University of Exeter

Department of Computer Science

Many Objective Particle Swarm Optimisation: An Investigation into Strengthening Convergence by Controlling Dominance Area

Matthaus Martin Woolard

September 2014

Supervised by Dr Jonathan Fieldsend

Submitted by Matthaus Martin Woolard, to the University of Exeter as a thesis for the degree of Master by Research in Computer Science , September 2014.

I certify that all material in this thesis which is not my own work has been identified and that no material has previously been submitted and approved for the award of a degree by this or any other University.

(signature) . . . . .  . . . . . .

Matthaus Martin Woolard

Declaration

Chapters 3 & 4 published as:

> Woolard, M. M. and Fieldsend, J. E. (2013). *On the Effect of Selection and Archiving Operators in Many-Objective Particle Swarm Optimisation.* In Proceedings of the 14th Annual Conference on Genetic and Evolutionary Computation, pages 129 – 136 . ACM

Matthaus Martin Woolard

I would like to thank my supervisor Dr Jonathan Fieldsend and the entire Nature Inspired Computation Group at the University of Exeter

Abstract

The application of single and multi-objective particle swarm optimisation (PSO) is widespread, however in *many*-objective optimisation (problems with four or more competing objectives) traditional PSO has been less well examined. Recent progress on many-objective evolutionary optimisers has lead to the adoption of a variety of non-Pareto quality measures, it is therefore of interest to see how well PSO copes in this domain, and how non-Pareto quality measures perform when integrated into PSO. Here we review the current state of the art in multi- and many-objective PSO optimisation. We compare and contract the performance of canonical PSO, using a wide range of many-objective quality measures, on a number of different parametrised test functions for up to 30 competing objectives. We examine quality measures as *selection* operators for guides when truncated non-dominated archives of guides are maintained, and *maintenance* operators, for choosing which solutions should be maintained as guides from one generation to the next. We investigate in detail two Pareto strengthening methods, Controlling Dominance Area of Solutions (CDAS) and Self-Controlling Dominance Area of Solutions (S-CDAS). We find that CDAS and S-CDAS perform exceptionally well as a quality measures to determine archive membership for global and local guides. However, for convergence only at the cost of diversity and spread across the optimal front, single objective canonical PSO run using a linear sum of objectives, has the best performance overall.

Nomenclature

$\mathbf{x}, \mathbf{u}, \mathbf{v}$	Vectors in parameter/search/domain/design space
$f_i(\mathbf{x})$	i^{th} Objective function mapping $\mathbf{x}$ to the i^{th} objective
$\mathbf{f}(\mathbf{x})$	Objective functions mapping $\mathbf{x}$ to objective space
$\prec$	Pareto dominance (for first introduction see section 2.5.1).
$\mathbf{w}$	Inertia of a particle.
c_1	Constraints on the velocity contribution from personal best. (Cognitive learning factor)
c_2	Constraints on the velocity contribution from global/local best. (Social learning factor)
χ	Overall constraint of shift in position.
$\mathbf{p}$	Personal guide.
$\mathbf{g}$	Global guide.
$\mathcal{P}$	The set of optimal Pareto solutions.
$\mathcal{F}$	The image of $\mathcal{P}$ through $\mathbf{f}$.
$\mathbf{s}$	Vector of mapping coefficients for CDAS.
d, n, p	Number of search parameters.
m	Number of objectives.

Glossary of terms

PSO Particle swarm optimisation (see section 2.4).

MOPSO Multi/Many objective particles swarm optimisation (see section 2.6.8).

EA Evolutionary algorithm.

GA Genetic algorithm.

FR Favour relation (a ranking method see section 2.6.1).

AR Average ranking (a ranking method see section 2.6.4).

SR Sum of average rank (a ranking method see section 2.6.5).

RR Uniform random

KO k-optimality (a ranking method see section 2.6.2).

CDAS Controlling dominance area of solutions (a geometric remapping of the objective space see section 2.6.6).

CDAS-R Controlling dominance area of solutions as a ranking method (a ranking method see section 3.1.1).

CD Crowding distance (a class of ranking method see section 2.6.3).

S-CDAS Self controlling dominance area of solutions (an adaptive geometric remapping of the objective space see section 2.6.7).

GD Generational distance (a measure of quality see section 2.8.1)

IGD Inverse generational distance (a measure of quality see section 2.8.2)

DTLZ Set of Scalable Multi-Objective Optimization Test Problems by K. Deb, L. Thiele, M. Laumanns and E. Zitzler (see section 2.7.1).

WFG Set of Scalable Multi-Objective Optimization Test Problems (see section 2.7.2).

Contents

1. Introduction

As humanity strives to develop ever more complicated systems the need for assisted design and analysis grows. The use of computational resources to aid these processes has grown in response to this need. With academia working in parallel with applied industry various optimisation techniques have been developed over the past years. Here we will cover a small selection of these focusing upon the nature inspired particle swarm optimisation heuristic and how this method has been applied to both single and multi/many-objective tasks.

The adaptation of models and processes observed in nature and applied to the computational domain aims to allow such a model to search/optimise/innovate over many differing problems without needing explicit problem knowledge. Providing what is known as a 'black box' system, earlier versions of these are now regularly used throughout industry to solve complex scheduling, design and parameter tuning tasks. The existing applications of these systems span the entire industrial domain with financial organisations optimising trading algorithms to cutting-edge medical research searching human genome for potential cures to disease (Brabazon and O'Neill, 2006; Greene et al., 2008).

Since the first of these algorithms, inspired by evolution itself, the field has expanded into a diverse range of approaches, spanning animal behaviour to molecular and chemical interactions. Within this study we will investigate the applications of a particular algorithm known as particle swarm optimisation, this draws from the behaviour of flocking birds and fish, to provide a searching behaviour.

The particle swarm optimisation (PSO) heuristic is a population-based approach where the emergent intelligence of the system comes about through the communication strategies used between the individual 'dumb' members of the swarm. The traditional PSO heuristic, introduced in Kennedy and Eberhart (1995), considers the swarm/population to be fully connected, where each member/particle is in communication with every other member/particle. In addition to this, each individual member/particle holds a personal memory of the best place it has been, then by combining its personal preference with the

best one found throughout the swarm/population the individual is able to move towards the combined location of these two good solutions. However, each member/particle has an internal inertia based upon its previous last movement - as well as guides (solutions that have been chosen from personal and swam memory) from previous iterations thereby continue to influence the search. This has two effects: firstly to allow each particle to cover more ground (promoting a diverse search), the swarm swirls in on an optimal solution instead of following a direct path. Secondly to smooth out the effects of selecting guides from different sectors of the search space from iteration to iteration without this the swarm could resemble random (Brownian) motion when two (personal and global) guides are selected at opposing sides of the parameter space.

What follows is a close examination of how the particle swarm optimisation heuristic can be used to solve problems with more than one competing quality measure. Preceding the main body of the study there will first be an analysis of the current literature surrounding PSO and nature inspired optimisation in general. This will formally introduce the topic, discuss the current state of the art in multi/many- objective optimisation and go into the details of the algorithms/methods used later.

Following the literature review chapter 3 introduces a novel ranking method: Controlling Dominance Area of Solutions - Ranking (CDAS-R). This is applied as a guide selection method and compared, on four test problems from 2 to 20 objectives, to 6 other commonly used selection methods. Chapter 4 follows on with an investigation into the maintenance of personal and global solution archives. Here the same 7 selection methods from chapter 3 are adapted for archive truncation and compared alongside a Pareto strengthening approach, Controlling Dominance Area of Solutions (CDAS), on the same set of problems as in chapter 3.

The strong performance of CDAS in chapter 4 leads to an analysis of the front regions it prefers. Here an additional adaptation to CDAS, self-CDAS (S-CDAS) is compared with CDAS on multiple different front topologies. An addition to CDAS - mapped CDAS, is proposed. Then in chapter 6 an in-depth experimental analysis on 14 test problems from 2 to 30 objectives gives a comparison between CDAS, S-CDAS and the proposed mapped CDAS. Finally in chapter 7 the findings from each chapter are collated and future work is discussed.

2. Literature Review

Here we investigate how optimisation methods, in particular nature inspired algorithms, have been applied and documented within the literature. From the general overview of numerical optimisation, and the broader nature inspired field, we will investigate the population-based nature inspired method known as particle swarm optimisation. Optimisation with multiple criteria is also introduced, and the appropriate adaptations that need to be made to the particle swarm algorithm such that it can perform appropriately on these multiple criteria functions are discussed.

2.1. Optimisation of non-trivial systems

There are many problems that can be represented in the pure mathematical form, ideally a form that can be indefinitely integrated and differentiated, however even these problems can become too unwieldy to analytically find optimal solutions. When such a system is analytically solvable we may still find ourselves struggling to find the correct optimal solutions due to additional boundary conditions which may be in themselves difficult to analytically incorporate into the system.

Moving from the theoretical mathematical domain into applied engineering, real-world applications pose many additional problems. What may initially appear to be a simple system mathematically, a roller-coaster ride for example, can become immensely complicated once one must incorporate the vast number of not fully understood external factors, for example weathering or air resistance, making the modified mathematical model non-analytically solvable. This gives rise to the requirements of numerical methods for solving these complex and sometimes uncertain problems.

2.2. Numerical Optimisation

To solve the complexities that can arrive in non-trivial systems, as described above in section 2.1, mathematicians have used algebraic methodologies to develop numerical pro-

cesses which allow for the discovery of approximate solutions. All of these methods, with the exception of random walks, maintain at least some memory of previous solutions and commonly a means of determining their quality in comparison to a new solution, for example interval bisection will maintain the solutions bounding the current search interval bisection (Carnahan et al., 1990).

As problems became more involved, with less knowledge of the solution landscape, new methodologies are needed. Such systems need to be able to search a space of potentially multiple parameters to find an optimal solution. With little or no knowledge of how these parameters may interact and connect it becomes increasingly difficult to derive a deterministic algorithm to optimise these problems.

The first heuristics widely applied to these problems are known as hill climbers, similar to interval bisection but with the addition of random seeding. These systems aiming to traverse the space defined by the quality of any given solution, try to climb to optimal solutions. Hill climbers are still used today in many industrial applications where the objective space is smooth with few or no local optima[1]. (Schwefel, 1981; Xi et al., 2004)

2.3. Nature Inspired Computation

For a long time it has been known that systems in nature effectively solve problems in a far better way than we are able, e.g ants searching for the shortest path between nest and food sources - on longer time-scales the evolutionary process has produced a vast variety of solutions to the problem of surviving on this planet. The field of nature inspired computation describes algorithms which seek to emulate some attributes of these systems. Nature inspired computation does not describe a full simulation of the processes observed in nature but rather a simplified model which we hope provides an adequate approximation of the behaviour within our computational limits.

From the fundamental processes of life on our planet, natural selection and evolution, to the finer granularity of the resultant connected systems, neural pathways and cognitive learning, nature inspired computing is a massive field (Marrow, 2000). In particular here we are interested in those systems which can be applied for optimisation, the most obvious being natural selection, and how these processes are manifest in algorithmic computational terms. Among the first of these systems to be applied was the evolutionary process. Such algorithms are known as Evolutionary Algorithms (EA) (Bäck and Schwefel, 1993).

[1]Local optima: solutions that with respect to their local region appear the best but globally are not.

Evolutionary Algorithms, a subset of evolutionary computing, applying the principles of crossover, mutation and selection as seen in nature to arbitrary genomes which could represent any number of parameters, are used to solve many problems. By considering a population which passes through selection, crossover and mutation at each time step EAs are able to search across a large proportion of the space simultaneously. With crossover operations between two 'fit' parents generating a new solution with some genetic material from each parent, potentially this new solution could be in a radically different location to its two parents. Mutation provides small and local variations similar the hill climber heuristic. Still one of the most popular areas of nature inspired computing, EAs are used throughout academia and industry today (Bäck and Schwefel, 1993; Zitzler et al., 2002; Zitzler and Thiele, 1999; Zitzler et al., 2000; Fonseca and Fleming, 1995).

While evolutionary techniques are still popular, they are now in competition with a new suite of nature inspired algorithms, many of which were originally developed for particular problem types, and have since been adapted to a larger spectrum of problems bringing them into competition with EAs. Among these are collection of algorithms which attempt to simulate the behaviour of living organisms during their day-to-day activities. Particle swarm optimisation, which is inspired by the flocking behaviours of birds and fish is the subject of our further investigation(Kennedy and Eberhart, 1995; Yang, 2010).

2.4. Particle Swarm Optimization

Since its inception by Kennedy and Eberhart (1995) the particle swarm optimisation (PSO) heuristic has gained rapid popularity as a technique to facilitate single objective optimisation. Like the standard evolutionary algorithm (EA) methods of genetic algorithms (GAs) and evolution strategies (ESs), PSO was inspired by nature, but instead of evolution it was the flocking and swarming behaviour of birds and insects that motivated its development.

A population (swarm) of individual solutions is maintained in PSO, whose representation is typically a vector of floating point decision parameters, which are used in a solution's (particle's) evaluation. During the optimisation process of PSO (following initialisation), members of this population are flown (have their parameters adjusted) according to their previous 'flying experience'. This flying experience is both in terms of the particle as an individual, and as a member of a wider group (the entire swarm, or a subset of it). The general PSO model implements this by adjusting an individual's decision parameters to

make them 'closer' to the decision parameters of two other solutions; a neighbourhood guide (which may be global or local), and the best evaluated position found previously by that individual. A particle's position also includes some temporal adjustment via a *velocity* vector, which tracks the movement the particle made in the previous iteration of the optimiser, and uses this to adjust the particle's position in the current iteration.

Since its first inception there have been many modifications, adjustments, and improvements proposed to the PSO heuristic: controlling which particles in the swarm communicate with each other, adding additional swarms which communicate occasionally between each other but are otherwise independent, and incorporating hierarchical communication structure upon the particles (Janson and Middendorf, 2005). As the PSO heuristic has been applied to multiple criteria problems, in particular those where the criteria (or fitness function/objectives) compete, modifications have been made to increase the rate that PSO finds good solutions, the rate of convergence.

2.4.1. The PSO heuristic

The PSO heuristic was first proposed for the optimisation of continuous non-linear functions (Kennedy and Eberhart, 1995). A fixed population of solutions is used, where each solution (or particle) is represented by a point in n-dimensional space. The kth particle is commonly represented as $\mathbf{x}_k = (x_{k,1}, x_{k,2}, \dots x_{k,n})$, with its performance evaluated on a given problem and stored. Each particle maintains knowledge of its best previous evaluated position, represented as $\mathbf{p}_k$, and also has knowledge of the single best solution found so far in some defined neighbourhood, $\mathbf{g}_k$, often this is a global neighbourhood (all particles are considered), however other neighbourhood definitions are also popular. The rate of position change of a particle then depends upon its previous personal best position, its neighbourhood best, and its previous velocity. For particle k this velocity is $\mathbf{v}_k = (v_{k,1}, \dots, v_{k,n})$. The general algorithm for the adjustment of these velocities is:

$$v_{k,j} := w v_{k,j} + c_1 r_1 (p_{k,j} - x_{k,j}) + c_2 r_2 (g_{k,j} - x_{k,j}), \tag{2.1}$$

and the position is updated as:

$$x_{k,j} := x_{k,j} + \chi v_{k,j}, \quad j = 1, \dots, n, \tag{2.2}$$

where w, c_1, c_2, $\chi \geq 0$. w is the inertia of a particle, c_1 and c_2 are constraints on the velocity toward local best and neighbourhood best - referred to as the *cognitive* and *social* learning factors respectively, χ is a constraint on the overall shift in position, and $r_1, r_2 \sim \mathcal{U}(0,1)$.

As discussed by Fieldsend (2004), in this classical form of PSO each particle $\mathbf{x}_k$ is flown toward $\mathbf{p}_k$, $\mathbf{g}_k$ and $\mathbf{v}_k$. This, in effect, means that a hypercuboid is generated in solution/particle space, the bounds of which are the sum of the distances from $\mathbf{x}_k$ to the other three guides (weighted by the appropriate multiplier constants from (2.1) and (2.2)). Formally, the length of the jth dimension of the containing hypercuboid of $\mathbf{x}_k$ is:

$$l_j = \chi(wv_{k,j} + c_1(p_{k,j} - x_{k,j}) + c_2(g_{k,j} - x_{k,j})). \tag{2.3}$$

A particle $\mathbf{x}_k$ can therefore effectively move to any point within this hypercuboid (determined by the draws of $\mathbf{r}_1$ and $\mathbf{r}_2$), but not *outside* of it. Note that depending on the values of χ, c_1 and c_2, it is possible for one or more of $\mathbf{v}_k$, $\mathbf{p}_k$ and $\mathbf{g}_k$ to lie outside this bounded region. This restriction on a particle's movement means that local optima within this bound may be found, but any global optima outside will not be found on this iteration by $\mathbf{x}_k$, and may never be attainable. When there is a single global best for the entire swarm, then $\mathbf{g}_k$ is the same for all k.

2.4.2. Variations of PSO

Within the general PSO heuristic there are many aspects which can be varied and tweaked depending on the problem specific attributes. One addition to the standard heuristic is, for example, the introduction of turbulence, known as craziness in the original paper (Kennedy and Eberhart, 1995), which applies a small perturbation to the location of the particle before or after movement (Fieldsend and Singh, 2002a). Other variations include altering the concept of global memory replacing it with a network where each particle only communicates with the adjacent particles on this communication graph (Peer et al., 2003).

Turbulence

As proposed in the original paper (Kennedy and Eberhart, 1995), craziness now more commonly known as turbulence is PSO's implementation of the mutation operation seen in EAs. Typically the turbulence is applied using values drawn from the normal distribution

centred about the particles location (Fieldsend and Singh, 2002b). This however is not the only way in which turbulence could be applied to PSO, for example adaptive turbulence operations when the turbulence can be computed as a function of the velocity (Liu et al., 2007).

Turbulence-based PSO variations have now become popular as they provide the additional level of local search and extra protection against early convergence, where the entire population collapses on one local region (Abraham and Liu, 2009).

Communication within the swarm

The originally proposed PSO heuristic allowed all particles to communicate with each other equally however this can lead to the entire population collapsing down to one location prematurely, as all the particles may see that one location as the best and if on their journey to this location they do not find any better locations all the particles may end up sitting in the same local area. To overcome this a variety of modifications to how particles communicate have been proposed, these typically aim to increase diversity and discourage premature convergence. The most popular are ring/wheel, star and tree topologies. (Li, 2010; Krink et al., 2002; Settles, 2005)

Ring topology orders the particles such that each particle communicates with its two neighbours, forming a ring of particles. Thus for any particle to receive information about a better location its immediate neighbours must have found that location. Altering this such that all particles communicate only with a shared parent particle, known as star topology, in middle ground can be found between ring extremes and the traditional PSO. Taking star topology further the development of hierarchical structures within the communication system such that children only receive communication from their parents forming a directed tree of communication has been shown to perform well on some problems. (Janson and Middendorf, 2005; Miyagawa and Saito, 2009)

Within this study we are limiting scope to the investigation of the original PSO algorithm, a fully connected topology.

Velocity Limiting

It is been found to be beneficial in many applications of PSO to limit the maximum velocity of any given particle. This is independent of χ which alters the influence of movement. By constricting the maximum velocity, either absolutely or through some

saturating function, unwanted rapid movement of particles can be avoided which increases the chance of searching more of the space since an unrestricted movement may jump over interesting sectors especially early on in the search process (Wilke et al., 2007).

Handling boundary conditions

With the ability of a particle to move outside of the boundaries of the problem it is important to consider how such common occurrences are handled. There are various methods, most of which are independent of PSO and applied across many evolutionary such processes, each of which influences search in its own unique way.

Re-sampling, while not always possible, is considered to provide the least additional influence to the search. This is where the movement calculation is recomputed with a new $\mathbf{r}_1$ and $\mathbf{r}_2$ when a particle is projected to a location outside the boundary. This process is however only possible if there exist values for which the solution will not end up out of bounds, this is due to the momentum of the particle. Truncation is another popular approach, this is where a particle that is found to be out of bounds is corrected and placed at its intersection point on the boundary. This however does increase the probability of particles finding themselves on the edge of a given domain. Rebound is where the particle is bounced back in the direction it came from proportional to the amount it overstepped the boundary (Xu and Rahmat-Samii, 2007).

2.5. Many- and Multi Objective Optimisation

In a large number of design applications there are multiple competing quantitative measures that define the quality of a solution. For instance, in designing the ubiquitous widget, a company may wish to minimise its production cost, but also maximise/minimise one or more widget performance properties. These objectives cannot be typically met by a single solution, so, by adjusting the various design parameters, the firm may seek to discover what possible combinations of these objectives are available, given a set of constraints (for instance legal requirements and size limits of the product).

2.5.1. Pareto Dominance

With more than one objective it is no longer possible to clearly identify an absolute ordering of the quality of solutions. While it is impossible to always compute a complete ordering it is possible to maintain a transitive ordering (i.e. if $a > b$ and $b > c$ then

$a > c$) this makes the order of computation irrelevant. The curve (for two objectives) or surface (more than two objectives) that describes the optimal trade-off possibilities between objectives is known as the Pareto front, $\mathcal{F}$. A feasible solution lying on the Pareto front cannot improve any objective without degrading at least one of the others, and, given the constraints of the model, no solutions exist beyond the Pareto front. The goal, therefore, of *multi-objective* algorithms (MOAs) is to locate the Pareto front of these non-dominated solutions. More formally, a *multi-objective* problem can be defined, without loss of generality, as:

$$\min_{\mathbf{x} \in X \subset \Re^n} f_i(\mathbf{x}) \quad \forall i = 1, \ldots, m \tag{2.4}$$

subject to any non-negative and equality constraints:

$$\mathbf{e}(\mathbf{x}) \equiv (e_1(\mathbf{x}), \ldots e_a(\mathbf{x}) \geq 0), \tag{2.5}$$

and

$$\mathbf{b}(\mathbf{x}) \equiv (b_1(\mathbf{x}), \ldots b_d(\mathbf{x}) = 0). \tag{2.6}$$

If there are m different objectives, then the image of the feasible search space, X, through $\mathbf{f}()$ can be denoted by $Y \subset \Re^m$. Elements of Y are commonly referred to as objective vectors (or criteria vectors). As often the objectives being optimised are in competition, there is typically no single global optimum to *multi-objective* problems, rather a set of globally optimal solutions exist (potentially infinite in cardinality), referred to as the Pareto set, containing Pareto optimal solutions. A decision vector $\mathbf{x}$ is said to be Pareto optimal ($\mathbf{x} \in \mathcal{P}$) iff $\nexists \mathbf{u} \in X, \mathbf{u} \prec \mathbf{x}$, where the $\prec$ (dominance) relationship is defined as:

$$\mathbf{u} \prec \mathbf{x} \ \texttt{if} \ \ (f_i(\mathbf{u}) \leq f_i(\mathbf{x}), \forall i) \wedge (\exists i \,|\, f_i(\mathbf{u}) < f_i(\mathbf{x})). \tag{2.7}$$

Via dominance a partial order can be placed on pairs of decision vectors; either vector $\mathbf{x}$ dominates $\mathbf{u}$ (in which case we can say that $\mathbf{x}$ is *better* than $\mathbf{u}$), $\mathbf{u}$ dominates $\mathbf{x}$ ($\mathbf{u}$ is better), or neither dominate each other (they are mutually non-dominating), in which case, without additional preference information about the objectives, we are *indifferent* between the solutions. This relationship can be seen geometrically in Figure 2.1 where the shaped region depicts the area dominated under minimisation.

As the number of objectives increases, so does the relative proportion of objective space which is *mutually non-dominating* with a solution ($1 - \frac{1}{2^{m-1}}$). Because of this, Pareto

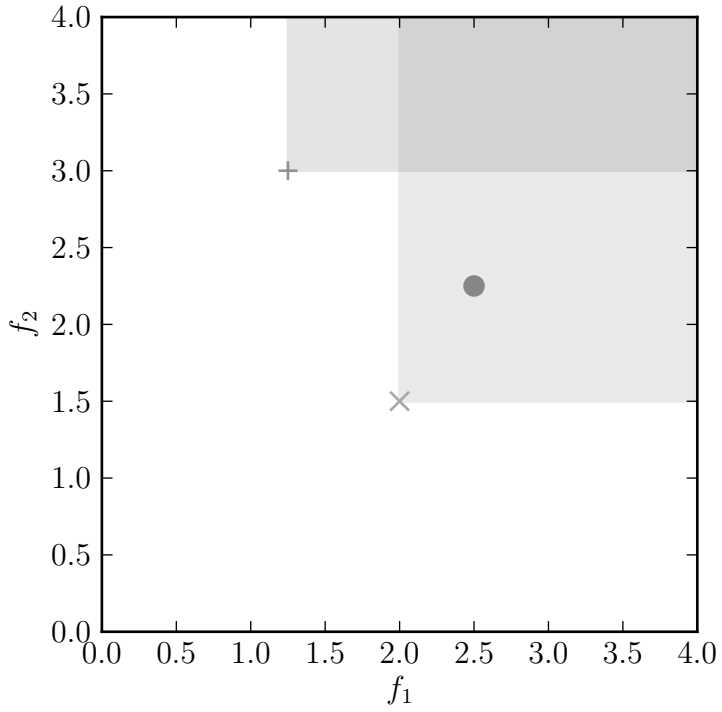


Figure 2.1.: Pareto dominance, regions dominated under minimisation by 'X' and '+' are shaded. Both 'X' and '+' are *mutually non-dominating* and 'X' dominates 'O'

quality measures on solutions rapidly lose their discriminating capabilities as m increases, as the probability that any other point in space is *incomparable* with another point fast approaches 1. Additionally, as the overwhelming likelihood is that *any* solution evaluated when m is large is mutually non-dominating with the set of solutions found so far, any archive of *mutually non-dominating* solutions stored rapidly reaches capacity (if limited by size), or grows at such a rate as to impede algorithm convergence (by spreading out solutions in a region which is not close to the optimal front $\mathcal{F}$).

2.6. Ranking methods

The PSO heuristic (and most other optimisers) require multiple iterations/generations to complete the optimisation. With PSO each particle is updated/flown once per generation using the best solutions from the previous generations. Selectional of a solution from a *mutually non-dominating* set to use in the next generation of an algorithm becomes a non-trivial task as the *mutually non-dominating* set of grows larger than the number of solutions used in the subsequent generation.

In light of this a number of other quality measures have recently been devised in the literature, which aim to provide a degree of differentiation between solutions which would

be viewed as otherwise equivalent when using a Pareto comparison.

Here we will investigate a selection of ranking methods that have been used for *multi-* and *many-objective* optimisation. In chapters 3 and 4 we see an empirical study comparing their effectiveness on many objective problems.

2.6.1. Favour relation

Adapting the concept of dominance the favour relation (FR) (Drechsler et al., 2001) comparator and ranking method considers the proportion of dominance of one solution over another. With this proportional-dominance a directed graph is produced connecting the solution to each other, with a direct edge from any solution proportionally dominated to its proportionally dominating counterpart. This graph inevitably results in many cycles which can be collapsed into single nodes containing multiple solutions. The resultant graph is a directed tree upon which an ordering can be computed providing a ranking to the solutions within the nodes of the tree.

$$fav(\mathbf{u},\mathbf{v}) = \begin{cases} \mathbf{u} \prec_{fav} \mathbf{v} & \text{if } |\{i \mid f_i(\mathbf{u}) < f_i(\mathbf{v})\}| > |\{i \mid f_i(\mathbf{v}) < f_i(\mathbf{u})\}| \\ \mathbf{v} \prec_{fav} \mathbf{u} & \text{if } |\{i \mid f_i(\mathbf{u}) < f_i(\mathbf{v})\}| < |\{i \mid f_i(\mathbf{v}) < f_i(\mathbf{u})\}| \\ \mathbf{u} \equiv_{fav} \mathbf{v} & \text{otherwise} \end{cases} \tag{2.8}$$

As can be seen from (2.8), if $\mathbf{u} \prec \mathbf{v}$ then $\mathbf{u} \prec_{fav} \mathbf{v}$, however if $\mathbf{u}$ and $\mathbf{v}$ are mutually non-dominating (incomparable under Pareto comparison) $\mathbf{u}$ will still dominate under the favour relation if is it better on more objectives. When $\mathbf{u} \prec_{fav} \mathbf{v}$ a directed edge from $\mathbf{v}$ to $\mathbf{u}$ is constructed in the relations graph.

Data: $\mathbf{G}$ = graph of all solutions with *no* edges

Data: $\mathbf{P}$ = set of all solutions

for $i \in \{i|0 < i < |\mathbf{P}|\}$ **do**

  for $j \in \{j|i < j \leq |\mathbf{P}|\}$ **do**

    if $\mathbf{P}_i \prec_{fav} \mathbf{P}_j$ **then**

      $\mathbf{G} = \mathbf{G}+$ directed edge from $\mathbf{P}_j$ to $\mathbf{P}_i$

    else if $\mathbf{P}_j \prec_{fav} \mathbf{P}_i$ **then**

      $\mathbf{G} = \mathbf{G}+$ directed edge from $\mathbf{P}_i$ to $\mathbf{P}_j$

    end

  end

end

Algorithm 1: Favour relation (FR) ranking algorithm

There have been many adaptations to the concept of partial dominance, some take on a more basic version of the Favour relation algorithm, ranking each solution by the mean number of objectives that it dominates all other solutions. (Moritz et al., 2013)

2.6.2. *k*-optimality

k-optimality (KO) proposed in Di Pierro et al. (2007) ranks solutions by dominance on subsets of objects.

$$rank_{\mathbf{u}}^{KO} = \max\left(\left\{k|\mathbf{u} \in F^{\mathbb{O}}, \forall \mathbb{O}, |\mathbb{O}| = m - k\right\}\right) \tag{2.9}$$

where $\mathbb{O} \subset \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_i(\mathbf{x}), \ldots, f_m(\mathbf{x})\}$ is a subset of objective functions and $F^{\mathbb{O}}$ is the non-dominated set defined given the set of objective functions defined by $\mathbb{O}$. [2]

As each combination of objectives (with removal) is computed for each solution, complexity grows rapidly with the number of objectives (Algorithm 2). However when comparing solutions with missing data on one or more objectives this approach holds some promise in comparison to others.

[2]Note: the computation cost of calculating this for a set of solutions makes it unusable beyond 20 objectives for all but expensive optimisation problems, as it tends to swamp the objective evaluation time cost.

$\mathbf{P}$ = set of all solutions

$\mathbf{R} = \{m | \forall \mathbf{x} \in \mathbf{P}\}$

$\mathbf{D} = \{False | \forall \mathbf{x} \in \mathbf{P}\}$

for $k \in \{k | 1 < k < m\}$ **do**
  $found = False$
  $\mathbf{C}$ = the set of combinations of indices $\binom{m}{m-k+1}$
  for $i \in \{i | 0 < i \leq |\mathbf{P}|, \mathbf{D}_i \neq True\}$ **do**
    for $j \in \{j | 0 < j \leq |\mathbf{P}|, j \neq i\}$ **do**
      for $l \in \{l | 0 < l \leq |\mathbf{C}|\}$ **do**
        $\mathbf{V} = \{v | v \in \mathbf{C}_l\}$
        $flag = False$
        for $s \in \{s | 0 < s \leq |\mathbf{V}|\}$ **do**
          $p = \mathbf{V}_s$
          if $f_p(\mathbf{P}_i) < f_p(\mathbf{P}_j)$ **then**
            $flag = True$
            break
          end
        end
        if $flag \equiv True$ **then**
          $\mathbf{D}_i = True$
          break
        end
      end
      if $\mathbf{D}_i \equiv True$ **then**
        break
      end
    end
    if $\mathbf{D}_i \not\equiv True$ **then**
      $\mathbf{R}_i = \mathbf{R}_i - 1$
      $found = True$
    end
  end
  if $found \equiv False$ **then**
    $k = k - 1$
    break
  end
end

Algorithm 2: k-optimality ranking algorithm

2.6.3. Crowding Distance

There have been many ranking methodologies that aim to provide a preference based upon the distribution of solutions across the approaching front. Generally these methodologies are used for archiving constraints as they provide a method for removing solutions that hopefully has a minimal effect on the coverage.

One of the most popular methodologies is crowding distance as used in the NSGA-II algorithm. This computes a ranking based upon the relative volumes of the hyper-cube surrounding each solution with the provision that solutions on the edge have an infinite hypercube therefore giving them an infinite rank. (When applied, a high rank is preferred as this indicates solutions that are further from others.)

Like many other ranking systems it is difficult to identify subsets of the population that are independent, therefore when used for removal typically the entire populations ranking must be recomputed after each individual solution is removed.

We implement the crowding distance (CD) as used in NSGA-II, which computes the size of the hypercube around each $\mathbf{u} \in F$ (Deb et al., 2000).

With increasing number of dimensions finding a uniform spread of points on the true Pareto front becomes very difficult. CD attempts to maximise the even spread of solutions on the front (and has recently been used in*many-objective* archive maintenance, e.g. de Carvalho and Pozo (2011)).

Data: $\mathbf{P}$ = set of all solutions

Data: $rank_s = 0, \forall s \in \mathbf{P}$

Data: m = number of objectives

for $i \in \{i | 1 < i \leq m\}$ **do**

$\mathbf{S} = sort(\mathbf{P}, i)$ sorted set of solution sorted upon the i^{th} objective in ascending order

$rank_{smin} = \infty, \quad smin = \mathbf{S}_1$ first solution in the sorted set.

$rank_{smax} = \infty, \quad smax = \mathbf{S}_{|\mathbf{S}|}$ last solution in sorted set. $range = smax_i - smin_i$

if $range > 0$ **then**

for $j \in \{j | 2 < j \leq |\mathbf{S} - 1\}$ **do**

$sprevious = \mathbf{S}_{j-1}$

$scurrent = \mathbf{S}_j$

$snext = \mathbf{S}_{j+1}$

$rank_{scurrent} = rank_{scurrent} + \frac{snext_i - sprevious_i}{range}$

end

end

end

Algorithm 3: Crowding Distance (CD) algorithm

2.6.4. Average Ranking

Since it is difficult and sometimes impossible to know the scaling for each objective some methods attempt to not use the location but rather the relative location with respect to the population. These systems employ the ordering of solutions for each objective allowing them this independence. However there is no knowledge of whether or not the entire populations has collapsed upon a small region of objective space.

Average Ranking (AR) maps the fitness values $f_i(\mathbf{x})$ to a single value which is used directly to rank solutions. AR is equivalent to Average Weighted Ranking (Bentley and Wakefield, 1997) with a weight of 1 for all objectives.

$$rank_{\mathbf{u}}^{AR} = \sum^{\forall i} \mathbb{S}_{\mathbf{u}}^{i} \tag{2.10}$$

Where $\mathbb{S}^i$ is the sorted set of objective values for the i^{th} objective and $\mathbb{S}_{\mathbf{u}}^{i}$ is the location of the objective vector corresponding to $\mathbf{u}$ in the sorted set.

2.6.5. Sum of Ratios

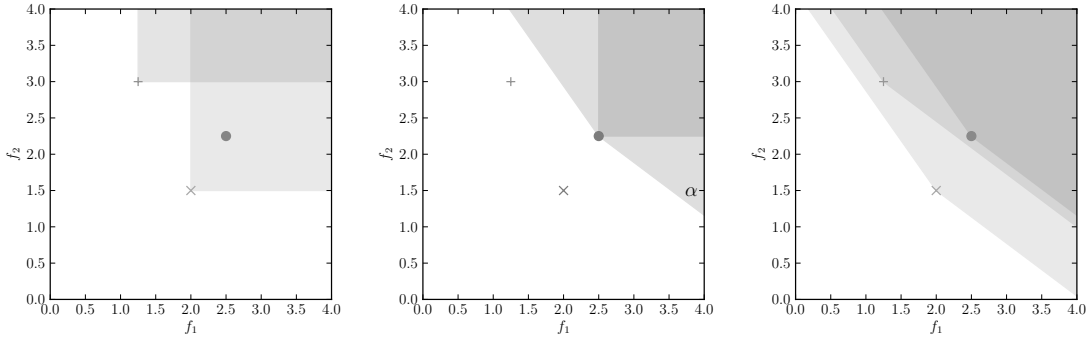
Sum of Ratios (SR) is similar to AR but attempts to incorporate some information of the distribution, normalising by the range on values upon each objective. Similar to average ranking this provides a form of ordering however weights its objectives influence by the range of solutions upon that objective.

$$rank^{SR}_{\mathbf{u}} = \sum^{\forall i} \frac{f_i(\mathbf{u}) - \min(\mathbb{S}^i)}{\max(\mathbb{S}^i) - \min(\mathbb{S}^i)} \tag{2.11}$$

Where $\mathbb{S}^i$ is the sorted set of objective values for the i^{th} objective and $\mathbb{S}^i_{\mathbf{u}}$ is the location of the objective vector corresponding to $\mathbf{u}$ in the sorted set. SR is equivalent to Sum of Weighted Ratios (Bentley and Wakefield, 1997) with a weight of 1 for all objectives.

2.6.6. Controlling Dominance Area of solutions

Sato et al. (2007a) proposes altering the area dominated by a solution by adjusting the angles that meet the axis bounding this area, as depicted in Figure 2.2. Controlling Dominance Area of solutions (CDAS), originally proposed for multi objective problems, has in resent years gained popularity in *many-objective* algorithms. (Sato et al., 2007b,c; Ishibuchi et al., 2008; de Carvalho and Pozo, 2010).



(a) Pareto Dominance, with 'x' dominating 'o'.

(b) CDAS Dominance, α shows the new angle used to define the new dominating region for one solution.

(c) CDAS Dominance, shown on all solutions, here 'x' dominates '+' which also dominates 'o'

Figure 2.2.: Dominance modification, shaded region shows the region that is dominated by the solution at its apex (under minimisation on all objectives).

CDAS has one adjustable vector of parameters $\mathbf{s}$. The modified Pareto comparison is easy to compute by transforming the objective space so that standard Pareto comparison in this space is analogous to the modification of the dominated regions. The area dominated can be increased or reduced setting $s_i, \{s_i\}^m_{i=1}$ either smaller or larger than 0.5 respectively.

The transformed objective values $\mathbf{f}'(\mathbf{s}, \mathbf{x})$ are computed in equation 2.12:

$$f_i'(\mathbf{s}, \mathbf{x}) = \frac{r(\mathbf{x}) \cdot \sin\left(\omega_i(\mathbf{x}) + s_i\pi\right)}{\sin\left(s_i\pi\right)} \tag{2.12}$$

$$\text{Where:} \quad r(\mathbf{x}) = ||\mathbf{f}(\mathbf{x})||_2 \tag{2.13}$$

$$\omega_i(\mathbf{x}) = \arccos\left(\frac{f_i(\mathbf{x})}{r(\mathbf{x})}\right) \tag{2.14}$$

Here we will use the same s_i value for all objectives, therefore henceforth it will simply be denoted as s.

It has recently been applied to many objective PSO (de Carvalho and Pozo, 2011) with promising results upon the DTLZ test problems (Deb et al., 2002).

The nature of the transform when $s_i < 0.5$ increases selection pressure upon the edges and centre of a convex front (see Figure 3.1) to varying degrees, depending upon the shape of the non-dominated front being transformed. A full investigation of this will follow in chapter 5.

2.6.7. Self Controlling Dominance Area of Solutions

Self Controlling Dominance Area of Solutions (S-CDAS), which has been proposed as an improvement to CDAS (Sato et al., 2010), appears insensitive to parameter choices on the range of problem landscapes and wider algorithm configurations with which it has been tested. As opposed to CDAS, which was originally developed with *multi-objective* problems with 2-3 objectives only, S-CDAS was from its inception developed with awareness of *many-objective* problems - so there is little surprise in how it has supplanted CDAS in the recent *many-objective* literature (Junior et al., 2012; Sato et al., 2010). However there has been very little comparison between CDAS and S-CDAS outside of the original papers.

S-CDAS requires an additional parameter to be set, δ. This is less dependent upon the problem than the s parameter in CDAS, here we have chosen to use $\delta = 0.001$. Notably there is no suggested specification in the literature of a good δ value to use. Unlike CDAS it is not clear that the resultant weightings provided are a finer grain subset of traditional Pareto sorting - it is possible for solutions that are *not* within the first *mutually non-*

dominating front to share the same rank with those inside.

$$a = [1, 2, 3] \tag{2.15}$$

$$b = [0, 1, 2] \tag{2.16}$$

$$c = [2, 0, 4] \tag{2.17}$$

Here solution b dominates a. With b, c *mutually non-dominating* and a, c *mutually non-dominating*. However the ranks are:

	Solution	Rank
a	$[1, 2, 3]$	1
b	$[0, 1, 2]$	0
c	$[2, 0, 4]$	1

S-CDAS provides an adaptive $\mathbf{s}$ parameter for CDAS, this adaptation is linked to the relative location of each solution within the population. To attain this firstly all locations are translated such that the minimum values on each objective is δ and a matrix of min/max values is constructed:

$$L_{i,j} = \begin{bmatrix} f_1^{max} & f_2^{min} & \cdots & f_m^{min} \\ f_1^{min} & f_2^{max} & \cdots & f_m^{min} \\ \vdots & & f_i^{max} & \vdots \\ f_1^{min} & f_2^{min} & \cdots & f_m^{max} \end{bmatrix} - \delta I^{m \times m} \tag{2.18}$$

$$\mathbf{O} = \left(f_1^{min}, f_2^{min}, \ldots f_i^{min}, \ldots, f_m^{min}\right) - \delta \mathbf{1} \tag{2.19}$$

With $L_{i,j}$ φ is computed for each solution $\mathbf{x}$.

$$\varphi_i(\mathbf{x}) = \arcsin\left(\frac{r(\mathbf{x}) \sin\left(\omega_i(\mathbf{x})\right)}{||L_i - f(\mathbf{x})||_2}\right) \tag{2.20}$$

$$\text{Where:} \quad r(\mathbf{x}) = ||\mathbf{f}(\mathbf{x}) - \mathbf{O}||_2 \tag{2.21}$$

$$\omega_i(\mathbf{x}) = \arccos\left(\frac{f_i(\mathbf{x}) - O_i}{r(\mathbf{x})}\right) \tag{2.22}$$

then each other solution $\mathbf{y}$ has its position updated with respect to the solution $\mathbf{x}$:

$$f'_i(\mathbf{y}, \mathbf{x}) = \frac{r(\mathbf{y}) \sin\left(\omega_i(\mathbf{y}) + \varphi_i(\mathbf{x})\right)}{\sin\left(\varphi_i(\mathbf{x})\right)} \tag{2.23}$$

if $f'(\mathbf{y}, \mathbf{x})$ is dominated by $f(\mathbf{x}) - \mathbf{O}$ then the rank of $\mathbf{y}$ is increased.

From this we can see that computationally S-CDAS stands at a disadvantage to CDAS as it requires upto $\mathcal{O}\left(n^2 - n\right)$ domination checks for each generation opposed to $\mathcal{O}\left(n \log(n)\right)$ for CDAS, n as the number of solutions to be compared. Also the magnitude, $r(\mathbf{x})$ and $\omega_i(\mathbf{x}), \forall i$ must be computed afresh once for each new population[3]. This is in stark comparison to CDAS where all the transformed locations can be computed once and used across multiple generations and populations.

Unlike CDAS which gives a ternary label to solutions (*mutationally non-dominating* or *dominating/dominated*), S-CDAS provides an integer rank for each solution, with lower ranks considered to be better (Junior et al., 2012; Sato et al., 2010). Therefore there are many ways to incorporate it into an algorithm.

2.6.8. Many and Multi Objective PSO (MOPSO)

A decade ago (circa 2002), researchers began publishing *multi-objective* (MO) variants of Particle Swarm Optimisation (PSO) Coello Coello and Lechunga (2002); Fieldsend and Singh (2002a); Hu and Eberhart (2002); Parsopoulos and Vrahatis (2002) (although an unpublished paper on the area exists from 1999 Moore and Chapman (1999)), typically referred to as MOPSO algorithms. Since these works there has been a large growth in the number and range of MOPSO algorithms published in the literature, which has largely tracked the growth of, and range of, general *multi-objective* evolutionary algorithms (MOEAs), with comparison/selection/variation operators popularised in the MOEA field rapidly being converted into aspects of MOPSOs when direct analogies could be drawn (e.g. the use of *dominance*, *hypervolume indicator*, *clustering*, *archive maintenance*, *mutation/turbulence operators*, etc.). As the number of distinct MOPSOs has grown, a number of papers have provided overviews of the range of approaches that can be taken, along with some empirical comparisons (e.g. Fieldsend (2004); Padhye (2009); Padhye et al. (2009); Reyes-Sierra and Coello Coello (2006)). However there has been relatively little

[3] $r(\mathbf{x})$ and $\omega(\mathbf{x})$ are functions of $\mathbf{O}$ so must be recomputed if the range of objective values on any objects changes.

work thus far examining *many*-objective PSO performance (i.e., on problems with four or more objectives) Wickramasinghe and Li (2009); de Carvalho and Pozo (2011). de Carvalho & Poze found crowding distance to be effective for archive maintenance when combined with CDAS and Average Ranking (AR) for selection in many objective PSO (de Carvalho and Pozo, 2011).

Within the scientific literature there have been a multitude of different MOPSO variants proposed, most recently the consideration of intercommunicating sub swarms, each with its own goal or search algorithm has been hailed as an effective approach on some problem domains.

Multi Swarm PSO

Even with the inherent diversity provide by the PSO heuristic there are still many problems where using a single variation of PSO (even with complex intercommunication topologies) the main problem being one is unable to effectively converge and stay diverse. Recently (Voglis et al., 2013) a multi swarm was proposed that allowed each sub-swarm to use different variations of the PSO algorithm (e.g. different selection/archiving criteria) while maintaining an inter swarm communication of best locations. This allowed each sub-swarm to utilise different domain/objective properties of the problems simultaneously.

2.7. Test problems

When developing algorithms and comparing the performance we need ways of efficiently providing problems upon which these algorithms can be tested. Typically real-world applications are not the best problems for initially comparing the performance, as many of these problems are costly for each fitness function evaluation. Additionally real-world problems are often unsolved, commonly no perfect set of solutions have been found or can be proven to have been found. To overcome this there have been various synthetic problems developed for which we do have a perfect mathematical set of solutions, and yet when solved in numerical manner aim to replicate the types of conditions that algorithms applied to real world problems must face.

Test problems fall into two categories continuous, and non-continuous. These are defined by whether or not the domain parameters are continuous or non-continuous. Examples of discreet problems are those of sequencing problems, such as shortest Euclidean path (Martello and Toth, 1990). Some problems have a mix of continuous and non-continuous

parameters, e.g scheduling problems where each event can be taken in multiple (distinct locations) but the time is a continuous parameter, these problems are commonly decomposed into fully discrete problems by pre-defining possible start and end time sockets (Taillard, 1993).

Considering the scope we limit ourselves here to continuous problems only, the two major collections of problems we consider are the DTLZ and walking fish group (WFG) problems. Both of which provide us problems which can be parametrised both in objective and in search space, this aims to give a degree of control to the difficulty of the search for each problem.

2.7.1. DTLZ

DTLZ, is considered the first fully scalable *multi-objective* test suit. Scalable here is the ability to freely select the number of objective and parameters for each problem. Only requiring that the number of parameters is greater than objectives. This provides a degree of freedom when testing algorithms, importantly it lets the user separate the performance on different problem types and objective/parameter number. This means that unlike its predecessors, such as the ZDT collection (Zitzler et al., 2000), DTLZ is both a multi-and many objective suite. The ability to scale the problem over objectives and parameters is achieved by separating into two characteristic components, a function is used to compute the distance from the optimal front, with the first $m - 1$, (m = number of objectives) parameters. Another set of functions is used to define the shape of the optimal front, this uses the remaining components of the parameter vector deciding the location on the surface. To correctly analyse results we must understand the properties of each of these problems. Here we will discuss DTLZ problems 1 to 6, the properties and shortcomings. See figure 2.3.

The DTLZ1 problem which combines the linear simplex front shape with many deceptive fronts. A deceptive front is a non-optimal set of solutions which sit in a local optimal, where moving a small amount in any direction in objective space will result in a worse solution. Additionally, the encoding of the distance function which produces theses deceptive fronts results in regions of the objective space having no solutions. These regions form bangs completely separating each deceptive front, taking the shape of the optimal front i.e. a simplex. The frequency of these deceptive fronts increases as we approach the optimal front. DTLZ2 is the easiest of these problems consisting of a monotonic minimising

Problem	Separability	Bias	Geometry	Derived Shape
DTLZ 1	S	NB	Linear, Descriptive Fronts	Simplex
DTLZ 2	S	NB	Convex	Hypersphere Centered on the origin
DTLZ 3	S	NB	Convex, Descriptive Fronts	Hypersphere Centered on the origin
DTLZ 4	S	B	Convex	Hypersphere
DTLZ 5	*	*	Degraded Convex	Hypersphere
DTLZ 6	S	B	Discontinues, Mixed	Distributed on the surface of a Hypersphere

Figure 2.3.: Comparisons of the DTLZ. With S = Separable, (Optimal solution can be found by optimising one parameter at a time in a single pass), NS = Not Separable. B = Biased, there is a high probability of finding solutions is some regions of the objective space & across the optimal front than others (In these problems the degree of this is parametrised). NB = Not Biased. The Geometry/Shape of the optimal front is described looking down onto the surface from the positive sector. (i.e a hypersphere is describe as a convex shape)
* (Huband et al., 2006) identified that the optimal set of DTLZ5 is not fully defined.

gradient towards the optimal front. It therefore is commonly used to compare the ability of an algorithm to fully cover the front, unlike other problems where it is common to not have a single solution close/on the optimal front. With DTLZ2 even in high numbers of objectives a random sample will find a solution close to the optimal front (Huband et al., 2006).

DTLZ3 is a combination of DTLZ1 and DTLZ2, taking the convex hyper-spherical shape from DTLZ2 and the deceptive fronts function as DTLZ1. DTLZ4 takes on the same shape as DTLZ2, but with a non-uniform mapping from search space, such that sampling the optimal region of the search space uniformly will give a strongly non-uniform distribution across the optimal front in objective space. DTLZ5 is, unlike the above problems, in that it collapses down to a lower dimensional surface/line (Huband et al., 2006).

The major flaw in DTLZ is the location of the optimal set in parameter space. (Huband et al., 2006) Any vector $\mathbf{x}$, see equation 2.24, is on the optimal front. This makes the problems unrepresentative of real-world applications.

$$x_i = 0.5 \qquad \forall i < m \tag{2.24}$$

$$x_i \in [0, 1] \qquad \forall i \geq m \tag{2.25}$$

However, there has been work applying a transforms to parameter space prior to computation of the DTLZ test problems, aiming to overcome this shortfall. (Deb et al., 2006a)

2.7.2. Walking Fish Group

Here we describe the more recent addition to the multi- many objective test problems, the walking fish group toolkit, introduced in Huband et al. (2006), was designed to give the ability to develop test problems by combining a set of multiple functions. Much like the DTLZ problems the walking fish group toolkit provides a collection of shape functions which can be can combined with transforms to add features to the problem. Unlike its predecessors the difficulty can be controlled directly by the test designer, by a set of combinable transforms. The original authors proposed a set of 9 combinations of these to form a new test suit. Known as Walking Fish Group (WFG) 1-9 these problems have become popular for the many objectives. The WFG 1-9 problems are built in such a way as to not have the same shortfalls as DTLZ, a non-linear transform is applied to the parameter vectors before they are passed onto the shape and distance functions. This transform is also used to control the problems complexity, by changing the density of solutions through the space.

Problem	Separability	Bias	Geometry	Derived Shape
WFG1	S	B	Concave/Convex (Mixed)	Step function mapped onto the surface of a hypersphere
WFG2	NS	NB	Concave, Discontinues	Step function mapped onto the surface of a hypersphere
WFG3	NS	NB	Linear, Degenerated	A Lower dimensional (line/surface) on the center of the simplex
WFG4	S	NB	Convex	Lineally Scaled hypersphere, with different scaling on each objective
WFG5	S	NB	Convex	-
WFG6	NS	NB	Convex	-
WFG7	S	B	Convex	-
WFG8	NS	B	Convex	-
WFG9	NS	B	Convex	-

Figure 2.4.: Comparisons of the Walking Fish Group Problems. With S = Separable, (Optimal solution can be found by optimising one parameter at a time in a single pass), NS = Not Separable. B = Biased, there is a high probability of finding solutions is some regions of the objective space & acrosse the optimality front than others (In these problems the degree of this is parametrised). NB = Not Biased. The Geometry/Shape of the optimal front is described looking down onto the surface from the positive sector.

2.8. Performance metrics

For multi and many-objectives both the optimal *mutually non-dominating* solutions and any sub-optimal sets can have a large cardinality. Compare these sets, to determine the performance of any *mutually non-dominating* set, is our topic here. One way of doing this is to measure the distance between the sets to provide a single score. Van Veldhuizen and Lamont (1998) introduced the Generational Distance (GD) which provides the average of the distance from each solution to *its* closest point upon the optimal set.

Alone however, generational distance is not a good equivalence measure as it is possible for a very small set of solutions, which do not spread across the entire optimal front evenly, to score better than a set of solutions which are a closer approximation in shape to the optimal front. Therefore Van Veldhuizen and Lamont (1998) proposed a measure to accompany GD, Inverse Generational Distance (IGD), this takes the average distance from each solution on the optimal front to its closest solution on the approximated set. This metric provides an indication of the distribution across the entire Pareto optimal front, provided that this is known and an even distribution of solutions can be found upon it with which to compute these metrics.

Since IGD is also dependent on the distance to the optimal front, it does not independently measure the distribution. Many other spacing metrics have been proposed however, there is little consensus within the current *multi-* and *many-objective* literature as to which spacing metrics provide an accurate measure. This comes into play in particular for problems where the optimal front has sudden changes in its local gradient, such as the WFG1 and 2 functions (see section 2.7.2), in these cases the spacing metrics which do not incorporate the shape of the optimal front penalised solutions which are evenly distributed across the front since there are regions, those sections with a greater rate of change of gradient, for which the Euclidean distance is shorter even though the distance along the surface of the optimal front is equal.

Here therefore we will continue our discussion considering Generational Distance and Inverse Generational Distance *only*. By comparing these in tandem one is able to observe both the convergence, closeness to the optimal set, and distribution/diversity, spread across the optimal set.

2.8.1. Generational Distance

As introduced above, Generational Distance is the measure of the average, across all solutions, of the distance from a given solution on the approximated set to its closest location on the optimal set. Here we define this in a formal manner and investigate some anomalies that have given rise to the proposition of a modified GD, known here as GD_p.

Given $\mathcal{P} = \{x_1, x_2, \ldots, x_N\}$, the approximated Pareto set of solutions, and $\mathcal{F} = \{y_1, y_2, \ldots, y_M\}$ an evenly distributed sample of solutions upon the optimal front. Coello et al. (2007) reports the GD computed as follows:

$$\mathrm{GD}\left(\mathcal{P}, \mathcal{F}\right) = \frac{1}{N}\left(\sum_{\forall x_i \in \mathcal{P}} dist(x_i, \mathcal{F})^p\right)^{\frac{1}{p}} \tag{2.26}$$

Here the 2-norm is used to compute the $dist(x_i, \mathcal{F})$, with a p value of 2:

$$dist(x_i, \mathcal{F}) = \min\left(||x_i - y_j||_2\right) \quad \forall y_i \in \mathcal{F} \tag{2.27}$$

As demonstrated in Schütze et al. (2012) showing that as the number of solutions in the approximated set increases differences in the GD values occur. To mitigate this the adoption of comparing similar archive sizes only has become the norm, however since many evolutionary algorithms now have varying archive sizes we face difficulties in comparison from one generation to the next. Therefore an alternation to the GD was proposed in Schütze et al. (2012) to avoid this effect by using the power mean to average the distances:

$$\mathrm{GD}_p\left(\mathcal{P}, \mathcal{F}\right) = \left(\frac{1}{N}\sum_{\forall x_i \in \mathcal{P}} dist(x_i, \mathcal{F})^p\right)^{\frac{1}{p}} \tag{2.28}$$

2.8.2. Inverse Generational Distance

Inverse generational distance, as discussed above, is best used when compared side-by-side with GD. IGD is the measure of the average, across all solutions upon the optimal front, of the distance from each solution on the optimal front to its closest solution in the approximated front. For IGD to provide a good evaluation a large number of evenly distributed solutions must be obtained upon the optimal front. The algorithm as reported

Coello et al. (2007) in is:

$$\text{IGD}\left(\mathcal{P},\mathcal{F}\right) = \frac{1}{M}\left(\sum_{\forall y_i \in \mathcal{F}} dist(y_i,\mathcal{P})^p\right)^{\frac{1}{p}} \tag{2.29}$$

As with the GD here we will consider only the 2-norm with $p = 2$, the $dist(y_i, \mathcal{P})$ is analogous to equation 2.27:

$$dist(y_i,\mathcal{P}) = \min\left(||y_i - x_j||_2\right) \quad \forall x_i \in \mathcal{P} \tag{2.30}$$

Schütze et al. (2012) propose the use of the power mean for the IGD as they have for GD to similarly overcome difficulties with differing population sizes:

$$\text{IGD}_p\left(\mathcal{P},\mathcal{F}\right) = \left(\frac{1}{M}\sum_{\forall y_i \in \mathcal{F}} dist(y_i,\mathcal{P})^p\right)^{\frac{1}{p}} \tag{2.31}$$

Summary

Here we have seen a introduction to the current literature surrounding *many-objective* particle swarm optimisation. In the next chapter, we will investigate how altering the selection of guides for the many objective PSO algorithm affects the search. We will investigate how the application of additional orderings upon the first Pareto front can be used to archive this. From there a further investigation into achieving methods will follow.

3. The Effect of Selection Operators on Many objective Search

As discussed in Section 2.6.8 the PSO algorithm fails to converge properly where there are many objectives. Here we present an investigation into altering which solutions are selected from the global and local archives. This therefore affects the searching behaviour of the swarm. A comparison of six ranking methodologies, along with standard random selection, upon the first four DTLZ problems ranging from 2 to 20 objectives is provided (see Section 2.7.1) In addition a new ranking method is proposed, Controlling Dominance Area of Solutions for Ranking (CDAS-R).

3.1. Using ranked solutions to guide selection

As the non-dominated set tends to grow rapidly as the number of objectives increases it is popular across many Nature Inspired optimisers to select solutions for crossover (or other related operation) by computing an additional ordering upon the *mutually non-dominating* set. Most implementations of this will make use of some tournament based selection to ensure a good degree of diversity is still evident.

A tournament selection is where some subset of the solution set is chosen at random and among those solutions the best rank is selected. Here we will consider a tournament size of 5 as used previously in Corne and Knowles (2007).

3.1.1. CDAS as a ranking method

De Carvalho and Pozo (2011) demonstrated how effective CDAS, see Section 2.6.6, can be when used to strengthen the Pareto comparison for MOPSO. However, CDAS can only provide one additional degree of granularity. We propose the use of the CDAS algorithm to compute a ranking here, known as CDAS-R, in order to generate a finer gradation on the ranking it can provide.

By iteratively reducing the values in $\mathbf{s}$ it is possible to apply a ranking on a mutually non-dominated set F based upon the minimum values in $\mathbf{s}$ for which $\mathbf{u} \in F$ is within the new non-dominated front achieved by this mapping ($F'_{\mathbf{s}}$).[1] To our knowledge this is the first work to investigate modifying CDAS in this fashion, so we shall now provide further details of this approach. The rank of a solution, $\mathbf{u}$ is determined as:

$$rank_{\mathbf{u}}^{CDAS-R} = \min(\{S|\mathbf{u} \in F'_{\mathbf{s}}\}). \tag{3.1}$$

As with the original CDAS algorithm since ω_i and r are independent of s they can be precomputed once on the first accepted insertion into an archive (in the case of PSO, as detailed in Section 2.4, this could be either personal or global/local). If a fixed set of d transform vectors, $S = \{\mathbf{s}_j\}_{j=1}^{d}$, is considered, then $f'_i(\boldsymbol{\omega}, \mathbf{s}, \mathbf{x}), \forall j$ can also be computed only when first required, and stored for future use (across guide archives and generations).

Since if $\mathbf{u} \notin F'_{\mathbf{s}_j}$ then $\mathbf{u} \notin F'_{\mathbf{s}_{j+1}}$ where $\mathbf{s}_{j+1} \leq \mathbf{s}_j$. Iterative construction of sets for a particular rank can be determined sequentially as:

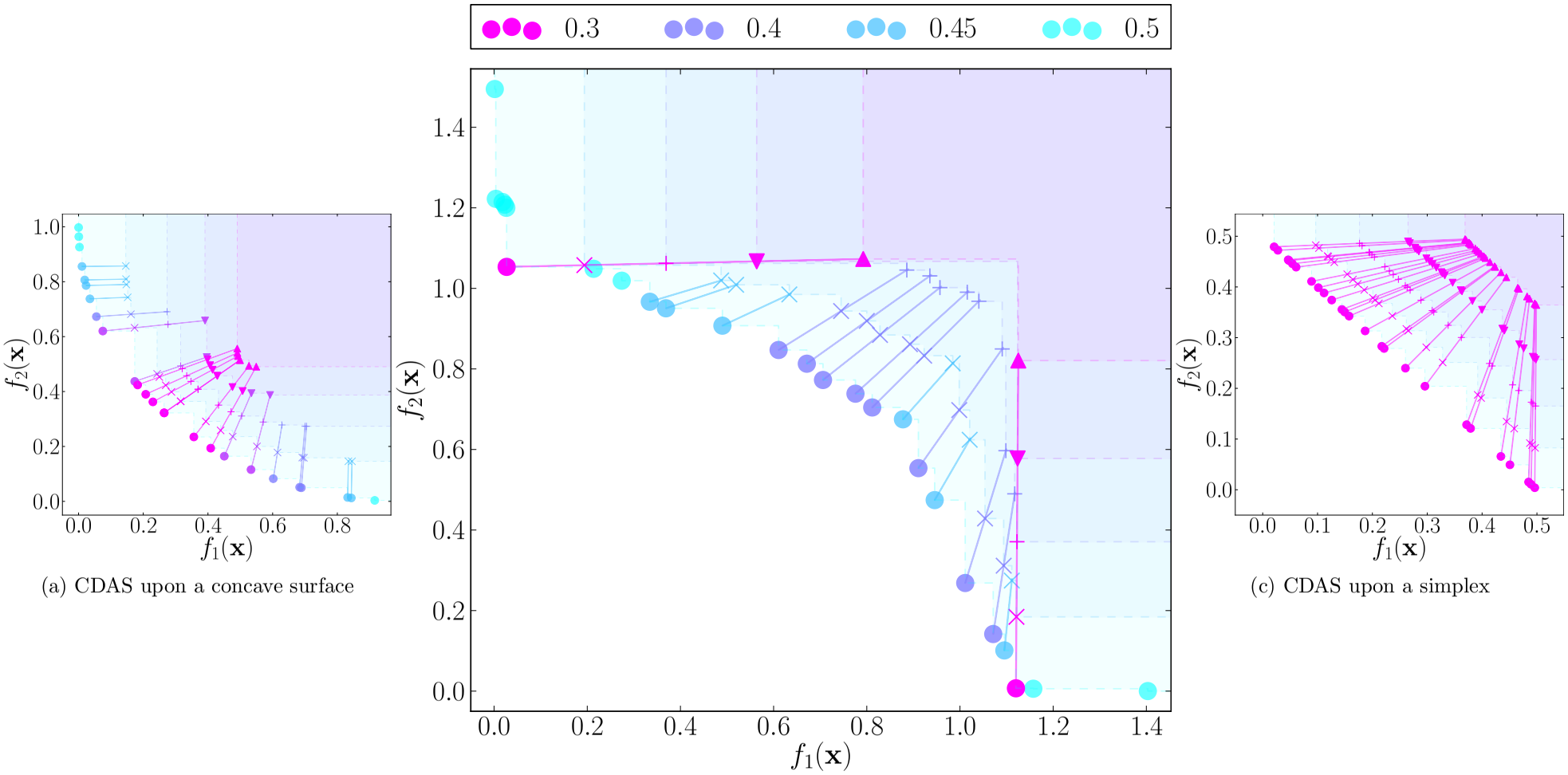
$$F'_{\mathbf{s}_{j+1}} = \left\{\mathbf{u} \in F'_{\mathbf{s}_j} \,|\, \nexists \quad \mathbf{v} \in F'_{\mathbf{s}_j}, \mathbf{v} \prec_{CDAS}^{\mathbf{s}_{j+1}} \mathbf{u}\right\}. \tag{3.2}$$

With $\prec_{CDAS}^{\mathbf{s}_{j+1}}$ defined as the dominance using the mapped CDAS values for $\mathbf{s}_{j+1}$ which only need to be computed once $\forall \mathbf{v} \in F'_{\mathbf{s}_j}$. Note that $F = F'_{\mathbf{0.5}}$ (as when $\mathbf{s}_j = \mathbf{0.5}$ the mapping is the same as standard dominance).

Since $\forall \mathbf{u} \in F'_{\mathbf{s}_i}$ the rank of $\mathbf{u}$ is independent of all solutions $\mathbf{v} \in F'_{\mathbf{s}_j}, \forall j \leq i$ it is possible to remove the worst solutions without affecting the rank of any other solution.

As we will see later, in Chapter 5, the CDAS transform can, with an appropriate $\mathbf{s}$ vector set, provide a very strong selection preference in comparison CDAS-R gives more diversity. In Figure 3.1 the solutions are ranked by the lowest $\mathbf{s}$ for which they are still *non-dominated*, with lower number being preferred when considering as a CDAS-R ranking. The shells are the result of a CDAS mapping on the front with decreasing values of $\mathbf{s}$. Here we chose $\mathbf{s} \in \{\mathbf{0.5}, \mathbf{0.45}, \mathbf{0.40}, \ldots, \mathbf{0.3}\}$. Further investigation of CDAS and CDAS-R preferences will be investigated further in Chapter 5.

[1] It is simplest to have all elements of $\mathbf{s}$ set to the same value – in situations where the objectives are known to live on different ranges the solutions can be normalised by the observed range in the stored solutions before projecting to the new locations using (2.12). This approach makes considering the order on potential $\mathbf{s}$ easier to consider.



(a) CDAS upon a concave surface

(b) CDAS upon a randomly generated DTLZ2 front

(c) CDAS upon a simplex

Figure 3.1.: CDAS-R ranked front, and the CDAS mapped fronts. The mapped sets of points '$\circ$', '$\times$', '$+$', '$\triangledown$', '$\triangle$', illustrate the set of non-dominated projected points from the original set for $\mathbf{s} = \mathbf{0.5}$ to $\mathbf{0.3}$. Note for the non-linear front how the set membership shrinks with the reduction in $\mathbf{s}$. The original non-dominated set (circles) are coloured according to their imputed CDAS rank.

3.2. Experimental Design

Here we examine the impact of using different many-objective quality measures *during* the optimisation process of a standard MOPSO algorithm. We compare across a wider range of measures than considered in other recent work in the field (de Carvalho and Pozo, 2011), and across a wider range of test problems.

Here we look at the effect of using one of the quality measures as a *selector*, and keep the design of the optimisers consistent apart from this variation, allowing us to isolate its effect. In contrast to this in chapter 4 the effect of using these quality measures for *archive maintenance* is explored.

The MOPSO algorithm is as described in Section 2.6.8. A set of non-dominated solutions is maintained for the swarm as the source of global bests, and also each particle has a set of non-dominated solutions which they maintain and provide their personal bests. These sets are bounded at 100 elements, and if this limit is breached, random removal is performed until 100 elements is reached. This simple MOPSO is then run 30 times with seven different selection protocols for determining the particular global best and personal best to be chosen for each particle in each generation. The DTLZ1-4 test functions are used with recommended parametrisation by Deb et al. (2002), see Section 2.7.1, for objectives $= \{2, 3, 5, 10, 15, 20\}$. The protocols were: Favour Relation (FR see Section 2.6.1), K-optimality (KO see Section 2.6.2), CDAS-R (see Section 3.1.1), Crowding Distance (CD see Section 2.6.3), Average Ranking (AR see Section 2.6.4), Sum of Ratios (SR see Section 2.6.5), and the baseline of random selection from the non-dominated sets (i.e., Pareto-based selection), which we denote by RR. As the first six protocols rank the non-dominated sets, these rankings were used to select guides based on tournament selection of size 5 (as used in Corne and Knowles (2007), where a subset of the many-objective operators considered here were compared in terms of their ability to *discriminate* between non-dominated solutions, and empirically evaluated within a GA). Element-wise truncation is used to manage boundary conditions, initial velocities are set at $\mathbf{0}$ and initial particle locations are distributed uniformly at random within the feasible search space. The PSO parameters are as described in Sec. 2.4, with $w = 0.5$, $c_1, c_2 = 2$ and $\chi = 1$. The number of swarm members is set at 100, and the optimisers were run for 500 generations resulting in 50,000 function evaluations.

Optimiser performance is tracked using the widely used generational distance (GD) and inverse generational distance (IGD) measures Coello Coello et al. (2002) (see section 2.8),

which quantify the convergence to the Pareto front, and the spread and convergence to the Pareto front respectively.

3.3. Analysis of Results

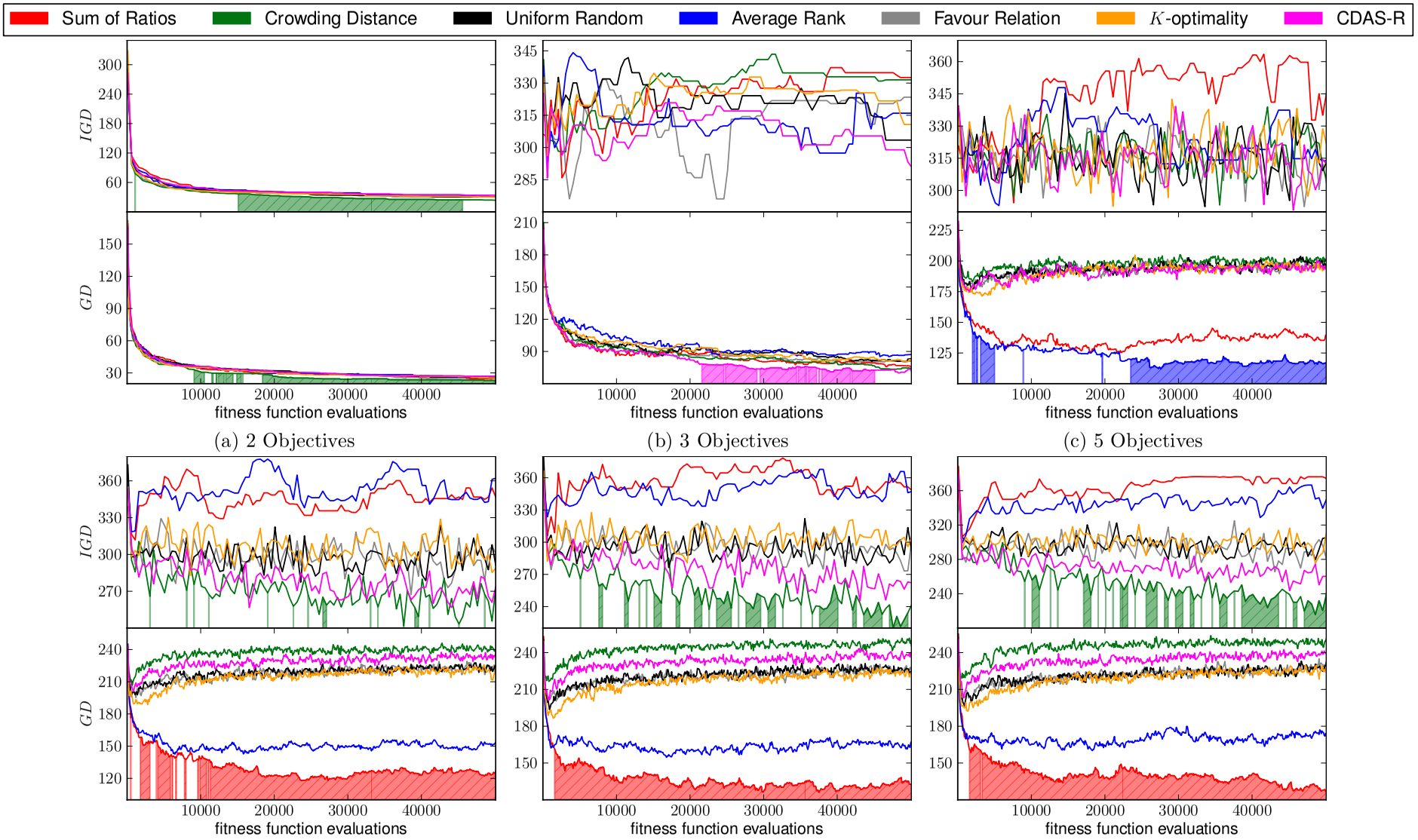
Figures 3.2, 3.3, 3.4 & 3.5 present the results of the experiments. An algorithm which is performing significantly better (as assessed by pairwise comparisons with all the other methods) is highlighted via a shaded area between its median performance line and the abscissa across the range of generations for which this is the case.[2] Plots are arranged with each page focusing upon an individual DTLZ function and objectives increasing from the top left of each plot (with the IGD and GD plots being adjacent vertically).

From Figure 3.2 we can see that, for 10+ objectives, although the Crowding Distance (CD) tends to be lower than the other methods on IGD, it is not often significantly so. From examining the GD plots, we can see that the improvement in the coverage of CD, is actually occurring simultaneously with a divergence on the GD – with SR finding significantly better converged solutions for 10+ objectives (albeit at a cost to *its* IGD). For lower numbers of objectives, ≤ 5, there is less of a clear cut pattern of performance. For DTLZ1 we therefore see there is a clear trade-off for these methods with respect to convergence and coverage high objectives.

The results tend to trend together on the two quality measures on DTLZ2 (see Figure 3.3) with AR, CDAS-R and SR all competitive on both IGD and GD, although CDAS-R tends to perform better in the early stages. As discuses in Section 2.6.6, this problem is the easiest of the DTLZ collection which for convergence at least is confirmed here with all algorithms attaining significantly better GD performance than on any other problem.

For DTLZ3, AR is seen to perform significantly better across the quality measures for many objective tests (5+ objectives). For DTLZ4, SR generally out-performs the others, although in the earlier stages for 5 and 10 objectives (Figures 3.2c & 3.2d), AR is better. We postulate that the relativly poor performance of AR here, in comparison to *its* results on DTLZ3, is because AR only takes into account the ordering and not the geometric location of solutions it will then give greater and greater preference towards the edge in comparison with SR (as it prefers solutions that are in the centre of its ordering, rather than the geometric centre Garza-Fabre et al. (2010)). This comes into play upon the DTLZ4 function where there is a strongly non-uniform distribution of solutions.

[2] In some cases the vertical axes are plotted below 0 to show this better.



(a) 2 Objectives (b) 3 Objectives (c) 5 Objectives

(d) 10 Objectives (e) 15 Objectives (f) 20 Objectives

Figure 3.2.: Selection on DTLZ1: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).

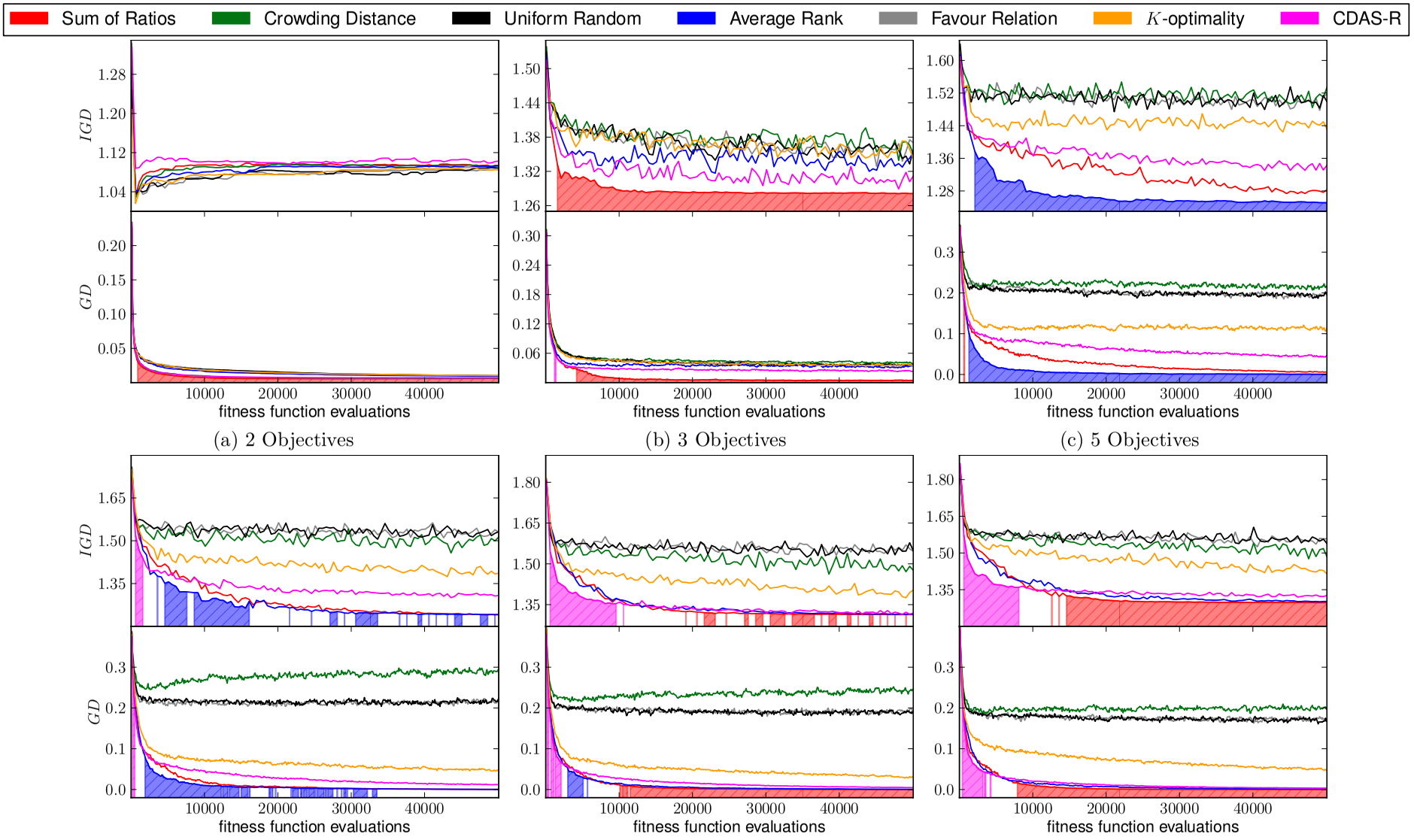
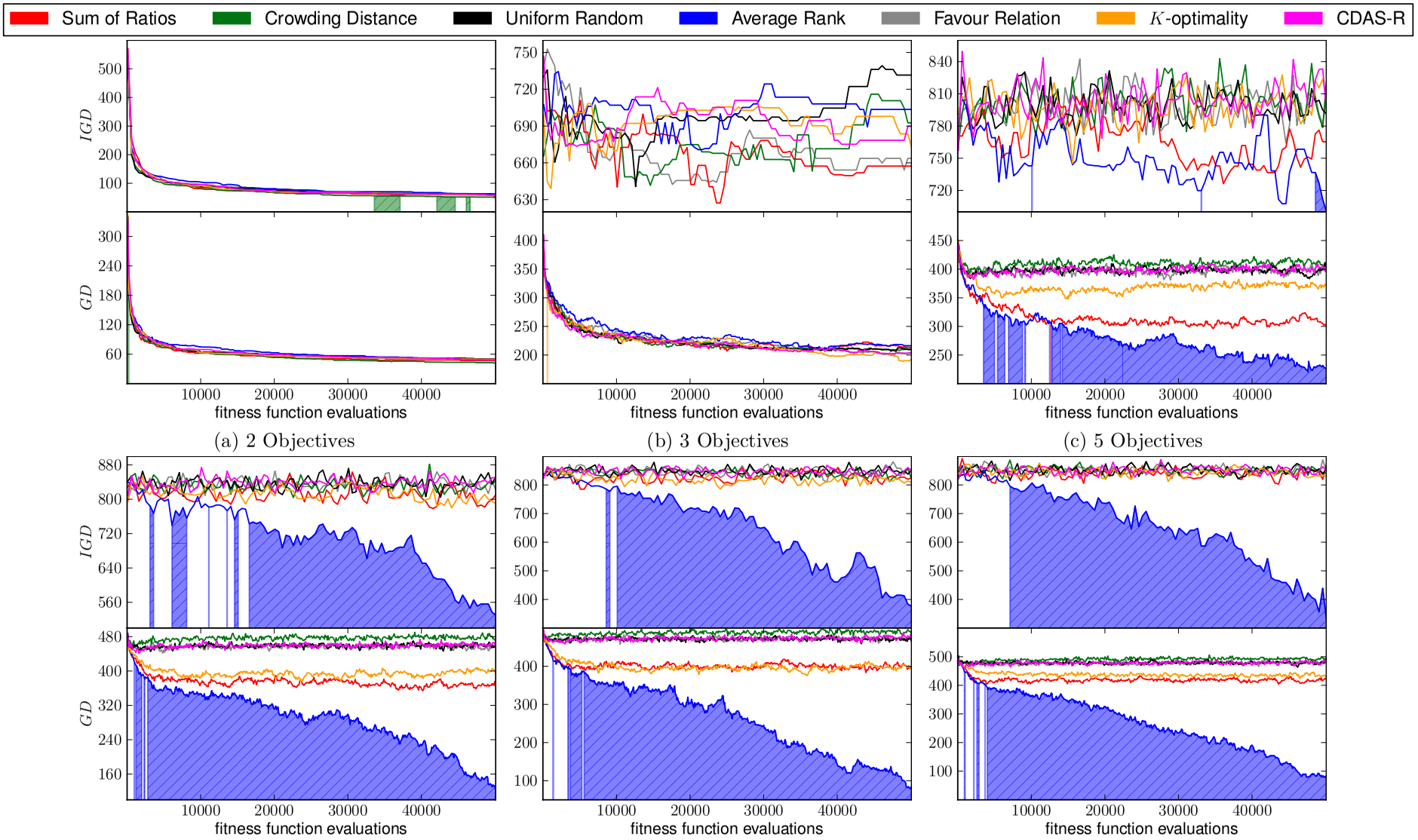


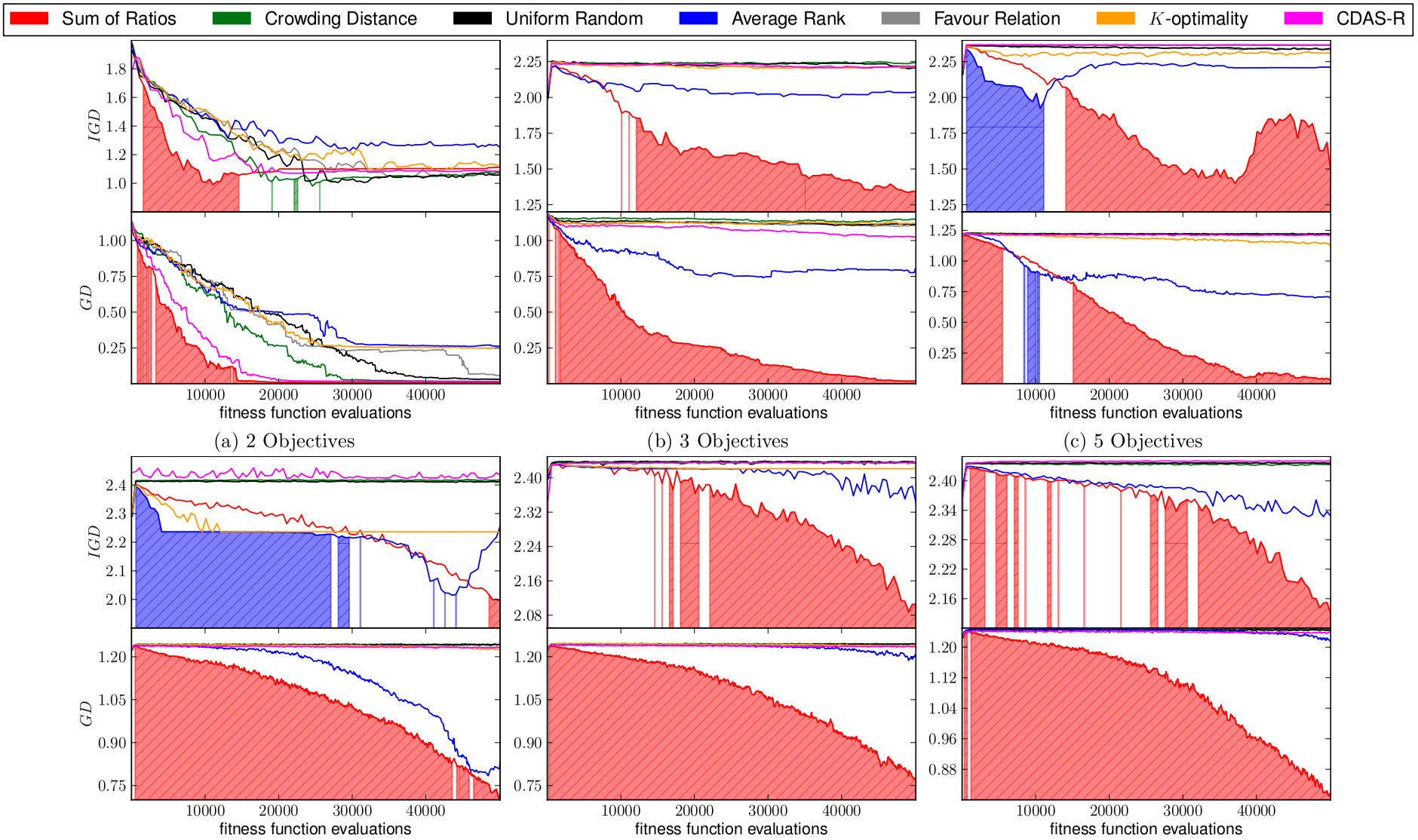
Figure 3.3.: Selection on DTLZ2: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).



(a) 2 Objectives (b) 3 Objectives (c) 5 Objectives

(d) 10 Objectives (e) 15 Objectives (f) 20 Objectives

Figure 3.4.: Selection on DTLZ3: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).



(a) 2 Objectives (b) 3 Objectives (c) 5 Objectives

(d) 10 Objectives (e) 15 Objectives (f) 20 Objectives

Figure 3.5.: Selection on DTLZ4: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).

3.4. Discussion

In relation to selection approaches, SR tends to perform the best, however we note that AR performs dramatically better on DTLZ3 for 10+ objectives (although interestingly not on DTLZ1, which has a similar deceptive front structure, but whose front *shape* is different). Corne and Knowles (2007) concluded from their experiments that AR was better than SR (using a GA on a combinatorial problem), however the results here would indicate a less-clear cut ordering, and a dependence on problem type, front shape and number of objectives – with AR tending to do better on lower order many objectives problems.

Here we have seen the strong effect of directly altering the *selection* processes. In the upcoming chapter we will investigate how altering the solutions that are within the archive can effect the the section and thereby the swarm. For this we will keep the selection method constant.

4. The Effect of Archive Operators on Many objective PSO

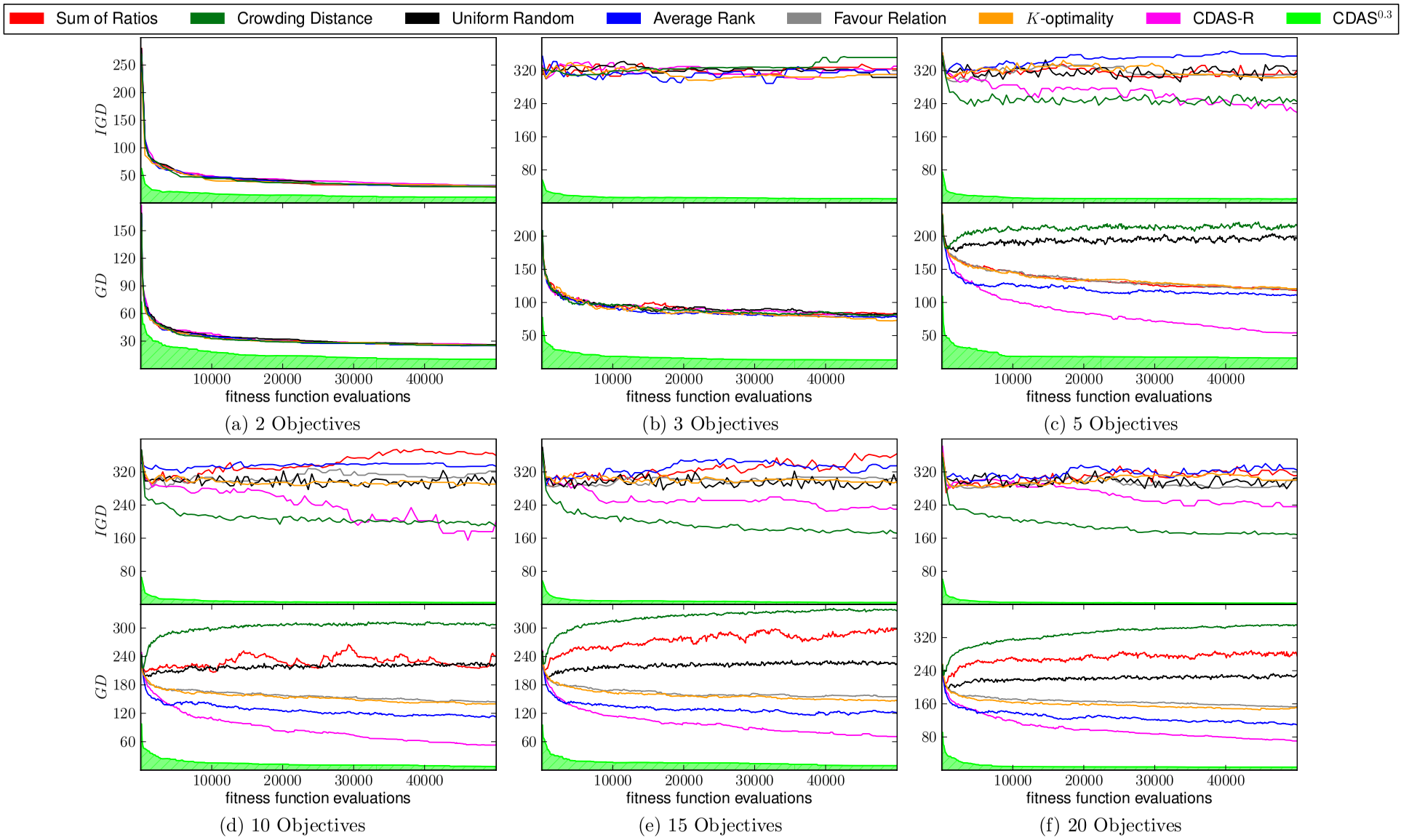
In many-objectives it becomes computationally undesirable to operate with an unconstrained archive. (See further discussion in Section 2.5.1.) In addition to this providing a large number of solutions from which the guide is draw is not beneficial to encouraging convergence. In this chapter we examine methods used to restrict an archive's size, and a method for restricting the solutions that are initially accepted into the archived by strengthening the Pareto comparison.

Here we will look at the effect of using each of the quality measures used in Chapter 3 for *archive maintenance*, and keep the design of the optimisers consistent apart from this variation, allowing us to isolate its effect.

4.1. Experimental Design

In this set of experiments we use the MOPSO is as described as Chapter 3 however, the selection is performed at random from the non-dominated guide sets maintained. *Entry* into the guide sets however is now determined by one of eight protocols: Favour Relation (FR see section2.6.1), K-optimality (KO see section 2.6.2), CDAS-R (see section 3.1.1), Crowding Distance (CD see section 2.6.3, Average Ranking (AR see section 2.6.4), Sum of Ratios (SR see section 2.6.5), Controlling Dominance Area of Solutions (with $\mathbf{s} = \mathbf{0.3}$ we denote this as CDAS$^{\mathbf{0.3}}$), and the baseline of random truncation of a non-dominated set. For the first six measures when the non-dominated sets breach the capacity limits, their contents are ranked by the quality measure, and the top 100 ranked solutions are kept (the others are iteratively discarded[1]). For CDAS$^{\mathbf{0.3}}$ (CDAS with $s = 0.3$) the archive only contains those solutions which are non-dominated under the CDAS transformation when $\mathbf{s} = \mathbf{0.3}$, if the set exceeds 100 elements, then it is truncated by random removal.

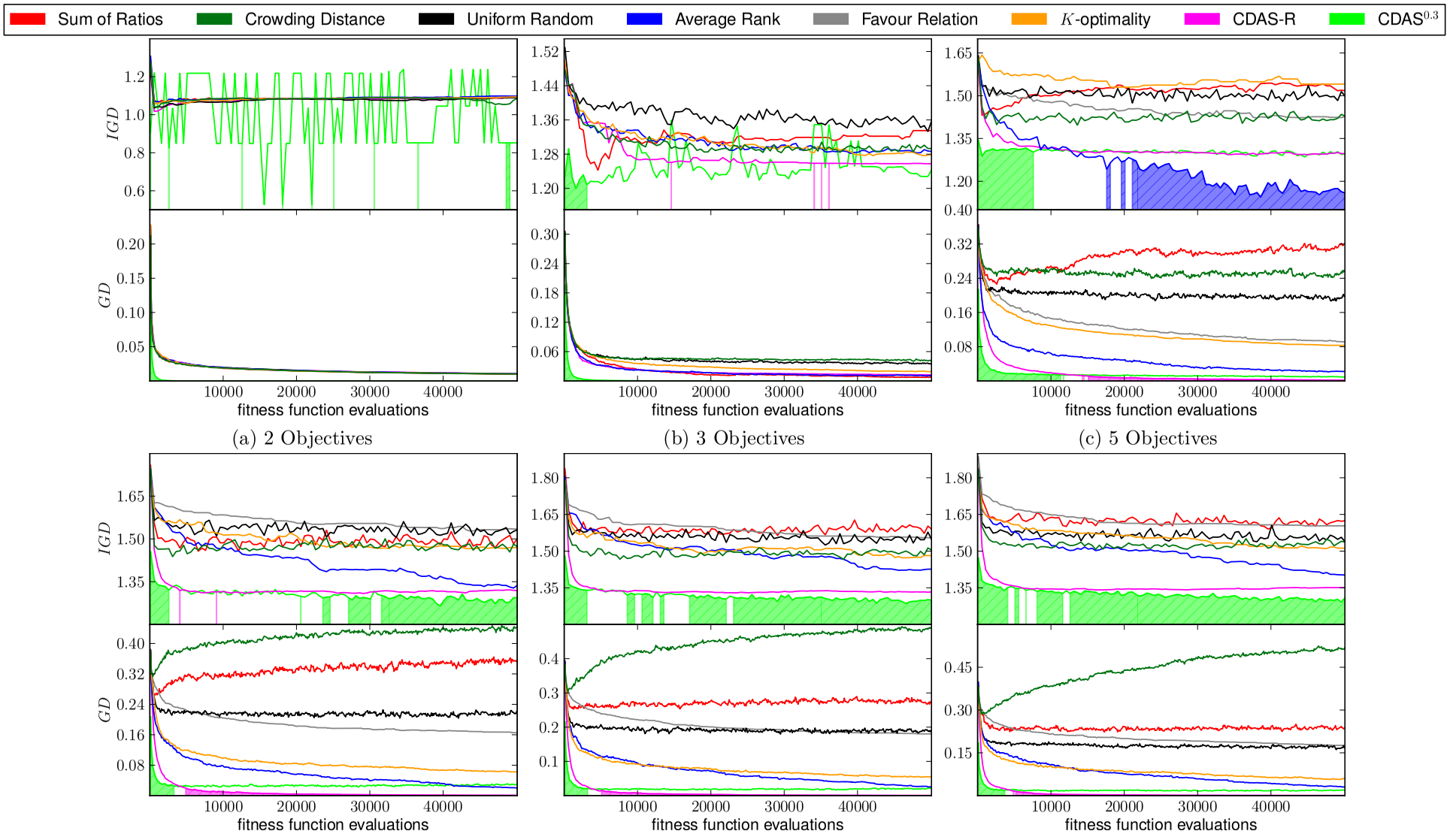
[1]Solutions must often be discarded one-by-one as for most of the quality measures described their value is affected by the set membership, and thus must be recomputed each time.



(a) 2 Objectives (b) 3 Objectives (c) 5 Objectives

(d) 10 Objectives (e) 15 Objectives (f) 20 Objectives

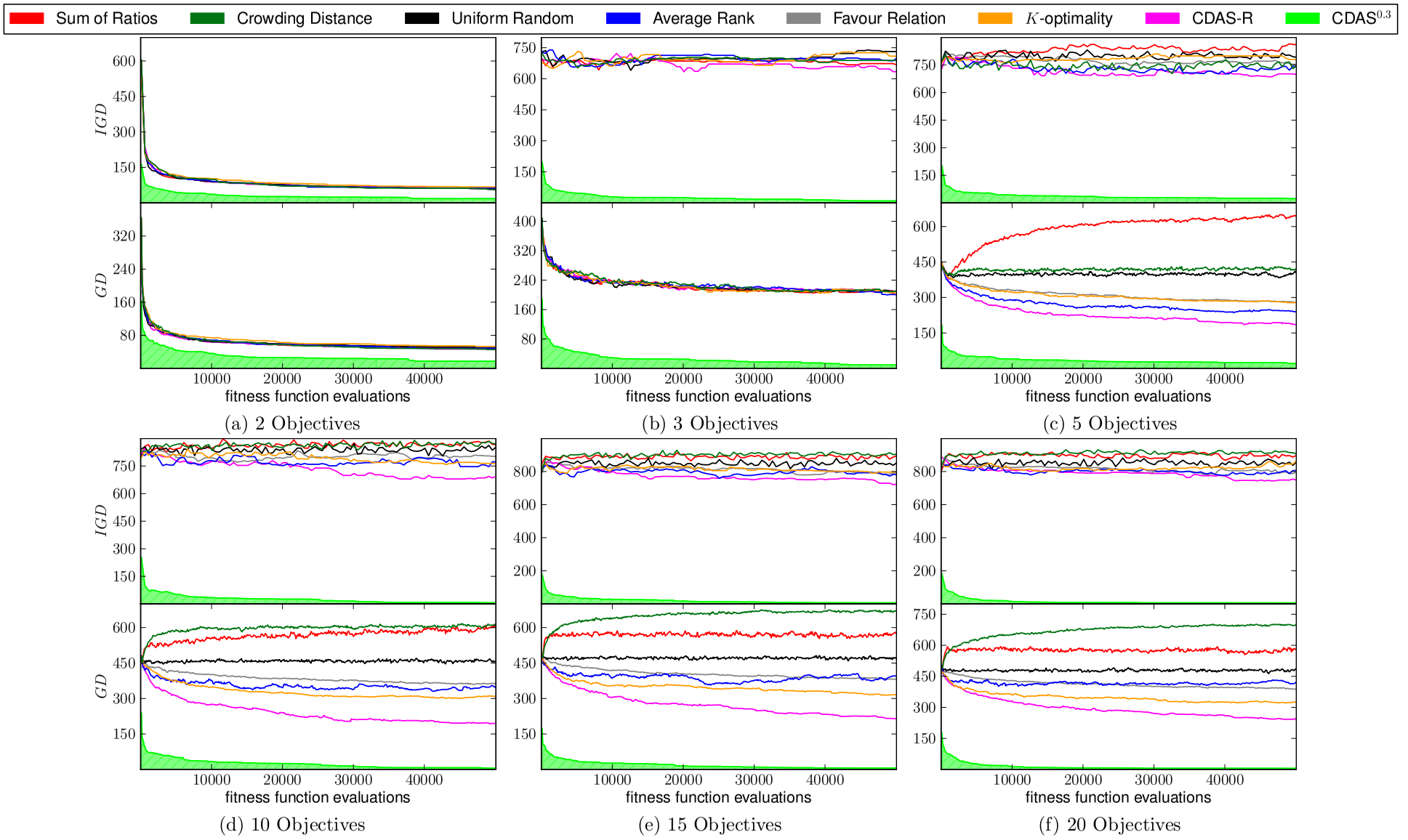
Figure 4.1.: DTLZ1 Archive maintenance: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).



(a) 2 Objectives (b) 3 Objectives (c) 5 Objectives

(d) 10 Objectives (e) 15 Objectives (f) 20 Objectives

Figure 4.2.: DTLZ2 Archive maintenance: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).



(a) 2 Objectives (b) 3 Objectives (c) 5 Objectives

(d) 10 Objectives (e) 15 Objectives (f) 20 Objectives

Figure 4.3.: DTLZ3 Archive maintenance: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).

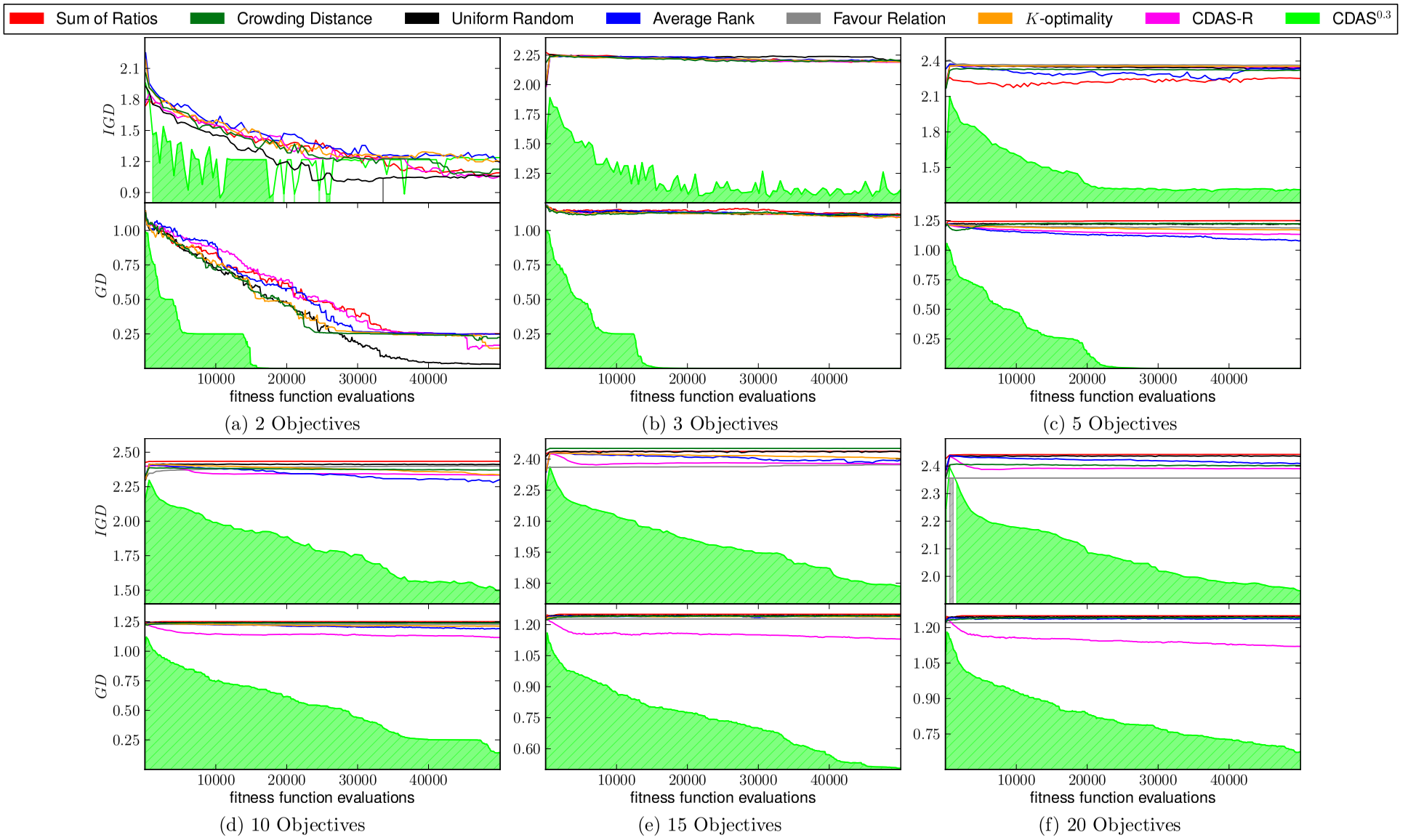


Figure 4.4.: DTLZ4 Archive maintenance: Median IGD and GD values over 30 runs plotted for each method. The shaded area underneath a method indicates that it is significantly better over the range of the shaded region compared to *all* other methods (using pairwise comparisons using the non-parametric Mann-Whitney U test, at the 5% level).

4.2. Analysis of results

In Figures 4.1,4.2,4.3 & 4.4 results are presented in same the fashion as in chapter 3. The analysis of the maintenance approaches is much simpler than those for selection. CDAS$^{0.3}$ is seen to perform significantly and substantially better than all seven other approaches across DTLZ1, 3 and 4, and to rapidly converge for all problems bar DTLZ4, where the convergence tends to take longer. For DTLZ2 although the IGD value is seen to be significantly better for 10, 15 and 20 objectives, it loses out to CDAS-R on the GD measure. We note that there appears to be a limiting value apparent for CDAS$^{0.3}$ across all of the problems – e.g., on DTLZ2 one can see that the IGD floors at about 1.35 across *all* objective cardinalities. The reason for this, is due to the mapping, as shown in Figure 3.1a, where when solutions are found in the extremities of a convex $\mathcal{F}$, those in the centre are not non-dominated in the CDAS projection and therefore not stored when the values in $\mathbf{s}$ are small. As such, there is a limit on the IGD values that can be obtained (although not the GD). For DTLZ1, although the deceptive fronts are linear, CDAS still is seen to converge, as once one solution is discovered on a lower front than currently stored, a much larger number of the previous front will be discarded even if not Pareto dominated, due to the mapping of CDAS. This is supported by Figure 4.5, which shows the median global guide archive size as the MOPSO optimiser proceeds when using CDAS$^{0.3}$ archive maintenance. For DTLZ1, the archive size does not reach 100 at all with 5-objectives, and still takes quite a few generations to reach when in higher dimensions. Figure 4.5 also indicates another of the drivers of the CDAS$^{0.3}$ convergence, as we can see that for all bar DTLZ2, the archive is not truncated until later in the run, or not at all – therefore there is a persistent convergence pressure when using CDAS$^{0.3}$ from both the global selection and the personal guide selection (that is, solutions will persist from one generation to the next unless they are dominated under CDAS mapping, rather than removal due to storage constraints). For the other archiving maintenance methods the maximum size is typically reached within three generations (i.e. 400 function evaluations). CDAS-R does not do as well as CDAS$^{0.3}$ for archive maintenance, however like the other methods CDAS-R puts a rank on the non-dominated solutions found, and limits the guide archives to the 100 best of these, so there is not the same degree of convergence pressure as with CDAS$^{0.3}$ (i.e. the archive fills rapidly and requires truncation).

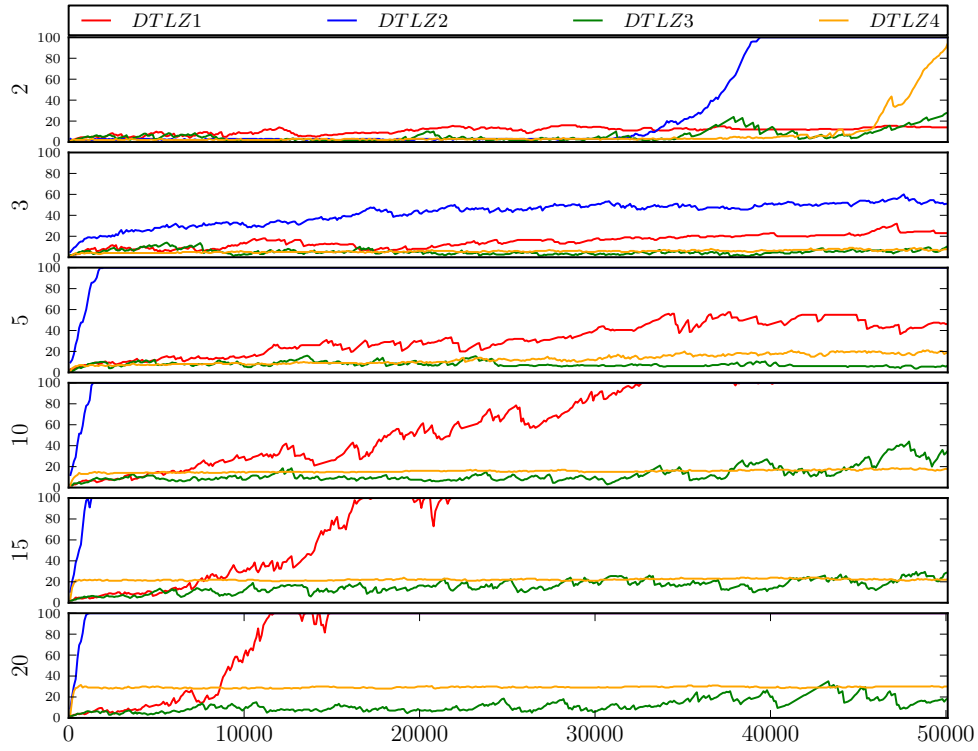


Figure 4.5.: Median archive size (y-axis) for CDAS$^{0.3}$ on different objectives (2 objective top, down to 20 objective bottem) over time (x-axis in fittness function evaluations).

4.3. Discussion

The general results with CDAS are seen to be in keeping with other recent work in the area (de Carvalho and Pozo, 2011), which use a CDAS-based MOPSO on DTLZ2 and DTLZ4, and compare it to a hybrid MOPSO combining AR and CD. We find here that CDAS as used for guide maintenance provides significantly better results across both the test functions, and the ranges of objective numbers we have assessed. Its only apparent weakness is manifest in DTLZ2. However, as we have discussed above, the reason for this is due to transformation of the objective space. CDAS with low values in $\mathbf{s}$ promotes convergence to the extremities of the Pareto front, but once the solutions stored are in the vicinity of $\mathcal{F}$, the centre of $\mathcal{F}$ is always projected to dominated locations in the new mapping when the front is convex (which puts a floor on the IGD achievable). With respect to the GD – we see that CDAS-R actually achieves slightly lower values than CDAS$^{0.3}$, and we conject that this may be due to it storing a greater range of guides – and therefore when these are in a reasonably converged position, the chances of getting

even closer solutions anywhere across the front is improved (whereas CDAS$^{0.3}$ will only accept those on the extremity).

The results, when compared to those in chapter 3, indicate that archive maintenance has a lesser effect on the final quality of solutions returned, in terms of IGD and GD, compared with the effect of the selector choice *apart* from when using CDAS for archive maintenance, where the IGD and GD values are significantly lower than any selector results (barring DTLZ2).

Based upon the results here, we recommend those applying many-objective particle swarm optimisers to strongly consider used CDAS-based archiving approaches. Leading on from this in Chapters 5 & 6 we will take a more in-depth investigation into Controlling Dominance Area of Solution and the newer parameter free derivative, Self Controlling Dominance Area of Solution.

5. Control of Dominance Area of Solutions

As seen throughout the literature (Ishibuchi et al., 2008) and when considering the plain PSO in chapters 3 & 4 many popular multi-objective optimisers are unable to effectively provide convergence on many objective problems, with the root cause often being their use of Pareto dominance quality measures, which do not discriminate effectively in high dimensions. An approach that has seen strong results on many objective problems is the controlling dominance area of solutions (CDAS) method and its newer adaptation Self-CDAS.

CDAS acts by projecting the objective criteria associated with solutions into a different space (of the same dimensionality), with Pareto quality assessments occurring in this new mapping. By altering the parameters of this mapping, the system is able to increase and decrease the selection pressure provided by typical Pareto ranking. Here we investigate why controlling dominance area of solutions is so successful on many objective problems and how it performs with a variety of different front topologies. Focusing our investigation into the effect that front shape has upon the operator. Later in chapter 6 we will examine how CDAS performs on a larger range of problems.

5.1. Introduction

Pareto dominance has, since the conception of multi-objective optimisation, maintained a strong position as the leading approach in most multi, and now many, objective optimisation algorithms. However Pareto dominance alone has been found lacking and unable to maintain good convergence whilst encouraging diversity when the number of objectives increases beyond four. To assist algorithms dealing with many objectives, various finer grained ranking based methodologies have been proposed. Some of these attempt to alter the set of solutions available for future selection by either strengthening or weakening the

criteria surrounding the Pareto operation.

These approaches typically alter the region which is dominated by a each solution. This modification ranges from adding a hypercube surrounding each solution, dominating all other solutions within, to adjusting the angle subtended from the axes to the solution that bounds the dominated region. Controlling dominance area of solutions (CDAS) uses the second of these, namely modifying the angles bounding the dominated region. While originally developed for multi objective optimisation (2 to 3 objectives) CDAS has been effectively used on many objective problems. The key difference in its application to many objectives is the switch from weakening the Pareto dominance to strongly strengthening.

By strengthening Pareto dominance a subset of the Pareto *mutually non-dominated* solutions are accepted. With low numbers of objectives, as those for which Pareto was originally developed, removing solutions from the first *mutually non-dominating* front is not considered to be highly beneficial. However as the number objectives increases so does the number of *mutually non-dominating solutions*, especially those in the first front, giving rise to difficulties with traditional multi-objective optimisers. With a smaller set of solutions to select future generations from the hope is to encourage convergence which otherwise for many objective problems has been a difficulty.

Alternatively, there have been various methods proposed which make use of a decision-maker's prior preference to particular regions or region in objective space (Zitzler and Künzli, 2004; Deb et al., 2006b). The reference point method (Deb et al., 2006b) and light beam search (Deb and Kumar, 2007) have been the most popular, with their successful application to a range of many objective problems (Wickramasinghe and Li, 2009). However, as these approaches are concerned with a specific region of objective space, they do not promote coverage across the entire front, and therefore have limited use when the feasible objective space properties are not known and/or there is no prior preference knowledge.

To provide a greater convergence, and/or even distribution of solutions, a range of methods have been proposed which impose an additional ordering upon the Pareto *mutually non-dominating* solutions to provide a finer degree of granularity when selecting or removing a solution from an archive/population – that is, to provide a rank order on solutions (the Pareto relationship only providing a partial order) see sections 2.6.4, 2.6.1. Alternately as discussed in Ishibuchi et al. (2008) the Pareto dominance method can be altered to reduce the number of solutions which are *mutually non-dominating*. Popular among these methods are α-dominance (Ikeda et al., 2001), ϵ-dominance (Laumanns et al.,

2002) and Controlling Dominance Area Of Solutions (Sato et al., 2007b). Each increases the region which a solution dominates, thereby dominating solutions which under Pareto would be *mutually non-dominating*. This chapter is an investigation of how Controlling Dominance Area Of Solutions of how the selection pressure provided is effected by the *shape* and *spacing* of the *mutually non-dominating* set.

5.2. Preferences of CDAS and S-CDAS

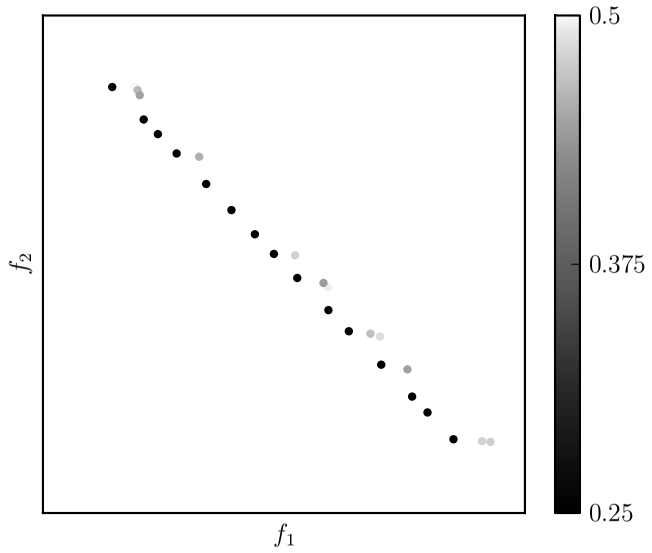
As both CDAS and S-CDAS operate upon the entire population, so that domination or ranks are functions of the locations of all the solutions in objective space respectively, it is of interest how these algorithms discriminate between solutions - considering different distributions of solutions in objective space. The investigation will take a graphical approach analysing the preference CDAS and S-CDAS have for solutions within different populations in 2 and 3 dimensions. To show multiple s values for CDAS we have combined the results together giving a rank indicating the smallest s values for which a solution is still *mutually non-dominating*, this is the same rank as the CDAS-R ranking introduced in Section 2.6.6.

Our investigation will cover three primitive Pareto *mutually non-dominating* front types under minimisation. Figure 5.1 is the first of these fronts, a randomly uniform sample of solutions[1] from a simplex. In Figures 5.1a & 5.1b additional Pareto *mutually non-dominating* solutions set back from the simplex have been added. The rank of solutions for CDAS and S-CDAS is indicated by shade, for CDAS the corresponding s values can be seen upon the legend on the right of each figure.

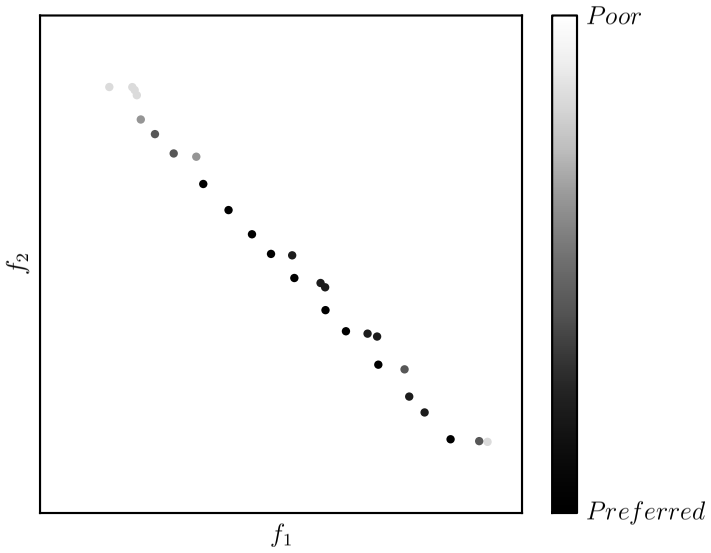
For S-CDAS we consider two different embeddings into optimisers: elitist restriction (ER), keeping the n best solutions, where n is the maximum size for the population/archive; elitist selection (ES), keeping only the best solutions. These different applications of S-CDAS provide two perspectives for these graphs: ER effectively cuts off the population to keep the n darkest solutions; ES keeps *only* the darkest solutions, therefore this will normally be a much smaller set of solutions than those retained with ER (assuming a typical population/archive size of 100 (Garza-Fabre et al., 2010; Leong and Yen, 2006)).

Considering the 2 dimensional plots, Figures 5.1a & 5.1b, the most striking feature is how even with a medium s value CDAS ignores the additional offset solutions, keeping

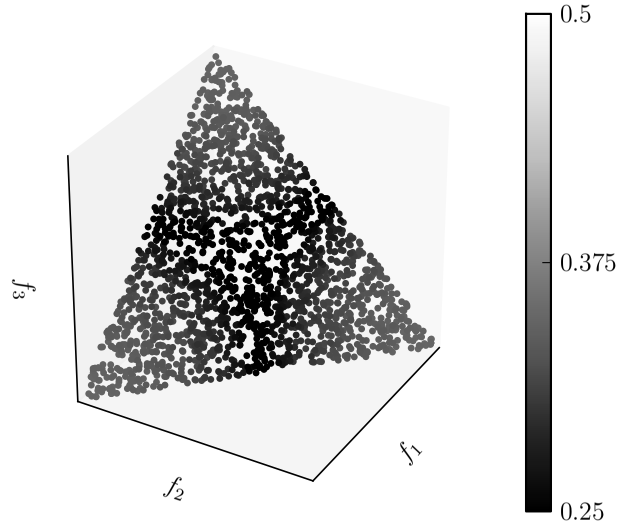
[1] Down sampled to create a more evenly distributed sample using the largest bounding hypercube method see section 2.6.3



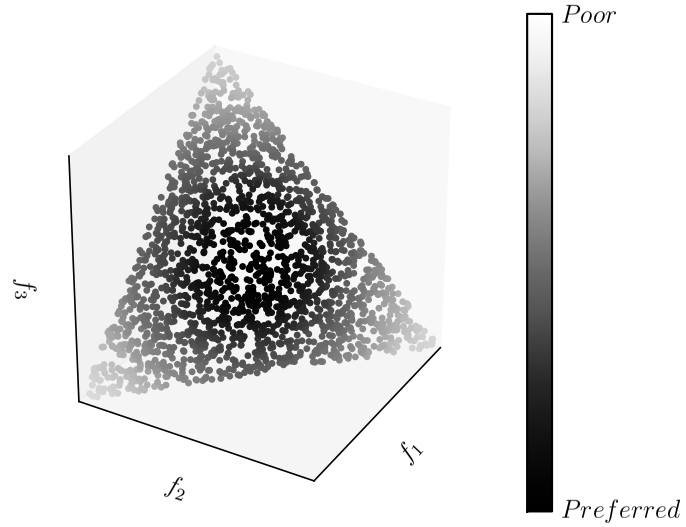
(a) Effect of CDAS upon a simplex 2d front



(b) Effect of S-CDAS upon a simplex 2d front



(c) Effect of CDAS upon a simplex 3d front



(d) Effect of S-CDAS upon a simplex 3d front

Figure 5.1.: rankings upon a simplex front. The lighter a solution the sooner it will be lost as the selection pressure is increased.

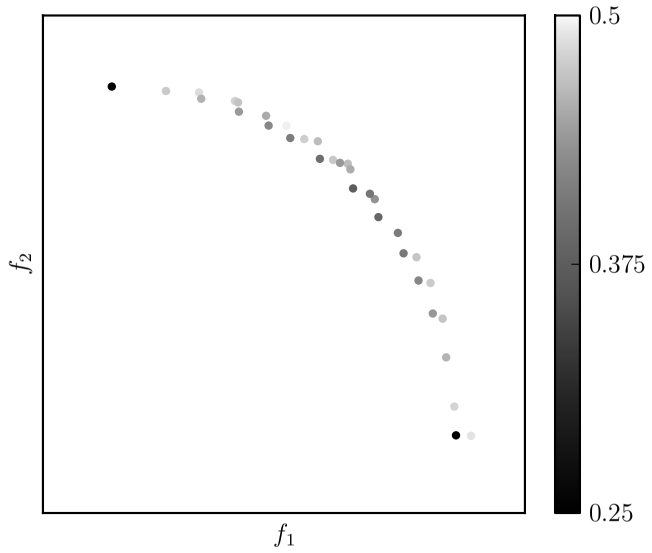
all the solutions from the simplex even at the lowest limit of s. While S-CDAS struggles to distinguish between these additions solutions and those upon the simplex, here the Euclidean distance to the origin appears to have a much greater effect upon the rank of the solutions. For ER and possibly ES some of the additional solutions will be included in the population, whereas for CDAS with $s \leq 0.4$ these solutions are excluded. In Figures 5.1c & 5.1d a 3-dimensional simplex is used, the results between S-CDAS and CDAS here are similar.

Our second front shape is the positive sector of a hypersphere centred at the origin, Figure 5.2. This convex hull is common, as the optimal set, among multi/many-objective test problems (Huband et al., 2006). Here the behaviour can be seen, in Figures 5.2c & 5.2d, where CDAS general prefers the center with some extreme 'edge' solutions having even greater favour - while S-CDAS likes the edge with the center having the lowest preference. For S-CDAS with the ES implementation this means that only the solutions upon the very edge are preserved in comparison, CDAS with a medium s value prefers both the extreme 'edge' solutions and some central solutions. From both 2D and 3D plots we can see that for CDAS to result in a small set a *very* strong s value must be chosen, $s \leq 0.325$.

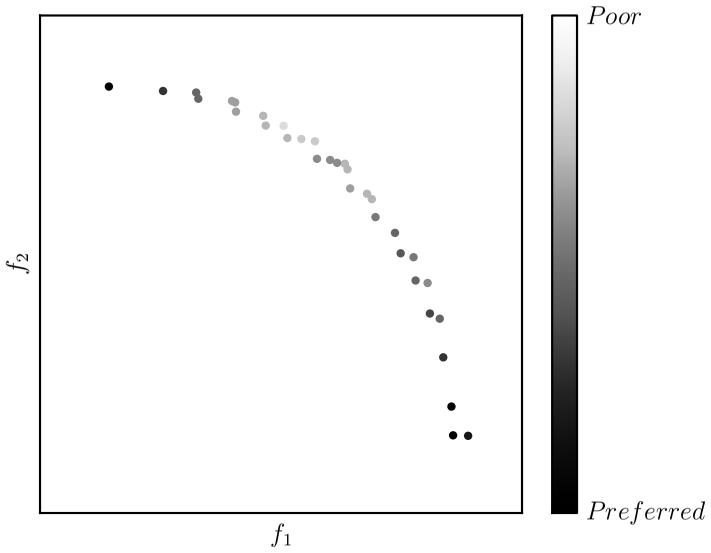
Figure 5.3, the negative section of a hyper-sphere, shows a convex hull of solutions. As with the other front shapes covered so far, in Figures 5.1 & 5.2, CDAS maintains a preference for central region. In this case comparing CDAS to S-CDAS ranks shows how given the correct choice of s CDAS could select similar solutions to ES and ER versions of S-CDAS. However this does not include the additional Pareto solutions that are set back from the concave section, in Figures 5.3a & 5.3b, which even with a high s value CDAS will discard these - S-CDAS on the other hand appears to pay little attention to whether the solutions are upon the concave section.

These first three shapes are all *very* primitive, considering that during an optimisation the shape of the population in objective space can form any number of distributions (which are free to change from generation to generation), even in cases where the optimal set lies upon one of these primitives. So to further understand how the optimiser chooses solutions we must consider more complicated shapes.

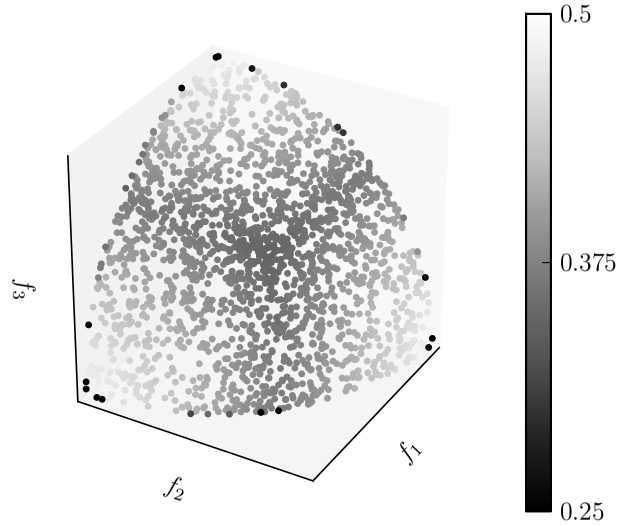
Combining the convex and concave shapes investigated above, Figure 5.4 shows that for both 2 & 3 dimensions S-CDAS and CDAS differ - CDAS maintains a liking towards the approaching concave section while also taking some solutions, depending on s, from



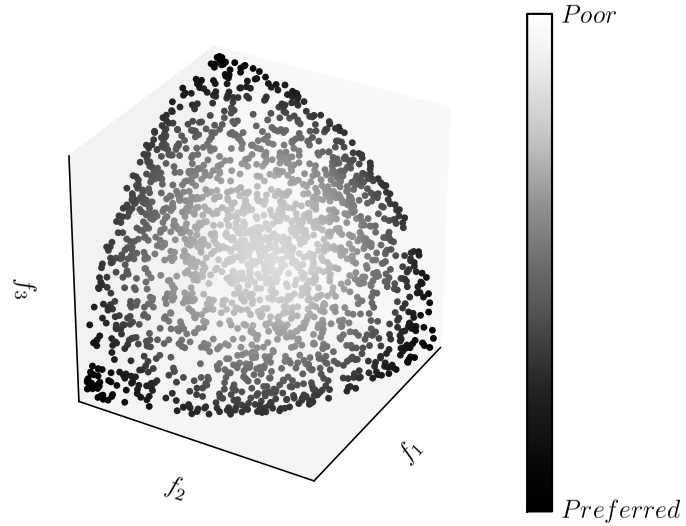
(a) CDAS convex front



(b) S-CDAS convex front

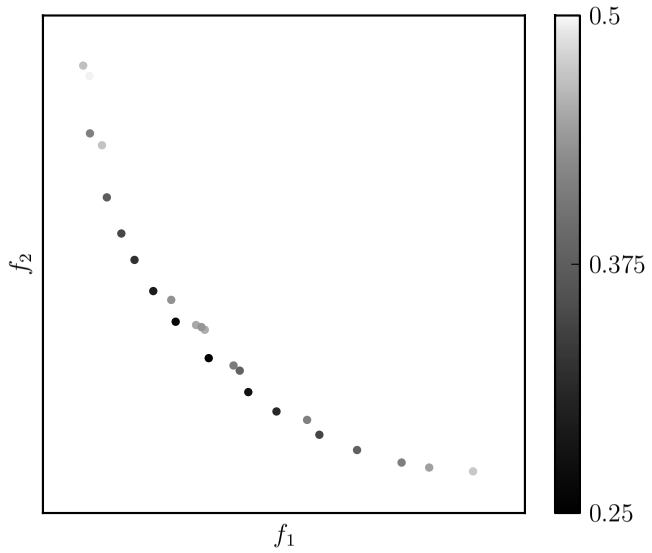


(c) CDAS convex front

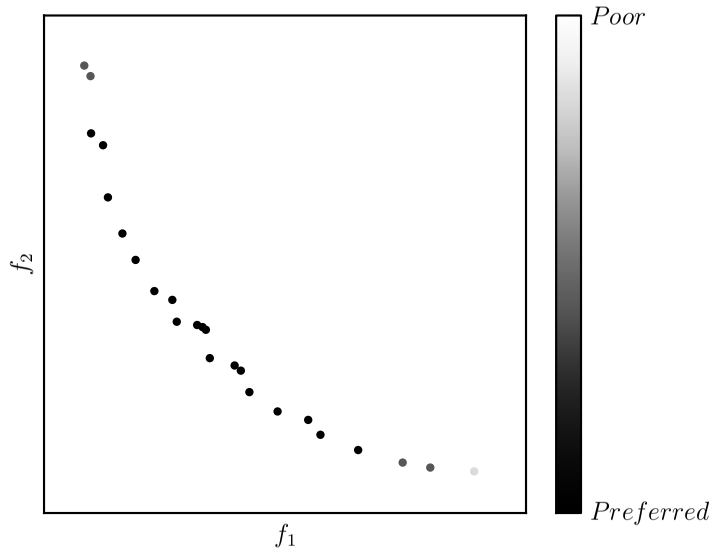


(d) S-CDAS convex front

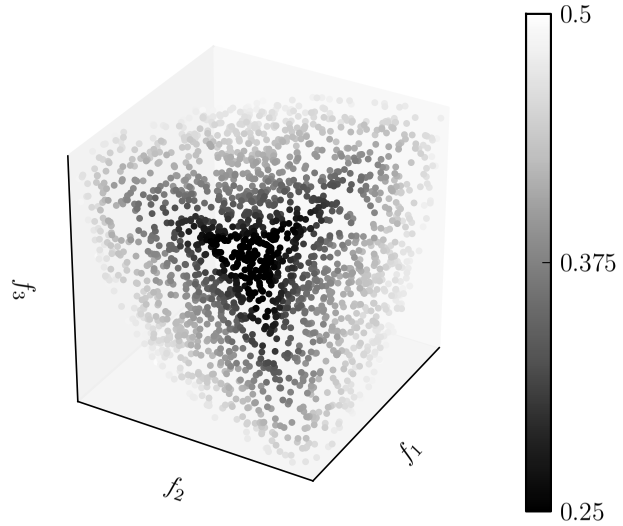
Figure 5.2.: Effect of CDAS (left) and S-CDAS (right) upon a convex front. This is constructed as the positive section of the hyper-sphere.



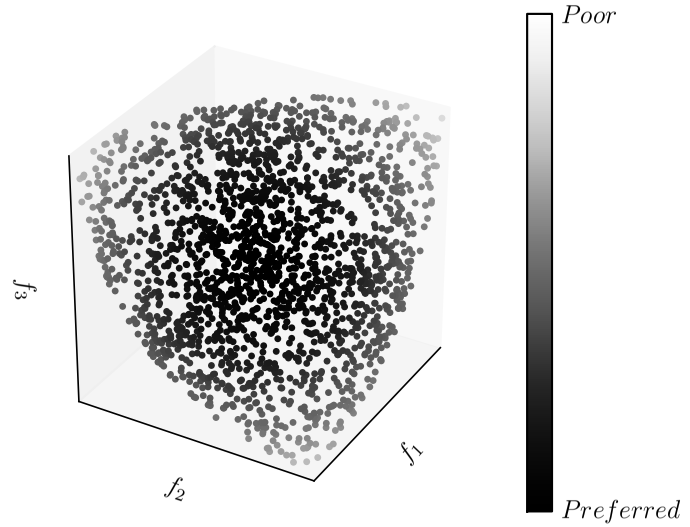
(a) CDAS concave front



(b) S-CDAS concave front

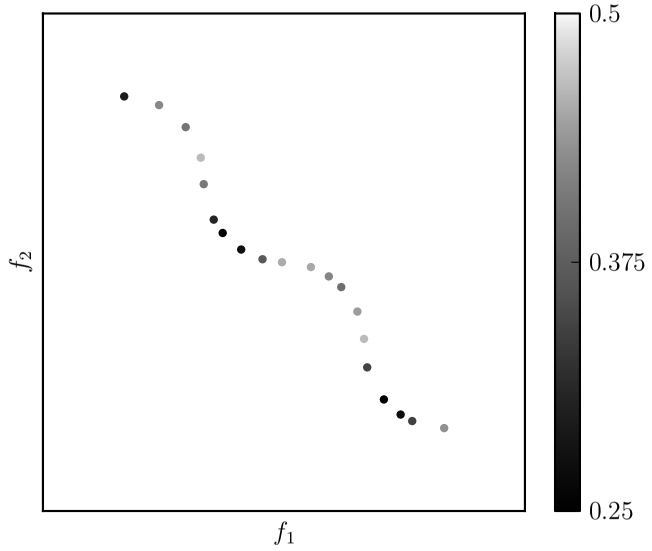


(c) CDAS concave front

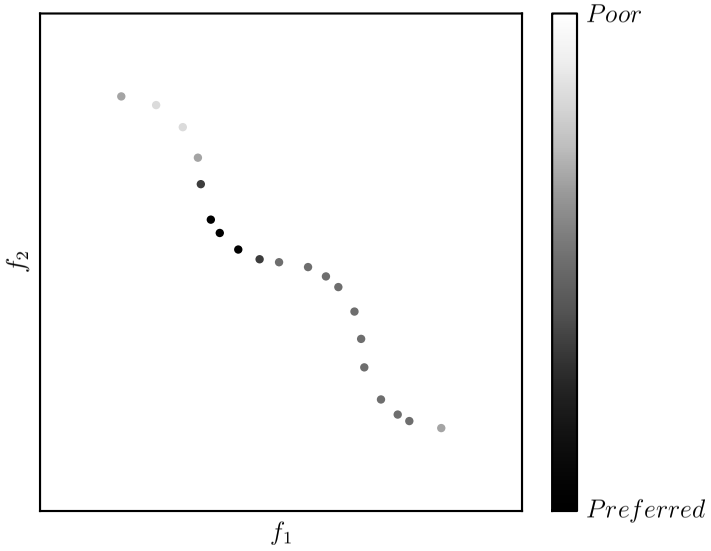


(d) S-CDAS concave front

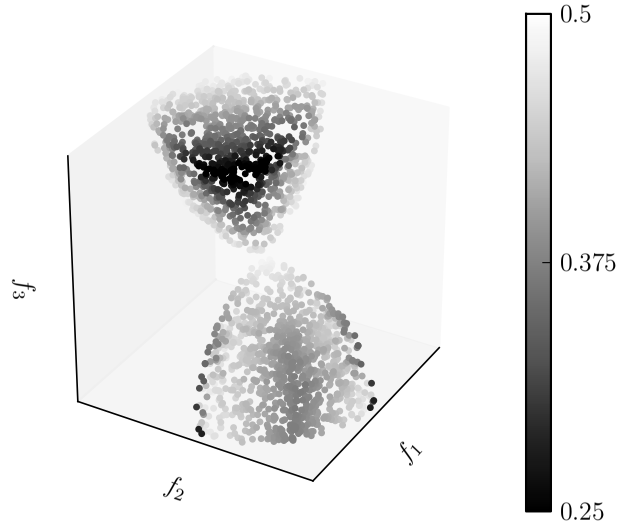
Figure 5.3.: Effect of CDAS (left) and S-CDAS (right) upon a concave front. The negative sector of a hyper-sphere translated into the positive sector.



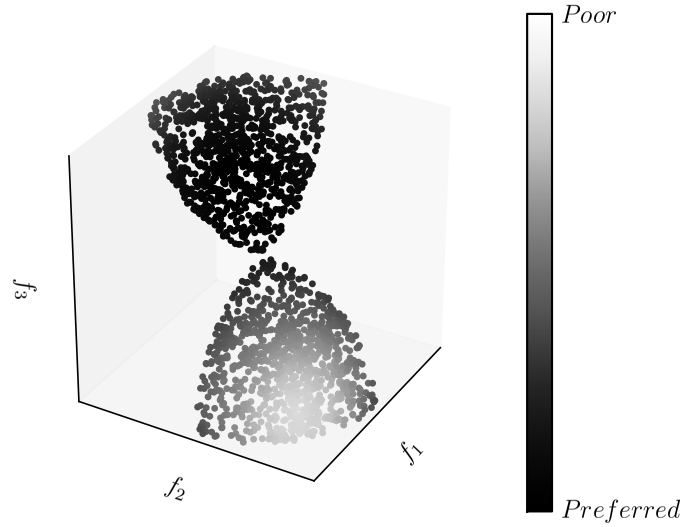
(a) CDAS mixed front



(b) S-CDAS mixed front



(c) CDAS mixed front



(d) S-CDAS mixed front

Figure 5.4.: Combination of convex and concave spherical section used to create the surface.

the center of *each* convex section. S-CDAS however prefers the concave hull closest to the origin over all others, and considers solutions in the convex section as worse the further they are from the origin. For both algorithms combining the fronts together has not significantly altered how they perform on the individual sections, convex/concave.

For CDAS and S-CDAS to be to considered for application to real work problems we must discuss how they manage on other less primitive shapes. We will consider the previous primitive fronts but scaled on their first objective by a factor of 2.5. For S-CDAS φ is computed for each objective independently, and is influenced by the range of the objective values on that objective. As S-CDAS does not need to set an s value, it would be expected to be better at performing on fronts with different objective ranges.

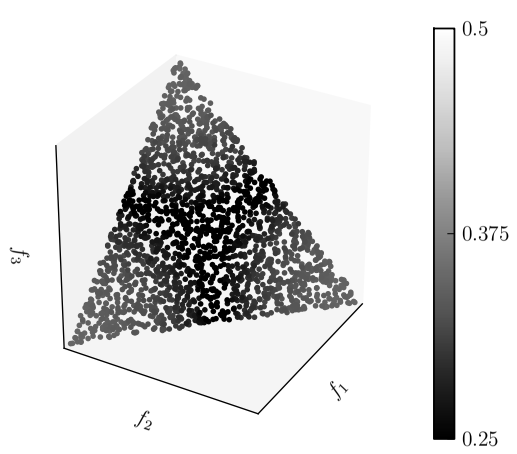
The additional question we ask is how these algorithms perform upon subsets of the above fronts, Where sections have been removed. For this we plot the difference in rank between the front will all solutions and that with some removed.

For the simplex, shown in figures 5.5, with sections removed neither of the two algorithms result in a large difference in ranking however, CDAS has some impact preferring keeping the new edges for even stronger s values. S-CDAS has a small impact which results in some of the solutions in the disconnected section being lost under moderate selection.

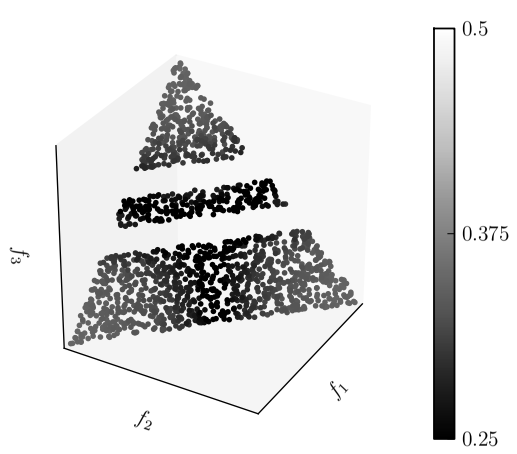
Many problems contain discontinuities, and even for those that do not during optimisation the current approximated *mutually non-dominating* set may have large sections of the objective space unaccounted-for. In figures 5.6 we show how the rank of solutions changes when sections of the front are removed. This has little affect on CDAS, only strengthening some solutions but never weakening any. (This is as expected see section 3.1.1) S-CDAS however has a greater change, with sections both gaining and losing rankings.

5.3. Combating the differences in shape and scale

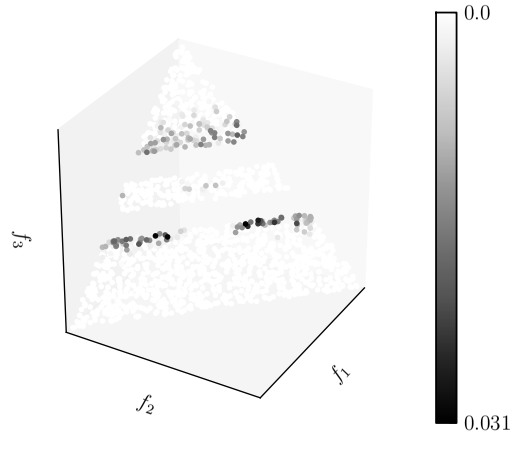
In Figures 5.8 & 5.7, the same primitive front shapes are presented as before in Figures 5.1, 5.2 & 5.3 with one alteration, the scaling of the first objective by a factor of 2.5. While it is possible within CDAS to use different s_i for each objective, having prior knowledge of the ranges on each objective cannot be assumed for many real world problems. Therefore here CDAS uses the same s values for all objectives as with the previous unscaled examples. Here the scaling has less of an effect upon S-CDAS then CDAS, however it is worth noting that there is still a some effect. This is most clearly seen when comparing the simplex shaped fronts in Figure 5.7d to Figure 5.1d. The clear preference given by both algorithms



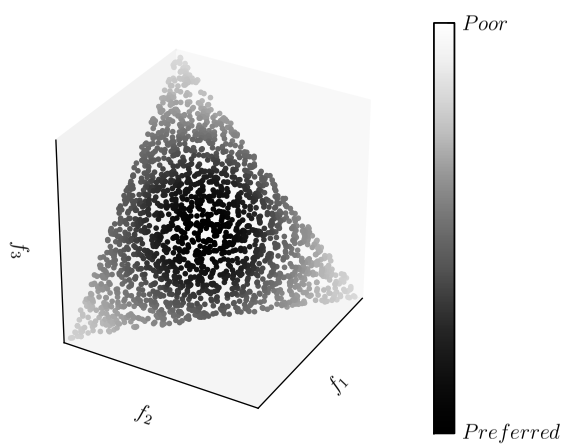
(a) CDAS ranking on simplex



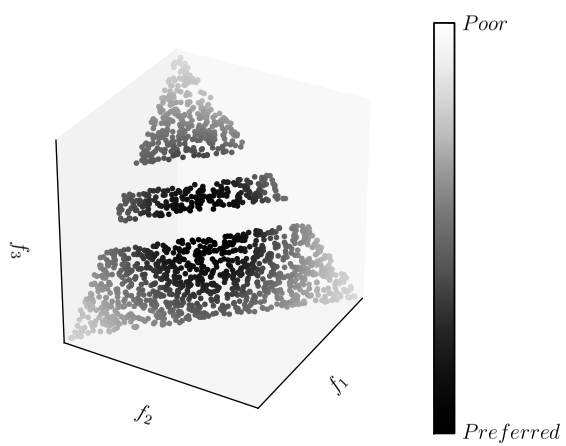
(b) CDAS ranking on simplex with sections removed



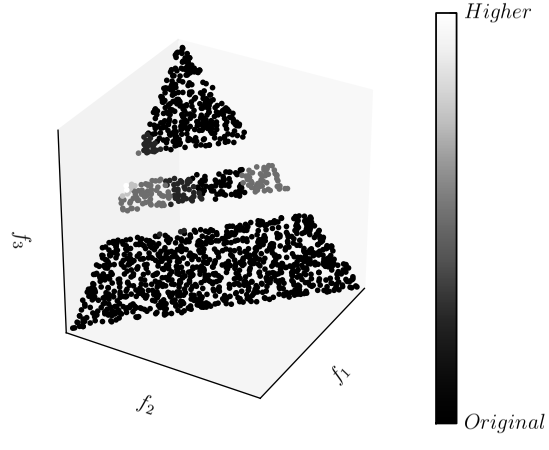
(c) Difference between 5.5b - 5.5a, darker solutions have greater s value tolerance.



(d) S-CDAS ranking on simplex

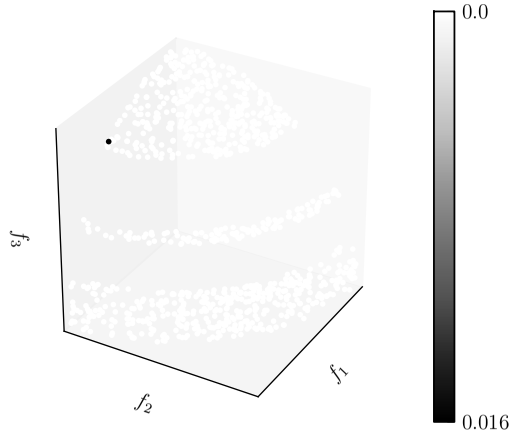


(e) S-CDAS ranking on simplex with sections removed

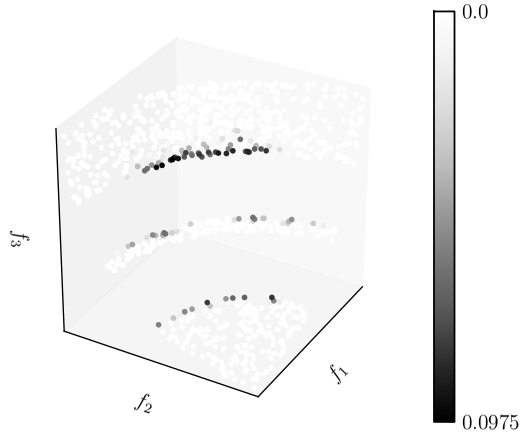


(f) Difference between 5.5e - 5.5d, higher ranks are worse for S-CDAS

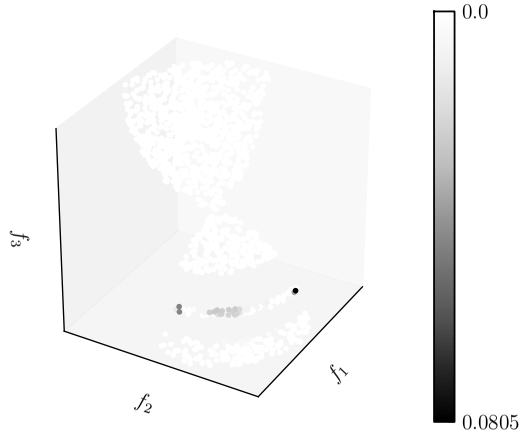
Figure 5.5.: CDAS (top row) S-CDAS (bottom), front with section removed



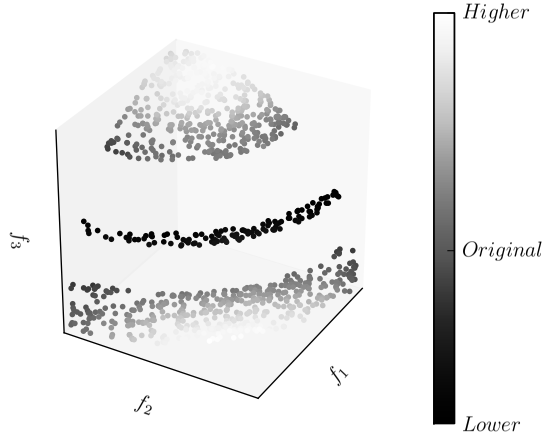
(a) CDAS Convex



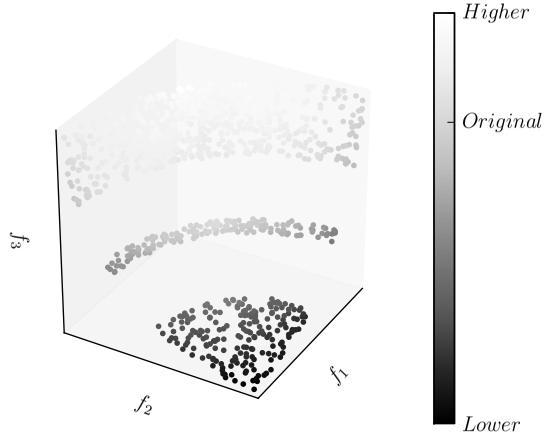
(b) CDAS Concave



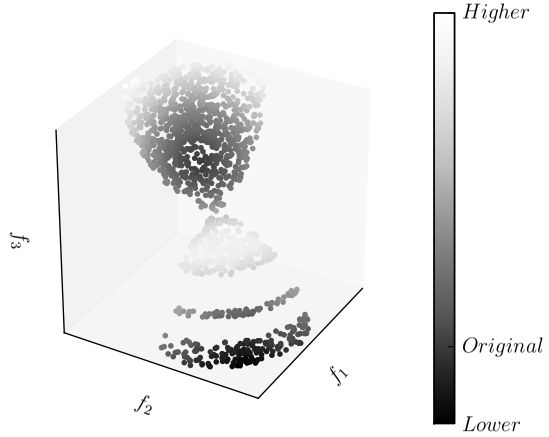
(c) CDAS Mixed



(d) S-CDAS Convex

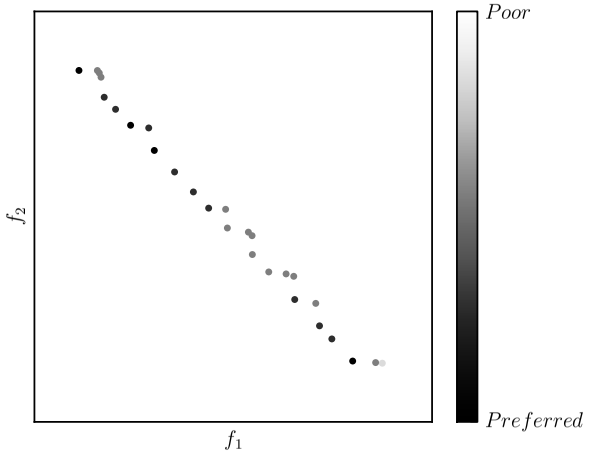


(e) S-CDAS Concave

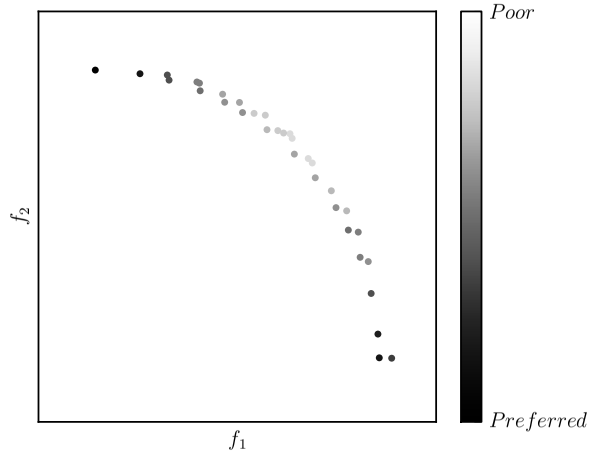


(f) S-CDAS Mixed

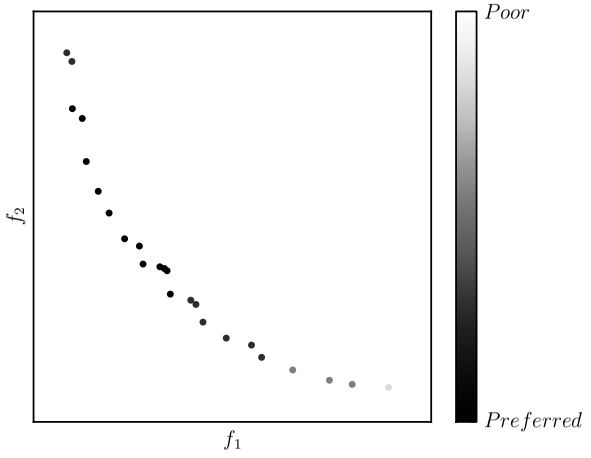
Figure 5.6.: CDAS (top row) S-CDAS (bottom), difference in ranks when sections are removed.



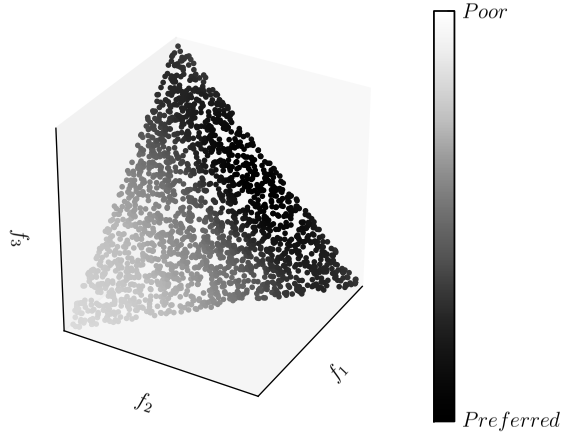
(a) S-CDAS on a scaled simplex



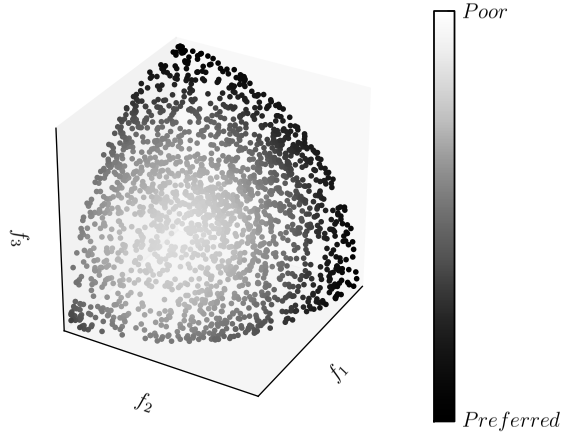
(b) S-CDAS on a scaled convex



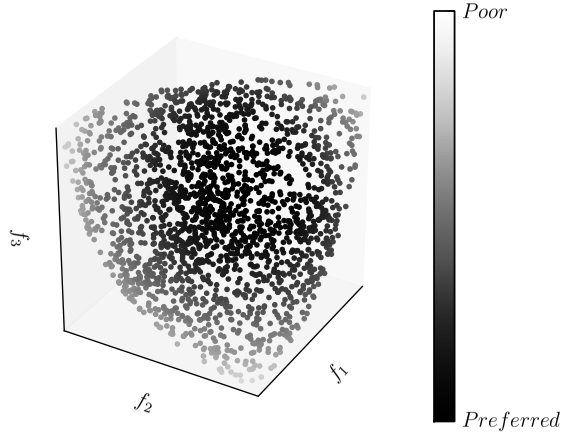
(c) S-CDAS on a scaled concave



(d) S-CDAS on a scaled simplex

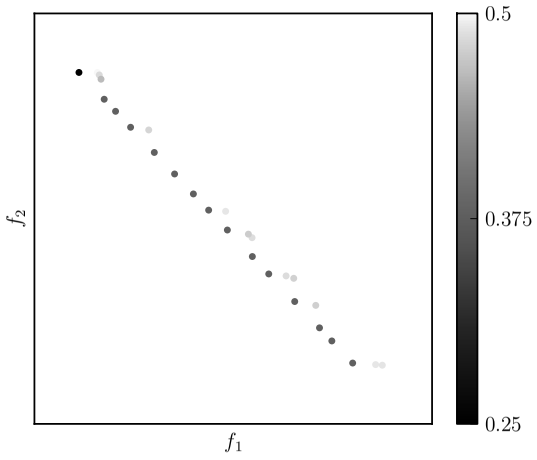


(e) S-CDAS on a scaled convex

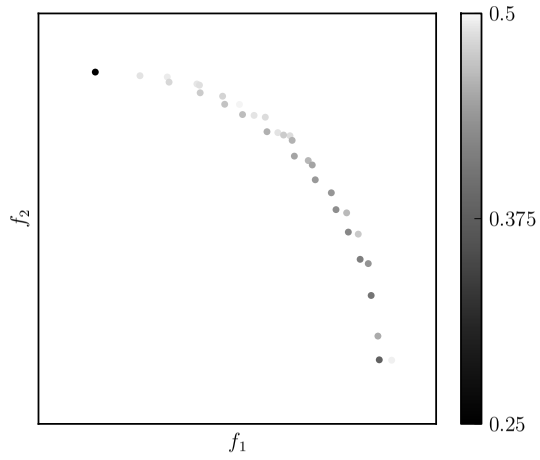


(f) S-CDAS on a scaled concave

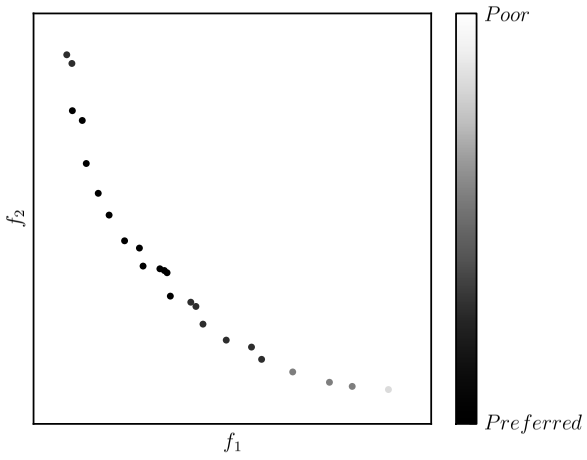
Figure 5.7.: S-CDAS ranking on scaled front shapes: with the first objective scaled by a factor of 2.5



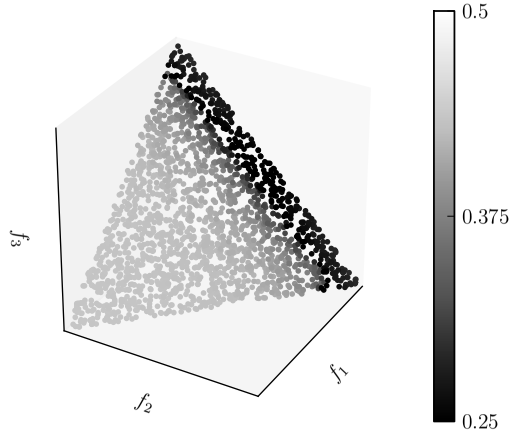
(a) CDAS on a scaled simplex



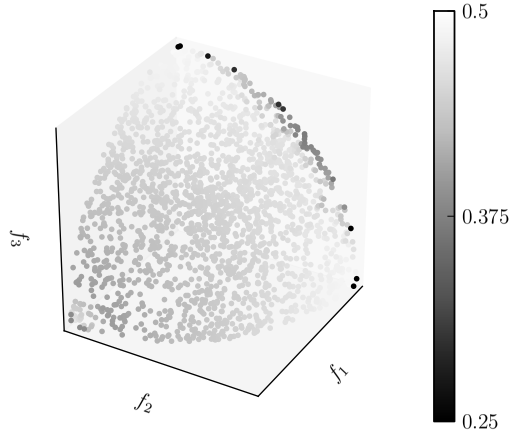
(b) CDAS on a scaled convex



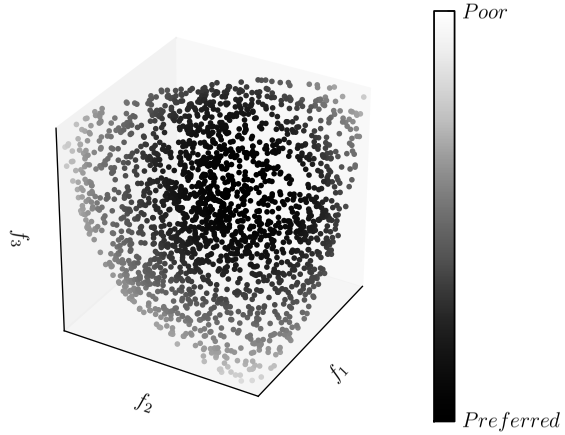
(c) CDAS on a scaled concave



(d) CDAS on a scaled simplex



(e) CDAS on a scaled convex



(f) CDAS on a scaled concave

Figure 5.8.: CDAS ranking on scaled front shapes: with the first objective scaled by a factor of 2.5

to solutions closer to the origin may lead them to only converge down in one region of the front, not maintaining good diversity.

As we have seen so far CDAS and S-CDAS are highly susceptible to the shape of the front, both in terms of primitive and any non-uniform scaling of the front. To overcome this we propose a normalisation and re-mapping methodology that aims to provide a wide range of solutions, while providing strong convergence pressure. To do so first the *mutually non-dominating* set is mapped to a pre-determined shape. Here the CDAS transform is computed, using a pre-determined s value, resulting in a new set of solutions which are *mutually non-dominating* in the transformed CDAS space. With this we aim to provide a shape independent many-objective non-parametrised Pareto strengthening method.

As the range of solutions on each objective is not fixed the first step is to re-scale these ranges similarly so that they can be effectively mapped onto one of the primitive shapes. Here we will introduce a means of re-scaling each objective independently to obtain a better distribution when mapped to a uniformly scaled primitive.

Given a set of solutions $\mathcal{P}$ they are first rescaled:

$$\mathbf{f}(\mathbf{x}) = \frac{\mathbf{f}(\mathbf{x}) - \min(\mathcal{P})}{\max(\mathcal{P}) - \min(\mathcal{P}) + \epsilon} + \epsilon \tag{5.1}$$

Where min and max are vectors of the min/max objective values across all solutions in $\mathcal{P}$ respectively. With ϵ set as a small offset value to ensure no division by zero, (e.g. $\epsilon = 0.001$). This effectively rescales on each objective such that all objective values sit in the range $[\epsilon, 1 + \epsilon)$.

To re-map the solutions to a fixed front shape they are projected down towards the origin onto the preferred shape. To project a population of solutions onto the positive segment of a hyper-sphere the objective vector is normalised to have unit length 1. To project onto a simplex the objective vector is rescaled such that the sum of the objective values is constant for all solutions (here we rescaled to 1). In the general case a ray is drawn from the solution to the origin the solution is then mapped to the point of intersection of this ray and the target surface.

5.3.1. Re-scaled CDAS

Here we will investigate how re-scaling the objectives prior to computing the CDAS transform can alter which regions of the objective space are preferred, we will compare how

CDAS performs on fonts scaled on one objective to this normalised CDAS algorithm.

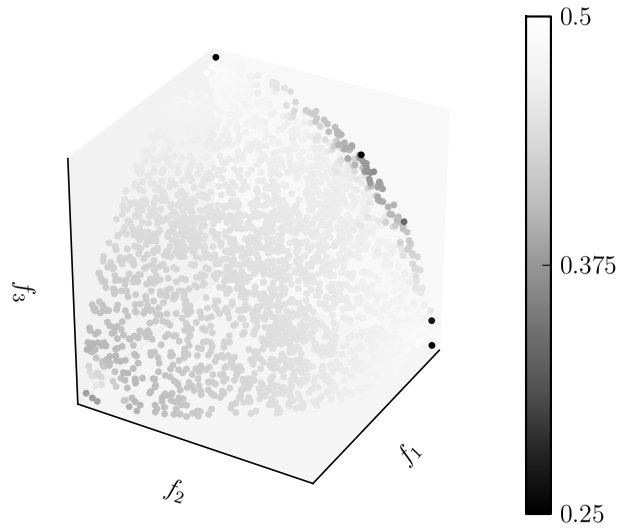
In Figure 5.9, we can see how normalising the objectives prior to computing the CDAS transform results in the same distribution of preferences as with the non-scaled front shapes seen before, see section 5.2.

The general CDAS algorithm must be adapted slightly to incorporate this normalisation, it is important that for the normalisation process only *non-dominated* solutions are considered for the maximum and minimum. Therefore solutions that are being added to the archive must be Pareto *mutually non-dominating* with those within the archive prior to any normalisation process. Following this at the end of each generation, the point at which the archive is updated for the next generation, the solution values are normalised and transformed into CDAS space where another Pareto examination takes place, only solutions which are *non-dominated* in this transformed normalised space have the chance to continue to the next generation and be used for selection. If after the CDAS process there are still more solutions in the archive than its maximum size then any number of truncation methods can be used to trim the archive. This does however lead to a reduction in efficiency in computational comparison to traditional CDAS, where all currently accepted solutions can have their transformed location in CDAS space stored and thereby save re-computing this for every generation.

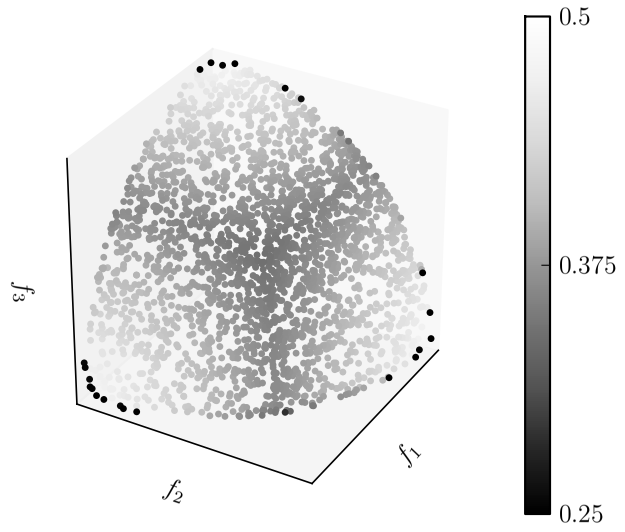
5.4. Summary

Here we have seen how the effect of CDAS & S-CDAS is highly variable across different front shapes in 2 & 3 dimensions. S-CDAS is less affected by scaling however it lacks the ability to distinguish between *mutually non-dominating* solutions upon sub-optimal fronts, e.g. DTLZ1 see section 2.7.1. In addition removing sections of the front made a much larger impact upon the ranking of solutions for S-CDAS where solutions both strengthening and weakening of the ranks occurred whereas for CDAS only marginal strengthening along the edges of the removed sections was seen. This change could effect how the search progresses as some distinct regions will be favoured over others based upon there size. (See S-SDAS equation in section 2.6.7.) CDAS however is highly substitutable to the effects of scaling, we can mitigate for this. However in non-scaled problems this may provide undue preference to objectives with a larger range at this stage in optimisation.

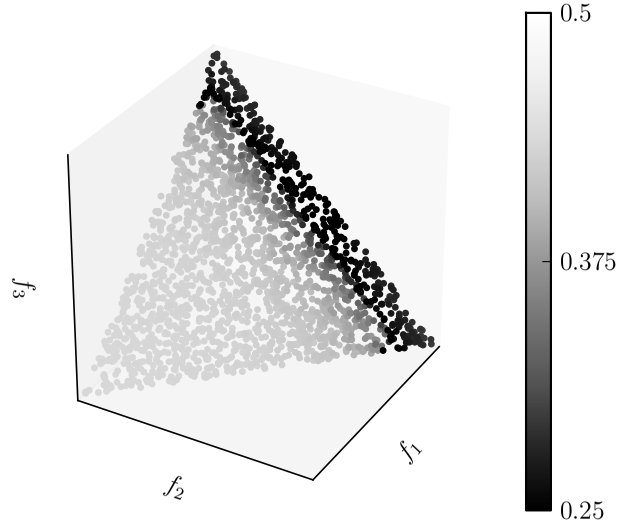
In Chapter 4 we have seen how efective CDAS was on the DTLZ test problems we will take this further in Chapter 6 where we will compare how CDAS, S-CDAS and other



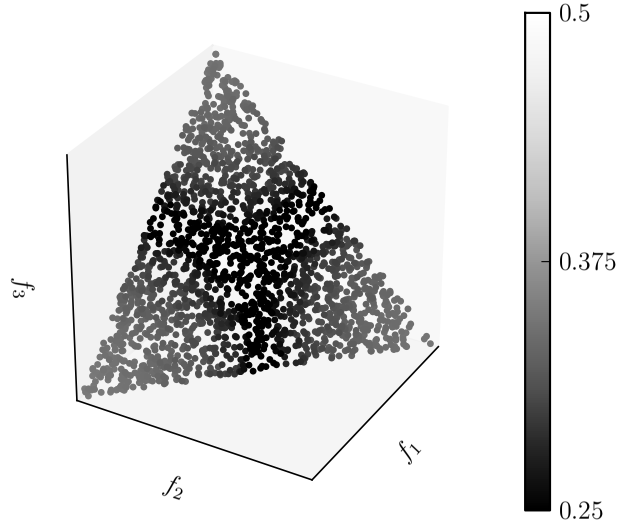
(a) Traditional CDAS



(b) Normalised CDAS



(c) Traditional CDAS



(d) Normalised CDAS

Figure 5.9.: Scaled Convex and Simplex fronts: scaled by 2.5 on the first objective

alterations on CDAS perform on a larger range of DTLZ problems and the newer Walking Fish Group problem set.

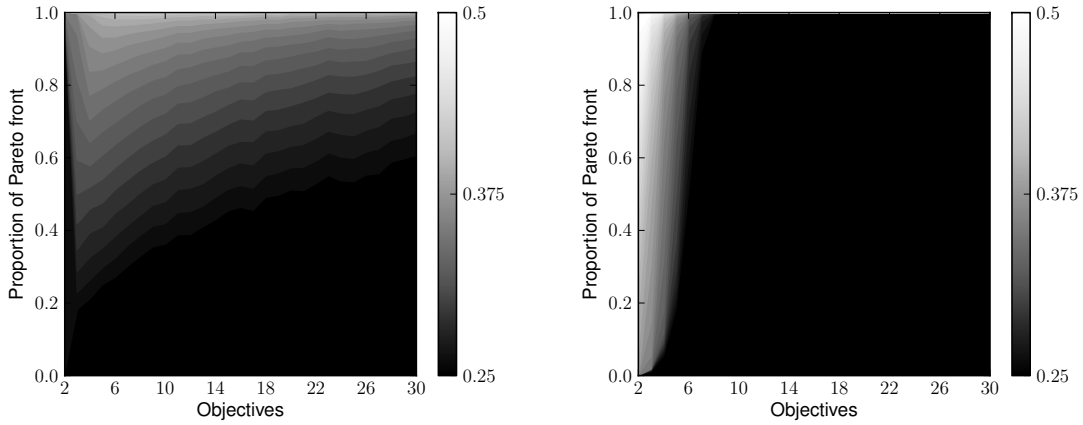
6. CDAS Comparison

Leading on from Chapter 5 here we will investigate how CDAS performs when applied to a larger range of problems, and how the consideration of problem-topology affects the search. Additionally we propose a mapping approach to facilitate an application of CDAS with fewer parameters.

6.1. Mapping to other shapes

As seen in section 5.2 and Chapter 4 the distribution of the *mutually non-dominating* solutions has a large effect upon the impact of CDAS. Here we propose re-mapping the solution set to a pre-determined surface topology prior to computing the CDAS transform. This aims to provide a problem independent application of CDAS. Some information about the topology of the problem is inevitably lost when re-mapping, however we are able to effectively select which regions of a given front are used to propagate the search further by altering the shape of the target surface. So the question arises which shape to choose? In section 5.2 we have seen a range of primitive shapes compared, however only up to 3 objectives. Here we will aim to identify which will provide a good convergence pressure even at 30 objectives.

Figure 6.1 shows the proportion of a front for each CDAS-R rank (See section 3.1.1). This gives an indication of the effect CDAS has on these *pure* primitive front shapes as the number of objectives increases. Everson et al. (2013) noted how the proportion of solutions on the edges of a given front increases with objectives - this will affect the CDAS-R ranking. To select a front shape we require some solutions to not be accepted across all objectives. Considering the Convex case, illustrated in Figure 6.1b, it is clear that from 6 objectives onwards almost 100% of the front is maintained even for the strongest s values. With both Concave and Mixed, Figures 6.1c & 6.1d respectively, there is more selection pressure. However, the Simplex front in Figure 6.1a maintains some convergence pressure for 30 objectives, when using a small s value.



(a) Simplex front

(b) Convex front

(c) Concave front

(d) Mixed front

Figure 6.1.: Median distribution of ranks for CDAS-R: With the median taken over 30 runs each with a uniformly random front of 1000 solutions upon the respective front. The y-axis shows the proportion of the front associated with each minimum s value, which is indicated by shading. The x-axis shows the number of objectives.

6.2. Empirical examination

Here we will set out a range of experiments to gain further insight into how CDAS and S-CDAS perform over many test problems. Our methods will be implemented into a many-objective particle swarm optimisation algorithm (MOPSO) see section 2.6.8.

6.2.1. Experimental designs

For the experiments we take a basic MOPSO implementation with 100 particles in a *fully-connected* population. Each archive, those used for personal guides and global guides, is restricted to a maximum of 100 solutions. To better isolate the behaviour of CDAS and S-CDAS these archives are maintained using *uniformly-at-random* truncation, except in the case of where the S-CDAS ranks are used for truncation. Selection of guides from

these archives is also *uniformly-at-random*. A statistical Pareto archive, to which all new solutions are submitted, is also maintained. For computational speed this archive has a maximum size of 1000 solutions. Each variation is run 30 times starting from a uniformly random population. The parameters of the PSO search, using element wise truncation to manage particles leaving the problem domain, are $c_1, c_2 = 2$ and $\omega = 0.4$ with $\chi = 1$ (Xu and Rahmat-Samii, 2007).

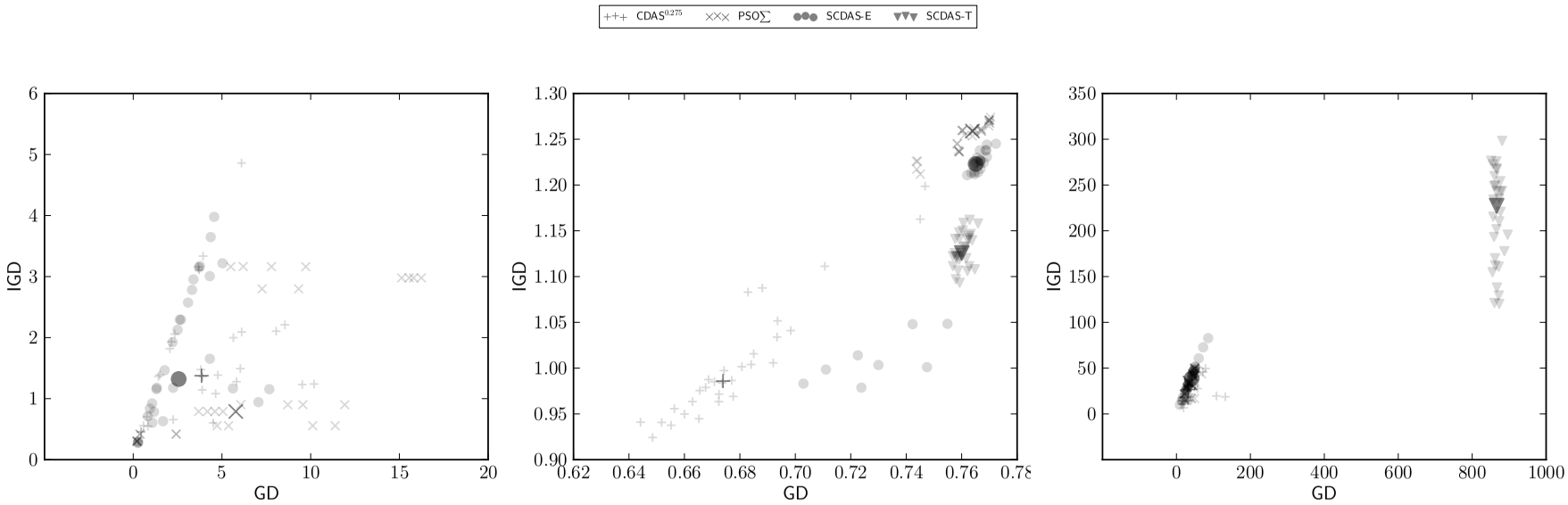
The problems tested upon were the DTLZ (1, 2, 3, 4, & 6)[1] (Deb et al., 2002) and the Walking Fish Group (WFG) (1-9) (Huband et al., 2005) using suggested parametrisation from the original papers. All tests were run on 2, 3, 5, 10, 15, 20, 25 & 30 objectives. Generational distance (GD) and inverse generation distance (IGD) provide a means of comparing the convergence and coverage of these algorithms respectively. However the originally proposed GD and IGD have been shown to poorly compare sets of solutions of different sizes (Schütze et al., 2012). Therefore here we use the proposed power GD_p and IGD_p methods (see sections 2.8.1 & 2.8.2) with $p = 2$ using a uniform sample of 10,000 solutions on the pareto optimal front.

To generate the sample of $10,000$ optimal solutions a larger *uniformly-at-random* sample of $50,000$ was taken. Then it was reduced using the crowding distance (CD) operator (see section 2.6.3) to select a uniformly distributed set of solutions upon the optimal surface. The same set of optimal solutions was used across all the tests. Due to computation constraints imposed by the complexity of IGD_p the sample size was not increased for the higher objectives. This means that as the objectives increase the sampled density reduces rapidly. For 30 objectives we must keep in mind that we are only considering a subset of the optimal front.

To better compare the robustness of CDAS and S-CDAS upon these problems a range of alternative algorithms were also run:

[1] We are not considering the DTLZ 5 test problem due to the observations of Huband et al. (2005) where they identified that the optimal is not fully defined, therefore we cannot rely on the GD and IGD metrics.

PSO	A many-objective PSO with selection uniformly at random. This is equivalent to using CDAS with $s = 0.5$.
CDAS^s	CDAS combined within a many-objective PSO to limit the solutions accepted to both local and global guide archives. The s parameter denotes the s_i values used for all the objectives.
CDAS$^{0.250}$	This is a special case of CDAS^s where the CDAS mapping can be computed as: $\text{CDAS}(u)_i = u_i + \sum_{j \neq i} u_j^2$
SCDAS-E	An elitist implementation of S-CDAS in PSO with only the solutions of the minimum rank preserved onto the next iteration. The rank is computed upon a Pareot *mutually non-dominating* set of solutions.
SCDAS-T	S-CDAS ranks used when truncating the archives to 100 solutions, elitist truncations, keeping the best 100 solutions of the *mutually non-dominating* set.
$\lvert\text{CDAS}^s_{Shape}\rvert$	CDAS computed on solutions that have been rescaled and mapped to a Convex or Simplex front shape respectively. As discussed in section 5.3.
PSO$\sum$	Single objective PSO, summing the objective values using equal weights for all objectives.
Random Walk	A uniform random sampling of parameter space, committing 100 solutions to the archive at each generation.



(a) DTLZ1 for 30 objectives

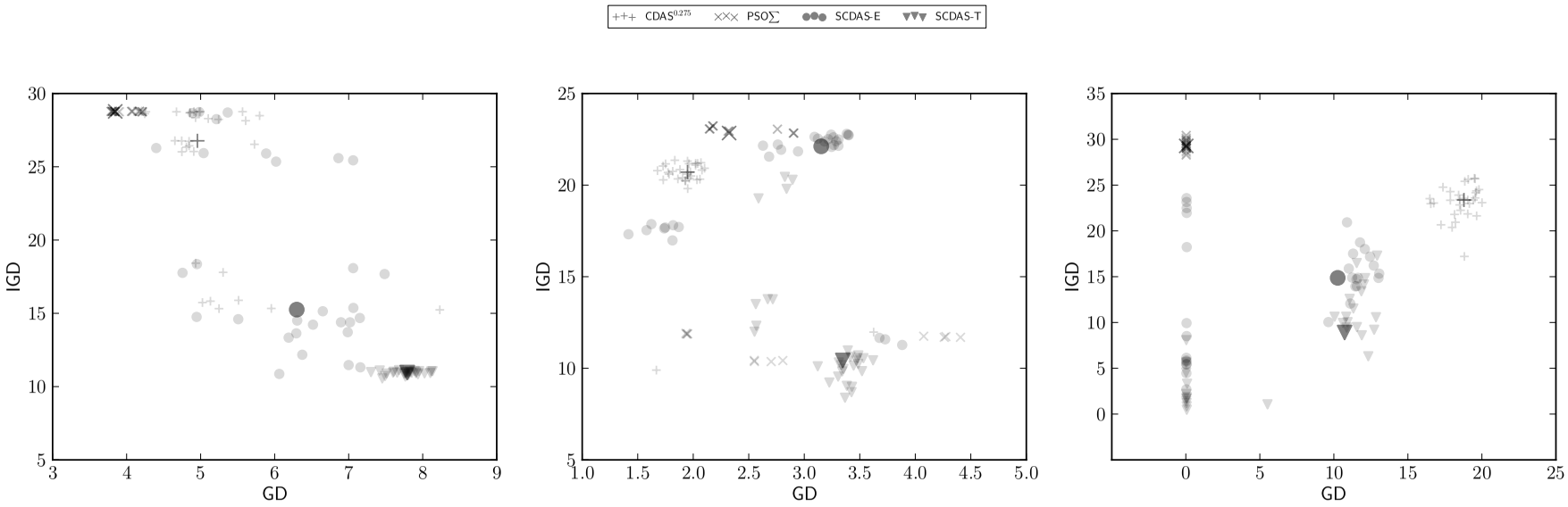
(b) DTLZ2 for 30 objectives

(c) DTLZ3 for 30 objectives

(d) DTLZ4 for 30 objectives

(e) DTLZ6 for 30 objectives

Figure 6.2.: DTLZ test problems, comparing GD and IGD over multiple runs. Each small symbol indicates an individual execution of an algorithm. The larger and darker symbols indicate the median value. Each sample is taken from the archive at the end of the optimisation.



(a) WFG1 for 30 objectives

(b) WFG2 for 30 objectives

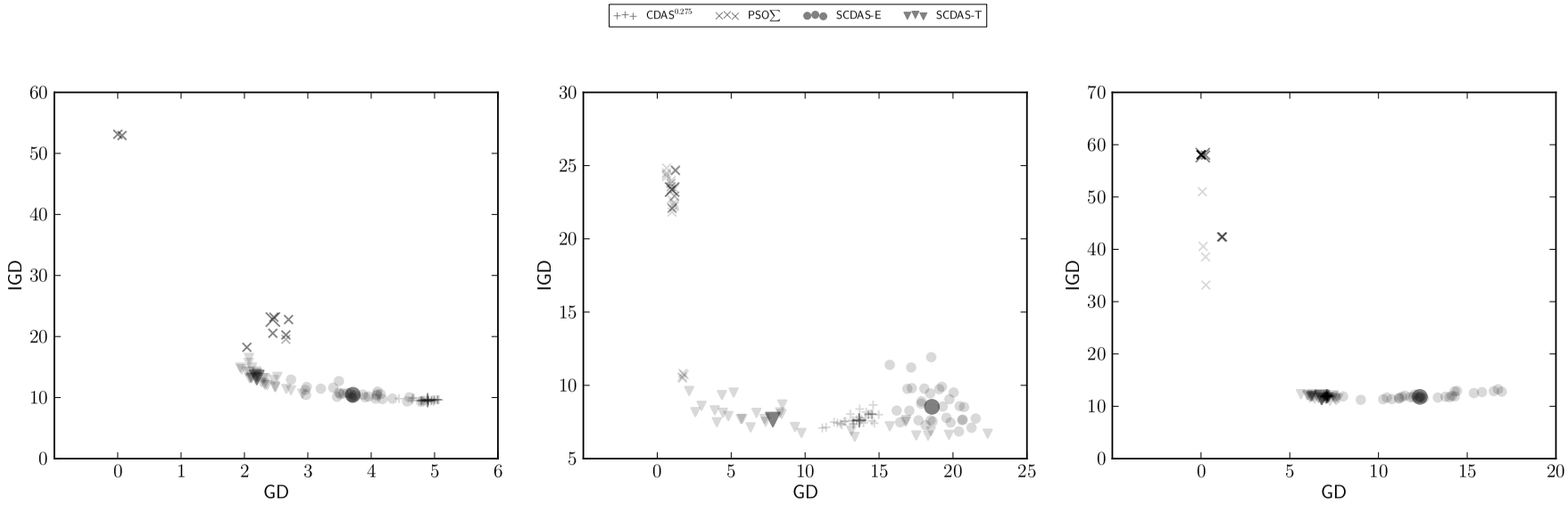
(c) WFG3 for 30 objectives

(d) WFG4 for 30 objectives

(e) WFG5 for 30 objectives

(f) WFG6 for 30 objectives

Figure 6.3.: WFG 1-6 test problems, comparing GD and IGD over multiple runs. Each small symbol indicates an individual execution of an algorithm. The larger and darker symbols indicate the median value. Each sample is taken from the archive at the end of the optimisation.



(a) WFG7 for 30 objectives

(b) WFG8 for 30 objectives

(c) WFG9 for 30 objectives

Figure 6.4.: WFG 7-9 test problems, comparing GD and IGD over multiple runs. Each small symbol indicates an individual execution of an algorithm. The larger and darker symbols indicate the median value. Each sample is taken from the archive at the end of the optimisation.

6.3. Empirical Results

In figures 6.2, 6.3 & 6.4 a subset of the algorithms are compared. After 50,000 fitness function evaluations the final GD and IGD scores are plotted for PSO$\sum$, CDAS$^{0.275}$, SCDAS-E & SCDAS-T. Lower GD and IGD scores are considered better (see sections 2.8.1 & 2.8.2). Each small symbol indicates an individual execution of an algorithm. The larger and darker symbols indicate the median value for that respective algorithm. The statical archive is used to compute the GD and IGD.

For the DTLZ test problems, figure 6.2[2], there is a substantial positive linear correlation between GD and IGD. This correlation forms a lower bound for GD as a function of IGD. It can also be seen on lower objectives (see Figure 6.5). We may attribute this to the nature of the DTLZ problems as the distance to the optimal front is separable from the dimensions that determine the position upon the front (see section 2.7.1).

Problem	Objective Where $IGD \propto GD$
DTLZ 1	2, 3, 5, 10, 15, 20, 25, 30
DTLZ 3	2, 3, 5, 10, 15, 20, 25, 30
DTLZ 4	2, 3, 5, 10, 15, 20, 25, 30
DTLZ 6	10, 15, 20, 25, 30

Figure 6.5.: Objectives for DTLZ problems where $IGD \propto GD$.
For DTLZ2 the correlation is less pronounced (especially as the objectives increase 20+). This may be due to the solutions being comparatively closer to the optimal front.

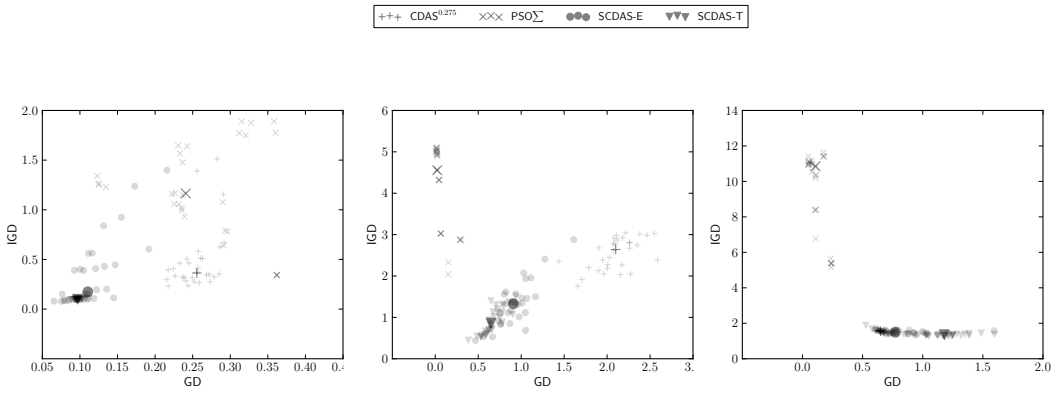
In contrast, the WFG problems (figures 6.3 & 6.4) produce a Pareto trade-off between the metrics. This can be most clearly seen for WFG1, 7, 8 & 9 where even the individual runs of each algorithm form a competing trade-off front. PSO$\sum$ constantly performs well with respect to convergence, but is unable to spread across the front. SCDAS-T in general provides better coverage across the front in comparison to SCDAS-E on all problems except for WFG7.

A large number of individual runs on WFG3 (figure 6.3c) achieve solutions close to the optimal set, with comparatively low GD scores. All the algorithms apart from CDAS$^{0.275}$ show this behaviour. SCDAS-E and SCDAS-T both have median GD score that is a long way from this extrema. We postulate this may be due to the *degraded* nature of the WFG3

[2]To maintain readability on figure 6.2a SCDAS-T is excluded from the graph as all of its runs are significantly worse on both metrics.

problem.

Unlike DTLZ for the WFG problems the relative performance of algorithms changes with the number of objectives. To highlight this see figure 6.6, where the development of WFG4 from 3 to 10 objective is shown. PSO$\sum$ stands out in this figure as its GD score is largely unchanged across all objectives. The competition between GD and IGD can be seen from 10 objectives onwards.



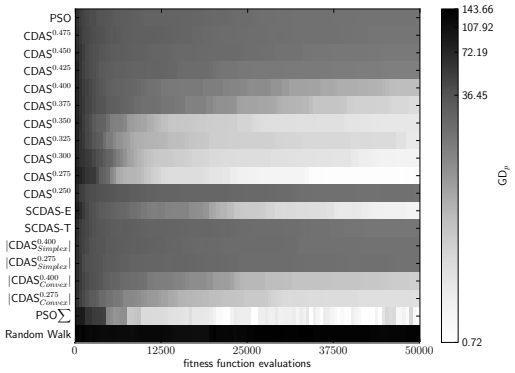
(a) WFG4 for 3 objectives (b) WFG4 for 5 objectives (c) WFG4 for 10 objectives

Figure 6.6.: Development of WFG4 from 3 to 10 objectives. Each small symbol indicates an individual execution of the algorithms. The larger and darker symbols indicate the median value. Each sample is taken from the statistical archive at the end of the optimisation.

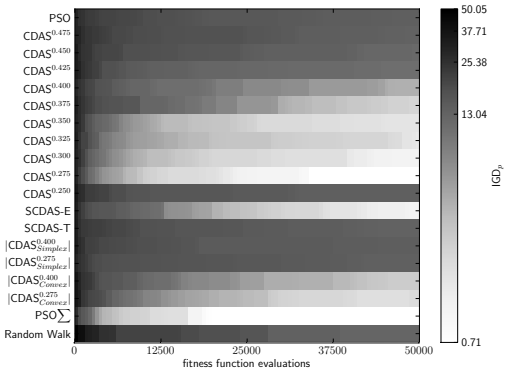
The investigation over all the algorithms will now be considered. Only a subset of the results will be graphically shown here, the full set can be seen in appendix A. Figures 6.7 to 6.13 show the median state of the statical archive over time as the optimization progresses. Each algorithm is identified by a row on the *y-axis*. Fitness function evaluations are shown on the *x-axis* (with the GD and IGD plots being adjacent horizontally). The GD and IGD score is shown by the shading, the color gradient uses a log scale to emphasise the differences between the algorithms.

We observe from Figure 6.7, and appendix A.1 that there is very little variation in the relative performance between algorithms from 2 to 30 objectives. The exception to this is $|\text{CDAS}^{0.400}_{convex}|$ and $|\text{CDAS}^{0.275}_{convex}|$, both of these algorithms fail to maintain their position for high objectives.

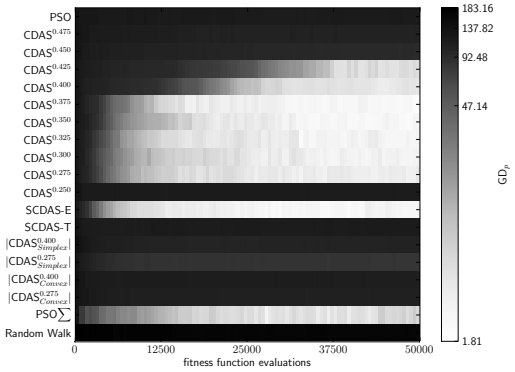
The strong performance of PSO$\sum$ on DTLZ1 for both GD and IGD is somewhat of a surprise. This algorithm provides a very strong selection pressure by only maintaining in each personal and global archive the best solutions after summing over all objectives. We hypothesize that this is due to a combination of the nature of DTLZ, as discussed above, and the front topology of DTLZ1. All solutions on the leading simplex front have



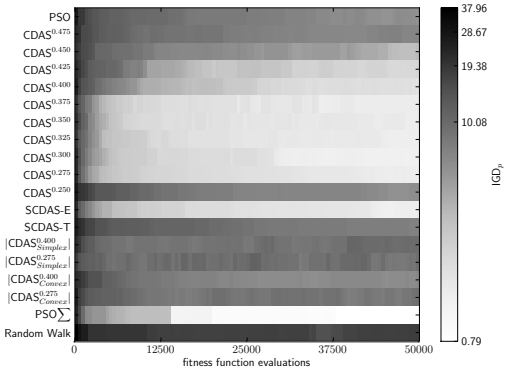
(a) DTLZ1 Median GD on 2 objectives



(b) DTLZ1 Median IGD on 2 objectives



(c) DTLZ1 Median GD on 30 objectives



(d) DTLZ1 Median IGD on 30 objectives

Figure 6.7.: Results for DTLZ1 with 2 & 30 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.

the same fitness value for PSO$\sum$ - given a distribution of solutions on multiple different simplexes, PSO$\sum$ is able to always select the best simplex. The investigation in Chapter 5 and figure 6.1a shows us that both CDAS and SCDAS have high selection pressure on the simplex fronts. With the multiple deceptive front in by DTLZ1 a strong selection pressure allows these optimisers to drive down to better fronts, this confirms the earlier findings in Chapter 5.

For DTLZ3 the performance is similar to that on DTLZ1, see appendix A.3. PSO$\sum$, SCDAS-E & strong CDAS take the lead, however they do not manage to achieve as good GD or IGD scores. Even after 50,000 fitness function evaluations the best median GD is > 23 in comparison to DTLZ1 where the best GD score is 0.72 (on 2 objectives). Additionally weaker CDAS is less competitive on DTLZ3 compared to DTLZ1. The difference in GD and IGD score highlights the importance of front shape. Both problems share the same deceptive fronts in objective space however DTLZ3 has the convex shape

taken from DTLZ2, the positive sector of a hyper-sphere. On DTLZ3 PSO$\sum$ will favor the extreme points on the corners of the hyper-spherical section. This provides a very strong selection pressure to both drill down - since the dimensions responsible for positions on the front in DTLZ are independent of the distance to the optimal front it is then easy for particles in the swarm to migrate across the front, filling the statical archive, with many solutions upon the same deceptive front.

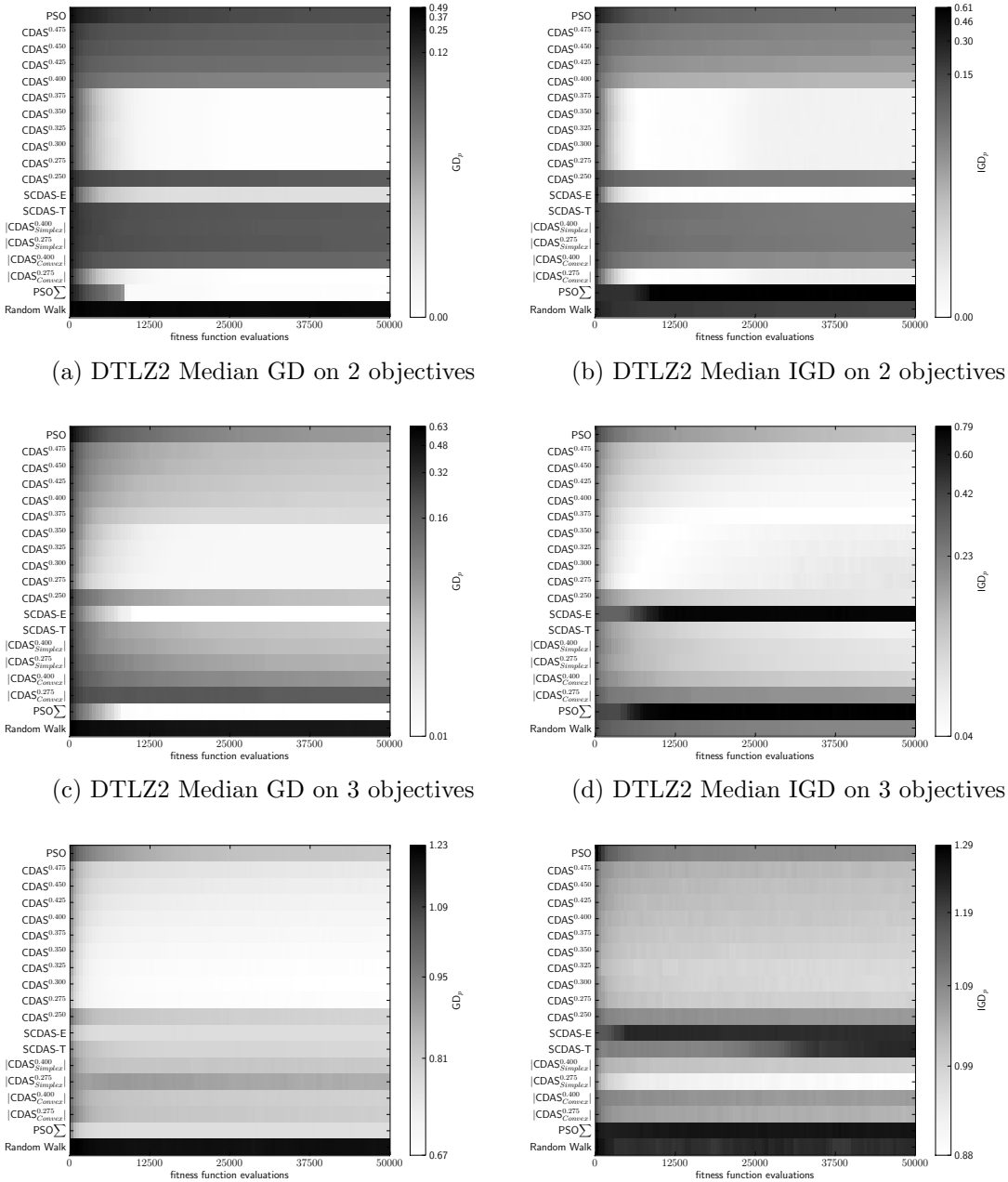
In Figure 6.8 the results of our experiments on DTLZ2 are presented. This is the only DTLZ test problem where we do not see GD and IGD hand in hand on higher objectives (also seen in chapters 4 & 3). We see a clear difference between the 2 and 3+ objectives for CDAS, SCDAS-E, $|\text{CDAS}^{s}_{convex}|$ and PSO$\sum$. CDAS with $s \in [0.275, 0.350]$ performs well on both GD and IGD for 2 objectives, however for 3+ objectives its IGD score diverges as the GD score improves. SCDAS-E provides the same good GD & IGD for 2 objectives, on 3+ objectives the IGD score is very poor.

The scaled and mapped $|\text{CDAS}^{0.275}_{convex}|$ likewise has good GD and IGD scores on 2 objectives, but is unable to perform on 3+ on either GD or IGD. We postulate the mapping of all solutions onto the convex surface that forms the same shape as the optimal front reduces the ability to effectively distinguish between distant and close solutions for DTLZ2 on higher objectives. Since all solutions for DTLZ2 fall within the unit hyper-cube close to the front. On 2 objectives the mapping and re-scaling therefore have little effect[3], therefore $|\text{CDAS}^{s}_{convex}|$ behaves very similar to plain CDAS. PSO$\sum$ while effective at providing good GD values is completely unable to maintain a distribution selection across the front, as discussed above this may be due to its selection only of solutions in the extreme corners of the optimal front.

Overall for DTLZ2 we consider the trade-off between GD and IGD most likely due to the proximity to the optimal front obtained by all tests: unlike DTLZ3 or 4, which both share the same front shape, DTLZ2 is widely considered the easiest test in the DTLZ test suits. With so many solutions extremely close to the optimal set it is difficult to maintain diversity, since once found many solutions will never be dominated. If the selection achieves used for personal and global best solutions except very few new solutions the same guides are used over and over again, this can result in a collapsing population.

The latter development in IGD, from 15+ objective, of $|\text{CDAS}^{0.275}_{simplex}|$ we consider to be another identification of the nature of the objective space for DTLZ2. With all solutions

[3]For DTLZ2 on 2 objectives a set of 100 *mutually non-dominating* random solutions mapped to the front is enough to effectively describe the full surface.



(a) DTLZ2 Median GD on 2 objectives

(b) DTLZ2 Median IGD on 2 objectives

(c) DTLZ2 Median GD on 3 objectives

(d) DTLZ2 Median IGD on 3 objectives

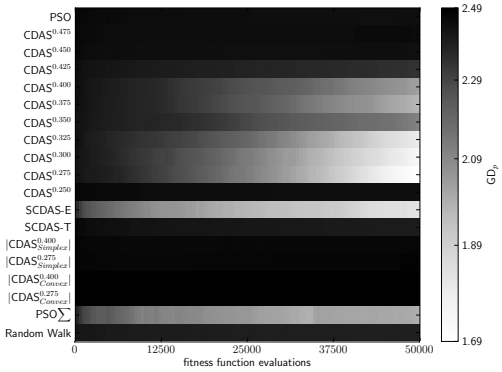
(e) DTLZ2 Median GD on 30 objectives

(f) DTLZ2 Median IGD on 30 objectives

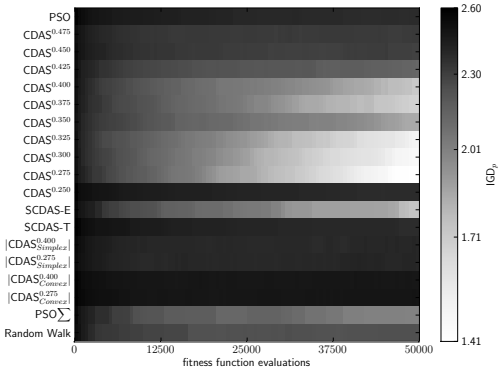
Figure 6.8.: Results for DTLZ2 with 2, 3 & 30 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.

in the unit hyper-cube, the perimeter of the front is naturally easier to sample than the center. The mapping to a simplex results in strong selection preference to the center of the front enabling the statical archive to be filled not only with perimeter points.

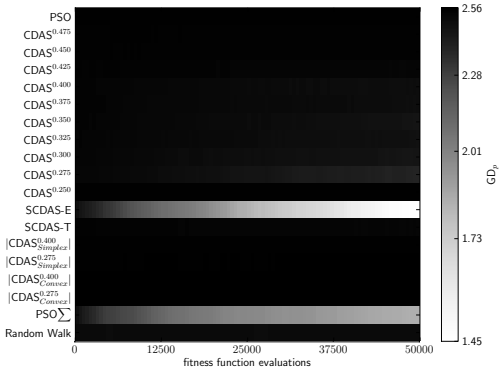
DTLZ4 while forming the same optimal shape as DTLZ2 & DTLZ3, DTLZ4 has a strongly non-uniform mapping resulting in segments of the optimal sufferance to be much



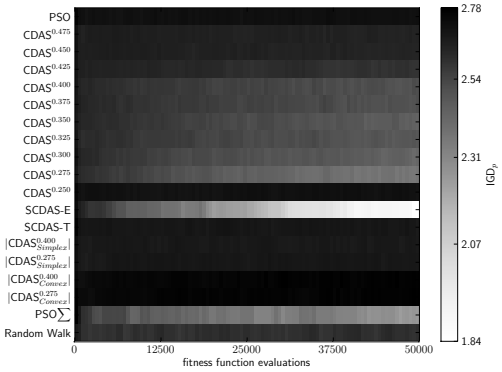
(a) DTLZ4 Median GD on 5 objectives



(b) DTLZ4 Median IGD on 5 objectives



(c) DTLZ4 Median GD on 10 objectives

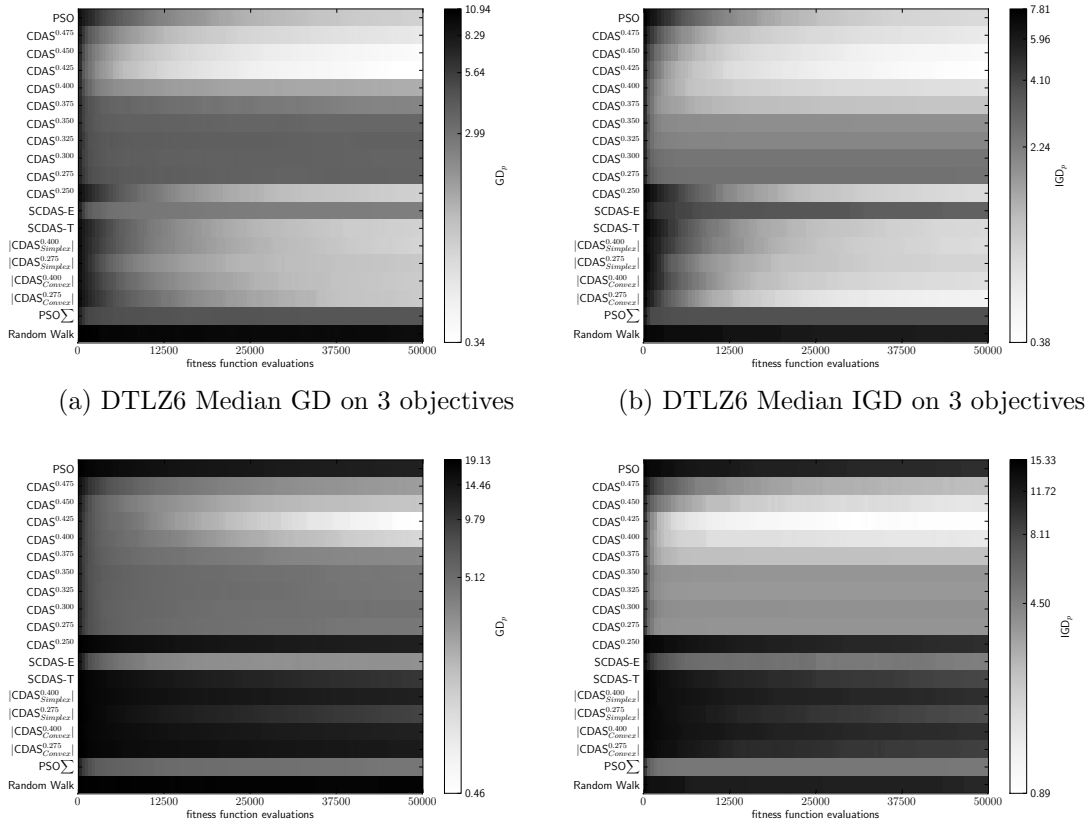


(d) DTLZ4 Median IGD on 10 objectives

Figure 6.9.: Results for DTLZ4 with 5 & 10 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.

more difficult to find. In addition unlike DTLZ2 the solutions of objective space are not constrained to the unit hyper-cube. In chapters 3 & 4 we have seen the difficulty optimisers had converging and spreading across the front, even with CDAS converging slowly. In figure 6.9 we observer that CDAS, seen to perform well in chapter 4, is unable to compete with SCDAS-E or PSO$\sum$ for 10+ objectives. The relative ranking of S-CDAS gives a kind of crowding distance operator (see section 2.6.3), with solutions further apart more likely to be given equal ranks. This is important in this case as it encourages the optimiser to search the more sparse parts of the objective space (see chapter 5 for further discussion of this on 2 and 3 dimensional fronts).

The results of our experiments on DTLZ 6 can be seen in Figure 6.10. Unlike on the other DTLZ problems it appears a weak implementation of CDAS is more effective than strong CDAS. With s values ranging from 0.475 to 0.400 performing best on lower objectives and finally on 30 objectives just $s = 0.475$. Since DTLZ6 is the only DTLZ



(a) DTLZ6 Median GD on 3 objectives

(b) DTLZ6 Median IGD on 3 objectives

(c) DTLZ6 Median GD on 5 objectives

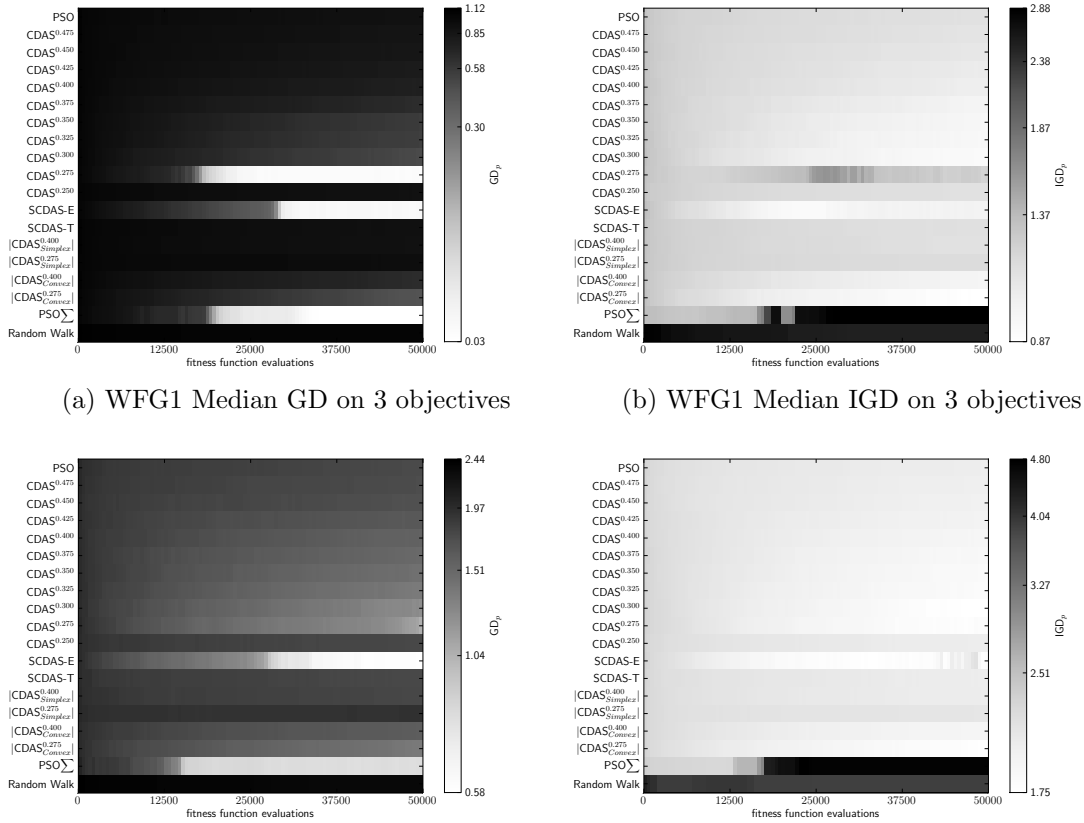
(d) DTLZ6 Median IGD on 5 objectives

Figure 6.10.: Results for DTLZ6 with 3& 5 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.

test problem that contains multiple unconnected optimal segments in objective space the strong CDAS may result only selection the edges of some of these disjoint segments, see figure 5.6c in chapter 5. A weaker CDAS will still provide some needed selection pressure without ignoring the many disjoint segments. Weak CDAS is able to perform relatively well in comparison to the best algorithm on DTLZ3, with GD scores on 30 objectives around 2.

In this problem there is a clear switch of behavior between the multi- and many objective experiments. On 2 & 3 objectives the very strong limiting implementation of CDAS with $s = 0.25$ shows some promise alongside SCDAS-T, plain PSO and all of the mapping algorithms. In comparison form 5+ objectives $\text{CDAS}^{0.250}$, SCDAS-T, PSO and the mapped functions perform badly, and SCDAS-E performs well. The increasing number objectives rapidly grows the proportion of empty space between each disjoint front, we believe this may effect these algorithms, but further work is needed on the topic of disjoint

fronts to give insight into the problem.



(a) WFG1 Median GD on 3 objectives

(b) WFG1 Median IGD on 3 objectives

(c) WFG1 Median GD on 5 objectives

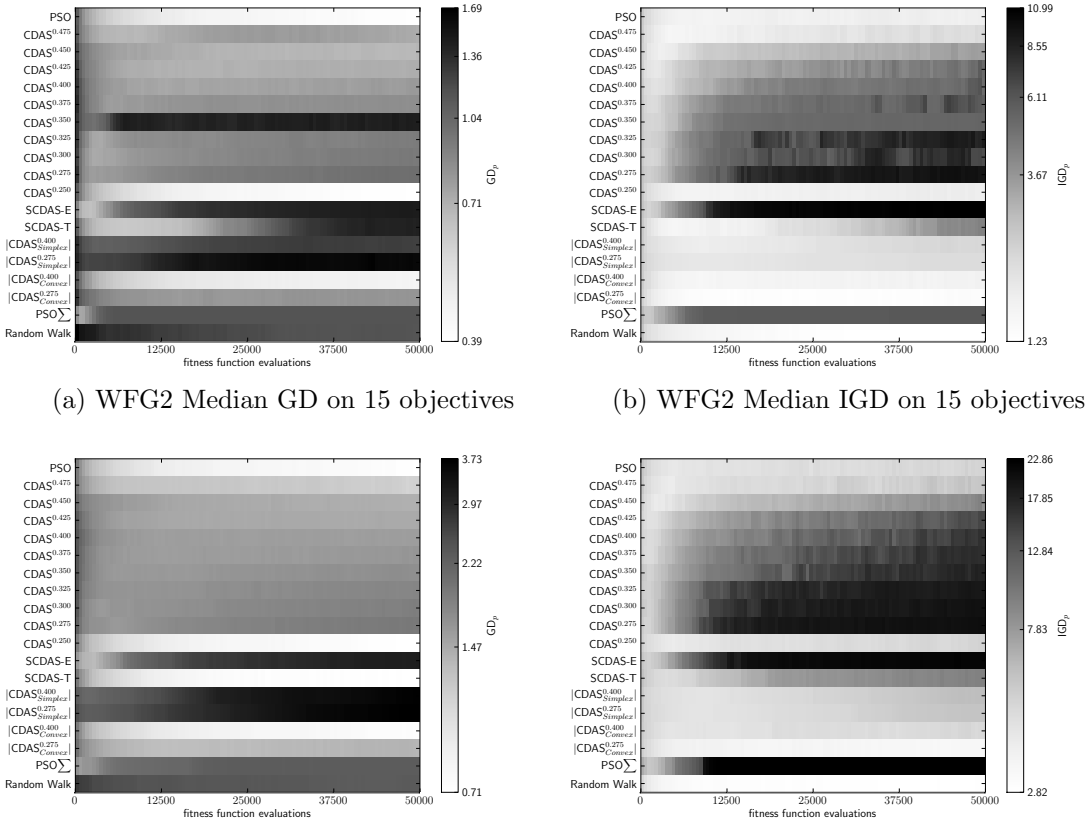
(d) WFG1 Median IGD on 5 objectives

Figure 6.11.: Results for WFG1 with 3 & 5 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs. For all objectives see appendix A Figure A.6.

In Figure 6.11, the results on WFG1 are presented. As discussed above there is a clear trade-off between GD and IGD. The exceptions to this for WFG1 are on 2 objectives where strong CDAS achieves good GD & IGD. And SCDAS-E up-to 15 objectives maintains good convergence on GD without significant penalties on IGD.

Figure 6.12 shows the results for WFG2, which has many disjoint segments on the optimal front. On 2 & 3 objectives PSO, week CDAS, CDAS^0.250, SCDAS-T and the mapping function show promising results. On 5+ objectives PSO, $|\text{CDAS}_{converx}|$ & CDAS^0.250 continue to perform well on both GD & IGD. From this we can see that one can perform well on WFG2 with an algorithm without strong selection pressure is needed: plain PSO and $|\text{CDAS}^{0.400}_{converx}|$ both provide minimal selection pressure, in comparison to the other algorithms. This is similar to our findings above for DTLZ6.

WFG3 to WFG9 all display the same polarity between multi- and many-objectives,



(a) WFG2 Median GD on 15 objectives

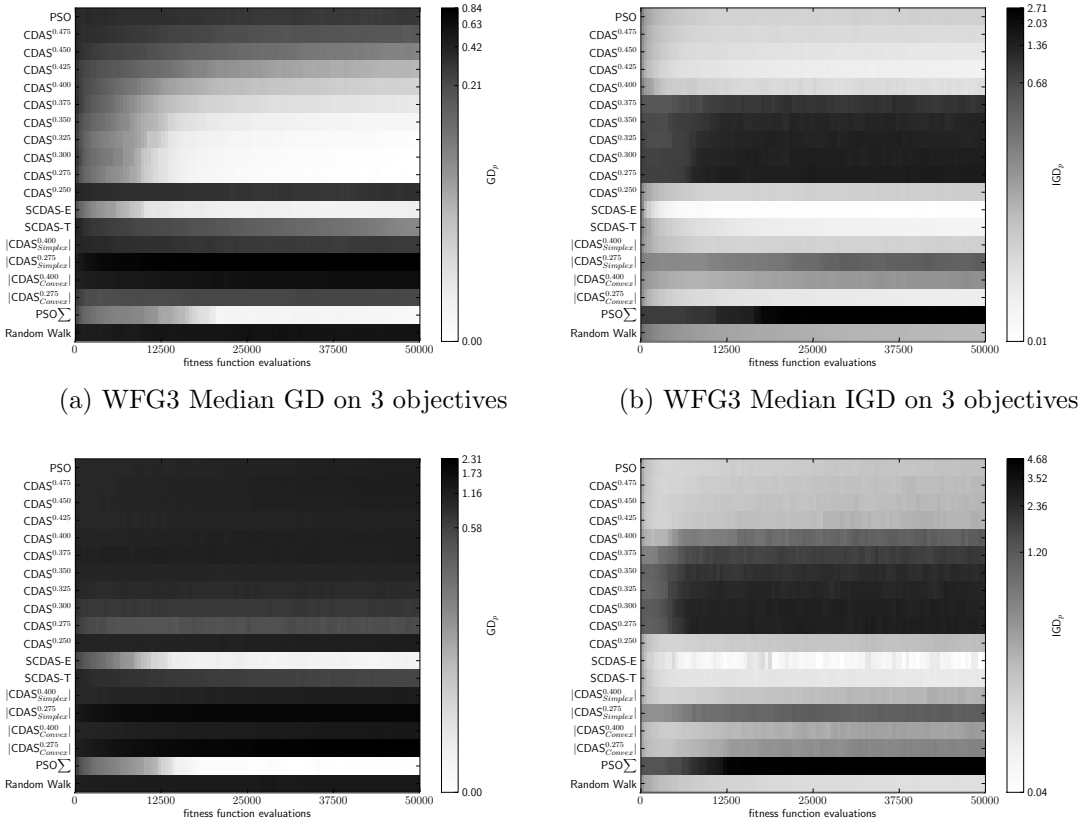
(b) WFG2 Median IGD on 15 objectives

(c) WFG2 Median GD on 30 objectives

(d) WFG2 Median IGD on 30 objectives

Figure 6.12.: Results for WFG2 with 15 & 30 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.

see figures 6.13 & A.9 – A.14. The change in IGD, on most problems, is from 15+ objectives lagging behind the shift in GD that is between 5 and 10 objectives. Across all of these problems on low objectives algorithms with less selection pressure outperform their counterparts on both metrics. For higher objectives GD and IGD are in compensation. PSO$\sum$ with its high selection pressure provides good convergence from 10+ objectives for all of these problems. Overall the best IGD on high objectives is provided by Random Walk. While all algorithm behave the same on these problems there are a few anomalies: As mentioned above the *degraded* nature of the WFG3 front may contribute to the strong performance of SCDAS-E from 3 to 25 objectives - further extension of chapter 5 is needed to confirm this.



(a) WFG3 Median GD on 3 objectives

(b) WFG3 Median IGD on 3 objectives

(c) WFG3 Median GD on 5 objectives

(d) WFG3 Median IGD on 5 objectives

Figure 6.13.: Results for WFG3 with 3 & 5 objectives. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.

6.4. Summary & Discussion

Across this range of experiments we can see a common divide between the multi- and many-objective problems: In the WFG problems it is the most clear, however it can be seen in all of the problems, as discussed by (Huband et al., 2006) & (Ishibuchi et al., 2008). Where it is attributed to the increased dimensions of objective space resulting in more equivalent solutions under Pareto comparison reducing the convergence rate. This causes algorithms that perform well on 2 and 3 objectives to experience an explosive growth in archive size at the number of objectives increases. However among the algorithms we have observed here S-CDAS has in many cases managed to bridge this gap providing adequate performance on both multi-and many objective problems. The counter example of this is DTLZ2 as discussed in section 6.3.

In the experiments here, and in previous studies, we have strong evidence to suggest

that the DTLZ test problems are significantly easier for many-objective algorithms to solve than the WFG suite. DTLZ 1, 2, 3, 4 & 6 appear trivial in comparison to the WFG problems: Almost all algorithms attain some convergence for DTLZ on GD and IGD simultaneously across the full objective range. Whereas no algorithms managed this feat for the WFG suite. This has previously been touched upon by the authors of the WFG suite, they hypothesised that the nature and distribution of the optimal solutions in parameter space for DTLZ allow many algorithms to easily converge and spread is out across the optimal front, see sections 2.7.1 & 2.7.2. This trade-off on WFG between GD and IGD is clearly seen in the figures plotting GD vs IGD, where the individual runs form a trade-off surface.

S-CDAS appears from these experiments to be the best algorithm overall for the DTLZ problems, providing good convergence and spread across the optimal set, for both low and high objectives on the majority of test problems. Here we have examined two implementations of S-CDAS: the elitist selection version, in general performed better on the DTLZ test problems, whereas there is a less clear difference for WFG problems. These two implementations span the extremes of selection pressure providable by SCDAS. The comparatively better performance of the elitist implementation on DTLZ may be attributable to the reduced need to diversify during search, since once a single solution at a given distance from optimal front is discovered only a subset of dimensions in the search space need to be searched to find all solutions at this distance. On the WFG problems there is a trade-off between algorithms performing well on GD and those performing well on IGD for S-CDAS both elitist and truncation provide better IGD scores than PSO$\sum$ but have worse convergence.

Our overall results show S-CDAS as the leading method for providing good convergence and spread thought the front. However this is not as clear cut as the dominance of CDAS over selection methods in Chapter 4. Even though S-CDAS removes the need to set a correct vector of $\mathbf{s}$ values we see that the use of the S-CDAS rank is the deciding factor. In Chapter 5, the region preferred by S-CDAS is less affected by the scaling of the objective range. This may be a contributing factor to S-CDAS's IGD performance on the WFG problems.

The strong convergence of PSO$\sum$ on many of the problems investigated, but not all, is perhaps not so surprising. We postulate that while PSO$\sum$ is an effective means for providing convergence alone it is unable to provide good distribution across the front. A

case for this is DTLZ 2 where comparatively speaking PSO$\sum$ is only able to provide GD scores and not IGD. As discussed for S-CDAS PSO$\sum$ has too high a selection pressure to perform well on any of the WFG problems with respect IGD.

In summary we would suggest the application of S-CDAS to most problems, however the implementation into the heuristic has a large effect. We have seen that by altering the application of S-CDAS we are able to attain good convergence and spread in the majority of our experiments. However in cases where only a single good solution is needed we would suggest the application of PSO$\sum$ or single objective reduction/decomposition methods. Finally we suggest that the DTLZ test suite be only used mainly to analyse the performance of algorithms close to the optimal set, however harder problems are needed for a full comparison of algorithms.

7. Summary

In this study we have investigated how modifying the Pareto dominance calculation can be used to encourage convergence on many objective problems with particle swarm optimisation. We have proposed two new novel adaptations to existing algorithms and analysed their application across a broad range of standard tests.

In Chapter 2 we introduced the PSO heuristic and discussed the state of the art for many objective optimisations. Traditionally within multi-objective PSO optimisation a range of *selection operators* have been applied to improve convergence and coverage. In Chapter 3 we first looked at how altering the *selection* methods effects the search. Here we introduced a new method, CDAS-R, to provide a ranking on solutions used for selection. Our findings here, on DTLZ 1-4, clearly indicated Sum of Average Rank (SR) and Average Rank (AR) as the best selection operators.

We follow this in Chapter 4 with an initial investigation into the use of ranking methods when controlling the archive of solutions stored. From these results here we see our first indication of the significant improvement provided by CDAS. We see CDAS providing both rapid initial convergence and significantly better Generational Distance (GD) and Inverse Generational Distance (IGD) scores in comparison to the selection operators used on the same problems in Chapter 3. Here we identified how the performance of CDAS is not only a function of the number of objectives and the complexity of the problem, e.g. the existence of deceptive fronts, but also the shape of the optimal front in objective space. Chapter 5 follows this analysis further investigating a newer variant of CDAS, namely S-CDAS. This included an in-depth investigation into the selection properties of these 2 algorithms. Here we examined 2 and 3 objective examples in detail observing how each algorithm preferred different subsections of each front shape. This re-confirmed our findings from Chapter 4 that CDAS is highly dependent upon the shape of the optimal set and current approaching Pareto front. From this we observed how S-CDAS performed much better with respect to distribution across the front when the Pareto front is not uniformly scaled across objectives.

Chapter 6 continues with a deeper empirical examination of both CDAS and S-CDAS. These are compared with a novel adaptation of CDAS were the Pareto set is first mapped to a predefined surface shape before CDAS is applied. On almost all the higher many-objective tests the single objective implementation of PSO (PSO$\sum$) provided fast, good convergence. This is in tandem with the collapse in diversity across the front. While this was expected, S-CDAS was shown to be an effective algorithm on the majority of many-objective problems. It was able to provide good convergence on almost all problems, while achieving a good distribution across the front for most of the problems. In addition, as seen in Chapters 3 & 4 we see a clear difference between multi and many objective performance, with the dominant algorithms switching roles as the objectives increase from 3 to 5. Finally our findings suggest further understanding of the DTLZ and WFG test problems are needed. We noted DTLZ proves significantly easier than WFG for algorithms to maintain good convergence and diversity simultaneously.

7.1. Future work

We believe MOPSO will have its place in future application in many objective problems. Further work will be needed to develop methods that can maintain diversity and convergence on difficult problems, such as the WFG test suit. We see a split in goals emerging: development of generic heuristics that aim to solve all/most problems with little of no parametrisation and developing tool/insights into problems to help the correct heuristic selection. We believe the second approach will lead to better application to industry and use behold the academic community. Further work looking in detail into the behavior of selection and archiving operators on higher dimensional solution sets is critical for this.

This work has been focused upon the incorporation of methods to modify the Pareto domination calculations and their adaptation/implementation into Particle Swarm optimisation. To keep our study focused we have not considered a range of important factors in many-objective PSO. An investigation into the affect of maximum archive size, including an analysis of the computation cost that large archives may bring would be helpful. Many EMOs use a combination of selection and archiving methods, within MOPSO we have an additional degree of freedom since we can also control the methods used for personal memory independently of those for global memory. Further work is needed to compare CDAS & S-CDAS to the other dominance modification methods such as α-dominance & ϵ-dominance.

MOPSO with CDAS may also be used to provide a decision-maker's prior preference heuristic, in a similar way to light-beam search, the $\mathbf{s}$ vector can be manipulated to prefer a mixture of objectives. We would like to also consider another possible impersonation of CDAS-R where the rank is the number of different $\mathbf{s}$ vectors for which a solution is dominated, vectors could be pre-determined by the decision-maker or chosen to uniformly sample the space. This would allow the decision-maker to have multiple distinct preferred targets within one optimisation.

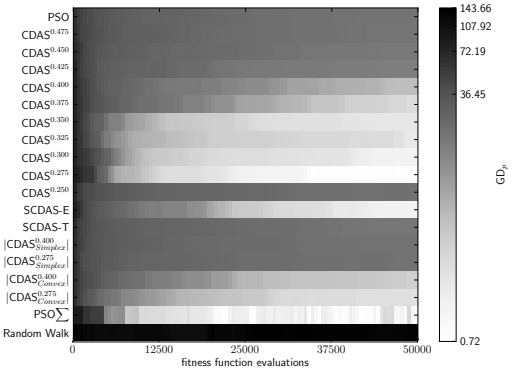
Within our work we have used a single value for the $\mathbf{s}$ vector in CDAS, we would be interested in an investigation to the effect of randomly varying this such that each each particle has a fixed random vector set at the beginning of the optimization. This may result in each particle focusing on a different regions of objective space. Another possible implementation of this could form a multi-swarm approach with each sub-swarm using a different $\mathbf{s}$, with strong selection, this is very close to decomposition methods: since CDAS in each sub-swarm will favor different combinations of objectives. Another implementation could take the mapping approach to a multi-swarm with each sub-swarm taking a different mapping.

An addition that the concept of sub-swarms provides is the possibility to train each sub-swarm on a subset of objectives. It would be important for non-separable problems that each objective is trained pairwise with each-other in at least one sub swarm. Possibly such sub-swarms might need to propagate solutions through each-other via a tree of swarms, such that the child swarms use subsets of objectives from the parent.

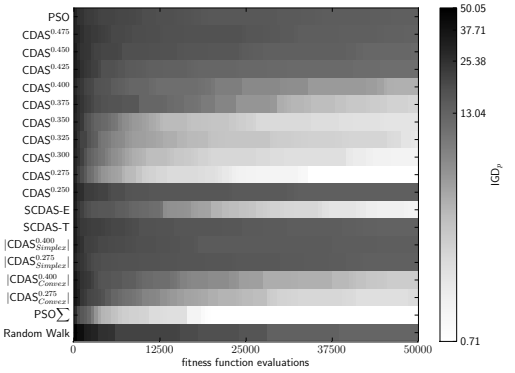
Appendices

A. Experimental Results

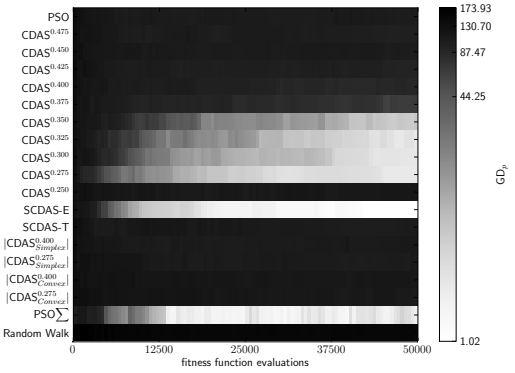
Here the results of the experiments undertaken in Chapter 6 are presented in full. Each problem is presented in turn with objective numbers increasing down the page. With the different algorithms on the y-axis and fitness function evaluations on the x-axis. The shading indicates the score of the respective metric. GD and IGD are computed upon the statistical archive of 1000 solutions. All metrics are taking the median over 30 runs.



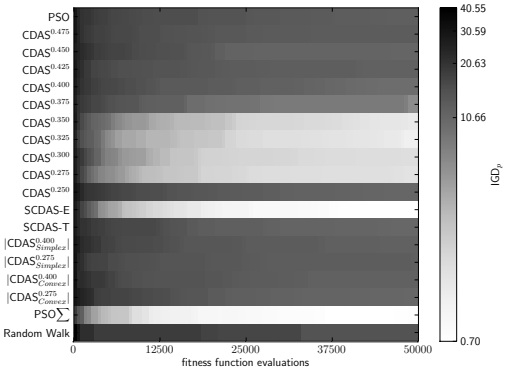
(a) DTLZ1 Median GD on 2 objectives



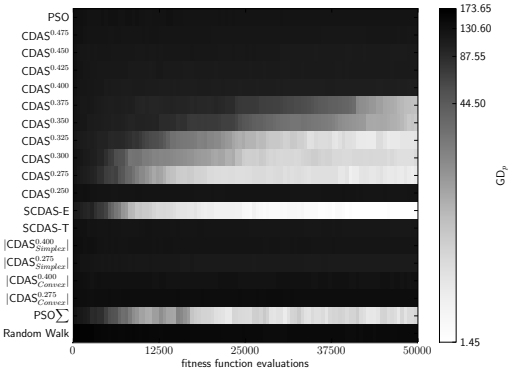
(b) DTLZ1 Median IGD on 2 objectives



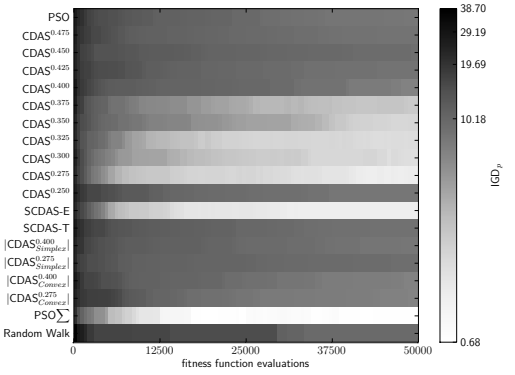
(c) DTLZ1 Median GD on 3 objectives



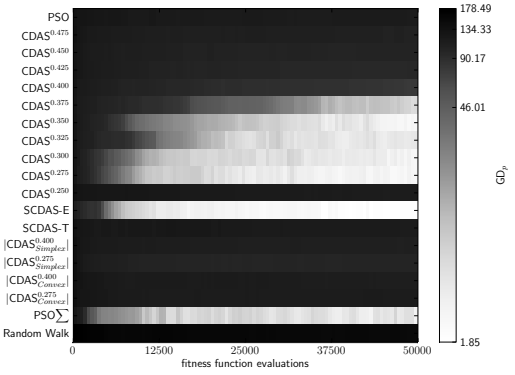
(d) DTLZ1 Median IGD on 3 objectives



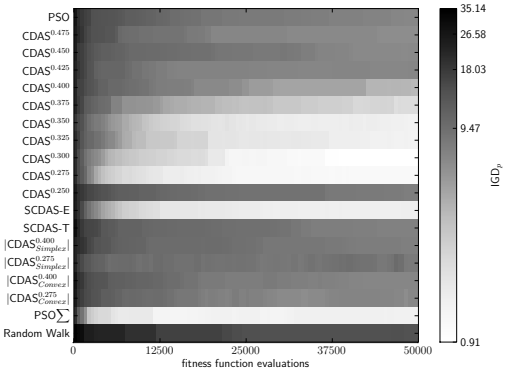
(e) DTLZ1 Median GD on 5 objectives



(f) DTLZ1 Median IGD on 5 objectives

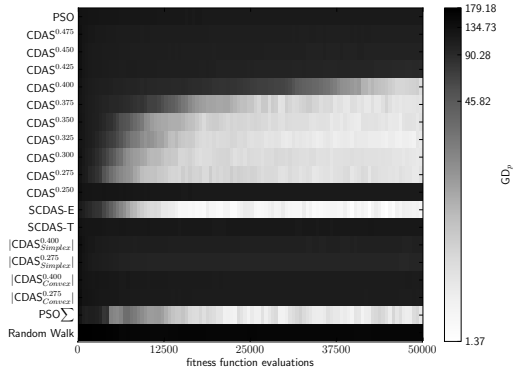


(g) DTLZ1 Median GD on 10 objectives

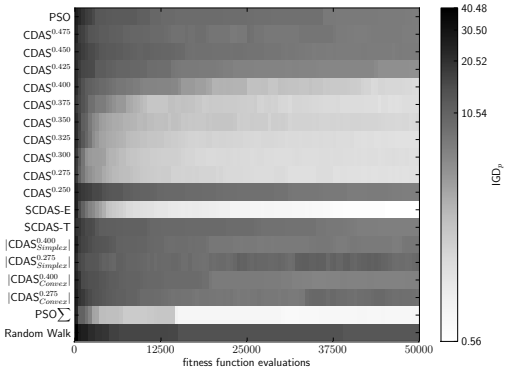


(h) DTLZ1 Median IGD on 10 objectives

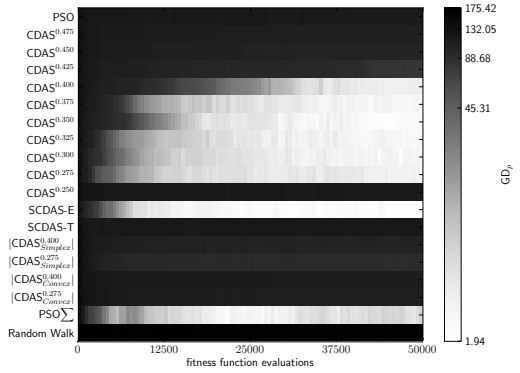
Figure A.1.: DTLZ1.



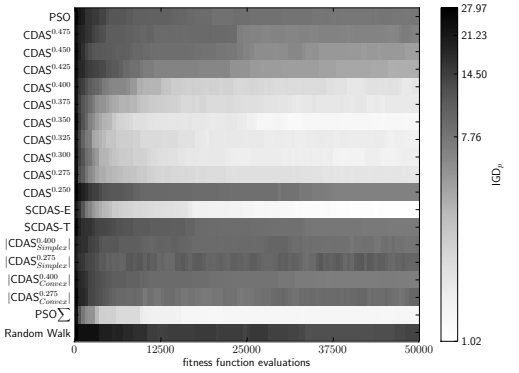
(i) DTLZ1 Median GD on 15 objectives



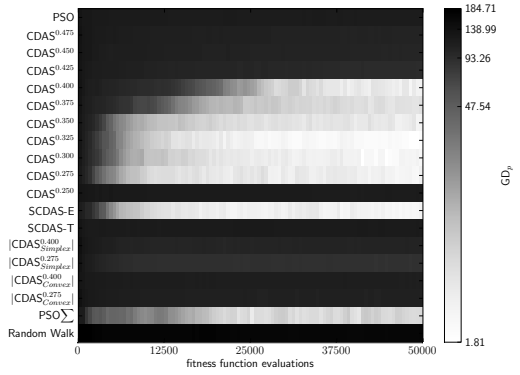
(j) DTLZ1 Median IGD on 15 objectives



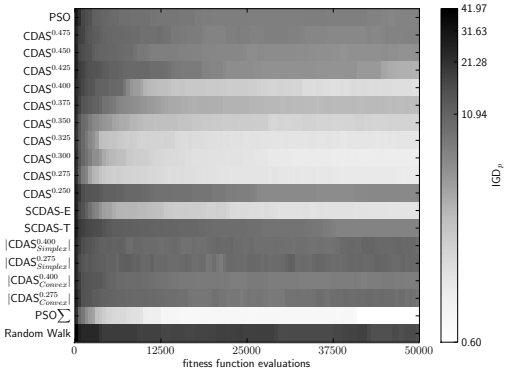
(k) DTLZ1 Median GD on 20 objectives



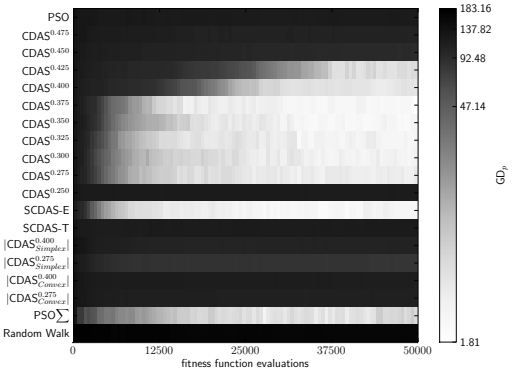
(l) DTLZ1 Median IGD on 20 objectives



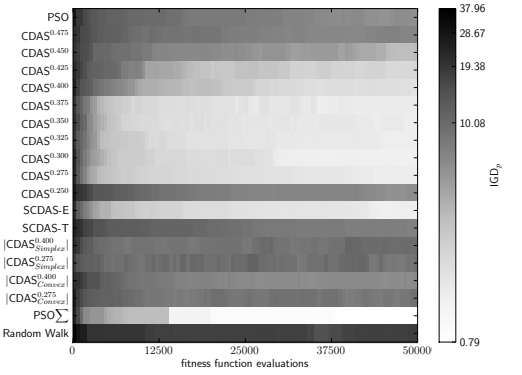
(m) DTLZ1 Median GD on 25 objectives



(n) DTLZ1 Median IGD on 25 objectives

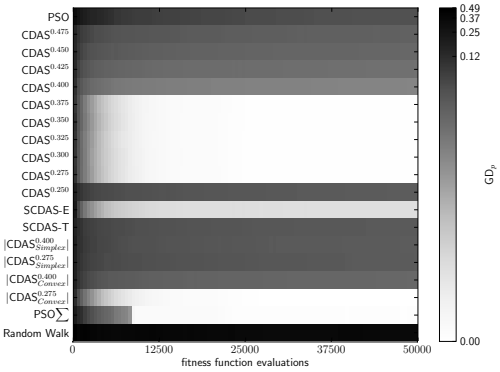


(o) DTLZ1 Median GD on 30 objectives

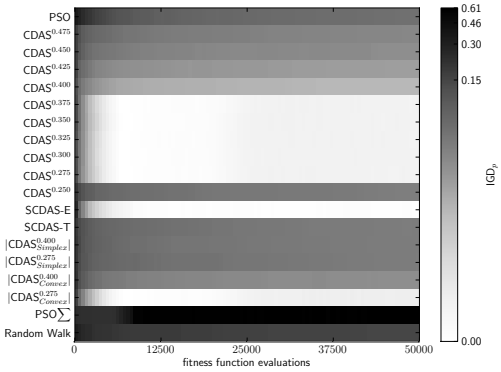


(p) DTLZ1 Median IGD on 30 objectives

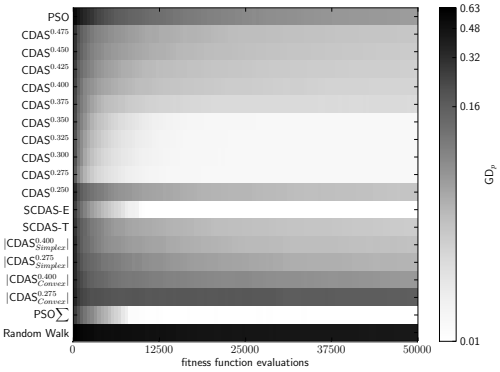
Figure A.1.: DTLZ1.



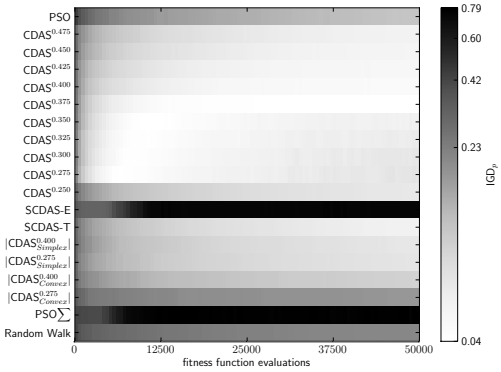
(a) DTLZ2 Median GD on 2 objectives



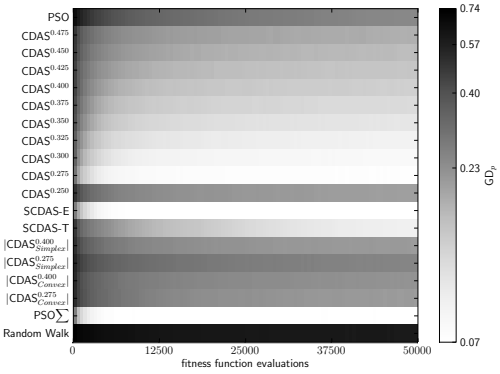
(b) DTLZ2 Median IGD on 2 objectives



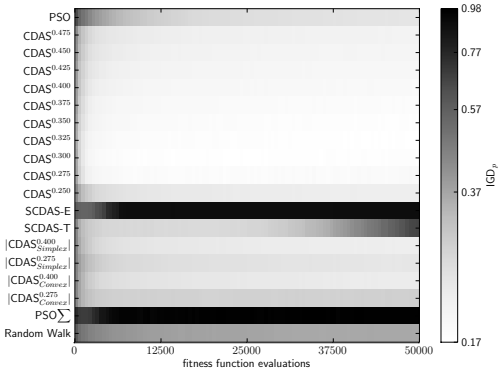
(c) DTLZ2 Median GD on 3 objectives



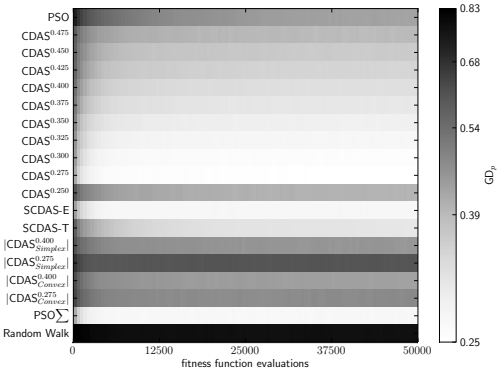
(d) DTLZ2 Median IGD on 3 objectives



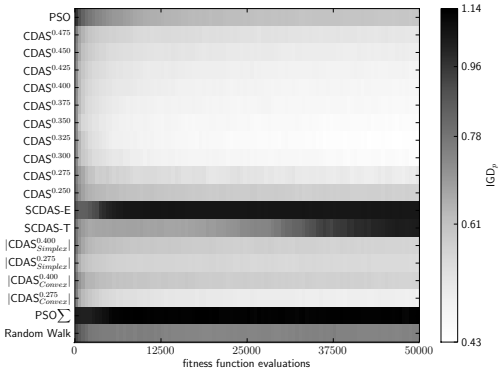
(e) DTLZ2 Median GD on 5 objectives



(f) DTLZ2 Median IGD on 5 objectives

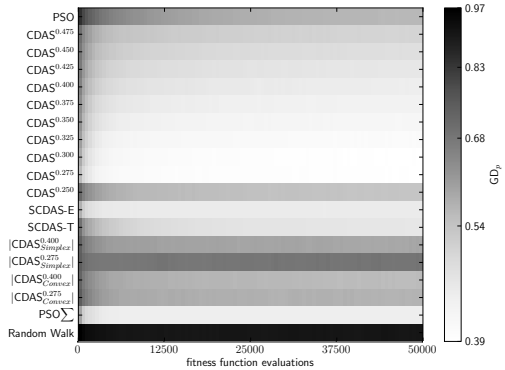


(g) DTLZ2 Median GD on 10 objectives

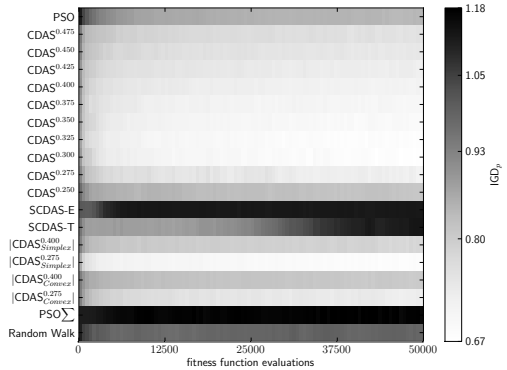


(h) DTLZ2 Median IGD on 10 objectives

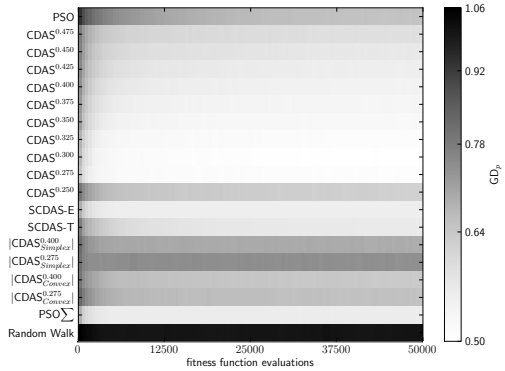
Figure A.2.: DTLZ2.



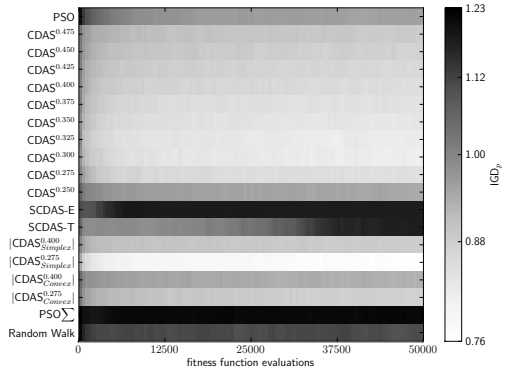
(i) DTLZ2 Median GD on 15 objectives



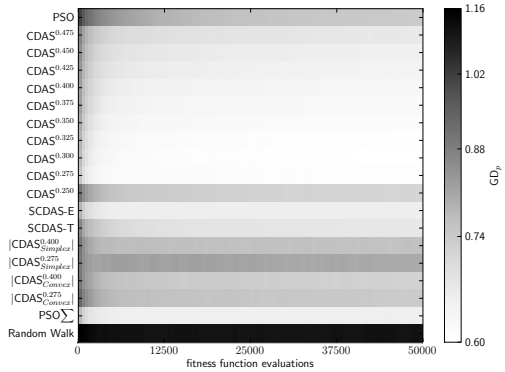
(j) DTLZ2 Median IGD on 15 objectives



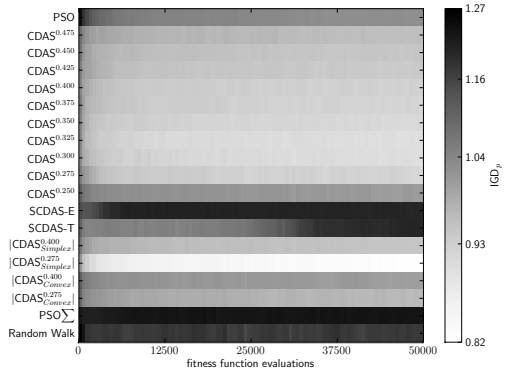
(k) DTLZ2 Median GD on 20 objectives



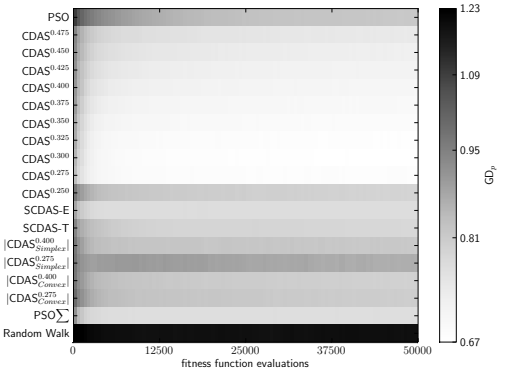
(l) DTLZ2 Median IGD on 20 objectives



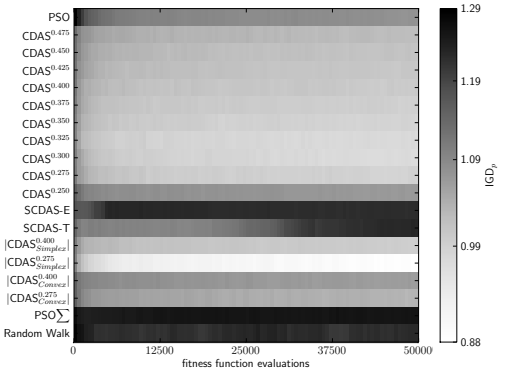
(m) DTLZ2 Median GD on 25 objectives



(n) DTLZ2 Median IGD on 25 objectives

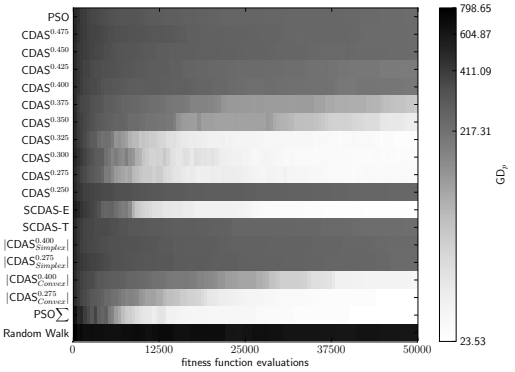


(o) DTLZ2 Median GD on 30 objectives

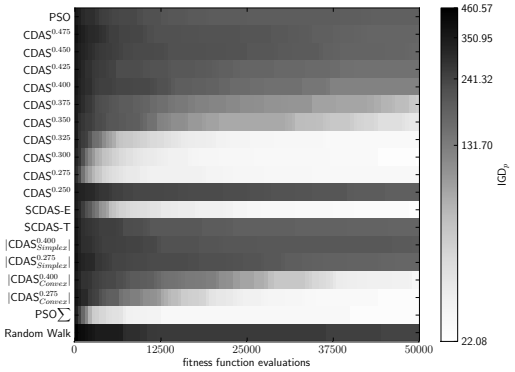


(p) DTLZ2 Median IGD on 30 objectives

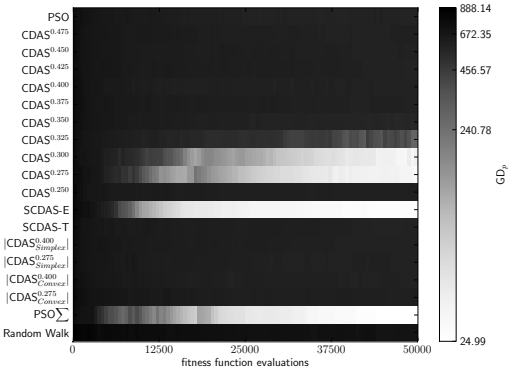
Figure A.2.: DTLZ2.



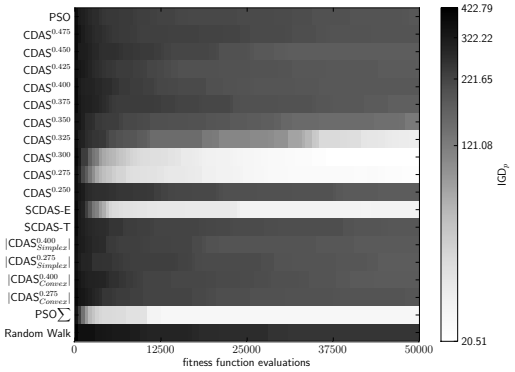
(a) DTLZ3 Median GD on 2 objectives



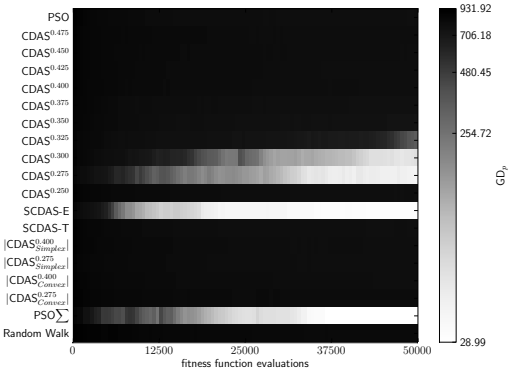
(b) DTLZ3 Median IGD on 2 objectives



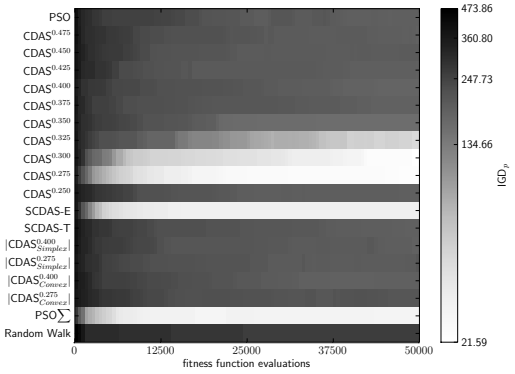
(c) DTLZ3 Median GD on 3 objectives



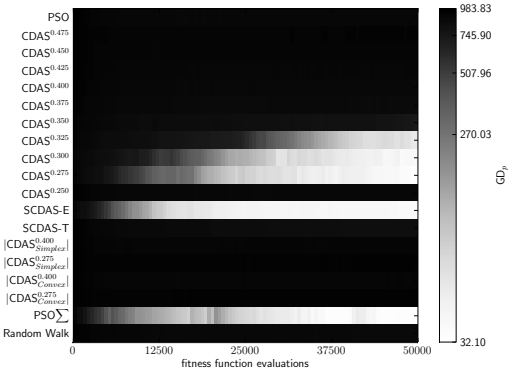
(d) DTLZ3 Median IGD on 3 objectives



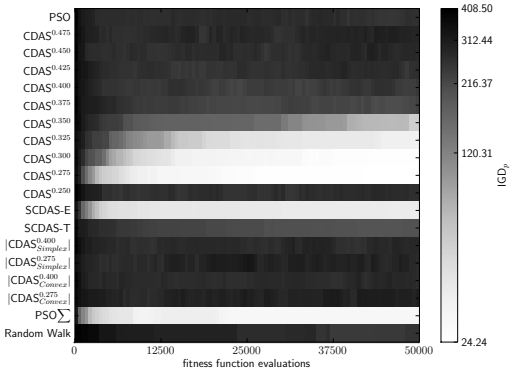
(e) DTLZ3 Median GD on 5 objectives



(f) DTLZ3 Median IGD on 5 objectives

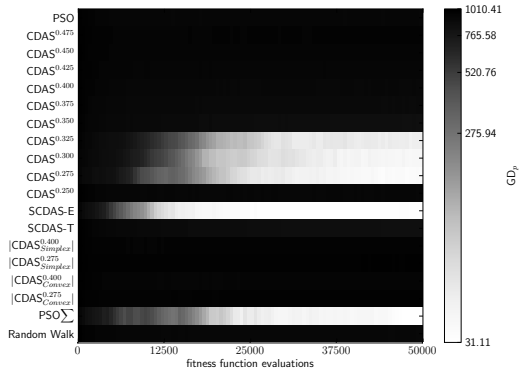


(g) DTLZ3 Median GD on 10 objectives

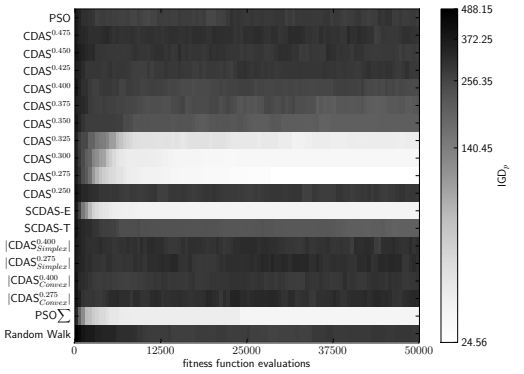


(h) DTLZ3 Median IGD on 10 objectives

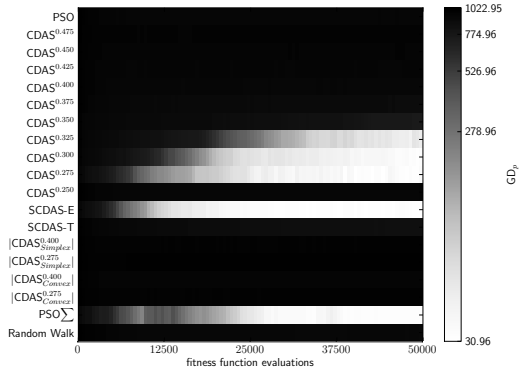
Figure A.3.: DTLZ3.



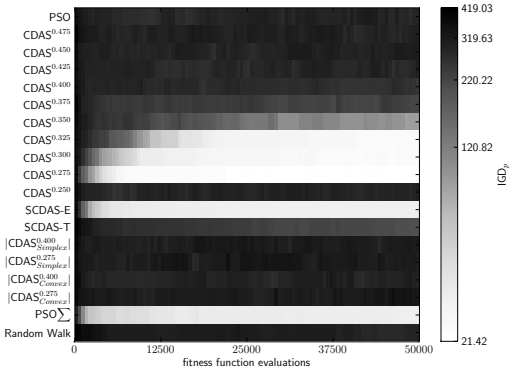
(i) DTLZ3 Median GD on 15 objectives



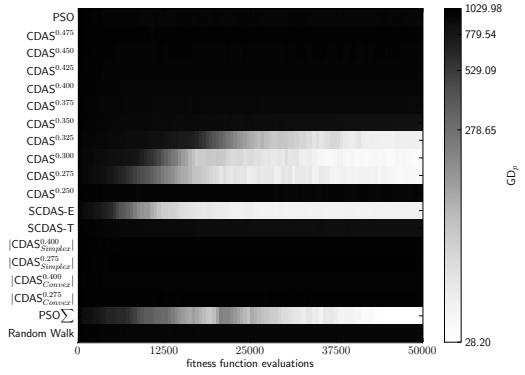
(j) DTLZ3 Median IGD on 15 objectives



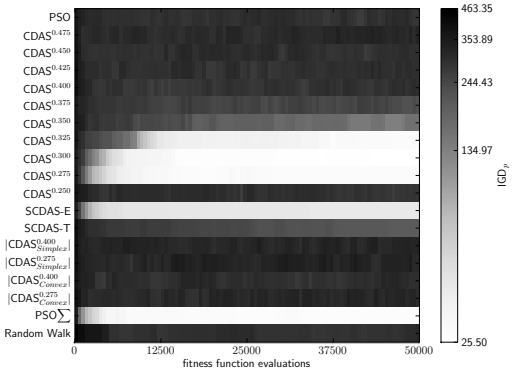
(k) DTLZ3 Median GD on 20 objectives



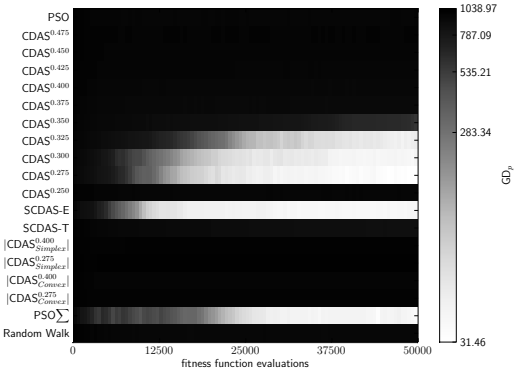
(l) DTLZ3 Median IGD on 20 objectives



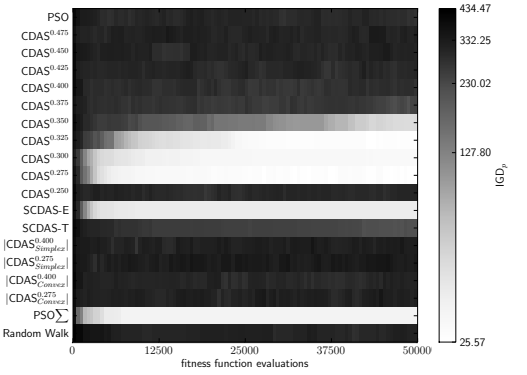
(m) DTLZ3 Median GD on 25 objectives



(n) DTLZ3 Median IGD on 25 objectives

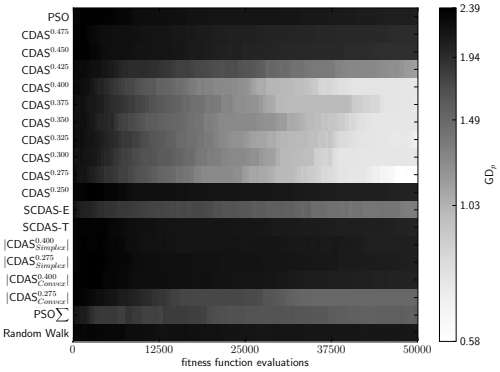


(o) DTLZ3 Median GD on 30 objectives

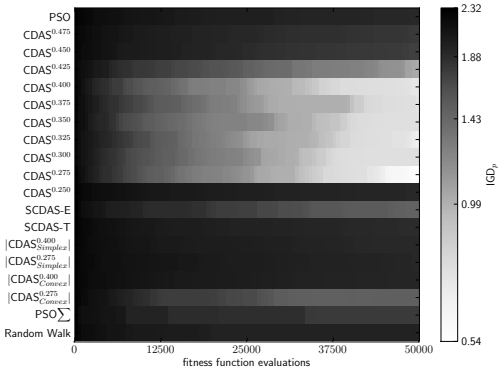


(p) DTLZ3 Median IGD on 30 objectives

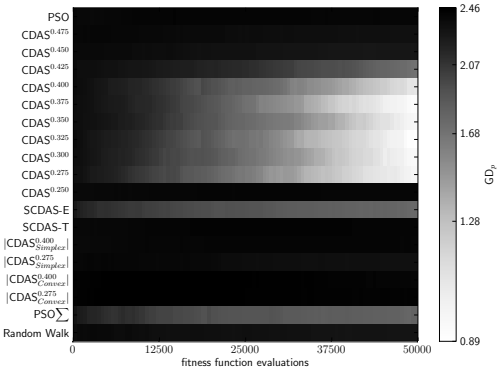
Figure A.3.: DTLZ3.



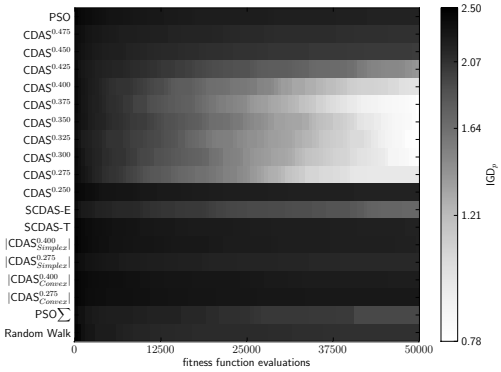
(a) DTLZ4 Median GD on 2 objectives



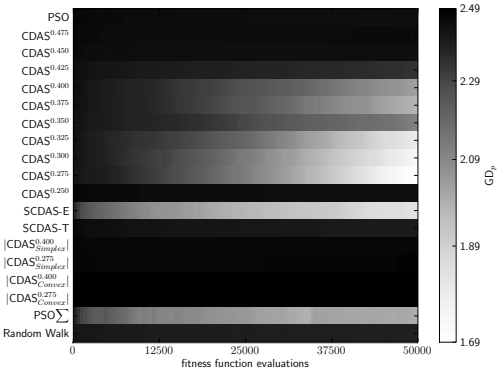
(b) DTLZ4 Median IGD on 2 objectives



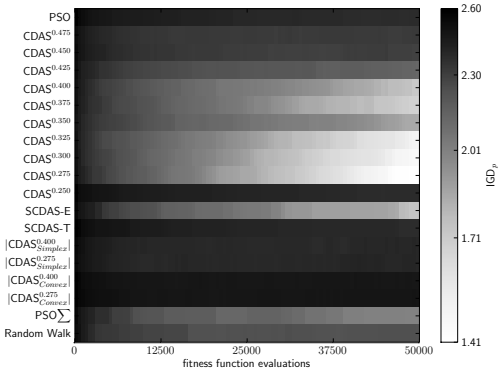
(c) DTLZ4 Median GD on 3 objectives



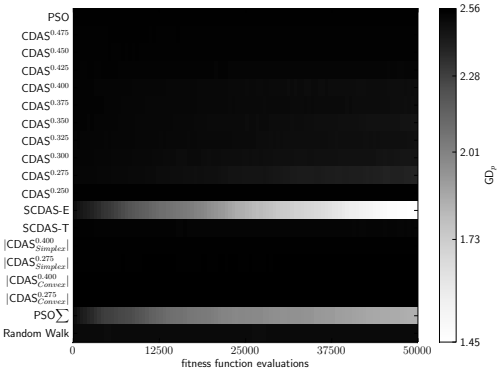
(d) DTLZ4 Median IGD on 3 objectives



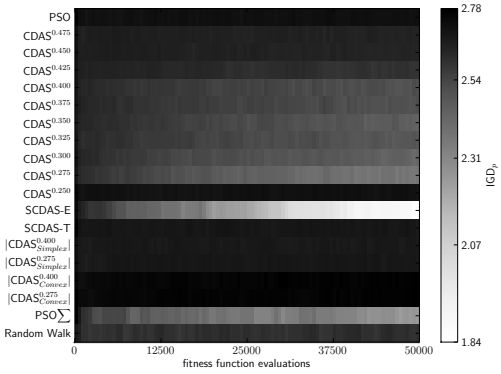
(e) DTLZ4 Median GD on 5 objectives



(f) DTLZ4 Median IGD on 5 objectives

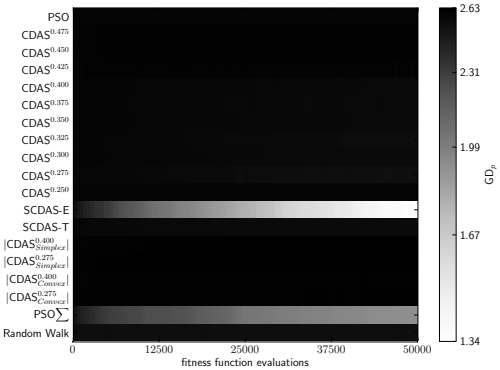


(g) DTLZ4 Median GD on 10 objectives

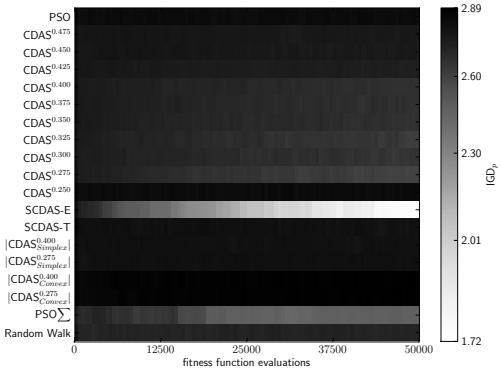


(h) DTLZ4 Median IGD on 10 objectives

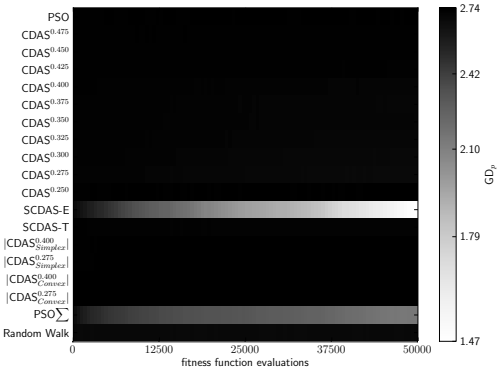
Figure A.4.: DTLZ4.



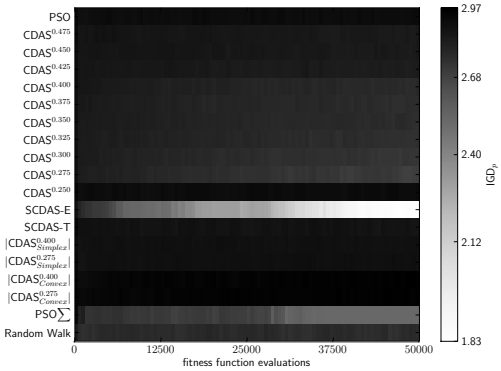
(i) DTLZ4 Median GD on 15 objectives



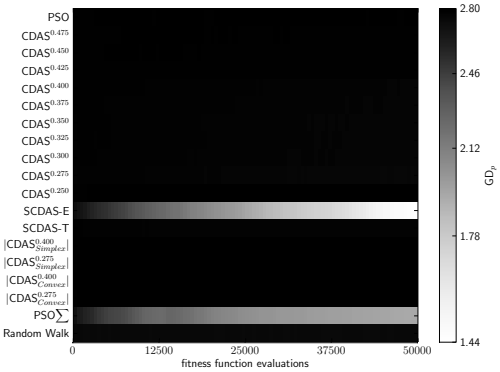
(j) DTLZ4 Median IGD on 15 objectives



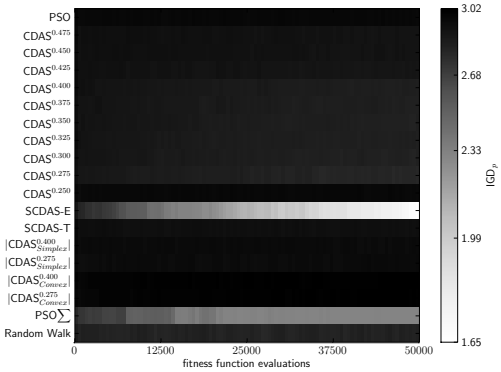
(k) DTLZ4 Median GD on 20 objectives



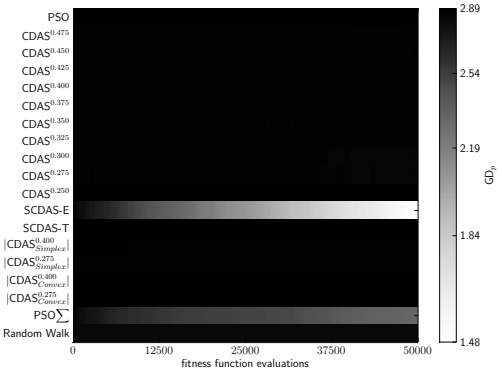
(l) DTLZ4 Median IGD on 20 objectives



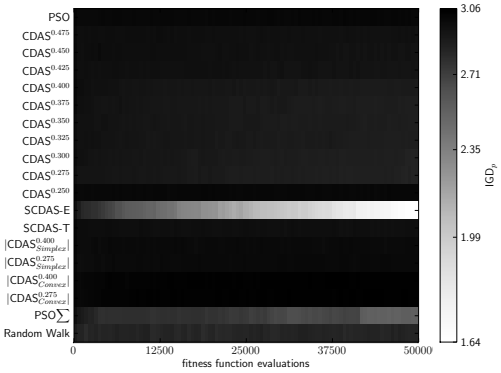
(m) DTLZ4 Median GD on 25 objectives



(n) DTLZ4 Median IGD on 25 objectives

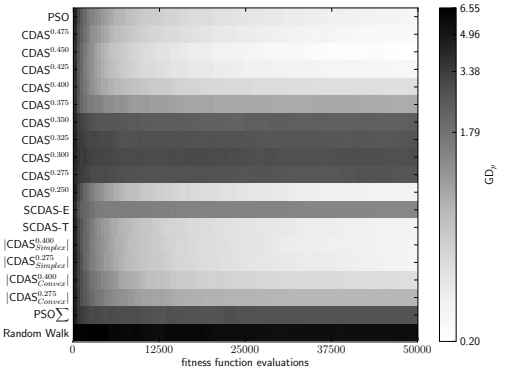


(o) DTLZ4 Median GD on 30 objectives

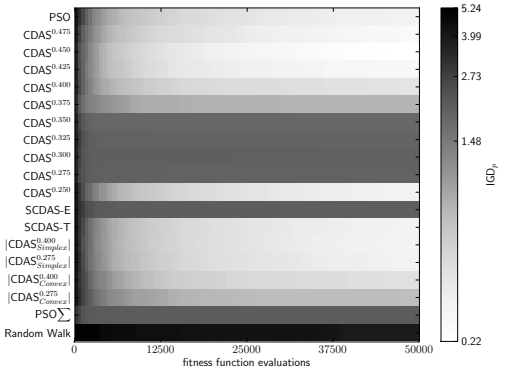


(p) DTLZ4 Median IGD on 30 objectives

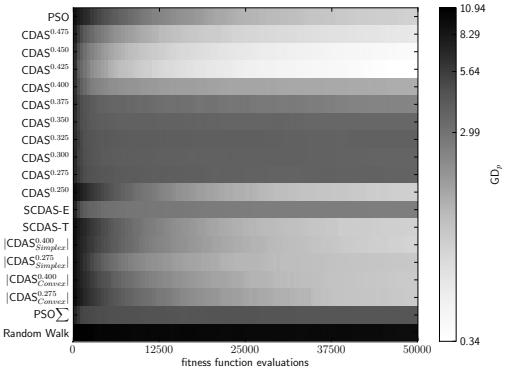
Figure A.4.: DTLZ4.



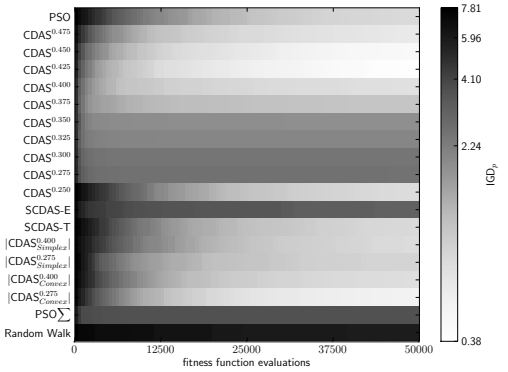
(a) DTLZ6 Median GD on 2 objectives



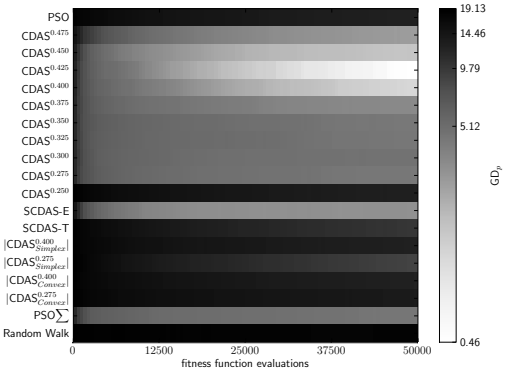
(b) DTLZ6 Median IGD on 2 objectives



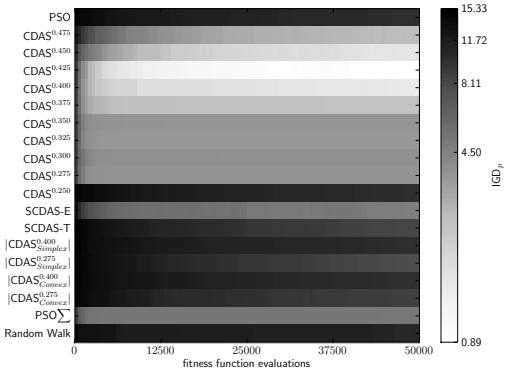
(c) DTLZ6 Median GD on 3 objectives



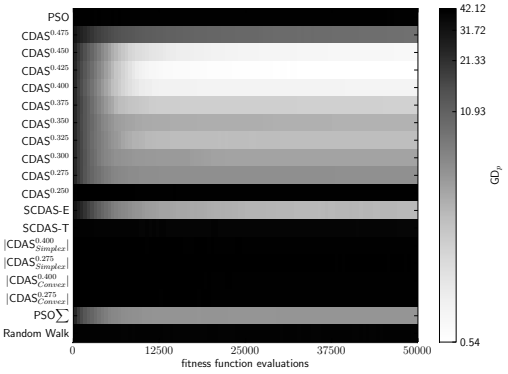
(d) DTLZ6 Median IGD on 3 objectives



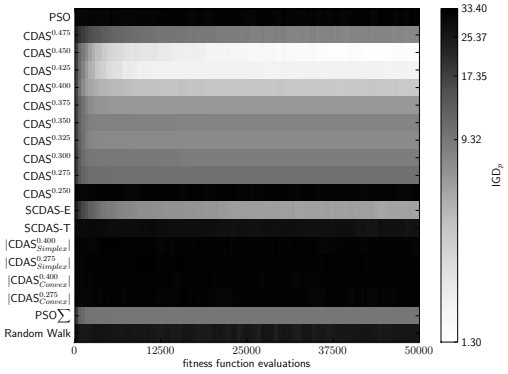
(e) DTLZ6 Median GD on 5 objectives



(f) DTLZ6 Median IGD on 5 objectives

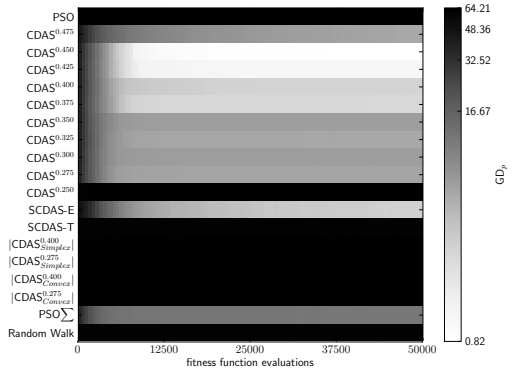


(g) DTLZ6 Median GD on 10 objectives

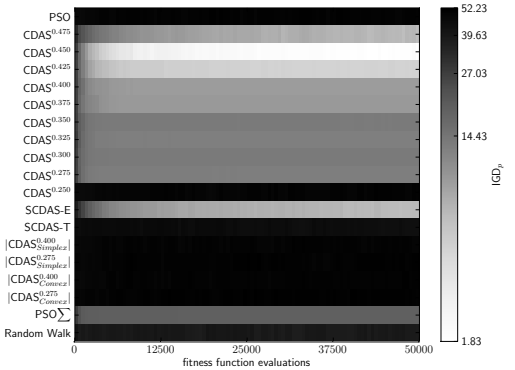


(h) DTLZ6 Median IGD on 10 objectives

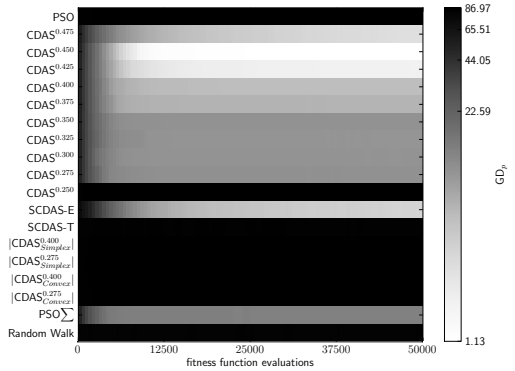
Figure A.5.: DTLZ6.



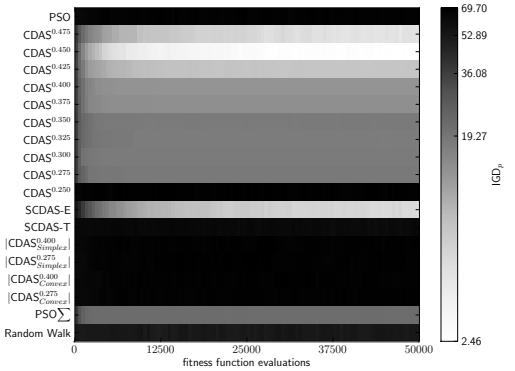
(i) DTLZ6 Median GD on 15 objectives



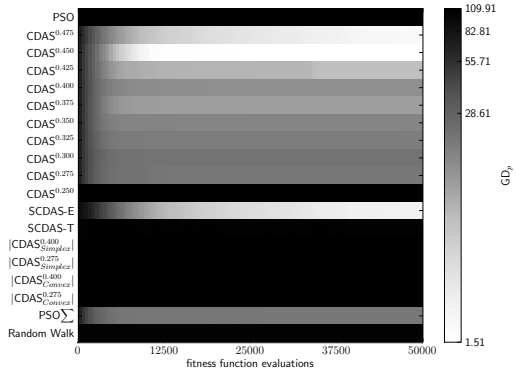
(j) DTLZ6 Median IGD on 15 objectives



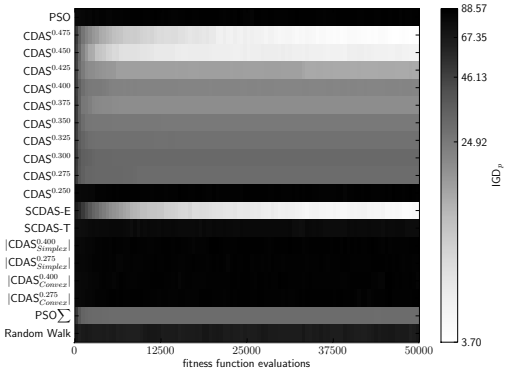
(k) DTLZ6 Median GD on 20 objectives



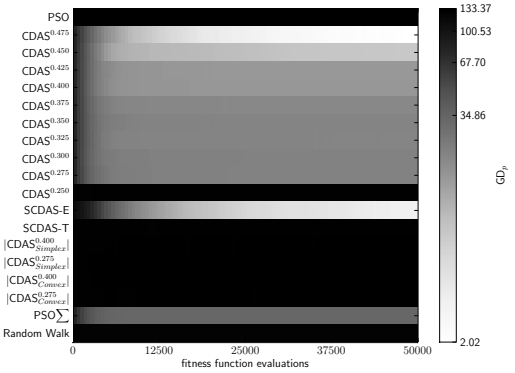
(l) DTLZ6 Median IGD on 20 objectives



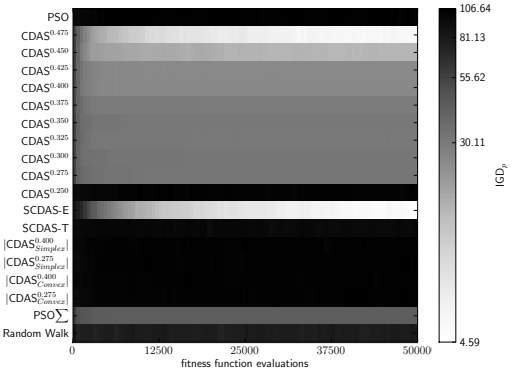
(m) DTLZ6 Median GD on 25 objectives



(n) DTLZ6 Median IGD on 25 objectives

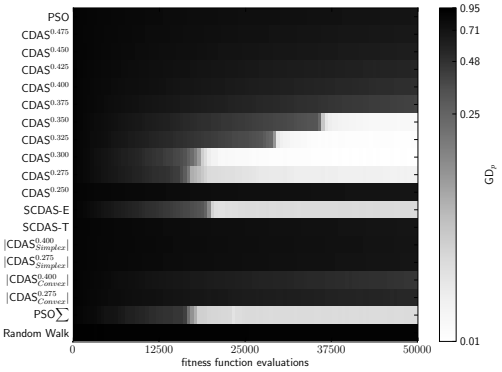


(o) DTLZ6 Median GD on 30 objectives

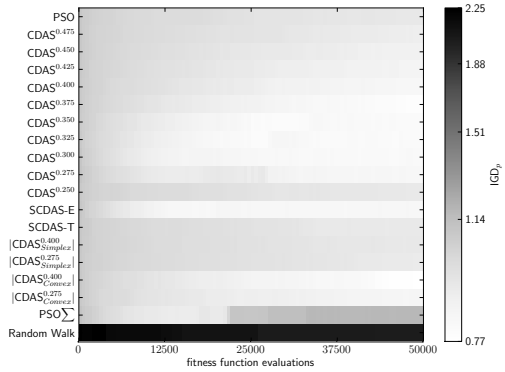


(p) DTLZ6 Median IGD on 30 objectives

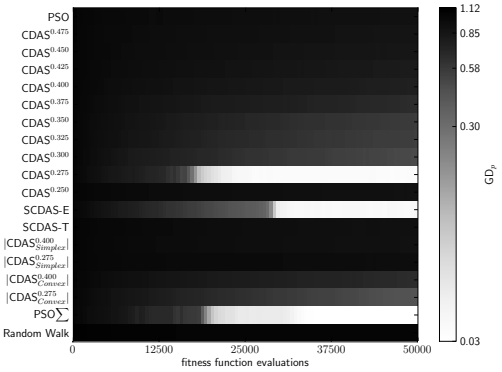
Figure A.5.: DTLZ6.



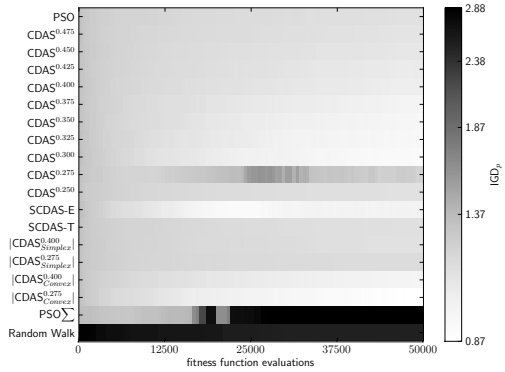
(a) WFG1 Median GD on 2 objectives



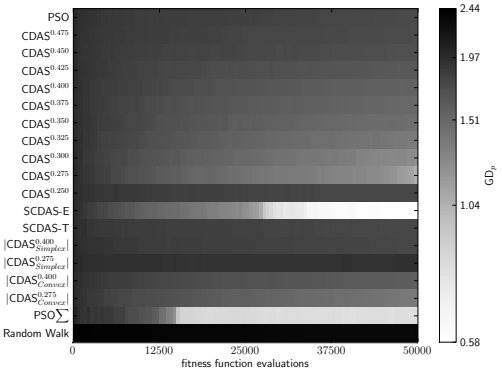
(b) WFG1 Median IGD on 2 objectives



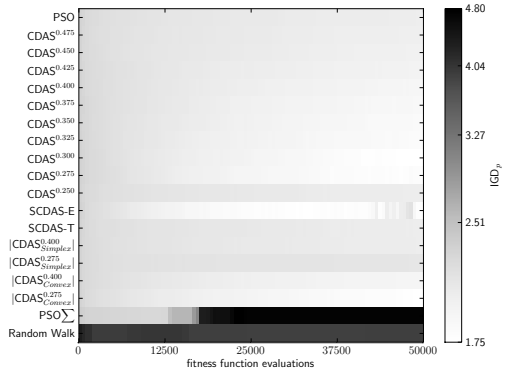
(c) WFG1 Median GD on 3 objectives



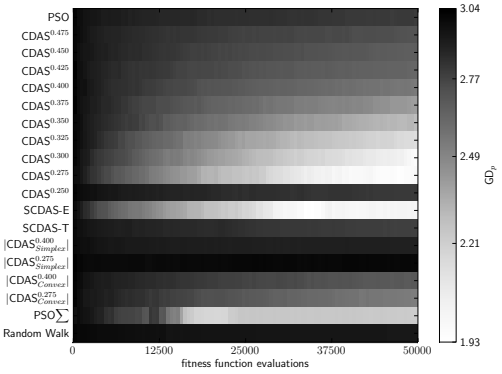
(d) WFG1 Median IGD on 3 objectives



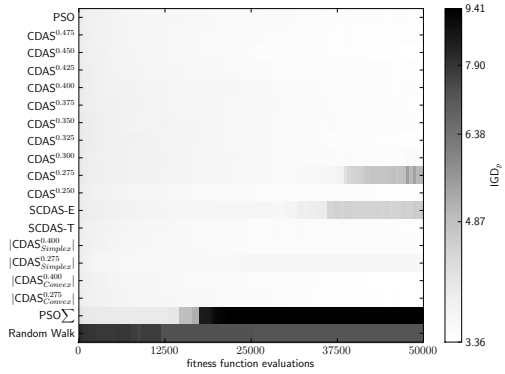
(e) WFG1 Median GD on 5 objectives



(f) WFG1 Median IGD on 5 objectives

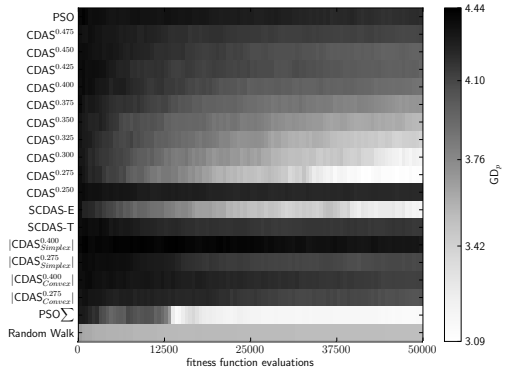


(g) WFG1 Median GD on 10 objectives

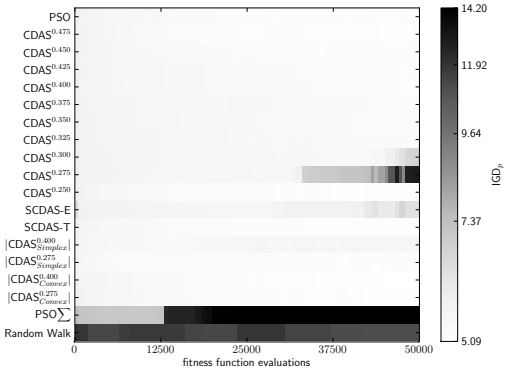


(h) WFG1 Median IGD on 10 objectives

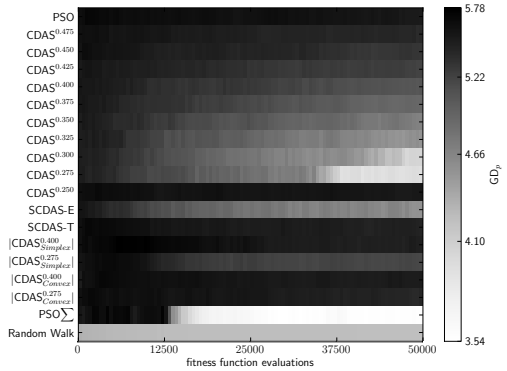
Figure A.6.: WFG1.



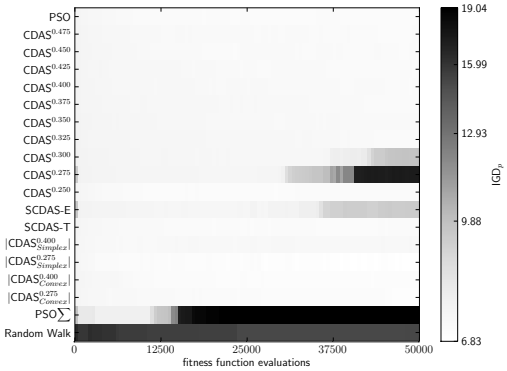
(i) WFG1 Median GD on 15 objectives



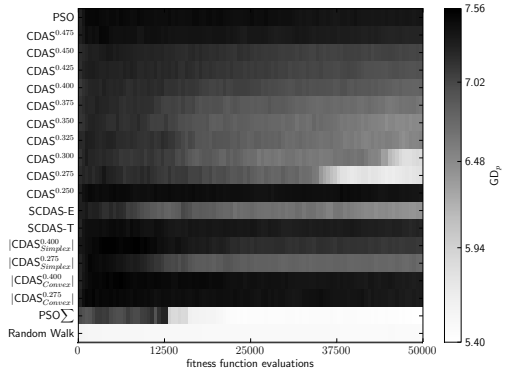
(j) WFG1 Median IGD on 15 objectives



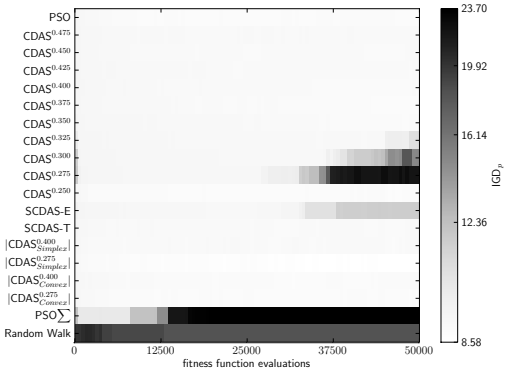
(k) WFG1 Median GD on 20 objectives



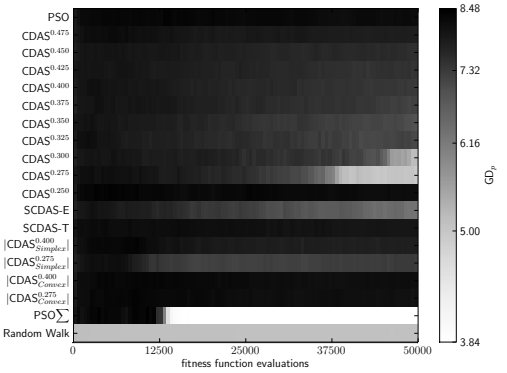
(l) WFG1 Median IGD on 20 objectives



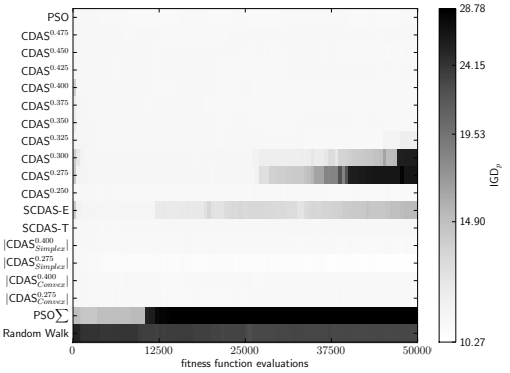
(m) WFG1 Median GD on 25 objectives



(n) WFG1 Median IGD on 25 objectives

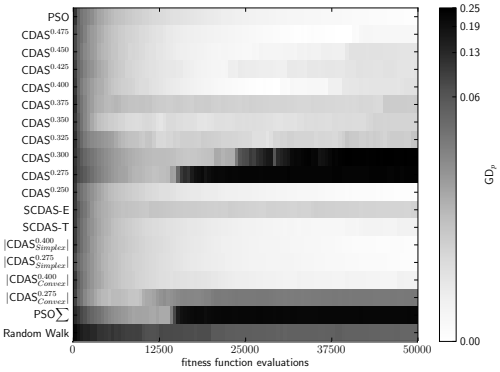


(o) WFG1 Median GD on 30 objectives

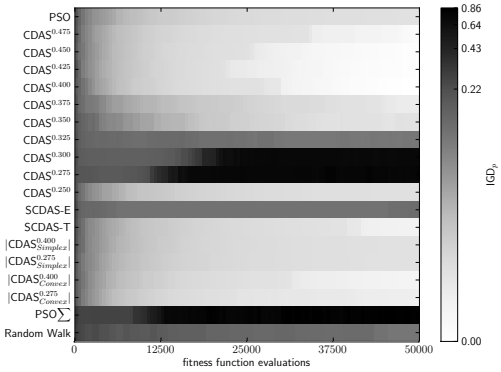


(p) WFG1 Median IGD on 30 objectives

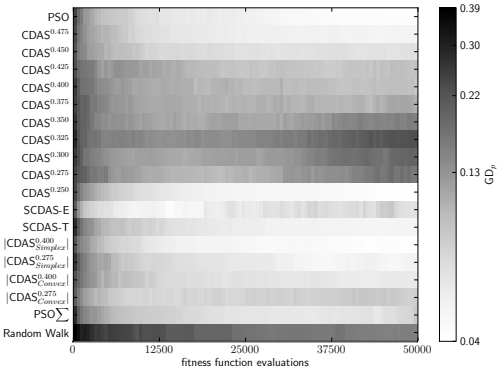
Figure A.6.: WFG1.



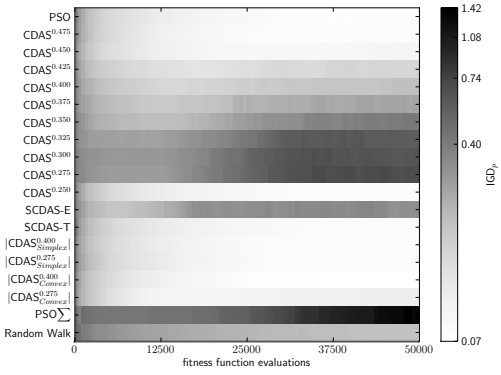
(a) WFG2 Median GD on 2 objectives



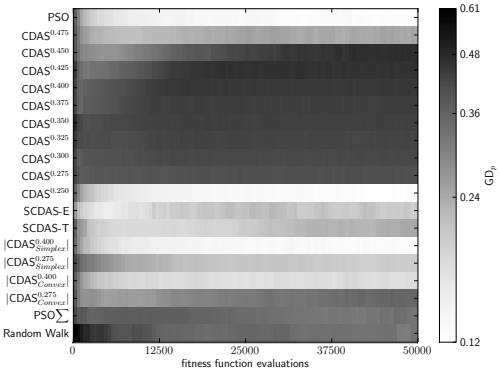
(b) WFG2 Median IGD on 2 objectives



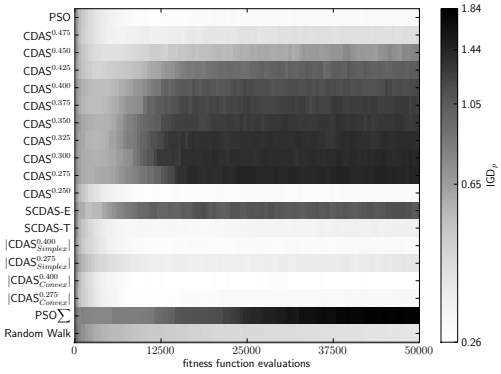
(c) WFG2 Median GD on 3 objectives



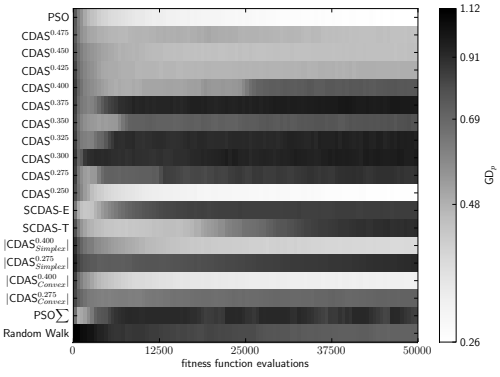
(d) WFG2 Median IGD on 3 objectives



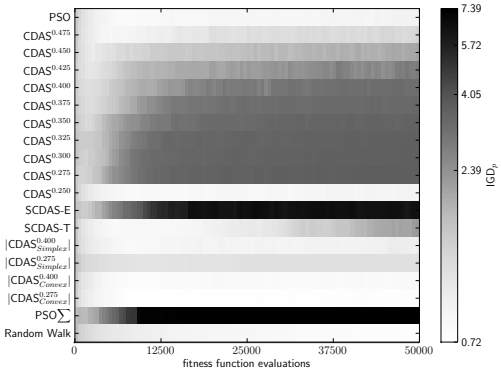
(e) WFG2 Median GD on 5 objectives



(f) WFG2 Median IGD on 5 objectives

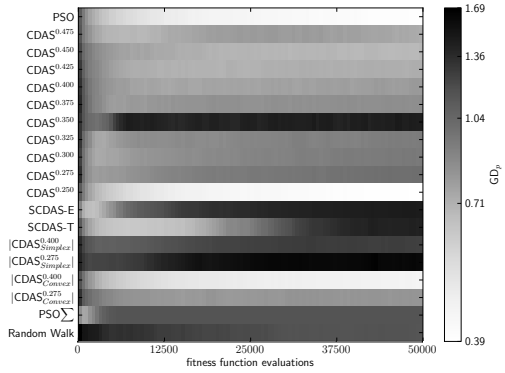


(g) WFG2 Median GD on 10 objectives

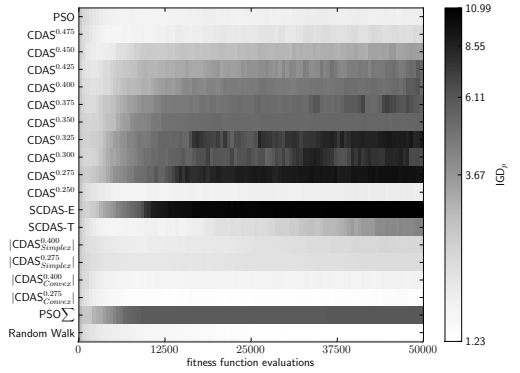


(h) WFG2 Median IGD on 10 objectives

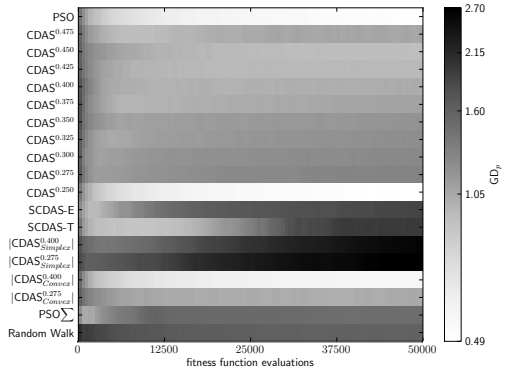
Figure A.7.: WFG2.



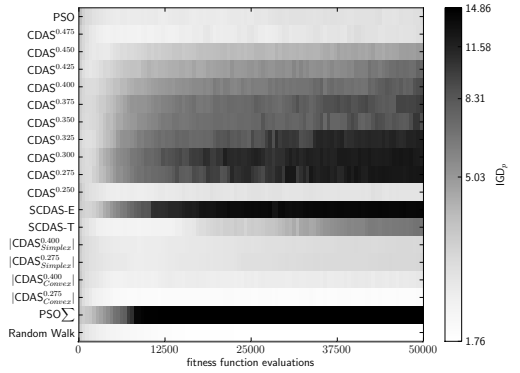
(i) WFG2 Median GD on 15 objectives



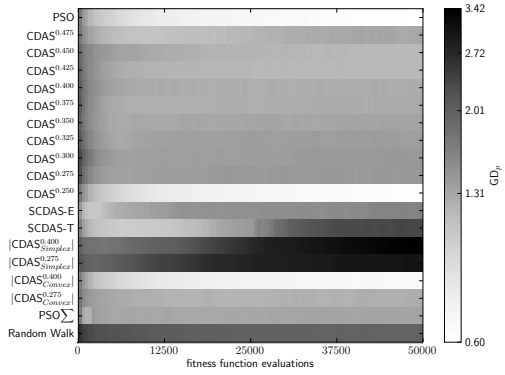
(j) WFG2 Median IGD on 15 objectives



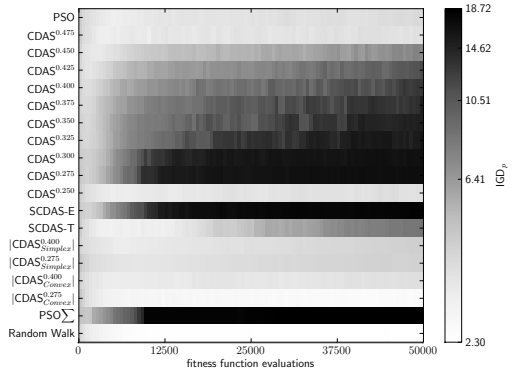
(k) WFG2 Median GD on 20 objectives



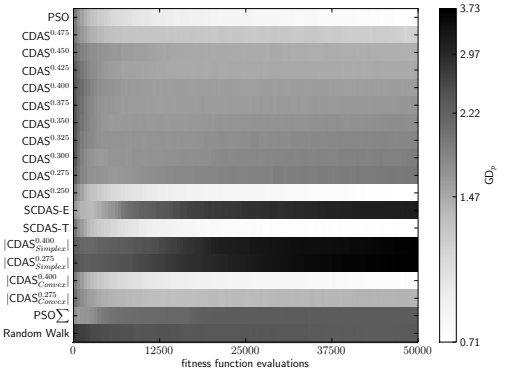
(l) WFG2 Median IGD on 20 objectives



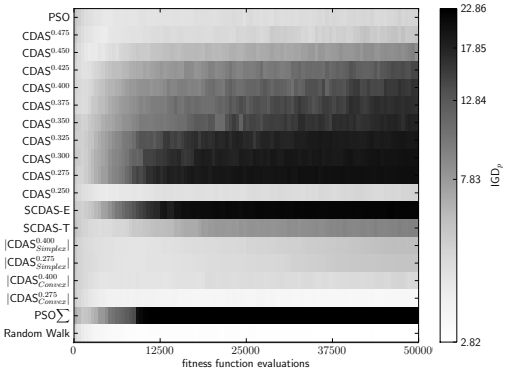
(m) WFG2 Median GD on 25 objectives



(n) WFG2 Median IGD on 25 objectives

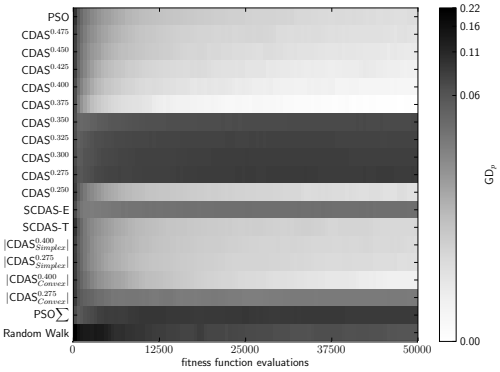


(o) WFG2 Median GD on 30 objectives

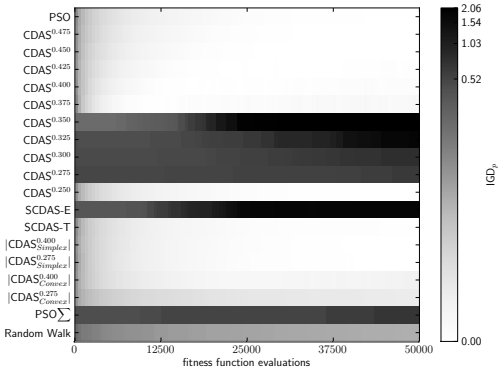


(p) WFG2 Median IGD on 30 objectives

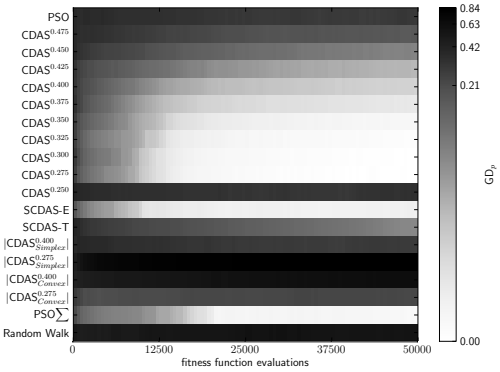
Figure A.7.: WFG2.



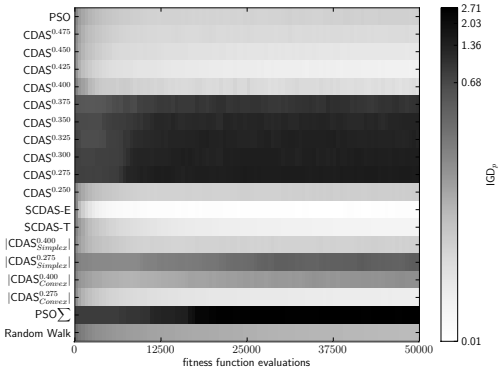
(a) WFG3 Median GD on 2 objectives



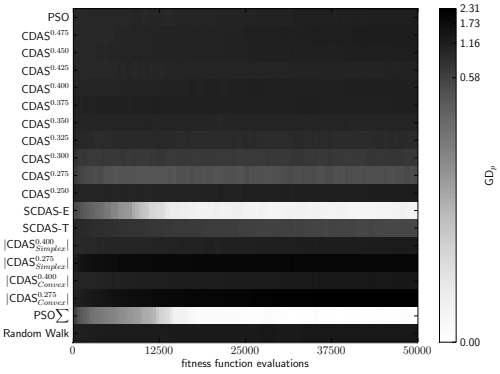
(b) WFG3 Median IGD on 2 objectives



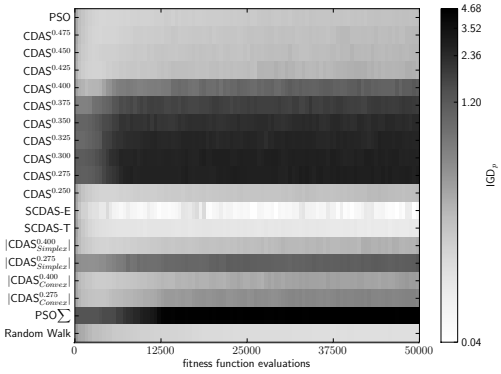
(c) WFG3 Median GD on 3 objectives



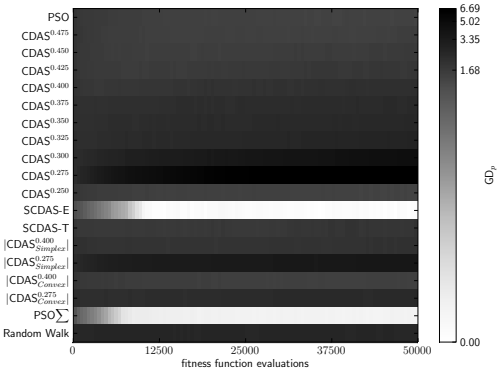
(d) WFG3 Median IGD on 3 objectives



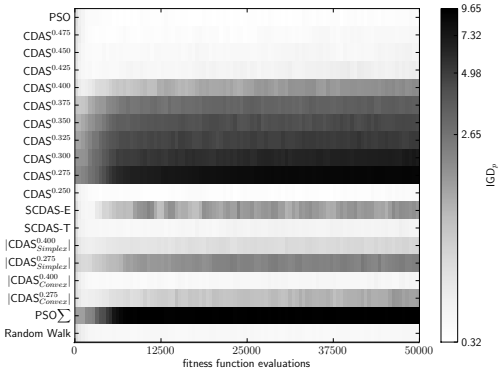
(e) WFG3 Median GD on 5 objectives



(f) WFG3 Median IGD on 5 objectives

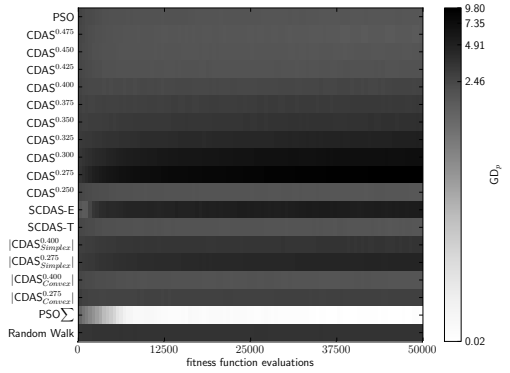


(g) WFG3 Median GD on 10 objectives

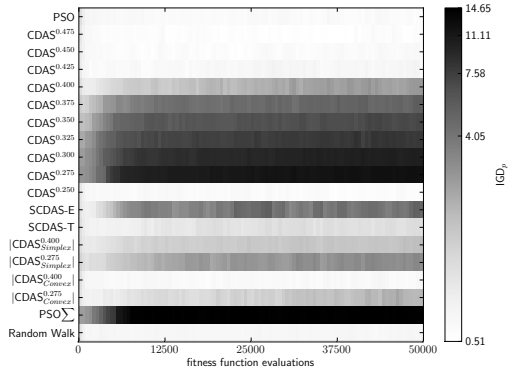


(h) WFG3 Median IGD on 10 objectives

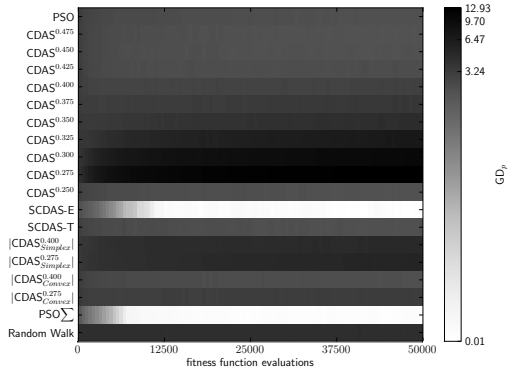
Figure A.8.: WFG3.



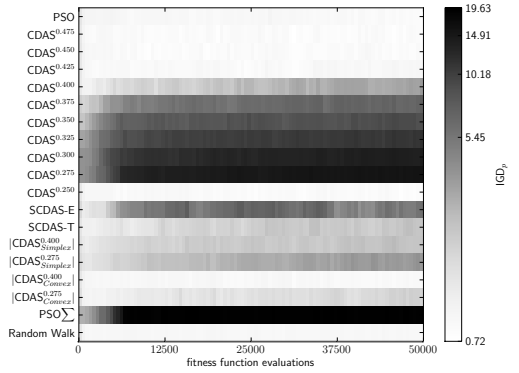
(i) WFG3 Median GD on 15 objectives



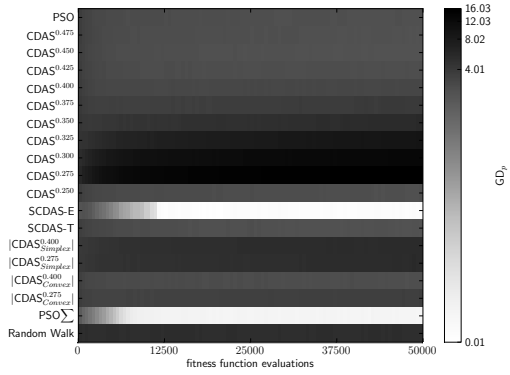
(j) WFG3 Median IGD on 15 objectives



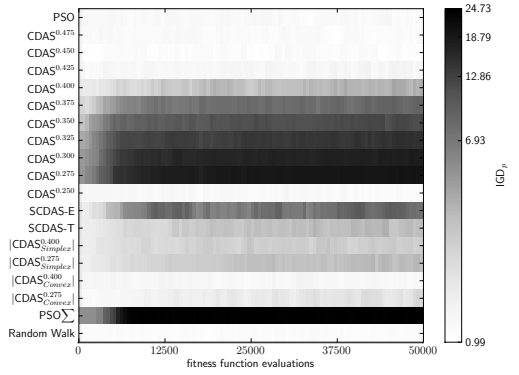
(k) WFG3 Median GD on 20 objectives



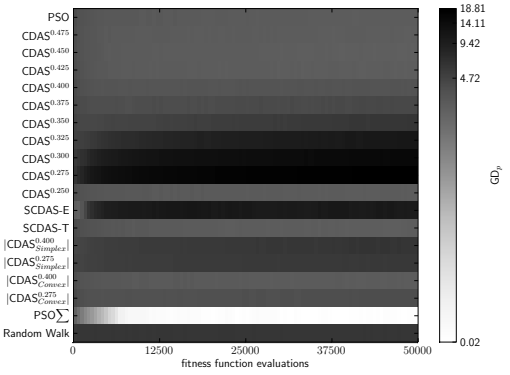
(l) WFG3 Median IGD on 20 objectives



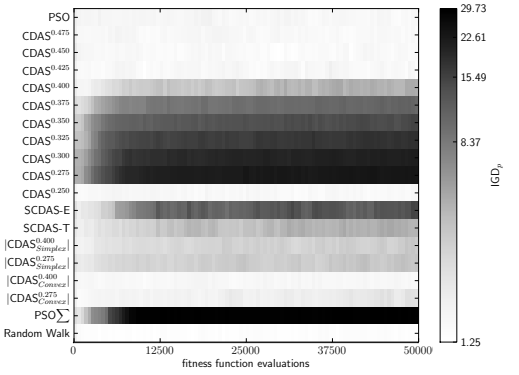
(m) WFG3 Median GD on 25 objectives



(n) WFG3 Median IGD on 25 objectives

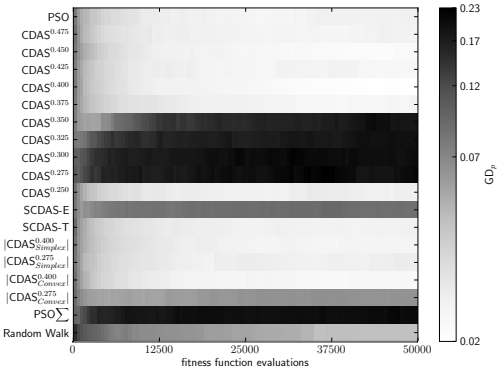


(o) WFG3 Median GD on 30 objectives

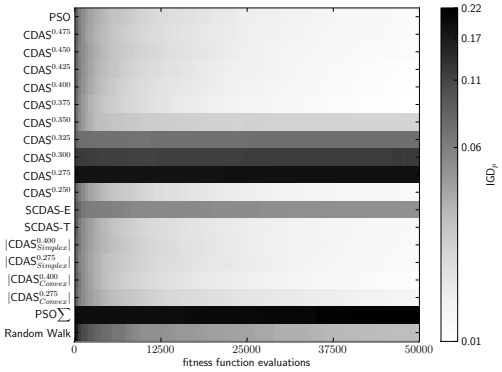


(p) WFG3 Median IGD on 30 objectives

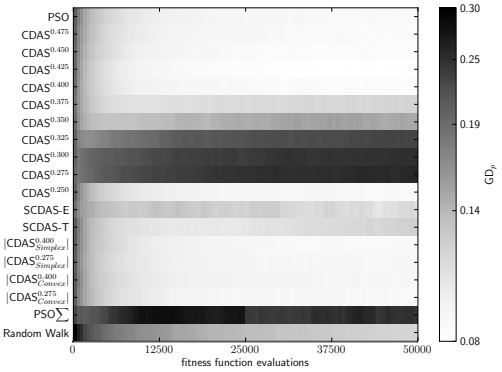
Figure A.8.: WFG3.



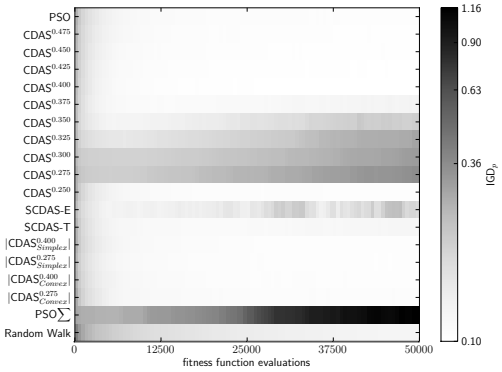
(a) WFG4 Median GD on 2 objectives



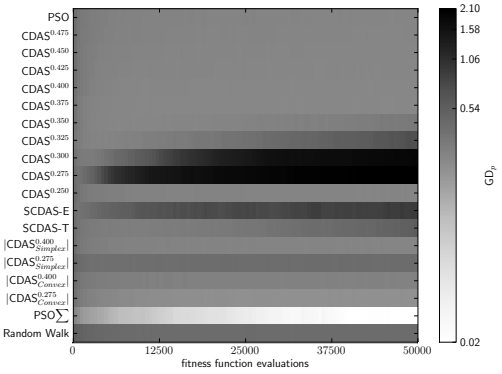
(b) WFG4 Median IGD on 2 objectives



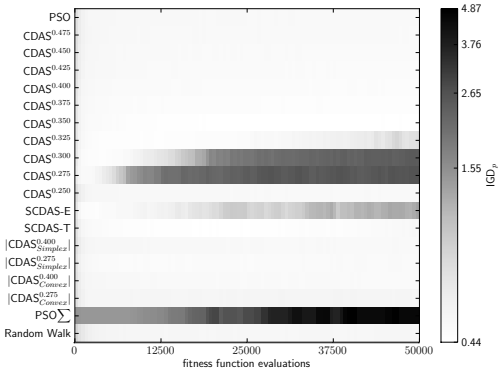
(c) WFG4 Median GD on 3 objectives



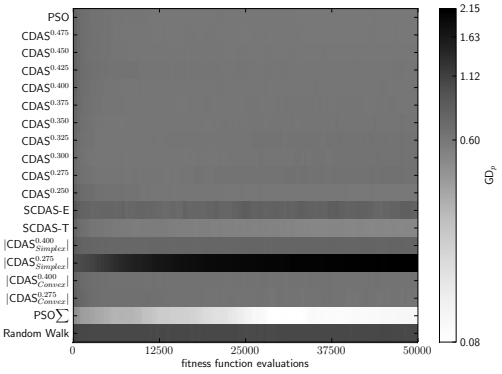
(d) WFG4 Median IGD on 3 objectives



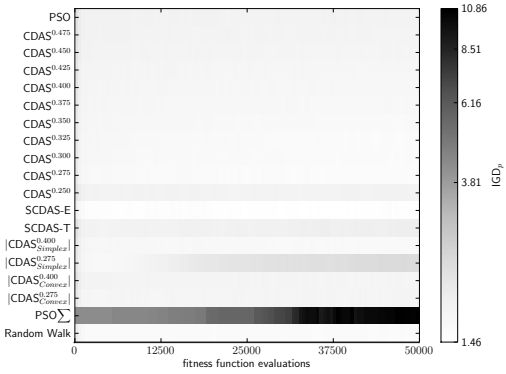
(e) WFG4 Median GD on 5 objectives



(f) WFG4 Median IGD on 5 objectives

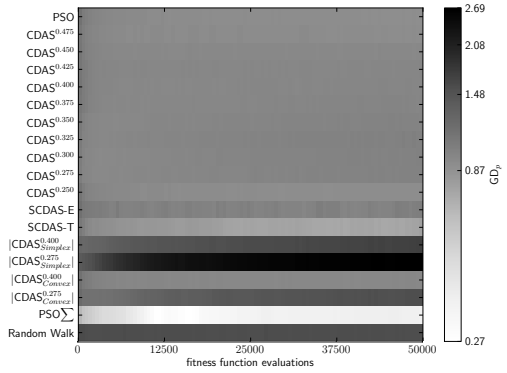


(g) WFG4 Median GD on 10 objectives

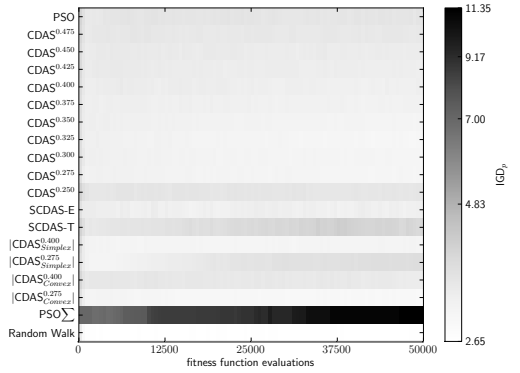


(h) WFG4 Median IGD on 10 objectives

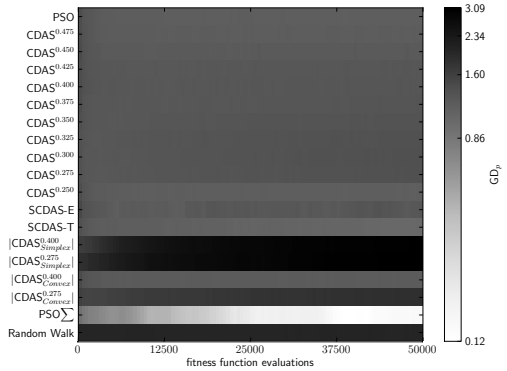
Figure A.9.: WFG4.



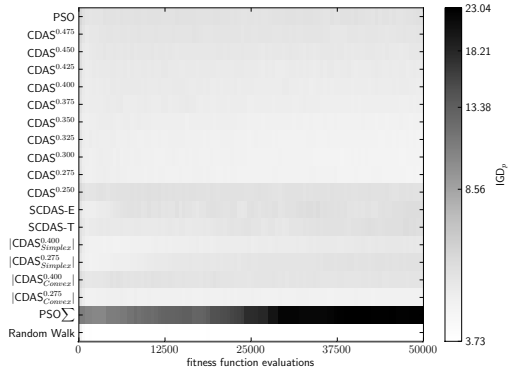
(i) WFG4 Median GD on 15 objectives



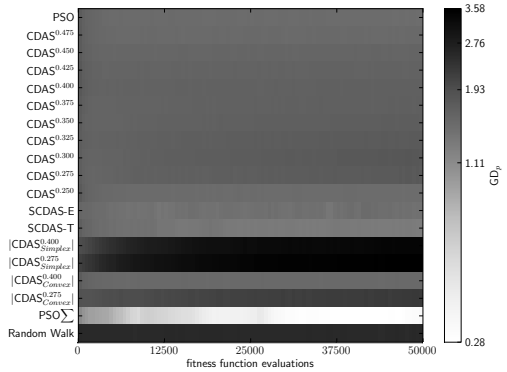
(j) WFG4 Median IGD on 15 objectives



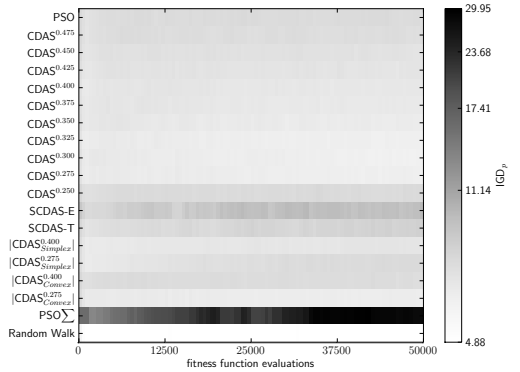
(k) WFG4 Median GD on 20 objectives



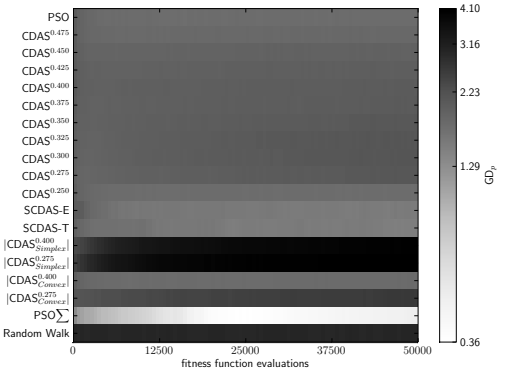
(l) WFG4 Median IGD on 20 objectives



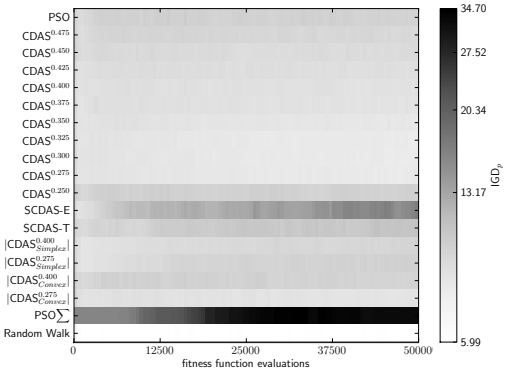
(m) WFG4 Median GD on 25 objectives



(n) WFG4 Median IGD on 25 objectives

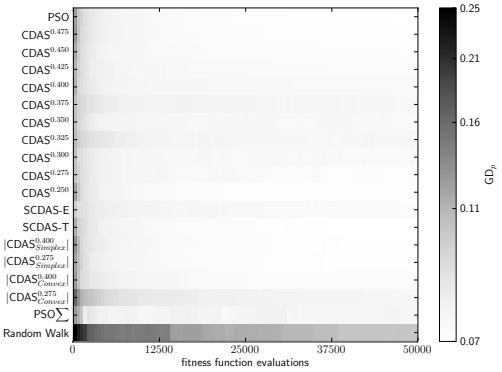


(o) WFG4 Median GD on 30 objectives

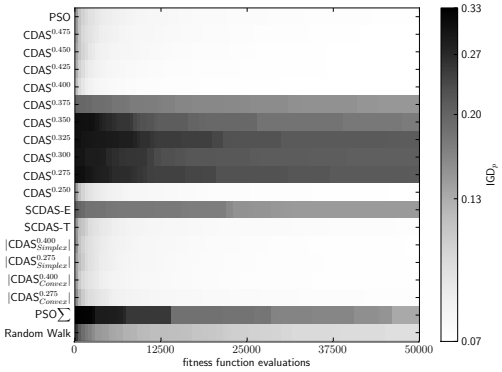


(p) WFG4 Median IGD on 30 objectives

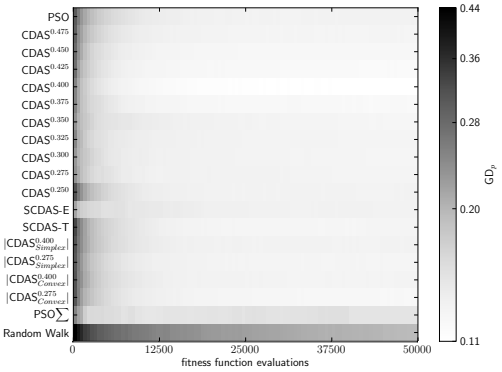
Figure A.9.: WFG4.



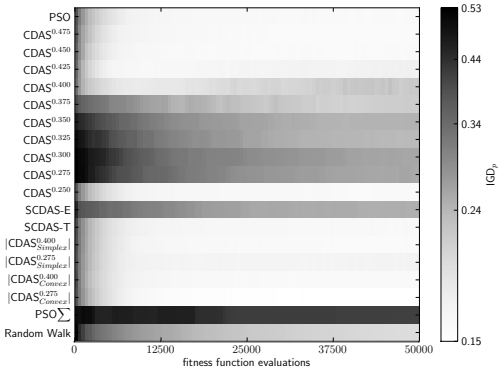
(a) WFG5 Median GD on 2 objectives



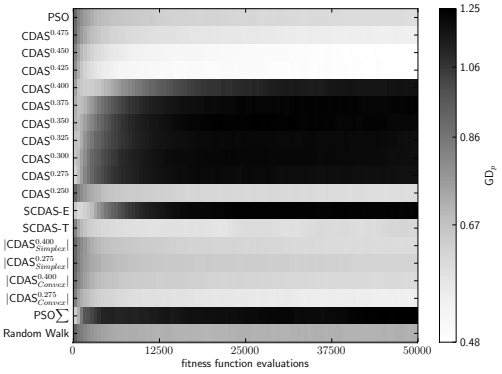
(b) WFG5 Median IGD on 2 objectives



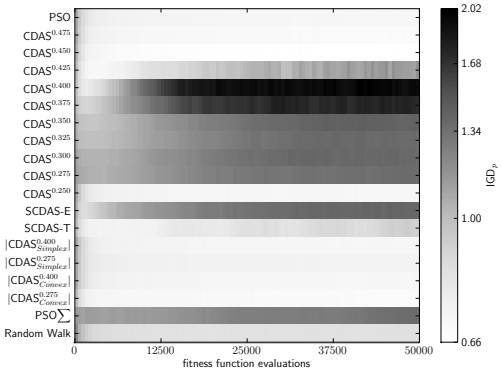
(c) WFG5 Median GD on 3 objectives



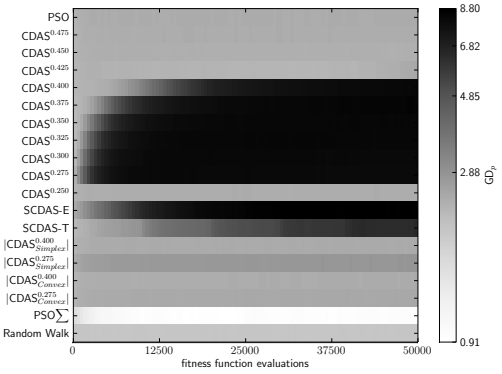
(d) WFG5 Median IGD on 3 objectives



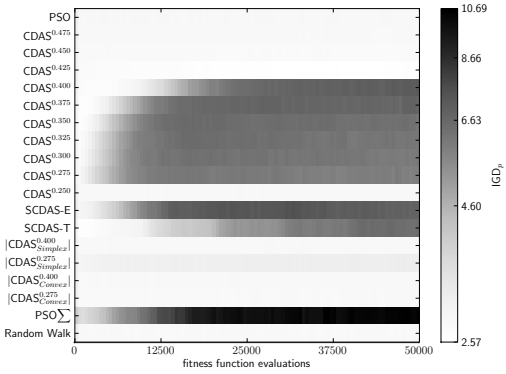
(e) WFG5 Median GD on 5 objectives



(f) WFG5 Median IGD on 5 objectives

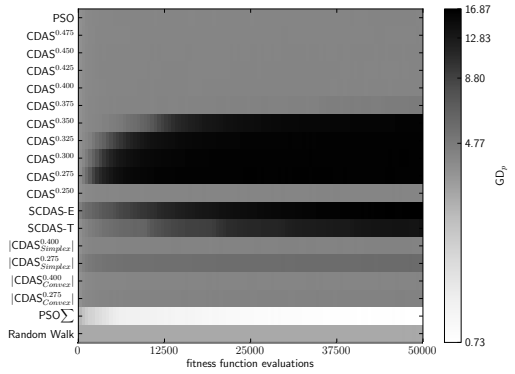


(g) WFG5 Median GD on 10 objectives

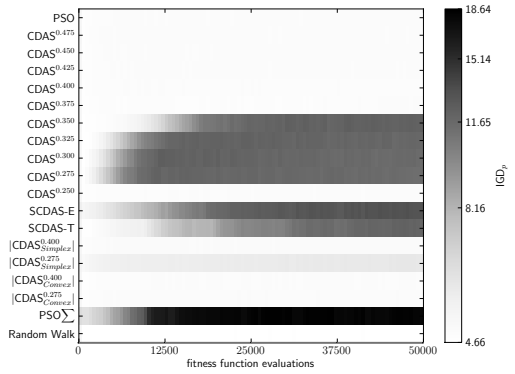


(h) WFG5 Median IGD on 10 objectives

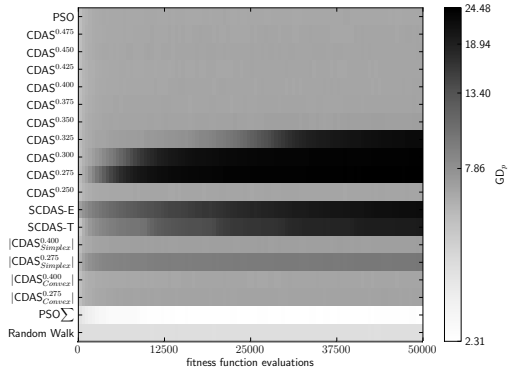
Figure A.10.: WFG5.



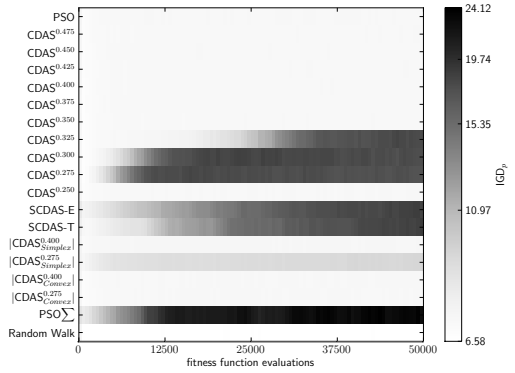
(i) WFG5 Median GD on 15 objectives



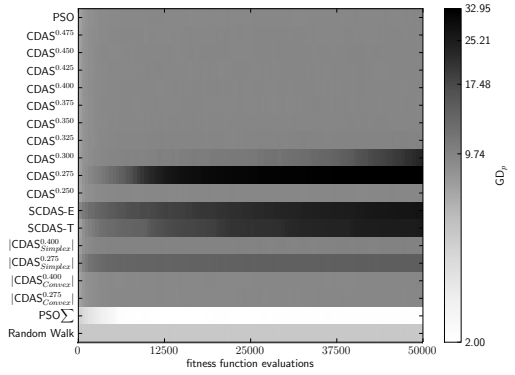
(j) WFG5 Median IGD on 15 objectives



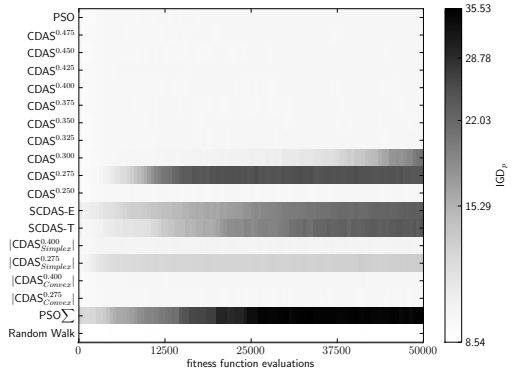
(k) WFG5 Median GD on 20 objectives



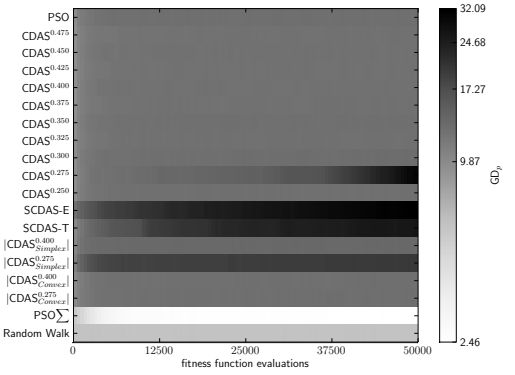
(l) WFG5 Median IGD on 20 objectives



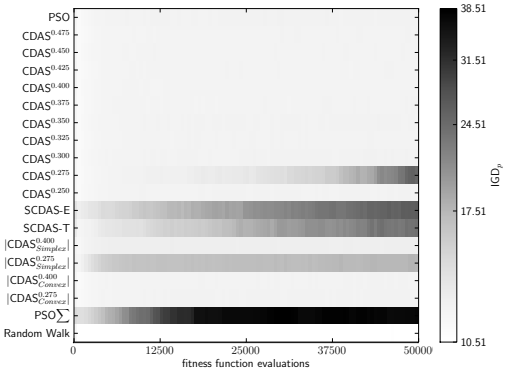
(m) WFG5 Median GD on 25 objectives



(n) WFG5 Median IGD on 25 objectives

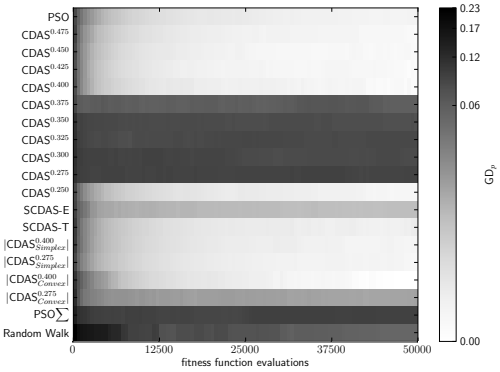


(o) WFG5 Median GD on 30 objectives

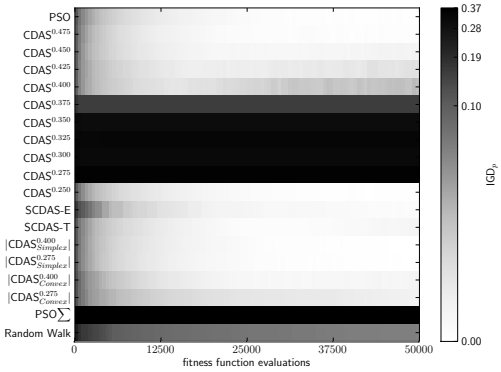


(p) WFG5 Median IGD on 30 objectives

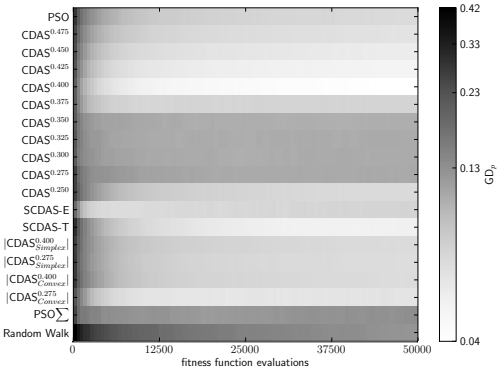
Figure A.10.: WFG5.



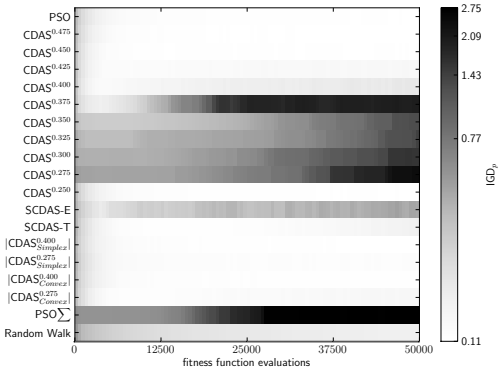
(a) WFG6 Median GD on 2 objectives



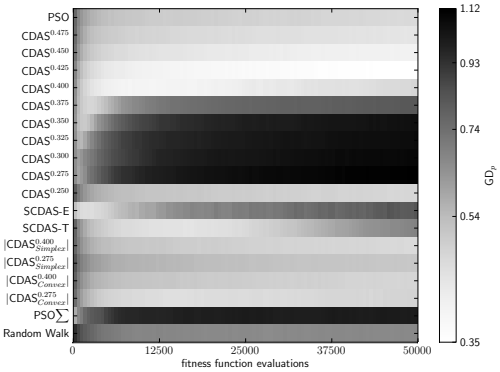
(b) WFG6 Median IGD on 2 objectives



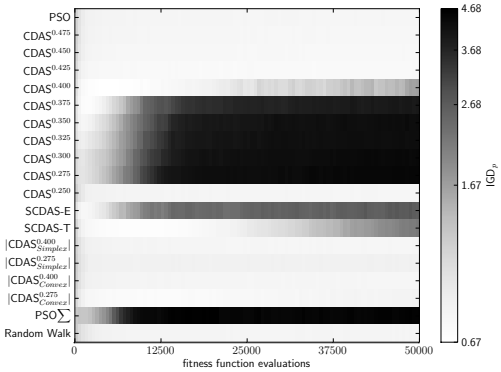
(c) WFG6 Median GD on 3 objectives



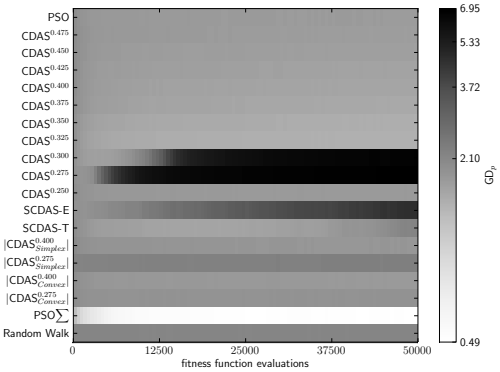
(d) WFG6 Median IGD on 3 objectives



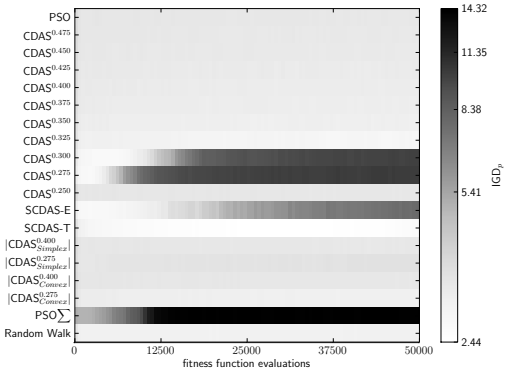
(e) WFG6 Median GD on 5 objectives



(f) WFG6 Median IGD on 5 objectives

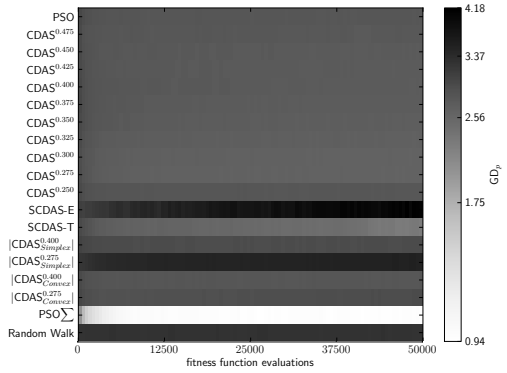


(g) WFG6 Median GD on 10 objectives

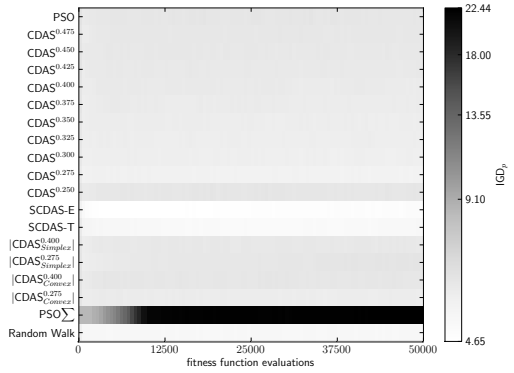


(h) WFG6 Median IGD on 10 objectives

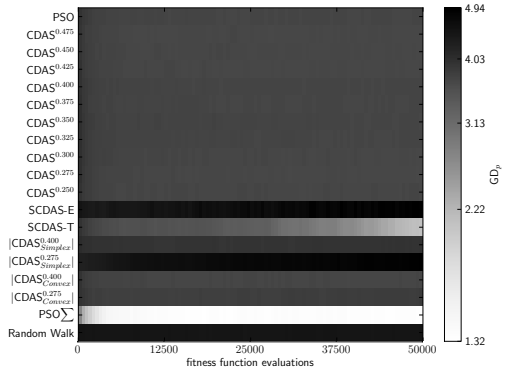
Figure A.11.: WFG6.



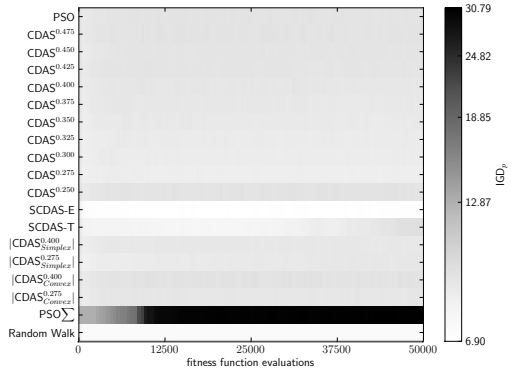
(i) WFG6 Median GD on 15 objectives



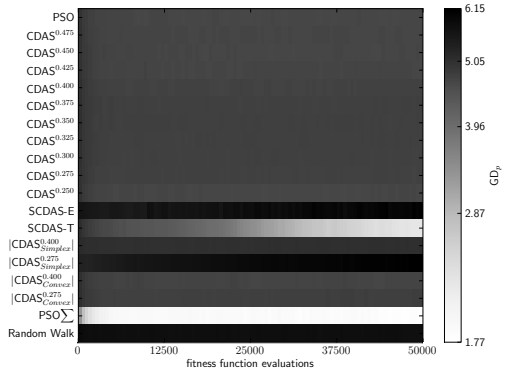
(j) WFG6 Median IGD on 15 objectives



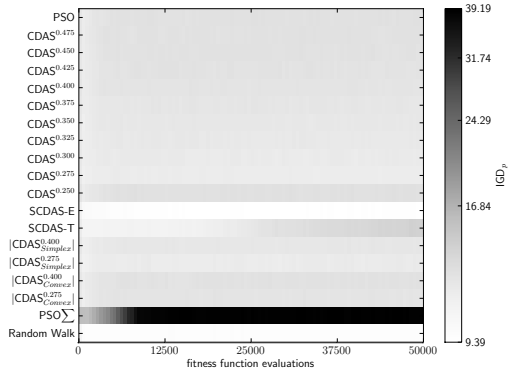
(k) WFG6 Median GD on 20 objectives



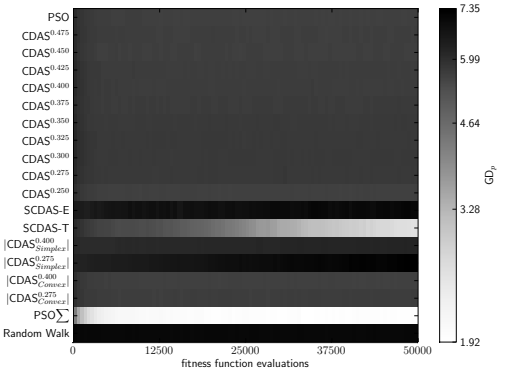
(l) WFG6 Median IGD on 20 objectives



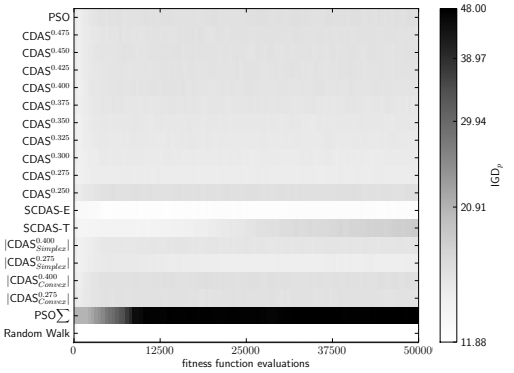
(m) WFG6 Median GD on 25 objectives



(n) WFG6 Median IGD on 25 objectives

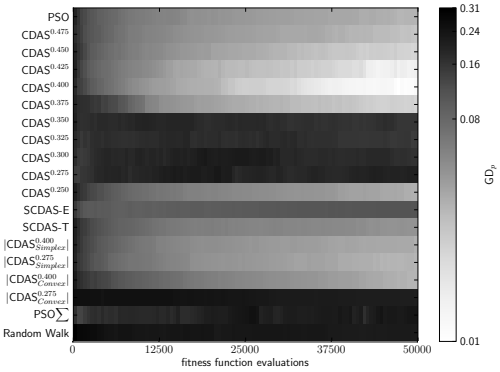


(o) WFG6 Median GD on 30 objectives

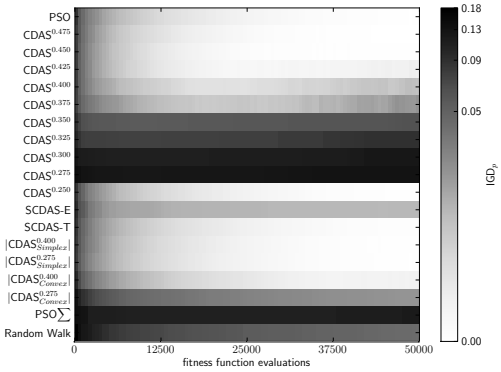


(p) WFG6 Median IGD on 30 objectives

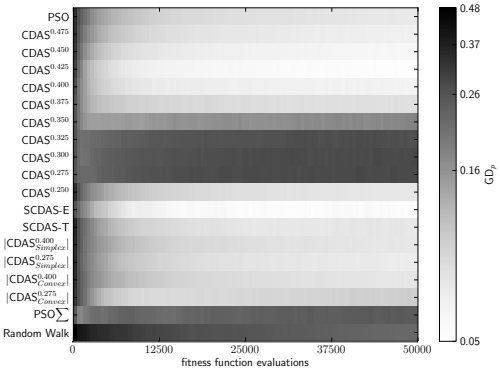
Figure A.11.: WFG6.



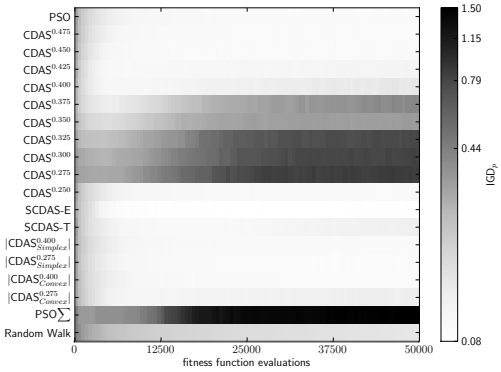
(a) WFG7 Median GD on 2 objectives



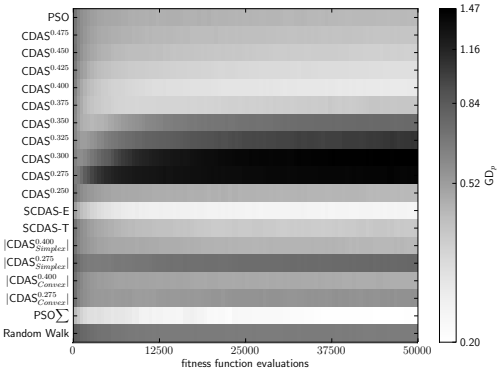
(b) WFG7 Median IGD on 2 objectives



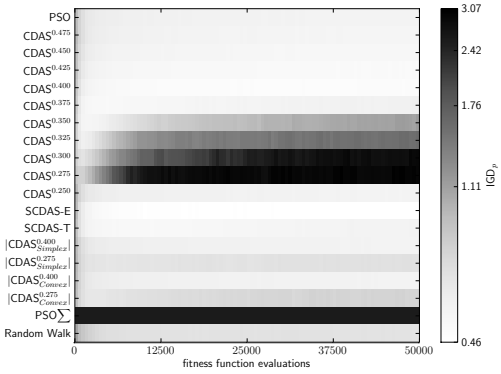
(c) WFG7 Median GD on 3 objectives



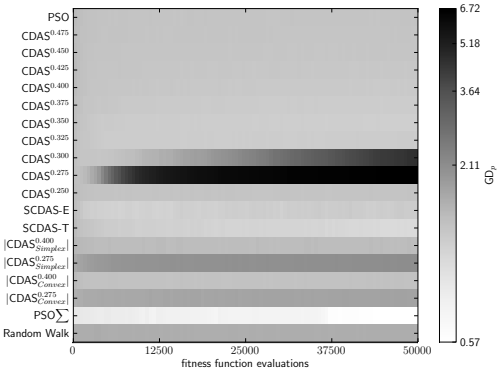
(d) WFG7 Median IGD on 3 objectives



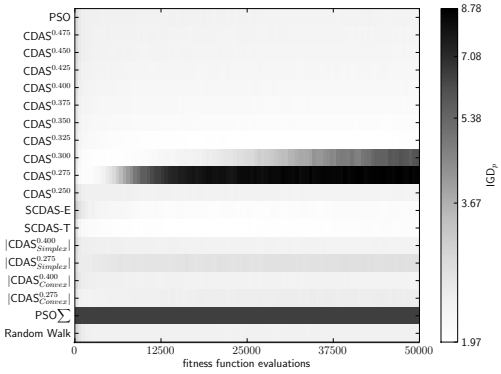
(e) WFG7 Median GD on 5 objectives



(f) WFG7 Median IGD on 5 objectives

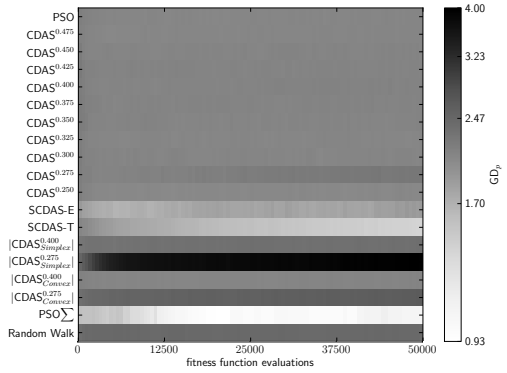


(g) WFG7 Median GD on 10 objectives

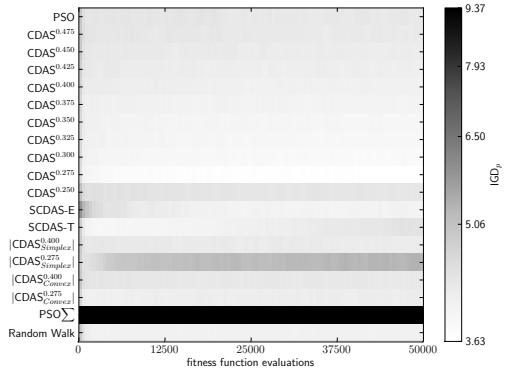


(h) WFG7 Median IGD on 10 objectives

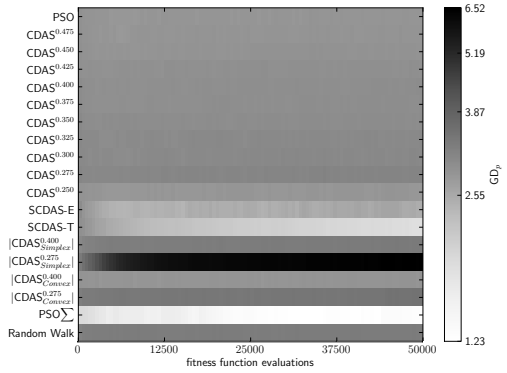
Figure A.12.: WFG7.



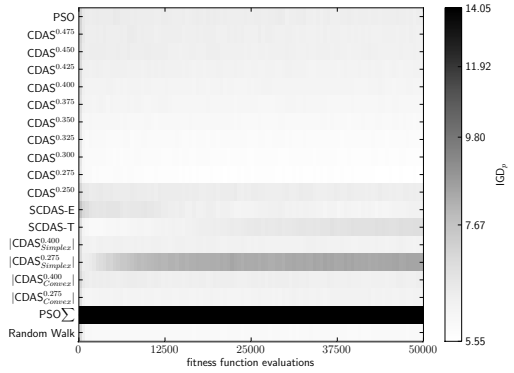
(i) WFG7 Median GD on 15 objectives



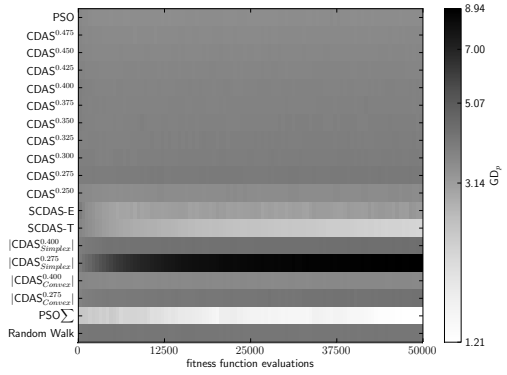
(j) WFG7 Median IGD on 15 objectives



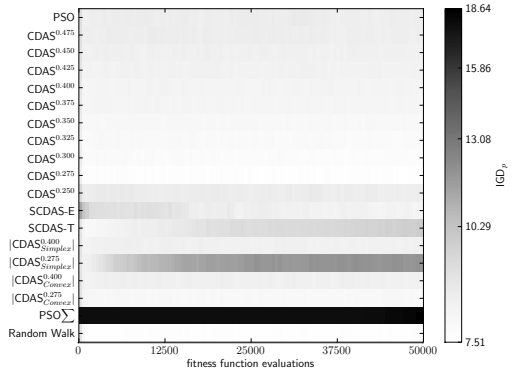
(k) WFG7 Median GD on 20 objectives



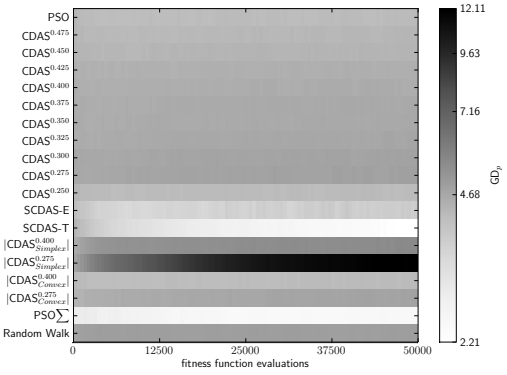
(l) WFG7 Median IGD on 20 objectives



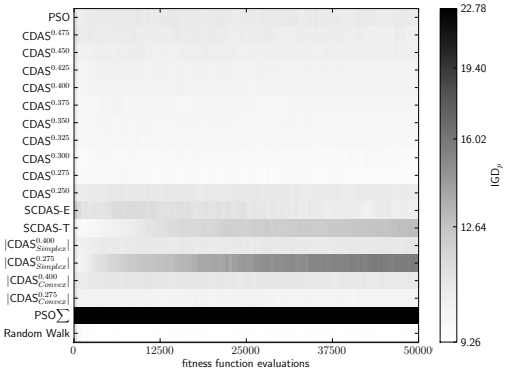
(m) WFG7 Median GD on 25 objectives



(n) WFG7 Median IGD on 25 objectives

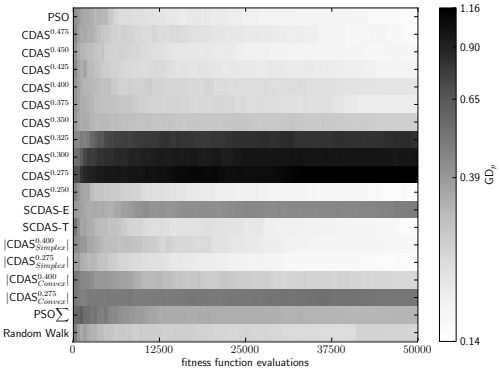


(o) WFG7 Median GD on 30 objectives

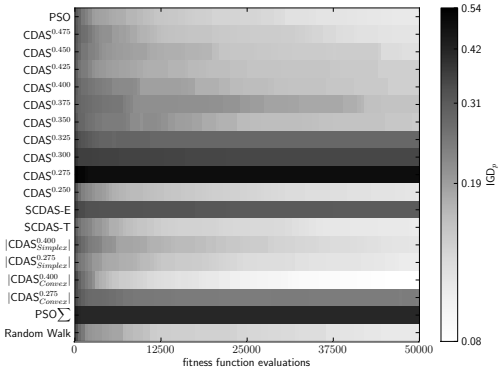


(p) WFG7 Median IGD on 30 objectives

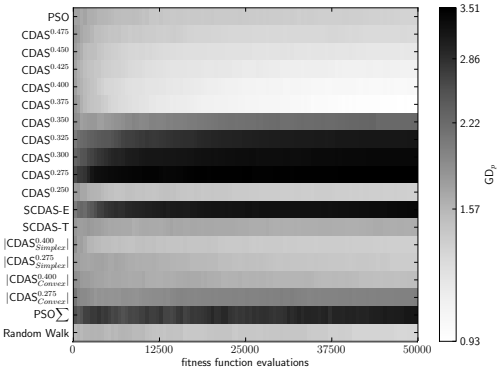
Figure A.12.: WFG7.



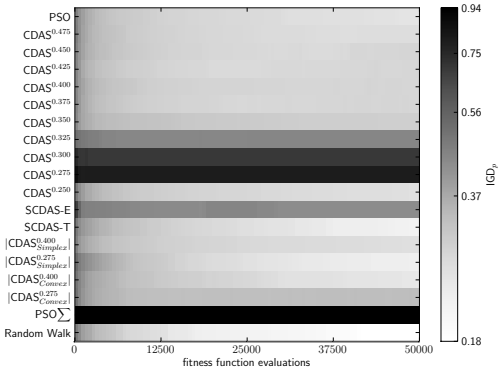
(a) WFG8 Median GD on 2 objectives



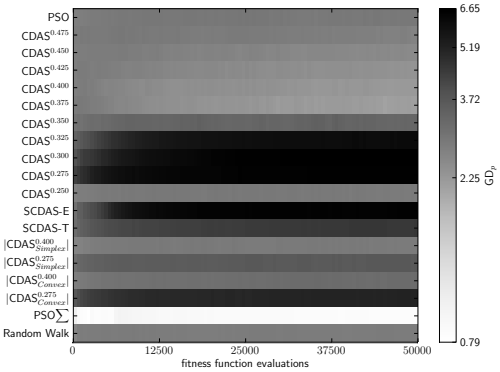
(b) WFG8 Median IGD on 2 objectives



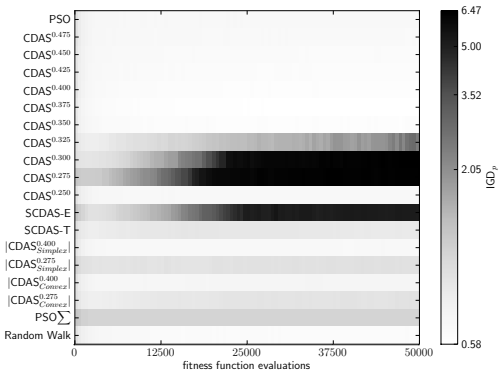
(c) WFG8 Median GD on 3 objectives



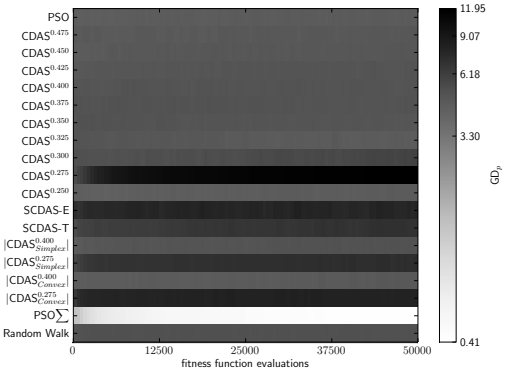
(d) WFG8 Median IGD on 3 objectives



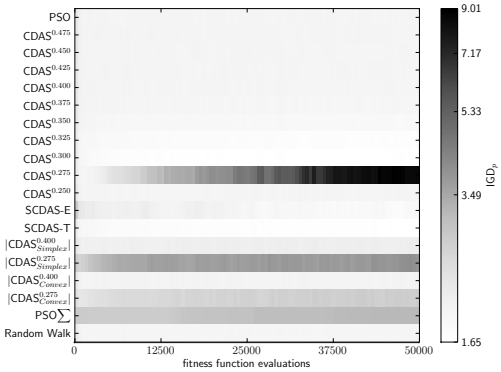
(e) WFG8 Median GD on 5 objectives



(f) WFG8 Median IGD on 5 objectives

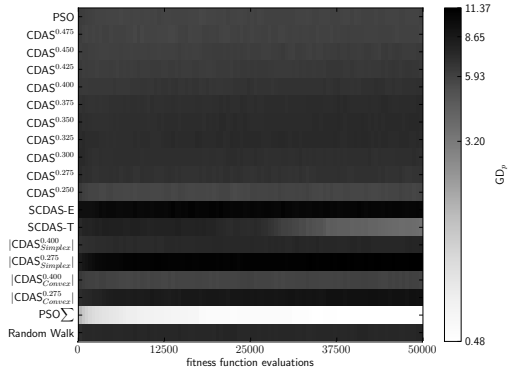


(g) WFG8 Median GD on 10 objectives

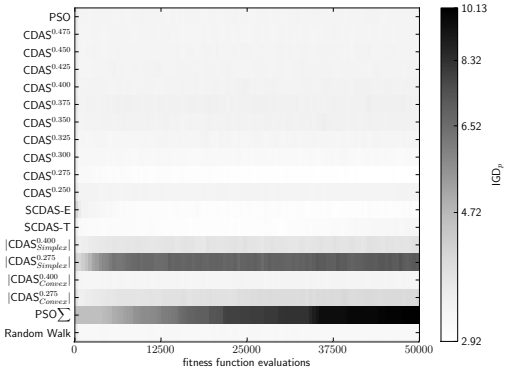


(h) WFG8 Median IGD on 10 objectives

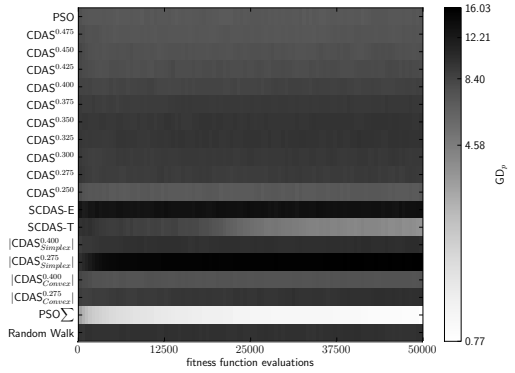
Figure A.13.: WFG8.



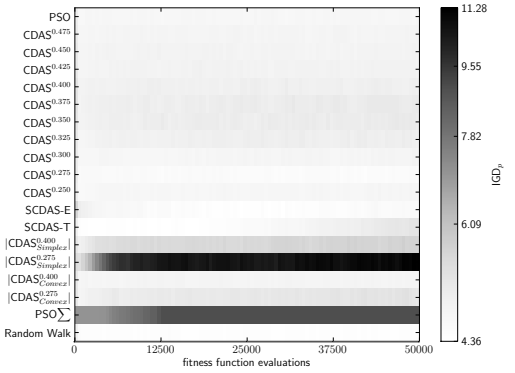
(i) WFG8 Median GD on 15 objectives



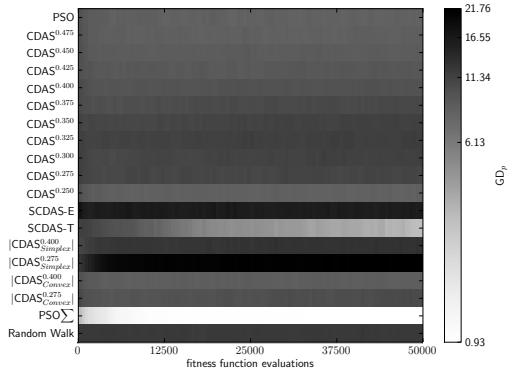
(j) WFG8 Median IGD on 15 objectives



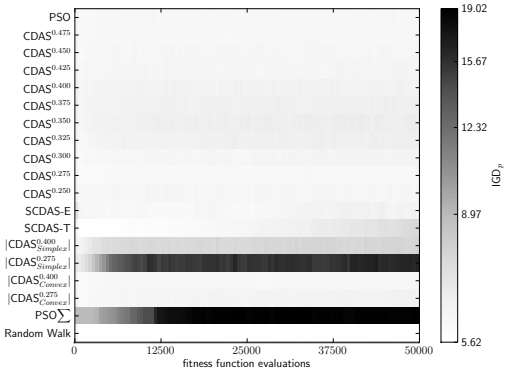
(k) WFG8 Median GD on 20 objectives



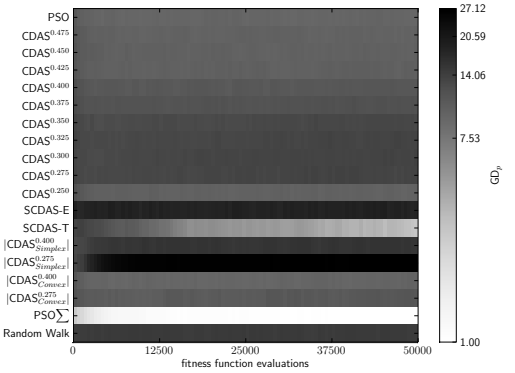
(l) WFG8 Median IGD on 20 objectives



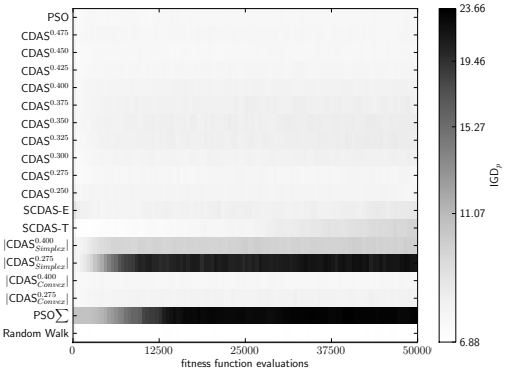
(m) WFG8 Median GD on 25 objectives



(n) WFG8 Median IGD on 25 objectives

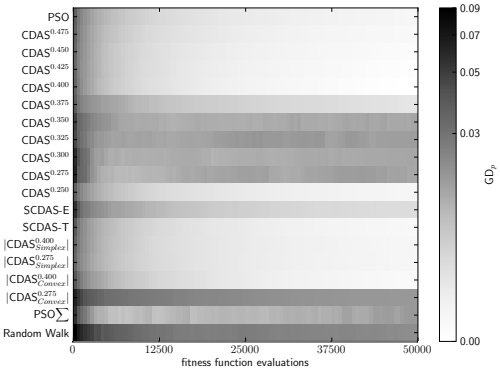


(o) WFG8 Median GD on 30 objectives

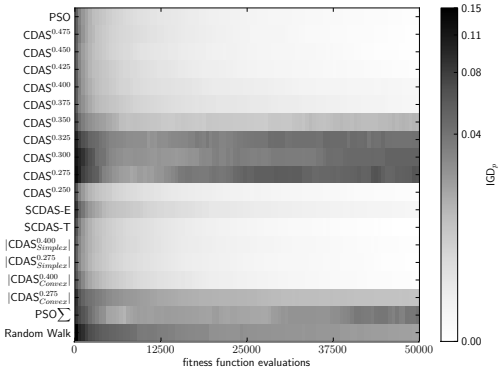


(p) WFG8 Median IGD on 30 objectives

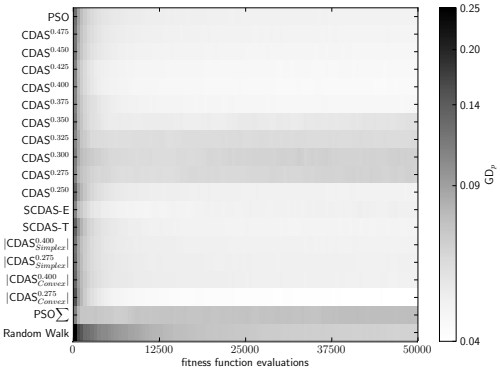
Figure A.13.: WFG8.



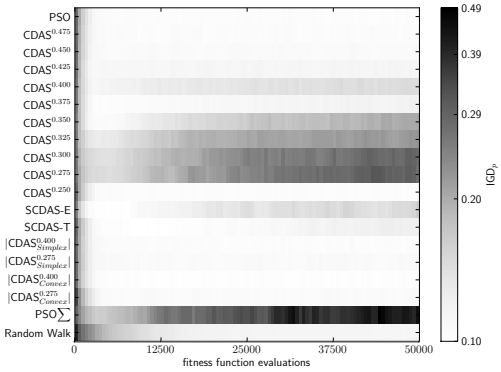
(a) WFG9 Median GD on 2 objectives



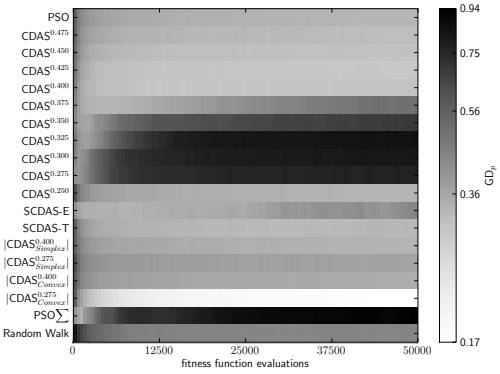
(b) WFG9 Median IGD on 2 objectives



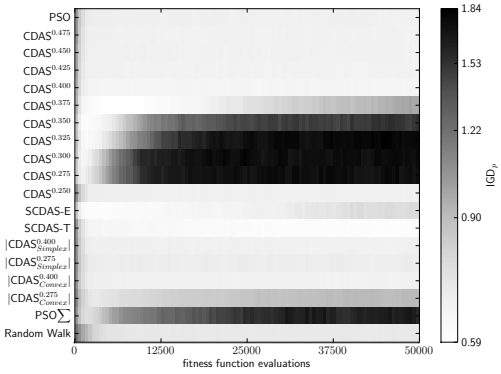
(c) WFG9 Median GD on 3 objectives



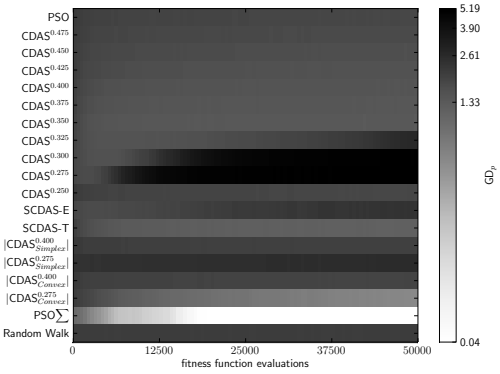
(d) WFG9 Median IGD on 3 objectives



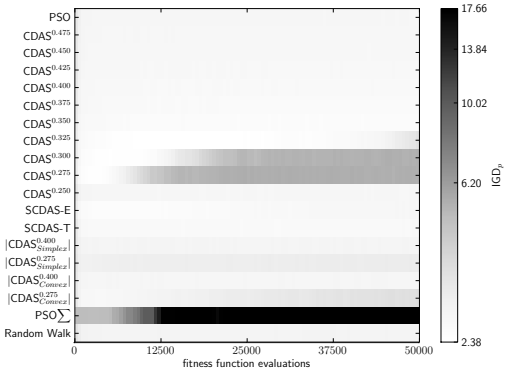
(e) WFG9 Median GD on 5 objectives



(f) WFG9 Median IGD on 5 objectives

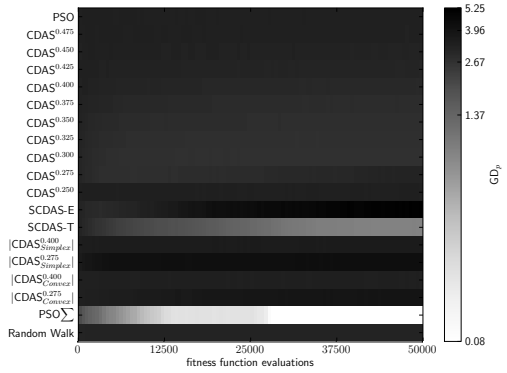


(g) WFG9 Median GD on 10 objectives

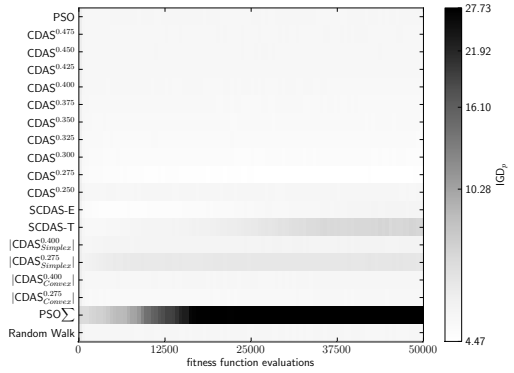


(h) WFG9 Median IGD on 10 objectives

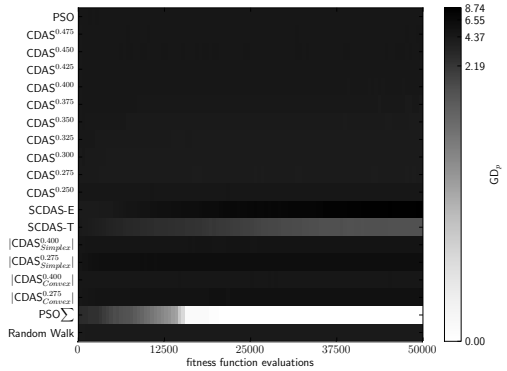
Figure A.14.: WFG9.



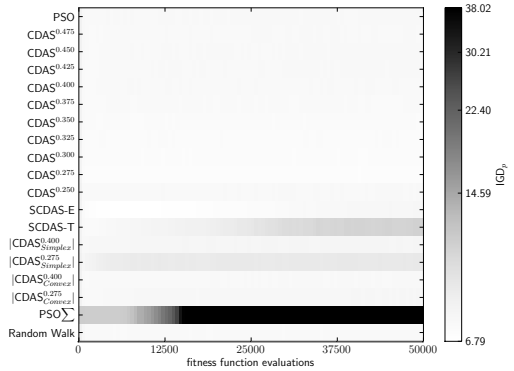
(i) WFG9 Median GD on 15 objectives



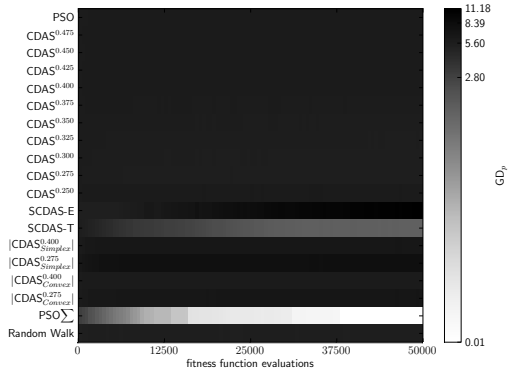
(j) WFG9 Median IGD on 15 objectives



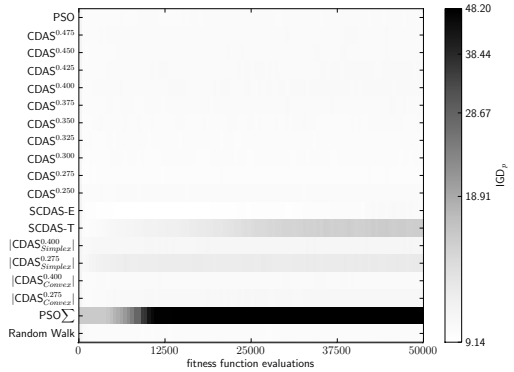
(k) WFG9 Median GD on 20 objectives



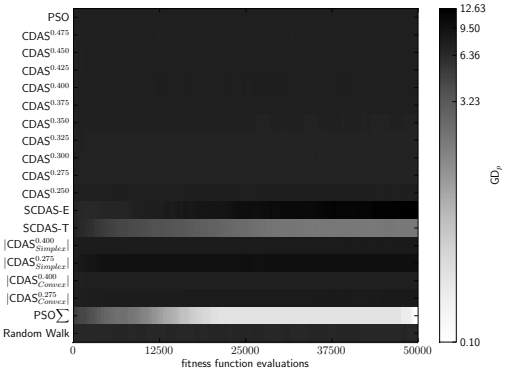
(l) WFG9 Median IGD on 20 objectives



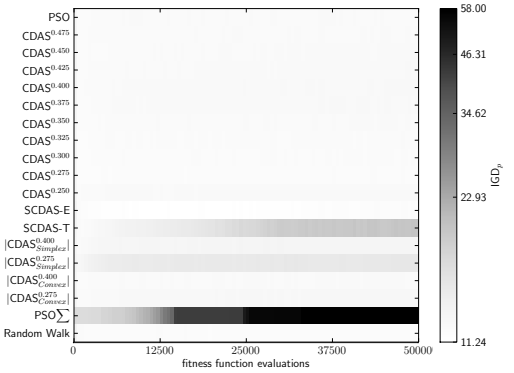
(m) WFG9 Median GD on 25 objectives



(n) WFG9 Median IGD on 25 objectives



(o) WFG9 Median GD on 30 objectives



(p) WFG9 Median IGD on 30 objectives

Figure A.14.: WFG9.

Bibliography

Abraham, A. and Liu, H. (2009). Turbulent Particle Swarm Optimization Using Fuzzy Parameter Tuning. In *Foundations of Computational Intelligence Volume 3*, pages 291–312. Springer.

Bäck, T. and Schwefel, H. (1993). An Overview of Evolutionary Algorithms for Parameter Optimization. *Evolutionary Computation*, 1:1–23.

Bentley, P. and Wakefield, J. (1997). Finding Acceptable Pareto-Optimal Solutions using Multiobjective Genetic Algorithms. *Soft Computing in Engineering Design and Manufacturing*, 5:231–240.

Brabazon, A. and O'Neill, M. (2006). *Biologically Inspired Algorithms for Financial Modelling*. Springer.

Carnahan, B., Luther, H. A., and Wilkes, J. O. (1990). *Applied numerical methods*. RE Krieger Publishing Company.

Coello, C. A. C., Lamont, G. B., and Van Veldhuisen, D. A. (2007). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer.

Coello Coello, C. and Lechunga, M. (2002). MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 1051–1056. IEEE.

Coello Coello, C. A., Van Veldhuizen, D. A., and Lamont, G. B. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers.

Corne, D. and Knowles, J. (2007). Techniques for Highly Multiobjective Optimisation: Some Nondominated Points are Better than Others. In *Proceedings of the 9th annual Conference on Genetic and Evolutionary Computation*, pages 773–780. ACM.

de Carvalho, A. B. and Pozo, A. (2010). Analyzing the Control of Dominance Area of Solutions in Particle Swarm Optimization for Many-Objective. In *Proceedings of the 10th International Conference on Hybrid Intelligent Systems*, pages 103–108. IEEE.

de Carvalho, A. B. and Pozo, A. (2011). Using Different Many-Objective Techniques in Particle Swarm Optimization for Many Objective Problems: An Empirical Study. *International Journal of Computer Information Systems and Industrial Management Applications*, 3:96–107.

Deb, K., Agrawal, S., Pratap, A., and Meyarivan, T. (2000). A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *Proceedings of the 6th Conference on Parallel Problem Solving from Nature*, pages 849–858. Springer.

Deb, K. and Kumar, A. (2007). Light Beam Search Based Multi-Objective Optimization using Evolutionary Algorithms. In *Proceedings of the 2007 Congress on Evolutionary Computation*, pages 2125–2132. IEEE.

Deb, K., Sinha, A., and Kukkonen, S. (2006a). Multi-objective Test Problems, Linkages, and Evolutionary Methodologies. In *Proceedings of the 8th annual Conference on Genetic and Evolutionary Computation*, pages 1141–1148. ACM.

Deb, K., Sundar, J., Udaya Bhaskara Rao, N., and Chaudhuri, S. (2006b). Reference Point Based Multi-objective Optimization Using Evolutionary Algorithms. *International Journal of Computational Intelligence Research*, 2(3):273–286.

Deb, K., Thiele, L., Laumanns, M., and Zitzler, E. (2002). Scalable Multi-Objective Optimization Test Problems. In *Proceedings of the 2002 Congress on Evolutionary Computation*, volume 1, pages 825–830. IEEE.

Di Pierro, F., Khu, S.-T., and Savic, D. (2007). An Investigation on Preference Order Ranking Scheme for Multiobjective Evolutionary Optimization. *IEEE Transactions on Evolutionary Computation*, 11(1):17 –45.

Drechsler, N., Drechsler, R., and Becker, B. (2001). Multi-objective Optimisation Based on Relation Favour. In *Evolutionary Multi-Criterion Optimization*, pages 154–166. Springer.

Everson, R. M., Walker, D. J., and Fieldsend, J. E. (2013). Edges of Mutually Non-

dominating Sets. In *Proceedings of the 15th annual Conference on Genetic and Evolutionary Computation*, pages 607–614. ACM.

Fieldsend, J. (2004). Multi-Objective Particle Swarm Optimisation Methods. Technical report, Department of Computer Science, University of Exeter.

Fieldsend, J. and Singh, S. (2002a). A Multi-Objective Algorithm based upon Particle Swarm Optimisation, an Efficient Data Structure and Turbulence. In *Proceedings of the 2002 U.K. Workshop on Computational Intelligence*, pages 37–44, Birmingham, UK.

Fieldsend, J. E. and Singh, S. (2002b). On the Selection of Gbest, Lbest and Pbest Individuals, the Use of turbulence and the impact of inertia in multi-objective PSO. Technical report, Department of Computer Science, University of Exeter.

Fonseca, C. M. and Fleming, P. J. (1995). An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1):1–16.

Garza-Fabre, M., Toscano-Pulido, G., and Coello, C. (2010). Two Novel Approaches for Many-Objective Optimization. In *Proceedings of the 2010 Congress on Evolutionary Computation*, pages 1 –8. IEEE.

Greene, C. S., White, B. C., and Moore, J. H. (2008). Ant Colony Optimization for Genome-Wide Genetic Analysis. In *Ant Colony Optimization and Swarm Intelligence*, pages 37–47. Springer.

Hu, X. and Eberhart, R. (2002). Multiobjective Optimization Using Dynamic Neighborhood Particle Swarm Optimization. In *Proceedings of the 2002 Congress on Evolutionary Computation*, volume 2, pages 1677–1681. IEEE.

Huband, S., Barone, L., While, L., and Hingston, P. (2005). A Scalable Multi-Objective Test Problem Toolkit. In *Proceedings of the 3th International Conference on Evolutionary Multi-Criterion Optimization*, pages 280–295. Springer.

Huband, S., Hingston, P., Barone, L., and While, L. (2006). A Review of Multiobjective Test Problems and a Scalable Test Problem Toolkit. *IEEE Transactions on Evolutionary Computation*, 10(5):477–506.

Ikeda, K., Kita, H., and Kobayashi, S. (2001). Failure of Pareto-based MOEAs: Does Non-dominated Really Mean Near to Optimal? In *Proceedings of the 2001 Congress on Evolutionary Computation*, volume 2, pages 957–962. IEEE.

Ishibuchi, H., Tsukamoto, N., and Nojima, Y. (2008). Evolutionary Many-Objective Optimization: A short review. In *Proceedings of the 2008 Congress on Evolutionary Computation*, pages 2419–2426. IEEE.

Janson, S. and Middendorf, M. (2005). A Hierarchical Particle Swarm Optimizer and its Adaptive Variant. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 35(6):1272–1282.

Junior, C., Rodrigues, O., Britto, A., and Pozo, A. (2012). Self-Controlling Dominance Particle Swarm Optimization. In *Proceedings of the 2012 Brazilian Symposium on Neural Networks*, pages 172–177. IEEE.

Kennedy, J. and Eberhart, R. (1995). Particle Swarm Optimization. In *Proceedings of the 1995 International Conference on Neural Networks*, pages 1942–1948. IEEE.

Krink, T., VesterstrOm, J. S., and Riget, J. (2002). Particle Swarm Optimisation with Spatial Particle Extension. In *Proceedings of the 2002 World on Congress on Computational Intelligence*, volume 2, pages 1474–1479. IEEE.

Laumanns, M., Thiele, L., Deb, K., and Zitzler, E. (2002). Combining Convergence and Diversity in Evolutionary Multiobjective Optimization. *Evolutionary computation*, 10(3):263–282.

Leong, W.-F. and Yen, G. G. (2006). Dynamic Population size in PSO-based Multiobjective Optimization. In *Proceedings of the 2006 Congress on Evolutionary Computation*, pages 6182–6189. IEEE.

Li, X. (2010). Niching Without Niching Parameters: Particle Swarm Optimization using a Ring Topology. *IEEE Transactions on Evolutionary Computation*, 14(1):150–169.

Liu, H., Abraham, A., and Zhang, W. (2007). A Fuzzy Adaptive Rurbulent Particle Swarm Optimisation. *International Journal of Innovative Computing and Applications*, 1(1):39–47.

Marrow, P. (2000). Nature-Inspired Computing Technology and Applications. *BT Technology Journal*, 18(4):13–23.

Martello, S. and Toth, P. (1990). *Knapsack Problems*. Wiley New York.

Miyagawa, E. and Saito, T. (2009). Particle Swarm Optimizers with Growing Tree Topology. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 92(9):2275–2282.

Moore, J. and Chapman, R. (1999). Application Of Particle Swarm To Multiobjective Optimization. (Unpublished) Auburn University.

Moritz, R. L., Reich, E., Schwarz, M., Bernt, M., and Middendorf, M. (2013). Refined Ranking Relations for Multi Objective Optimization Andapplication to P-ACO. In *Proceedings of the 15th annual Conference on Genetic and Evolutionary Computation*, pages 65–72. ACM.

Padhye, N. (2009). Comparison of Archiving Methods in Multi-Objective Particle Swarm Optimization (MOPSO): Empirical Study. In *Proceedings of the 11th annual Conference on Genetic and Evolutionary Computation*. ACM.

Padhye, N., Branke, J., and Mostaghim, S. (2009). Empirical Comparison of MOPSO Methods: Guide Selection and Diversity Preservation. In *Proceedings of the 11th annual Conference on Genetic and Evolutionary Computation*. ACM.

Parsopoulos, K. and Vrahatis, M. (2002). Particle Swarm Optimization Method in Multiobjective Problems. In *Proceedings of the 2002 Symposium on Applied Computing*, pages 603–607. ACM.

Peer, E., Van den Bergh, F., and Engelbrecht, A. (2003). Using Neighbourhoods with the Guaranteed Convergence PSO. In *Proceedings of the 2003 Swarm Intelligence Symposium*, pages 235–242. IEEE.

Reyes-Sierra, M. and Coello Coello, C. A. (2006). Multi-Objective Particle Swarm Optimizers: A Survey of the State-of-the-Art. *International Journal of Computational Intelligence Research*, 2(3):287–308.

Sato, H., Aguirre, H., and Tanaka, K. (2007a). Controlling Dominance Area of Solutions in Multiobjective Evolutionary Algorithms and Performance Analysis on Multiobjective 0/1 Knapsack Problems. *IPSJ Digital Courier*, 3:703–718.

Sato, H., Aguirre, H. E., and Tanaka, K. (2007b). Controlling Dominance Area of Solutions and Its Impact on the Performance of MOEAs. In *Proceedings of the 4th International Conference on Evolutionary Multi-Criterion Optimization*, pages 5–20. Springer.

Sato, H., Aguirre, H. E., and Tanaka, K. (2007c). Local Dominance Including Control of Dominance Area of Solutions in MOEAs. In *Proceedings of the 2007 Symposium on Computational Intelligence in Multicriteria Decision Making*, pages 310–317. IEEE.

Sato, H., Aguirre, H. E., and Tanaka, K. (2010). Self-Controlling Dominance Area of Solutions in Evolutionary Many-Objective Optimization. In *Simulated Evolution and Learning*, pages 455–465. Springer.

Schütze, O., Esquivel, X., Lara, A., and Coello, C. A. C. (2012). Using the Averaged Hausdorff Distance as a Performance Measure in Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation*, 16(4):504–522.

Schwefel, H. (1981). *Numerical Optimization of Computer Models*. John Wiley & Sons, Inc.

Settles, M. (2005). An Introduction to Particle Swarm Optimization. Technical report, Department of Computer Science, University of Idaho.

Taillard, E. (1993). Benchmarks for Basic Scheduling Problems. *European Journal of Operational Research*, 64(2):278–285.

Van Veldhuizen, D. A. and Lamont, G. B. (1998). Evolutionary Computation and Convergence to a Pareto Front. In *Late Breaking Papers at the Genetic Programming 1998 Conference*, pages 221–228.

Voglis, C., Hadjidoukas, P. E., Parsopoulos, K. E., Papageorgiou, D. G., and Lagaris, I. E. (2013). Adaptive Memetic Particle Swarm Optimization with Variable Local Search Pool Size. In *Proceedings of the 15th annual Conference on Genetic and Evolutionary Computation*, pages 113–120. ACM.

Wickramasinghe, U. K. and Li, X. (2009). Using a Distance Metric to Guide PSO Algorithms for Many-Objective Optimization. In *Proceedings of the 11th annual Conference on Genetic and Evolutionary Computation*, pages 667–674. ACM.

Wilke, D. N., Kok, S., and Groenwold, A. A. (2007). Comparison of Linear and Classical Velocity Update Rules in Particle Swarm Optimization: Notes on Diversity. *International Journal for Numerical Methods in Engineering*, 70(8):962–984.

Xi, B., Liu, Z., Raghavachari, M., Xia, C. H., and Zhang, L. (2004). A smart Hill-Climbing Algorithm for Application Server Configuration. In *Proceedings of the 13th international conference on World Wide Web*, pages 287–296. ACM.

Xu, S. and Rahmat-Samii, Y. (2007). Boundary conditions in particle swarm optimization revisited. *IEEE Transactions on Antennas and Propagation*, 55(3):760–765.

Yang, X.-S. (2010). *Nature-inspired metaheuristic algorithms*. Luniver Press.

Zitzler, E., Deb, K., and Thiele, L. (2000). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation*, 8(2):173–195.

Zitzler, E. and Künzli, S. (2004). Indicator-based Selection in Multiobjective Search. In *Proceeding of the 8th International Conference on Parallel Problem Solving from Nature*, pages 832–842. Springer.

Zitzler, E., Laumanns, M., and Thiele, L. (2002). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In *Proceeding of the 2002 International Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems*, pages 95–100.

Zitzler, E. and Thiele, L. (1999). Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271.