Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

New Archive Based Evolutionary Multi-Objective Algorithms

Tesis que presenta

Xavier Esquivel Sánchez

para obtener el Grado de

Maestro en Ciencias en Computación

Director de la Tesis

Dr. Oliver Schütze

México, D.F. **Noviembre 2010**

Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Nuevo Algoritmo Evolutivo Multi-Objetivo Basado en un Archivo

Tesis que presenta

Xavier Esquivel Sánchez

para obtener el Grado de

Maestro en Ciencias en Computación

Director de la Tesis

Dr. Oliver Schütze

México, D.F. **Noviembre 2010**

Dedicatoria

A mi esposa, Elena, por su apoyo, comprensión y por ser la mujer que me ha ayudado a tener sueños y metas.

A mis padres Josefina y Javier, quienes con sus esfuerzos y preocupaciones, me han proveído de todo lo necesario para lograr mis metas.

En memoria de mi abuelo Angel.

Agradecimientos

Al CINVESTAV por permitir formar parte de esta gran institución.

Al CONACyT por el apoyo brindado, sin éste no hubiese podido llevar a cabo este proyecto.

Al Dr. Oliver Schütze, por haberme dado la oportunidad de elaborar la tesis bajo su dirección.

Abstract

In many problems in industry and finance the problem arises that several objectives have to be optimized concurrently. So far, the most widely used approach to compute the solution set (the *Pareto front*) of such Multi-Objective Optimization roblems (MOPs) are the Multi-Objective Avolutionary Algorithms (MOEAs).

In this thesis, we develop and investigate a new archive-based MOEA (i.e., a MOEA equipped with an external archive in order to guarantee convergence of the search process). For this, we will propose two new variation operators, the Shifted Polynomial Mutation (SPM) and Biased Intermediate Crossover (BIC), which are intended to meet the requeriments of the archivers we are using (archivers based on the concept of ϵ-dominance). Based on theoretical and empirical observations we construct the MOEA ELMA (Evolutionary Lipschitz Multi-Objective Algorithm) which is aiming, roughly speaking, for a good Hausdorff approximations of the Pareto front of the given MOP.

In order to evaluate and compare ELMA in a fair way, we develop further on a new performance indicator which is measuring the averaged Hausdorff distance to the Pareto front. The new indicator, Δ_p, is composed of (slight variants) of the well-know indicators Generational Distance (GD) and Inverted Generational Distance (IGD) and can be applied to both discrete or continuos model.

Finally, we compare ELMA against ϵ-MOEA, which is a widely used archive-based MOEA and which can be viewed as the base algorithm of ELMA. It turns out that ELMA outperforms the ϵ-MOEA on several benchmark models such as the ZDT models, indicating that ELMA is a step forward in the design of fast and reliable MOEAs.

Resumen

En muchos problemas en la industria y finanzas, se tiene el problema de minimizar más de un objetivo, i.e., estos objetivos deben ser minimizados simultneamente. Por el momento, los métodos más utilizados para aproximar el conjunto solucin (*frente de Pareto*) de un Problema de Optimizacin Multi-Objetivo (POM) son los Algoritmos Evolutivos de Optimizacin Multi-Objetivo (AEOM).

En la presente tesis, hemos desarrollado e investigado un nuevo algoritmo evolutivo basado en un archivo (i.e. AEOM equipados con un archivo externo con el fin de garantizar convergencia del proceso de búsqueda). Para esto, nosotros proponemos dos operadores variacionales, la Shifted Polynomial Mutation (SPM) y Biased Intermediate Crossover (BIC). Estos operadores están hechos según las necesidades de los archivos que utilizamos (archivos basados en el concepto de dominancia-ϵ). Teniendo como base aspectos teóricos y observaciones, hemos construido un nuevo algoritmo evolutivo, ELMA (Evolutionary Lipschitz Multi-Objective Algorithm). El objetivo de este algoritmo es obtener una buena aproximacin, en términos de la distancia de Hausdorff, en relacin con el frente de Pareto.

Con el fin de evaluar nuestro algoritmo, ELMA, de forma justa, hemos desarrollado un indicador que mide, en promedio, la distancia de Hausdorff con respecto al frente de Pareto. El nuevo indicador, Δ_p, esta compuesto por dos indicadores utilizados en la literatura (pequeñas variantes), Generational Distance (GD) e Inverted Generational Distance (IGD). El nuevo indicador puede ser aplicado en modelos discretos o continuos.

Finalmente, comparamos nuestro algoritmo, ELMA, contra el algoritmo ϵ-MOEA, que es un algoritmo evolutivo basado en un archivo ampliamente utilizado. Hemos observado que la aproximación obtenida, en diferentes modelos como ZDT, por nuestro algoritmo es mejor que ϵ-MOEA y, además es un paso adelante en el diseño de un algoritmo evolutivo rápido y confiable.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

In many areas like economics and engineering, complex optimization problems can not be described by only one objective function (for one such example see [8]). As a result, the Multi-Objective Optimization Problem (MOP) arises, which investigates the concurrent optimization of several objective functions. The pecularity of MOPs is that the set of optimal solutions (the Pareto set, denoted as P_Q) is typically not given by only one point as for 'classical' scalar optimization problems, but rather forms a $(k-1)$-dimensional object, where k is the number of objectives.

One widely used class of algorithms to approximate P_Q is given by Multi-Objective Evolutionary Algorithms (MOEAs). The population approach of Evolutionary Algorithms (EA) allows for an efficient way to find several points near to the Pareto set simultaneously. However, there is no algorithm which gurantees convergence toward the optimal set with a predicted approximation quality.

The MOEAs proposed in the literature have their own biological inspiration as well as their mechanism to maintain diversity among the population [14, 6]. Nevertheless, the variation operators have so far been of little interest in evolutionary multi-objective optimization. In general, most algorithms use standard operators derived from single objective optimization.

In this work, we deal with the design of archive-based MOEAs i.e., MOEAs equipped with an external archive. In particular, we design two variation operators (one mutation and one crossover operator) that are tailored to a class of archiving strategies and propose a new evolutionary algorithm ELMA, that can be viewed as a variant of ϵ-MOEA [12], a well-known archive-based MOEA.

One pecularity of ELMA, worth mentioning here, is that it aims for good Hausdorff approximations of the Pareto front of the given MOP. Even though there exist so far a large variety of quality indicators for the evaluation of the outcome set of a MOEA (see, e.g. [48, 5, 6, 7, 52]), none of them can be used here, since all of them have been designed for different purposes. Even the 'classical' Hausdorff distance is not an optimal choice since stochastic search algorithms—such as ELMA—tend to generate outliers during the search process. As a possible remedy, we suggest in this work a new indicator, Δ_p, which can be viewed as an 'averaged Hausdorff distance' and which is more 'fair' to evaluate the outcome sets of stochastic search algorithms.

The remainder of this document is organized as follows.

Chapter 2. Background: We discuss the main concepts from multi-objective optimization. One sub-class of MOEAs, archive-based MOEAs, is described, in particular the ϵ-MOEA. We also describe other state-of-the-art MOEAs reported in the literature. A description of the performance measures is given. Finally, we give an overview of existing variation operators, with a special attention on the operators Simulated Binary Crossover (SBX) and Polynomial Mutation (PM).

Chapter 3. Archive Based MOEA: We start with the investigation of a particular algorithmic framework which we use to describe the working principle of the new variation operators. This will lead to the algorithm Evolutionary Lipschitz Multi-Objective Algorithm (ELMA), which can be viewed as a modification of ϵ-MOEA.

Chapter 4. Δ_p: A New Indicator to Measure the Averaged Hausdorff Distance to the Pareto Front: We make an investigation of two performance measure indicators used in the literature, Generational Distance (GD) and Inverted Generational Distance (IGD). We argue that it makes sense to use a modification of the original operators. Based on these two, we propose a modification of these indicators and, a novel indicator composed of these modifications will be defined. The last subsection of this chapter describes an algorithm to discretize the Pareto front in a suitable way, which speeds up the computational effort to obtain the indicator value.

Chapter 5. Numerical Results: In this chapter, we present our numerical results. The first section describes all models used to evaluate ELMA and they are compared against ϵ-MOEA. Further, we give an attempt to evaluate the newly developed indicator.

Chapter 6. Conclusions and Future Work: We provide the conclusion and mention some of the paths for future research.

Chapter 2

Background

Multi-objective optimization deals with the concurrent optimization (minimization or maximization) of several objectives. The pecularity of a Multi-Objective Optimization Problem (MOP) is that the set of optimal solutions (the Pareto set) is typically not given by only one point as for 'classical' scalar optimization problems but rather forms a $(k-1)$-dimensional object, where k is the number of objectives.

2.1 Multi-Objective Optimization Problem

Mathematically speaking, a MOP is defined as follows:

$$\min_{x \in Q \subset \mathbb{R}^n} \{F(x)\} \tag{MOP}$$

where function F is defined as a vector of objective functions,

$$F : \mathbb{R}^n \to \mathbb{R}^k, \quad F(x) = (f_1(x), \ldots, f_k(x)), \tag{2.1}$$

and where each $f_i : Q \to \mathbb{R}$ is continuous, and, $Q \subset \mathbb{R}^n$ is the feasible set or domain of F. Here we restrict ourselves on bound constraints, i.e., Q can be expressed by an n-dimensional box

$$Q = B_{l,u} = \{x \in Q \ : \ l_i \leq x_i \leq u_i, \ i = 1, \ldots, n\}, \tag{2.2}$$

where $l, u \in \mathbb{R}^n$ with $l_i \leq u_i, i = 1, \ldots, n$ are the vector of lower and upper bounds for each parameter value.

In scalar optimization, the decision if a point v is better than a point w is simply done by looking at the objectives $f(v)$ and $f(w)$. This can not be done in case several objectives are considered. In that case, the concept of dominance is typically used.

Definition 1. Let v, w, $\in Q$. Then the vector v is less than w ($v <_p w$), if $v_i < w_i$ for all $i \in \{1, \ldots, n\}$. The relation $\leq_p$ is defined analogously.

a) A vector $y \in Q$ is dominated by a vector $x \in Q$ (in short: $x \prec y$) with respect to (MOP) if $F(x) \leq_p F(y)$ and $F(x) \neq F(y)$ (i.e., there exists $j \in \{1, \ldots, n\}$ such that $f_j(x) < f_j(y)$).

b) A point $x \in \mathbb{R}^n$ is called Pareto optimal or a Pareto point if there is no $y \in Q$ which dominates x.

The set of Pareto optimal points are defined as:

$$P_Q = \{x \in Q \mid \nexists\, y \in Q \text{ and } y \prec x\} \tag{2.3}$$

The image of the Pareto set $F(P_Q)$ is called the Pareto front.

Example 1. As an example of the previous definition consider the following MOP. The function $F = (f_1, f_2)$, with $f_1(x) = x^2$, $f_2(x) = (x-1)^2$ and $Q = [-1, 3]$. The Pareto set of this example is given by $P_Q = [0, 1]$. Fig. 2.1 shows these two sets and, for every y outside the interval [0, 1] there exists a point $x \in P_Q$ such that it dominates, $x \prec y$.

Another concept of optimality is given by ϵ-dominance which is defined as follows [29]:

Definition 2. Let $\epsilon = (\epsilon_i, \ldots, \epsilon_k) \in \mathbb{R}^k_+$ and x, $y \in Q$, x is said to ϵ-dominate y (in short: $x \prec_\epsilon y$) with respect to (MOP) if

a) $f_i(x) - \epsilon_i \leq f_i(y)\ \forall i = 1, \ldots, k$ and

b) $f_j(x) - \epsilon_j < f_j(y)$ for at least one $j \in \{1, \ldots, k\}$

The concept of ϵ-dominance allows a finite size representation of the Pareto front. However, it does not prevent of obtaining gaps in the approximation when the Pareto front contains dents, or is 'flat', in certain regions [46]. In contrast, when using the classical dominance relation such gaps typically do not occur, but, theoretically infinitely many points are needed for a perfect representation.

Definition 3 ([27])**.**

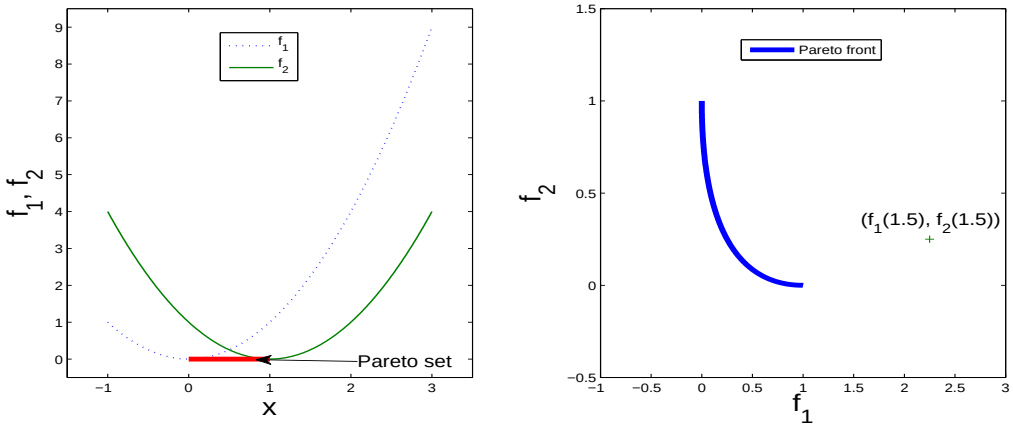


Figure 2.1: Multi-objective problem of Example 1, Pareto set (left), Pareto front (right). There is a point outside the interval [0, 1], the point $x = 1.5$, which is not a Pareto point.

a) Let $\epsilon \in \mathbb{R}^k_+$. A set A_ϵ is called an ϵ-approximate Pareto set of (MOP) if every point $x \in Q$ is ϵ-dominated by at least one $a \in A_\epsilon$, i.e.

$$\forall x \in Q : \exists a \in A_\epsilon \text{ such that } a \prec_\epsilon x$$

b A set $A^*_\epsilon \subset Q$ is called an ϵ-Pareto set if A^*_ϵ is an ϵ-approximate Pareto set and if every point $a \in A^*_\epsilon$ is a Pareto point of (MOP).

The above definitions are represented in Fig. 2.2. Since the Pareto set of MOP is usually not practical because of its size, the ϵ-approximate Pareto set is a better practical solution concept as it not only represents all vectors F but also consists of a smaller number of elements. However, an ϵ-Pareto set is more attractive as it consists of Pareto vectors only.

2.2 Evolutionary Algorithms

Evolutionary algorithm (EA) is a search method that is inspired by natural selection and, particularly the 'survival of the fittest' principle that exists in the biological world. The idea behind this technique is described as follows: given a quality function to be maximized, we can randomly create a set of candidate solutions in the function domain. We then apply the quality function to these candidates as an abstract fitness measure, the higher will be the better. On the basis of these fitness values some of the better candidates

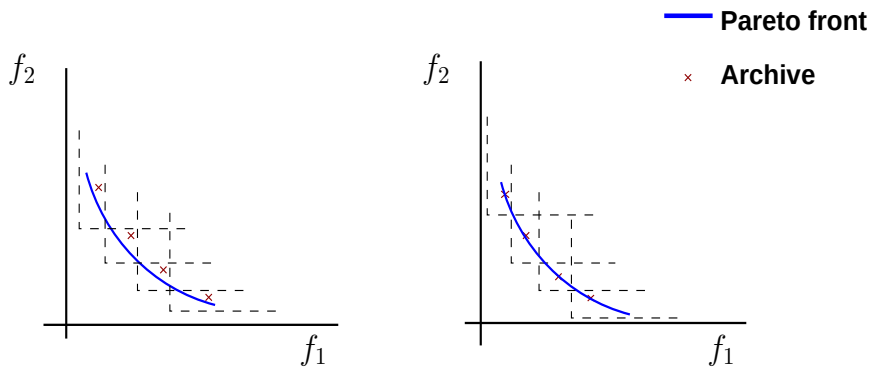


Figure 2.2: Graphs representations the concept of ϵ-approximate Pareto front (left) and ϵ-Pareto front (right).

are chosen to create the next generation. This is done by applying variation operators, recombination (crossover) and mutation, Fig. 2.3 shows a diagram of this technique [16].

There are three main paradigms within evolutionary algorithms: evolutionary programming (EP), evolution strategy (ES), and genetic algorithm (GA). In the following, we give a brief description of them.

Evolutionary programming was proposed by Lawrence J. Fogel [30], who originally adopted finite state machines as predictors and evolved them. The parent selection in EP is deterministic and every member of the population creates exactly one offspring via mutation. The crossover in EP is not used, as the members of the population are viewed as part of a specific species rather than members of the same species.

Evolution strategies were proposed by Ingo Rechenberg and Hans-Paul Schwefel [47], and it is normally used for continuous parameter optimization, adopting the mutation operator as the main means of creating offspring. The mutation operator is based on a normal (Gaussian) distribution (requiring the mean and the standard deviation as parameters). The mutation parameters are usually changed during a run of the algorithm, allowing a good exploration at the beginning and a better exploitation of solutions by the end of the evolutionary process.

Genetic algorithm was initially proposed by John Holland in his book *Adaptation in Natural and Artificial Systems* [22]. Genetic algorithms were originally encoded as binary strings, fitness proportionate selection, a low probability of mutation, and a higher

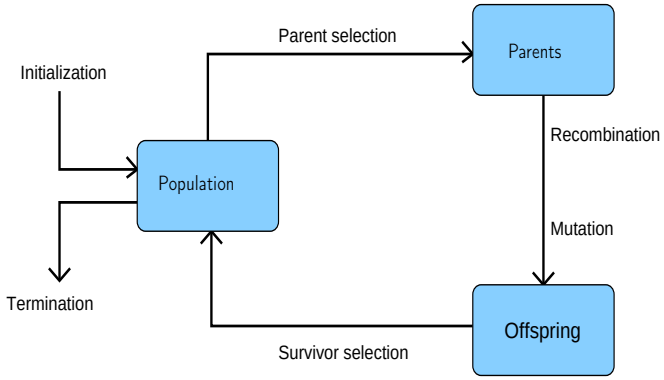


Figure 2.3: General scheme of an evolutionary algorithm.

probability of crossover. This is commonly refered to as the simple genetic algorithm. In this study, we will develop two variation operators, the so called crossover and mutation. The operators are based on a probability distribution function, this allows a real numbers representation, whereas the simple genetic algorithm allows binary representaion. In this work we use real numbers encoding for the decision variables.

2.3 Multi-Objective Evolutionary Algorithms

One widely used class of algorithm to approximate P_Q is given by Multi-Objective Evolutionary Algorithms (MOEAs). Evolutionary algorithms use a selection based on fitness. Moreover, most of these MOEAs use the concept of *dominance* in their search. One advantage of MOEAs is that they are population-based and are thus able to find several candidate solutions in a single run.

This procedure is practical because the user gets an opportunity to investigate a number of other trade-off solutions before choosing an optimal solution. This feature has attracted numerous researchers to develop different MOEAs. Some of the most popular MOEAs in current use are: NSGA-II [23], SPEA2 [15] and ϵ-MOEA [12].

The goal of a MOEA is to obtain an approximation of the solution set with a good quality. To generate such solutions, a MOEA uses a population. They generate new candidates at each iteration using variation operators called crossover and mutation. So far no state-of-the-art MOEA can guarantee convergence toward the set of interest.

There have been many approaches to multi-objective optimization using evolutionary algorithms (EAs), beginning with Schaffer's, vector-evaluated genetic algorithm (VEGA) [40]. In this algorithm, the population is randomly divided into sub-populations that are, each them, assigned a fitness (and subject to selection) according to a different objective function, but the parent selection and recombination were performed globally.

Later, Goldberg suggested the use of fitness based on dominance rather than absolute objective scores [18], coupled with niching and/or speciation methods to preserve diversity[1]. This breakthrough threw a dramatic increase in research activity in this area. Now, there are several Multi-Objective Evolutionary Algorithms. We will discuss the most representative of them.

1. Fonseca and Fleming's **Multi-Objective Genetic Algorithm** (MOGA) [4]. This assigns a raw fitness to each solution equal to the number of members of the current population that it dominates, plus one. It uses fitness sharing amongst solutions of the same rank, coupled with fitness proportional selection to help to promote diversity.

2. Srinivas and Deb's **Nondominated Sorting Genetic Algorithm** (NSGA) [32]. This works in a similar way, but assigns fitness based on dividing the population into a number of 'fronts' of equal domination. Each point in a given front gets as its raw fitness the number af all solutions in inferior fronts.

3. Horn et al.'s **Niched Pareto Genetic Algorithm** (NPGA) [20]. This algorithm differs in that it uses a modified version of tournament selection rather than fitness proportionate with sharing. The tournament operator works by comparing two solutions first on the basis of whether they dominate each other, and then second on the number of similar solutions already in the new population.

4. Zitzler and Thiele's **Strength Pareto Evolutionary Algorithm** (SPEA) [53]. At each generation all the nondominated individuals are copied into an external archive. All of them have a strength value, which is equal to the number of solution that the individual dominates. The diversity in the algorithm is governed by a clustering technique called average linkage method [31].

During the 1990's much work was done elsewhere in the EAs research community, developing methods for reducing dependence on parameter settings. Theoretical breakthroughs

[1] **Diversity** of a population is a measure of the number of different solutions present [16].

were achieved showing that elitism[2] was required in order to keep progress of a MOEA. In the light of this research Deb proposed the revised NSGA-II [23], which achieves elitism using an explicit diversity maintenance scheme, and has a reduced dependence on its parameters.

Another prominent algorithm, is the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [15], which achieves the elitist effect by using an archive containing a fixed number of nondominated points discovered during the search process. It maintains a fixed archive and considers the number of archived points close to a new solution, as well as dominance information, when updating the archive.

2.3.1 Archives

MOEAs can be viewed as particular stochastic search algorithms. A general stochastic search algorithm was presented in [27], and is described in Alg. 2.1.

Algorithm 2.1 Generic Stochastic Search Algorithm

1: $P_0 \subset Q$ drawn at random
2: $A_0 = ArchiveUpdate(P_0, \emptyset)$
3: **for** $j = 0, 1, 2, \ldots$ **do**
4: $\quad P_{j+1} = Generate(P_j)$
5: $\quad A_{j+1} = ArchiveUpdate(P_{j+1}, A_j)$
6: **end for**

Here, P_j represents the candidate set (Population), A_j contains a 'suitable' subset of the obtained data (Archive) at iteration j, *Generate* is a procedure that creates new solutions at each iteration, and *ArchiveUpdate* determines a new archive based on the old archive and the new population.

Remarkably, so far all existing convergence results have been done on archive based MOEAs (e.g., [37, 38, 27, 26, 45, 43, 46]). On the other hand, development of algorithms such as adaptation of variation operators to the particular needs in order to speed up the convergence, is relatively scarce but mandatory to obtain fast and reliable MOEAs.

Laumanns et al. [27] proposed two archiving strategies which keep progress towards the Pareto optimal set mantaining archives consisting of mutually nondominated solutions.

[2] **Elitism** is called to the fact of retaining intact the best individual in each generation.

Moreover, the archive guarantees an upper bound of archive sizes which can be adjusted a priori. However, no convergence of sequence of archives is guaranteed for continuous problems (albeit for discrete problems). An extension to the continuous case was given by Schütze et al. [45, 46].

2.4 ϵ-Multi-Objective Evolutionary Algorithm

The ϵ-MOEA is a steady-state[3] MOEA based on the ϵ-dominance, [12]. The image space is divided into a number of grids (or hyper-boxes) and diversity is maintained by ensuring that a grid or hyper-box can be occupied by only one solution. A block diagram of ϵ-MOEA is depicted in Fig. 2.4.

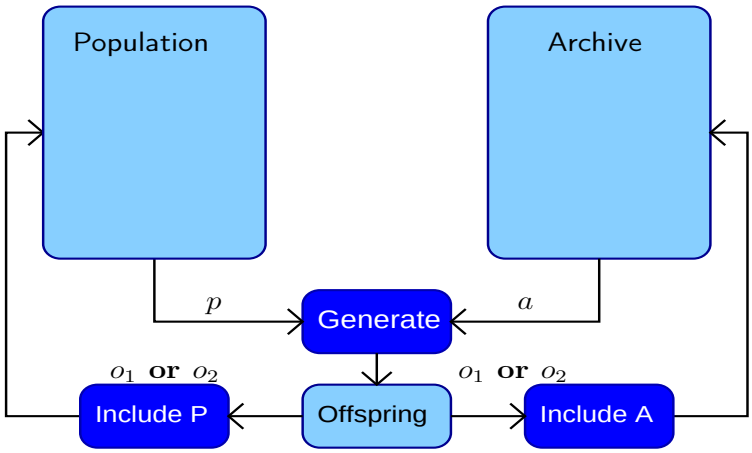


Figure 2.4: A block diagram of ϵ-MOEA.

In the proposed MOEA, there are two populations: (1) a population that contains either dominated or nondominated solutions P_j, and (2) an archive population A_j, with only nondominated solutions, the index j is the iteration counter. A pseudo-code is shown in Alg. 2.2.

ϵ-MOEA begins with an initial population P_0 drawn at random. An archive A_0, is assigned with the nondominated solutions of P_0. Note, until line number 2 of Alg.

[3]**Steady state** refers to a scheme in which every offspring is compared with the parent population as soon as it is created, in contrast with the generational evolutionary algorithm in which all offspring are created before comparing them with the population [16].

Algorithm 2.2 ϵ-MOEA

1: $P_0 \subset Q$ drawn at random
2: $A_0 = \{p' \in P_0 \mid \nexists p \in P_0 \backslash \{p'\}\ p \prec p'\}$
3: **for** $j = 0, 1, \ldots$ **do**
4: for some $p \in P_j$ and $a \in A_j$
5: $O = Generate(p, a)$
6: $P_{j+1} = PopulationUpdate(O, P_j)$
7: $A_{j+1} = ArchiveUpdate\_\epsilon MOEA(O, A_j)$
8: **end for**

2.2 the archive can contain more than one nondominated solution in the same hyperbox. Alg. 2.2 is composed of four stages (*Selection, Generate, PopulationUpdate* and $ArchiveUpdate\_\epsilon MOEA$) which are decribed in the following.

Selection process
To choose a solution p from P_j, two population members from P_j are picked at random and a domination check (in the usual sense) is made. If one solution dominates the other, one is chosen, otherwise, it indicates that two picked solutions are nondominated to each other and one is chosen at random. To choose a solution a from A_j randomly pick a solution from A_j.

Generate
Solutions p and a are used to create two offspring solutions (o_1 and o_2). *Generate* process is driven by the SBX operator [9, 11, 10] (two variants of this operator, for unbounded and bounded domains, will be studied in detail in Section 2.7.4 in page 27).

Population Update
Offspring solutions are compared with the population P_j for their possible inclusion. For dealing whether or not the offspring o_1 or o_2 will replace any population member, it compares the offspring with all the population members. If the offspring dominates one or more population members, then the offspring replaces one of them (chosen at random). If any population member dominates the offspring, it is not accepted. When both test fail, the offspring replaces a randomly chosen population member.

Archive Update
Inclusion into the archive of offspring o_1 or o_2 is compared with each member in the archive for ϵ-dominance. Every solution in the archive is assigned an identification array (box) as follows, for the case in which we are minimizing a MOP:

$$\text{box}(a) = \left\{ \left\lfloor \frac{f_j^{\max}(a) - f_j^{\min}(a)}{\epsilon_j} \right\rfloor \mid j = 1, \ldots, k \right\} \text{ for some } a \in A \tag{2.4}$$

Where $f_j^{\min}$ is the minimum possible value of the j–th objective, $f_j^{\max}$ is the maximum possible value of the j–th objective and ϵ_j is the allowable tolerance in the j–th objective beyond which two values are significant to the user. Notation to refer a component of the vector box will be as $\text{box}_j(a)$ $j \in \{1, \ldots, k\}$. The identification array of an offspring box(o) contains the component of the point o in the image space. Alg. 2.3 shows the mechanism to add an offspring to the archive. The update function is similar to the one described in [27], except for the case when an offspring belongs to the same hyperbox with an archive element. *ArchiveUpdate* in [27] verifies only the dominance, in contrast with the *ArchiveUpdate_ϵMOEA* which checks the Euclidean distance to the **B** vector, lines 10 – 12 of Alg. 2.3.

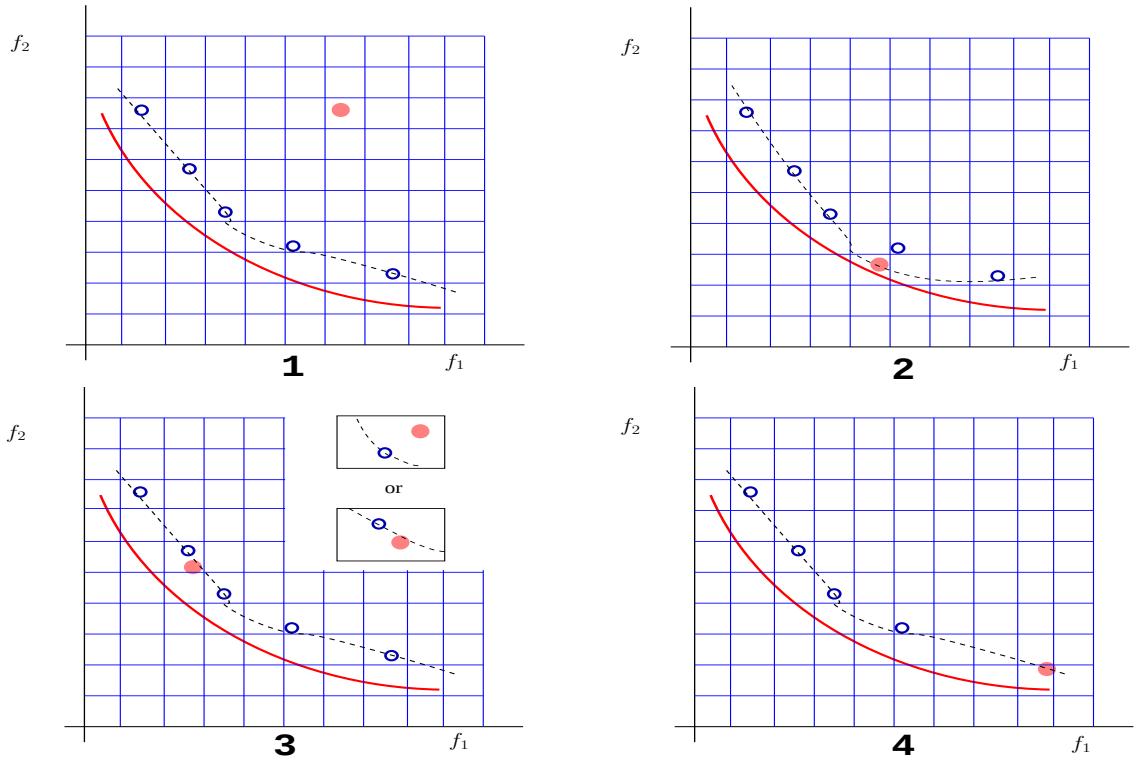


Figure 2.5: Four cases of accepting offspring in the archive are illustrated. Details are on the text below.

Algorithm 2.3 *ArchiveUpdate_ϵMOEA*

Require: A, p
Ensure: A'

$D = \{a \in A \mid box(p) \prec box(a)\}$
if $D \neq \emptyset$ **then**
    $A' = \{p\} \cup A \backslash D$
else
    if $\left(\exists a \in A \mid box(a) = box(p)\right)$ **then**
        if $p \prec a$ **then**
            $A' = \{p\} \cup A \backslash \{a\}$
        else
            if $\left(p \nprec a \text{ and } a \nprec p\right)$ **then**
                $\mathbf{B} = \{\epsilon_1 box_1(p), \ldots, \epsilon_k box_k(p)\}$
                if $\|\mathbf{B} - F(a)\|_2 > \|\mathbf{B} - F(p)\|_2$ **then**
                    $A' = \{p\} \cup A \backslash \{a\}$
                else
                    $A' = A$
                end if
            end if
        end if
    else
        if $\left(\nexists a \in A \mid box(a) \prec box(p)\right)$ **then**
            $A' = A \cup \{p\}$
        else
            $A' = A$
        end if
    end if
end if

Alg. 2.3 and Fig. 2.5 will be used to describe the four cases which may appear in the $ArchiveUpdate\_\epsilon MOEA$ procedure:

1) If the identification array of an offspring box(o) dominates at least one archive element box(a), then the candidate is going to be accepted and all identification arrays which are dominated are deleted, see Fig. 2.5(2), Alg. 2.3 lines 1 – 3.

2) If offspring share the same hyper-box with an archive member, i.e. box(o) = box(a), then they are checked for the usual domination. If the offspring dominates the archive member or is nondominated to the archive member but is closer to the **B** vector than the archive member, then the offspring is accepted and the archive member is deleted, see Fig. 2.5(3), Alg. 2.3 lines 5 – 12.

3) If the offspring does not share the same hyper-box and its hyper-box is not dominated by another hyper-box from an element into the archive, then the candidate is accepted, Fig. 2.5(4), Alg. 2.3 lines 19 and 20.

4) The last event means, that the hyper-box of the offspring is dominated by some hyper-box from a member in the archive and is not accepted, Fig. 2.5(1), Alg. 2.3 lines 14 and 22.

ϵ-MOEA is a steady state algorithm and, it emphasizes nondominated solutions. It does not allows more than one solution per hyper-box but it does not prevent to obtain gaps in the approximation when the Pareto front contains dents or is 'flat' in certain regions [46], see Figs. 2.6 and 2.7.

In recent years, Schütze et al. [45, 46], proposed other update strategies for continuous search spaces, where the sequence of archives converges to different sets of interest in the limit. All the new update strategies make use of ϵ-dominance in order to manage the archive size while maintaining a certain approximation quality in the limit. Further, by adding some features the occurrence of gaps in the limit archives could be prevented.

2.5 *ArchiveUpdate* Procedures

This section will establish four archiving strategies to obtain a finite Pareto set approximations for stochastic multi-objective optimization algorithms working in continuous domains. The four archiving strategies have proved convergence either to the ϵ-approximate Pareto set or to the ϵ-Pareto set in the limit. Furthermore, the author ([45, 46]) gives

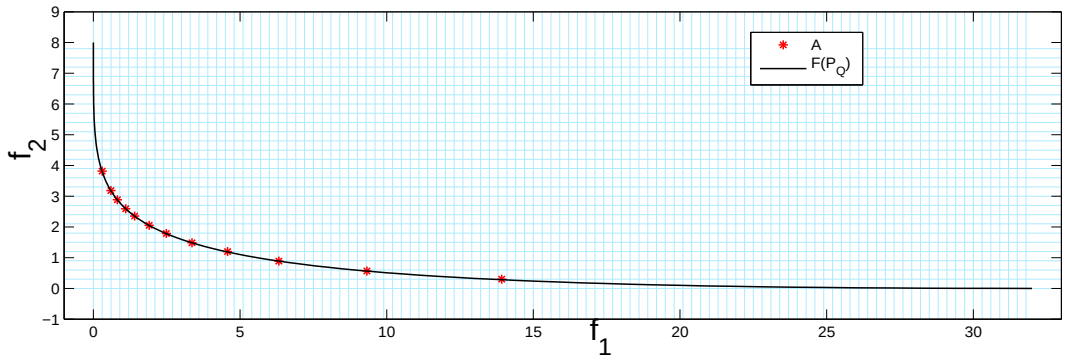


Figure 2.6: ϵ-MOEA was executed with a function that contains some 'flats' regions, the characteristic of applying the dominance over the hyper-boxes implies the loss of the 'flat' region.

bounds on the approximation quality and on the cardinality of the limit solution set. We will give a brief description of them in the following, but the details can be found in [45, 46].

We will assume that the entries of $\epsilon \in \mathbb{R}^k_+$ are 'small' and archives are referred as A.

2.5.1 ArchiveUpdateEps1

An application of the Alg. 2.1, where ArchiveUpdateEps1 (Alg. 2.4) is used to update the archive. This leads to a sequence of archives A_l, $l \in \mathbb{N}$, such that there exists with probability one an $l_0 \in \mathbb{N}$ such that A_l is an ϵ-Approximate Pareto set for all $l \geq l_0$.

Algorithm 2.4 $A = ArchiveUpdateEps1(P, A_0)$

```
A = A_0
for all p ∈ P do
    if ∃a ∈ A : a ≺_ε p then
        continue
    end if
    for all a ∈ A do
        if p ≺ a then
            A = A\{a}
        end if
    end for
    A = A ∪ {p}
end for
```

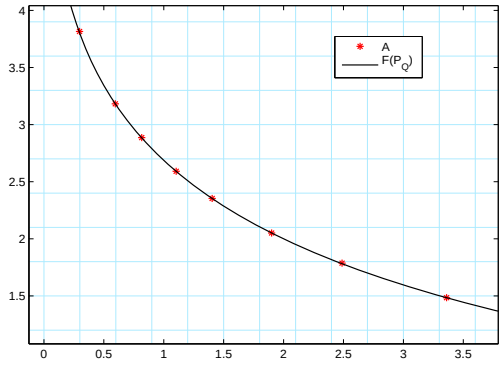


Figure 2.7: A zoom in of Fig. 2.6. Note, there is only one element per hyper-box.

The above archive is very similar to the one poposed in [27]. Line 3 in Alg. 2.4 makes it possible that the sequence of the achives keep a steady state solution which forms an ϵ-approximate Pareto set after finitely many steps. The distance from the archive to the Pareto front (image space), $\text{dist}(F(A), F(P_Q))$ (see Section 2.5.3, Definition 4), might be larger than ϵ, whenever some elements in the archive are not ϵ-dominated and they are just a 'little better' not in all objectives, but enough to be out of the region of ϵ-dominance, see Fig. 2.8.

Further, assume that the elements a_1, a_2 and a_3 are inserted into the archive in this order, (see Fig. 2.8), these points will not be removed in the subsequent steps.

2.5.2 ArchiveUpdateEps2

An application of the Alg. 2.1, where ArchiveUpdateEps2 (Alg. 2.5) is used to update the archive, leads to a sequence of archives A_l, $l \in \mathbb{N}$, such that there exists with probability one an $l_0 \in \mathbb{N}$ such that A_l is an ϵ-approximate Pareto set for all $l \geq l_0$ and $\lim_{l\to\infty} \text{dist}(A_l, P_Q) = 0$.

A new element p, will be added to the archive A by *ArchiveUpdateEps*2 only in one of the following two cases: (1) if there is not an $a \in A$ which ϵ-dominates p and (2), if p dominates an element $a \in A$, which will be in turn discarded from the archive. In the limit all the elements into the archive will be Pareto points but, the quality of Pareto front gets lost. *ArchiveUpdateEps*2 still keeps the problem as *ArchiveUpdateEps*1 explained above. Alg.s 2.4 and 2.5 converge to an ϵ-approximate Pareto set and an ϵ-Pareto set, respectively. However, they can not guarantee that the limit archives do not reveal gaps in the Pareto front.

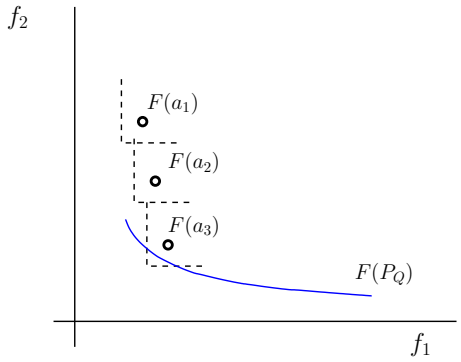


Figure 2.8: Possible example of a set which was generated by *ArchiveUpdateEps*1 with $\text{dist}(F(A), F(P_Q)) >> \epsilon$

Algorithm 2.5 $A = ArchiveUpdateEps2(P, A_0)$

```
1:  A = A_0
2:  for all p ∈ P do
3:      if ∄a ∈ A : a ≺_ε p then
4:          A = A ∪ {p}
5:      end if
6:      for all a ∈ A do
7:          if p ≺ a then
8:              A = A ∪ {p}\{a}
9:          end if
10:     end for
11: end for
```

2.5.3 ArchiveUpdateTight1

We will start with three definitions that are required to explain the next two archives.

Definition 4. Let $u,\ v \in \mathbb{R}^n$ and $A, B \subset \mathbb{R}^n$. The maximum norm distance d_∞, the semi-distance $\text{dist}(\cdot,\cdot)$ and the Hausdorff distance $d_H(\cdot,\cdot)$ are defined as follows:

a) $d_\infty = \max\limits_{i=1,\ldots,n} \mid u_i \text{ - } v_i \mid$

b) $\text{dist}(u, A) = \inf\limits_{v \in A} d_\infty(u, v)$

c) $\text{dist}(B, A) = \sup\limits_{u \in B} d_\infty(u, A)$

d) $d_H(A, B) = \max\ \{\ \text{dist}(A, B),\ \text{dist}(B, A)\ \}$

If it is necessary to specify which metric norm, is being used in order to avoid ambiguity, we may use a subscript, as in $\text{dist}_q(u, A)$. Otherwise, it is assumed that the infinity norm is used.

Definition 5 ([46])**.**

a) A set $A_\epsilon \subset Q$ is called a Δ_M-*tight* ϵ-approximation Pareto set of (MOP) if A_ϵ is an ϵ-approximate Pareto set and

$$\text{dist}(F(P_Q), F(A_\epsilon)) \leq \Delta_M$$

b) A set $A^*_\epsilon \subset Q$ is called a $\Delta_M - tight$ ϵ-Pareto set if A^*_ϵ is an ϵ-Pareto set of (MOP) and $d_H(F(P_Q), F(A^*_\epsilon)) \leq \Delta_M$.

Definition 6 ([46])**.** Let $\epsilon \in \mathbb{R}^*_+$

a) A set $A_\epsilon \subset Q$ with $|A_\epsilon| \geq 2$ is called a (Δ_M, Δ_m)-*tight* ϵ-approximation Pareto set if A^*_ϵ is an $\Delta_M - tight$ ϵ-approximation Pareto set and

$$\text{dist}(F(a), F(A \backslash \{a\})) \geq \Delta_m\ ,\ \forall a \in A$$

b) $(\Delta_M, \Delta_m) - tight$ ϵ-Pareto set is defined analogously.

Point *a*) of Definition 5, says that the archive will be an ϵ-approximation Pareto set and the maximum distance of a point in Pareto front to the image of an archive element is not larger than Δ_M. Point *a*) in definition 6 gives the uniformity level Δ_m. Points *b*), of the two above definitions indicate that all the elements in the archive are Pareto points.

Alg. 2.6 describes the procedure of $ArchiveUpdateTight1$. An application of the Alg. 2.1, where $ArchiveUpdateTight1$ is used to update the archive, leads to a sequence of archives A_l, $l \in \mathbb{N}$, such that there exists with probability one an $l_0 \in \mathbb{N}$ such that A_l is an (Δ, ϵ)-tight ϵ-approximate Pareto set for all $l \geq l_0$.

Algorithm 2.6 $A = ArchiveUpdateTight1(P, A_0)$

```
1: A = A_0
2: for all p ∈ P do
3:     if (∃a ∈ A : a ≺ p) or (∃a_1 ∈ A : a_1 ≺_ε p and ∃a_2 ∈ A : d_∞(F(a_2), F(p)) ≤ Δ)
   then
4:         CONTINUE
5:     end if
6:     for all a ∈ A do
7:         if p ≺ a then
8:             A = A ∪ {a}
9:         end if
10:    end for
11:    A = A ∪ {p}
12: end for
```

2.5.4 ArchiveUpdateTight2

An application of the Alg. 2.1, where $ArchiveUpdateTight2$ is used to update the archive, leads to a sequence of archives A_l, $l \in \mathbb{N}$, such that there exists with probability one an $l_0 \in \mathbb{N}$ such that A_l is an Δ-tight ϵ-approximate Pareto set for all $l \geq l_0$ and $\lim_{l \to \infty} \text{dist}(A_l, P_Q) = 0$

The difference of the two archives $ArchiveUpdateTight1$ and $ArchiveUpdateTight2$ is the strategy to accept a candidate solution $p \in P$. Given an arhive A, $ArchiveUpdateTight1$ accepts p if either there not exists an element into the archive which dominates p or there not exists an element into the archive which ϵ-dominates p and the distance from every point into the archive is bigger than Δ, the same holds for $ArchiveUpdateTight2$. The difference in $ArchiveUpdateTight2$, with respect to $ArchiveUpdateTight1$, is that, it accepts an element p if there exists an element into the archive which is dominated by p,

Algorithm 2.7 $A = ArchiveUpdateTight2(P, A_0)$

```
1: A = A_0
2: for all p ∈ P do
3:     if (∄a ∈ A : a ≺_ε p) or (∄a ∈ A : a ≺ p and ∀a ∈ A : d_∞(F(a), F(p)) > Δ) then
4:         A = A ∪ {p}
5:     end if
6:     for all a ∈ A do
7:         if p ≺ a then
8:             A = A ∪ {p}\{a}
9:         end if
10:    end for
11: end for
```

even if the distance is not bigger that Δ.

2.6 Performance Measures

Assessing the quality of an evolutionary algorithm commonly implies experimental comparisons between the resulting MOEA and other MOEAs or traditional algorithms. The convergence to an approximation solution set and maintenance of a good quality are two distinct and somewhat conflicting goals of multi-objective optimization. No single metric can decide the performance of an algorithm.

To be more precise, given a candidate set (or archive) $A = \{a_1, \ldots, a_N\}$ (in image space) and a discrete (or discretized) Pareto front $F(P_Q) = \{y_1, \ldots, y_M\}$, to refer a component of these points. We will use the notation $a_{i,j}$ for some $i \in \{1, \ldots, |A|\}$ and $j \in \{1, \ldots, k\}$. The following indicators are already used in different MOEA studies.

2.6.1 Error ratio

The error ratio indicator counts the number of candidates that are Pareto points [48].

$$\text{ER} = \frac{\sum_{i=1}^{|A|} e_i}{|A|} \tag{2.5}$$

Where $e_i = 0$ if $a \in F(P_Q)$ for some $a \in A$, and $e_i = 1$, otherwise. Hence, the archive

has to be a subset of the discrete Pareto front in order to obtain a value of zero, see Fig. 2.9.

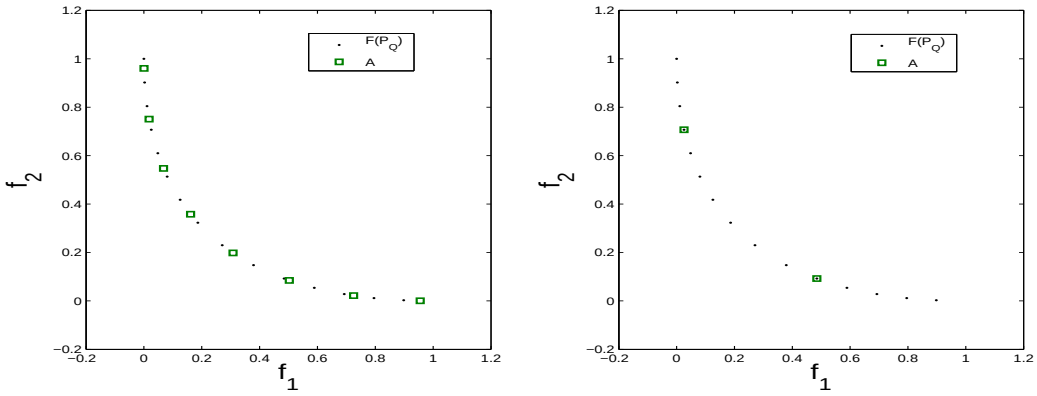


Figure 2.9: To the left, we show a case in which the value of the error ratio is equal to 1, even when all the points into the archive are Pareto points, but they are not a proper subset of the discretization of the Pareto front, $F(P_Q)$. To the right, we show a case in which we obtain the error ratio equal to zero with only two elements in the archive. In this case, both elements in A are Pareto points but the quality is poor.

2.6.2 Set coverage indicator

This indicator calculates the proportion of solutions in A_2 which are dominated by solutions in A_1, [54].

$$\mathcal{C}(A_1, A_2) = \frac{|\{b \in A_2 | \exists a \in A_1 : a \leq b \text{ and } a \neq b\}|}{|A_2|} \tag{2.6}$$

2.6.3 Generational distance

Generational distance (GD) measures, on average, the distance from the archive A to the Pareto front $F(P_Q)$ [48]. A value of zero means that the archive is contained in the Pareto front. If the value of $p = 1$, then the GD has a geometric meaning that is the average of all the distances from the archive to the Pareto front. If all elements into the archive are 'near' the Pareto front, GD will have a low value, even if the archive contains a single element, see Fig. 2.10.

$$\mathrm{GD}(A, F(P_Q)) = \frac{1}{|A|}\left(\sum_{i=1}^{|A|} \mathrm{dist}_2(a_i, F(P_Q))^p\right)^{\frac{1}{p}} \tag{2.7}$$

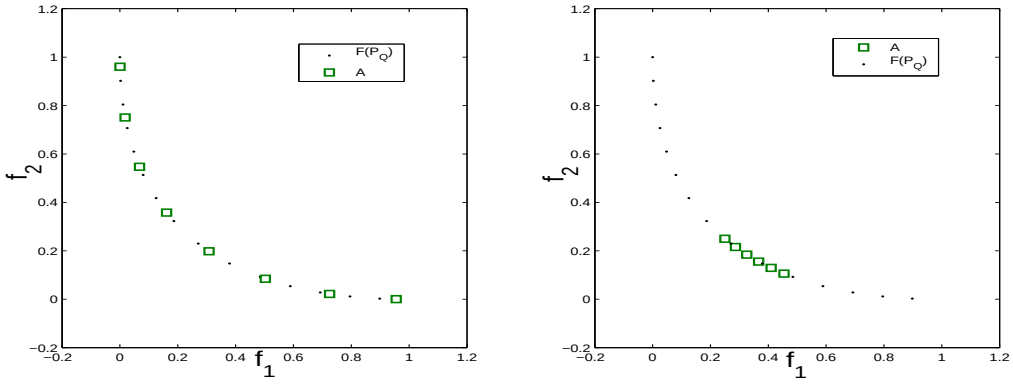


Figure 2.10: To the left, we show a case in which the archive contains Pareto points, but they are not a subset of the discrete Pareto front. To the right, we show a case in which the archive contains Pareto points, but all of them are near to each other and the value of GD might be similar to the previous case.

2.6.4 Inverted generational distance

The Inverted Generational Distance (IGD) indicator measures, on average, the distance from the Pareto front $F(P_Q)$, to the archive A [5]. The IGD indicator gets a low value only if all the elements in the archive are too close to the Pareto front, but all of them are distributed across the Pareto front. If the archive contains one or more elements far away from the rest of the elements (outliers). Those points have no influence in the IGD value, see Fig. 2.11.

$$\mathrm{IGD}(A, F(P_Q)) = \frac{1}{|F(P_Q)|}\left(\sum_{i=1}^{|F(P_Q)|} \mathrm{dist}_2(y_i, A)^p\right)^{\frac{1}{p}} \tag{2.8}$$

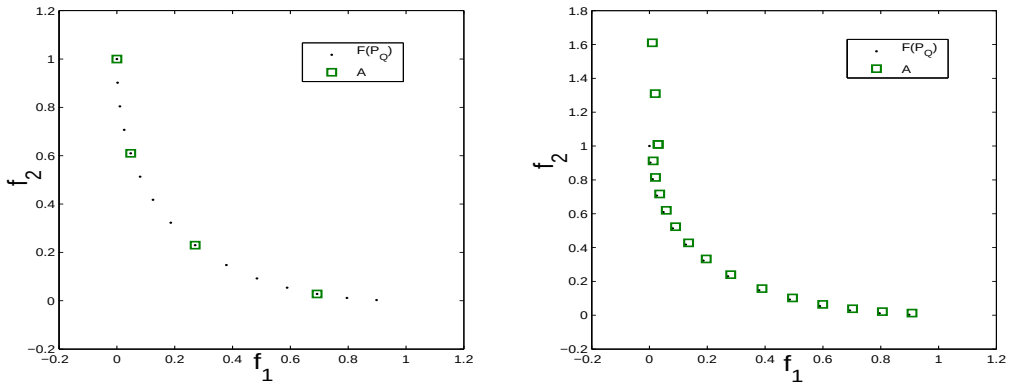


Figure 2.11: To the left, we show a case in which the IGD value is not zero, even when the archive is a subset of the discrete Pareto front. To right, we show a case in which the archive has a good approximation with respect to the discrete Pareto front but it contains outliers that are not detected by the IGD indicator.

2.6.5 ϵ-indicator

The ϵ-indicator measures the smallest amount ϵ, that must be used to move the archive contents such that every point in $F(P_Q)$ is covered [6]. A good approximation of ϵ is the maximal of the minimal distances from the Pareto front to the archive, i.e. $\text{dist}_2(F(P_Q), A)$, see Fig. 2.12.

$$I_\epsilon(A, P_Q) = \min\{\epsilon \in \mathbb{R} \mid \forall y \in F(P_Q) \exists a \in A : a - \epsilon \leq_\epsilon y - \epsilon \text{ and } a - \epsilon \neq y - \epsilon\} \quad (2.9)$$

2.6.6 Hypervolume

The hypervolume Pareto compliant indicator is defined as the area of coverage by the archive in image space [7, 52]. This is equal to the summation of all the rectangular areas, bounded by some reference point, $r \in \mathbb{R}^k$, see Fig. 2.13.

$$HYP(A) = \text{volume}\left(\bigcup_{i=1}^{|A|}[r_1, a_{i,1}] \times \cdots \times [r_k, a_{i,k}]\right) \quad (2.10)$$

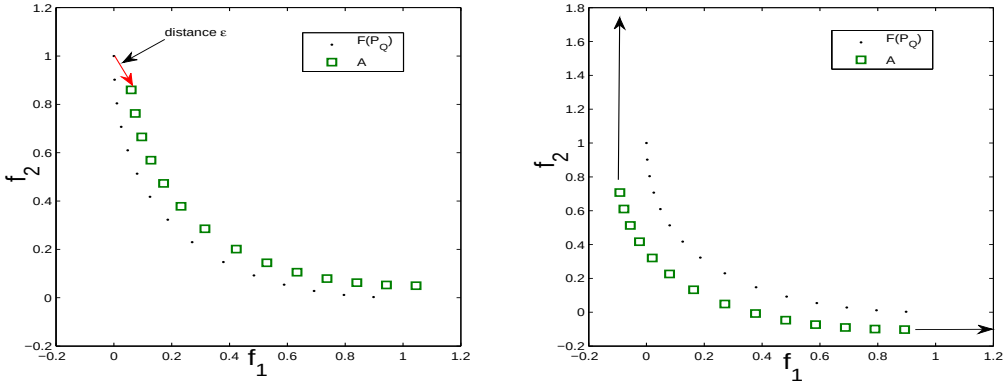


Figure 2.12: A red arrow represents maximal distance from Pareto front to the archive (left). Elements in the archive with the substraction of $\epsilon = \text{dist}_2(F(P_Q), A)$ (right).

2.6.7 Spacing

The spacing indicator describes the spread of the vectors in the archive [7, 41], i.e., it measures the distance variance of neighboring vectors into the archive.

$$\hat{\mathbf{d}}(\hat{a}) = \min_{a \in A \setminus \{\hat{a}\}} \|\hat{a} - a\|_1 \qquad \overline{\mathbf{d}} = \frac{\sum_{i=1}^{|A|} \hat{\mathbf{d}}(\mathbf{a_i})}{|A|}$$

$$S(A) = \sqrt{\frac{1}{|A| - 1} \sum_{i=1}^{|A|} (\overline{\mathbf{d}} - \hat{\mathbf{d}}(a_i))^2} \tag{2.11}$$

2.7 Variation Operators

Genetic algorithms are principally composed by two operators. First, the recombination (crossover), uses information from two parents (individuals) to generate the offspring. The second operator, mutation, is applied to one offspring and delivers a modification of it. This section will explain some of the variation operators for real numbers representation. We will emphasize here on the Simulated Binary Crossover (SBX) and Polynomial based Mutation (PM) [9].

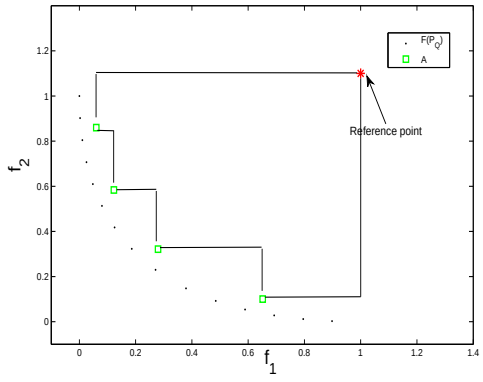


Figure 2.13: The hypervolume enclosed by the non dominated solutions.

The notation used to distingish parents and offspring will be: p_1, p_2 and o_1, $o_2 \in \mathbb{R}^n$ for parents and offspring, respectively. We will use $\tilde{o}$ for a mutated offspring. Index i will be used to express the components of the parents and offspring, $i = 1, \ldots, n$, and the j index will be used to refer to one parent or offspring, $j \in \{1, 2\}$.

2.7.1 Intermediate crossover

Intermediate crossover (IC) is one of the two recombination mechanisms in evolution strategies [47, 1] . Intermediate crossover indicates that the components of the offspring are obtained by calculating the arithmetic mean of the corresponding component of both parents. Schwefel proposed the generalized intermediate crossover (GIC) by allowing an arbitrary number α in the interval [0, 1], and furthermore choose a new value for every component [47].

$$o_{1,i} = \begin{cases} p_{1,i} + (p_{2,i} - p_{1,i})/2 & \text{intermediate crossover} \\ p_{1,i} + \alpha(p_{2,i} - p_{1,i}) & \text{generalized intermediate crossover} \end{cases} \quad i = 1, \ldots, n$$

Intermediate and generalized intermediate crossover always assume $p_1 \leq_p p_2$, see Fig. 2.14. GIC leads to the offspring over the line in which p_1 and p_2 are in parameter space. We can see, this operator as $o = p_1 + \alpha\nu$ where direction $\nu = (p_2 - p_1)$.

2.7.2 Blend crossover

Blend crossover (BLX-α) requires two parent solutions [17]. It always assumes $p_1 \leq_p p_2$, BLX-α randomly picks a solution in the range $[p_{1,i} - \alpha(p_{2,i} - p_{1,i}),\ p_{2,i} + \alpha(p_{2,i} - p_{1,i})]$.

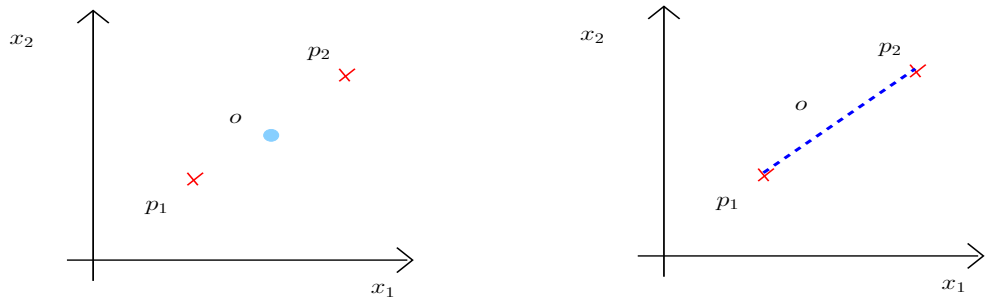


Figure 2.14: Example in $\mathbb{R}^2$, intermediate crossover yields the center (left). Generalized intermediate crossover allows for a result located anywhere on the diagonal line (right).

For every component of the offspring a random number u in [0, 1] will be chosen. α is any non negative real value. If α approaches to zero, the offspring will be enclosed by their parents. If α gets bigger, the probability to obtain an offspring far away increases, see Fig. 2.15. Offspring, then will be as follows.

$$\gamma_i = (1 + 2\alpha)u - \alpha$$

$$o_{j,i} = (1 - \alpha)p_{1,i} + \alpha p_{2,i} \; i = 1, \ldots, n$$

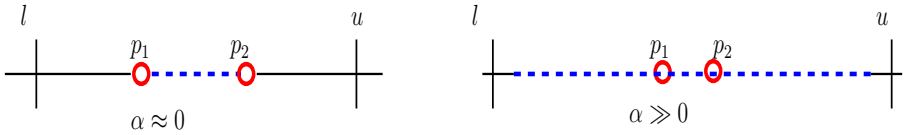


Figure 2.15: Example in one dimension $p_1, p_2, u, l \in \mathbb{R}$. If α is near zero, the offspring will be located anywhere in the dot line (left). If α is bigger than zero the offspring will appear far away (right).

2.7.3 Fuzzy recombination operator

The fuzzy recombination (FR) operator creates offspring based on a triangular probability distribution ϕ [49]. This distribution has its peak located at the parent solution and the base is proportional to the difference in parents times a tunable parameter $d \in \mathbb{R}$. As d is large, solutions away from the parents can get created, but always the probability Pr, to get an offspring near to their parents is high, Fig. 2.16.

$$Pr(o_j) \in \{\phi(x_i), \phi(y_i)\}$$

with a triangular probability distributions $\phi(r)$ having the values $p_{1,i}$ and $p_{2,i}$ ($p_{1,i} \leq p_{2,i}$ always hold).

$$\begin{aligned} p_{1,i} - d \cdot (p_{2,i} - p_{1,i}) \leq r \leq p_{1,i} + d \cdot (p_{2,i} - p_{1,i}) \\ p_{2,i} - d \cdot (p_{2,i} - p_{1,i}) \leq r \leq p_{2,i} + d \cdot (p_{2,i} - p_{1,i}) \end{aligned}$$

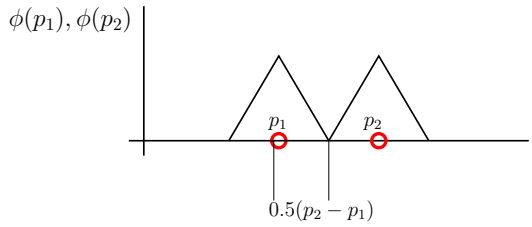


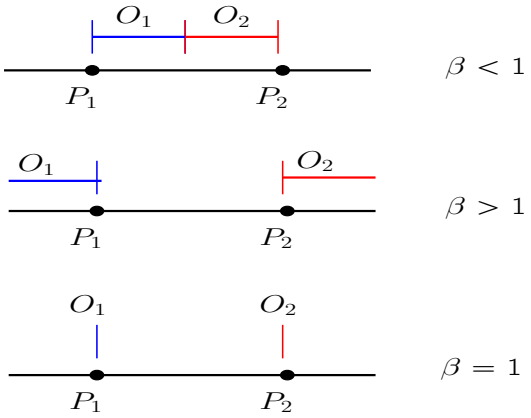
Figure 2.16: Example of fuzzy recombination, $p_1, p_2 \in \mathbb{R}$. This example shows the distribution probability function with $d = 0.5$. Triangular distributions do not overlap to each other.

2.7.4 Simulated binary crossover

Deb et al. developed the Simulated Binary Cossover (SBX) which requires two parent solutions to create two children solutions [9]. The SBX operator simulates the single point crossover operator in binary strings. The author shows that the average of the decoded values (the real number to binary string customized for GA) are the same before and after the crossover operator. The distance from the offspring to their parents is given by $h_i = \frac{1}{2}\beta(p_{2,i} - p_{1,i})$, assume $p_1 \leq_p p_2$. The distance h is regulated by the value β, see Fig. 2.17. This value is obtained by a probability distribution function (*pdf*) [4]. Offspring are calculated as follows:

$$\begin{aligned} o_{1,i} &= \tfrac{1}{2}(p_{1,i} + p_{2,i}) - h_i \\ o_{2,i} &= \tfrac{1}{2}(p_{1,i} + p_{2,i}) + h_i \end{aligned} \tag{2.12}$$

[4]**Probability Distribution Function (*pdf*)** of a continuous random variable is a function that describes the relative likelihood for this random variable to occur at a given point in some interval [19].



Example in one dimension of the distance h, derived from SBX. Offspring are enclosed by the parents when $\beta < 1$. If $\beta > 1$, then the offspring points enclose the parents. Otherwise if $\beta = 1$, the offspring are the same to their parents.

Figure 2.17:

The probability distribution used to create an offspring is given as follows.

$$\mathcal{P}(\beta) = \begin{cases} \frac{1}{2}(\eta_c + 1)\beta^{\eta_c} & \text{if } \beta \leq 1, \\ \\ \frac{1}{2}(\eta_c + 1)\frac{1}{(\beta^{\eta_c+2})} & \text{otherwise} \end{cases} \tag{2.13}$$

The value of η_c is any non-negative real number. A large value of η_c gives a higher probability for creating the offspring near to its parents and, a small value of η_c allows distant offspring. Fig. 2.18 shows a plot of the *pdf* with different values of η_c. There is a 50% chance that $\beta < 1$ and the rest that $\beta > 1$. This chance, is only for the unbounded case. We will see later on that is not true for a bounded domain.

We calculate the β value by equating the area under the probability curve $u \in [0, 1]$, as follows.

$$\beta = \begin{cases} (2u)^{\frac{1}{\eta_c+1}} & \text{if } u \leq 0.5, \\ \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{\eta_c+1}} & \text{otherwise} \end{cases} \tag{2.14}$$

Algorithm 2.8 gives a procedure to create two offspring (o_1 and o_2) from two parent solutions p_1 and p_2.

Fig. 2.19 shows an example of the SBX operator in $\mathbb{R}^2$ with a feasible region $Q = [0, 1]^2$. Some offspring are out of the feasible region, specially with an small value of η_c.

Since we are dealing with a bounded domain Q, we have to know the largest possible value of β. Note that the β value does not represent the step size. We observed that

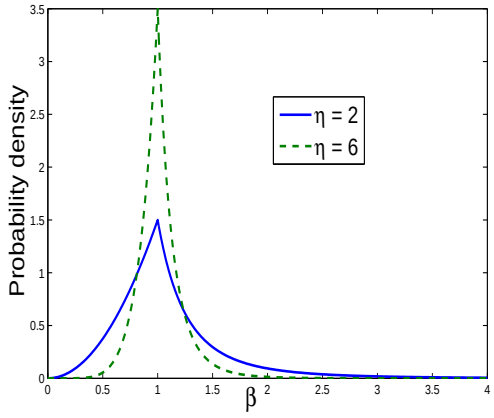


Figure 2.18: Probability distribution function, used to calculate the value of β, using different values of η_c.

Algorithm 2.8 $\{o_1, o_2\} = \text{SBX}(p_1, p_2)$

1: **for** $i = 1, \ldots, n$ **do**
2: choose a random number $u \in [0, 1]$
3: calculate the value of β, using Eq. (2.14)
4: compute the offspring by using the Eq. (2.12)
5: **end for**

when β is in the interval [0, 1], and the size h_i goes from 0 to $\frac{1}{2}\big(p_{2,i} - p_{1,i}\big)$, the offspring will appear in feasible region, because they are going to be enclosed by their parents. The task is find the greatest value of β, given the nearest parent to the boundary $\min(p_{1,i} - l_i, u_i - p_{2,i})$. Therefore, the greatest value of β is calculated as follows (A bounded version of SBX was proposed by Deb et al. [10]).

$$\beta_{\max,i} = 1 + \frac{2}{p_{2,i} - p_{1,i}} \min(p_{1,i} - l_i, u_i - p_{2,i}) \tag{2.15}$$

Eq. (2.15) obtains the greatest value of β assuming $p_1 \leq_p p_2$ holds all the time. In the mean time we know the largest value of $\beta_{\max,i}$ for a bounded domain Q, but it is necesary to modify the *pdf* for purpose of a bounded domain. Theoretically, the domain of *pdf* in Eq. (2.13) is $[0, \infty)$. Indeed, as u approaches to one, the value of β goes to infinity (Eq. (2.14)). The *pdf* for bounded domains is given by the Eq. (2.16).

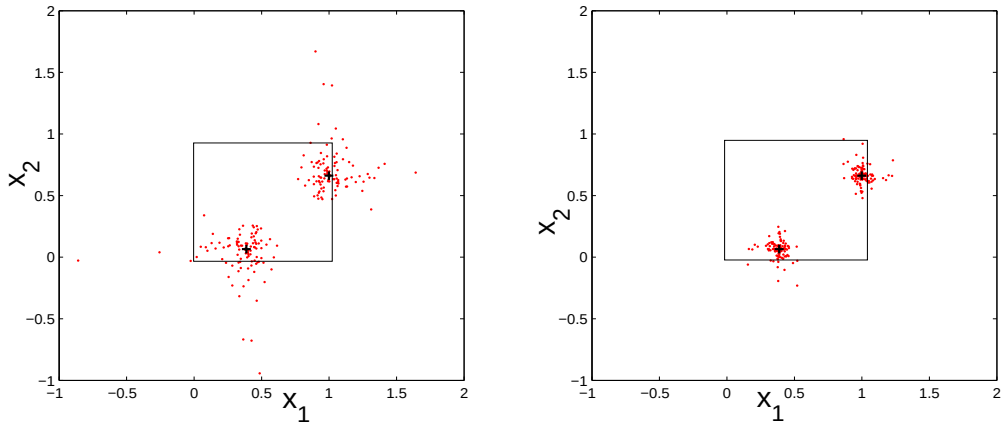


Figure 2.19: An example of the SBX operator in $\mathbb{R}^2$ with a feasible region $Q = [0, 1]^2$. The parents are marked with a cross and offspring with dots. 100 offspring with $\eta_c = 2$ (left), 100 offspring with $\eta_c = 6$ (right).

$$\mathcal{P}(\beta) = \begin{cases} \frac{1}{\alpha}(\eta_c + 1)\beta^{\eta_c} & \text{if } \beta \leq 1 \\ \\ \frac{1}{\alpha}(\eta_c + 1)\frac{1}{(\beta^{\eta_c+2})} & \text{otherwise } \beta \in (1, \beta_{max}] \end{cases} \tag{2.16}$$

The difference in Eq. (2.16) with respect to Eq. (2.13) is an extra parameter α. Parameter α is not adjusted a priori as η_c. It is the trick to obtain offspring inside the feasible region. It allows that the probability goes from 0 to 1 in the interval $\beta \in [0, 1]$ and all offspring will appear inside the domain as a result, i.e. the *pdf* gets cut according to the value of α for a given set of parents. Eq. (2.17) gives the value of β for the previous *pdf* Eq. (2.16). This is equal for the unbounded SBX with $\alpha = 2$.

$$\beta = \begin{cases} (\alpha u)^{\frac{1}{\eta_c+1}} & \text{if } u \leq \frac{1}{\alpha} \\ \\ \left(\frac{1}{2-u\alpha}\right)^{\frac{1}{\eta_c+1}} & \text{otherwise} \end{cases} \tag{2.17}$$

Extra parameter α depends of $\beta_{\max}$. If α is near to one, then one of the parents is too close to the boundary and the offspring will appear enclosed by their parents (*pdf* has sense in the interval [0, 1] or very near to this interval). If α is close to two, then parents are far away to the boundary and the *pdf* looks like the Eq. (2.13). α is given by Eq. (2.18).

$$\alpha = 2 - \beta_{\max}^{-(\eta_c+1)} \tag{2.18}$$

Algorithm 2.9 makes a sketch to use SBX for bounded domains.

Algorithm 2.9 $\{o_1, o_2\} = \text{SBX}(p_1, p_2)$

1: **for** $i = 1, \ldots, n$ **do**
2: choose a random number $u \in [0, 1]$
3: calculate the value of β, using Eqs. (2.17), (2.18) and (2.15)
4: compute offspring by using Eq. (2.12)
5: **end for**

Example 2. This example illustrates the mechanism of the bounded SBX and the difference with itself for the unbounded algorithm. Given two parents p_1 and $p_2 \in \mathbb{R}^2$, $p_1 = (0.1, 0.0)$, $p_2 = (0.5, 0.5)$, $\eta_c = 2$ and a feasible region $Q = [0, 1]^2$. The maximum beta value to every comoponet are: $\beta_{\max,1} = 1.5$, $\alpha_1 = 1.7$ and $\beta_{\max,2} = 1$, $\alpha_2 = 1$ for the first and the second component, respectively, see Eq. (2.15) and (2.18). The first component of offspring either appears enclosed by its parents or at most it will have a step size of 0.1, because the nearest parent is 0.1 to the boundary. The second component of offspring must be enclosed by its parents. There exists a parent which is in the boundary, Fig. 2.20 shows 100 runs of Algorithm 2.9.

pdf changes its domain according to the values of α and $\beta_{\max}$. The porpuse of the latter is to avoid acquiring offspring out of the domain Q, but it already holds the property $\int_0^{\beta_{\max}} \mathrm{d}\beta = 1$. The chance $\beta < 1$ could be bigger than 50%. Fig. 2.21 shows the domain of *pdf* to the previos example.

SBX is endowed to generates offspring inside the domain while decreases its capability to cover all the parameter space. Moreover, all Eq.s (2.12) – (2.18) expect that $p_1 \leq_p p_2$. Indeed, this can keep the balance, swapping every $p_{1,i} > p_{2,i}$. The effect of swapping some, but not all components $p_{1,i}$ and $p_{2,i}$, leads to parents to become completely different than the old ones (image space too). The authors of SBX applied this swap, as can be seen in his implementations within several algorithms[5].

[5] **http://www.iitk.ac.in/kangal/codes.shtml** contains implementations of the following two algorithms; NSGA-II and ϵ-MOEA, both of which use SBX.

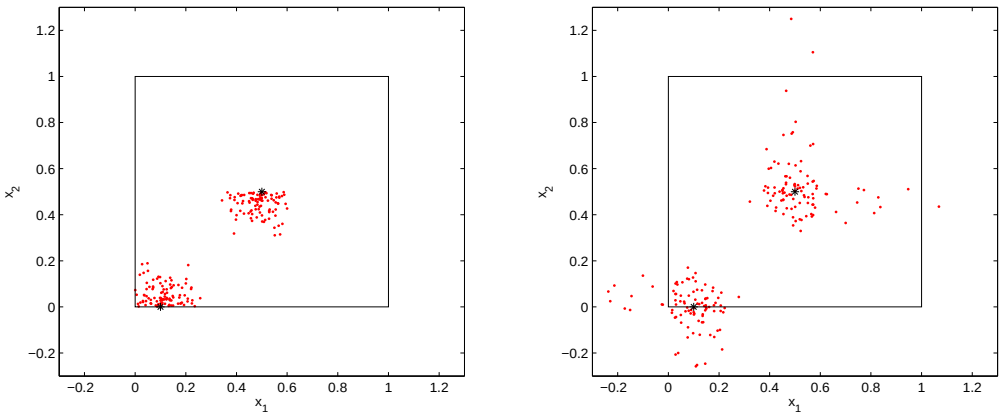


Figure 2.20: Algorithm 2.9 generates 100 offspring. The second parent $p_2 = (0.5, 0.5)$ has its offspring at most 0.1 distance from itself over the first component, altough it is far enough to the boundary and the second component is enclosed by $p_{1,2}$ and $p_{2,2}$ (left). Algorithm 2.8 generates 100 offspring, with a low value of $\eta_c = 2$. There is a posibility to obtain offspring outside the feasible region (right).

2.7.5 Polynomial mutation

The Polynomial Mutation (PM) operator is based on a *pdf* similar to SBX. PM makes a small perturbation to offspring o, to obtain the mutated offspring $\tilde{o}$. The maximum step size is equal to the difference in the upper and lower bound [10].

$$\tilde{o_i} = o_i + \beta(u_i - l_i) \ , \ \beta \in [-1, 1] \tag{2.19}$$

where parameter β is calculated from the probability distribution $\mathcal{P}(\beta) = 0.5(\eta_m + 1)(1 - |\beta|)^{\eta_m}$, $\beta \in [-1, 1]$:

$$\beta = \begin{cases} (2u)^{\frac{1}{\eta_m + 1}} - 1 & \text{if } u < 0.5 \\ \\ 1 - \big(2 - 2u\big)^{\frac{1}{\eta_m + 1}} & \text{otherwise} \end{cases} \tag{2.20}$$

Algorithm 2.10 illustrates the procedure of PM.

u is a random number in the interval [0, 1], Algorithm 2.10 line 2. It represents the area under the *pdf* and is used to calculate β, η_m has the same role that η_c in the SBX operator (Eq. (2.20)). Fig. 2.22 shows an example of the PM operator and its *pdf* for differents values of η_m.

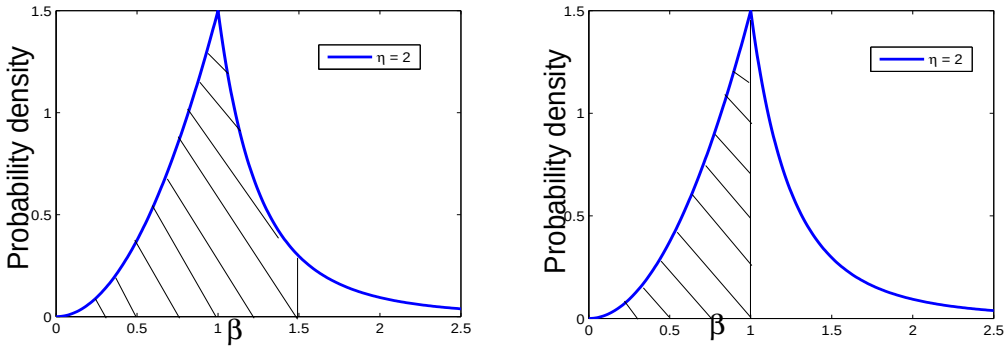


Figure 2.21: Sloping lines under the curve cover the whole domain of the *pdf* with $\eta_c = 2$, $\beta \in [0, 1.5]$. The first component for Example 2 (in page 31) is shown at the left and $\beta \in [0, 1]$ for the second component in Example 2 (in page 31) is shown to the right.

Algorithm 2.10 $\tilde{o} = \mathrm{PM}(o)$

1: **for** $i = 1, \ldots, n$ **do**
2: choose a random number $u \in [0, 1]$
3: calculate the value of β, using Eq. (2.20)
4: compute mutation by using Eq. (2.19)
5: **end for**

Whenever an offspring has mutated there is a chance that it appears outside the feasible domain. A solution to this problem is the same as in SBX. It adds an extra parameter α, which is not adjusted a priori. It gives a relation between the nearest boundary and the size of the domain as follows.

$$\alpha_i = \frac{\min(o_i - l_i, u_i - o_i)}{u_i - l_i} \tag{2.21}$$

PM applies the same idea of modify domain of the *pdf* in order to suit it, (Eq. (2.19)) for a bounded domain. The latter increases the probability that the mutated offspring is near to the offspring. As a result, if the offspring is too close to the boundary, mutations become 'small' perturbations and the rest of the feasible region is ignored. Here we give a posible probability distribution function for PM, Eq. (2.22).

$$\mathcal{P}(\beta) = \frac{(1 - |\beta|)^{\eta_m}(\eta_m + 1)}{2(1 - (1 - \alpha)^{\eta_m + 1})} \ , \ \beta \in [-\alpha, \alpha] \tag{2.22}$$

Eq. (2.23) gives the value of beta, given a random number $u \in [0, 1]$ which represents

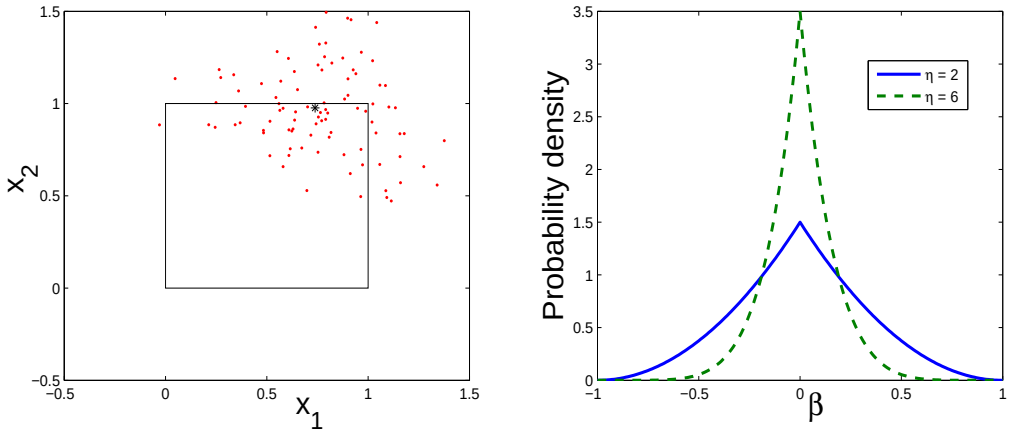


Figure 2.22: Feasible region $Q = [0,1]^2$. Algorithm 2.10 generated 100 mutations of an offspring. In some cases the mutated individual is outside the feasible region (left). *pdf* used in polynomial mutation with $\eta_m = 2$ and 6 (right).

the area under the curve.

$$\beta = \begin{cases} \left(2u + (1-2u)(1-\alpha)^{\eta_m+1}\right)^{\frac{1}{\eta_m+1}} - 1 & \text{if } u \leq 0.5 \\ 1 - \left(2(1-u) + 2(u-0.5)(1-\alpha)^{\frac{1}{\eta_m+1}}\right)^{\frac{1}{\eta_m+1}} & \text{otherwise} \end{cases} \tag{2.23}$$

Example 3. To ilustrate polynomial mutation we take the next points $o \in \mathbb{R}^2$ and a feasible region is limited by the two vectors $l = (0,0)$ and $u = (1,1)$, i.e. $Q = [0,1]^2$. The value of α is equal in both components $\alpha = 0.1$. The maximum step size in which offspring will mutate is 0.1, because the minimal distance from o to the boundary is 0.1. Nevertheless, it is far enough from the upper bound, Fig. 2.23.

The consequence of reducing the domain of the *pdf* is that this becomes a 'small' perturbation whenever offspring is near to the boundary. Moreover *pdf* still holds the property $\int_{-\alpha}^{\alpha} \mathcal{P}(\beta)\mathrm{d}\beta = 1$.

The SBX and PM operators looks like a small movement from either the parents or offspring, respectively. In any case, we can express those operators as $o = p + h\nu$ for crossover and $\tilde{o} = o + h\nu$ for mutation and $h \in \mathbb{R}$. The direction is choosen at random $\nu \in \mathbb{R}^n$. The latter will hold for SBX whenever $p_1 \leq_p p_2$.

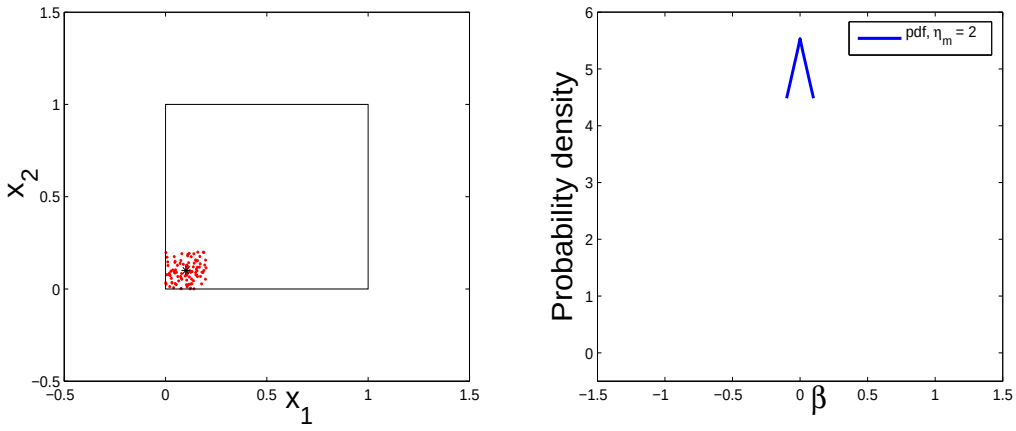


Figure 2.23: The maximum step size of offspring will be at most 0.1, 100 mutations η_m = 2 (left). *pdf* is suited in the domain $\beta \in [-0.1, 0.1]$ (right).

Chapter 3

Archive Based MOEA

In this work, we make a first attempt to design variation operators that are tailored to a particular class of algorithms (i.e., archivers that are based on the concept of ϵ-dominance) that allow for a finite size representation of the solution set of a MOP while maintaining a certain approximation quality. The resulting MOEA can be viewed as a variant of ϵ-MOEA ([12], see also Chapter 2).

3.1 The Algorithmic Framework

The algorithms that we describe in this chapter follow the same scheme which we describe here: for every point $x \in Q$ under consideration we define first a search direction $\nu \in \mathbb{R}^n$ and then a step size $h \in \mathbb{R}_+$, i.e., each new candidate solution x_{new} is generated from x as follows

$$x_{\text{new}} = x + h\nu. \tag{3.1}$$

That is, our methods have a certain relation to line search methods used in 'classical' numerical optimization (e.g., [33]). To maintain feasibility (i.e., to ensure that $x_{\text{new}} \in Q$) we have chosen to restrict ourselves to step sizes $h \leq h_{\max}$ such that x_{new} is inside the domain (i.e., we apply no projection methods). Since Q is a box, $h_{\max}$ can be computed explicitly: it is $h_{\max} = \min_{i=1,..,n} h_i$, where

$$h_i = \begin{cases} (u_i - x_i)/\nu_i & \text{if } \nu_i > 0 \\ (x_i - l_i)/\nu_i & \text{if } \nu_i < 0 \\ \infty & \text{otherwise} \end{cases} \tag{3.2}$$

Another aspect which is used in the operators is the desired distance between archive entries. Here, we are particularly interested in the use of archivers that are based on

ϵ-dominance, and all such archivers share one characteristic: if two points $x, x_{\text{new}} \in Q$ are 'near' to each other, x_{new} is ϵ-dominated by x. Thus, it is very likely that this new candidate solution is discarded from the archive regardless if x_{new} represents an improvement or not (in fact, this is a general major drawback of the discretization, e.g. as is done in ϵ-MOEA. One can include dominance replacements in the archive, as is done in *ArchiveUpdateEps*2. However, this leads typically to a larger upper bound of the archive magnitudes and, hence to a large computational effort, see [45]). Hence, too short step sizes h should be avoided in (3.1). Given an archive entry $a \in Q$ which is already near to P_Q, an optimal 'neighbor solution' of a is located along P_Q, and the 'ideal' distance from $F(a)$ to $F(b)$ is certainly

$$\|F(b) - F(a)\|_\infty \approx 2\epsilon \tag{3.3}$$

To make use of this desired distance (note that (3.3) is defined in objective space while (3.1) is defined in parameter space) we can utilize the Lipschitz continuity of F: assume F is Lipschitz continuous with Lipschitz constant L.

Definition 7 (Lipschitz constant)**.** A function $f : Q \to \mathbb{R}$ is called *Lipschitz continuous* on Q if there exist an $L \geq 0$ s.t.

$$\|f(x) - f(y)\| \leq L\|x - y\| \qquad \forall x, y \in Q \tag{3.4}$$

L is called the *Lipschitz constant* of f where L is chosen as the smallest value such that (3.4) holds

The Lipschitz constant gives us a measure of how the relation between the parameter and image space is. With the Lipschitz constant we can know whether a small perturbation in the parameter space yields a small change in the image space, or the opposite, a small perturbation yields a big change in the image space.

We can estimate the distance of a and b by

$$\|a - b\|_\infty \approx \frac{2\epsilon}{L}. \tag{3.5}$$

Hence, an estimation for the step size h^* to obtain the desired distance in objective space is (assuming that b is generated by a as in (3.1)):

$$h^* \approx \frac{2\epsilon}{L\|\nu\|_\infty}. \tag{3.6}$$

It remains how to approximate L. Since we may assume that ϵ is 'small' and hence a and b are 'close' together, it is sufficient to estimate L at a in direction ν (here $\nu = b - a$). For this, one can e.g. use

$$L_a \approx \|\nabla_\nu f_1(a), \ldots, \nabla_\nu f_k(a)\|_\infty, \tag{3.7}$$

where $\nabla_\nu f_i(a)$ denotes the directional derivative of f_i at a along ν. If gradient information is not at hand, or its use/approximation should be avoided due to the computational cost, one can alternatively approximate L_a by

$$L_a \approx \frac{\|F(a + \bar{t}\nu) - F(a)\|_\infty}{\bar{t}\|\nu\|_\infty}, \tag{3.8}$$

where $\bar{t} \in \mathbb{R}_+$ is 'small'. The cost for the latter approximation is apparently given by one additional function call.

In the above approximation, to estimate the step size h^* using the Lipschitz constant, we have the ideal step size to obtain a distance of 2ϵ in image space, but this operator is going to be developed to be embedded into a genetic algorithm. This implies that the approximation of the step size h^* could be endowed to obtain it in a probabilistic way, because the genetic algorithm works in a probabilistic environment.

The random process, for which the step size will be calculated, will have a probability distribution function, *pdf*. It has a peak next to the ideal step size, h^*, and a lower probability to appear near to the parent or near to the boundary. The *pdf* must respect the position where the probability is higher. This position changes in order to suit the Lipschitz constant, and the domain of the *pdf* has to go from the parent to the boundary in the direction of ν. The new *pdf* will have an external parameter η. The purpose of it is similar to the operators SBX and PM. η increases the probability near to h^* for the new *pdf*. The probability for the step size h is given by

$$\mathcal{P}(h) = \begin{cases} \frac{1}{h_{\max}}(\eta+1)\left(\frac{h}{h^*}\right)^\eta & 0 \le h \le h^* \\ \\ \frac{1}{h_{\max}}(\eta+1)\left(\frac{h_{\max}-h}{h_{\max}-h^*}\right)^\eta & h^* < h \le h_{\max} \end{cases}, \tag{3.9}$$

see Fig. 3.1 for one such example.

The variable h, unlike variable β in SBX and PM, represents the step size, and it is not a requirement that p_1 has to be partially less than p_2, i.e. $p_1 \le_p p_2$. In fact, the *pdf* is endowed to adjust its domain. It does not change its shape except for the case when either $h_{\max} > U_i$ or $h_{\max} < L_i$, $h_{\max}$ is set equal to h^*.

The function $\mathcal{P}(h)$ determines the distance h from the parent to the offspring based on the area under the function. The total area under $\mathcal{P}(h)$ is given by the integral from 0

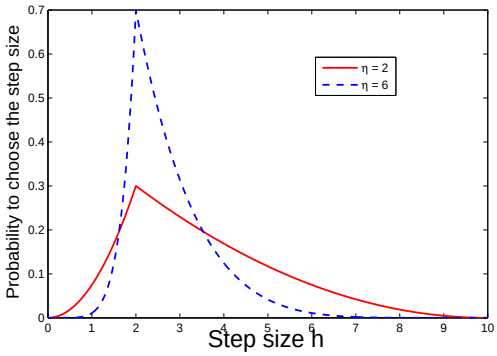


Figure 3.1: The image shows the *pdf* for two different values of $\eta = 2, 6$, the maximum step size, $h_{\max} = 10$ and the peak of the *pdf* is $h^* = 2$.

to $h_{\max}$, $\int_0^{h_{\max}} \mathcal{P}(h)\mathrm{d}h = 1$. The variable h is obtained with a random number $u \in [0, 1]$. We solve the integral and get the value of h.

$$h = \begin{cases} (h_{\max} u (h^*)^{\eta})^{\frac{1}{\eta+1}} & u \in \left[0, \frac{h^*}{h_{\max}}\right] \\ h_{\max} - \left(\left(1 - \hat{u} - \frac{h^*}{h_{\max}}\right)\left(h_{\max}\left(h_{\max} - h^*\right)^{\eta}\right)\right)^{\frac{1}{\eta+1}} & u \in \left(\frac{h^*}{h_{\max}}, 1\right] \\ & \text{and } \hat{u} = u - \frac{h^*}{h_{\max}} \end{cases} \tag{3.10}$$

Alg. 3.1 describes the procedure to obtain the step size in a probabilistic sense.

Algorithm 3.1 h = GenerateStepSize(h^*,$h_{\max}$)

1: **if** $h_{\max} \leq h^*$ **then**
2: $\quad h^* = h_{\max}$ **return**
3: **end if**
4: choose $u \in [0, 1]$ uniformly at random
5: compute h as in Eq. (3.10)

3.2 The Variation Operators

In the following, we define two variation operators which are based on the ideas presented in the previous subsection.

3.2.1 Shifted polynomial mutation

The mutation operator we present here, Shifted Polynomial Mutation (SPM), is a variant of the widely used polynomial mutation (PM, see [14]). We adapt this to our method with one exception: the peak of the probability distribution is as in Eq. (3.3). As discussed above, this is not desirable when using archivers that are based on ϵ-dominance, since by this the probability is relatively high that the offspring $\tilde{o}$ gets discarded from the archive since it is ϵ-dominated by the archive entry o. Instead of using the probability distribution of PM, we suggest to shift this peak such that the highest probability is to choose $\tilde{o}$ such that the difference to o in objective space is 2ϵ as discussed above. To be more precise, given an element $o \in Q$ the new offspring $\tilde{o}$ is generated as follows:

Algorithm 3.2 $\tilde{o} =$ ShiftedPolynomialMutation(o)

1: choose $\nu \in \mathbb{R}^n$, $\|\nu\|_\infty = 1$, uniformly at random
2: compute h^* and $h_{\max}$, Eq. (3.6) and (3.2)
3: $h =$ GenerateStepSize (h^*,$h_{\max}$)
4: $\tilde{o} = o + h\nu$

Example 4. To understand the SPM operator, of the discussion made above, we first apply the SPM with the next example, the function $F : \mathbb{R}^2 \to \mathbb{R}^2$, [46].

$$F(x) = \begin{pmatrix} (x_1 - 1)^4 + (x_2 - 1)^4 \\ (x_1 + 1)^2 + (x_2 + 1)^2 \end{pmatrix} \tag{3.11}$$

The Pareto set of function 3.11 is given by

$$P_Q = \left\{ \begin{pmatrix} x \\ x \end{pmatrix} : x \in [-1, 1] \right\}$$

Fig. 3.2 shows 100 mutations to the offspring $o = (-0.8, 0.8)$ with $\eta = 4$ and $h^* = 3$ using Alg. 3.2.

3.2.2 Biased intermediate crossover

The next operator we propose here, is called Biased Intermediate Crossover (BIC), which is a modification of the Intermediate Crossover (IC, see [47]). In fact, it is the extension of the SPM for the crossover scenario: given two parent solutions p_1 and p_2, a possible crossover of the two points is to select intermediate candidates on the connecting line $l = p_1 + \mathbb{R}(p_2 - p_1)$. However, also a search in the opposite direction could be interesting since the Pareto set forms at least locally a $(k-1)$-manifold. To be more precise, the BIC

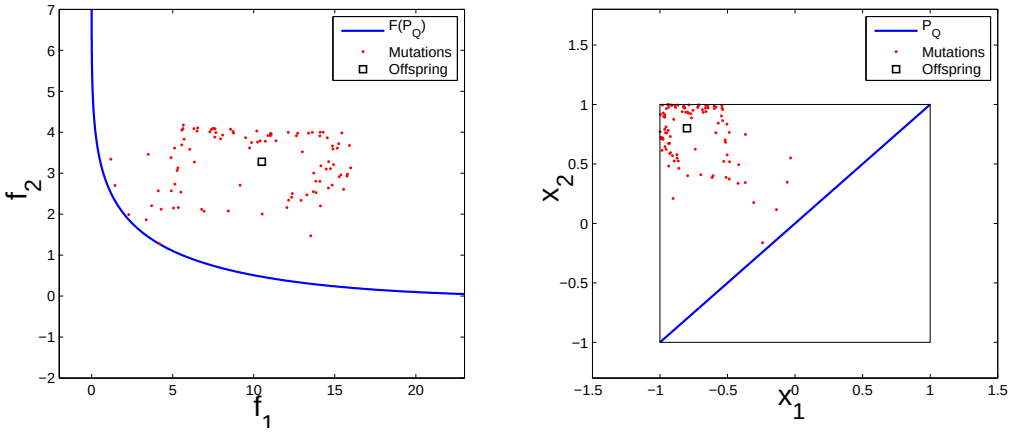


Figure 3.2: The mutations are marked with dots, offspring with □, the Pareto front and set with a solid line, 100 mutations in image space (left), mutations in parameter space (right).

works as follows: given p_1 and p_2, two offspring are generated on l from each p_j toward the other solution, and another two offspring in the opposite direction. In all cases, the search direction is $\nu = \pm(p_2 - p_1)$, and the step size control can be handled as in SPM since in all four cases a point and a direction is at hand. The procedure is summarized in Alg. 3.3. To avoid confusion, we denote by $h^*(p, \nu)$ the step size (3.6) for given p and ν. The following images show the latter comments. The offspring were generated by Alg. 3.3 using the same function that in the SPM.

Algorithm 3.3 $\{o_1, o_2, o_3, o_4\}$ = BiasedIntermediateCrossover(p_1,p_2)

1: $\nu_1 = p_2 - p_1$
2: $\nu_2 = -\nu_1$
3: $h_{1,1}$ = GenerateStepSize($h^*(p_1, \nu_1)$,$h_{\max}(p_1, \nu_1)$)
4: $o_1 = p_1 + h_{1,1}\nu_1$
5: $h_{1,2}$ = GenerateStepSize($h^*(p_1, \nu_2)$,$h_{\max}(p_1, \nu_2)$)
6: $o_2 = p_1 + h_{1,2}\nu_2$
7: $h_{2,1}$ = GenerateStepSize($h^*(p_2, \nu_1)$,$h_{\max}(p_2, \nu_1)$)
8: $o_3 = p_2 + h_{2,1}\nu_1$
9: $h_{2,2}$ = GenerateStepSize($h^*(p_2, \nu_2)$,$h_{\max}(p_2, \nu_2)$)
10: $o_4 = p_2 + h_{2,2}\nu_2$

The method is in particular effective when the parent solutions are already 'near' to

the Pareto set, see Fig. 3.3, or both parents are non dominated to each other, see Fig. 3.4. In the particular case in which one parent dominates the other one while, is a Pareto point, the offspring will be dominated, and four function calls will be lost, see Fig. 3.5.

Example 5. Fig.s 3.3, 3.4, 3.5 show examples of the BIC operator using the Function 3.11.

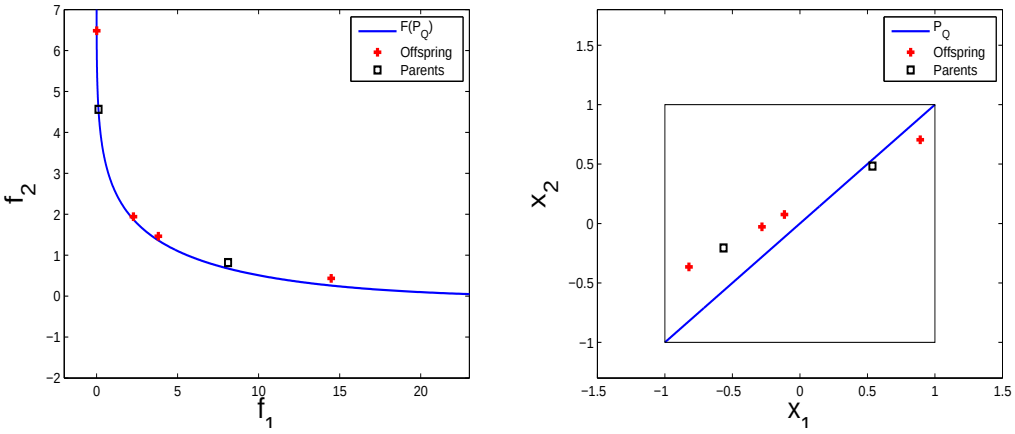


Figure 3.3: The offspring are 'near' Pareto front (left). Both parents are 'near' the Pareto set (right)

In the last case, one parent dominates the other one, which is not allowed for MOEAs based on the archive model. These kinds of algorithms only keep the nondominated individuals. These archives are based on ϵ-dominance and guarantee an upper bound of the archiver size as well as convergence (see Schütze et al. [45, 46]).

3.3 Two Archive Based MOEAs

Here, we propose two archive-based MOEAs which incorporate the elements discussed above. The first algorithm, MOEA1, is a straightforward application of the novel variation operators within an archive-based MOEA. Results have shown that this algorithm is well suited for the treatment of low-dimensional MOPs. This, however, has certain limitations when dealing with higher dimensional models. For the treatment of the latter, we propose in the sequel, ELMA (Evolutionary Lipschitz Multi-Objective Algorithm) which is a more sophisticated version of MOEA1 and also is advantageous over its predecessor on higher dimensional models.

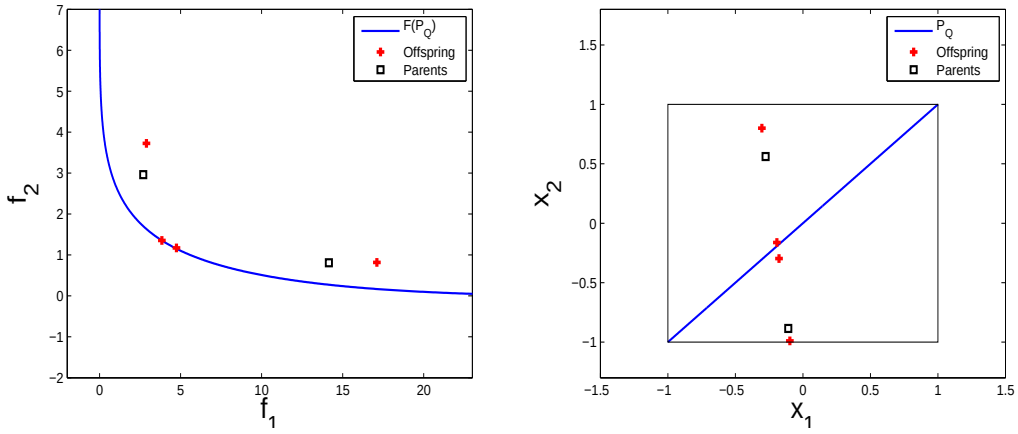


Figure 3.4: Both parents are non dominated to each other, they acquire two very good approximations in image space (left) and parameter space (right).

3.3.1 A new proposed MOEA

In the following, we propose our first archive based MOEA which is a classical combination of generation operators (crossover followed by mutation) and an external archive. For the archive, we have chosen to take $ArchiveUpdateTight2$ since we have observed that by this, a suitable approximation of the Pareto front can be obtained (see [46] or Section 2.5 in page 19 for a brief discussion of the archiver). A pseudo-code of the resulting algorithm (called here MOEA1) can be found in Alg. 3.4. Hereby, p_m denotes the probability for mutation, i.e., the probability that SPM is applied.

Now, we describe our choice of mutation p_m: unlike 'classical' genetic algorithms, we start with a relatively high value of p_m and reduce this value during the run of the algorithm. This is motivated as follows: as discussed in [3], if a candidate x is 'far away' from the Pareto set, there is, a nearly 50% chance to find a dominating (i.e., 'better') solution in the neigborhood of x with random local search (explained in [3] by the use of descent and ascent cones). Probability p_m will start with a high value, in spite of the fact that, how it is mentioned before, genetic algorithms normally use a low probability of mutation. Therefore, for a randomly chosen search direction, ν in SPM there is nearly 50% chance that this direction is a descent direction at x_0 (i.e. there exists an $h^* \in \mathbb{R}_+$ such that $F(x_0 + h^*\nu) <_p F(x_0)$). If, on the other side, a point x_0 is 'close' to the Pareto set, then the size of the descent cone is extremely narrow, resulting in a small probability for a randomly chosen vector to be a descent direction [3]. The two scenarios are depicted

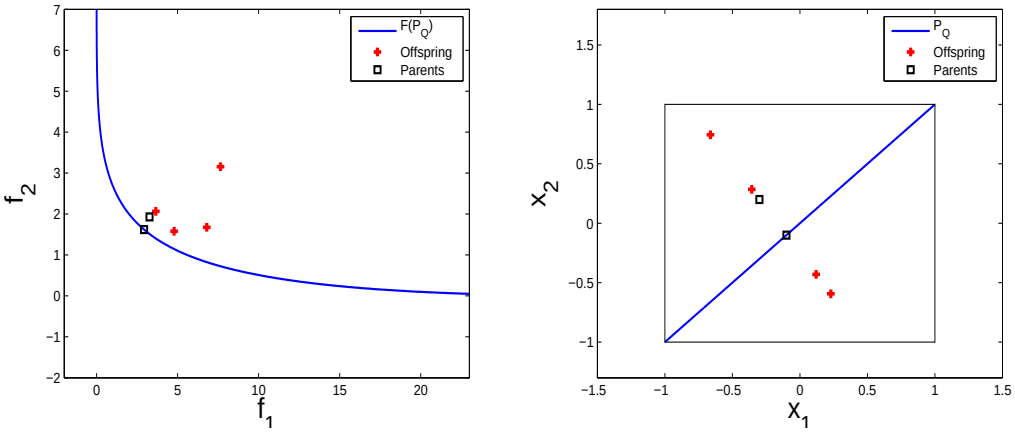


Figure 3.5: One parent is dominated while the other one is a Pareto point. The four offspring are dominated by the last one. Image space (left) and parameter space (right).

in Fig. 3.6 for a bi-objective problem. Hereby, {-, -} and {+, +} denote the descent and ascent cone respectively. The symbol {-, +} indicates that in this direction an improvement according to f_1 can be achieved while the values of f_2 will increase. Descent cones have already been used to develop evolutionary algorithms, see [42, 25].

One straightforward strategy (also used used for the following computations) is to handles the choice of mutation probability as follows: p_m is set as the number of accepted offspring divided by the number of offspring generated by the previous iteration, i.e.,

$$p_{m,i} = \frac{\#\{\text{accepted offpring in iteration step } i-1\}}{\#\{\text{generated offpring in iteration step } i-1\}}. \tag{3.12}$$

The following example describes the behavior of Alg. 3.4.

Example 6. A multi-objective optimization problem called PS1 [34]:

$$\begin{aligned} f_i &: \mathbb{R}^n \to \mathbb{R} \\ f_i(x) &= \sum_{j=1}^{n} (x_j - a_{i,j})^2 \end{aligned} \tag{PS1}$$

where $a_{i,j}$ denotes the j-th entry of $a_i \in \mathbb{R}^n$, and $i = 1, \ldots, k$. The Pareto set of the (PS1) is given by a simplex $S(a_1, \ldots, a_k)$[1]. A bi-objective problem, $f_{\{1,2\}} : \mathbb{R}^{10} \to \mathbb{R}$,

[1]**Definition** *Simplex*

Algorithm 3.4 MOEA1

Generate a random population P_0
$A_o = ArchiveUpdateTight2(\emptyset, P_0)$
$j = 0$
repeat
    Choose two points, $a_1, a_2 \in A_j$
    $\{o_1, o_2, o_3, o_4\}$ = BiasedIntermediateCrossover(a_1,a_2)
    for $k = 1, \ldots, 4$ **do**
        $u \in [0, 1]$ uniformly at random
        if $u \leq p_m$ **then**
            o_k = ShiftedPolynomialMutation(o_k)
        end if
    end for
    $A_{j+1} = ArchiveUpdateTight2(A_j, \{o_1, o_2, o_3, o_4\})$
    $j = j + 1$
until number of function calls have been reached

with $a_1 = (4, 9, 6, 9, 5, 7, 3, 9, 10, 1)$ and $a_2 = (6, 8, 9, 8, 7, 2, 5, 7, 3, 10)$. Fig. 3.7 shows a representative simulation of an application of the Alg. 3.4 with the (PS1) problem, after 15 simulations, we have chosen one of them.

The performance of Alg. 3.4 can neither be measured by a single function nor by a graphical evaluation. Numerical results will later define a proper metric. We observed from experiments carried out with Alg. 3.4 a good performance with low-dimensional functions [2, 24, 21, 28]. However, it has troubles with MOPs of higher dimensionality.

A set of test functions known as the ZDT (Zitzler-Deb-Thiele) [51] are frequently used to evaluate a new evolutionary algorithm. Alg. 3.4 has shown difficulties in this set of functions. The question arises, as to how can we improve the Alg. 3.4? Our proposal is to combine the generation process of ϵ-MOEA with our generation process in Alg. 3.4. A related work used components of ϵ-MOEA. They used NSGA-II at the explorative stage, implicity SBX is used in NSGA-II. The authors created an explotative stage based on the Nelder and Mead method [50]. Our generation process has to harnesse the power of

Let $a_1, a_2, \ldots, a_k \in \mathbb{R}^n$, then the set

$$S(a_1, a_2, \ldots, a_k) = \left\{ \sum_{i=1}^{k} \lambda_i a_i \mid \lambda_i \in [0, 1] \; \forall i \; \sum_{i=1}^{k} \lambda_i = 1 \right\}$$

is called the (k-1)-simplex of $a_1, a_2, \ldots, a_k$.

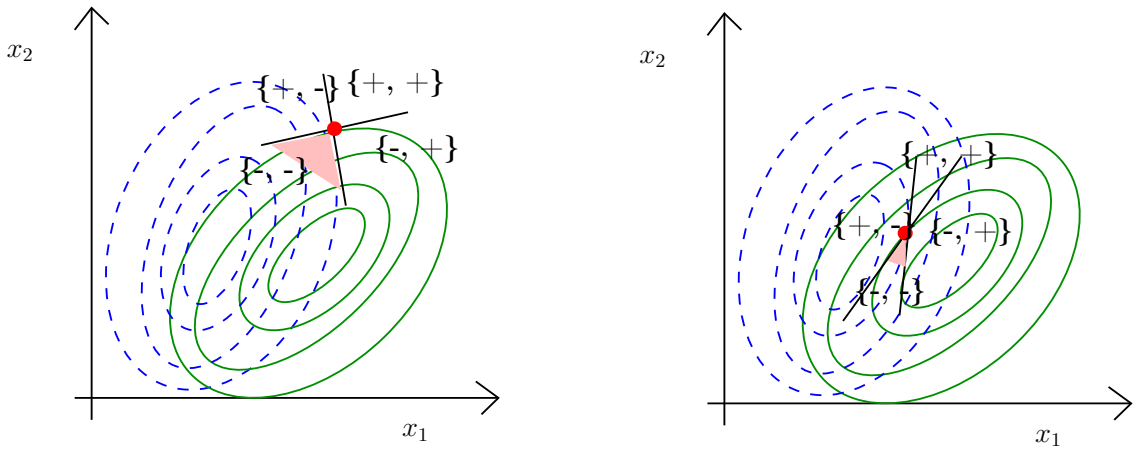


Figure 3.6: Descent cone (shaded) for an MOP with two parameters and two objectives (f_1 solid line and f_2 dot line), during the initial (left) and the final (right) stages of convergence. The descent cone shrinks to zero during the final stages of convergence.

exploration made by the SBX operator. Additionally, it is an extra operator to use when the archive contains only one element (i.e., the archive 'collapses').

3.3.2 Evolutionary lipschitz multi-objective algorithm

Here, we will develop an algorithm that adopts the generation process used in ϵ-MOEA together with our generation process. The proposed algorithm is a more sophisticated version of MOEA1 and is advantageous over its predecessor on higher dimensional models. We formulate the requeriments with the purpose of improvement of Alg. 3.4 as follows.

1. Economize the required number of function calls, since to calculate the Lipschitz constant, an extra function call is needed.

2. What to do, if the archive collapses? both operators, BIC and SBX, demand two points.

3. How to supress the outliers[2]?.

[2]**An outlier** is an observation that is numerically distant from the population, **http://en.wikipedia.org/wiki/Outlier**.

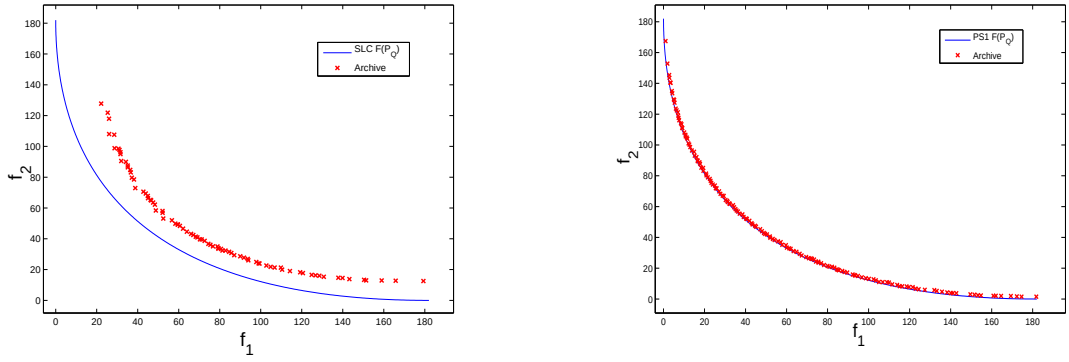


Figure 3.7: Pareto front of problem PS1 (solid line), and final archive after the application of Alg. 3.4 (crosses). Final archive after 3000 function calls with $p_m = 0.02$. The best archive, in terms of the Hausdorff distance, of the 15 simulations (left). Final archive using a decreasing sequence for p_m, starting with a probability $p_m = 0.4$ (right).

3.3.2.1 Economize the required number of function calls

Eq. (3.8) gives a possibility to calculate the Lipschitz constant but, it has the requisite of an extra function call. Whenever the Lipschitz constant has to be calculated for a new offspring o_{new}, there exists another element into the archive a, which is a distance less or equal to Δ (image space) from o_{new}. This will be used in substitution to the extra function call.

To calculate the Lipschitz constant of a new offspring, we follow the next procedure:

1. o_{new} **if** $\exists a \in A$ such that $d_\infty(F(a), F(o_{new})) \leq \Delta$ **then**

$$L_{o_{new}} \approx \frac{\|F(a) - F(o_{new})\|_\infty}{\|a - o_{new}\|_\infty}$$

2. **otherwise** Apply Eq. (3.8)

Moreover, the Lipschitz constant of an offspring will be obtained if an offspring was accepted by the archive. In this case, the generation and an update process has been done. Otherwise the constant will not be calculated.

3.3.2.2 What to do, if the archive collapses?

During the execution of an evolutionary algorithm, there is a possibility to create an offspring, which dominates all points in the archive. This leads to the event that the

archive contains a single solution. If the BIC operator is applied with the single point in the archive and a random point, possibly dominated, then two offspring will go in the direction of the random point. If the scenario is similar to that in Fig. 3.5, offspring will be rejected and four function calls will get lost.

To handle the archive collapse, we use a mechanism from ϵ-MOEA. We use the SBX operator with the single element in the archive and a random point, until the archive has more than one element. The random point 'emulates' the external population in ϵ-MOEA.

3.3.2.3 Supressing the outliers

The existence of outliers can decrease the performance of the algorithm. They have the same probability to be chosen like the rest of elements in the archive and apply crossover as a result. To illustrate this, we give a possible scenario.

Example 7. Let 1) ZDT1 in two dimensions, a bi-objective problem be given the Pareto set by $P_Q = [0,1] \times \{0\}$ and the Pareto front is the curve $F : [0,1] \to \mathbb{R}^2$, $F(P_Q) = \begin{pmatrix} x_1 \\ 1 - \sqrt{x_1} \end{pmatrix}$ (see [51]). 2) Further, an archive $A \subset \mathbb{R}^2$, $|A| = 20$ is as follows: nineteen points uniformly at random with $x_{i,1} \in [0.1, 1]$, $x_{i,2} = 0.01$ $i = 2, \ldots, 20$ and one outlier at $x_1 = (0.09, 0.2)$.
We choose the outlier as the first parent and the second parent is chosen at random. Using BIC, it is in general linearly, that outliers generate more outliers, see Fig. 3.8. if outliers arise, they will start to multiply in subsequent iterations.

Identification of outliers is not an easy task. Since the Pareto front is typically a priori not known, it is difficult to know whether a point out of a candidate set is an outlier or not. Further, if the number of objectives increases, the number of outliers typically grows, i.e., the problem gets harder with increasing number k of objectives. One possible remedy is as follows: One can 'clean' the current archive by re-inserting them into the archive, however, with a different order of consideration. The underlying idea of this is that outliers (that are hopefully not inserted first) are ϵ-dominated by other archive entries, and are thus discarded from the 'new' archive. Referring to the sketch of *ArchiveUpdateEps*1 in the previous chapter, (for details see [45, 46]) the insertion order is crucial. In case of a bi-objective problem, if we sort over one objective and insert in ascending order.

Given a determined archive A of size n, we will obtain a sorted archive, $\tilde{A}$:

$$\tilde{A} = \Big\{ f_i(a_1), \ldots, f_i(a_n) \Big\},\ i \in \{1, 2\} \text{ and } f_i(a_1) \leq f_i(a_2) \leq \ldots \leq f_i(a_n)$$

We proceed as follows to remove outliers

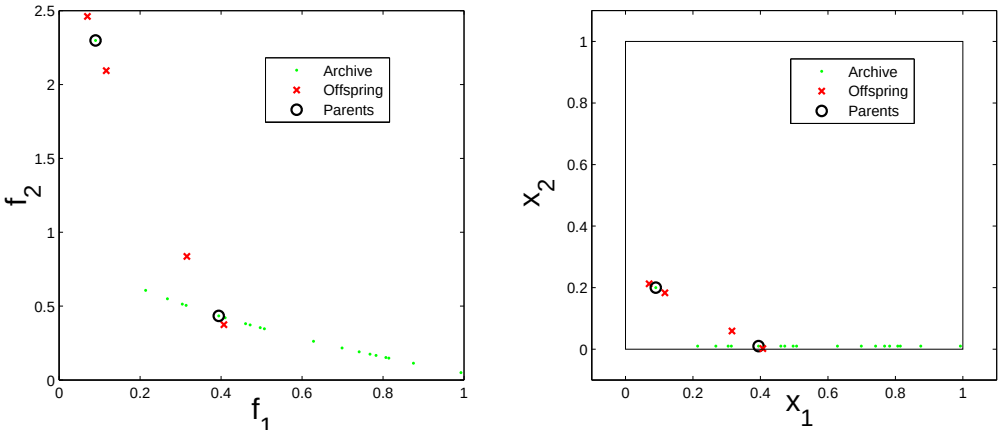


Figure 3.8: Example using ZDT1 function in $\mathbb{R}^2$. Elements in the archive are marked as dots, offspring are marked with a cross, selected parents with a black circle and the feasible domain (parameter space) is denoted as a rectangle. Applying the BIC operator, with a fixed step size $h^* = 0.01$, it generates two new outliers. They can begin to multiply in subsequent iterations (see image space left). The archive in parameter space shows that most of its elements are 'near' the Pareto set. It is desired to choose some of them except the outlier (right).

$$
\begin{aligned}
&\text{pvt} = \lfloor \tfrac{n}{2} \rfloor \\
&\tilde{A}_a = \{f_i(a_{\text{pvt}}), f_i(a_{\text{pvt}+1}), \ldots, f_i(a_n)\} \\
&\tilde{A}_b = \{f_i(a_{\text{pvt}-1}), f_i(a_{\text{pvt}-2}), \ldots, f_i(a_1)\} \\
&\tilde{A}_1 = ArchiveUpdateEps2(\tilde{A}_a, \emptyset); \\
&\tilde{A}_2 = ArchiveUpdateEps2(\tilde{A}_b, \emptyset); \\
&A = \tilde{A}_1 \cup \tilde{A}_2
\end{aligned}
$$

It is important to note that the order of insertion of elements with $ArchiveUpdateEps2$ has to be from a_{pvt} to n in case of $\tilde{A}_a$ and from $a_{\text{pvt}-1}$ to a_1 for $\tilde{A}_b$, otherwise the outliers will not be deleted from the archive. The above mechanism prevents the flat region of the Pareto front from appearing, due to the properties of $ArchiveUpdateEps2$. In case the number of objectives is more than two, the 'archive into archive' strategy cannot be applied as above. Here, we have chosen a random order of the two disjunct sub-archives $\tilde{A}_a$ and $\tilde{A}_b$ (however, a better choice of the sub-archives is subject to future research).

3.3.2.4 Integrating operators of ϵ-MOEA into ELMA

Extra information about SBX will be given in this subsection before we can state the final algorithm. The SBX operator always assumes that $p_1 \leq_p p_2$, something which in practice can not hold forever. The author of SBX gives as a solution to exchange whenever $p_{1,i} > p_{2,i}$.

Exchange of parents increases the capabilities of the search, in particular when they are not 'near' and, when using a step size much bigger than the value that could be found with Eq. (3.6). The external parameter η_c has an important role here. If η_m is equal to one, the step size would be bigger than we expected. It could also be a step size much bigger than the value that we could find with Eq. (3.6). A particular effect of this is that it could eliminate some outliers in some cases. See Fig. 3.9 for one such example with the ZDT1 problem.

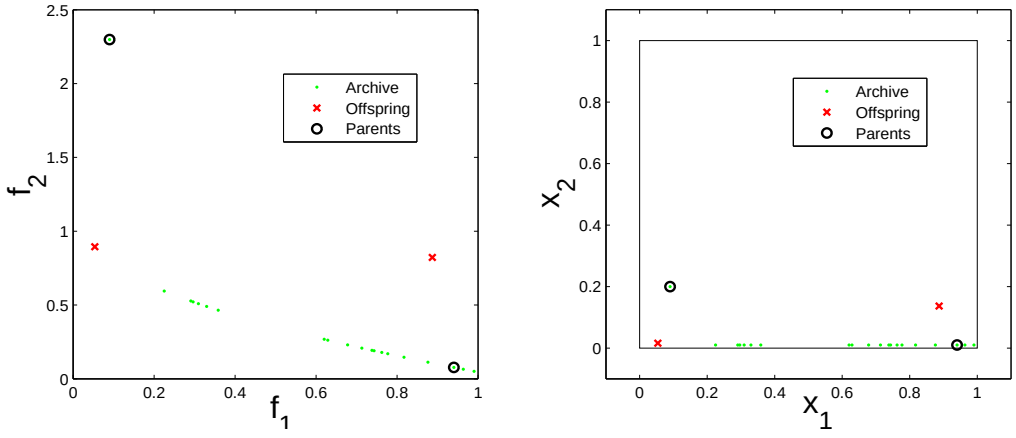


Figure 3.9: Example using the ZDT1 problem in $\mathbb{R}^2$. Elements in the archive are marked as dots, offspring are marked with a cross, and selected parents with a black circle. The application of the SBX operator can reduce the outliers. This operator will delete the outliers after *ArchiveUpdate* pocedure and a good approximation will be added (left). The first parent $p_1 = (0.09, 0.2)$, is bigger over the second component $p_{1,2} > p_{2,2}$ and, a swapping will make an improvement in addition to suppressing the outlier (right).

Now we are in the position to state the algorithm. Since the two novel operators are both of local nature, we will hybridize them with established operators leading to *Generator*1 (Alg. 3.6) and *Generator*2 (Alg. 3.7). Both generators consist of a crossover

strategy followed by mutation similar to the general scheme of an evolutionary algorithm [14]. For the archive we have chosen to take *ArchiveUpdateEps*2 [45], since it is able to maintain an ϵ-Pareto set of the given MOP [27]. The new MOEA *Evolutionary Lipschitz Multi-Objective Algorithm* (ELMA) is outlined in Alg. 3.5.

Algorithm 3.5 $\{A\}$ = ELMA()

Require: Initial probabilities p_m, every few iterations remove outliers r
Ensure: The final archive (A)
$P_0 \subset Q$ drawn at random
$A_0 = ArchiveUpdateEps2(P_0, \emptyset)$
$j = 0$
while number of function evaluations not reached **do**
    choose $p_1, p_2 \in A_j$ with $p_1 \neq p_2$ at random
    $O_1 = Generator1(p_1,p_2)$
    $O_2 = Generator2(p_1,p_2)$
    $A_{j+1} = ArchiveUpdateEps2(O_1 \cup O_2,\ A_j)$
    if $j \mod r$ **then**
        $A_{j+1} = ArchiveUpdateEps2(A_{j+1},\ \emptyset)$
    end if
    $j = j + 1$
end while

Algorithm 3.6 $O = Generator1(p_1, p_2)$

$\{o_1, .., o_4\}$ = BiasedIntermediateCrossover (p_1,p_2)
for $i = 1, .., 4$ **do**
    choose $u \in [0, 1]$ uniformly at random
    if $u \leq p_m$ **then**
        o_i = ShiftedPolynomialMutation(o_i)
    end if
end for
$O = \{o_1, .., o_4\}$

Algorithm 3.7 $O = Generator2(p_1, p_2)$

1: $\{o_1, o_2\}$ = SBX (p_1,p_2)
2: **for** $i = 1, 2$ **do**
3: choose $u \in [0, 1]$ uniformly at random
4: **if** $u \leq p_m$ **then**
5: o_i = PolynomialMutation(o_i)
6: **end if**
7: **end for**
8: $O = \{o_1, o_2\}$

Chapter 4

Δ_p: A New Indicator to Measure the Averaged Hausdorff Distance to the Pareto Front

Assessing the quality of an evolutionary algorithm commonly implies experimental comparisons between the resulting MOEA and other MOEAs or traditional algorithms. The convergence to an approximation solution set and maintenance of a good quality are two distinct and somewhat conflicting goals of multi-objective optimization. Therefore, no single metric can decide the performance of an algorithm.

This chapter begins with a study of two commonly used indicators, namely the Generational Distance (GD) and Inverted Generational Distance (IGD). Further on, we will define a new performance indicator, Δ_p, which can be viewed as an 'averaged Hausdorff distance' between the outcome set and the Pareto front which is composed of GD and IGD (to be more precise, slight modifications of them). Finally, we give a possible way to discretize a Pareto front by means of multi-objective continuation [46], which we call the L-Method.

4.1 Investigating the Indicators

In the following, we will give a description of indicators GD and IGD (see Section 1.6 in page 21, Eq. (2.7) and (2.8)) in relation to the properties of a metric.

Definition 8 ([36])**.** Suppose X is a set and d is a real function defined on the Cartesian product $X \times X$. Then d is called a metric on X if, and only if, for each $a, b, c \in X$,

a) (Positive Property) $d(a, b) \geq 0$ with equality if, and only if, $a = b$;

b) (Symmetric Property) $d(a, b) = d(b, a)$; and

c) (Triangle Inequality) $d(a, c) \leq d(a, b) + d(b, c)$.

d is called a *semi-metric*, if properties a) and b) are satisfied. If a semi-metric satisfies the relaxed triangle inequality

$$d(a, c) \leq \sigma(d(a, b) + d(b, c)), \forall\ a, b, c \in X \tag{4.1}$$

for a value $\sigma \geq 1$, d is called a *pseudo-metric*. In the following we will consider X as the set of compact subsets of the $\mathbb{R}^k$. A well-known metric on X is the Hausdorff distance d_H (see Section 1.5.3).

4.1.1 Generational distance

Given a candidate set $A = \{a_1, \ldots, a_N\}$ (in image space) and a Pareto front $F(P_Q) = \{y_1, \ldots, y_M\}$, the Generational Distance (GD, see [48]) can be written as follows:

$$\mathrm{GD}(A, F(P_Q)) = \frac{1}{|A|} \left(\sum_{i=1}^{|A|} \mathrm{dist}_2(a_i, F(P_Q))^p \right)^{\frac{1}{p}} \tag{GD}$$

Due to the non-negativity norms, GD is also non negative, i.e., it is $\mathrm{GD}(A, F(P_Q)) \geq 0$ for all finite sets A and $F(P_Q)$. However, it is equal to zero only when A is contained in $F(P_Q)$, i.e.,

$$GD(A, F(P_Q)) = 0 \iff A \subset F(P_Q) \tag{4.2}$$

Since A does not have to be equal to $F(P_Q)$, the positive property does not hold. Further, GD is not symmetric. As an example, let A be a proper subset of $F(P_Q)$. Then, it is $\mathrm{GD}(A, F(P_Q)) = 0$ and $\mathrm{GD}(F(P_Q), A) > 0$. Finally, GD does not satisfy the triangle inequality. Assume, for instance, $A = \{(4, 4)^T, (7, 4)^T\}$, $B = \{(1, 7)^T, (6, 5)^T\}$, and $C = \{(2, 10)^T, (4, 10)^T\}$. Then, it is $\mathrm{GD}(A, C) > \mathrm{GD}(A, B) + \mathrm{GD}(B, C)$ for $p = 1, 2$.

The normalization strategy in GD decreases the indicator value as the magnitude of A grows, for a bounded domain. It is worth noting that this effect becomes stronger with increasing p for dist_p. In the context of archive based MOEAs, it is advantageous from this point of view to 'fill' the archive with more solutions since typically larger sets yield better GD values. However, this leads to trouble for MOEAs which are based on archives that are not bounded by an a priori defined value (but rather indirectly, e.g., by the use of ϵ-dominance as in [27, 12, 45, 46]). A 'perfect' archiver (with respect to GD) is hence the one that accepts *all* (or at least as many as possible) candidate solutions. An effect

which is certainly not desired.

As an alternative version of GD that avoids the effect discussed above, we propose a modification of the indicator, by using the power mean[1] to average the distances $\text{dist}(a_i, F(P_Q))$, i.e.:

$$\text{GD}_p(X,Y) = \left(\frac{1}{|A|} \sum_{i=1}^{|A|} \text{dist}(a_i, F(P_Q))^p \right)^{1/p} = \frac{\|d_{AF}\|_p}{\sqrt[p]{|A|}} \tag{4.3}$$

We name the new indicator here GD_p (i.e., with the index p). The 'new' indicator does not have the unwanted characteristic as discussed above and seems hence to be more fair for a comparison of sets with different magnitudes. Nevertheless, the metric properties are the same as for GD.

Apparently, GD_p has a relation to dist, i.e.,

$$\text{GD}_\infty(A, F(P_Q)) = \text{dist}(A, F(P_Q)) \tag{4.4}$$

That it, for $p < \infty$, GD_p can be viewed as an 'averaged' version of dist.

To illustrate this, we consider a bi-objective problem, ZDT1, which has as its domain $Q = [0,1]^{30}$. As the number of randomly chosen points N that form the archive A within Q increases, the GD value goes down while GD_p remains 'stable'. Table 4.1 shows values obtained with GD and GD_p with different norms and sizes of N. Even though, there are nondominated points GD goes down as N increases.

4.1.2 Inverted generational distance

Analog to the GD indicator, we will propose the same modification for IGD_p, which has the same metric properties as GD_p. The new indicator is related to many distance measurements used in the EMO literature.

The IGD indicator as proposed in [5] can be written as follows:

$$\text{IGD}(A, F(P_Q)) = \frac{1}{|F(P_Q)|} \left(\sum_{i=1}^{|F(P_Q)|} \text{dist}_2(y_i, A)^p \right)^{\frac{1}{p}}, \tag{IGD}$$

[1] Also known as generalized mean or Hölder mean.

Table 4.1: Numerical values of GD and GD_p indicators for different norms and archive sizes.

p	Indicator	N = 100	N = 1000	N = 10,000	N = 40,000
$p = 1$	GD	3.007298	3.007833	2.998742	3.003585
	GD_1	3.007298	3.007833	2.998742	3.003585
$p = 2$	GD	0.308157	0.097281	0.030647	0.015353
	GD_2	3.081573	3.076308	3.064666	3.070502
$p = 5$	GD	0.083783	0.013054	0.002059	0.000681
	GD_5	3.335445	3.278910	3.262850	3.270808
$p = \infty$	GD	0.055364	0.005085	0.000541	0.000145
	GD_∞	5.536430	5.085181	5.413225	5.807405

where A is a candidate set $A = \{a_1, \ldots, a_N\}$ (in image space) and $F(P_Q)$ is discrete Pareto front, $F(P_Q) = \{y_1, \ldots, y_M\}$.

Due to the analogy of GD and IGD in principle the same argumentation can be applied to justify a modification of the operator. In the context of multi-objective optimization, a (suitable) discretization $F(P_Q)$ of the Pareto front has to be chosen. Analog to the discussion for GD, the IGD value gets better when choosing a finer discretization of the Pareto front: assume we are given an archive A, and two discretizations $F_1(P_Q)$ and $F_2(P_Q)$ of the Pareto front, where $F_2(P_Q)$ is finer than $F_1(P_Q)$ (i.e., better in the Hausdorff sense and contains more elements). Then, it is $IGD(F_2(P_Q), F(A)) < IGD(F_1(P_Q), F(A))$. Though this problem can in principle be avoided by fixing a discretization of the Pareto front is also an unwanted effect. To void the effect discussed above, we propose a modification of the indicator to average the distances $\mathrm{dist}(y_i, A)$, i.e.:

$$\mathrm{IGD}_p(A, F(P_Q)) = \left(\frac{1}{|F(P_Q)|} \sum_{i=1}^{|F(P_Q)|} \mathrm{dist}(y_i, A)^p \right)^{1/p} = \frac{\|d_{FA}\|_p}{\sqrt[p]{|F(P_Q)|}} \tag{4.5}$$

Apparently, IGD_p has a relation to dist, i.e.,

$$\mathrm{IGD}_\infty(A, F(P_Q)) = \mathrm{dist}(F(P_Q), A) \tag{4.6}$$

Since it is

$$IGD(A, F(P_Q)) = GD(F(P_Q), A) \tag{4.7}$$

the properties of GD apply for IGD. In particular, it is

$$IGD(A, F(P_Q)) = 0 \iff F(P_Q) \subset A \tag{4.8}$$

A relation to other distances will be given after the following definition.

Definition 9 ([39])**.** Let d be a metric, $\delta > 0$, and $D \subset Z$ be a discrete set. D is called a d_δ representation of Z if for any $z \in Z$ there exists an element $y \in D$ such that $d(z, y) \leq \delta$.

The next proposition gives the relation of IGD_∞ to the measurements based on ϵ-dominance. Hereby, we use IGD^q_∞ to indicate that the q-norm is used for $\mathrm{dist}_q(a, B)$ (see Section 1.6).

Proposition 1. Let $A \subset \mathbb{R}^n$ be given.

a) A is a c-approximate Pareto set of the MOP, where $c = \mathrm{IGD}^q_\infty(F(A), F(P_Q))$.

b) $I_{\epsilon^+}(A, P_Q) = \mathrm{IGD}^\infty_\infty(F(A), F(P_Q))$

Proof. Ad a): It is

$$IGD^q_\infty(F(A), F(P_Q)) = \max_{p \in P_Q} \min_{a \in A} \|F(p) - F(a)\|_q. \tag{4.9}$$

That is, for all $p \in P_Q$ there exists an $a \in A$ such that $\|F(p) - F(a)\|_q \leq c$. Since in particular $|f_i(p) - f_i(a)| \leq c$ for all $i = 1, \ldots, k$ it is also $a \prec_{1c} p$, and the claim follows.
Ad b): It is

$$\begin{aligned} I_{\epsilon^+}(A, P_Q) &= \min_{\epsilon \in \mathbb{R}_+} \{\forall p \in P_Q \exists a \in A \ : \ F(a) - 1\epsilon \leq_p F(p)\} \\ IGD^\infty_\infty(F(A), F(P_Q)) &= \max_{p \in P_Q} \min_{a \in A} \|F(p) - F(a)\|_\infty =: c, \end{aligned} \tag{4.10}$$

and there exist $\bar{p} \in P_Q$, $\bar{a} \in A$ such that

$$\|F(\bar{p}) - F(\bar{a})\|_\infty = c. \tag{4.11}$$

That is, for all $p \in P_Q$ there exists an $a \in A$ such that $\|F(p) - F(a)\|_\infty \leq c$. Since also here $|f_i(p) - f_i(a)| \leq c$ for all $i = 1, \ldots, k$ it follows that $F(a) - 1\epsilon \leq_p F(p)$. This together with (4.11) completes the proof. □

For an example of Proposition 1(a) see Section 1.6, Fig. 2.11. Finally, to understand the relation with Definition 9, let $A \in \mathbb{R}^n$, then its image $F(A)$, is a d_δ representation of the Pareto front iff

$$\begin{aligned} \mathrm{GD}_\infty(F(A), F(P_Q)) &= 0 \quad \text{and} \\ \mathrm{IGD}_\infty(F(A), F(P_Q)) &\leq \delta, \end{aligned} \tag{4.12}$$

where dist_2 is the metric induced by the 2-norm (or in more general terms, the q-norm)

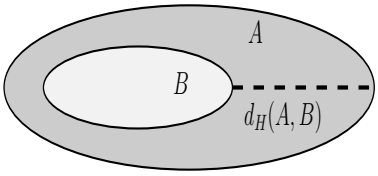


Figure 4.1: If B is a proper subset of A, then it follows that $\text{dist}(B, A) = 0$ and $\text{dist}(A, B) > 0$, and hence $d_H(A, B) = \text{dist}(A, B) > 0$.

4.2 A New Performance Indicator

This section describes a 'new' indicator which is composed by the two indicators discussed above (GD_p and IGD_p). We will also see the relationship of this 'new' indicator with the Hausdorff metric d_H. We begin with a description of the Hausdorff metric, which is a widely used tool to measure the distance between different objects in several research fields.

4.2.1 The Hausdorff metric

The idea behind d_H is to define a metric on a collection of subsets of a metric space in terms a given metric dist_q. The metric extends to all nonempty closed bounded subsets of an arbitrary metric space X. Using Definition 4, Section 2.5.3; Given two sets $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_m\}$ (possible $n = m$) the Hausdorff distance is defined as:

$$d_H(A, B) = \max\{\text{dist}(A, B), \text{dist}(B, A)\} \tag{HD}$$

The metric identifies the point $a \in A$ that is farthest from any point of B, and measures the distance from a to its nearest neighbor in B. Thus the Hausdorff distance $d_H(A, B)$, measures the degree of mismatch between two sets. Fig. 4.1 illustrated d_H, the set B is contained in the set A, and the biggest distance is from an $a \in A$ to set B.

4.2.2 A 'new' proposal indicator

The new indicator is motivated by the Hausdorff metric and the relation of GD_p and IGD_p with dist. We define the new indicator Δ_p as follows.

Definition 10. Let $A = \{a_1, \ldots, a_N\} \subset \mathbb{R}^k$ and $B = \{b_1, \ldots, b_M\} \subset \mathbb{R}^k$ be finite and

nonempty sets. Then we define $\Delta_p(A, B)$ by

$$\begin{aligned}\Delta_p(A,B) &= \max\{GD_p(A,B), IGD_p(A,B)\} \\ &= \max\left\{ \left(\frac{1}{N}\sum_{i=1}^{N} \mathrm{dist}(a_i,B)^p\right)^{1/p}, \left(\frac{1}{M}\sum_{i=1}^{M} \mathrm{dist}(b_i,A)^p\right)^{1/p} \right\}\end{aligned} \tag{4.13}$$

The metric properties of the new indicator Δ_p are stronger than the one of GD_p and IGD_p, due to the combination of the two indicators. In particular, when $p = \infty$ this becomes a metric, since it holds $\Delta_\infty(A, B) = d_H(A, B)$, see Eq.s (4.4) and (4.6). On the other hand, for all values of $p < \infty$ and a bounded archive size, Δ_p defines a semi-metric and, this can be viewed as an 'averaged Hausdorff distance'. The indicator Δ_p does not satisfy the triangle inequality for $p < \infty$ which is caused by the averaging of the distances. As an example let $A = \{(5, 8)^T, (9, 9)^T\}$, $B = \{(2, 8)^T, (6, 9)^T\}$ and $C = \{(1, 9)^T, (2, 5)^T\}$. Then $\Delta_\infty(A, C) = \Delta_\infty(A, B) + \Delta_\infty(B, C)$ and $\Delta_1(A, C) > \Delta_1(A, B) + \Delta_1(B, C)$.

Proposition 2. Δ_p is a semi-metric for $1 \leq p < \infty$ and a metric for $p = \infty$.

Proof. The positive property follows directly by the non-negativity of the norm and the Eq.s (4.2) and (4.8). The symmetry follows by the construction of Δ_p. Hence, Δ_p is a semi-norm.
Let $p = \infty$, then

$$\begin{aligned}\Delta_\infty(X,Y) &= \max\left(\max_{i=1,\ldots,|X|}(\mathrm{dist}(x_i,Y)), \max_{i=1,\ldots,|Y|}(\mathrm{dist}(y_i,X))\right) \\ &= \max(\mathrm{dist}(X,Y), \mathrm{dist}(Y,X)) = d_H(X,Y),\end{aligned} \tag{4.14}$$

i.e., for $p = \infty$ the indicator Δ_p coincides with the Hausdorff distance. □

Since $F(P_Q)$ is given, we can assume that we are given a finite approximation $Y \subset \mathbb{R}^k$ of the Pareto front with $d_H(Y, F(P_Q)) \leq \delta$ (i.e., Y contains no outliers, see below for one possible heuristic for the generation of Y for bi-objective problems). The natural question that arises in this context is the resulting discretization error that has to be considered when comparing different indicator values. Here, we define the approximation error in a straightforward way: given an archive A, the Pareto front $F(P_Q)$ and its discretization Y, we define the error e.g. for GD_p as $|GD_p(F(A), F(P_Q)) - GD_p(F(A), Y))|$ (analog for the other indicators).
The following result shows that the discretization error for the three indicators under investigation is equal to the approximation quality of Y.

Proposition 3. Let $A \subset \mathbb{R}^n$ be finite, $F(P_Q)$ and let $Y \subset \mathbb{R}^k$ be finite such that $d_H(F(P_Q), Y) \leq \delta$. Then

a) $|GD_p(F(A), F(P_Q)) - GD_p(F(A), Y))| \leq \delta$

b) $|IGD_p(F(A), F(P_Q)) - IGD_p(F(A), Y))| \leq \delta$

c) $|\Delta_p(F(A), F(P_Q)) - \Delta_p(F(A), Y))| \leq \delta$

Proof. See [44]. □

Example 8. Assume a hypothetical discrete Pareto front is given by P where $p_i = ((i-1) \cdot 0.1, 1-(i-1) \cdot 0.1)^T$, $i = 1, \ldots, 11$. Further, we are given two approximations of P: A is identical to P except for the first element $a_1 = (0.001, 10)^T$ (an 'outlier'), i.e., $A = \{a_1, p_2, \ldots, p_{11}\}$. B is a translation of P defined by $b_i = p_i + (2, 2)^T$, $i = 1, \ldots, 11$, Fig. 4.2 shows a plot of these three sets. Now, we apply the indicator $\Delta_p(A, P)$ and $\Delta_p(B, P)$, Table 4.2 shows the numerical values of Δ_p for different values of p. A is a 'better' approximation, and B is 'better' for $p \geq 3$.

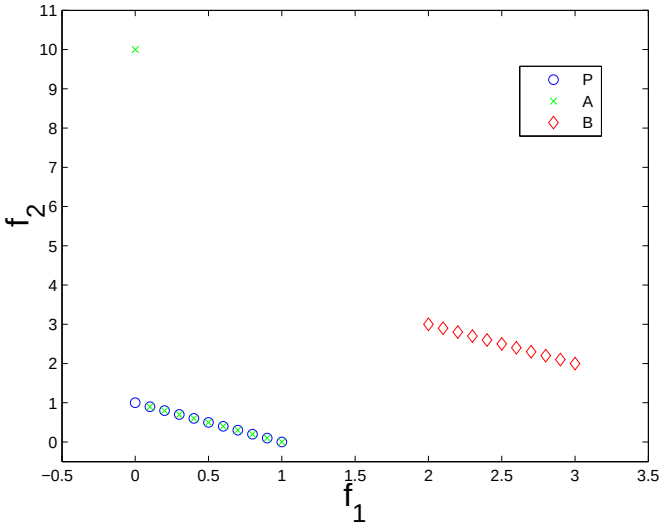


Figure 4.2: Hypothetical example for a Pareto front (P) and two different approximations A and B.

The choice of the p-norm is the way to handle the 'outlier trade off'. If p is smaller, the outlier does not have a big influence. On the other hand, if p is increased, then the largest distances in $GD(A, B)$ dominanted, and hence, outliers influence the value of $\Delta_p(A, B)$.

Table 4.2: Values of $\Delta_p(A, P)$ and $\Delta_p(B, P)$. The higher the value of p, the more outliers are penalized by Δ_p.

	$p = 1$	$p = 2$	$p = 3$	$p = 5$	$p = 10$	$p = \infty$
$\Delta_p(A, P)$	0.818	2.714	4.047	5.571	7.080	9.000
$\Delta_p(B, P)$	2.828	2.828	2.828	2.828	2.828	2.828

The worst case is when $p = \infty$ and, the farthest distances are considered, but in turn Δ_p defines a metric. Table 4.3 shows the percentage of the triangle inequality violations for different values of p for a sequence of randomly chosen sets A, B and C with different magnitudes. The larger p, the fewer triangle inequality violations are observed, and hence, the 'nearer' Δ_p is to a metric.

Table 4.3: Percentage of the triangle violations for different values of p. Here 100,000 different sets A, B, and C with magnitude N = 2, 4, 6, 10 and 100 have been chosen, and each entry of each set has been chosen randomly from $[0, 10]^2$ (The non vanishing values for $p = \infty$ are due to round-off errors).

	$p = 1$	$p = 2$	$p = 5$	$p = 10$	$p = 20$	$p = \infty$
N = 2	0.541	0.15	0.026	0.008	0.005	0.006
N = 4	0.249	0.06	0.019	0.009	0.005	0.002
N = 6	0.105	0.033	0.008	0.003	0.001	0
N = 10	0.02	0.002	0.004	0.001	0	0
N = 100	0	0	0	0	0	0

4.3 Discretization of the Pareto Front

In almost all benchmark functions provide the P_Q or $F(P_Q)$ as an analitical expression and in an 'easy' form. Assume we are given either P_Q or $F(P_Q)$, the question is to get a 'suitable' discretization $P = \{p_1, \ldots, p_n\}$, $p_i \in P_Q$, such that $Y = F(P)$ serves as a Pareto front approximation with $d_H(Y, F(P_Q)) \leq \delta$, where $\delta \in \mathbb{R}_+$ is given a priori. The present section provides a method to discretize the Pareto front Y for bi-objective problems with the desired Hausdorff distance. We support the method with elements of step size control for multi-objective continuation [46].

We assume that all models are continuous, differentiable and connected. By the connectedness of $F(P_Q)$ and the characteristic of Pareto fronts there exists a curve $\gamma : [a, b] \to \mathbb{R}^2$ such that $\gamma(t)$ is equal to $F(P_Q)$. Consider a Pareto front of the form

$$\begin{aligned} \gamma &: [a, b] \to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} \mathrm{t} \\ \mathrm{g(t)} \end{pmatrix}, \end{aligned} \tag{4.15}$$

where $g : [a, b] \to \mathbb{R}$ (i.e., the first objective is given by t as in the Okabe or ZDT benchmark models which are widely used in the EMO literature). The main problem is to estimate the distance

$$\|\gamma(t_{i+1}) - \gamma(t_i)\|_\infty \approx \delta \tag{4.16}$$

of two consecutive elements. The step size $h = t_{i+1} - t_i$, is calculated in the same way as in Eq. (3.6). The Lipschitz constant L_t, can be calculated using finite series as in Eq. (3.8) or, if the gradient is available, then it is used. Alg. 4.1 shows the procedure to obtain a discretization given the Pareto front $\gamma(t)$.

Algorithm 4.1 $Y = L$-Method1($[a, b]$, δ)

1: $t = a$
2: $Y = \emptyset$
3: **while** $t \leq b$ **do**
4: $\quad Y = Y \cup \gamma(t)$
5: $\quad L_t = \|\gamma'(t)\|_\infty$
6: $\quad h = \frac{\delta}{L_t}$
7: $\quad t = t + h$
8: **end while**

To obtain a discretization Y given a Pareto set by the simplex $S(p_a, p_b)$, p_a and $p_b \in \mathbb{R}^n$ (i.e. the Pareto set is a line), in Alg. 4.1, line 7 is changed in the form $t = t + h\nu$ with $\nu = p_b - p_a$. Alg. 4.2 shows a pseudo-code to obtain a discretization Y, when $P_Q = S(p_a, p_b)$.

The difference of the two previous algorithms is that, L-Method1 requires the equation of the Pareto front. If the Pareto set forms a simplex $S(p_a, p_b)$, then L-Method2 is used. In the following, we assume P_Q is not a line, then the step size h will be the length of the curve

$$P_Q : [a, b] \to \mathbb{R}^n. \tag{4.17}$$

To move along the curve $P_Q(t)$, a distance h from t_i, the final point t_{i+1} has to be known. The final point is obtained by the integral $\int_{t_i}^{t_{i+1}} \|P'_Q(t)\|_\infty \mathrm{d}t = h$. If we obtain $\|P'_Q(t)\|_\infty$

Algorithm 4.2 $Y = L$-Method2(p_a, p_b, δ)

```
t = p_a
Y = ∅
ν = p_b − p_a
while h > 0 using Eq. (3.2) do
    Y = Y ∪ F(t)
    L_t = ‖F'(t)‖_∞
    h = δ / L_t
    t = t + hν
end while
```

$= G(t)$ separately, then the next point t_{i+1} is obtained as follows: $t_{i+1} = \frac{h}{G(t)} + t_i$. Alg. 4.3 shows as discretizing the Pareto front when the Pareto set is given.

Algorithm 4.3 $Y = L$-Method3($[a, b]$, δ)

```
t = a
s = P_Q(t)
Y = ∅
while t ≤ b do
    Y = Y ∪ F(s)
    L_t = ‖F'(s)‖_∞
    h = δ / L_t
    G = ‖P'_Q(t)‖_∞
    t = h / G + t
    s = P_Q(t)
end while
```

Chapter 5

Numerical Results

Here, we attempt to demonstrate the usefulness of the novel indicator as well as the benefit of the new algorithm. First, we show examples of discretizations of the Pareto front as discussed in Section 4.3. Next, we intend to show empirically that modern MOEAs indeed comply (to a certain extent) with Δ_p. For this, we have chosen to apply NSGA-II on a benchmark model for the evaluation of ArchiveUpdateTight$i$, $i = 1, 2$ and their relation with Δ_p. Finally we evaluate the new algorithm proposed with the new indicator Δ_p plus the GD_p and IGD_p indicators.

5.1 Generating Discretizations of the Pareto Front

We address the problem of generating a 'suitable' discretization of $F(P_Q)$. We will take advantage of the present section and discretize functions that are going to be used to evaluate ELMA. The first five models (ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6) were proposed by Deb et al. [51]. The functions called (Deb2) and (Deb3) were proposed in [13]. The function (Lis) was proposed in [28] and, finally the (Oka2) function was proposed in [35].

ZDT1 This problem has a convex and connected Pareto front.

$$\begin{aligned} f_1(x) =& x_1 \\ f_2(x) =& g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}}\right) \\ & \text{where :} \\ g(x) =& 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i \end{aligned} \tag{ZDT1}$$

with $n = 30$ and $x_i \in [0, 1]$ for $i = 1, \ldots, 30$. The Pareto set is given by

$$P_Q = [0, 1] \times \{0^{n-1}\}, \tag{ZDT1-P_Q}$$

and the Pareto front is given by

$$\begin{aligned} \gamma &: [0, 1] \to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - \sqrt{t} \end{pmatrix} \end{aligned} \tag{ZDT1-$F(P_Q)$}$$

Fig. 5.1 shows results for different values of δ, on application of Alg. 4.1.

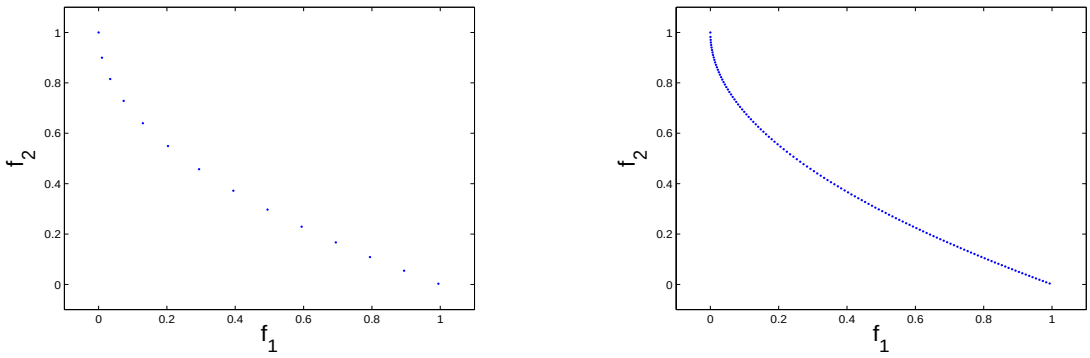


Figure 5.1: Discretizations of the Pareto front of model ZDT1 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

ZDT2 This problem has a nonconvex and connected Pareto front.

$$\begin{aligned} f_1(x) =& x_1 \\ f_2(x) =& g(x) \cdot \left(1 - \left(\frac{f_1(x)}{g(x)}\right)^2\right) \\ & \text{where :} \\ g(x) =& 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i \end{aligned} \tag{ZDT2}$$

with $n = 30$ and $x_i \in [0, 1]$ for $i = 1, \ldots, 30$. The Pareto set is given by

$$P_Q = [0, 1] \times \{0^{n-1}\}, \tag{ZDT2-P_Q}$$

and the Pareto front is given by

$$\begin{aligned} \gamma &: [0,1] \to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1-t^2 \end{pmatrix} \end{aligned} \qquad \text{(ZDT2-}F(P_Q)\text{)}$$

Fig. 5.2 shows results for different values of δ, on application of Alg. 4.1.

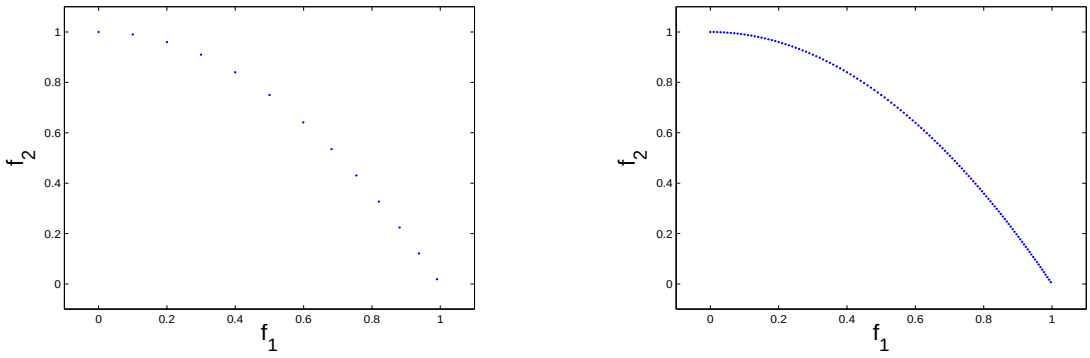


Figure 5.2: Discretizations of the Pareto front of model ZDT2 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

ZDT3 This problem has a finite number of disconnected Pareto fronts.

$$\begin{aligned} f_1(x) =& x_1 \\ f_2(x) =& g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}} - \frac{f_1(x)}{g(x)} \cdot \sin(10\pi f_1(x))\right) \\ & \text{where :} \\ g(x) =& 1 + \frac{9}{n-1} \cdot \sum_{i=2}^{n} x_i \end{aligned} \qquad \text{(ZDT3)}$$

with $n = 30$ and $x_i \in [0,1]$ for $i = 1, \ldots, 30$. The approximation (because not all points of $x_1 \in [0,1]$ are Pareto points) of the Pareto set is given by

$$P_Q \approx [0,1] \times \{0^{n-1}\} \ , \qquad \text{(ZDT3-}P_Q\text{)}$$

and the Pareto front is contained inside

$$\begin{aligned} \gamma &: [0,1] \to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - \sqrt{t} - t \cdot \sin(10\pi t) \end{pmatrix} \end{aligned} \qquad \text{(ZDT3-}F(P_Q)\text{)}$$

Fig. 5.3 shows results for different values of δ, on application of Alg. 4.1, note that after the application of the algorithm all the dominated points have to be removed.

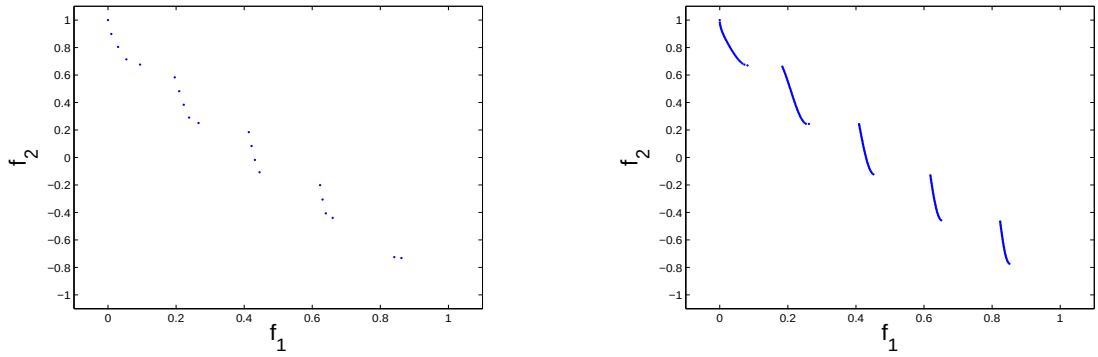


Figure 5.3: Discretizations of the Pareto front of model ZDT3 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

ZDT4 This problem has a convex Pareto front.

$$\begin{aligned} f_1(x) =& x_1 \\ f_2(x) =& g(x) \cdot \left(1 - \sqrt{\frac{f_1(x)}{g(x)}}\right) \\ &\text{where :} \\ g(x) =& 1 + 10 \cdot (n-1) + \sum_{i=2}^{n} \left(x_i^2 - 10\cos(4\pi x_i)\right) \end{aligned} \tag{ZDT4}$$

with $n = 10$ and $x_1 \in [0, 1]$ and $x_i \in [-5, 5]$ for $i = 2, \ldots, 10$. The Pareto set is given by

$$P_Q = [0, 1] \times \{0^{n-1}\} \ , \tag{ZDT4-P_Q}$$

and the Pareto front is given by

$$\begin{aligned} \gamma : [0, 1] &\to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - \sqrt{t} \end{pmatrix} \end{aligned} \tag{ZDT4-$F(P_Q)$}$$

Fig. 5.3 shows results for different values of δ, on application of Alg. 4.1.

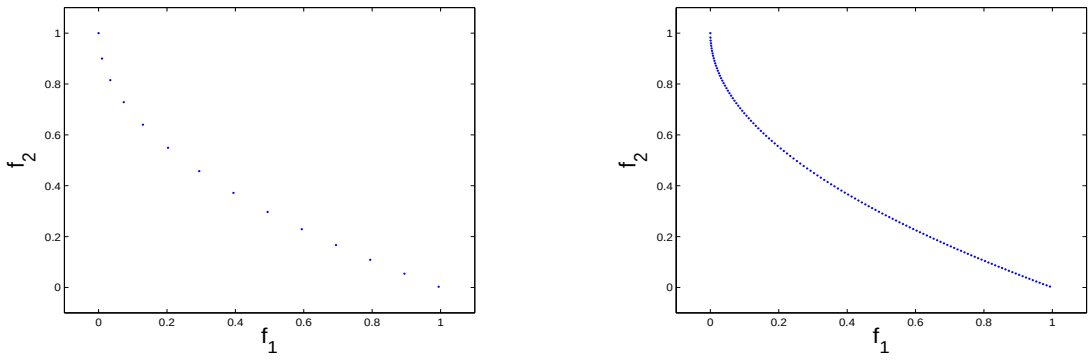


Figure 5.4: Discretizations of the Pareto front of model ZDT4 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

ZDT6 This problem has a nonconvex Pareto front.

$$\begin{aligned} f_1(x) =& 1 - \mathrm{e}^{-4x_1} \cdot \sin^6(6\pi x_1) \\ f_2(x) =& g(x) \cdot \left(1 - \left(\frac{f_1}{g(x)}\right)^2\right) \\ & \text{where :} \\ g(x) =& 1 + 9 \cdot \left(\frac{1}{9}\sum_{i=2}^{n} x_i\right)^{0.25} \end{aligned} \tag{ZDT6}$$

with $n = 10$ and $x_1 \in [0, 1]$ for $i = 1, \ldots, 10$. The Pareto set is given by

$$P_Q = [0, 1] \times \{0^{n-1}\} \ , \tag{ZDT6-P_Q}$$

To write the Pareto front in the form of Eq. (4.15), note that the parameter t is in the image of f_1, see Eq. (ZDT6). Numerically, by means of a software such as MATLAB, we estimate $t \in [0.28, 1]$. The Pareto front of the ZDT6 problem can be written as follows.

$$\begin{aligned} \gamma : [0.28, 1] &\to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - t^2 \end{pmatrix} \end{aligned} \tag{ZDT6-$F(P_Q)$}$$

Fig. 5.5 shows results for different values of δ, on application of Alg. 4.1.

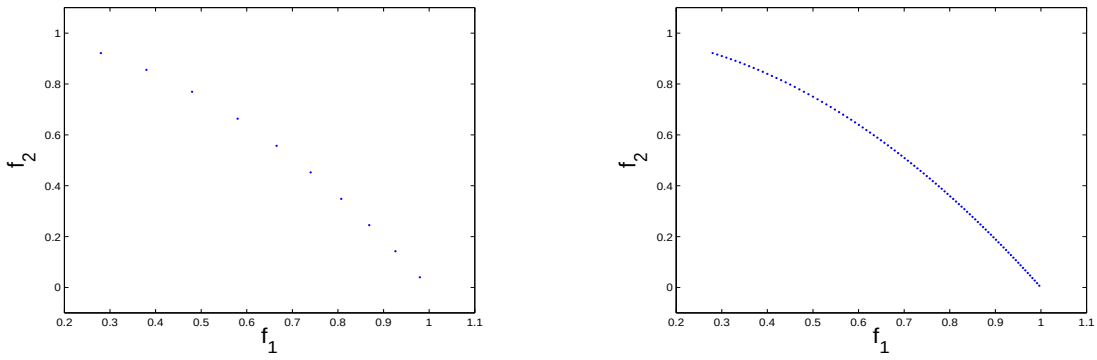


Figure 5.5: Discretizations of the Pareto front of model ZDT6 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

Deb Fun2 This problem has a finite number of disconnected Pareto fronts.

$$\begin{aligned} f_1(x) =& x_1 \\ f_2(x) =& (1 + 10x_2) \cdot g(x) \\ & \text{where :} \\ g(x) =& 1 - \left(\frac{f_1(x)}{(1 + 10x_2)}\right)^2 - \frac{f_1(x)}{(1 + 10x_2)} \cdot \sin(12\pi f_1) \end{aligned} \tag{Deb2}$$

with $x_i \in [0, 1]$ for $i = 1, 2$. The Pareto set is given by

$$P_Q = [0, 1] \times \{0\} \tag{Deb2-P_Q}$$

and the Pareto front is contained inside

$$\begin{aligned} \gamma : [0, 1] &\to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - t^2 - t\sin(12\pi t) \end{pmatrix} \end{aligned} \tag{Deb2-$F(P_Q)$}$$

Fig. 5.6 shows results for different values of δ, on application of Alg. 4.1. Note that after the application of the algorithm all the dominated points have to be removed.

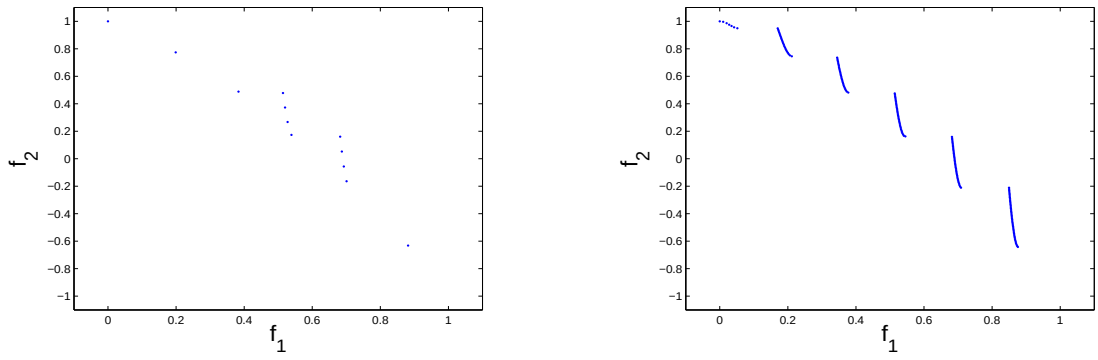


Figure 5.6: Discretizations of the Pareto front of model Deb2 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

Deb Fun3 This problem has a nonconvex and connected Pareto front.

$$\begin{aligned} f_1(x) =& 1 - \mathrm{e}^{-4x_1} \sin(10\pi x_1)^4 \\ f_2(x) =& \left(1 + x_2^2\right) \cdot g(x) \\ & \text{where :} \\ g(x) =& \begin{cases} 1 - \left(\frac{f_1(x)}{(1+x_2^2)}\right)^{10} & \text{if } f_1(x) \leq (1 + x_2^2) \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{Deb3}$$

with $x_i \in [0, 1]$ for $i = 1, 2$. The Pareto set is given by

$$P_Q = [0, 1] \times \{0\} \tag{Deb3-P_Q}$$

and the Pareto front is given by

$$\begin{aligned} \gamma : [0, 1] &\to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - t^{10} \end{pmatrix} \end{aligned} \tag{Deb3-$F(P_Q)$}$$

Fig. 5.7 shows results for different values of δ, on application of Alg. 4.1.

Lis This problem has a nonconvex and connected Pareto front.

$$\begin{aligned} f_1(x) &= \left(x_1^2 + x_2^2\right)^{\frac{1}{8}} \\ f_2(x) &= \left((x_1 - 0.5)^2 + (x_2 - 0.5)^2\right)^{\frac{1}{4}} \end{aligned} \tag{Lis}$$

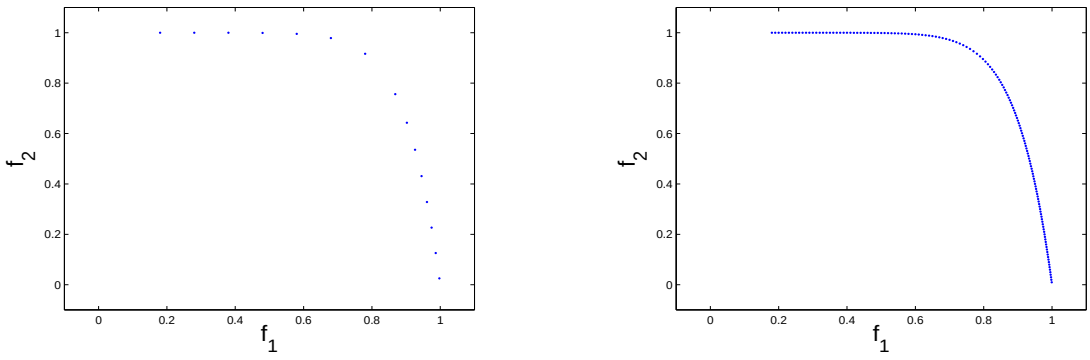


Figure 5.7: Discretizations of the Pareto front of model Deb3 using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

with $x_i \in [-5, 10]$ for $i = 1, 2$. The Pareto set is given by the simplex

$$\begin{aligned} P_Q =& S(p_a, p_b) \\ \text{where :} & \\ p_a =& (0,0)^T \text{ and } p_b = (0.5, 0.5)^T \end{aligned} \tag{Lis-P_Q}$$

There is no analytical expression for the Pareto front. Fig. 5.8 shows results for different values of δ, on application of Alg. 4.2 with a starting point $p_a = (0,0)^T$ to the direction $\nu = (1,1)^T$.

Okabe Fun2 This problem has a nonconvex and connected Pareto front.

$$\begin{aligned} f_1 =& x_1 \\ f_2 =& 1 - \frac{1}{4\pi^2}(x_1 + \pi)^2 + |x_2 - 5\cos(x_1)|^{\frac{1}{3}} + |x_3 - 5\sin(x_1)|^{\frac{1}{3}} \end{aligned} \tag{Oka2}$$

with $x_1 \in [-\pi, \pi]$ and $x_2, x_3 \in [-5, 5]$. The Pareto set is given by

$$\begin{aligned} P_Q : [-\pi, \pi] &\to \mathbb{R}^3 \\ P_Q(t) &= \begin{pmatrix} t \\ 5\cos(t) \\ 5\sin(t) \end{pmatrix} \end{aligned} \tag{Oka2-P_Q}$$

and the Pareto front is given by

$$\begin{aligned} \gamma : [-\pi, \pi] &\to \mathbb{R}^2 \\ \gamma(t) &= \begin{pmatrix} t \\ 1 - \frac{1}{4\pi^2}(t + \pi)^2 \end{pmatrix} \end{aligned} \tag{Oka2-$F(P_Q)$}$$

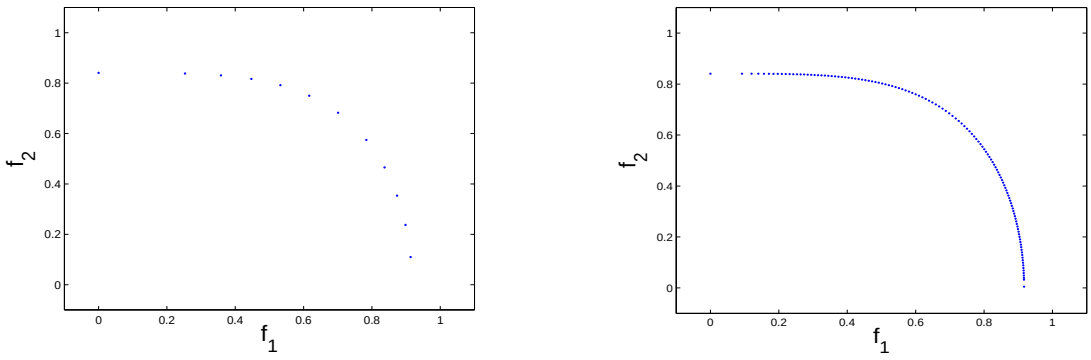
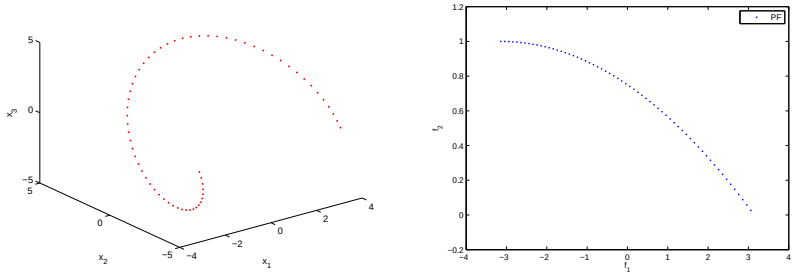
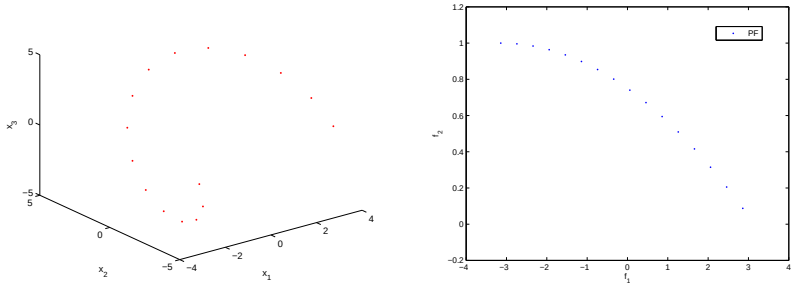


Figure 5.8: Discretizations of the Pareto front of problem Lis using a continuation method together with the step size control described in Section 4.3. We show the results for $\delta = 0.1$ (left) and $\delta = 0.01$ (right).

Fig. 5.9 shows results for different values of δ, on application of Alg. 4.3 with $\|G(t)\|_\infty = \max\left\{1, |5\sin(t)|, |5\cos(t)|\right\}$.



(a) $\delta = 0.1$



(b) $\delta = 0.4$

Figure 5.9: Discretizations of the Pareto front of model Oka2 using a continuation method together with the step size control described in Section 4.3.

5.2 Measuring the Performance of NSGA-II on DTLZ1

Next, we are interested in measuring the performance of a modern Pareto-based MOEA on a benchmark model (DTLZ1). Here, we have decided for the well-known algorithm NSGA-II, since this is a widely accepted state-of-the-art MOEA, and (DTLZ1) contains weakly optimal Pareto points which are easily detected—but not easily discarded—by a MOEA.

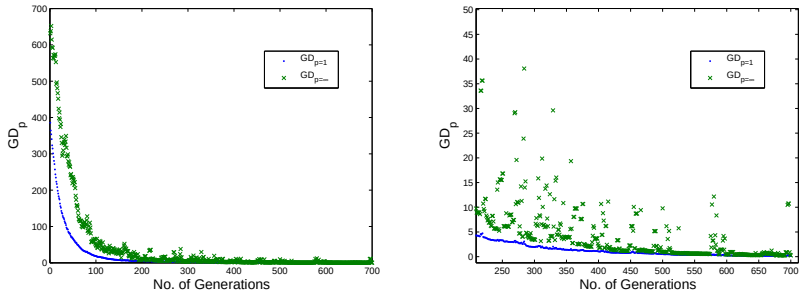
DTLZ1

$$\begin{aligned} f_1(x) =& \frac{1}{2} x_1 x_2 \left(1 + g(x)\right) \\ f_2(x) =& \frac{1}{2} x_1 \left(1 - x_2\right) \left(1 + g(x)\right) \\ f_3(x) =& \frac{1}{2} \left(1 - x_1\right) \left(1 + g(x)\right) \\ & \text{where :} \\ g(x) =& 100 \left(10 + \sum_{i=3}^{n} \left(x_i - 0.5\right)^2 - \cos(20\pi \left(x_i - 0.5\right)) \right) \end{aligned} \tag{DTLZ1}$$

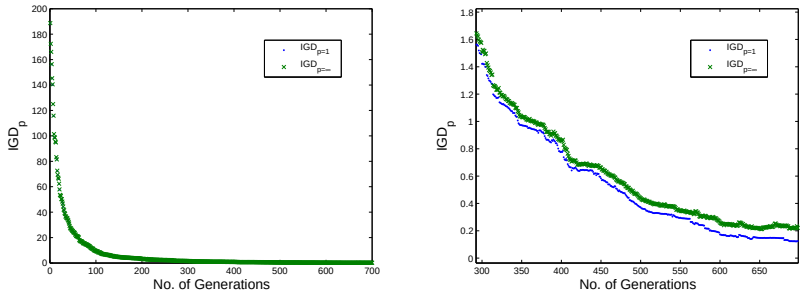
with $n = 12$ and $x_i \in [0, 1]$ for $i = 1, \ldots, 12$. The Pareto front is given by the simplex:

$$\begin{aligned} P_Q =& S(p_a, p_b, p_c) \\ & \text{where :} \\ p_a =& (0.5, 0, 0)^T \\ p_b =& (0, 0.5, 0)^T \\ p_c =& (0, 0, 0.5)^T \end{aligned} \tag{DTLZ1-$F(P_Q)$}$$

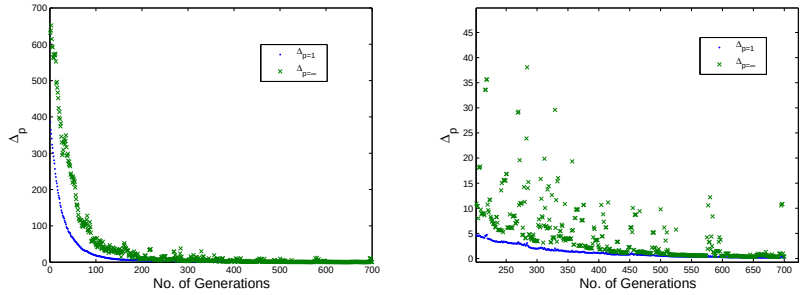
Fig. 5.10 and Table 5.1 show the values of GD_p, IGD_p, and Δ_p for the extreme values $p = 1$ and $p = \infty$ for the first 700 generations (averaged over 50 independent runs using population size $N_{pop} = 60$). In general, a convergent behavior can be observed, which differs, however, for the different values of p: while for $p = 1$ all curves of the indicators values are nearly 'smooth'. This is not the case for $p = \infty$, where jumps in the indicator values can be observed.



(a) GD_p Values



(b) IGD_p Values



(c) Δ_p Values

Figure 5.10: Numerical results of NSGA-II on the DTLZ1 model, measured by GD_p, IGD_p, and Δ_p for $p = 1$ and $p = \infty$ (compare to Table 5.1). The results are averaged over 50 independent ranges of generations. The left figures show the result of the entire run, and the figures on the right show a zoom.

Table 5.1: Numerical results of NSGA-II on the DTLZ1 model, measured by GD_p, IGD_p, and Δ_p for $p = 1$ and $p = \infty$.

	No. of Generations						
	100	200	300	400	500	600	700
GD_1	18.586	4.682	1.953	1.061	0.670	0.239	0.128
GD_∞	40.051	10.954	3.510	2.205	9,791	0.466	0.253
IGD_1	9.327	3.123	1.421	0.778	0.371	0.173	0.124
IGD_∞	9.467	3.217	1.506	0.861	0.438	0.260	0.233
Δ_1	18.586	4.682	1.953	1.061	0.670	0.239	0.128
Δ_∞	40.051	10.954	3.510	2.205	9,791	0.466	0.253

5.3 Evaluation of ArchiveUpdateTight Results

Here we will use Definitions 5 and 6 (see Section 2.5.3 or [46] for a further explanation), in order to see the relation between the indicator developed and the archiving strategies used in ELMA.

*ArchiveUpdateTight*1 It generates an (δ, ϵ)-tight ϵ-approximate Pareto set A_1, where $\delta \in \mathbb{R}_+$, $\epsilon \in \mathbb{R}_+^k$ are discretization parameters. For a limit archive it holds

$$\begin{aligned} &\text{dist}(F(A_1), F(P_Q)) \leq \epsilon, \quad \text{and} \\ &\text{dist}(F(P_Q), F(A_1)) \leq \delta. \end{aligned} \tag{5.1}$$

Since ϵ-approximate solutions are considered to be 'good enough' by *ArchiveUpdateTight*1, they are not replaced by dominating solutions any more. That, the uniformity level ϵ (i.e., $\|F(a_1) - F(a_2)\|_\infty \geq \epsilon \ \forall a_1, a_2 \in A_1, \ a_1 \neq a_2$) can be guaranteed, but no convergence toward the Pareto front.

*ArchiveUpdateTight*2 It generates an δ-tight ϵ-approximate Pareto set, by *ArchiveUpdateTight*2. It is expected that for a limit archive it holds

$$\begin{aligned} &\text{dist}(F(A_2), F(P_Q)) = 0, \quad \text{and} \\ &\text{dist}(F(P_Q), F(A_2)) \leq \delta. \end{aligned} \tag{5.2}$$

The images of the archive entries (*ArchiveUpdateTight*2) have to converge toward the Pareto front, altough the uniformity gets lost.

To investigate the performance of the two archivers we will use following MOP.

$$\min_{x \in Q} F(x) = x, \tag{5.3}$$

where $F : \mathbb{R}^k \to \mathbb{R}^k$ and the domain Q is given by

$$Q = \left\{ x \in \mathbb{R}^k \ : \ x_i \in [0, 10], i = 1, \ldots, k, \text{ and } \sum_{i=1}^{k} x_i \geq 1 \right\}. \tag{5.4}$$

Hereby, Pareto set and front are given by the $(k-1)$-standard simplex

$$P_Q = F(P_Q) = S(e_1, \ldots, e_k) \tag{5.5}$$

Fig. 5.11 shows the final archive acquired for *ArchiveUpdateTight*1 and *ArchiveUpdateTight*2 with 1×10^6 randomly choosen points. The values used are $\epsilon = \delta = 0.1$ (see Section 2.5.3 Definitions 5 and 6), and the discretization error $\delta_{err} = 0.001$ (Section 4.2.2, Proposition 3). Table 5.2 shows that as the p-norm increases, the three indicators (GD_p, IGD_p and Δ_p) are close to the value of δ, see Eq. (5.1). Table 5.3 shows that as the p-norm increases, the indicator (GD_p) is close to zero and the two indicators (IGD_p and Δ_p) are close to the value of δ, see Eq. (5.2).

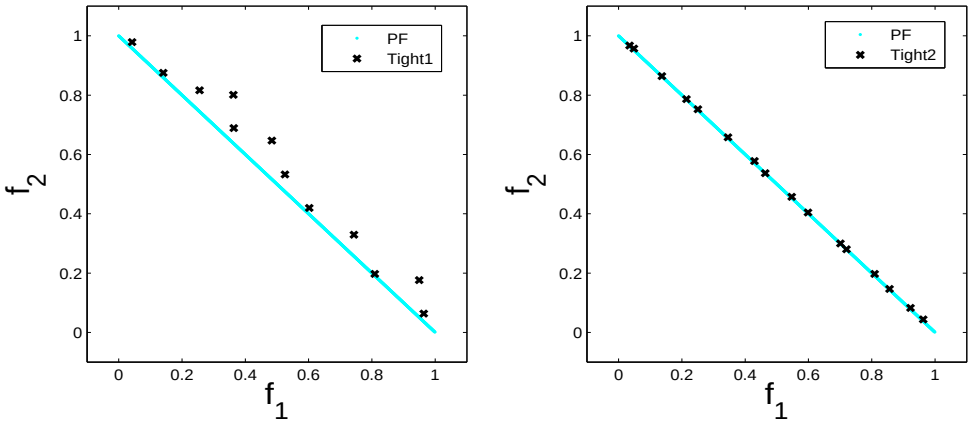


Figure 5.11: Final archive acquired of 1×10^6 randomly choosen points, *ArchiveUpdateTight*1 (left), *ArchiveUpdateTight*2 (right).

Table 5.2: Values of GD_p, IGD_p and Δ_p for $ArchiveUpdateTight1$ with a discretization error $\delta_{err} = 0.001$.

	$p = 1$	$p = 2$	$p = 10$	$p = \infty$
GD_p	0.0377	0.0496	0.0759	0.0912
IGD_p	0.0506	0.0562	0.0760	0.1027
Δ_p	0.0506	0.0562	0.0760	0.1027

Table 5.3: Values of GD_p, IGD_p and Δ_p for $ArchiveUpdateTight2$ with a discretization error $\delta_{err} = 0.001$.

	$p = 1$	$p = 2$	$p = 10$	$p = \infty$
GD_p	0.0006	0.0007	0.0012	0.0016
IGD_p	0.0270	0.0320	0.0484	0.0694
Δ_p	0.0270	0.0320	0.0484	0.0694

5.4 Evaluation of ELMA Algorithm

To evaluate the performance of the new algorithm we have chosen to take the ZDT benchmark suite plus four further models (see Section 5.1). We compare our results to the original version of $\epsilon - MOEA$ as well as a slight variant of this algorithm, ϵ-MOEA+Eps2, which is identical to the original algorithm but equipped with the archiver $ArchiveUpdateEps2$, we have chosen the value of $\eta = 1$ for all operators. This is done to investigate the effect of the archiver within the MOEA. To compare the performance, we have chosen to use the GD_p, IGD_p and the Δ_p indicator. To be more precise, given a candidate set (or archive) $A = \{a_1, \ldots, a_N\}$ (in image space) and a discrete (or discretized) Pareto front $F(P_Q) = \{y_1, \ldots, y_M\}$ with an error of $\delta = 0.001$.

A low of Δ_p indicates a low value of both GD_p and IGD_p, and the approximation can be considered to be 'good' in the Hausdorff sense. In all indicators, the value of p is used to handle the outliers in the candidate set which can be explained by looking at the extremes: for $p = 1$, the distances are averaged while for $p = \infty$ only the candidate with the largest distance is considered. The p-norm is used which is replaced by the maximum norm for $p = \infty$.

Tables from 5.5 to 5.13 and Figs. from 5.12 to 5.20 show the results for the test functions (the PF in the tables means the discrete Pareto front). When considering the averaged

results (i.e., $p = 1$), the new algorithm outperforms (in some cases even significantly) the two ϵ-MOEA versions. The differences, however, get smaller when increasing the value of p which indicates that ELMA tends to generate and maintain outliers in the approximation. One exception is the highly multimodal model ZDT4, where ϵ-MOEA clearly outperforms ELMA. A possible explanation is that the new algortihm gets more likely trapped in locally optimal regions due to the use of the new operators (which are both of local nature).

Table 5.4 shows the size of the archivers for each of the algorithms as well as the number of evaluations.

Table 5.4: Size of Archivers and number of evaluations

Function	ELMA+Eps2 Archive size	ϵ-MOEA+EPS2 Archive size	ϵ-MOEA Archive size	Evaluations
ZDT1	59.95	44.35	41.60	5005
ZDT2	64.65	21.75	21.45	5001
ZDT3	33.75	35.00	25.75	5003
ZDT4	82.35	71.30	68.40	5004
ZDT6	39.70	14.05	14.75	5002
Deb_fun2	23.80	29.30	22.10	1501
Deb_fun3	33.25	36.40	30.70	1002
Lis	45.30	36.35	33.95	1505
Oka_fun2	17.05	16.05	12.45	5001

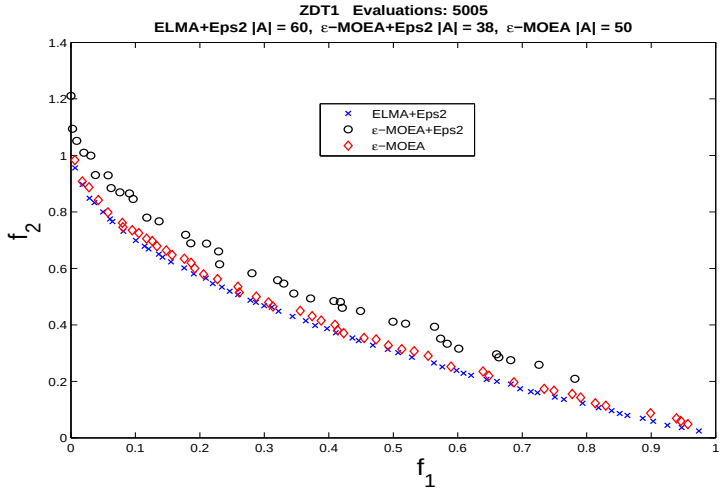


Figure 5.12: Representative solutions of the three algorithms on ZDT1.

Table 5.5: Numerical results for ZDT1 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0229	0.0249	0.0341	0.0740
Δ_p(ϵ-MOEA+Eps2, PF)	0.0604	0.0683	0.0984	0.1935
Δ_p(ϵ-MOEA, PF)	0.0535	0.0601	0.0778	0.1395
GD$_p$(ELMA+Eps2, PF)	0.0195	0.0200	0.0215	0.0336
GD$_p$(ϵ-MOEA+Eps2, PF)	0.0560	0.0610	0.0835	0.1631
GD$_p$(ϵ-MOEA, PF)	0.0455	0.0470	0.0499	0.0656
IGD$_p$(ELMA+Eps2, PF)	0.0227	0.0247	0.0333	0.0673
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.0584	0.0634	0.0773	0.1292
IGD$_p$(ϵ-MOEA, PF)	0.0534	0.0599	0.0775	0.1390

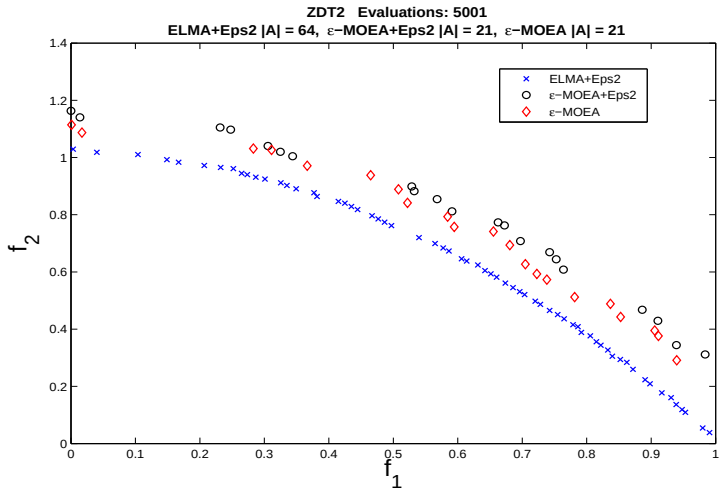


Figure 5.13: Representative solutions of the three algorithms on ZDT2.

Table 5.6: Numerical results for ZDT2 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0144	0.0162	0.0245	0.0545
Δ_p(ϵ-MOEA+Eps2, PF)	0.2183	0.2346	0.2856	0.4753
Δ_p(ϵ-MOEA, PF)	0.1775	0.1940	0.2415	0.4066
GD$_p$(ELMA+Eps2, PF)	0.0112	0.0122	0.0164	0.0299
GD$_p$(ϵ-MOEA+Eps2, PF)	0.1820	0.1839	0.1911	0.2560
GD$_p$(ϵ-MOEA, PF)	0.1364	0.1374	0.1401	0.1656
IGD$_p$(ELMA+Eps2, PF)	0.0144	0.0156	0.0212	0.0468
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.2183	0.2346	0.2856	0.4753
IGD$_p$(ϵ-MOEA, PF)	0.1775	0.1940	0.2415	0.4066

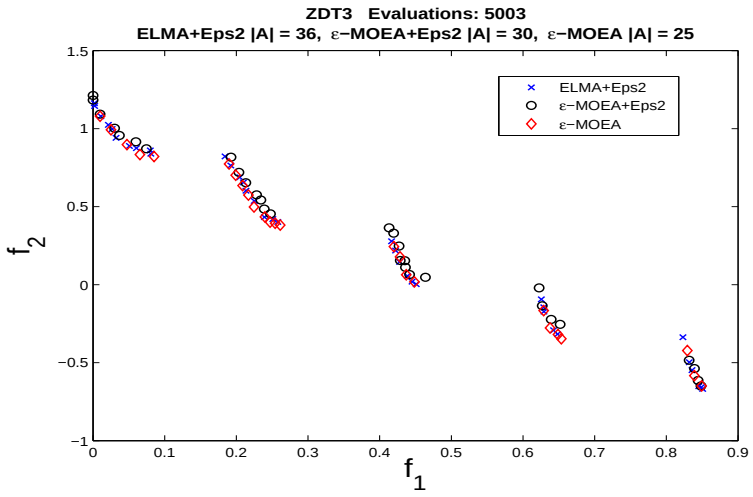


Figure 5.14: Representative solutions of the three algorithms on ZDT3.

Table 5.7: Numerical results for ZDT3 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0369	0.0442	0.0667	0.1311
Δ_p(ϵ-MOEA+Eps2, PF)	0.0862	0.1184	0.1942	0.3597
Δ_p(ϵ-MOEA, PF)	0.0579	0.0733	0.1057	0.1837
GD$_p$(ELMA+Eps2, PF)	0.0245	0.0311	0.0495	0.0913
GD$_p$(ϵ-MOEA+Eps2, PF)	0.0711	0.1003	0.1691	0.3165
GD$_p$(ϵ-MOEA, PF)	0.0378	0.0510	0.0824	0.1428
IGD$_p$(ELMA+Eps2, PF)	0.0369	0.0436	0.0603	0.1166
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.0849	0.1087	0.1531	0.2555
IGD$_p$(ϵ-MOEA, PF)	0.0579	0.0732	0.1041	0.1790

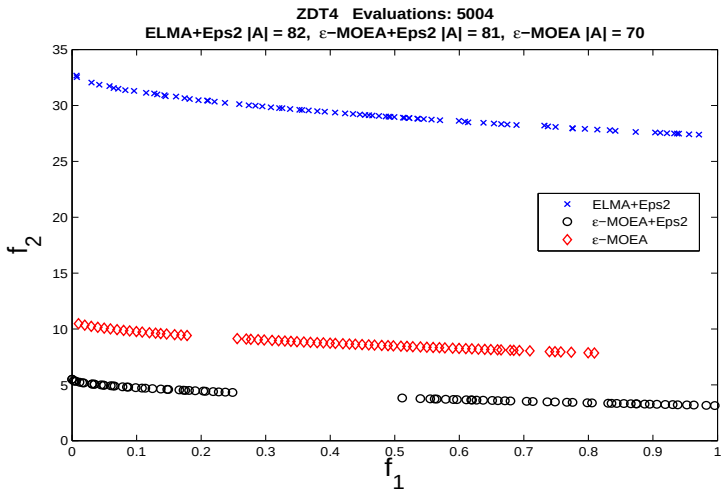


Figure 5.15: Representative solutions of the three algorithms ON ZDT4.

Table 5.8: Numerical results for ZDT4 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	19.0547	19.1791	19.9281	24.0174
Δ_p(ϵ-MOEA+Eps2, PF)	7.9959	8.0403	8.1726	9.6835
Δ_p(ϵ-MOEA, PF)	9.8138	9.8506	9.9646	11.5986
GD$_p$(ELMA+Eps2, PF)	19.0547	19.1791	19.9281	24.0174
GD$_p$(ϵ-MOEA+Eps2, PF)	7.9856	8.0300	8.1624	9.6795
GD$_p$(ϵ-MOEA, PF)	9.8021	9.8389	9.9529	11.5872
IGD$_p$(ELMA+Eps2, PF)	18.1554	18.1583	18.1671	18.5697
IGD$_p$(ϵ-MOEA+Eps2, PF)	7.4404	7.4462	7.4629	7.8471
IGD$_p$(ϵ-MOEA, PF)	9.3215	9.3263	9.3404	9.7302

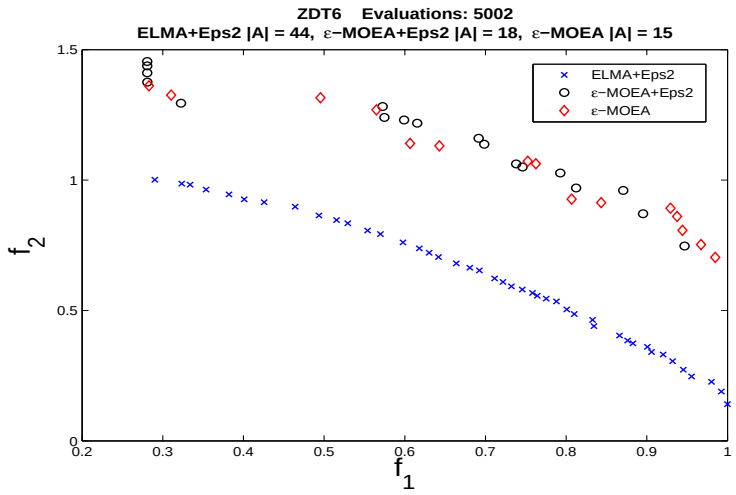


Figure 5.16: Representative solutions of the three algorithms on ZDT6.

Table 5.9: Numerical results for ZDT6 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0970	0.0987	0.1073	0.1862
Δ_p(ϵ-MOEA+Eps2, PF)	0.7465	0.7621	0.8119	1.0969
Δ_p(ϵ-MOEA, PF)	0.5478	0.5618	0.6078	0.8570
GD$_p$(ELMA+Eps2, PF)	0.0943	0.0946	0.0955	0.1142
GD$_p$(ϵ-MOEA+Eps2, PF)	0.6842	0.6884	0.7024	0.8504
GD$_p$(ϵ-MOEA, PF)	0.4887	0.4901	0.4943	0.5536
IGD$_p$(ELMA+Eps2, PF)	0.0968	0.0986	0.1073	0.1862
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.7452	0.7615	0.8119	1.0969
IGD$_p$(ϵ-MOEA, PF)	0.5478	0.5618	0.6078	0.8570

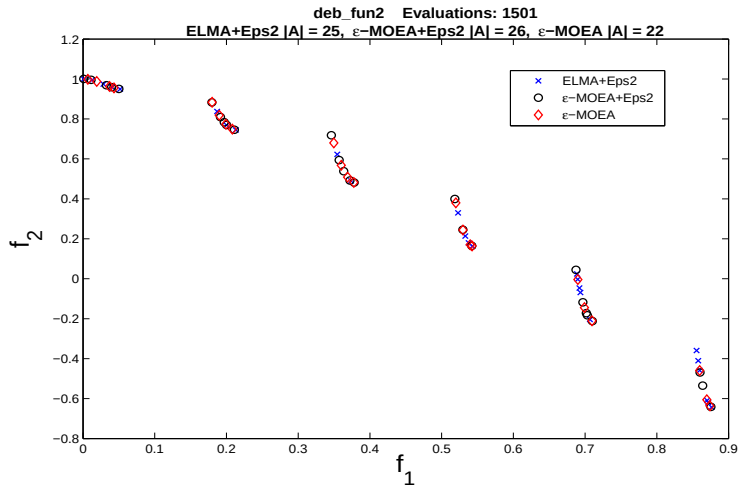


Figure 5.17: Representative solutions of the three algorithms on Deb_fun2.

Table 5.10: Numerical results for Deb_fun2 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0507	0.0743	0.1168	0.2116
Δ_p(ϵ-MOEA+Eps2, PF)	0.0557	0.0816	0.1251	0.2192
Δ_p(ϵ-MOEA, PF)	0.0583	0.0839	0.1295	0.2277
GD$_p$(ELMA+Eps2, PF)	0.0005	0.0007	0.0013	0.0024
GD$_p$(ϵ-MOEA+Eps2, PF)	0.0015	0.0043	0.0105	0.0201
GD$_p$(ϵ-MOEA, PF)	0.0016	0.0050	0.0115	0.0209
IGD$_p$(ELMA+Eps2, PF)	0.0507	0.0743	0.1168	0.2116
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.0557	0.0816	0.1251	0.2192
IGD$_p$(ϵ-MOEA, PF)	0.0583	0.0839	0.1295	0.2277

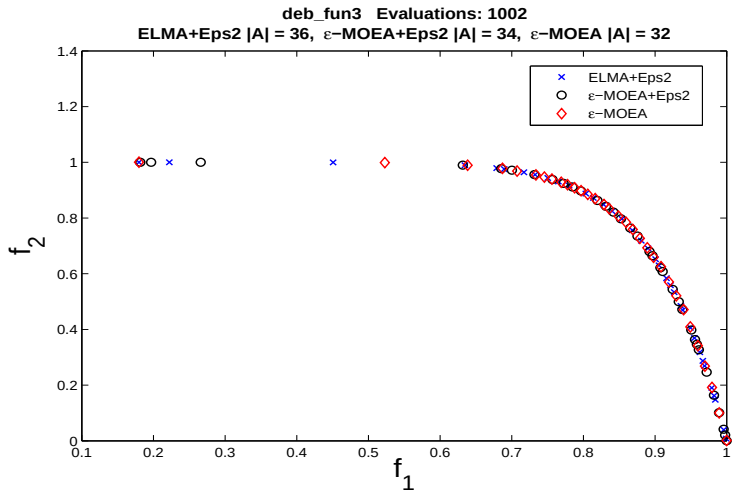


Figure 5.18: Representative solutions of the three algorithms on Deb_fun3.

Table 5.11: Numerical results for Deb_fun3 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0368	0.0620	0.1074	0.2003
Δ_p(ϵ-MOEA+Eps2, PF)	0.0356	0.0597	0.1032	0.1927
Δ_p(ϵ-MOEA, PF)	0.0320	0.0500	0.0851	0.1617
GD$_p$(ELMA+Eps2, PF)	0.0004	0.0006	0.0010	0.0017
GD$_p$(ϵ-MOEA+Eps2, PF)	0.0004	0.0004	0.0006	0.0010
GD$_p$(ϵ-MOEA, PF)	0.0003	0.0004	0.0005	0.0008
IGD$_p$(ELMA+Eps2, PF)	0.0368	0.0620	0.1074	0.2003
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.0356	0.0597	0.1032	0.1927
IGD$_p$(ϵ-MOEA, PF)	0.0320	0.0500	0.0851	0.1617

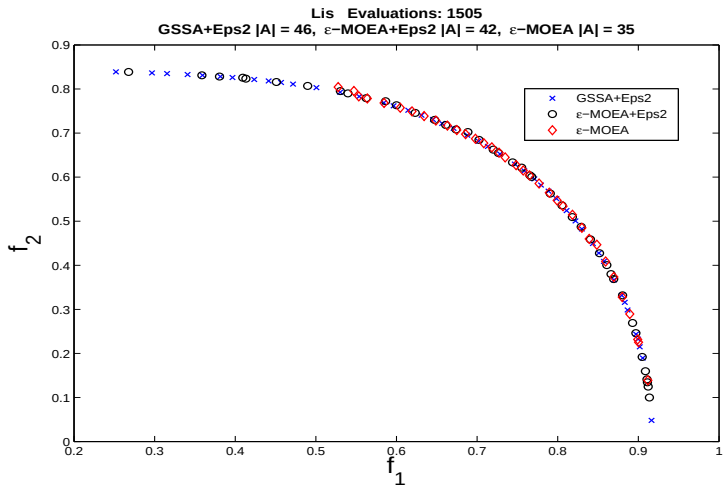


Figure 5.19: Representative solutions of the three algorithms on Lis.

Table 5.12: Numerical results for Lis (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	0.0192	0.0361	0.0784	0.2247
Δ_p(ϵ-MOEA+Eps2, PF)	0.0631	0.1106	0.1906	0.4155
Δ_p(ϵ-MOEA, PF)	0.0657	0.1162	0.2015	0.4381
GD$_p$(ELMA+Eps2, PF)	0.0005	0.0006	0.0010	0.0021
GD$_p$(ϵ-MOEA+Eps2, PF)	0.0016	0.0025	0.0044	0.0084
GD$_p$(ϵ-MOEA, PF)	0.0013	0.0022	0.0042	0.0077
IGD$_p$(ELMA+Eps2, PF)	0.0192	0.0361	0.0784	0.2247
IGD$_p$(ϵ-MOEA+Eps2, PF)	0.0631	0.1106	0.1906	0.4155
IGD$_p$(ϵ-MOEA, PF)	0.0657	0.1162	0.2015	0.4381

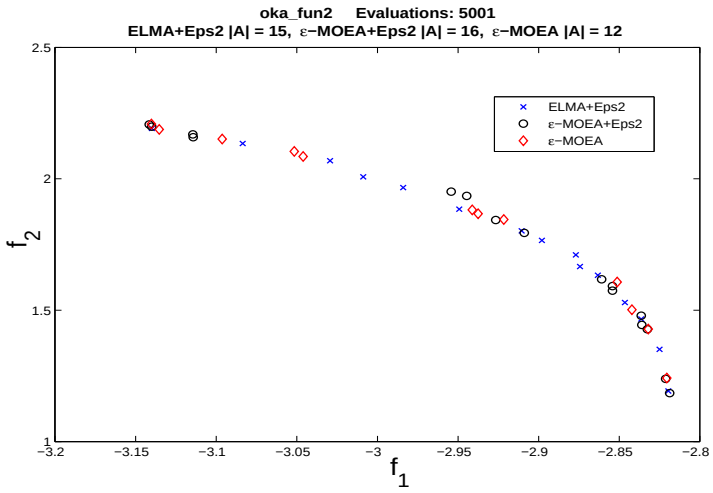


Figure 5.20: Representative solutions of the three algorithms on Oka_fun2.

Table 5.13: Numerical results for Oka_fun2 (averaged over 30 runs).

	p=1	p=2	p=5	$p = \infty$
Δ_p(ELMA+Eps2, PF)	2.5371	2.9914	3.7104	5.4530
Δ_p(ϵ-MOEA+Eps2, PF)	2.4830	2.9186	3.6114	5.3063
Δ_p(ϵ-MOEA, PF)	2.3745	2.7952	3.4789	5.1531
GD$_p$(ELMA+Eps2, PF)	0.7450	0.7999	0.9030	1.1966
GD$_p$(ϵ-MOEA+Eps2, PF)	0.8030	0.8661	0.9706	1.2149
GD$_p$(ϵ-MOEA, PF)	0.8107	0.8729	0.9724	1.2028
IGD$_p$(ELMA+Eps2, PF)	2.5371	2.9914	3.7104	5.4530
IGD$_p$(ϵ-MOEA+Eps2, PF)	2.4830	2.9186	3.6114	5.3063
IGD$_p$(ϵ-MOEA, PF)	2.3745	2.7952	3.4789	5.1531

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this work, we have made a first attempt to design variation operators of an archive based MOEA tailored to the characteristics of the external archive as well as the interplay of generation and the archive. In particular, we have considered archivers based on the concept of ϵ-dominance. Further, we have proposed and investigated a new performance indicator in order to give a 'fair' evaluation of the outcome set of ELMA. To be more precise, we have proposed the indicator Δ_p which can be considered as an 'averaged Hausdorff distance'.
A brief description of our contribution are provided next:

1. We proposed two variation operators (SPM and BIC), which are local search procedures based on Lipschitz estimations in order to detect suitable neighbor solutions of given archive entries. SPM has shown good performance with a high probability of ocurrence, while the local nature of BIC allows to be more effective when the parent solutions are already near to the Pareto set. The two operators are endowed to generate the step size in a probabilistic sense (the idea behind this was taken from the SBX operator).

2. Our first proposal (MOEA1) works only with the two new operators suited to its archive. The MOEA1 shows a good performance with models of low dimensions, but not with high-dimensional models. Based on this observation, we have presented an improved algorithm (ELMA) which is a variant of ϵ-MOEA (plus SPM and BIC) and offers a better performance on the ZDT benchmark suite than its base algorithm (except for ZDT4). Further, the ELMA algorithm is not sensitive to outliers.

3. We gave a brief study between the generation process and the archive. A special attention to the outliers problem was given. Further, an alternative way to supress them was provided. Even with the extra function call, necessary in SPM and BIC, the total number of function calls has been reduced. An idea to handle the algorithm if the archive has a single element is also explained.

4. We proposed a slight modification of GD and IGD. The new variant of GD (called GD_p) reduces the effect of decreasing its value by just adding more elements into the archive, even if most of them are dominated. The new variant of IGD (called IGD_p) has certain relations to other distance measures used in the EMO literature.

5. We proposed a 'new' indicator Δ_p, which consists of GD_p and IGD_p, and which can be viewed as an averaged Hausdorff distance. We also addressed one possiblity to handle the 'outlier trade off', penalizing single outliers but having a metric against supressing the influence of outliers. This, however, loses the metric properties by violating the triangle inequality. Furthermore, the discretization error is given (this can be handled by the discretization of the Pareto front).

6. We gave a technique to discretize the Pareto front if an analytical expression of the Pareto front or set is available.

6.2 Future Work

This work is certainly just an initial step and by far not complete yet. In the future, more investigation on the variation operators has to be done, including a more advanced interplay of the operators and the archive elements, and more tests have to be performed. In particular, the bias of the approach to generate outliers has to be suppressed as well as the possibility to get trapped in locally optimal regions (as for ZDT4) has to be reduced by refining the balance of global and local search, as well as, to extend the algorithm in order to work with functions with more than two objectives. Finally, the method should be modularized such that the algorithm works efficiently for different sets of interest related to an MOP (e.g., Pareto set, Pareto front, set of approximate solutions, etc.).

Bibliography

[1] T. Bäck. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, 1996.

[2] P. J. Bentley and J. P. Wakefield. Finding Acceptable Solutions in the Pareto-Optimal Range using Multiobjective Genetic Algorithms. In P. K. Chawdhry, R. Roy, and R. K. Pant, editors, *Soft Computing in Engineering Design and Manufacturing*, Part 5, pages 231–240, London, June 1997. Springer Verlag London Limited. (Presented at the 2nd On-line World Conference on Soft Computing in Design and Manufacturing (WSC2)).

[3] M. Brown and R. E. Smith. Directed Multi-Objective Optimisation. *International Journal of Computers, Systems and Signals*, 6(1):3–17, 2005.

[4] P. J. Fleming C.M. Fonseca. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proceedings of the Fifth International Conference on Genetic Algorithms*, pages 416–423, San Mateo, California, 1993. Morgan Kauffman Publishers.

[5] C. A. Coello Coello and Nareli Cruz Cortés. Solving Multiobjective Optimization Problems using an Artificial Immune System. *Genetic Programming and Evolvable Machines*, 6(2):163–190, June 2005.

[6] C. A. Coello Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Springer, New York, second edition, 2007.

[7] C. A. Coello Coello, David A. Van Veldhuizen, and Gary B. Lamont. *Evolutionary Algorithms for Solving Multi-Objective Problems*. Kluwer Academic Publishers, New York, May 2002. ISBN 0-3064-6762-3.

[8] L. G. de la Fraga and O. Schütze. Direct calibration by fitting of cuboids to a single image using differential evolution. *International Journal of Computer Vision*, 81(2):119–127, 2009.

[9] K. Deb and R. Agrawal. Simulated binary crossover for continuous search space. Technical report, Indian Institute of Technology kanpur, 1994.

[10] K. Deb and R. Agrawal. A niched-penalty approach for constraint handling in genetic algorithms. Technical report, Indian Institute of Technology kanpur, 1999.

[11] K. Deb and H. Beyer. Self-adaptive genetic algorithms with simulated binary crossover. *Evolutionary Computation*, 9(2):197–221, 2001.

[12] K. Deb, M. Mohan, and S. Mishra. Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. *Evol. Comput.*, 13(4):501–525, 2005.

[13] Kalyanmoy Deb. Multi-objective genetic algorithms: Problem difficulties and construction of test problems. *Evolutionary Computation*, 7:205–230, 1999.

[14] Kalyanmoy Deb. *Multi-Objective Optimization using Evolutionary Algorithms*. John Wiley & Sons, Chichester, UK, 2001. ISBN 0-471-87339-X.

[15] L. Thiele E. Zitzler, M. Laumanns. Spea2: Improving the strength pareto evolutionary algorithm. Technical report 103, computer engineering and networks laboratory (TIK), swiss federal institute of technology (eth) zurich, gloriastrasse 35, ch-8092 zurich, switzerland., May 2001.

[16] A. E. Eiben and J. E. Smith. *Introduction to Evolutionary Computing*. Springer, 2007. ISBN 978-3-540-40184-1.

[17] L. J. Eshelman and J. D. Schaffer. Real-coded genetic algorithms and interval-schemata. In Darrell L. Whitley, editor, *Foundation of Genetic Algorithms 2*, pages 187–202, San Mateo, CA, 1993. Morgan Kaufmann.

[18] D. E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, 1989.

[19] W. Hamming. *The Art of Probability / For Scientists and Engineers*. 1991.

[20] D. E. Goldberg J. Horn, N. Nafpliotis. A niched pareto genetic algorithm for multiobjective optimization. In *In Proceedings of the First IEEE Conference on Evolutionary Computation*, volume 1, pages 82–87, Piscataway, New Jersey, 1994. IEEE World Congress on Computational Intelligence.

[21] A. Jaszkiewicz. Genetic local search for multiple objective combinatorial optimization. Technical Report RA-014/98, Institute of Computing Science, Poznan University of Technology, 1998.

[22] H. John. *Adaptation in natural and artificial systems.* MIT Press, Cambridge, MA, USA, 1992.

[23] A. pratab K. Deb, S. Agrawal and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: Nsga-ii. *IEEE Transactions Evolutionary Computation*, 6(2):182–197, 2002.

[24] R. Kumar and P. Rockett. Improved Sampling of the Pareto-Front in Multiobjective Genetic Optimizations by Steady-State Evolution: A Pareto Converging Genetic Algorithm. *Evolutionary Computation*, 10(3):283–314, Fall 2002.

[25] A. Lara, G. Sanchez, C. A. Coello Coello, and O. Schütze. HCS: A New Local Search Strategy for Memetic Multi-Objective Evolutionary Algorithms. *IEEE Transactions on Evolutionary Computation*, 14(1):112–132, February 2010.

[26] M. Laumanns. Stochastic convergence of random search to fixed size pareto set approximations. Arxiv preprint arxiv:0711.2949, 2007.

[27] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282, 2002.

[28] J. Lis and A. E. Eiben. Multi-sexual genetic algorithm for multiobjective optimization. In *In Proceedings of the 1997 IEEE International Conference on Evolutionary Computaiton*, pages 59–64. IEEE, 1996.

[29] P. Loridan. ϵ-solutions in vector minimization problems. *Journal of Optimization, Theory and Application*, 42:265–276, 1984.

[30] A. C. M. Maxfield and L. Fogel. Articial intelligence through a simulation of evolution. In *Biophysics and Cybernetics Systems: Proceedings of the Second Cybernetics Sciences, Spartan Books.*, Washington D.C., 1965.

[31] N. Morse. Reducing the size of the nondominated set: Pruning by clustering. *Computers & OR*, 7(1-2):55–66, 1980.

[32] K. Deb N. Srinivas. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1994.

[33] J. Nocedal and S. Wright. *Numerical Optimization.* Springer Series in Operations Research and Financial Engineering. Springer, 2006.

[34] A. Lara O. Schütze and C. A. Coello Coello. On the influence of the number of objectives on the hardness of a multi-objective optimization problem. Technical report.

[35] T. Okabe, Y. Jin, M. Olhofer, and B. Sendhoff. On test functions for evolutionary multi-objective optimization. In *PPSN'04*, pages 792–802, 2004.

[36] Mícheál O'Searcoid. *Metric Spaces*. Springer, 2007. ISBN 1-84628-369-8.

[37] G. Rudolph. Finite Markov chain results in evolutionary computation: A tour d'horizon. *Fundamenta Informaticae*, 35:67–89, 1998.

[38] G. Rudolph and A. Agapie. On a multi-objective evolutionary algorithm and its convergence to the Pareto set. In *Congress on Evolutionary Computation (CEC2000)*, pages 1010–1016, 2000.

[39] S. Sayin. Measuring the quality of discrete representations of efficient sets in multiple objective mathematical programming. *Mathematical Programming*, 87(3):543–560, 2000.

[40] J.D. Schaffer. *Multiple Objective Optimization with Vector Evaluated Genetic Algorithms*. PhD thesis, Vanderbilt University, Tennessee, 1984.

[41] R. Schott. Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization. Master's thesis, Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, Massachusetts, May 1995.

[42] O. Schuetze, G. Sanchez, and C. A. Coello Coello. A new memetic strategy for the numerical treatment of multi-objective optimization problems. In *2008 Genetic and Evolutionary Computation Conference (GECCO'2008)*, pages 705–712. ACM Press, Atlanta, USA, July 2008. ISBN 978-1-60558-131-6.

[43] O. Schütze, C. A. Coello Coello, E. Tantar, and E.-G. Talbi. Computing finite size representations of the set of approximate solutions of an MOP with stochastic search algorithms. In *GECCO 2008: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, New York, NY, USA, 2008. ACM.

[44] O. Schütze, X. Esquivel, A. Lara, and C. Coello. A New Performance Indicator for MOEAs to Measure the Averaged Hausdorff Distance to the Pareto Front of a MOP. Technical report.

[45] O. Schütze, M. Laumanns, C. A. Coello Coello, M. Dellnitz, and E.-G. Talbi. Convergence of stochastic search algorithms to finite size Pareto set approximations. *Journal of Global Optimization*, 41(4):559–577, 2008.

[46] O. Schütze, M. Laumanns, E. Tantar, C. A. Coello Coello, and E.-G. Talbi. Computing gap free Pareto front approximations with stochastic search algorithms. *Evolutionary Computation*, 18(1):65–96, 2010.

[47] H.-P. Schwefel. *Evolution and Optimum Seeking: The Sixth Generation*. John Wiley & Sons, Inc., New York, NY, USA, 1993.

[48] D. A. Van Veldhuizen. *Multiobjective Evolutionary Algorithms: Classifications, Analyses, and New Innovations*. PhD thesis, Department of Electrical and Computer Engineering. Graduate School of Engineering. Air Force Institute of Technology, Wright-Patterson AFB, Ohio, May 1999.

[49] H.-M. Voigt, H. Mhlenbein, and D. Cvetkovic. Fuzzy recombination for the breeder genetic algorithm. In *Proc. Sixth Int. Conf. on Genetic Algorithms*, pages 104–111. Morgan Kaufmann Publishers, 1995.

[50] Saúl Zapotecas Martínez and C. A. Coello Coello. A Proposal to Hybridize Multi-Objective Evolutionary Algorithms with Non-Gradient Mathematical Programming Techniques. In Günter Rudolph, Thomas Jansen, Simon Lucas, Carlo Poloni, and Nicola Beume, editors, *Parallel Problem Solving from Nature–PPSN X*, pages 837–846. Springer. Lecture Notes in Computer Science Vol. 5199, Dortmund, Germany, September 2008.

[51] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8:173–195, 2000.

[52] E. Zitzler and L. Thiele. Multiobjective Optimization Using Evolutionary Algorithms—A Comparative Study. In A. E. Eiben, editor, *Parallel Problem Solving from Nature V*, pages 292–301, Amsterdam, September 1998. Springer-Verlag.

[53] E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength pareto approach. *Evolutionary Computation*, 3(4):257–271, 1999.

[54] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, November 1999.