. Past studies have shown that PSO is sensitive to the problem being optimized, and optimized parameters can greatly improve overall performance. By tuning or optimizing the values for the inertia weight, w, and acceleration coefficients, c_1, c_2, and c_3, separately for each subswarm, the MGPSO's performance may be able to further improve.

Different Neighborhood Structures

The MGPSO algorithm as presented in this study makes use of the fully-connected neighborhood structure. For the basic PSO the choice of a neighborhood structure, as discussed in Section 2.1.3, can influence the convergence rate and overall performance. A thorough investigation of the effect of different neighborhood structures on the MGPSO's performance should be conducted. Attention should be paid to the MGPSO's convergence rate, the quality of the found POF, and the susceptibility to local minima.

Multi-guided Particle Swarm Optimization for Many-objective Optimization

The results presented in this study showed that MGPSO is well-suited for MOO, that is, the optimization of problems with two or three objectives. Further study is required to understand how MGPSO could be adapted to deal with many-objective optimization problems. That is, the optimization of problems with more than three objectives.

Multi-guided Particle Swarm Optimization with Constraints

Many real-world problems have constraints that must be met for solutions to be valid. In this study, the problems that were considered do not have any constraints. Constraint handling adaptations of MGPSO should be investigated.

Multi-guided Particle Swarm Optimization in Dynamic Environments

Many real-world problems are not static by nature, instead, the search space changes or morphs during the search process. When the environment changes during the search process, the environment is referred to as a dynamic environment. Applicability of MGPSO to dynamic environments should be investigated.

Multi-guided Particle Swarm Optimization Stability Criteria Validation

While the modeling assumption utilized in Section 7.4 is minimal, it is still required to empirically verify whether or not the newly derived stability criteria are truly representative of the unsimplified PSO variant under consideration. Validation can be done by utilizing a method for empirically investigating the convergence region of PSO variants as proposed by Cleghorn and Engelbrecht [8, 10].

Bibliography

[1] Johannes Bader and Eckart Zitzler. HypE: An Algorithm for Fast Hypervolume-Based Many-Objective Optimization. *Evolutionary Computation*, 19(1):45–76, 2011. doi: 10.1162/EVCO_a_00009.

[2] Thomas Bartz-Beielstein, Philipp Limbourg, Jörn Mehnen, Karlsheinz Schmitt, Konstantinos E. Parsopoulos, and Michael N. Vrahatis. Particle Swarm Optimizers for Pareto Optimization with Enhanced Archiving Techniques. *Proceedings of IEEE Congress on Evolutionary Computation*, 3:1780–1787, 2003. doi: 10.1109/CEC.2003.1299888.

[3] Thomas Bartz-Beielstein, Philipp Limbourg, Jörn Mehnen, Karlsheinz Schmitt, Konstantinos E. Parsopoulos, and Michael N. Vrahatis. Particle Swarm Optimization for Pareto Optmization with Enhanced Archiving Techniques. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 1780–1787, 2003.

[4] Thomas Beielstein, Konstantinos E. Parsopoulos, and Michael N. Vrahatis. Tuning PSO parameters through sensitivity analysis. Technical report, Department of Computer Science, University of Dortmund, 2002.

[5] Nicola Beume and Günter Rudolph. Faster S-Metric Calculation by Considering Dominated Hypervolume as Klee's Measure Problem. Technical report, Sonderforschungsbereich 531 Computational Intelligence, Universität Dortmund, Dortmund, Germany, 2006.

[6] Mauro Birattari, Thomas Stützle, Luís Paquete, and Klaus Varrentrapp. A Racing Algorithm for Configuring Metaheuristics. In *Proceedings of the Genetic and*

Evolutionary Computation Conference, pages 11–18, 2002. ISBN 1558608788. doi: 10.1.1.20.1464.

[7] Christopher W. Cleghorn. *A Generalized Theoretical Deterministic Particle Swarm Model by*. Master's dissertation, University of Pretoria, 2013.

[8] Christopher W. Cleghorn and Andries P. Engelbrecht. Particle Swarm Convergence: Standardized Analysis and Topological Influence. In *Proceedings of ANTS, Swarm Intelligence*, pages 134–145, Switzerland, 2014. Springer International Publishing.

[9] Christopher W. Cleghorn and Andries P. Engelbrecht. Particle swarm convergence: An empirical investigation. In *Proceedings of the IEEE Congress of Evolutionary Computation*, volume 1, pages 2524–2530, 2014. ISBN 9781479914883. doi: 10.1109/CEC.2014.6900439.

[10] Christopher W. Cleghorn and Andries P. Engelbrecht. Particle swarm variants: standardized convergence analysis. *Swarm Intelligence*, 9(2-3):177–203, 2015. ISSN 19353820. doi: 10.1007/s11721-015-0109-7.

[11] Christopher W. Cleghorn and Andries P. Engelbrecht. Particle swarm stability: a theoretical extension using the non-stagnate distribution assumption. *Swarm Intelligence*, https://do:1–22, 2017.

[12] Maurice Clerc. The Swarm and the Queen: Towards a Deterministic and Adaptive Particle Swarm Optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 3, pages 1951–1957, 1999. ISBN 0-7803-5536-9. doi: 10.1109/CEC.1999.785513.

[13] Maurice Clerc. Think locally, act locally: The Way of Life of Cheap-PSO, an Adaptive Particle Swarm Optimizer. 2001. URL `http://clerc.maurice.free.fr/pso/`.

[14] Maurice Clerc and James Kennedy. The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space. *IEEE Transactions on Evolutionary Computation*, 6(1):58–73, 2002.

[15] Theuns Cloete, Gary Pampara, and Andries P. Engelbrecht. CIlib: A collaborative framework for Computational Intelligence algorithms-Part II. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1764–1773, 2008.

[16] Carlos A. Coello Coello and Margarita Reyes-Sierra. A Study of the Parallelization of a Coevolutionary Multi-Objective Evolutionary Algorithm. In *Proceedings of the Third Mexican International Conference on Artificial Intelligence*, pages 688–697, 2004. ISBN 978-3-540-24694-7. doi: 10.1007/978-3-540-24694-7.

[17] Carlos A. Coello Coello and Margarita Reyes-Sierra. Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, 2(3):287–308, 2006. ISSN 09741259. doi: 10.5019/j.ijcir.2006.68.

[18] Carlos A. Coello Coello and Maximino Salazar Lechuga. MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. *Proceedings of the IEEE Congress on Evolutionary Computation*, 2:1051–1056, 2002. doi: 10.1109/CEC.2002.1004388.

[19] Carlos A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer Science+Business Media, LLC, second edi edition, 2007. ISBN 978-0-387-33254-3.

[20] Yann Collette and Patrick Siarry. *Multiobjective optimization : principles and case studies*. Springer-Verlag Berlin Heidelberg, 2003. ISBN 3540401822. doi: 10.1007/978-3-662-08883-8.

[21] David W. Corne, Nick Jerram, Joshua D. Knowles, and Martin J. Oates. PESA-II: Region-based Selection in Evolutionary Multiobjective Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2001)*, pages 283–290, 2001. ISBN 1558607749. doi: citeulike-article-id:8133801.

[22] Viviane G. Da Fonseca, Carlos M. Fonseca, and Andreia O. Hall. Inferential Performance Assessment of Stochastic Optimisers and the Attainment Function.

In *Evolutionary Multi-Criterion Optimization*, volume 1993, pages 213–225, 2001. ISBN 0302-9743 3-540-41745-1. doi: 10.1007/3-540-44719-9_15.

[23] George B. Dantzig and Mukund N. Thapa. *Linear Programming : 2 : Theory and Extensions Springer Series in Operations Research*. Springer New York, 2003. ISBN 0387948333.

[24] Kalyanmoy Deb. Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, 7(3):205–230, 1999. ISSN 1063-6560. doi: 10.1162/evco.1999.7.3.205.

[25] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, 2001. ISBN 978-0470743614. doi: 10.1109/TEVC.2002.804322.

[26] Kalyanmoy Deb, Samir Agrawal, Amrit Pratap, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. *Parallel Problem Solving from Nature PPSN VI*, pages 849–858, 2000. ISSN 10871357. doi: 10.1007/3-540-45356-3.

[27] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multi-Objective Genetic Algorithm: NSGA-II. Technical report, Indian Institute of Technology, Kanpur: Kanput Genetic Algorithms Laboratory, 2000.

[28] Kalyanmoy Deb, Ashish Anand, and Dhiraj Joshi. A Computationally Efficient Evolutionary Algorithm for Real-Parameter Optimization. *Evolutionary Computation*, 10(4):371–395, 2002. ISSN 1063-6560. doi: 10.1162/106365602760972767.

[29] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002. ISSN 1089778X. doi: 10.1109/4235.996017.

[30] Russell C. Eberhart and James Kennedy. A New Optimizer using Particle Swarm Theory. In *Proceedings of the Sixth International Symposium on Micro Machine and Human Science*, pages 39–43, 1995. doi: 10.1109/MHS.1995.494215.

[31] Russell C. Eberhart and Yuhui Shi. Comparing Inertia Weights and Constriction Factors in Particle Swarm Optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 1, pages 84–88, 2000. doi: 10.1109/CEC.2000.870279.

[32] Andries P. Engelbrecht. *Fundamentals of Computational Swarm Intelligence*, volume 8. 2005. ISBN 978-0-470-09191-3. doi: 978-0470091913.

[33] Andries P. Engelbrecht. Particle swarm optimization: Global best or local best? *Proceedings - 1st BRICS Countries Congress on Computational Intelligence, BRICS-CCI 2013*, (1):124–135, 2013. doi: 10.1109/BRICS-CCI-CBIC.2013.31.

[34] Jonathan E. Fieldsend. Multi-objective particle swarm optimisation methods. Technical report, University of Exeter, 2004.

[35] Carlos M. Fonseca and Peter J. Fleming. On the Performance Assessment and Comparison of Stochastic Multiobjective Optimisers. In *Parallel problem solving from nature - PPSN IV*, volume 1141, pages 584–593, 1995.

[36] Carlos M. Fonseca, Viviane G. Da Fonseca, and Luís Paquete. Exploring the Performance of Stochastic Multiobjective Optimisers with the Second-Order Attainment Function. In *Evolutionary Multi-Criterion Optimization*, volume 3410, pages 250–264, 2005. ISBN 3-540-24983-4. doi: 10.1007/b106458.

[37] Carlos M. Fonseca, Andreia P. Guerreiro, Manuel López-Ibáñez, and Luís Paquete. On the Computation of the Empirical Attainment Function. In *Evolutionary Multi-Criterion Optimization: 6th International Conference, EMO 2011*, pages 106–120, 2011. ISBN 9783642198922. doi: 10.1007/978-3-642-19893-9_8.

[38] Félix-Antoine Fortin and Marc Parizeau. Revisiting the NSGA-II Crowding-Distance Computation. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 623–630, 2013. ISBN 978-1-4503-1963-8. doi: 10.1145/2463372.2463456.

[39] Cornelis J. Franken. Visual Exploration of Algorithm Parameter Space. *Proceedings of IEEE Congress on Evolutionary Computation*, pages 389–398, 2009. doi: 10.1109/CEC.2009.4982973.

[40] Ivan Chaman García, Carlos A. Coello Coello, and Alfredo Arias-Montaño. MOPSOhv: A New Hypervolume-based Multi-objective Particle Swarm Optimizer. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 266–273, 2014. ISBN 9781479914883. doi: 10.1109/CEC.2014.6900540.

[41] Veysel Gazi. Stochastic Stability Analysis of the Particle Dynamics in the PSO Algorithm. In *Proceedings of the IEEE International Symposium on Intelligent Control*, pages 708–713, 2012. ISBN 9781467346009.

[42] James E. Gentle. *Random number generation and Monte Carlo methods*. Number Statistics and Computing. Springer New York, 2nd edition, 2004. ISBN 0-387-00178-6.

[43] Jean Dickinson Gibbons and Subhabrata Chakraborti. *Nonparametric statistical inference*. Chapman and Hall/CRC Press, 5th edition, 2010. ISBN 978-1420077612.

[44] Chi K. Goh and Kay C. Tan. An Investigation on Noisy Environments in Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 11(3):354–381, 2007. ISSN 1089778X. doi: 10.1109/TEVC.2006.882428.

[45] Jacomine Grobler. *Particle swarm optimization and differential evolution for multi objective multiple machine scheduling*. Master's thesis, University of Pretoria, 2009.

[46] Jacomine Grobler and Andries P. Engelbrecht. Hybridizing PSO and DE for improved vector evaluated multi-objective optimization. *Proceedings of IEEE Congress on Evolutionary Computation*, pages 1255–1262, 2009. doi: 10.1109/CEC.2009.4983089.

[47] Kyle R. Harrison, Andries P. Engelbrecht, and Beatrice M. Ombuki-Berman. A Scalability Study of Multi-Objective Particle Swarm Optimizers. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 189–197, 2013. ISBN 978-1-4799-0454-9; 978-1-4799-0453-2.

[48] Kyle R. Harrison, Beatrice M. Ombuki-Berman, and Andries P. Engelbrecht. Knowledge transfer strategies for vector evaluated particle swarm optimization. In *Proceedings of the 7th International Conference on Evolutionary Multi-Criterion Optimization*, pages 171–184, 2013. ISBN 03029743 (ISSN); 9783642371394 (ISBN). doi: 10.1007/978-3-642-37140-0.

[49] Kyle R. Harrison, Andries P. Engelbrecht, and Beatrice M. Ombuki-Berman. The sad state of self-adaptive particle swarm optimizers. *Proceedings of IEEE Congress on Evolutionary Computation*, pages 431–439, 2016. doi: 10.1109/CEC.2016.7743826.

[50] Mardé Helbig and Andries P. Engelbrecht. Archive Management for Dynamic Multi-objective Optimisation Problems using Vector Evaluated Particle Swarm Optimisation. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pages 2047–2054, 2011. ISBN 978-1-4244-7833-0. doi: 10.1109/CEC.2011.5949867.

[51] Kenneth Holladay, Keith Pickens, and Gregory Miller. The Effect of Evaluation Time Variance on Asynchronous Particle Swarm Optimization. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 161–168, 2017. ISBN 9781509046010.

[52] Simon Huband, Luigi Barone, Lyndon While, and Phil Hingston. A Scalable Multi-objective Test Problem Toolkit. In *Proceedings of the Third International Conference on Evolutionary Multi-Criterion Optimization*, pages 280–295, 2005. ISBN 3-540-24983-4. doi: 10.1007/978-3-540-31880-4.

[53] Simon Huband, Phil Hingston, Luigi Barone, and Lyndon While. A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506, 2006. ISSN 1089778X. doi: 10.1109/TEVC.2005.861417.

[54] Ching-Lai Hwang and Abu S. M. Masud. *Multiple Objective Decision Making - Methods and Applications:A State-of-the-Art Survey*, volume 164. Springer-Verlag Berlin Heidelberg, 1979. ISBN 978-3-642-45511-7.

[55] Hisao Ishibuchi, Hiroyuki Masuda, Yuki Tanigaki, and Yusuke Nojima. Difficulties in Specifying Reference Points to Calculate the Inverted Generational Distance for Many-Objective Optimization Problems. *Proceedings of the IEEE Symposium on Swarm Intelligence*, pages 170–177, 2014. doi: 10.1109/MCDM.2014.7007204.

[56] M. Jiang, Y. P. Luo, and S. Y. Yang. Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm. *Information Processing Letters*, 102(1):8–16, 2007. ISSN 00200190. doi: 10.1016/j.ipl.2006.10.005.

[57] Siwei Jiang and Zhihua Cai. Enhance the Convergence and Diversity for ϵ-MOPSO by Uniform Design and Minimum Reduce Hypervolume. In *International Conference on Artificial Intelligence and Computational Intelligence Enhance*, pages 129–133, 2009. ISBN 9780769538167. doi: 10.1109/AICI.2009.496.

[58] Yaochu Jin, Markus Olhofer, and Bernhard Sendhoff. Dynamic Weighted Aggregation for Evolutionary Multi-Objective Optimization: Why Does It Work and How? In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1042–1049, 2001.

[59] Visakan Kadirkamanathan, Kirusnapillai Selvarajah, and Peter J. Fleming. Stability analysis of the particle dynamics in particle swarm optimizer. *IEEE Transactions on Evolutionary Computation*, 10(3):245–255, 2006. ISSN 1089-778X. doi: 10.1109/TEVC.2005.857077.

[60] James Kennedy. The Particle Swarm: Social Adaptation of Knowledge. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 303–308, 1997. ISBN 0-7803-3949-5. doi: 10.1109/ICEC.1997.592326.

[61] James Kennedy. Small Worlds and Mega-Minds: Effects of Neighborhood Topology on Particle Swarm Performance. In *Proceedings of the IEEE Congress of Evolutionary Computation*, volume 3, pages 1931–1938, 1999.

[62] James Kennedy and Russell C. Eberhart. Particle Swarm Optimization. In *Pro-*

ceedings of the IEEE International Conference on Neural Networks, volume 4, pages 1942–1948, 1995. doi: 10.1109/ICNN.1995.488968.

[63] James Kennedy and Russell C. Eberhart. *Swarm Intelligence*. Morgan Kaufmann, 2001. ISBN 1558605959.

[64] James Kennedy and Rui Mendes. Population Structure and Particle Swarm Performance. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 1671–1676, 2002. doi: 10.1109/CEC.2002.1004493.

[65] Ronald Klazar and Andries P. Engelbrecht. Parameter optimization by means of statistical quality guides in F-Race. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 2547–2552, 2014. ISBN 9781479914883. doi: 10.1109/CEC.2014.6900446.

[66] Joshua D. Knowles. A summary-attainment-surface plotting method for visualizing the performance of stochastic multiobjective optimizers. In *Proceedings - 5th International Conference on Intelligent Systems Design and Applications 2005, ISDA '05*, volume 2005, pages 552–557, 2005. ISBN 0769522866. doi: 10.1109/ISDA.2005.15.

[67] Joshua D. Knowles and David W. Corne. Approximating the nondominated front using the Pareto Archived Evolution Strategy. *Evolutionary Computation*, 8(2): 149–172, 2000. ISSN 1063-6560. doi: 10.1162/106365600568167.

[68] Joshua D. Knowles and David W. Corne. Properties of an Adaptive Archiving Algorithm for Storing Nondominated Vectors. *IEEE Transactions on Evolutionary Computation*, 7(2):100–116, 2003.

[69] Thiemo Krink, Jakob S. Vesterstrøm, and Jacques Riget. Particle swarm optimisation with spatial particle extension. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 2, pages 1474–1479, 2002. ISBN 0-7803-7282-4. doi: 10.1109/CEC.2002.1004460.

[70] Saku Kukkonen and Kalyanmoy Deb. A Fast and Effective Method for Pruning of Non-Dominated Solutions in Many-Objective Problems. *Parallel Problem Solving from Nature PPSN IX*, pages 554–562, 2006. ISSN 03029743. doi: 10.1007/11844297_56.

[71] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282, 2002. ISSN 1063-6560. doi: 10.1162/106365602760234108.

[72] Marco Laumanns, Lothar Thiele, Eckart Zitzler, and Kalyanmoy Deb. Archiving With Guaranteed Convergence And Diversity In Multi-objective Optimization. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 439–447, 2002. ISBN 1-55860-878-8.

[73] Manuel López-Ibáñez, Luís Paquete, and Stützle Thomas. Exploratory Analysis of Stochastic Local Search Algorithms in Biobjective Optimization. In Thomas Bartz-Beielstein, Marco Chiarandini, Luís Paquete, and Mike Preuss, editors, *Experimental Methods for the Analysis of Optimization Algorithms*, pages 209–222. Springer Berlin Heidelberg, 2010. ISBN 978-3-642-02538-9. doi: 10.1007/978-3-642-02538-9_9.

[74] Wain Matthysen, Andries P. Engelbrecht, and Katherine M. Malan. Analysis of Stagnation Behavior of Vector Evaluated Particle Swarm Optimization. In *Proceedings of the IEEE Symposium on Swarm Intelligence*, pages 155–163, 2013. ISBN 978-1-4673-6004-3. doi: 10.1109/SIS.2013.6615173.

[75] Rui Mendes, Paulo Cortez, Miguel Rocha, and José Neves. Particle Swarms for Feedforward Neural Network Training. In *Proceedings of the International Joint Conference on Neural Networks*, volume 6, pages 1895–1899, 2002. ISBN 0780372786. doi: 10.1109/IJCNN.2002.1007808.

[76] Stanley Milgram. The Small-World Problem. *Psychology Today*, 1(1):61–67, 1967.

[77] Jacqueline Moore and Richard Chapman. Application Of Particle Swarm To Multiobjective Optimization. Technical report, 1999.

[78] Sanaz Mostaghim and Jurgen Teich. The Role of ϵ-dominance in Multi Objective Particle Swarm Optimization Methods. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 1764–1771, 2003. ISBN 0780378040.

[79] Antonio J. Nebro, Juan J. Durillo, José García-Nieto, Carlos A. Coello Coello, Francisco Luna, and Enrique Alba. SMPSO: A New PSO-based Metaheuristic for Multi-objective Optimization. In *Proceedings of the IEEE Symposium on Multi-Criteria Decision-Making*, number 2, pages 66–73, 2009. ISBN 9781424427642. doi: 10.1109/MCDM.2009.4938830.

[80] Antonio J. Nebro, Juan J. Durillo, and Matthieu Vergne. Redesigning the jMetal Multi-Objective Optimization Framework. *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, pages 1093–1100, 2015. ISSN 9781450321389. doi: 10.1145/2739482.2768462.

[81] Ender Ozcan and Chilukuri K. Mohan. Analysis of a simple particle swarm optimization system. *Intelligent Engineering Systems through Artificial Neural Networks*, pages 253–258, 1998.

[82] Ender Ozcan and Chilukuri K. Mohan. Particle swarm optimization: Surfing the waves. *Proceedings of the 1999 Congress on Evolutionary Computation, CEC 1999*, 3(2):1939–1944, 1999. ISSN 10944087. doi: 10.1109/CEC.1999.785510.

[83] Gary Pampara, Andries P. Engelbrecht, and Theuns Cloete. CIlib: A collaborative framework for Computational Intelligence algorithms-Part I. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, pages 1750–1757, 2008. doi: 10.1109/IJCNN.2008.4634035.

[84] Konstantinos E. Parsopoulos and Michael N. Vrahatis. Recent approaches to global optimization problems through Particle Swarm Optimization. *Natural Computing*, 1(2):235–306, 2002.

[85] Konstantinos E. Parsopoulos and Michael N. Vrahatis. Particle swarm optimization method in multiobjective problems. In *Proceedings of the ACM Symposium on Applied Computing*, pages 603–607, 2002. ISBN 1-58113-445-2. doi: 10.1145/508791.508907.

[86] Konstantinos E. Parsopoulos and Michael N. Vrahatis. Multi-Objective Particles Swarm Optimization Approaches. In *IGI Global*, number November 2014, pages 20–42. 2008. ISBN 9781599044989. doi: 10.4018/978-1-59904-498-9.ch002.

[87] Konstantinos E. Parsopoulos, Dimitris K. Tasoulis, Nicos G. Pavlidis, Vassilis P. Plagianakos, and Michael N. Vrahatis. Vector Evaluated Differential Evolution for Multiobjective Optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 1, pages 204–211, 2004. ISBN 0-7803-8515-2. doi: 10.1109/CEC.2004.1330858.

[88] Konstantinos E. Parsopoulos, Dimitris K. Tasoulis, and Michael N. Vrahatis. Multiobjective Optimization using Parallel Vector Evaluated Particle Swarm Optimization. In *Proceedings of the IASTED International Conference on Artificial Intelligence and Applications*, volume 2, pages 823–828, 2004.

[89] Riccardo Poli. Mean and variance of the sampling distribution of particle swarm optimizers during stagnation. *IEEE Transactions on Evolutionary Computation*, 13(4):712–721, 2009. ISSN 1089778X. doi: 10.1109/TEVC.2008.2011744.

[90] Riccardo Poli and David Broomhead. Exact Analysis of the Sampling Distribution for the Canonical Particle Swarm Optimiser and Its Convergence During Stagnation. *Proceedings of the Conference on Genetic and Evolutionary Computation*, pages 134–141, 2007. doi: 10.1145/1276958.1276977.

[91] Margarita Reyes-Sierra and Carlos A. Coello Coello. A New Multi-Objective Particle Swarm Optimizer with Improved Selection and Diversity Mechanisms. Technical report, Evolutionary Computation Group at CINVESTAV-IPN, México, 2004.

[92] Margarita Reyes-Sierra and Carlos A. Coello Coello. Improving PSO-Based Multiobjective Optimization Using Crowding, Mutation and -Dominance. In *Proceedings*

of the Third International Conference on Evolutionary Multi-Criterion Optimization, volume 3410, pages 505–519. Berlin, Heidelberg, 2005. ISBN 978-3-540-31880-4. doi: 10.1007/978-3-540-31880-4.

[93] Jeffrey L. Ringuest. *Multiobjective optimization: behavioral and computational considerations*, volume 20. Kluwer Academic Publishers, 1993. ISBN 0-7923-9236-1. doi: 10.1007/978-1-4615-3612-3.

[94] Yoshikazu Sawaragi, Hirotaka Nakayama, and Tetsuzo Tanino. *Theory of Multiobjective Optimization*, volume 176. Academic Press, Inc. Orlando, 1985. ISBN 0-12-620370-9.

[95] J. David Schaffer. Multiple objective optimization with vector evaluated genetic algorithms. In *Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100, 1985. ISBN 0-8058-0426-9.

[96] Jason R. Schott. *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization*. Msc thesis, Massachusetts Institute of Technology, Cambridge, Massachusetts, 1995.

[97] Yuhui Shi and Russell C. Eberhart. Parameter Selection in Particle Swarm Optimization. In *International conference on evolutionary programming*, pages 591–600. Springer, 1998. doi: 10.1007/BFb0040810.

[98] Yuhui Shi and Russell C. Eberhart. A modified particle swarm optimizer. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 69–73, 1998. ISBN 0-7803-4869-9. doi: 10.1109/ICEC.1998.699146.

[99] Yuhui Shi and Russell C. Eberhart. Empirical study of particle swarm optimization. In *Proceedings of the IEEE Congress of Evolutionary Computation*, volume 3, pages 1945–1950, 1999. ISBN VO - 3. doi: 10.1109/CEC.1999.785511.

[100] Kevin I. Smith, Richard M. Everson, and Jonathan E. Fieldsend. Dominance measures for multi-objective simulated annealing. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 1, pages 23–30, 2004. ISBN 0-7803-8515-2. doi: 10.1109/CEC.2004.1330833.

[101] Ponnuthurai N. Suganthan. Particle swarm optimiser with neighbourhood operator. *Proceedings of the IEEE Congress of Evolutionary Computation*, 3:1958–1962, 1999. doi: 10.1109/CEC.1999.785514.

[102] Kay C. Tan, Eik F. Khor, and Tong H. Lee. *Multiobjective Evolutionary Algorithms and Applications*. Springer-Verlag London, 2005. ISBN 1-85233-836-9. doi: 10.1007/1-84628-132-6.

[103] Santosh Tiwari, Georges M Fadel, Patrick Koch, and Kalyanmoy Deb. AMGA : An Archive-based Micro Genetic Algorithm for. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 729–736, 2008. ISBN 9781605581309.

[104] Ioan C. Trelea. The Particle Swarm Optimization Algorithm: Convergence Analysis and Parameter Selection. *Information Processing Letters*, 85(6):317–325, 2003. doi: 10.1016/S0020-0190(02)00447-7.

[105] Frans Van den Bergh. *An Analysis of Particle Swarm Optimizers*. PhD thesis, University of Pretoria, 2001.

[106] Frans Van den Bergh and Andries P. Engelbrecht. A study of particle swarm optimization particle trajectories. *Information Sciences*, 176(8):937–971, 2006. ISSN 00200255. doi: 10.1016/j.ins.2005.02.003.

[107] David A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses and New Innovations*. Ph.d. thesis, Dayton, OH: Air Force Institute of Technology, 1999.

[108] David A. Van Veldhuizen and Gary B. Lamont. Multiobjective Evolutionary Algorithm Research: A History and Analysis. Technical report, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, Wright-Patterson AFB, OH 45433-7765, 1998.

[109] David A. Van Veldhuizen and Gary B. Lamont. On Measuring Multiobjective Evolutionary Algorithm Performance. In *Proceedings of the IEEE Congress on Evolutionary Computation*, volume 1, pages 204–211, 2000. ISBN 0780363752. doi: 10.1109/CEC.2000.870296.

[110] David A. Van Veldhuizen, Jesse B. Zydallis, and Gary B. Lamont. Considerations in engineering parallel multiobjective evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 7(2):144–173, 2003. ISSN 1089778X. doi: 10.1109/TEVC.2003.810751.

[111] Jakob S. Vesterstrøm, Jacques Riget, and Thiemo Krink. Division of Labor in Particle Swarm Optimisation. In *Proceedings of IEEE Congress on Evolutionary Computation*, pages 1570–1575, 2002. ISBN 0780372824.

[112] Duncan J. Watts and Steven H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998. doi: 10.1038/30918.

[113] Keiichiro Yasuda, Azuma Ide, and Nobuhiro Iwasaki. Adaptive Particle Swarm Optimization. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, volume 2, pages 1554–1559, 2003. doi: 10.1109/ICSMC.2003.1244633.

[114] Qingfu Zhang and Hui Li. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, 11 (6):712–731, 2007. ISSN 1089-778X. doi: 10.1109/TEVC.2007.892759.

[115] Yong-Ling Zheng, Long-Hua Ma, Li-Yan Zhang, and Ji-Xin Qian. On the convergence analysis and parameter selection in particle swarm optimization. In *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics*, volume 3, pages 1802–1807, 2003. ISBN 0-7803-7865-2. doi: 10.1109/ICMLC.2003.1259789.

[116] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. Doctorate thesis, Swiss Federal Institute of Technology Zurich, 1999.

[117] Eckart Zitzler and Lothar Thiele. Multiobjective Optimization Using Evolutionary Algorithms - A Comparative Case Study. In *Parallel Problem Solving from Nature - PPSN V*, pages 292–301, 1998. ISBN 3540650784. doi: 10.1007/BFb0056872.

[118] Eckart Zitzler, Kalyanmoy Deb, and Lothar Thiele. Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2): 173–195, 2000.

[119] Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical report, Swiss Federal Institute of Technology (ETH) Zurich, 2001.

[120] Fernando A. Zotes and Matilde S. Peñas. Particle swarm optimisation of interplanetary trajectories from Earth to Jupiter and Saturn. *Engineering Applications of Artificial Intelligence*, 25(1):189–199, 2012. ISSN 09521976. doi: 10.1016/j.engappai.2011.09.005.

Appendix A

Acronyms

This appendix lists terms and acronyms used throughout this study in alphabetical order:

$\mathcal{HV}$ Hypervolume.

AMS Archive management strategy.

ASSIL Attainment Surface Shaped Intersection Lines.

DVEPSO Dynamic Vector Evaluated Particle Swarm Optimization.

EC Evolutionary Computation.

GA Genetic Algorithm.

GD Generational Distance.

IGD Inverted Generational Distance.

KTS Knowledge Transfer Strategy.

MCDM Multiple-criteria Decision-making.

MGPSO Multi-guided Particle Swarm Optimization.

MOEA/D Multi-objective Evolutionary Algorithm based on Decomposition.

MOO Multi-objective Optimization.

MOP Multi-objective Problem.

NSGA II Non-dominated Sorting Genetic Algorithm II.

OMOPSO Optimized Multi-objective Particle Swarm Optimization.

PCX Parent-centric Crossover.

PCXA Parent-centric Crossover Archive.

PESA-II Pareto Envelope-based Selection Algorithm II.

POF Pareto-optimal Front.

POS Pareto-optimal Set.

PSO Particle Swarm Optimization.

SMPSO Speed-constrained Multi-objective Particle Swarm Optimization.

SPEA2 Strength Pareto Evolutionary Algorithm 2.

VEGA Vector Evaluated Genetic Algorithm.

VEPSO Vector Evaluated Particle Swarm Optimization.

WASSIL Weighted Attainment Surface Shaped Intersection Lines.

WFG Walking Fish Group.

ZDT Zitzler-Deb-Thiele.

Appendix B

Symbols

This appendix lists symbols used throughout this study.

$\mathcal{A}$	set of non-dominated archive solution
$\hat{\mathbf{a}}$	represent a non-dominated solution in the archive
$\hat{\mathbf{a}}_i(t)$	represent the archive guide of particle i at iteration t
C	crowding distribution
c_1, c_2, c_3	acceleration constants
$\ddot{c}$	adaptive grid selection pressure
$\check{c}$	adaptive grid, grid coordinate
D	distribution measure by Goh and Tan [44]
$\bar{D}$	maximum spread measure by Zitzler [116]
D_l	dispersion length
D_σ	dispersion angle standard deviation
D_θ	dispersion angle average
$\mathbf{e}_m$	standard basis vector for the m'th dimension
$\check{e}_m$	length of the adaptive grid's hypercube edge for objective m
$\mathcal{F}$	feasible space
f	objective function
f_m	minimization objective function for objective m (e.g. f_1)

f'_m	maximization objective function for objective m (e.g. f'_1)
f^*_m	optimal objective value for objective m
f^{**}_m	worst objective value for objective m of a Pareto-optimal solution
$f_{del,u}$	archive management strategy's deletion fitness
GD	generational distance by Van Veldhuizen and Lamont [108, 109]
$\|H\|$	number of solutions in the same grid coordinate
h	algorithm index
i	particle index
j	generic vector component index
k	Pareto-optimal solution index
m	objective function index
n_h	number of algorithms
n_k	number of Pareto-optimal solutions
n_l	number of intersection lines
n_m	number of sub-objectives
n_n	nearest neighbor archive management strategy neighborhood size
n_o	number of objective vectors (also known as candidate solutions)
n_q	number of Pareto-optimal solutions
n_r	ring neighborhood structure size
n_s	number of particles (also know as swarm size)
n_t	number of iterations used for movement diversity calculation
n_x	particle dimension
$\mathcal{O}$	objective space
$\mathcal{P}$	Pareto-optimal set
Q	set of non-dominated solutions that make up the Pareto-optimal front (POF)
Q_{true}	set of non-dominated solutions that make up the true POF
$\mathbf{q}_k$	solution k (also referred to as objective vector k)
$\mathbf{q}_l$	solution l (also referred to as objective vector l)

$\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3$	uniform random vectors with components between 0 and 1
$\mathcal{S}$	search space
$\bar{S}$	spacing measure by Schott [96]
S_m	sub-swarm for sub-objective m
T	competition pool size
t	current iteration
$\mathcal{V}_k$	hypercube formed between solution $\mathbf{q}_k$ and a reference point $\boldsymbol{\mathcal{W}}$
$\mathbf{v}_{min}$	minimum particle velocity
$\mathbf{v}_{max}$	maximum particle velocity
$\boldsymbol{\mathcal{W}}$	reference point for hypercube calculation
w	inertia weight
$\mathbf{x}_i(t)$	position of particle i at iteration t (also referred to as a decision vector)
$x_{ij}(t)$	j'th component of the objective vector i at iteration t
$\mathbf{y}_i$	represent the cognitive guide of particle i
$\hat{\mathbf{y}}_i$	represent the social guide of particle i
$\mathbf{z}^*$	ideal objective vector
$\mathbf{z}^{**}$	utopian objective vector
$\mathbf{z}^{\mathbf{nad}}$	nadir objective vector
Δ	spread measure
Δt	iteration iterator used for movement diversity calculation
ε_m	a small positive constant for objective m
$\boldsymbol{\varphi}_1, \boldsymbol{\varphi}_2$	uniform random vectors with components between 0 and 2
λ_i	exploitation tradeoff coefficient for particle i
σ	standard deviation
ϕ_1, ϕ_2	acceleration constants when using a constriction factor
χ	constriction factor
$\prec$	Pareto-dominance operator
$\preceq$	weak Pareto-dominance operator
$\prec_\varepsilon$	ε-dominance operator

Appendix C

Parameter Configurations

This appendix provides a list of the parameter configurations for the additional algorithms used throughout this study. The experimental work for the below listed algorithms were conducted using the jMetal framework [80] with the recommended well-performing parameter configurations found throughout the literature [21, 29, 79, 92, 114, 119]. Tables C.1 through C.6 list the parameter configurations.

Table C.1: MOEA/D parameters

Parameter	**Value**
Crossover Operator	Differential Evolution Crossover
CR	1.0
F	0.5
Mutation Probability	$\frac{1}{\text{number of variables}}$
Mutation Distribution Index	20.0
Mutation Operator	Polynomial Mutation
Selection Operator	Binary Tournament Selection with Ranking using Crowding Distance
Neighborhood Selection Probability	0.9
Neighborhood Size	20
Function Type	Tchebycheff
Maximum Number of Replaced Solutions	2
Population Size	50
Result Population Size	50

Table C.2: NSGA II parameters

Parameter	**Value**
Crossover Probability	0.9
Crossover Distribution Index	20.0
Crossover Operator	Simulated Binary Crossover (SBX)
Mutation Probability	$\frac{1}{\text{number of variables}}$
Mutation Distribution Index	20.0
Mutation Operator	Polynomial Mutation
Selection Operator	Binary Tournament Selection with Ranking using Crowding Distance
Population Size	50

Table C.3: OMOPSO parameters

Parameter	**Value**
Mutation Probability	$\frac{1}{\text{number of variables}}$
Uniform Mutation Perturbation	0.5
Nonuniform Mutation Perturbation	0.5
η	Problem dependent (tuned according to number of solutions desired)
Archive Type	Bounded with a deletion approach
Archive Size	50
Archive Deletion Approach	Crowding Distance based
Swarm Size	50

Table C.4: PESA-II parameters

Parameter	**Value**
Crossover Probability	0.9
Crossover Distribution Index	20.0
Crossover Operator	Simulated Binary Crossover (SBX)
Mutation Probability	$\frac{1}{\text{number of variables}}$
Mutation Distribution Index	20.0
Mutation Operator	Polynomial Mutation
Bisections	5
Archive Size	50
Population Size	50

Table C.5: SMPSO parameters

Parameter	**Value**
Mutation Probability	$\frac{1}{\text{number of variables}}$
Mutation Distribution Index	20.0
Mutation Operator	Polynomial Mutation
Archive Type	Bounded with a deletion approach
Archive Size	50
Archive Deletion Approach	Crowding Distance based
Swarm Size	50

Table C.6: SPEA2 parameters

Parameter	**Value**
Crossover Probability	0.9
Crossover Distribution Index	20.0
Crossover Operator	Simulated Binary Crossover (SBX)
Mutation Probability	$\frac{1}{\text{number of variables}}$
Mutation Distribution Index	20.0
Mutation Operator	Polynomial Mutation
Selection Operator	Binary Tournament Selection with Ranking using Crowding Distance
Population Size	50

Appendix D

Derived Publications

This appendix lists publications derived from the work presented in this study in order of publication date:

- C. Scheepers and A. P. Engelbrecht, Vector Evaluated Particle Swarm Optimization Exploration Behavior Part I: Explorative Analysis. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2016.
- C. Scheepers and A. P. Engelbrecht, Vector Evaluated Particle Swarm Optimization Exploration Behavior Part II: Quantitative Analysis. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2016.
- C. Scheepers and A. P. Engelbrecht, Vector Evaluated Particle Swarm Optimization Archive Management: Pareto Optimal Front Diversity Sensitivity Analysis. In *Proceedings of the IEEE Symposium Series on Computational Intelligence*, 2016.
- C. Scheepers and A. P. Engelbrecht, Misleading Pareto Optimal Front Diversity Metrics : Spacing and Distribution. In *Proceedings of the IEEE Symposium Series on Computational Intelligence*, 2016.
- C. Scheepers and A. P. Engelbrecht, Vector Evaluated Particle Swarm Optimization: The Archive's Influence on Performance. In *Proceedings of the IEEE Congress on Evolutionary Computation*, 2017.

- C. Scheepers and A. P. Engelbrecht, Quantified Pareto-optimal Front Comparisons using Attainment Surfaces. In *Proceedings of the IEEE Symposium Series on Computational Intelligence*, 2017.

- C. Scheepers and A. P. Engelbrecht, Comparing Performance of Multi-objective Algorithms using the Porcupine Measure. *Currently under review*, 2017.

- C. Scheepers and A. P. Engelbrecht, Multi-guided Particle Swarm Optimization: A Multi-swarm Multi-objective Particle Swarm Optimizer. *Currently under review*, 2017.