Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco
Departamento de Computación

El Algoritmo de Pareto Tracer para Problemas de Optimización Multiobjetivo con Restricciones Generales de Desigualdad

TESIS QUE PRESENTA

María Fernanda Beltrán Llorente

PARA OBTENER EL GRADO DE

Maestro en Ciencias en Computación

DIRECTOR DE LA TESIS

Dr. Oliver Steffen Schütze

México, CDMX **Febrero 2020**

Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Zacatenco Campus
Computer Science Deparment

The Pareto Tracer for General Inequality Constrained Multi-objective Optimization Problems

SUBMITTED BY

María Fernanda Beltrán Llorente

AS A FULFILLMENT OF THE REQUIREMENT FOR THE DEGREE OF

Master in Computer Science

ADVISOR

Dr. Oliver Steffen Schütze

Mexico, CDMX **February 2020**

Resumen

En muchos problemas de la vida real nos enfrentamos con el problema de optimizar varios objetivos al mismo tiempo. Por ejemplo, en diseño de producción los principales objetivos serían maximizar la calidad mientras se minimiza el costo de un producto dado. Normalmente, los objetivos considerados están en conflicto con los demás y por lo general no existe una única solución óptima. Por el contrario, podemos esperar un conjunto completo de soluciones optimas, el llamado frente de Pareto, y su imagen, el conjunto de Pareto. Problemas de este tipo son conocidos como problemas de optimización multiobjetivo o MOPs, por sus siglas en inglés. Existen diversas maneras de resolver MOPs, en este trabajo nos enfocamos en las técnicas de programación matemática, especialmente en métodos de continuación particulares, los cuales realizan una búsqueda a lo largo del conjunto de Pareto de un problema dado.

En esta tesis, extendemos el algoritmo *Pareto Tracer* (PT), un método de Optimización multiobjetivo recientemente propuesto, para el manejo eficiente de MOPs con restricciones generales de desigualdad. El algoritmo presenta una mejora sobre el algoritmo base ya que la versión original del algoritmo PT únicamente puede manejar restricciones de igualdad y restricciones de caja. Nuestro algoritmo permite aproximar localmente los conjuntos de soluciones de un MOP dado, sin importar el número de objetivos, variables, o el tipo de restricciones. Presentamos resultados de algunos problemas académicos de referencia y comparamos nuestro algoritmo con otros del estado del arte indicando la eficiencia de nuestro algoritmo. Nuestro principal objetivo es adaptar el método Pareto Tracer (PT) existente [36] para que sea capaz de manejar de manera confiable restricciones de desigualdad. PT es un método predictor corrector innovador (los detalles matemáticos del algoritmo se revisarán más adelante) que es capaz de manejar restricciones de desigualdad y de caja.

Abstract

In many real-world problems we are faced with the problem that several objectives have to be optimized at the same time. For example, in product design the principal goals would be to maximize the quality while minimizing the cost of the given product. Normally, the considered objectives are in conflict with each other and there is typically not one single optimal solution. Instead, we can expect an entire set of optimal solutions, the so-called Pareto set, and its image, the Pareto front. Problems of this kind are known as multi-objective optimization problems (MOPs). There are several ways to solve MOPs, in this work we focus on mathematical programming (MP) techniques, specially on particular continuation methods which perform a search along the Pareto set of a given problem.

In this thesis, we extend the *Pareto Tracer* (PT), a recently proposed multi-objective optimization method, for the efficient treatment of general inequality constrained MOPs. The algorithm presents an improvement over its base algorithm since the original version of the PT can only handle equality and box constraints. Our algorithm allows to locally approximate the solution sets of a given MOP with no restrictions on the number of objectives, variables, and the kind of the constraints. We present results on some academic benchmark problems and compare our algorithms to the state-of-the-art indicating the strength of our method. Our main goal is to adapt the existing Pareto Tracer (PT) method [36] so that it can reliably handle inequality constraints. PT is a novel predictor corrector method (mathematical details of the algorithm shall be reviewed before) that can handle equality and box constraints.

A mis padres y hermana, lo logramos...

Agradecimientos

Agradezco a Dios, porque nunca me abandona.

Agradezco a mi asesor de tesis, el Dr. Oliver Schütze, por su dedicación, el tiempo y los conocimientos que aportó a mi trabajo. Pero especialmente le agradezco por continuar conmigo hasta el final, por su paciencia, su tolerancia y su apoyo.

Agradezco especialmente a mis padres Carmen y Ricardo, por darme su apoyo y su amor incondicional, por enseñarme a no rendirme y a superarme cada día, los amo infinitamente.

Agradezco a mi hermana Catalina, por ser mi mejor amiga y mi compañera de vida, pero más aún, por ser un gran ejemplo a seguir.

A mis abuelos, Andy y Gela por cuidar de mi y consentirme en esta estapa tan difícil. A mi tía Angeles por abrirme las puertas de su casa y brindarme su amistad. A mis tíos Armando y Laura por todo el cariño y los buenos momentos que siempre me brindan. A mi prima Mariana, por ser una amiga para mi, por todos los momentos y locuras que hemos compartido juntas.

Agradezco a mis amigos Michel, Amin e Isaac, por compartir esta etapa conmigo, por estar en los buenos momentos y también en los malos, por su amistad y su apoyo, por hacer de esta una etapa inolvidable.

Agradezco a todos mis amigos por estar siempre a mi lado, en especial en esta etapa.

Adrezco al Dr. Luis Gerardo de la Fraga y al Dr. Amilcar Meneses por tomarse el tiempo de revisar este trabajo y enriqueserlo con sus comentarios.

Agradezco al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico que me brindó durante los dos años de la maestría.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

BOP bi-objective optimization problem. 2, 24–26, 45, 50, 58, 65

CHIM convex hull of individual minima. 13

DD descent direction. 15

FPO first Pareto optimal. 26, 27

IFT Implicit Theorem Function. 22, 23

IVP initial value problem. 19

KKT Karush-Kuhn-Tucker. 10, 16, 17, 23, 24, 29–31

MOEA multi-objective evolutionary algorithm. 1, 11, 45, 76

MOO multi-objective optimization. 1

MOP multi-objective optimization problem. v, vii, xv, 1–3, 5–8, 10–13, 15, 19, 21–25, 29–31, 33, 35, 36, 40, 42, 45, 72, 75

MP mathematical programming. vii, 1–3, 5, 11, 33

NBI Normal Boundary Intersection. 13, 14, 45–47, 50, 53, 54, 57, 58, 61, 65, 68, 69, 72

NSGA-II Nondominated Sorting Genetic Algorithm. xvii, 11, 45–58, 60, 61, 65, 67–70, 72

PC predictor-corrector. 23, 29

PF Pareto front. vii, xv, 1, 8–13, 25–27

PS Pareto set. vii, xv, 1, 2, 8, 9, 11, 12, 14, 22, 25

PT Pareto Tracer. v, vii, xv, 2, 29, 30, 33, 35, 36, 38–43, 45–47, 50, 53, 54, 57, 58, 61, 65, 68, 69, 72, 75, 76

SOP single-objective problem. 2, 12

WS weithed sum. xv, 12, 13

Chapter 1

Introduction

Optimization [37] is an important tool in Mathematics and Computer Science that requires to identify some objective, a quantitative measure of the performance of the system under study. This objective depends on certain characteristics called variables. The main goal in optimization is to find values of the variables, which improve somehow the value of the objective function. Often the variables are restricted, or constrained, in some way.

In multi-objective optimization (MOO) we simultaneously deal with several objective functions at the same time. Usually, those functions are in conflict, for example, two common goals in product design are to maximize the quality while to minimize the cost of the product. Therefore, we need to define an optimal solution according to more than one objective. For that purpose we use the notation of dominance, which states that a solution is better than another one if at least one of its objectives is improved while the others do not get worse.

The presence of multiple objectives in a problem, in principle, gives rise to a set of optimal solutions, instead of a single optimal solution. This demands a user to find as many solutions as possible. The knowledge of the solution set is essential for the decision maker to envision the optimal problem resolution. A lot of methods for the treatment of multi-objective optimization problems (MOPs) have been developed. Those methods compute the approximations of the solution set, called the Pareto set (PS), and its image, the Pareto front (PF). Two known classes of techniques for solving MOPs are mathematical programming (MP) techniques and multi-objective evolutionary algorithms (MOEAs). MP techniques are fast and can handle constraints, but they are of local nature, which means, they might get stuck in local minima. Set based stochastic search methods, such as the MOEAs are of global nature, ask for low requirements on the model and are universally applicable, but they need quite a few computational resources leading to slow converge rates.

1.1 Motivation

In particular, MP techniques are divided in three categories, according to the strategies that they use to solve the problems. Scalarization methods, for instance, suggest converting the MOP into a single-objective problem (SOP) by emphasizing one particular Pareto optimal solution at a time. Another group encloses the descent direction methods, that are generally fast local convergent algorithms focused in finding only one optimal point. Finally, continuations methods perform a search along the PS an are very efficient if one (or more) solutions is at hand.

Our main motivation rests on existing continuation methods, this is because most of the current ones are limited to cope just with equality constraints, that is the case of the algorithm proposed by Hillermier [29] and the Pareto Tracer algorithm [36]. Few of them can also handle inequality constrains [35, 39] but they are restricted to bi-objective optimization problems (BOPs).

1.2 Problem

In many real life situations one is faced with the problem that several objectives have to be optimized currently, resulting in a MOP. Sometimes there exist restrictions on the variables or the objective functions involved into the MOP. As stated above, there exist alternatives to solve them and we are interesting on MP techniques, specially on continuation methods. Although those are a good alternative, they are limited in the constraints and the number of objectives they can deal with.

In this thesis work we focus on MOPs with inequality constraints. We pretend to give the state of the art for a new algorithm able to handle with every class of constraints no matter the number of variables nor objectives of the problem.

1.3 Aims of the Thesis

Pareto Tracer (PT) [36] is a novel predictor corrector method (mathematical details of the algorithm shall be reviewed in the next chapter) that can handle equality and box constraints. Our main goal is to adapt the existing PT method so that it can reliably handle inequality constraints.

1.4 Final Contribution

This thesis contributes to the area of multi-objective optimization, specifically we introduce a variation of a existing continuation method that improves the original one, extending its scope into general inequality constrained MOPs. The method presented in this thesis work is applicable to any MOP, with no restrictions in the class or the number of the constraints nor the number of variables or objective functions. As result of this thesis work, the following conference paper has been pusblished "A new hybrid metaheuristic for equality constrained bi-objective optimization problems" [12].

1.5 Organization of the Thesis

This thesis consists of five chapters including this introductory chapter. The remainder of this work is organized as follows. Chapter 2 presents an extended theoretical background, we describe MOPs and review the related work, it is, the algorithms developed to solve them, paying special attention to MP techniques. The proposed algorithm is described in detail in Chapter 3, we begin showing the needed of adapt the existing Pareto Tracer algorithm, in order to can handle inequality constrained MOPs. Modified algorithm is further on given. In Chapter 4 we present numerical results on theoretical MOPs, later in this chapter, we compare our algorithm with some existing ones. Finally, in Chaper 5 we present our conclusions and possible paths for future work.

Chapter 2

Background

In this chapter we give a general background introducing the concepts required to understand the work developed in this thesis. Here we focus on multi-objective optimization problems (MOPs) and several ways that exist nowadays to solve them. The chapter is divided into two parts: in the first one we give a formal definition of a MOP and describe the optimality conditions that we use throughout this work. In the second part we address mathematical programming (MP) techniques, a category of methods for solving MOPs; we present and describe the commonly used algorithms that belong to this category and that are related to this work. Finally, we discuss the advantages and disadvantages of these methods and the need to develop new ones.

2.1 Multi-objective Optimization

In multi-objective optimization [4,29] several objective functions have to be optimized at the same time. For example, consider that we want to buy a motorcycle and we are interested in minimizing the cost while maximizing the speed. We represent this example in Figure 2.1. If the only objective of this problem was to minimize the cost, the optimal solution would be motorcycle A. On the other hand, if speed was the only objective, the optimal solution would be motorcycle B. Since our goal is satisfy both objectives, they can not be considered as two independent optimization problems, whose results are the two extreme solutions discussed above. There are other solutions between these two extreme points, which have different speeds and costs. Between any two of these, one is better in terms of one objective, but it is at expense of a sacrifice on the other objective. Thus, all the solutions, shown between A and B are also optimal.

In this work we consider MOPs that can be mathematically expressed as follows:

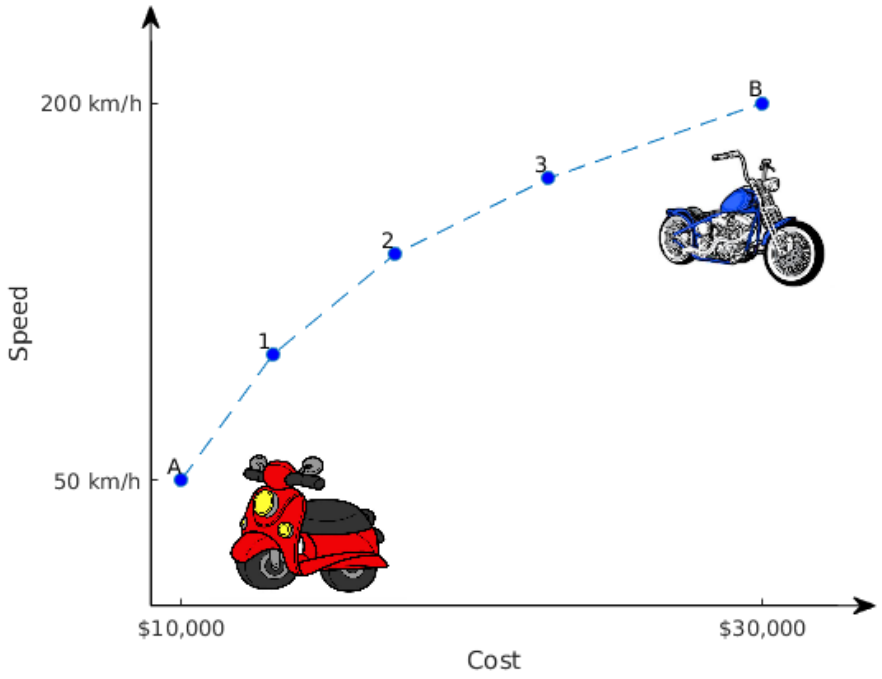


Figure 2.1: Hypothetical example of a MOP.

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} F(x), \\ &\text{s.t.} \quad g_i(x) \leq 0, \quad i = 1, \ldots, m \\ &\qquad h_j(x) = 0, \quad j = 1, \ldots, p, \end{aligned} \tag{2.1}$$

where the function $F(x) : \mathbb{R}^n \to \mathbb{R}^k$ represents the vector of k objective functions $F(x) = (f_1(x), \ldots, f_k(x))^T$, $f_i(x) : \mathbb{R}^n \to \mathbb{R}$, $x \in \mathbb{R}^n$ is the vector of n decision variables $x = (x_1, \ldots, x_n)^T$, $G(x) : \mathbb{R}^n \to \mathbb{R}^m$ and $H(x) : \mathbb{R}^n \to \mathbb{R}^p$ are inequality and equality constraints, respectively, with $G(x) = (g_1(x), \ldots, g_m(x))^T$, $g_i : \mathbb{R}^n \to \mathbb{R}$, $H(x) = (h_1(x), \ldots, h_p(x))^T$, $h_i : \mathbb{R}^n \to \mathbb{R}$. In this work the objective functions and the constraints are assumed to be continuously differentiable.

There is a special class of inequality constraints called *box constraints*, which are of the form $a_i \leq x_i \leq b_i$, where $a, b \in \mathbb{R}^n$ and set limits for each component of the variables vector x.

Depending on the type of the variables and the mathematical nature of the objective functions and the constraints, the MOPs are classified in different manners. If the variables involved are continuous and both the objective function and the constraints are linear, the problem is denominated linear programming problem. If any of the variables involved is integer or binary, while the constraints and the objective function are both linear, the problem is denominated mixed-integer linear programming

problem. Analogously, if the objective function or any constraint is nonlinear and all variables are continuous, the problem is denominated nonlinear programming problem. If additionally any variable is integer the corresponding problem is denominated mixed-integer nonlinear programming problem [10].

Definition 2.1. *The feasible region in decision space of a MOP is defined by*

$$P = \{x \in \mathbb{R}^n \mid g(x) \leq 0 \text{ and } h(x) = 0\}, \tag{2.2}$$

where $g : \mathbb{R}^n \to \mathbb{R}^m$ *and* $h : \mathbb{R}^n \to \mathbb{R}^p$ *are defined as the vector functions of the inequalities and equalities, respectively.*

As we mentioned before, in multi-objective optimization we consider more than one objective. Usually, objective functions are in conflict. For example, as we can see in Figure 2.2 while we minimize one of the two objective functions, the other one increases its value and vice versa. For this reason, no single point will minimize all the objective functions at once, but rather there is an entire set of optimal configurations. The optimality of a MOP is defined by the concept of dominance.

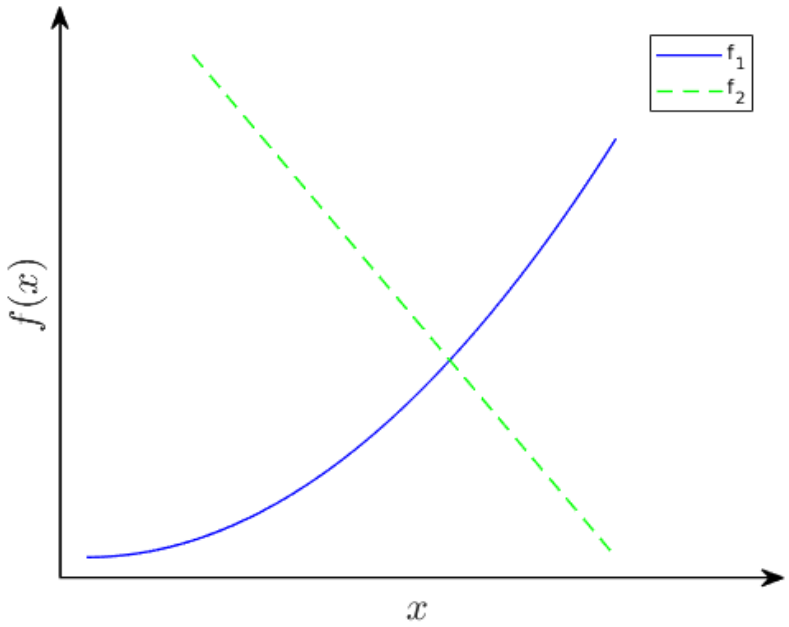


Figure 2.2: Conflicting functions, every function has a different behavior with respect to x.

In the following we give the definition of dominance with respect to (2.1).

Definition 2.2. *A point* $y \in \mathbb{R}^n$ *is dominated by a point* $x \in \mathbb{R}^n$ $(x \prec y)$ *with respect to a MOP if*

$$f_i(x) \leq f_i(y) \quad \forall i = 1, ..., k$$

and

$$f_j(x) < f_j(y)$$

for at least one index $j \in \{1, ..., k\}$. *Else,* y *is called non-dominated by* x.

Definition 2.3. *Pareto set and Pareto front.*

a) A point $x \in \mathbb{R}^n$ *is called Pareto optimal if there is no* $y \in \mathbb{R}^n$ *such that* $y \prec x$.

b) The set of optimal configurations for a MOP

$$PS = \{x \in P \mid \nexists y \in P : y \prec x\},$$

is called the Pareto set (PS).

c) The image $F(PS)$ *of the PS is called the Pareto front (PF).*

In general and under certain mild assumptions on the MOP, PS and PF form a $(k-1)$ dimensional objects [29].

Now we present an example of the PF and the PS of a MOP. Let's consider the MOP described in (2.3).

$$\begin{aligned} \min & \begin{cases} f_1(x) = x_1, \\ f_2(x) = x_2, \end{cases} \\ \text{s.t.} \quad & x_1^2 + x_2^2 - 0.1 - 1\cos\left(9\arctan\frac{x_1}{x_2}\right) \geq 0, \\ & (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.55, \\ & 0 \leq x_1 \leq \pi, \\ & 0 \leq x_2 \leq \pi. \end{aligned} \tag{2.3}$$

In Figure 2.3 it is represented the feasible region for (2.3), it corresponds to the area limited by the curves, which represent the inequality constraints. Figure 2.1 shows the PS and the PF of the MOP (for this particular case the PS and the PF of the problem are the same because of the problem definition).

A natural question, that arises from the fact that typically the solution set of a MOP is not given by a single point, is how to measure the performance of and algorithm for solving MOPs, aiming for the approximation of the entire PS and PF. One way

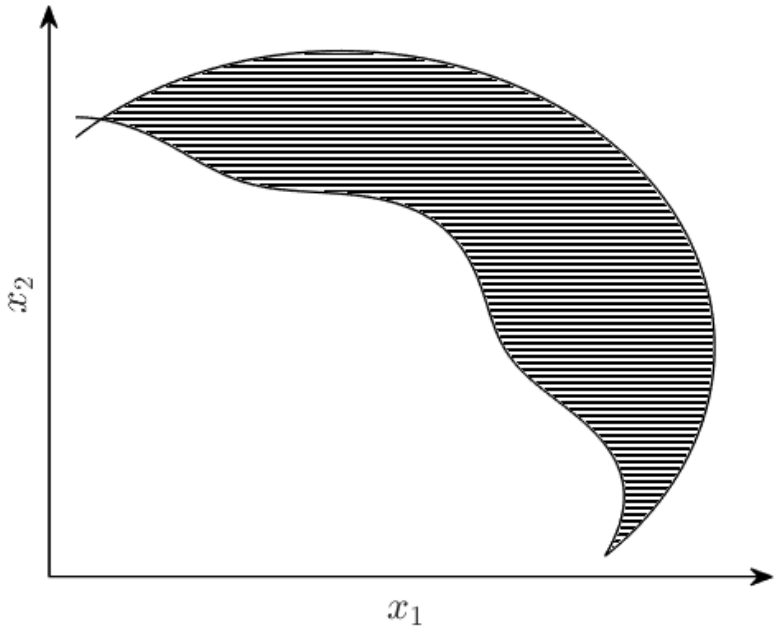
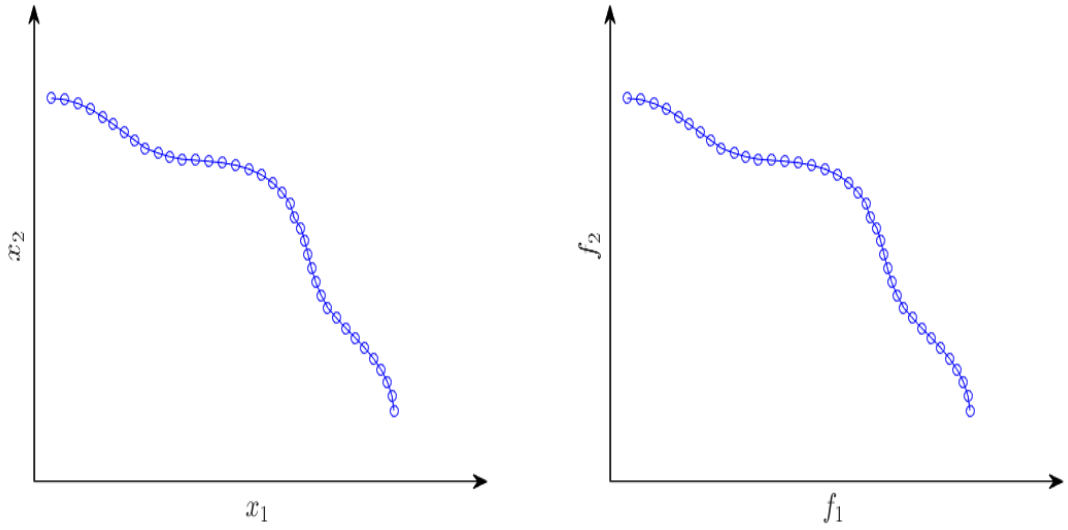


Figure 2.3: Feasible region of (2.3).



(a) Pareto set **(b)** Pareto front

Figure 2.4: PS and PF representation.

to do this is to measure the distance of the outcome set of the algorithm to the set of interest. One such distance function is the Hausdorff distance d_H. [28]. There also exits another another performance indicator called Δ_p [3], which can be viewed as an 'averaged Hausdorff distance' between the outcome set and the PF.

A first-order condition of optimallity for differentiable MOPs is given by the *Karush-Kuhn-Tucker* (KKT) conditions [31, 32]. Those are only necessary conditions, it is, the points that satisfy these requirements are just candidates for being a local minima. In (2.4) we present the KKT conditions for general MOPs.

Theorem 2.1. *Let f_i, g_i and h_i be continuously differentiable. If $x \in \mathbb{R}^n$ is optimal, then there exist vectors $\lambda \in \mathbb{R}^k$, $\gamma \in \mathbb{R}^m$ and $\alpha \in \mathbb{R}^p$ s.t.*

KKT for constrained MOPs
Let f_i, g_i and h_i be continuously differentiable. If $x \in \mathbb{R}^n$ is optimal, then there exist vectors $\lambda \in \mathbb{R}^k$, $\gamma \in \mathbb{R}^m$ and $\alpha \in \mathbb{R}^p$ s.t.

$$\begin{aligned}
\sum_{i=1}^{k} \lambda_i \nabla f_i(x) + \sum_{i=1}^{m} \gamma_i \nabla g_i(x) + \sum_{i=1}^{p} \alpha_i \nabla h_i(x) = 0, & \\
g_i(x) \leq 0, \quad & i = 1, \ldots, m, \\
h_i = 0, \quad & i = 1, \ldots, p, \\
\sum_{i=1}^{k} \lambda_i = 1, & \\
\lambda_i \geq 0, \quad & i = 1, \ldots, k, \\
\gamma_i \geq 0, \quad & i = 1, \ldots, m, \\
\gamma_i g_i(x) = 0, \quad & j = 1, \ldots, m.
\end{aligned} \tag{2.4}$$

Points x that satisfy (2.4) are called KKT points.

In case the MOP contains no constraints, the KKT conditions significantly reduce as we can see in (2.5).

Theorem 2.2. *KKT for unconstrained MOPs*
Let the MOP be unsconstrained and all the objectives be continuously differentiable. If x is an optimal point, then there exist scalars $\lambda_1 \ldots \lambda_k$ s.t.

$$\begin{aligned} &\sum_{i=1}^{k} \lambda_i = 1, \\ &\sum_{i=1}^{k} \lambda_i \nabla f_i(x) = 0, \\ &\lambda_i \geq 0. \end{aligned} \tag{2.5}$$

Two known classes of techniques for solving MOPs are mathematical programming (MP) techniques and multi-objective evolutionary algorithms (MOEAs).

MOEAs [9] are stochastic search algorithms which are very popular due to their global set base approach. They use a set (called population) of solutions. These solutions "evolve" through special operators based on the species evolutionary process, some examples of this operators are the mutation and the crossover. This evolutionary process is expected to lead the individuals covering all the PF. These methods ask for low requiremts on the model and are universally applicable, but they need quite a few resources leading to slow converge rates. One of the most famous algorithm of this category is the Nondominated Sorting Genetic Algorithm (NSGA-II) [15], which was proposed by Kalyanmoy Deb et al. in 2002. Following we briefly describe the algorithm as we will use it later on in order to make some comparisons with the algorithm presented in this work.

MOEAs that use non-dominated sorting and sharing have been criticized mainly for: their computational complexity, their non-elitism approach and the need to specify a sharing parameter. NSGA-II is a non-dominated sorting-based MOEA which alleviates all of the above three difficulties. Specifically, a selection operator is presented that creates a mating pool by combining the parent and offspring populations.

On the other hand, MP techniques are based on classical optimization methods, such as the Newton method. Those techniques are fast but also are of local nature, that is, they can get easily trapped in local minima and for this reason a starting point is needed in order to obtain the PS of a MOP. In next section we describe in detail this category.

2.2 Mathematical Programming Techniques

In this section we shortly describe some of the existing mathematical programming (MP) techniques. We consider three different categories that are described hereupon,

discussing about its advantages and disadvantages. Also we introduce some of the most representative methods of every category.

2.2.1 Scalarization Methods

Methods of this kind transform the MOP into a single-objective or scalar optimization problem (SOP) [25, 44]. They formulate a SOP such that optimal solutions of it are Pareto optimal solutions to the MOP. Doing so, they reduce the problem into an easier one, but only one solution is given in one single run of the algorithm and not the entire PS. Scalarization methods are sometimes used to approximate the entire PS by considering a sequence of SOPs.

In this approach, the parameters are not known in advance and the decision maker has to choose them. For some problems, this choice can be problematic. There are several scalarization methods [8, 20, 22], below we describe three of most applied ones.

Weighted Sum (WS) Method

This method was suggested by Gass and Staaty in 1995 [24, 34]. The idea is to associate each objective with a weighting coefficient and to minimize the WS of the objectives. The weighted sum method reads as follows:

$$\min_x f_w(x) = \sum_{i=1}^{k} w_i f_i(x), \qquad \text{where} \quad w_i \geq 0, \quad \sum_{i=1}^{k} w_i = 1.$$

However, by modifying the weights, different points on the PS can be found, but varying the weights systematically may not necessarily result in an even distribution of the points along the PS. For this reason, in some cases, a complete representation of the PS will not be achieved. By using this method it is impossible to obtain points on non-convex portions of the PS, it is the case in Figure 2.5. This is a serious and significant limitation because in the real world, if this method is used for non-convex problems, therefore, the PF generated will be incomplete, leaving out a set of solutions.

Normal Boundary Intersection (NBI)

This method was developed by Das and Dennis in 1996 [13]. It is a scalarization strategy specially designed for the generation of even distributed optimal solutions in

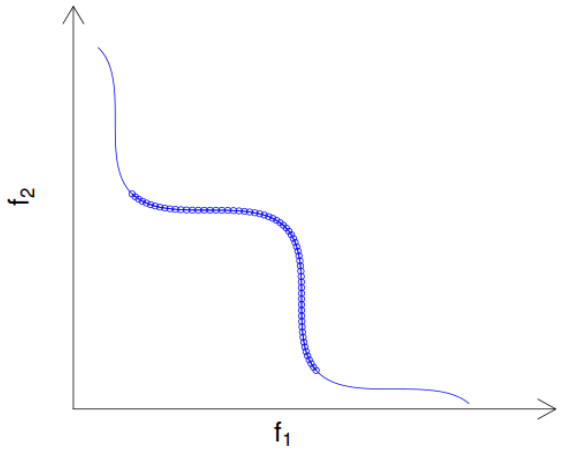


Figure 2.5: Drawbak of the WS method: the non-convex part of the PF, represented by points is not going to be achieved by the method.

the objective space. The first step is to compute an approximation of the convex hull of individual minima (CHIM). This is done via minimizing each objective function $f_i, i = 1, ..., k$.

Given a MOP, let $F_i^* = f(x_i^*)$ and x_i^* be the respective global minimizers of $f_i(x)$, $i = 1, \dots k$ and the vector $F^* = (f_1^*, ..., f_k^*)^T$ is referred to as the shadow minimum where $f_j^* = f_j(x_j^*)$ for $j = 1, .., k$. $\Phi \in \mathbb{R}^{k \times k}$ is a matrix called the pay-off matrix, whose ith column is given by $F_i^* - F^*$. Then the CHIM can be represented as the set of points in $\mathbb{R}^k$ that are convex combinations of $F_i^* - F^*$, i.e.

$$\left\{ \Phi\beta \;\middle|\; \beta \in \mathbb{R}^k, \sum_{i=1}^{k} \beta_i = 1, \beta_i \geq 0, , i = 1, 2, ..., k \right\}. \tag{2.6}$$

The second step is to solve the NBI-subproblem, described in (2.7), for selected values of $\beta \in \mathbb{R}^k$.

$$\max_{(x,t) \in \mathbb{R}^n \times \mathbb{R}} t, \tag{2.7}$$

$$\begin{aligned} \text{s.t.} \quad \Phi\beta + t\hat{n} &= F(x), \\ h_i(x) &= 0, \quad i = 1, ..., m, \\ g_i(x) &\leq 0, \quad i = 1, ..., p, \\ a_i \leq x &\leq b_i, \quad i = 1, ..., n. \end{aligned}$$

The idea is to generate a subset of the efficient set by solving the NBI-subproblem for various β .

One weak point of the NBI method is that for highly nonlinear problems, it is hard to obtain optimal solutions due to the indirect equality constraints. Further, NBI is not applicable to MOPs with $p \geq 4$. An advantage of this method is that there is not problem with concave and convex portions of the PS, using it we can find everything.

ε -constraint Method

This method was first proposed by Haimes et al. in 1971 [27]. It suggests to reformulate the MOP by just keeping one of the objectives and restricting the rest of the objectives within the user's specified values. Systematic modification of the values of the objective functions forming the additional constraints leads to the generation of an even distributed PS. The problem can be mathematically expressed as follows:

$$\min_x f_i(x)$$

$$\begin{aligned} \text{s.t.} \quad & f_j(x) \leq \varepsilon_j, \quad j \in \{1, ..., k\} \setminus \{i\} \\ & h_i(x) = 0 \quad i = 1, ..., m \\ & g_i(x) \leq 0 \quad i = 1, ..., p. \end{aligned}$$

The approach of this method is to minimize one objective, say $f_i(x)$, subject to the additional constraints $f_j(x) \leq \varepsilon_j, j = 1, ..., p, j \neq i$, and some $\varepsilon > 0$, where ε represents the "worst" value that f_j is allowed to take. It has been shown that if the solution of this method is unique, then it is efficient. Two issues with this approach are (i) that it is necessary to preselect which objective to minimize and (ii) the proper choice of the ε. Also, the algorithm has troubles with flat search regions.

There are some variations of this method that try to improve it, for example, the elastic constraint method [18, 19] is a modification that gives conditions on the characterization for efficient and properly efficient solutions and the special case given in [5], in which the epsilon is replaced by a (more informed) quantity related to the structure of the problem.

2.2.2 Descent Direction Methods

A descent direction (DD) is a direction in which all objectives decrease for sufficiently small step size. If all objectives are differentiable in x, then a DD, let's call it ν, can be characterized as

$$\langle \nabla f_i(x), \nu \rangle < 0, \qquad i = 1, ..., k. \tag{2.8}$$

In contrast to standard scalarization approaches for multiobjective optimization, in this methods we do not transform the problem at hand into a parameterized scalar optimization problem before settling to solve the transformed problem, instead, a line search is performed according to a DD.

Almost all methods that belong to this category have its own strategy for selecting the step length, the search direction and the stopping criteria. Below we describe three such methods.

Newton Method

It is an extension of Newton's method for unconstrained multi-objective optimization proposed by Fliege et al. in 2000 [21]. At each iterate the Newton's direction is obtained by minimizing the max-ordering scalarization of the variations on the quadratic approximations of the objective functions. The objective functions are assumed to be twice continuously differentiable and locally strongly convex.

In each iteration, the algorithm replaces each objective function with a quadratic model for it, as in the "clasical" Newton method. With these local models at hand, the max-scalarization of the *variations* of the quadratic approximations is minimized and a joint descent direction is obtained.

Now it is presented the definition of Newton direction for a MOP. As in the classical one-criterion case, the Newton direction will be a solution to a suitably defined problem involving quadratic approximations of the objective functions f_j. Moreover, again as in the scalar case, in a critical point, the Newton step will be $0 \in \mathbb{R}^n$.

For $x \in \mathbb{R}^n$, the *Newton direction* $s(x)$ at x is defined as the optimal solution of

$$\begin{cases} \min \max\limits_{j=1,...,k} \nabla f_j(x)^T s + \dfrac{1}{2} s^T \nabla^2 f_j(x) s, \\ \text{s.t. } s \in \mathbb{R}^n. \end{cases} \tag{2.9}$$

In order to find a direction of descent for all objective functions involved, it is necessary to solve a scalar optimization problem.

The optimal value of problem (2.9) will be denoted by $\theta(x)$. Hence,

$$\theta(x) = \inf_{s\in\mathbb{R}^n} \max_{j=1,\ldots,k} \nabla f_j(x)^T s + \frac{1}{2} s^T \nabla^2 f_j(x) s, \tag{2.10}$$

and

$$s(x) = arg \min_{s\in\mathbb{R}^n} \max_{j=1,\ldots,k} \nabla f_j(x)^T s + \frac{1}{2} s^T \nabla^2 f_j(x) s. \tag{2.11}$$

The problem (2.9) is equivalent to

$$\begin{cases} \min g(t,s) = t, \\ \text{s.t. } \nabla f_j(x)^T s + \frac{1}{2} s^T \nabla^2 f_j(x) s - t \leq 0 \quad (1 \leq j \leq k), \\ \quad (t,s) \in \mathbb{R} \times \mathbb{R}^n. \end{cases} \tag{2.12}$$

The Lagrangian of this problem is

$$L((t,s),\lambda) = t + \sum_{j=1}^{k} \lambda_j \left(\nabla f_j(x)^T s + \frac{1}{2} s^T \nabla^2 f_j(x) s - t \right). \tag{2.13}$$

Direct calculation of the Karush-Kuhn-Tucker conditions yields

$$\sum_{j=1}^{k} \lambda_j = 1, \quad \sum_{j=1}^{k} \lambda_j \left(\nabla T(x) + \nabla^2 f_j(x) s \right) = 0, \tag{2.14}$$

$$\lambda_i \geq 0, \quad \nabla f_j(x)^T s + \frac{1}{2} s^T \nabla^2 f_j(x) s \leq t \quad (1 \leq j \leq k), \tag{2.15}$$

$$\lambda_j \left(\nabla f_j(x)^T s + \frac{1}{2} s^T \nabla^2 f_j(x) s - t \right) = 0 \quad (1 \leq j \leq k). \tag{2.16}$$

Problem (2.12) has a unique solution, $(\theta(x), s(x))$. As this is a convex problem and has a Slater point, there exist a KKT multiplier $\lambda = \lambda(x)$, which, together with

$s = s(x)$ and $t = \theta(x)$, satisfies conditions (2.14)-(2.16). In particular, from (2.14) one obtains

$$s(x) = -\left[\sum_{j=1}^{k} \lambda_j(x)\nabla^2 f_j(x)\right]^{-1} \sum_{j=1}^{k} \lambda_j(x)\nabla f_j(x). \tag{2.17}$$

Hence, the Newton direction defined in this algorithm is a Newton direction for a standard scalar optimization problem, implicity induced by weighting the given objective functions by the a priori unknown KKT multipliers.

Now we are going to talk about the step size control using the Armijo rule. For the scalar case $(k = 1)F : U \to \mathbb{R}$, at a nonstationary point $x \in U$, the classical Armijo rule for the Newton search direction $s(x)$ can be read as follows

$$F(x + ts(x)) \leq F(x) + \beta t s(x)^T \nabla F(x), \tag{2.18}$$

with $\beta \in (0, 1)$. To accept a full Newton step close to a local optima where $\nabla^2 F > 0$ (it is, the Hessian of F is positive definite), one must choose $\beta \in (0, 1/2)$ [17]. Note that in this setting $(k = 1)$,

$$\theta(x) = \frac{1}{2} s(x)^T \nabla F(x). \tag{2.19}$$

So, we can rewrite the Armijo rule as

$$F(x + ts(x)) \leq F(x) + \sigma t \theta(x), \tag{2.20}$$

with the choice $\sigma = 2\beta \in (0, 1)$ allowing full Newton steps to be accepted close to a local optimum where $\nabla^2 F > 0$. The above inequality, interpreted componentwise, will be our criterion for accepting a step in the multiobjective Newton direction.

In algorithm 1 it is sketched the Newton algorithm for multicriteria optimization. At each step, at a nonstationary point, it is minimized the maximum of all local models as in (2.9) to obtain the Newton step (2.11), wich is a descent direction. Under suitable local assumptions, full Newton steps are always accepted and the generated sequence converges superlinear (or quadratically) to a local solution.

On the advantage we can remark that problems whose objective functions display a weak or moderate amount of curvature do not represent a challenge for the algo-

Algorithm 1 Newton algorithm for multicriteria optimization

Require: $x_0 \in U, 0 \leq \sigma < 1$

1: set $k = 0$
2: define $T = \{1/2^n | n = 0, 1, 2, ...\}$
3: **while** true **do**
4: Solve the direction search problem (2.9) to obtain $s(x_k)$ and $\theta(x_k)$ as in (2.11) and (2.10)
5: **if** $\theta(x_k) = 0$ **then**
6: Stop
7: **else**
8: Go to 10
9: **end if**
10: (*Line search*) Choose t_k as the largest $t \in T$ such that

$$x_k + ts(x_k) \in U,$$
$$f_j(x_k + ts(x_k)) \leq f_j(x_k) + \sigma t\alpha(x_k),\ j = 1, ..., m.$$

11: (*Update*) Define $x_{k+1} = x_k + t_k s(x_k)$ and set $k = k + 1$. Go to 4.
12: **end while**

rithm, also it works fine in the non-convex case. Although, this algorithm experiences difficulties when employed on problems exhibiting a high degree of curvature.

The Direct Search Descent Method

This is an iterative method based on the idea of steering the search along a predefined direction in objective space, which has the advantage that has a physical meaning and the search can be steered according to the given situation [41]. This method allows to find a (descent) direction $\nu = \nu(\alpha) \in \mathbb{R}^n$ for every search direction $\alpha \in \mathbb{R}^k$.

Let $x_0 \in \mathbb{R}^n$ be a given point and the vector $d \in \mathbb{R}^k$ presents a desired search direction in image space. More precisely, a search direction $\nu \in \mathbb{R}^n$ in parameter space is sought such that for $x_1 = x_0 + t\nu$, where $t \in \mathbb{R}_+$ is the step size, it holds:

$$\lim_{t \to 0} \frac{f_i(x_1) - f_i(x_0)}{t} = \quad \langle \nabla f_i(x_0), \nu \rangle = d_i, \quad i = 1, ..., k. \tag{2.21}$$

Using the Jacobian of F, we can write (2.21) in matrix notation as

$$\nabla F(x_0)\nu = d. \tag{2.22}$$

The solution of (2.22) is a descent direction provided that all components of d are negative. Typically the number of variables is higher that the number of objectives

in a given MOP ($n >> k$), so system (2.22) is (probably highly) underdeterminated, implying that its solution is not unique. One choice is take the solution with the lowest norm leading to

$$\nu_+ = \nabla F(x_0)^+ d, \tag{2.23}$$

where $\nabla F(x_0)^+$ denotes the pseudo inverse of $\nabla F(x_0)$ [1].

Note that (2.23) is the solution of (2.22) with the smallest Euclidean norm. Hence, given a small step size t, one expects for a step in direction ν_+ (decision space) the largest progress in $d-$direction (objective space). Also, the trajectory followed by this procedure is identical to the solution curve of the subsequent initial value problem (IVP):

$$\begin{aligned} x(0) &= x_0 \in \mathbb{R}^n, \quad t = 0, \\ \dot{x}(t) &= \nu_+(x(t)), \quad t > 0, \end{aligned} \tag{2.24}$$

whose limit point x^* is also a critical point of the considered MOP. Talking about the step size, the rank of the matrix $rank(\nabla F(x))$ can of course not be used to detect the endpoint of the curve numerically, but instead the condition k_2 number can be used as follows

$$k_2 = \sqrt{\frac{\rho_1}{\rho_2}} \geq \epsilon, \tag{2.25}$$

where ρ_1, ρ_2 are the largest and smallest singular value of $\nabla F(x)$, respectively; and ϵ is a given large threshold since we may expect than $k_2 \to \infty$ as $x(t) \to x^*$. Algorithm 2 describes a possible algorithm to trace the solution curve of IVP.

Steepest Descent Method

There exist a steepest descent method for unconstrained MOPs and a "feasible descent direction" method for the constrained case [23]. Here we present the first one. This method does not scalarize the original problem and neither ordering information nor weighting factors for the different objective functions are assumed to be known.

Let $range(A)$ denotes the range of the linear mapping given by the matrix A. A necessary condition for a point $x \in \mathbb{R}^n$ to be locally Pareto optimal is

[1] In the case the rank of $\nabla F(x_0) = J$ is maximal, the pseudo inverse is given by $J^+ = J^T(JJ^T)^{-1}$.

Algorithm 2 Directed search method

Require: Starting point $x_0 \in \mathbb{R}^n$ with $rank(\nabla F(x_0)) = k$, $tol \in \mathbb{R}_+$, convex weight $\alpha_0 \in \mathbb{R}^k$.

1: $i = 0$
2: **while** $k_2(\nabla F(x_i)) < tol$ **do**
3: compute $\nu_i = -\nabla F(x_i)^+ \alpha_i$
4: compute $t_i \in \mathbb{R}_+$
5: set $x_{i+1} = x_i + t_i \nu_i$
6: choose $\alpha_{i+1} \in \mathbb{R}^k$
7: set $i = i + 1$
8: **end while**

$$\text{range } (\nabla F(x) \cap (-\mathbb{R} + +))^k = \emptyset, \tag{2.26}$$

The idea of the general algorithm is the following: choose an x and check if (2.2.2) holds. If not, compute a direction ν and make a step with a suitably chosen step length from x along ν. This let us a new point, and the scheme can be repeat.

Now the process of computing the search direction is described. Suppose that it is given a point $x \in \mathbb{R}^n$. Define $A = \nabla F(x)$ and the function $f_x : \mathbb{R}^n \to \mathbb{R}^k$ by $f_x(\nu) = \max\{(A\nu)_i | i = 1, ..., k\}$.

Consider the unconstrained minimization problem

$$\begin{aligned} &\min f_x(\nu) + \frac{1}{2}||\nu||^2 \\ &\text{s.t.} \nu \in \mathbb{R}^n. \end{aligned} \tag{2.27}$$

Instead of solving problem (2.27) exactly, it is interesting to deal with inexact solutions. So, if x is not Pareto critical, we say that ν is an approximative solution of (2.27) with tolerance $\sigma \in [0, 1]$ if

$$f_x(\nu) + \frac{1}{2}||\nu||^2 \leq \sigma\alpha(x), \tag{2.28}$$

where, as above, $f_x(\nu) = \max_i(\nabla F(x)\nu)_i$ and $\alpha(x)$ is the optimum value of problem (2.27). Observe that for $\sigma = 1$ only the exact solution satisfies the above inequality.

Define $A \in \mathbb{R}^{k \times n}$,

$$||A||_{\infty,2} = \max_{x \neq 0} \frac{||Ax||_\infty}{||x||}. \tag{2.29}$$

Then $||.||_{\infty,2}$ is a norm in $\mathbb{R}^{k \times n}$ and

$$||A||_{\infty,2} = \max_{i=1,\ldots,k} ||A_i, .|| = \max_{i=1,\ldots,k} \left(\sum_{j=1}^{n} A_{i,j}^2 \right)^{\frac{1}{2}}. \tag{2.30}$$

In the following we explaine the computing of the step length. Suppose that we have a direction $\nu \in \mathbb{R}^n$ with $\nabla F(x)\nu < 0$. To compute a step length $t > 0$ we use an Armijo-like rule. Let $\beta \in [0, 1]$ be a prespecified constant. The condition to accept t is

$$F(x + t\nu) \leq F(x) + \beta t \nabla F(x)\nu. \tag{2.31}$$

We start with $t = 1$, and while this condition is not satisfied, we set $t = t/2$. Finiteness of this procedure follows from the fact that (2.2.2) holds strictly for $t > 0$ small enough.

The hold algorithm is presented following

Algorithm 3 Steepest descent for multi-objective optimization

Require: Choose $\beta \in (0, 1)$, $\sigma \in (0, 1]$ and $x^0 \in \mathbb{R}^n$

1: set $k = 0$
2: **if** $x^{(k)}$ is Pareto critical **then**
3: STOP
4: **end if**
5: Compute $\nu^{(k)}$, an approximative solution of (2.27) at $x = x^{(k)}$ with tolerance σ.
6: Compute a step length $t_k \in (0, 1]$ as the maximum of

$$T_k = \{t = 1/2^j | j \in \mathbb{N}, F(x^{(k)} + t\nu^{(k)}) \leq F(x^{(k)}) + \beta t \nabla F(x^{(k)})\nu^{(k)}\}$$

7: Set $x^{(k+1)} = x^{(k)} + t\nu^{(k)}$
8: $k = k + 1$
9: Go to 2

2.2.3 Continuation Methods

Those methods are also known as embedding or homotopy methods and have been applied to solve MOPs. The main motivation to use them for the numerical treatment

of MOPs comes from the fact, mentioned before, that in general and under certain mild assumptions on the MOP, the PS forms a $(k-1)$ dimensional object, where k represents the number of objectives involved in the MOP. Thus, specialized techniques capable to perform a search along the manifold of solutions promise to be efficient applied to this context.

In the following we shortly review general continuation methods, for details we refer to [1].

Let $H(x, \lambda) = 0$ be underdetermined system of nonlinear equations; in general, such a system implicity defines a curve or one-manifold of solutions points. Continuation methods numerically tracing such curves.

Assume that we are given

$$H(x) = 0, \tag{2.32}$$

where $H : \mathbb{R}^{N+1} \to \mathbb{R}^N$ is a smooth map. When we say that a map is smooth we shall mean that it has as many continuous derivatives as needed.

If we have a point $x \in \mathbb{R}^{N+1}$ that is a solution of (2.32) and the Jacobian matrix $H'(x)$ has maximum rank, i.e. $\text{rank}(H'(x)) = N$, then it follows from the Implicit Theorem Function (IFT) [29] that there exist a smooth curve $s \in (-\epsilon, \epsilon) \mapsto c(s) \in \mathbb{R}^{N+1}$ for some open interval $(-\epsilon, \epsilon)$ containing zero such that $c(0) = x$ and

$$H(c(s)) = 0, \quad \forall s \in (-\epsilon, \epsilon) \tag{2.33}$$

By differentiating (2.33) it follows that the tangent $c'(s)$ satisfies the equation

$$H'(c(s))c'(s) = 0 \tag{2.34}$$

and can be found via computing kernel vectors of $H'(x)$. This is done in literature by QR factorization [26] of $H'(x)^T$. If

$$H'(x) = QR = (q_1, q_2, ..., q_{N+1})R \tag{2.35}$$

for an orthogonal matrix $Q \in \mathbb{R}^{N+1} \times N + 1$ and a right upper triangle matrix $R \in \mathbb{R}^{N+1} \times N$, then the last column vector q_{N+1} of Q is such a desired kernel vector. The orientation of the curve can be inferred by monitoring the sign of

$$\det \begin{pmatrix} H'(x) \\ q_{N+1}^T \end{pmatrix}. \tag{2.36}$$

The relation between the classical continuation methods and multi-objective optimization is as follows: the first-order optimality conditions for MOPs (see Theorem 2.1, in page 10) lead to an undetermined system of equations. So, applying the IFT on the underlying system and under some mild conditions, it is inferred that the optimal solution set is a $(k-1)$ dimensional object.

Predictor-corrector Methods

The idea of predictor-corrector (PC) methods is to numerically trace the curve c define above by generating a sequence of points $x_i, i = 1, 2, ...,$ along the curve satisfying a chosen tolerance criterion, say $H(x) \leq \epsilon$ for some $\epsilon > 0$. We assume here that a regular [2] starting point $x_0 \in \mathbb{R}^{N+1}$ is given such that $H(x_0) = 0$.

We define the regular solution set as

$$M = \{x \in S | H(s) = 0, x \quad \text{regular}\}. \tag{2.37}$$

The PC methods consist of two steps which can in general be described as follows:

(P) Predict step $p_1, p_2, ..., s$ of distinct (and well distributed) points which are near to x^0 and to M.

(C) For $i = 1, 2, ..., s$ starting with the predicted point p_i, compute by some (typically few) iterative steps an approximated element x_i of M, i.e. $H(x_i) \approx 0$.

In Figure 2.6 we present a more illustrative example of the working principle of the PC methods.

Method by Hillermier

This method was proposed by Hillermier in [29]. It utilizes the KKT conditions and can cope with equality constraints. However, two of its drawbacks are the Hessians requirement and the lack of strategies to handle inequality constraints.

Consider the map

[2] A point $x \in \mathrm{R}^{N+k}$ is called regular if the first derivative, $\nabla H(x)$, has full rank.

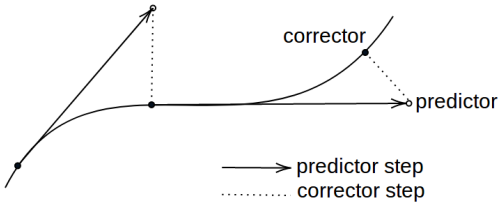


Figure 2.6: Example of PC performance.

$$\tilde{F}(x,\alpha,\lambda) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_j \nabla f_j(x) \sum_{j=1}^{p} \lambda_j \nabla h_j(x) \\ h(x) \\ \sum_{j=1}^{k} \alpha_j - 1 \end{pmatrix} = 0. \tag{2.38}$$

The set of KKT points of a nonlinear equality constrained MOP is contained in the zero set $\tilde{F}$ which motivates the continuation along $\tilde{F}^{-1}(0)$. For BOPs the method is the same as the general technique described in Section 2.2.3 but with the following modification: instead of computing the determinant in (2.36) to orientate the continuation, the author proposes to check whether the condition

$$[x - \tilde{x}] \cdot q \geq 0 \tag{2.39}$$

is met, where $x = (\mathrm{x}, \alpha, \lambda) \in \mathbb{R}^{n+k+p}$ is the current corrector point, $\tilde{x}$ is the previously solutions and q is the tangent vector. If that is not the case, the direction of q is flipped. Then, a suitable step length that guarantees a uniform spread of solutions on the front is sought. That is, for two consecutive solutions $\tilde{x}$ and x, it is desirable that

$$||F(x) - F(\tilde{x})|| \approx \tau, \tag{2.40}$$

where τ is specified by user. For this, one can take the step size

$$t = \frac{\tau}{||Jq||} \tag{2.41}$$

The multi-objective MOPs ($k > 1$) case is handled by taking kernel vectors of $\tilde{F}'^T$. Given that

$$\tilde{F}'(x, \alpha, \lambda) = QR = (q_1, q_2, ..., q_{n+k})R \tag{2.42}$$

for orthogonal matrix $Q \in \mathbb{R}^{(n+k+p)\times(n+k+p)}$ and a right upper triangular matrix $R \in \mathbb{R}^{(n+k+p)\times(n+p+11)}$, the last $k-1$ column vectors of Q forms an othonormal basis of the linearized solution set in the compound (x, α) space. Thus, one can e.g., move in the direction of the computed orthonormal vectors $q_{n+p+2}, ..., q_{n+p+k}$ to obtain predictors that aregrid-aligned along the tangent space of the optimal manifold in decision space. A problem of this election, though, is that after mapping the computed predictors to the objective space, the grid alignment is probably not kept. Finally, the continuation algorithm is stopped if one of the Lagrange multipliers $\alpha_j, j \in (1, ..., k)$ is negative, which indicates that a non-optimal solution has been found.

Equispaced Pareto Front Construction

This method was proposed by Victor Pereyra et. al. in 2013 [39]. The algorithm is based on convex combinations of the objectives and homotopy continuation. For simplicity we consider a bi-objective optimization problem (BOP) ($k = 2$), but the construction described here can be extended to MOPs. We introduce the scalar objective function:

$$f(x, \lambda) = (1 - \lambda)f_1(x) + \lambda f_2(x), \tag{2.43}$$

where $0 \leq \lambda \leq 1$, and the same problem constraints apply. From standard homotopy, the method starts at $\lambda = 0$ and then steps λ in some fashion, obtaining successive solutions and a discrete sample of the PS and PF. However, using this method there is no sure way to obtain a uniform sampling with it because the parametrization of the PF by λ is usually a very nonlinear unknown map.

In the current method we use the idea of intrinsic parametrization of the PF by using discrete arc length. Let x_1^* and x_2^* be the minimizers of $f_1(x)$ and $f_2(x)$, define $x_0 = x_1*$ and $x_{l+1} = x_2^*$ to be the end-points of the PS, where l is the desired number of points in the PF segment joining the points.

Given a collection $f_i = f(x_i), i = 0, 1, \ldots, l+1$ (where in the general case $f(x)$ is the vector $[f_1(x), \ldots, f_k(x)]^T$), we define the chord length of a polygon defined by the points as $S_l = \sum_{i=0}^{l} \|f_{i+1} - f_i\|$ where the l_2 norm is used as penalty function. If the

points are sampling of a smooth curve, then when the spacing between the points tends to zero, the chord length tends to the arc length of the curve. Now we define

$$\gamma = \frac{\alpha \|f(x_0) - f(x_{l+1})\|}{l}, \tag{2.44}$$

where the distance between the images of the minimizers times a factor $\alpha > 1$ is an estimate of the total chord length of the Pareto front (accounting for curvature). In the unlikely case that $f(x_0) = f(x_{l+1})$ we are finished because the PF would be a single point. Otherwise, $\gamma \neq 0$ and we impose the following equispacing constraint:

$$\|f(x) - f_{\text{prev}}\|^2 = \gamma^2, \tag{2.45}$$

where f_{prev} is a previous point in the homotopy process.

We then minimize the scalarized function subject to all constraints with λ as additional variable:

$$\begin{aligned} \min_{x,\lambda} \quad & (1-\lambda) f_1(x) + \lambda f_2(x) \\ \text{subject to} \quad & g(x) \leq 0, \quad x \in D, \\ & \|f(x) - f_{\text{prev}}\|^2 = \gamma^2 \,. \end{aligned} \tag{2.46}$$

With $f_{\text{prev}} = f(x_0)$, let the solution be (x_1, λ_1). We repeat the process with $f_{\text{prev}} = f(x_1)$ to obtain (x_1, λ_2), and so on. The corresponding discrete Pareto front is defined as the set $\{f(x_i),\, i = 0, \ldots, l+1\}$, where $\{x_i,\, i = 0, \ldots, l+1\}$ is the Pareto set. If the process is successful we obtain a discrete representation of the Pareto front with equispaced values $f(x_i)$. One disadvantage that we can note is with the use of this intrinsic parametrization, the λ parametrization of the front loses importance.

Zig Zag Search Method

The zigzag search method was proposed by Honggang Wang in 2012 [43], it is a new method that use a gradient-based zigzag search approach for BOPs. It progressively finds a set of non-dominated solutions by searching around the PF, making an efficient local search using the gradients of the objective functions.

The zigzag algorithm consists of three steps: finding the first Pareto optimal (FPO) search, zig search and zag search, which we describe in the following.

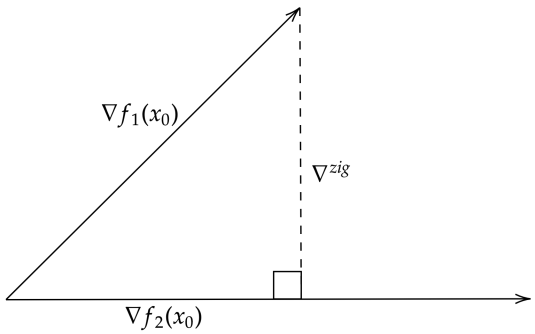


Figure 2.7: Projection of $\nabla f_1(x_0)$ onto $\nabla f_2(x_0)$.

An FPO search is based on a line search solution against objective one (f_1) while maintaining the smallest value of objective two (f_2). It consists of two main parts: 1) a regular line search returns an optimal solution for f_2; 2) a horizontal search for f_2 meaning a search along the projection of ∇f_2 onto ∇f_1:

$$\nabla^{FPO} = \nabla f_2(x_0) - \langle \nabla f_1(x_0), \nabla f_2(x_0) \rangle \frac{\nabla f_1(x_0)}{||\nabla f_1(x_0)||}. \tag{2.47}$$

A zig search tries to find a solution that relaxes the value of f_1 somewhat while keeping f_2 the same. It projects the gradient of f_1 onto f_2:

$$\nabla^{zig} = \nabla f_1(x_0) - \langle \nabla f_1(x_0), \nabla f_2(x_0) \rangle \frac{\nabla f_2(x_0)}{||\nabla f_2(x_0)||}. \tag{2.48}$$

Now, along this direction $x_1 = x_0 + t\nabla^{zig}$ is obtained and evaluated, as shown in Figure 2.7, t represents the step size. The zig function is sketched in Algorithm 4.

Similarly, zag search also searches along the gradient projection of one objective to another. However, it will follow the projection of f_2 onto f_1:

$$\nabla^{zag} = \nabla f_2(x_0) - \langle \nabla f_1(x), \nabla f_2(x_0) \rangle \frac{\nabla f_1(x_0)}{||\nabla f_1(x_0)||}. \tag{2.49}$$

Zag search tends to find the best solution for f_2 while trying to keep f_1 the same. And $x_1 = x_0 + t\nabla^{zag}$. The zig function is sketched in Algorithm 5.

Then, zigzag from the solution obtained from FPO enables the formation of a whole PF.

Algorithm 4 Zig function

Require: $x_0 \in \mathbb{R}^n$

if $\nabla f_1(x_0) = 0$ **then**
  set $\nabla^{zig} = \text{rand}()$
else if $\nabla f_2(x_0) = 0$ **then**
  set $\nabla^{zig} = \nabla f_1(x)$
else
  set $\alpha = angle(\nabla f_1(x_0), \nabla f_2(x_0))$
  if $\alpha != \pi$ **then**
    set $\nabla f_2(x) = \frac{\nabla f_2(x_0)}{||\nabla f_2(x_0)||}$
    set $\nabla^{zig} = \nabla f_1(x_0) - (\nabla f_1(x_0), \nabla f_2(x_0))\nabla f_2(x_0)$
    (project $\nabla f_1(x_0)$ on the othogonal plane of $\nabla f_2(x_0)$)
  else
    set $\nabla^{zig} = \nabla f_1(x_0)$
  end if
end if
$x = x_0 + t\nabla^{zig}$
set $x = project(x)$ (project x into X)

Algorithm 5 Zag function

Require: $x_0 \in \mathbb{R}^n$

set $n = 0$
set $x_{n+1} = x_n$
while $x_{n+1} \geq x_n$ **do**
  set $n = n + 1$
  if $\nabla f_2(x_n) = 0$ **then**
    set $\nabla f_2(x_n) = \text{rand}()$
  end if
  if $\nabla f_1(x_n) = 0$ **then**
    set $\nabla^{zag} = \nabla f_2(x_n)$
  else
    set $\alpha = angle(\nabla f_1(x_n), \nabla f_2(x_))$
    if $\alpha != \pi$ **then**
      set $\nabla f_1(x_n) = \frac{\nabla f_1(x_n)}{||\nabla f_1(x_n)||}$
      set $\nabla^{zag} = \nabla f_2(x_n) - \langle \nabla f_1(x_n), \nabla f_2(x_n) \rangle \nabla f_1(x_n)$
    else
      set $\nabla^{zag} = \nabla f_2(x_n)$
    end if
  end if
  set $x_{n+1} = x_n - t\nabla^{zag}$
  $x = project(x)$
end while
return $x = x_n$

Pareto Tracer Method

The Pareto Tracer (PT) algorithm was proposed by Martín and Schütze in 2015 [36]. It is a novel PC method for the numerical treatment of MOPs. The present author's algorithm utilizes $\tilde{F}$, but separates decision and weight space leading to significant savings in the overall computational cost.

For simplicity, unconstrained problems (2.50) are addressed and constraint handling presented later. That is, first we consider the problem

$$\min_{x \in \mathbb{R}^n} F(x) \tag{2.50}$$

Predictor

Let $x \in \mathbb{R}^n$ be a KKT point of (2.50) and $\alpha \in \mathbb{R}^k$ its associated Lagrange multiplier, by the KKT conditions we have

$$\tilde{F}(x, \alpha) = \begin{pmatrix} \sum_{i=1}^k \alpha_i \nabla f_i(x) \\ \sum_{i=1}^k \alpha_i - 1 \end{pmatrix} = 0. \tag{2.51}$$

Let $\nu \in \mathbb{R}^n$ and $\mu \in \mathbb{R}^k$, by differentiating (2.51) we obtain

$$\tilde{F}'(x, \alpha) \begin{pmatrix} \nu \\ \mu \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^k \alpha_i \nabla^2 f_i(x) & \nabla f_1(x) & \dots & \nabla f_k(x) \\ 0 & 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} \nu \\ \mu \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}. \tag{2.52}$$

By the second equation in (2.52) it follows that

$$\sum_{i=1}^k \mu_i = 0. \tag{2.53}$$

Now assume that a vector $\mu \neq 0$ is given such that (2.53) is fulfilled. Then, by the first equation in (2.52),

$$\sum_{i=1}^k \alpha_i \nabla^2 f_i(x) \nu = -\sum_{i=1}^k \mu_i \nabla f_i(x) = -J^T \mu, \tag{2.54}$$

where J denotes the Jacobian of F at x.

Assume also that the matrix

$$W_\alpha = \sum_{i=1}^{k} \alpha_i \nabla^2 f_i(x) \in \mathbb{R}^{n \times n} \tag{2.55}$$

is regular. Then, given μ, the vector ν_{mu} that satisfies (2.52) can be expressed as

$$\nu_\mu = -W_\alpha^{-1} J^T \mu. \tag{2.56}$$

Given a direction in decision space $\nu \in \mathbb{R}^n$, the corresponding movement in objective space for infinitesimal step sizes is given by

$$d = J\nu. \tag{2.57}$$

The orientation of the movements along the tangent space is related to (2.57), now we want to find the proper orientation vector $d \in \mathbb{R}^k$ such that

$$J\nu_{\mu_d} = d. \tag{2.58}$$

Since, further, μ_d has to satisfy (2.53), then assuming than $rank(J) = k - 1, \nu_{\mu_d}$ can be computed via (2.56) using the vector μ_d that solves

$$\begin{pmatrix} -JW_\alpha^{-1} J^T \\ 1 \dots 1 \end{pmatrix} \mu_d = \begin{pmatrix} d \\ 0 \end{pmatrix}. \tag{2.59}$$

So the predictor is given by $p = x + t\nu_\mu$ at the given KKT point x, it is required to select the step size t.

Corrector

PT takes the Newton method for unconstrained MOPs from [21] as corrector, where the Newton direction is defined as the solution of

$$\min_{(\nu,\delta)\in\mathbb{R}^n\times\mathbb{R}} \delta$$
$$\text{s.t. } \nabla f_i(x)^T\nu + \frac{1}{2}\nu^T\nabla^2 f_i(x)\nu \leq \delta, \quad i = 1, ..., k. \tag{2.60}$$

where δ serves has measure of the expected decrease in objective space produced by a line search in direction ν in parameter space.

Algorithm 6 shows one way to explore the Pareto set/front around a given solution x_0 which follows the steps described above.

Algorithm 6 Local exploration of the Pareto set/front around x_0

Require: KKT point x_0 of (2.50) with associated convex weight, directions $\mu_1, .., \mu_s \in \mathbb{R}^t, \tau > 0$
Ensure: KKT points $x_i, i = 1, ..., s$ around x_0
1: {PREDICTOR STEP}
2: **for** $i = 1, ..., s$ **do**
3: Compute $\nu_i = \nu_{mu_i}$ as in (2.56)
4: Compute t_i
5: Compute $p_i = x_0 + t_i + \nu_i$
6: **end for**
7: {CORRECTOR STEP}
8: **for** $i = 1, ..., s$ **do**
9: Compute KKT point x_i (and associated weight) using (2.60) starting with p_i
10: **end for**

Handling equality constraints

Now we are going to consider MOPs of the form

$$\min_{x\in\mathbb{R}^n} F_(x),$$
$$\text{s.t. } h_i = 0, i = 1, ..., p. \tag{2.61}$$

Lets consider the auxiliary KKT map $\tilde{F} : \mathbb{R}^{n+k+p} \to \mathbb{R}^{n+p+1}$

$$\tilde{F}(x, \alpha, \lambda) = \begin{pmatrix} \sum_{i=1}^k \alpha_i \nabla f_i(x) + \sum_{i=1}^p \lambda_i \nabla h_i(x) \\ h(x) \\ \sum_{i=1}^k \alpha_i - 1 \end{pmatrix} = 0. \tag{2.62}$$

Defining

$$W_{\alpha,\lambda} = \sum_{i=1}^{k} \alpha_i \nabla^2 f_i(x) + \sum_{i=1}^{p} \lambda_i \nabla^2 h_i(x) \in \mathbb{R}^{n \times n}, \tag{2.63}$$

and

$$H = \begin{pmatrix} \nabla h_1(x)^T \\ \vdots \\ \nabla h_p(x)^T \end{pmatrix} \in \mathbb{R}^{p \times m}, \tag{2.64}$$

leads to

$$\tilde{F}(x, \alpha, \lambda) = \begin{pmatrix} W_{\alpha,\lambda} & J^T & H^T \\ H & 0 & 0 \\ 0 & 1 \dots 1 & 0 \end{pmatrix} \in \mathbb{R}^{(n+p+1)\times(n+k+p)} \tag{2.65}$$

In order to compute the kernel vectors of (2.65) let $\nu \in \mathbb{R}^n$, $\mu \in \mathbb{R}^k$ and $\chi \in \mathbb{R}^p$ such that

$$\tilde{F}(x, \alpha, \lambda) = \begin{pmatrix} W_{\alpha,\lambda} & J^T & H^T \\ H & 0 & 0 \\ 0 & 1 \dots 1 & 0 \end{pmatrix} = \begin{pmatrix} \nu \\ \mu \\ \xi \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{2.66}$$

Again μ is chosen to satisfy (2.53) which reduces (2.66) to

$$\begin{pmatrix} W_{\alpha,\lambda} & H^T \\ H & 0 \end{pmatrix} \begin{pmatrix} \nu_\mu \\ \xi \end{pmatrix} = \begin{pmatrix} -J^T \mu \\ 0 \end{pmatrix}. \tag{2.67}$$

If $rank(W_{\alpha,\lambda}) = n$ and $rank(H) = p$ it follows that the solution of (2.67) is unique since then the matrix on the left-hand side is regular.

Corrector

For the corrector step the Newton method has been modified to the current context. The suggestion is to compute the Newton direction via

$$\begin{aligned} &\min_{(\nu,\delta)\in\mathbb{R}^n\times\mathbb{R}} \delta \\ &\text{s.t. } \nabla f_i(x)^T \nu + \frac{1}{2} \nu^T \nabla^2 f_i(x) \nu \leq \delta, \quad i = 1, ..., k, \\ &\quad h_i(x) + \nabla h_i(x)^T \nu = 0, \quad i = 1, ..., p. \end{aligned} \tag{2.68}$$

The difference of (2.68) over the unconstrained version (2.60) is the additional constraint $h(x)+Hv=0$. In [36] it is shown that the flow defined by this search direction let's from any initial solution to ta KKT point (if existing).

As a conclusion of this chapter we can state that there exist a wide variety of MP techniques that are applicable to MOPs. As state in the introductory chapter, this work is foucus on continuation methods, in special, the algorithm of PT. This algorithm is applicable to problems with an unlimited number of variables and objectives. So far, however, PT can only handle equality and box constraints. In the next chapter we will demonstrate how to modify PT to efficiently treat inequalities as well.

Chapter 3

PT for General Inequality Constrained MOPs

In this chapter we present the extensions of the Pareto Tracer for general inequality constrained MOPs. This chapter is divided in two parts. In the first one, we motivate the needed changes for PT and in the second one we present the modified algorithm in detail.

3.1 Motivation

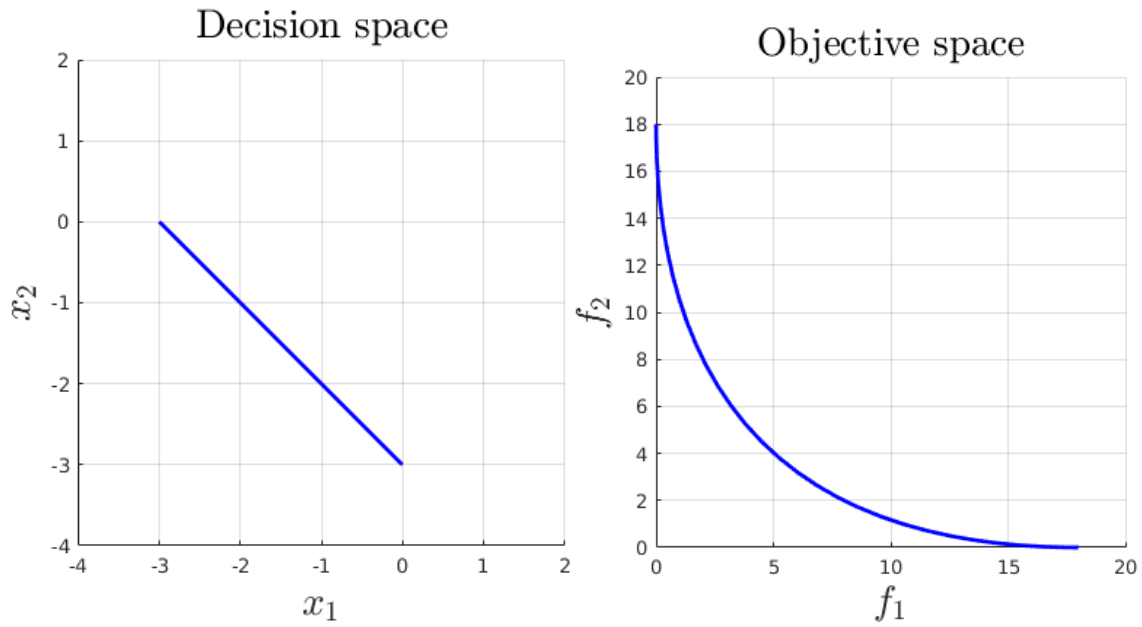
As we state in Chapter 2, PT is a predictor corrector method that can handle equality and box constraints. In real life we face with MOPs that involve not only equality and box constraints but also general inequality constraints. For that reason we aim to improve PT so that it can deal with any kind of constraints.

In order to show the deficiency of PT, we present an example that involves an inequality constrained MOP. Using this example we try to show that the algorithm is currently not able to solve inequality constrained MOPs correctly.

Let us consider the following unconstrained MOP (3.1).

$$\min \begin{cases} f_1(x) & = (x_1+3)^2 + x_2^2 \\ f_2(x) & = x_1^2 + (x_2+3)^2 \end{cases} \tag{3.1}$$

Figure 3.1 shows the solution sets of this problem obtained by using PT.



(a) (b)

Figure 3.1: Solutions sets of problem (3.1).

Next we consider the same problem with the following inequality constraints

$$g_1(x) = (x_1 + 1)^2 + (x_2 + 0.5)^2 \leq 1.6^2 \tag{3.2}$$

Note that $g_1(x)$ represents a restriction over the decision space and geometrically is a circle of diameter 1.6 and center $(-1, -0.5)$. Figure 3.2 shows the geometrical representation of the constraint. Adding this constraint to the unconstrained MOP (3.1) the feasible region of this new problem corresponds to the area inside the circle as shown in Figure 3.3. For the results in decision space of the new problem we expect two things: (i) preserves the solutions of the uncostrained MOP (3.1) that are inside the feasible region and (ii) a projection in the feasible set of all the solutions of the uncostrained MOP (3.1) that are not inside the feasible region. Figure 3.1 shows the solution in decision space that we aim, the expected solution set is represented by the red points.

As we mentioned before, PT is of local nature and the solution sets depend on the starting point. Hence, if we apply PT to this problem we can obtain different solution sets using different starting points. Figure 3.5 shows the solution sets obtained by applying PT with the starting point $(-3, -3)$ and Figure 3.6 shows the solution sets obtained with the starting point $(1, 1)$. It is important to remember that PT is not able to handle inequality constraints and for that reason we put the inequality constraint as an equality one, that's why the results in decision space of the two solutions sets mentioned before correspond to a part of the circle constraint.

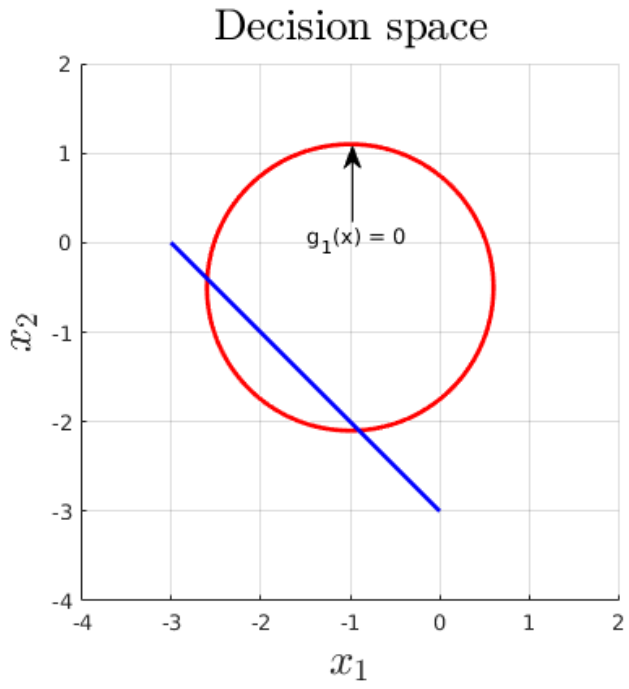


Figure 3.2: Pareto set of the unconstrained problem (3.2) and representation of g_1.

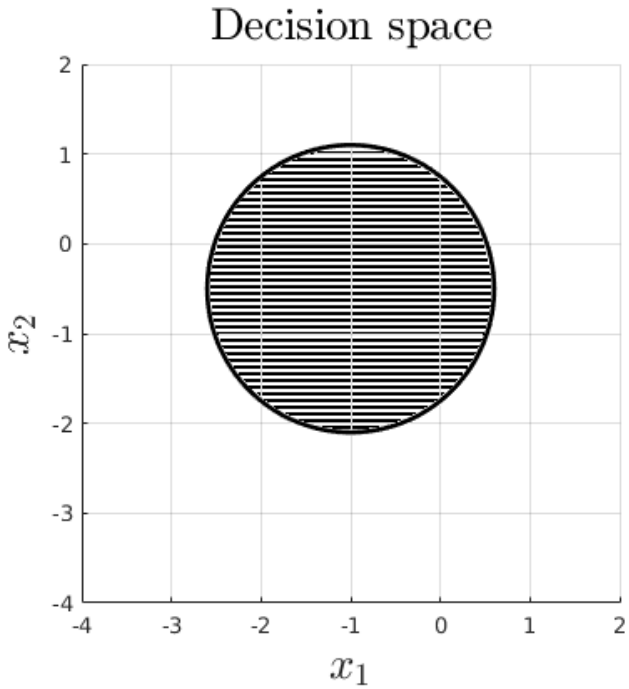


Figure 3.3: Feasible region of (3.1) according to g_1 .

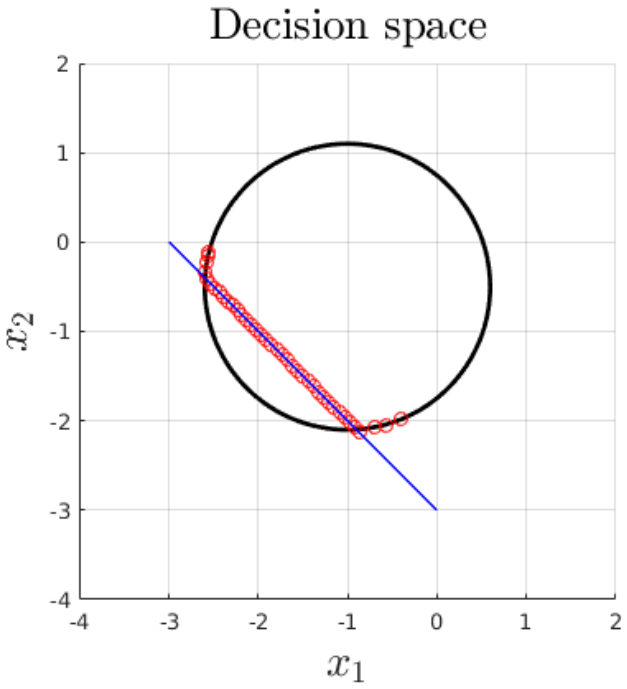
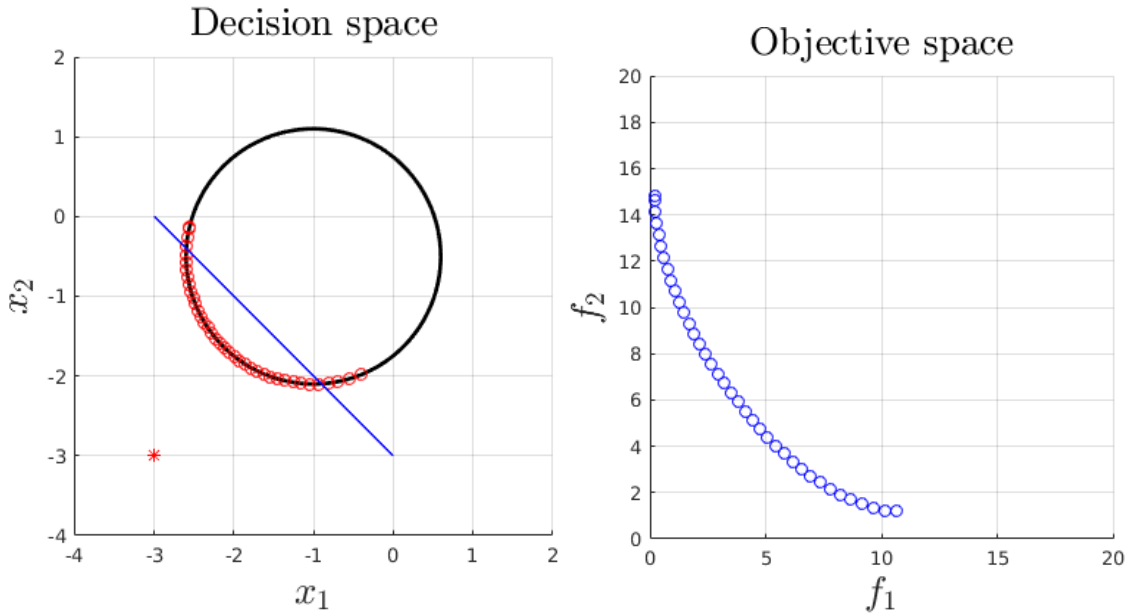


Figure 3.4: Pareto set of the constrained problem.



(a) **(b)**

Figure 3.5: Solution sets obtained by PT for problem (3.1) with (3.2) using the starting point (-3,-3).

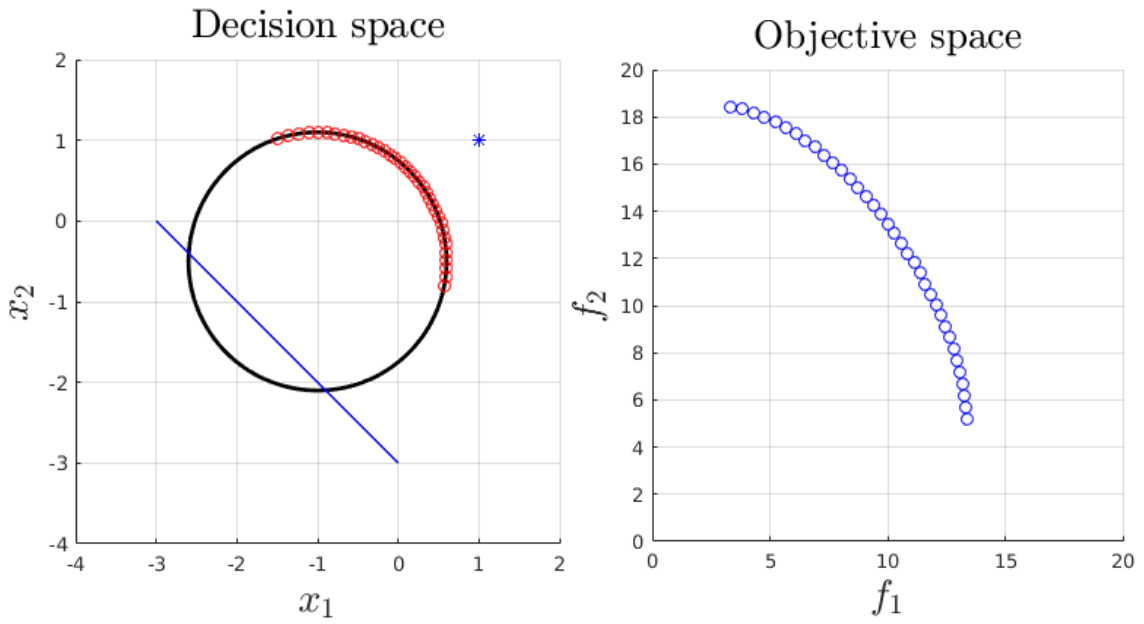


Figure 3.6: Solution sets obtained by PT for problem (3.1) with (3.2) using the starting point (1,1).

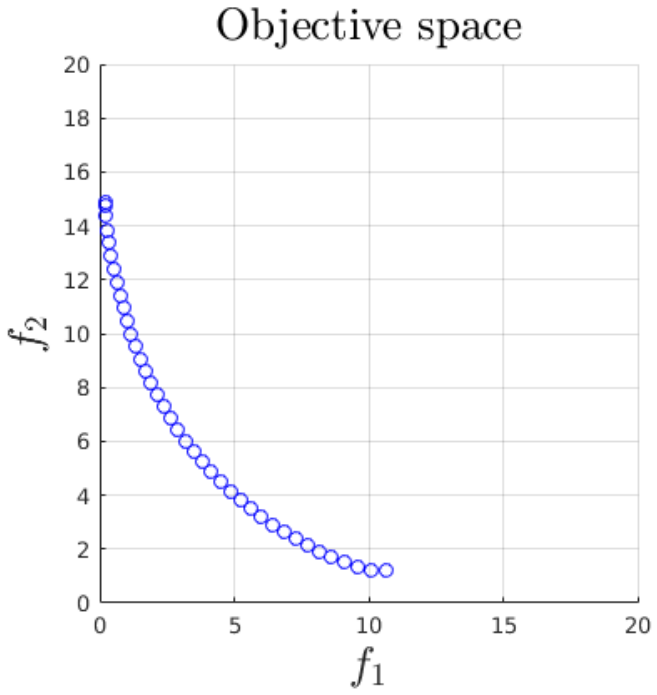


Figure 3.7: Pareto Front of the constrained problem.

If we compare the results in objective space shown in Figure 3.5 and 3.6 with the objective space of the real solution set shown in Figure 3.7 we observe that the last

set dominates the other two. We can say that an improper handling of constraints can lead us to a non-optimal solution set. For the above we are interested on improve the PT method in order it can deal with any kind of constraints. In the next section we introduce a modification of PT leading to an adequate handling of inequality constraints.

3.2 Algorithm

The challenge is to solve constrained MOPs of the form

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} F(x), \\ &\text{s.t.} \quad g_i(x) \leq 0, \quad i = 1,\ldots,m, \\ &\qquad h_j(x) = 0, \quad j = 1,\ldots,p. \end{aligned} \tag{3.3}$$

PT already handles equality constraints so we focused on the inequality constraints.

The primary idea is to consider the inequality constraints as equality ones, selecting in every iteration of the algorithm the correct ones. Later we are going to explain in detail how we make this selection.

As we explained before, the PT method consists of two steps: the predictor and the corrector. For the predictor step we use the same as the original PT method, if the problem has equality constraints we use the predictor for equality constrained MOPs (choosing the predictor direction ν by solving (2.67)) in other case we use the predictor for unconstrained MOPs (choosing the predictor direction ν by solving (2.59)). For the corrector step we also use the Newton method presented in [21] as follows:

$$\begin{aligned} &\min_{(\nu,\delta)\in\mathbb{R}^n\times\mathbb{R}} \quad \delta \\ &\text{s.t.} \quad \nabla f_i(x)^T\nu + \frac{1}{2}\nu^T\nabla^2 f_i(x)\nu \leq \delta, \quad i = 1,...,k, \\ &\qquad h_i(x) + \nabla h_i(x)^T\nu = 0, \quad i = 1,...,p. \\ &\qquad g_i(x) + \nabla g_i(x)^T\nu = 0, \quad i \in I, \end{aligned} \tag{3.4}$$

where I represents a list indices that we create as follows:

Algorithm 7 shows how to build the index set I. The algorithm checks all inequalitues g_i, and adds index i in the following two cases:

Algorithm 7 Build I

Require: x_p given by the predictor step

1: $I = \emptyset$
2: **for** $i = 1, ..., q$ **do**
3: **if** $g_i(x_p) > tol$ **then**
4: $I = I \cup i$
5: **else if** $g_i(x_p) \in (-tol, tol) \wedge \Delta g(x_p)^T \nu_{x_p} > 0$ **then**
6: $I = I \cup i$
7: **end if**
8: **end for**
9: **return** I

- If $g_i(x_p) > tol$, i.e. if x_p significantly violates const g_i.
- If $g_i(x_p) \in (-tol, tol) \wedge \Delta g(x_p)^T \nu_{x_p} > 0$, i.e. that the value of the constraint at the point given by the predictor step is almost zero and and the gradients point outside the feasible region.

If either of these two cases occurs we the need to consider the restriction g_i for the corrector step.

Now we present the solution sets obtained by the modified PT applied to the constrained problem used in the above section. We also use the same starting points as before.

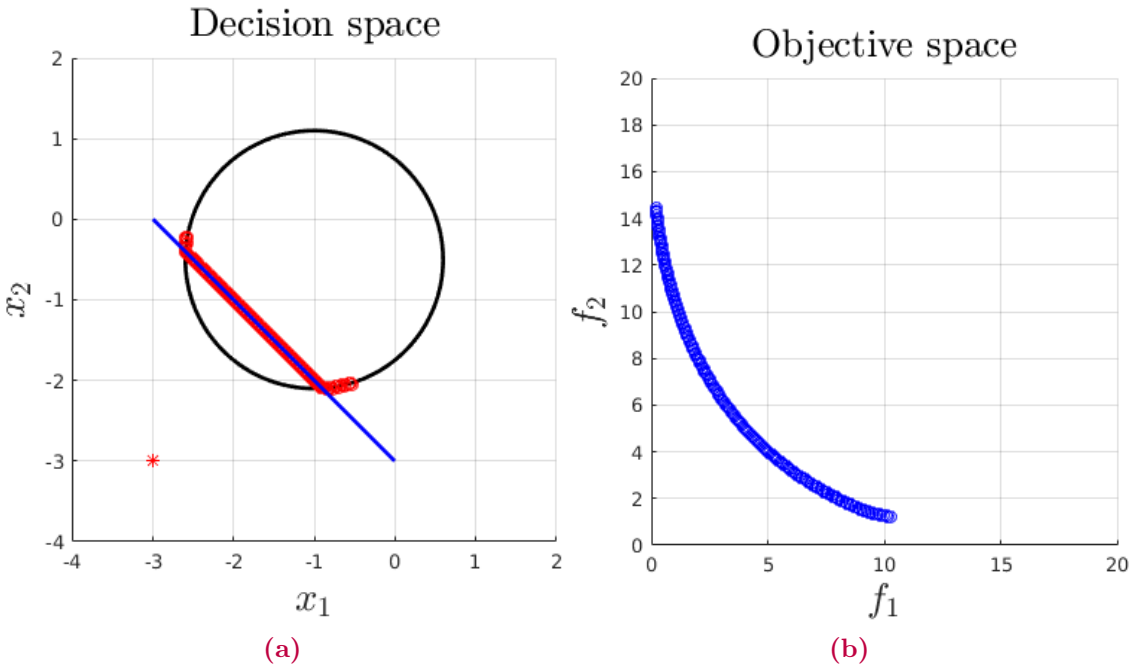


Figure 3.8: Solution sets obtained by PT for problem (3.1) with (3.2) using the starting point (-3,-3).

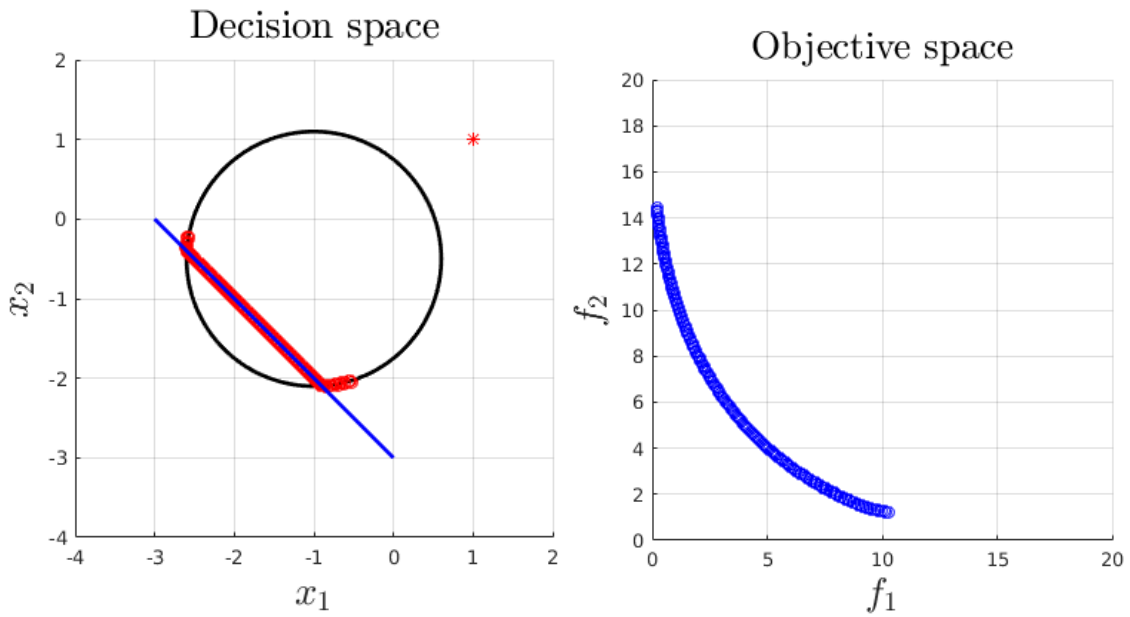


Figure 3.9: Solution sets obtained by the modified PT for problem (3.1) with (3.2) using the starting point (1,1).

Finally we present an example that involves two inequality constraints. Consider the following MOP

$$\begin{aligned} \min \quad & \begin{cases} f_1(x) = (x_1+3)^2 + (x_2-2)^2, \\ f_2(x) = x_1^2 + (x_2+3)^2, \end{cases} \\ \text{s.t.} \quad & g_1(x) = (x_1+1)^2 + x_2^2 \leq 2^2, \\ \text{with} \quad & g_2(x) = (x_1+2)^2 + (x_2+2)^2 \leq 2{\cdot}2 \end{aligned} \tag{3.5}$$

Figure 3.11 presents the geometrical representation of the problem (3.5).

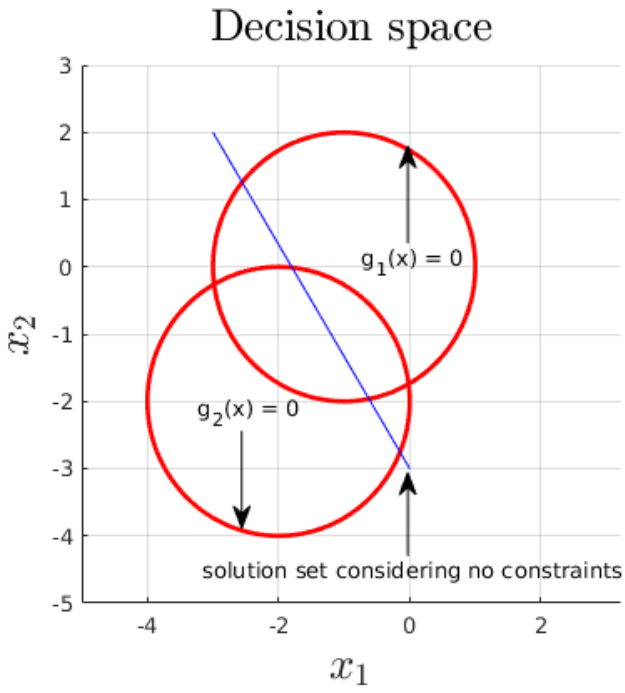
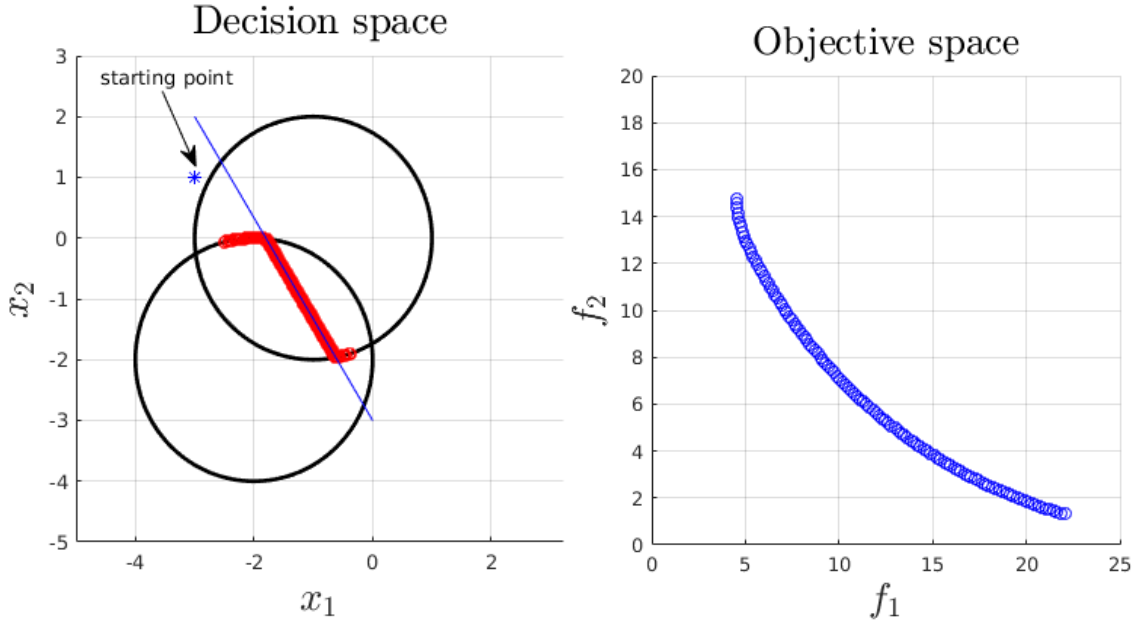


Figure 3.10: Geometrical representation of the problem (3.5).

Figure 3.11 presents the solution sets obtained by the modified PT for the problem (3.5) using the starting point (-3,1).



(a) **(b)**

Figure 3.11: Solution sets obtained by the modified PT for problem (3.5) using the starting point (-3,-3).

Chapter 4

Numerical Results

In this chapter we present numerical results obtained by PT for general inequality constrained MOPs, for simplicity we will refer to it as PT. In order to compare the algorithm, we also present the results obtained by the scalarization methods NBI and the ε -constraint method presented in Section 2.2.1 and by the MOEA NSGA-II. For all comparisons we use the quasi-Newton implementation of the PT that requires derivative information but no Hessians. In order to make a comparison of the different algorithms that require different derivative information we measure the overall function calls that are required in case automatic differentiation [11] is used to compute the derivatives and Hessians, respectively (we refer to it as "total evaluations"). As performance indicator we have chosen to use the averaged Husdorff distance Δ_2 [3, 40]. The average runs of the NSGA-II is 10 and we always consider the best approximation reached (in terms of Δ_2).

4.1 Binh and Korn Problem

Our first test example is a modification of the BOP from Binh and Korn [2] where we add two inequality constraints as follows:

$$\begin{aligned} \min \quad & \begin{cases} f_1(x) = 4x_1^2 + 4x_2^2, \\ f_2(x) = (x_1 - 5)^2 + (x_2 - 5)^2, \end{cases} \\ \text{s.t.} \quad & (x_1 - 2)^2 + (x_2 - 1)^2 \leq 2.3^2, \\ & (x_1 - 3)^2 + (x_2 - 3)^2 \geq 1.5^2, \\ \text{with} \quad & 0 \leq x_1 \leq 5, \\ & 0 \leq x_2 \leq 3. \end{aligned} \tag{4.1}$$

Figure 4.1 shows the feasible region of this problem. Figures 4.2 and 4.3 present the approximations of the solution sets of this problem obtained with the four different algorithms. Table 4.1 shows a comparison of the computational effort for all algorithms. As it can be seen, PT, NBI and NSGA-II all obtain almost perfect approximations in objective space (the performance of NSGA-II is slightly better than the other two. Note, however, that NSGA-II also uses a population size of 100 opposed to the others that generate around 50 solutions). Regarding the computational effort (in terms of "total evaluations") PT clearly outperforms the other algorithms.

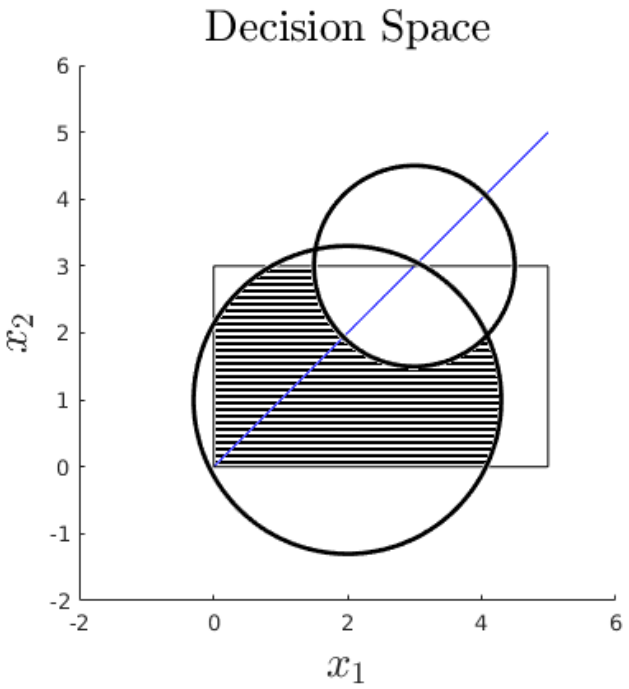


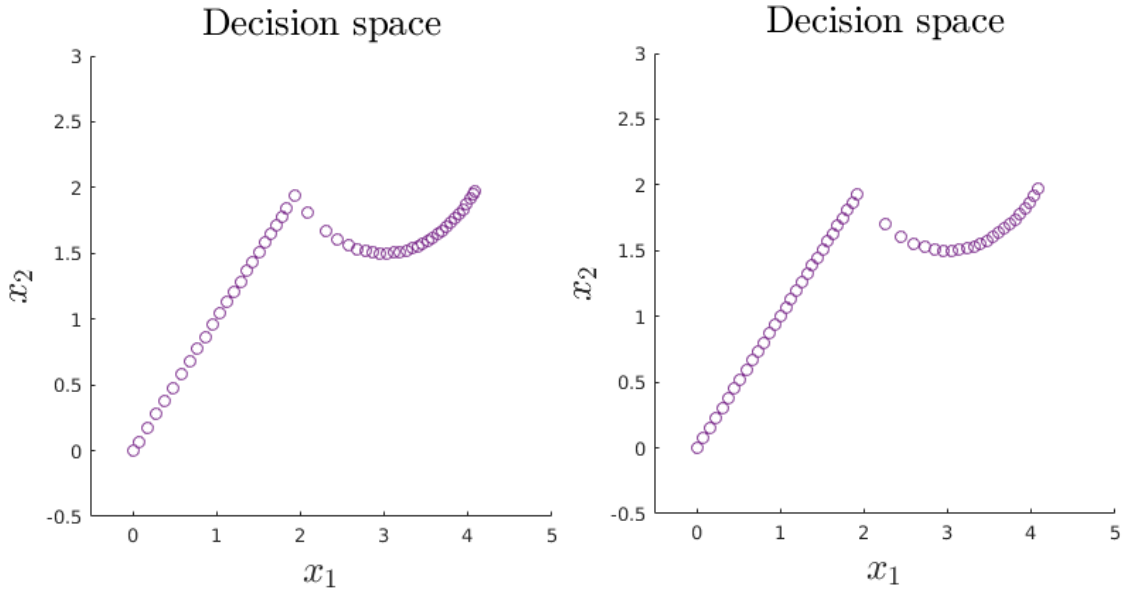
Figure 4.1: Feasible region of the Binh and Korn problem (4.1).

Table 4.1: Computational efforts for the Binh and Korn problem (4.1).

	PT	NBI	ε-constraint method	NSGA-II
Solutions	52	52	52	100
Function Evaluations	151	427	336	2000
Jacobian Evaluations	133	425	336	-
Hessian Evaluations	-	373	284	-
Total of Evaluations	683	8095	6224	2000
Δ_2	0.6050	0.6025	0.9272	0.5703

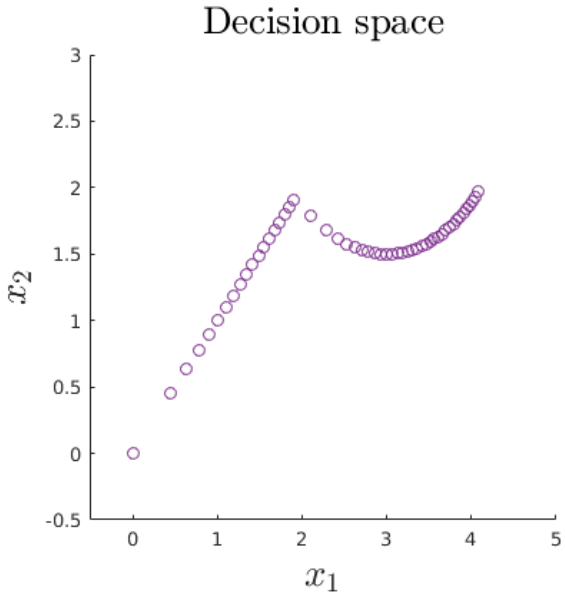
Table 4.2: Parameters used by NSGA-II for the Binh and Korn problem (4.1).

Population size	100
Number of generations	20
Probability of crossover	0.9
Probability of mutation	0.5

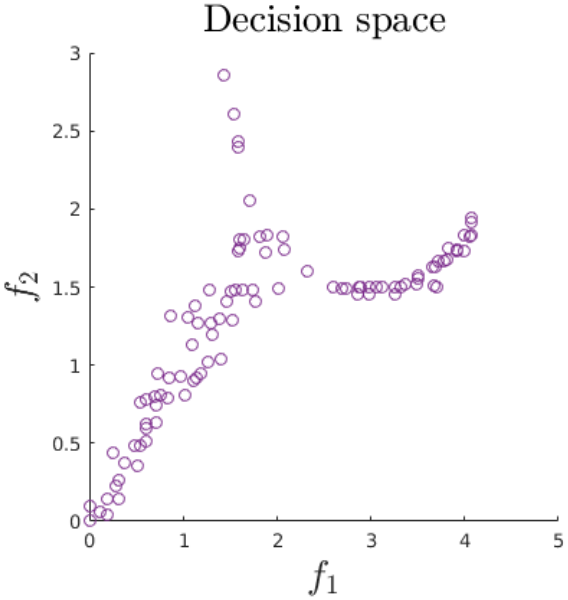


(a) PT

(b) NBI

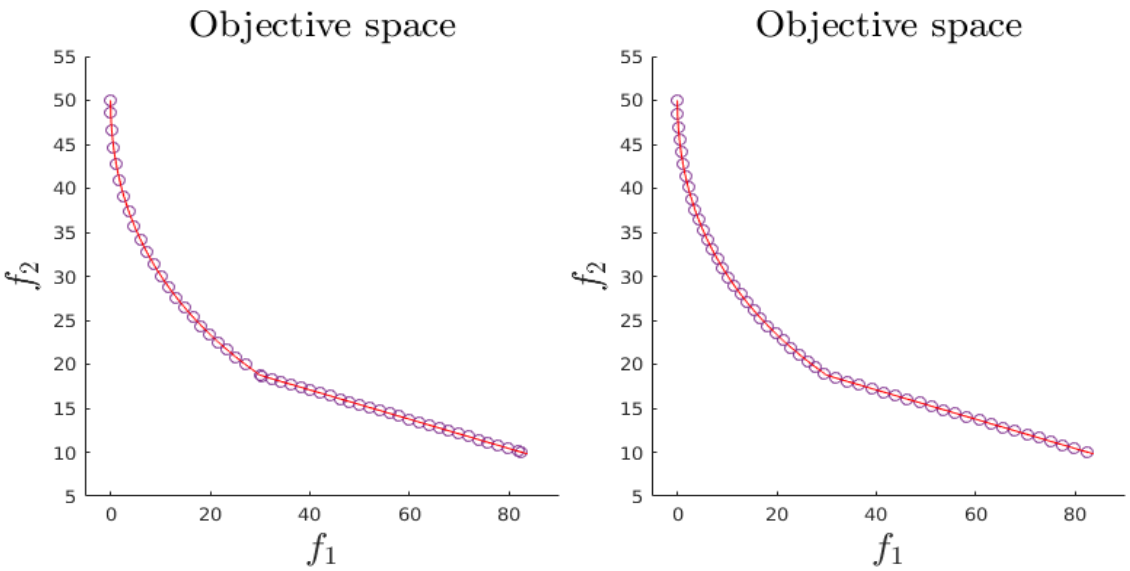


(c) ε-constraint method



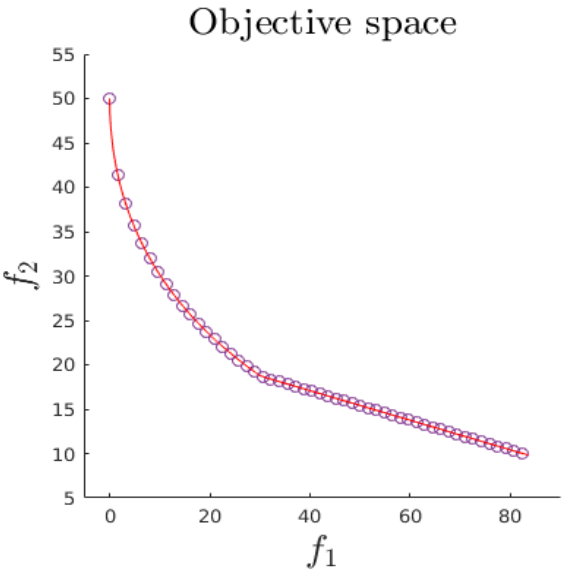
(d) NSGA-II

Figure 4.2: Results in decision space for the Binh and Korn problem (4.1).

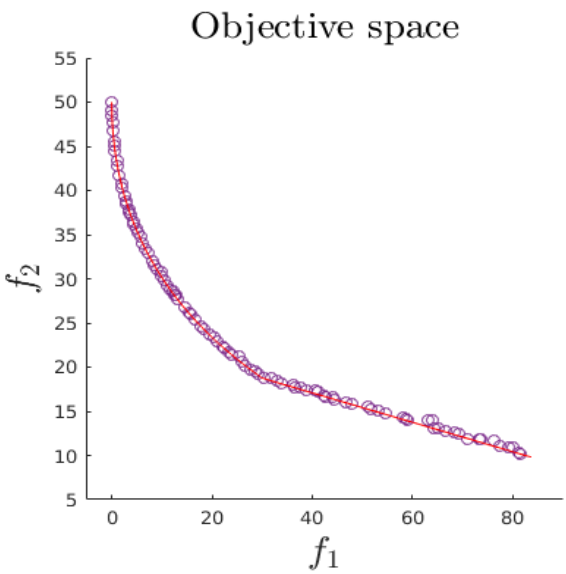


(a) PT

(b) NBI



(c) ε-constraint method



(d) NSGA-II

Figure 4.3: Results in objective space for the Binh and Korn problem (4.1).

4.2 Chakong and Haimes Problem

Our second test example is the BOP presented in (4.2) [7], which depends on two variables and is subject to one nonlinear inequality constraint and one linear inequality constraint. Figure 4.4 shows the feasible region of this problem. Figures 4.5 and 4.6 present the approximations of the solution sets of this problem obtained with the four different algorithms. Table 4.3 shows a comparison of the computational effort for all algorithms. As it can be seen all the algorithms obtain almost perfect approximations in objective space, being PT and NBI the most outstanding ones. In terms of the computational effort (i.e., "total evaluations") PT clearly outperforms the other algorithms. Note that NSGA-II shows the worst performance even while using a population size of 100 opposed to the others that generate around 80 solutions.

$$\begin{aligned} \min \quad & \begin{cases} f_1(x) = 2 + (x_1 - 2)^2 + (x_2 - 1)^2, \\ f_2(x) = 9x_1 - (x_2 - 1)^2, \end{cases} \\ \text{s.t.} \quad & x_1^2 + x_2^2 \leq 225, \\ & x_1 - 3x_2 + 10 \leq 0, \\ \text{with} \quad & -20 \leq x_1, x_2 \leq 20. \end{aligned} \tag{4.2}$$

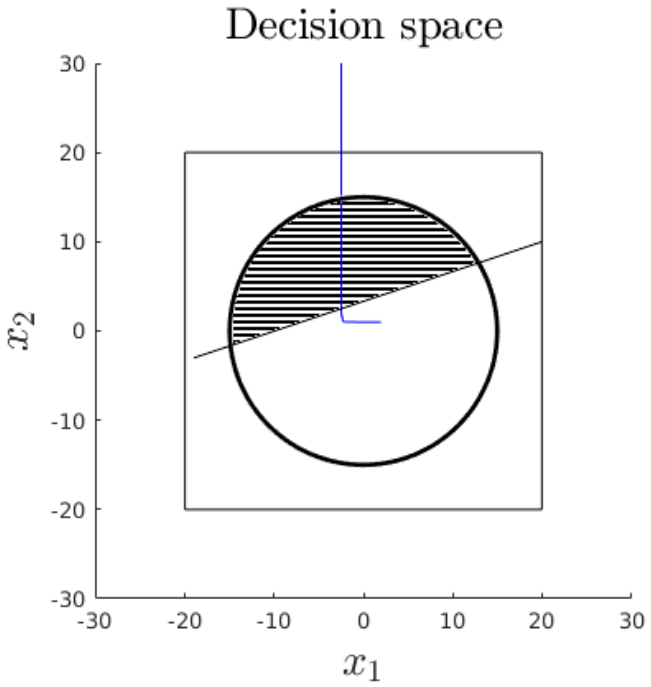
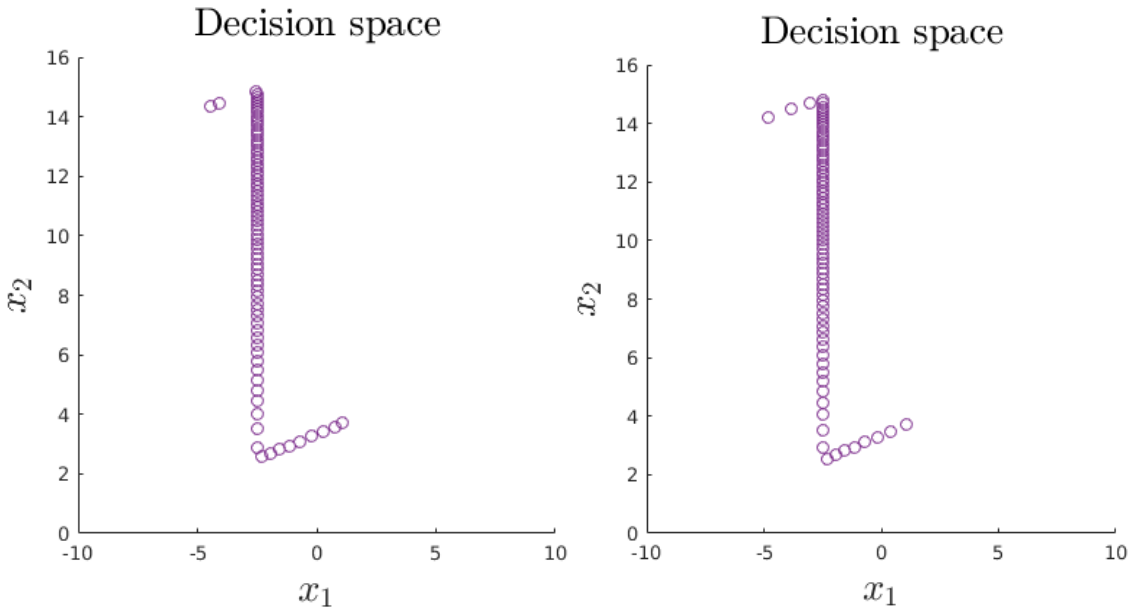
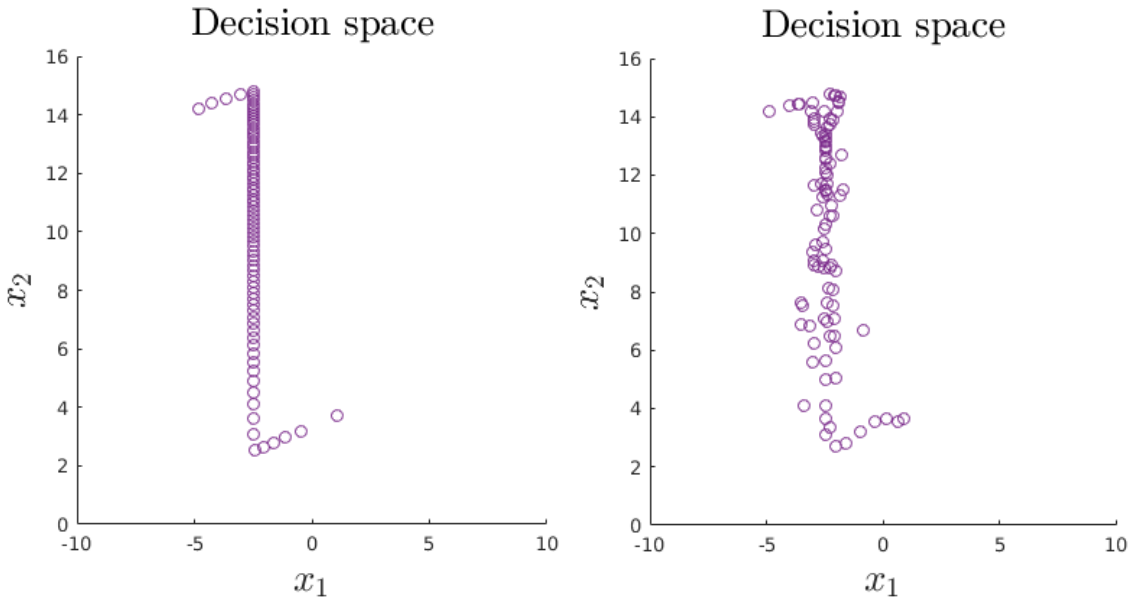


Figure 4.4: Feasible region of the Chakong and Haimes problem (4.2).

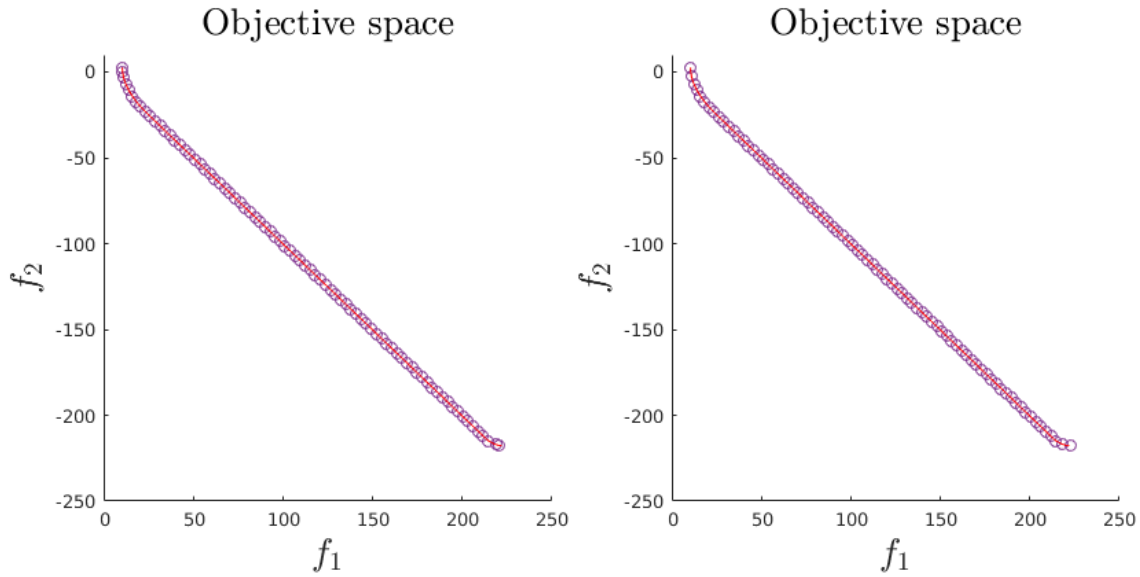


(a) PT **(b)** NBI



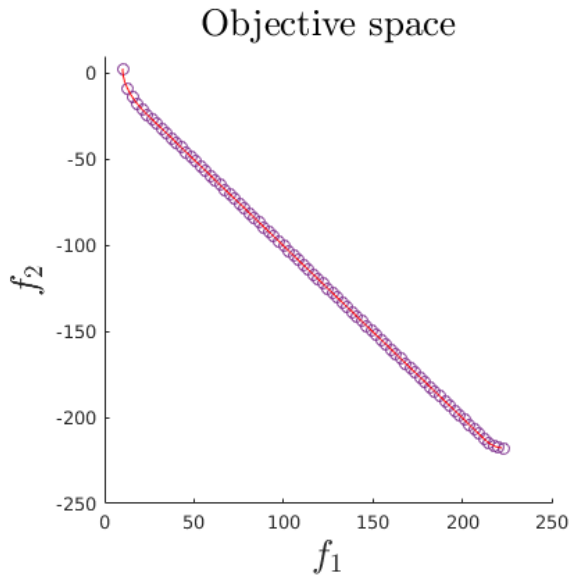
(c) ε-constraint method **(d)** NSGA-II

Figure 4.5: Results in decision space for the Chakong and Haimes problem (4.2).

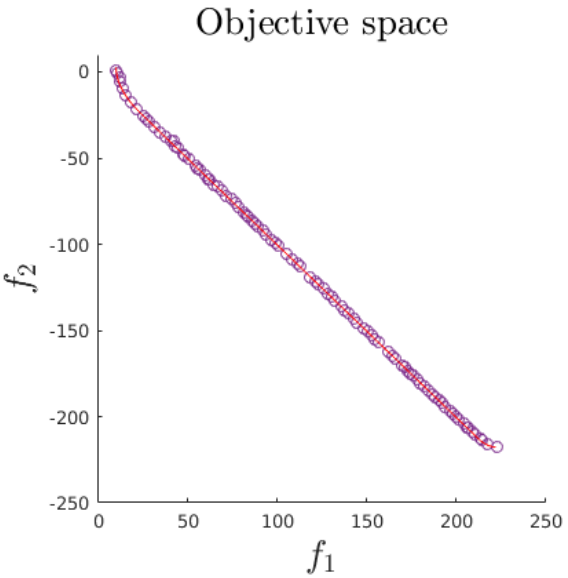


(a) PT

(b) NBI



(c) ε-constraint method



(d) NSGA-II

Figure 4.6: Results in objective space for the Chakong and Haimes problem (4.2).

Table 4.3: Computational efforts for the Chakong and Haimes problem (4.2).

	PT	NBI	ε-constraint	NSGA-II
Solutions	80	80	80	100
Function Evaluations	540	678	578	3000
Jacobian Evaluations	499	678	578	-
Hessian Evaluations	-	598	498	-
Total of Evaluations	2536	12958	10858	3000
Δ_2	1.1459	1.1457	1.2141	1.2307

Table 4.4: Parameters used by NSGA-II for the Chakong and Haimes problem (4.2).

Population size	100
Number of generations	30
Probability of crossover	0.9
probability of mutation	0.5

4.3 Tanaka Problem

Next we consider the problem of Tanaka [42], which depends on two variables and is subject to two nonlinear inequality constraints. Figure 4.7 shows the feasible region of this problem. Figures 4.8 and 4.9 present the approximation of the solution sets of this problem obtained with the different algorithms. Table 4.5 shows a comparison of the computational effort for all algorithms. As it can be seen, PT and NBI obtain almost perfect approximations in objective space while the $\varepsilon-$constraint method and NSGA-II leave out some of the solutions. Nevertheless, to obtain the solutions shows for PT we had to run the algorithm three times with three different starting points, opposed to the others where we need only one run of the algorithm. In terms of the computational effort, once again PT clearly outperforms the other algorithms.

$$
\begin{aligned}
\min = & \begin{cases} f_1(x) & = x_1, \\ f_2(x) & = x_2, \end{cases} \\
\text{s.t.} \quad & x_1^2 + x_2^2 - 1 - 0.1cos\left(16 \arctan\left(\frac{x_1}{x_2}\right)\right) \geq 0, \\
& (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5, \\
\text{with} \quad & 0 \leq x_1 \leq \pi \\
& 0 \leq x_2 \leq \pi.
\end{aligned}
\tag{4.3}
$$

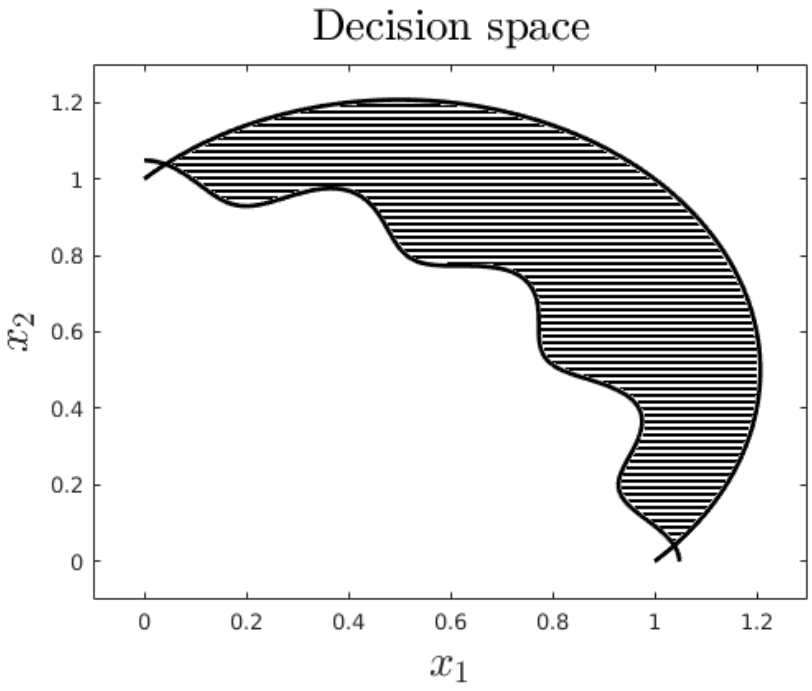
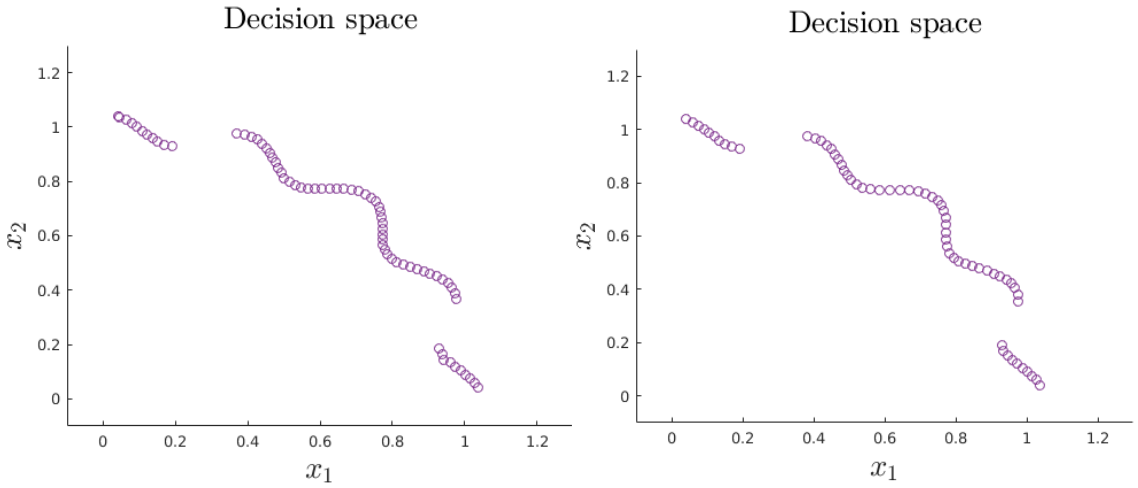
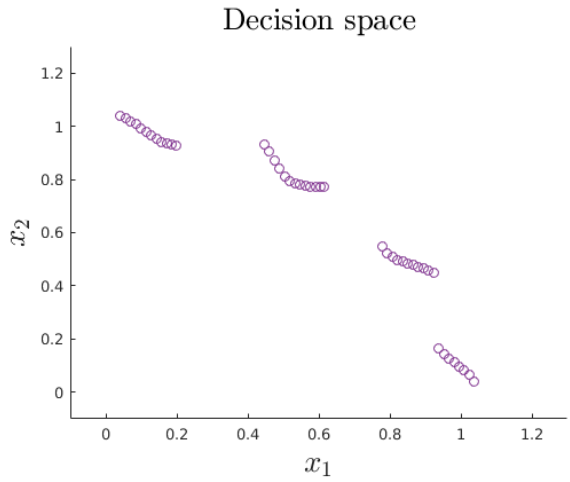


Figure 4.7: Feasible region of the Tanaka problem (4.3).

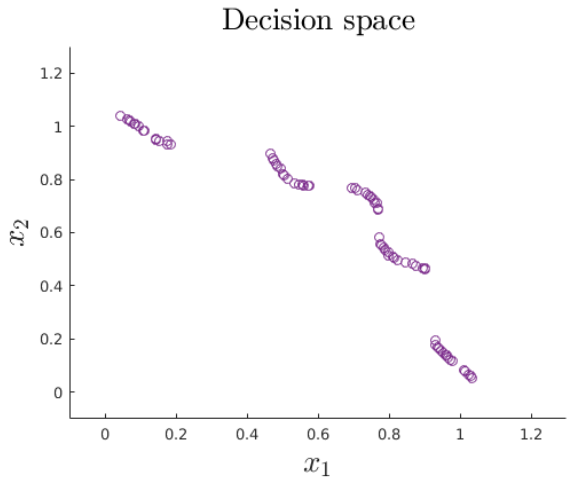


(a) PT

(b) NBI

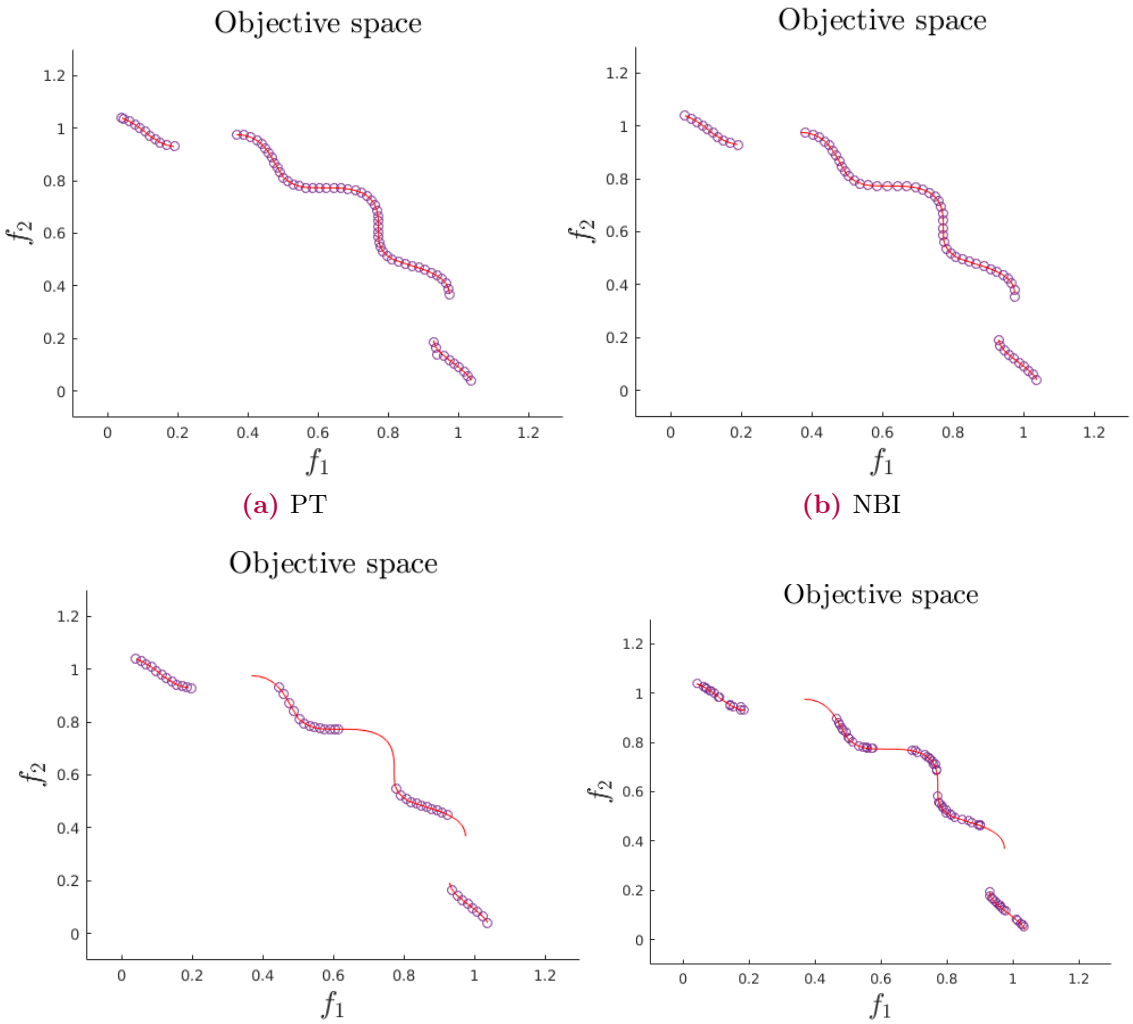


(c) ε-constraint method



(d) NSGA-II

Figure 4.8: Results in decision space for the Tanaka problem (4.3).



(a) PT **(b)** NBI

(c) ε-constraint method **(d)** NSGA-II

Figure 4.9: Results in the objective space for the Tanaka problem (4.3).

Table 4.5: Computational efforts for the Tanaka problem (4.3).

	PT	NBI	ε-constraint	NSGA-II
Solutions	70	63	70	100
Function Evaluations	606	488	2114	20000
Jacobian Evaluations	207	488	1022	-
Hessian Evaluations	-	418	998	-
Total of Evaluations	1434	9128	22170	20000
Δ_2	0.0154	0.0187	0.0516	0.0364

Table 4.6: Parameters used by NSGA-II for the Tanaka problem (4.3).

Population size	100
Number of generations	200
Probability of crossover	0.9
Probability of mutations	0.5

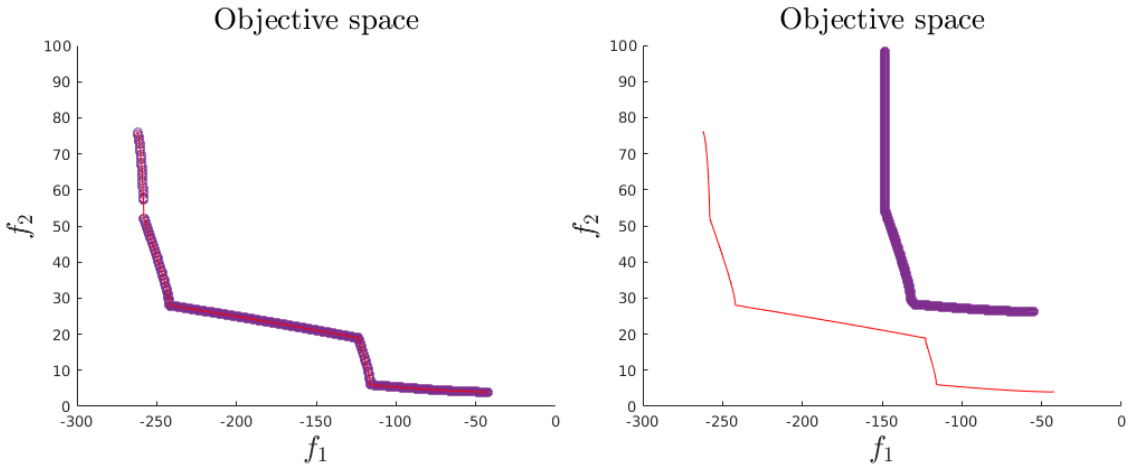
4.4 Example 4: Osykzka and Kundu Problem

Our next test example is the BOP (4.4) [38], which depends on five variables and is subject to four linear inequality constraints and two nonlinear inequality constraints. Figure 4.10 presents the approximation of the solution sets of this problem obtained with the four different algorithms. Table 4.7 shows a comparison of the computational effort for all algorithms. As it can be seen, NBI and $\varepsilon-$constraint show the worst performances, their approximations in objective space are not in the real solution set, instead they are far enough of it. Note that the performance of PT is considerably better than the performance of NSGA-II. The computational effort of PT clearly outperforms the other algorithms. Neverthless, to obtain the solutions shows for PT we had to run the algorithm four times with four different starting points, opposed to the others where we need only one run of the algorithm.

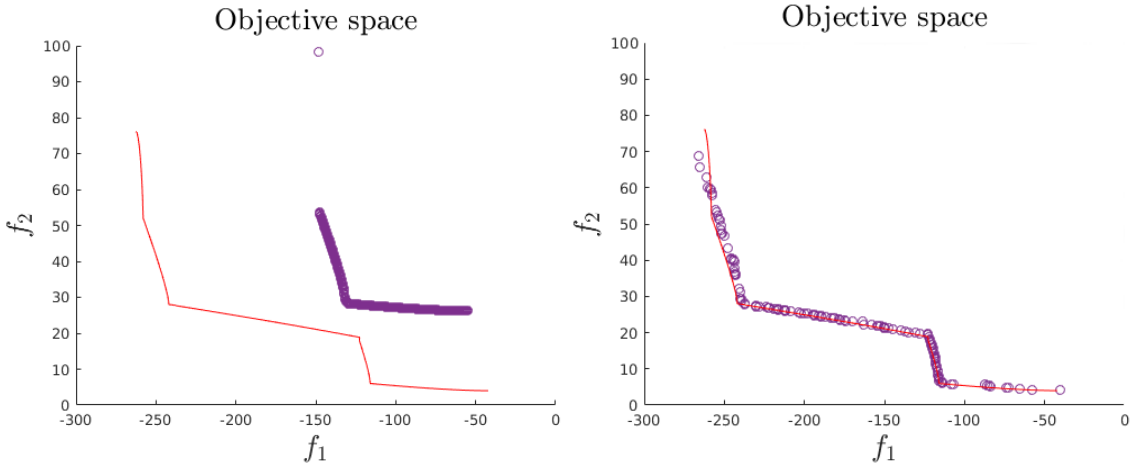
$$
\begin{aligned}
\min \quad & \begin{cases} f_1(x) & = -25(x_1-2)^2-(x_2-2)^2-(x_3-1)^2-(x_4-4)^2-(x_5-1)^2 \\ f_2(x) & = \sum_{i=1}^{6} x_i^2 \end{cases} \\
\text{s.t.} \quad & x_1+x_2-2 \geq 0 \\
& 6-x_1-x_2 \geq 0 \\
& 2-x_2+x_1 \geq 0 \\
& 2-x_1+3x_2 \geq 0 \\
& 4-(x_3-3)^2-x_4 \geq 0 \\
& (x_5-3)^2+x_6-4 \geq 0 \\
\text{with} \quad & 0 \leq x_1, x_2, x_6 \leq 10 \\
& 1 \leq x_3, x_5 \leq 5 \\
& 0 \leq x_4 \leq 6
\end{aligned}
\tag{4.4}
$$

Table 4.7: Computational efforts for the Osykzka and Kundu problem (4.4).

	PT	NBI	ε-constraint method	NSGA-II
Solutions	435	634	634	145
Function Evaluations	2051	11279	6606	20000
Jacobian Evaluations	850	11208	6570	-
Hessian Evaluations	-	10574	5936	-
Total of Evaluations	5451	479071	270326	20000
Δ_2	0.1801	60.7296	60.8988	2.8244



(a) PT **(b)** NBI



(c) ε-constraint method **(d)** NSGAII

Figure 4.10: Results in decision space for the Osykzka and Kundu problem (4.4).

Table 4.8: Parameters used by NSGA-II for the Osykzka and Kundu problem (4.4).

Population size	400
Number of generations	50
Probability of crossover	0.9
Probability of mutations	0.5

4.5 CTP1 Problem

Next we consider the CTP1 problem (4.5) [16], which depends on two variables and is subject to two nonlinear inequality constraints. Figure 4.11 shows the feasible region of this problem. Figures 4.12 and 4.13 present the approximation of the solution sets of this problem obtained with the four different algorithms. Table 4.9 shows a comparison of the computational effort for all algorithms. As it can be seen all the algorithms obtain almost perfect approximations in objective space, being the performance of NSGA-II the best of all. Note, however than NSGA-II and the $\varepsilon-$constraint method generate around 100 solutions opposed to the other two that generate around 60 solutions. Regarding the computational effort PT clearly outperforms the other algorithms. Nevertheless, to obtain the solutions shows for PT we had to run the algorithm four times with four different starting points, opposed to the others where we need only one run of the algorithm.

$$\begin{aligned} \min \quad & \begin{cases} f_1(x) & = x_1, \\ f_2(x) & = (1+x_2)\exp\left(-\frac{x_1}{1+x_2}\right), \end{cases} \\ \text{s.t.} \quad & \frac{f_2(x)}{0.858\exp\left(-0.541 f_1(x)\right)} \geq 1, \\ & \frac{f_2(x)}{0.728\exp\left(-0.295 f_1(x)\right)} \geq 1, \\ \text{with} \quad & 0 \leq x_1, x_2 \leq 1. \end{aligned} \tag{4.5}$$

Table 4.9: Computation efforts for the CTP1 problem (4.5).

	PT	NBI	ε-constraint	NSGA-II
Solutions	52	76	110	100
Function Evaluations	130	808	830	4000
Jacobian Evaluations	88	808	830	-
Hessian Evaluations	-	698	720	-
Total of Evaluations	482	4754	15670	4000
Δ_2	0.0130	0.0094	0.0809	0.0683

Table 4.10: Parameters used by NSGA-II for CTP1 problem (4.5).

Population size	100
Number of generations	40
Probability of crossover	0.9
Probability of mutations	0.5

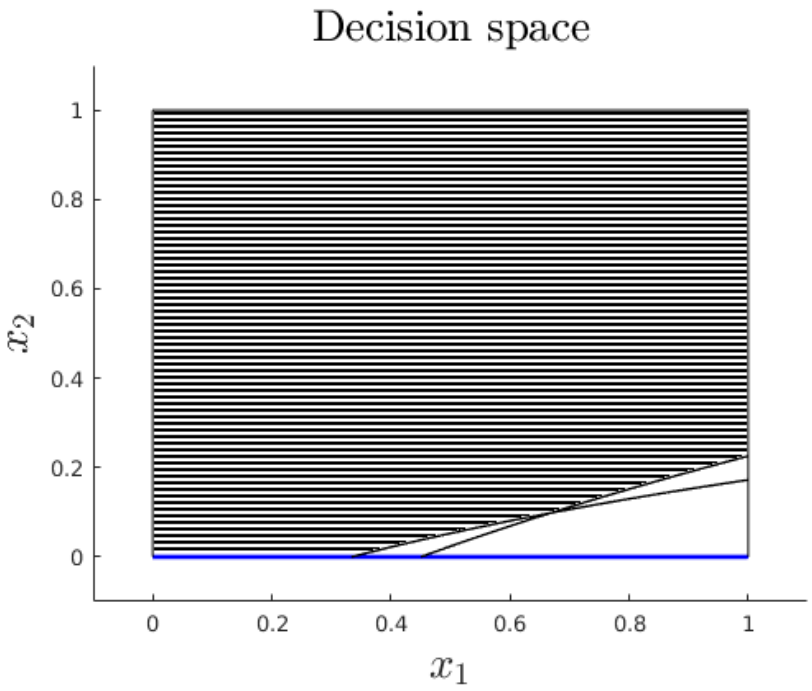
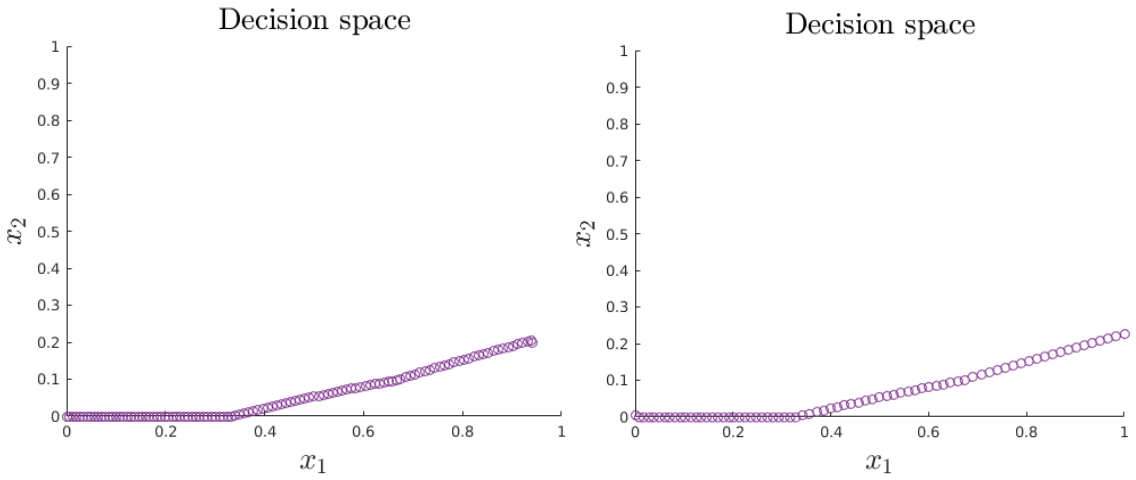
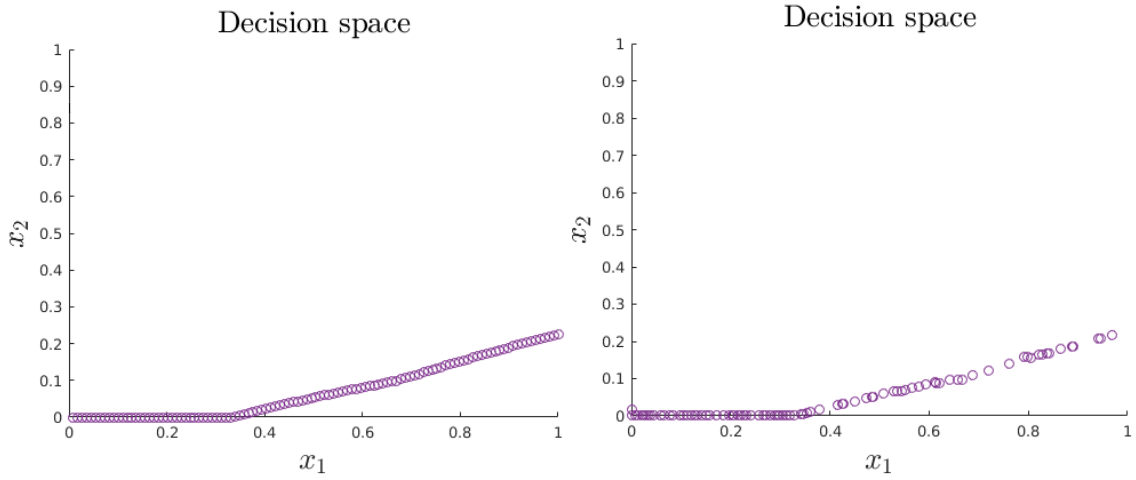


Figure 4.11: Feasible region of the CTP1 problem (4.5).

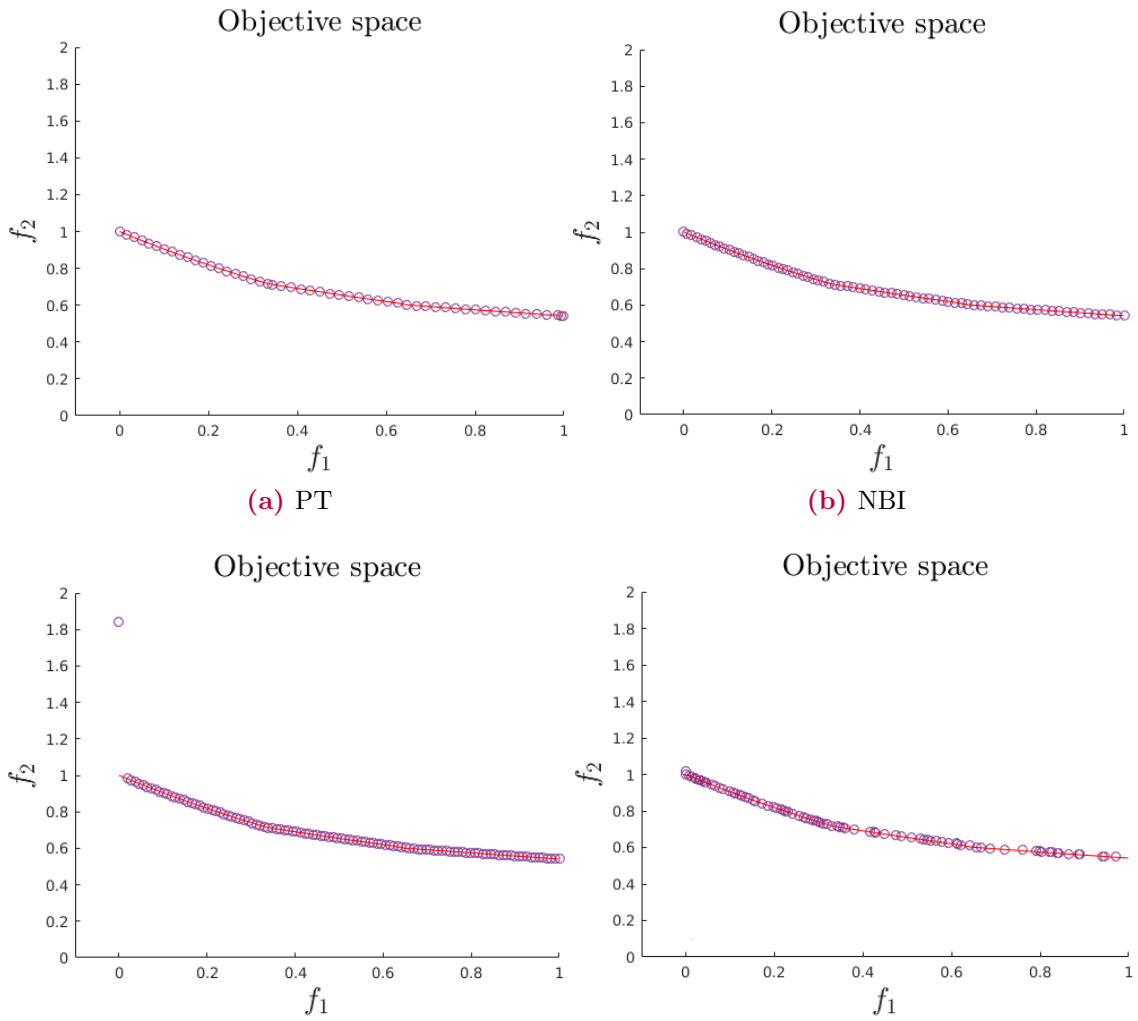


(a) PT **(b)** NBI



(c) ε-constraint method **(d)** NSGAII

Figure 4.12: Results in decision space for the CTP1 problem (4.5).



(a) PT **(b)** NBI

(c) ε-constraint method **(d)** NSGAII

Figure 4.13: Results in objective space for the CTP1 problem (4.5).

4.6 Const-Ex Problem

Our next test problem is the BOP (4.6) [14], which depends on two variables and is subject to two linear inequality constraints. Figure 4.14 shows the feasible region of this problem. Figures 4.15 and Figure 4.13 present the approximation of the solution sets of this problem obtained with the four different algorithms. Table 4.11 shows a comparison of the computational effort for all algorithms. As it can be seen, PT and NSGA-II obtain almost perfect approximations in objective space while the $\varepsilon-$constraint method and NBI leave out some of the solutions. The performance of PT is the best of all, also, this algorithm considerably outperforms the other algorithms in terms of computational effort.

$$\begin{aligned} \min \quad & \begin{cases} f_1(x) & = x_1, \\ f_2(x) & = \frac{1+x_2}{x_1}, \end{cases} \\ \text{s.t.} \quad & x_2 + 9x_1 \geq 6, \\ & -x_2 + 9x_1 \geq 1, \\ \text{with} \quad & 0.1 \leq x_1 \leq 1, \\ & 0 \leq x_2 \leq 5. \end{aligned} \tag{4.6}$$

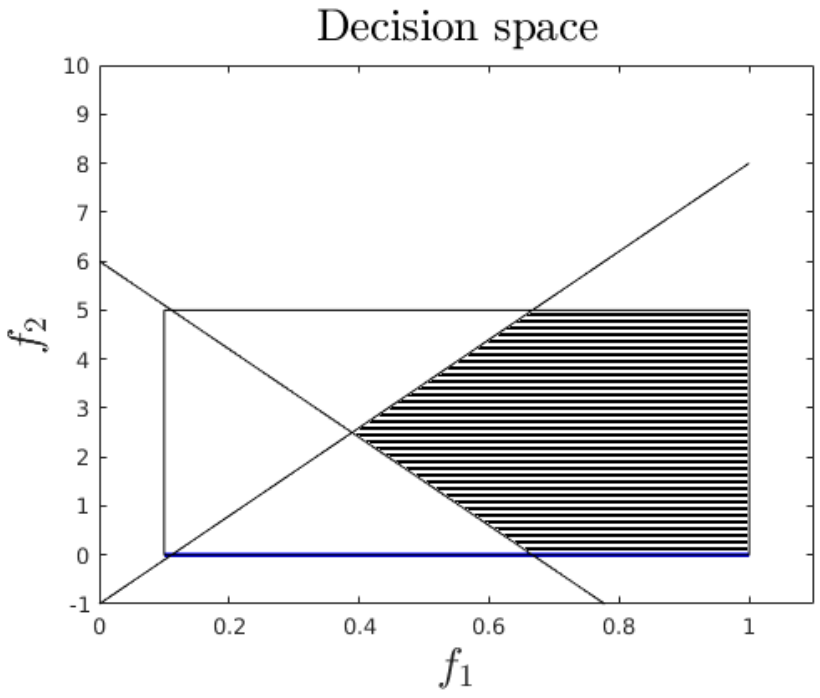
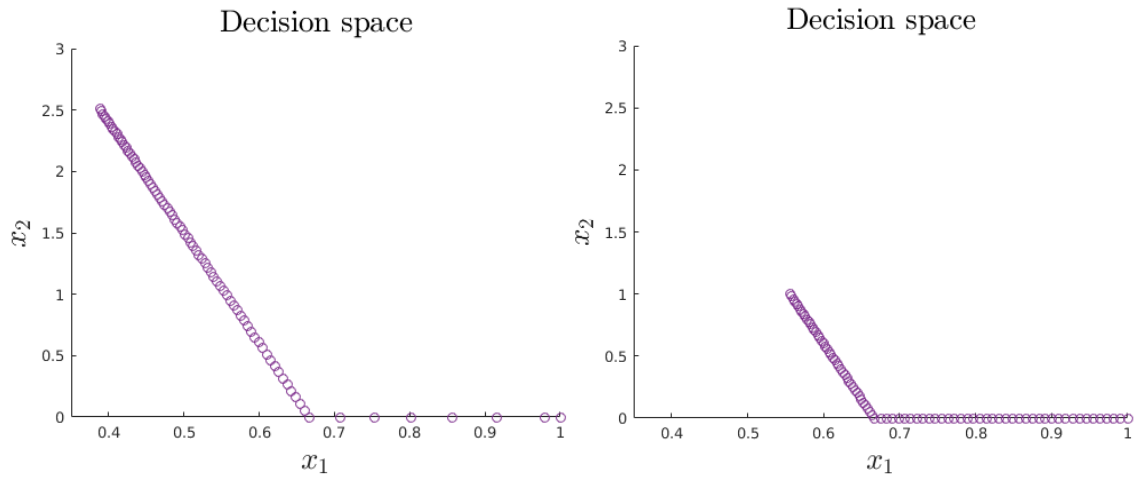
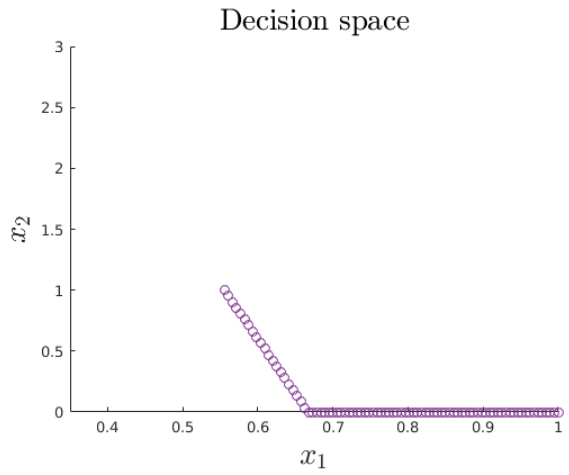


Figure 4.14: Feasible region of ConstEx problem (4.6).

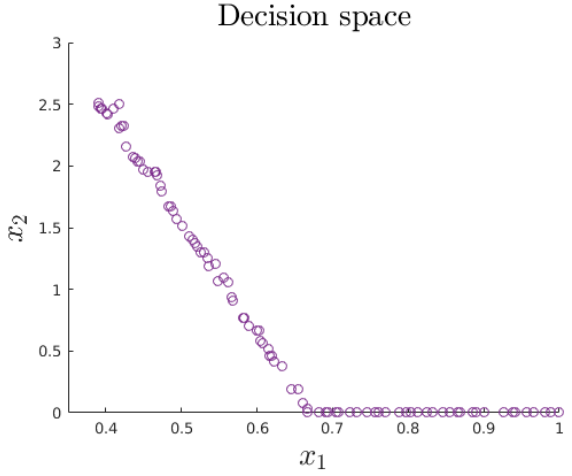


(a) PT

(b) NBI

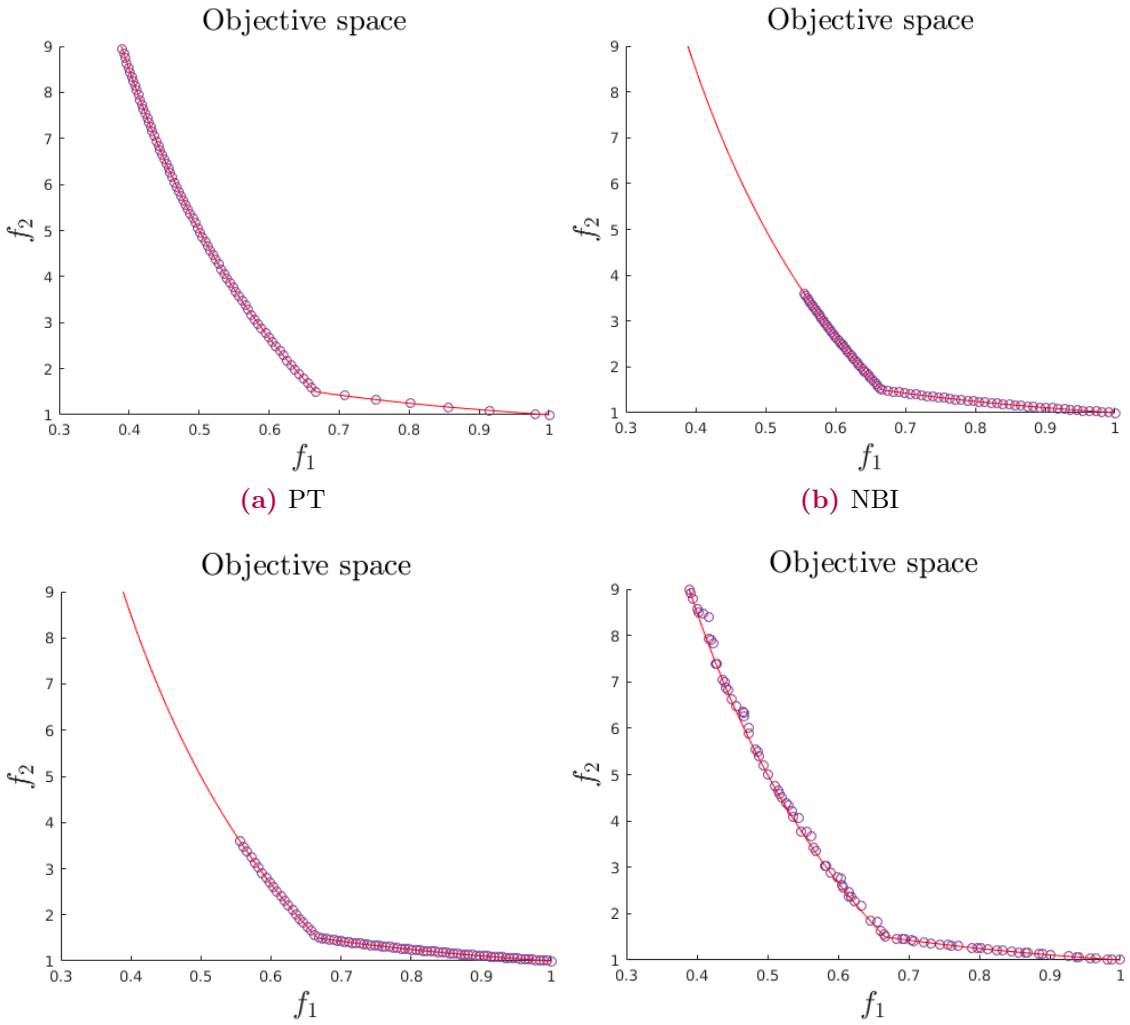


(c) ε-constraint method



(d) NSGAII

Figure 4.15: Results in decision space for the Const-Ex problem (4.6).



(a) PT **(b)** NBI

(c) ε-constraint method **(d)** NSGA-II

Figure 4.16: Results in objective space for the Const-Ex problem (4.6).

Table 4.11: Computational efforts for the Const-Ex problem (4.6).

	PT	NBI	ε-constraint	NSGA-II
Solutions	85	84	84	100
Function Evaluations	108	593	664	4000
Jacobian Evaluations	88	590	663	-
Hessian Evaluations	-	56	579	-
Total of Evaluations	460	3849	12580	4000
Δ_2	0.0286	2.5487	2.5488	0.0706

Table 4.12: Parameters used by NSGA-II for the Const-Ex problem (4.6).

Population size	100
Number of generations	40
Probability of crossover	0.9
Probability of mutations	0.5

4.7 Tamaki Test Problem

Next we consider the problem of Tamaki [33], which depends on three variables and is subject to one nonlinear inequality constraint. Figures 4.17 and Figure 4.18 present the approximation of the solution sets of this problem obtained with the three different algorithms (the implementation of the ε- constraint method that we were using [30] is not able to solve problems with more than two objectives). Table 4.13 shows a comparison of the computational effort for all algorithms. As it can be seen, PT and NBI both obtain almost perfect approximations, the performance of PT is slightly better than NBI. Note, however, that PT generates approximately 3 times the number of solutions than NBI. Regarding the computational effort PT clearly outperforms the other two algorithms.

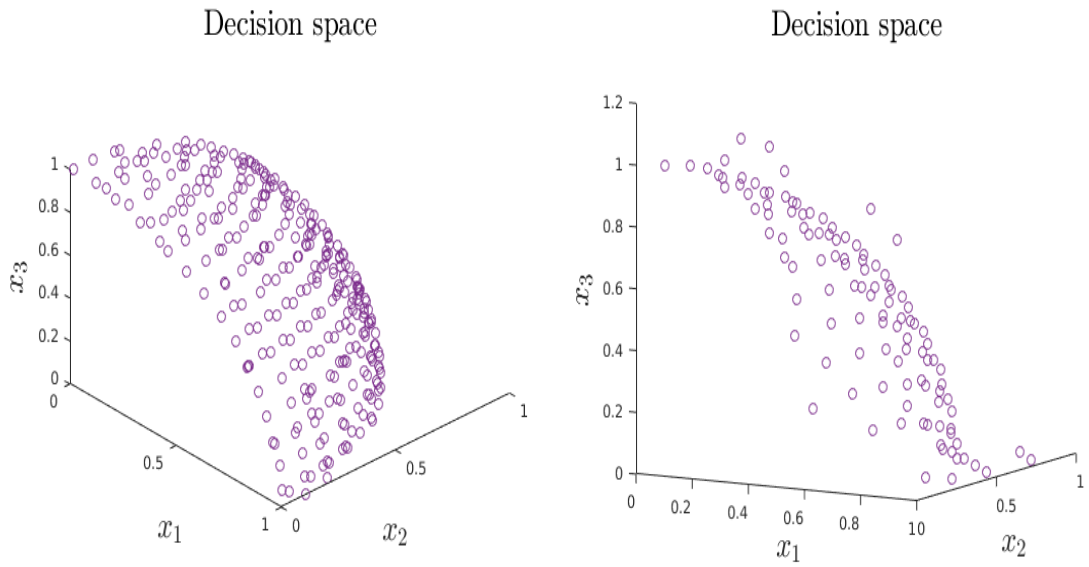
$$\begin{aligned} \min \quad & \begin{cases} f_1(x) & = x_1, \\ f_2(x) & = x_2, \\ f_3(x) & = x_3, \end{cases} \\ \text{s.t.} \quad & x_1^2 + x_2^2 + x_3^2 \geq 0, \\ \text{with} \quad & 0 \leq x_1 \leq 4. \end{aligned} \tag{4.7}$$

Table 4.13: Computational efforts for the Tamaki problem (4.7).

	PT	NBI	ε-constraint	NSGA-II
Solutions	305	112	N/A	52
Function Evaluations	2498	3758	N/A	50000
Jacobian Evaluations	1101	3758	N/A	-
Hessian Evaluations	-	3293	N/A	-
Total of Evaluations	6902	91236	N/A	500000
Δ_2	0.0380	0.6353	N/A	1.0204

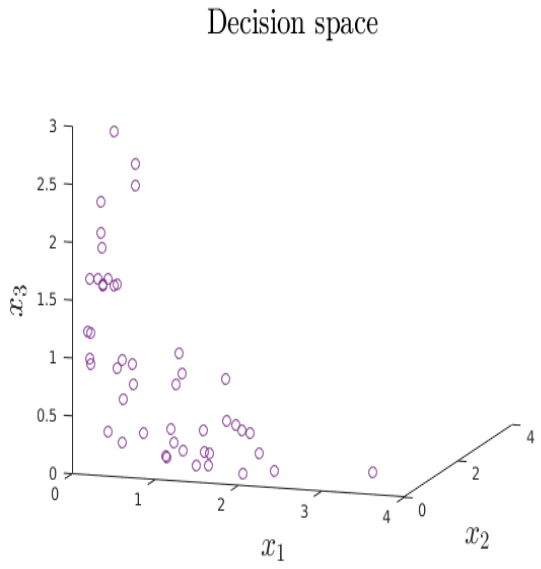
Table 4.14: Parameters used by NSGA-II for the Tamaki problem (4.7).

Population size	100
Number of generations	500
Probability of crossover	0.9
Probability of mutations	0.5



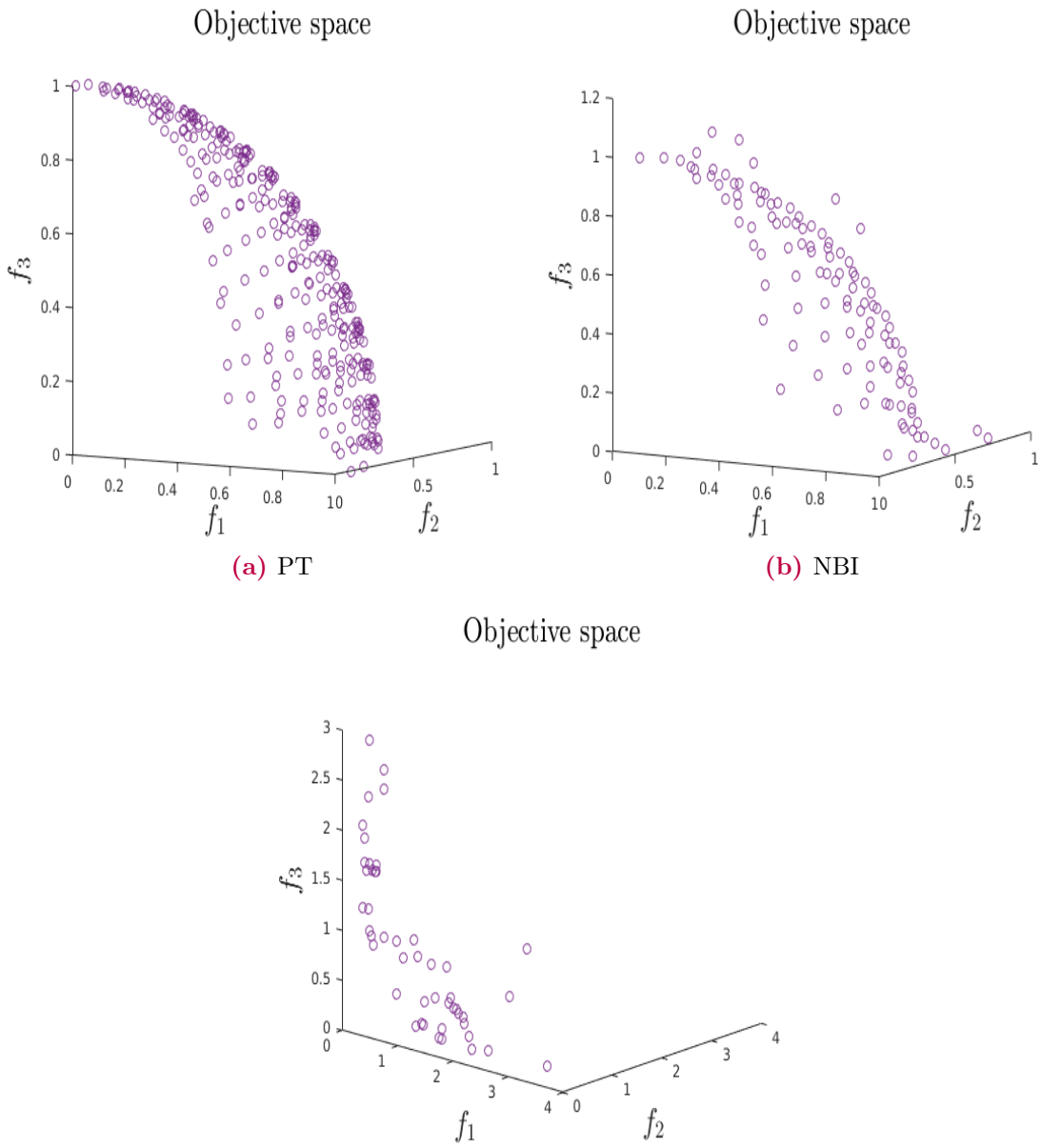
(a) PT

(b) NBI



(c) NSGA-II

Figure 4.17: Results in decision space for the Tamaki problem (4.7).



(a) PT

(b) NBI

(c) ε-constraint method

Figure 4.18: Results in objective space for the Tamaki problem (4.7).

4.8 Three Objective Test Problem

For our last test problem we propose the MOP (4.8), which depends on three variables and is subject to one linear equality constraint and one nonlinear inequality constraint. For this problem we obtained feasible solutions only using PT. Figures 4.19 and Figure 4.20 present the approximation of the solution sets obtained with PT. Table 4.8 shows a comparison of the computational effort for all algorithms. For this problem we can not compare the performance of the algorithms, instead, according with the indicator Δ_2 we can say that the performance of PT is good enough. Regarding the computational effort PT clearly outperforms the other algorithms.

$$\begin{aligned} \min \begin{cases} f_1(x) = (x_1+3)^2 + (x_2+3)^2 + (x_3+3)^2, \\ f_2(x) = (x_1-9)^2 + (x_2+5)^2 + (x_3+5)^2, \\ f_3(x) = (x_1-5)^2 + (x_2-8)^2 + x_3^2, \end{cases} \\ \text{s.t.} \quad x_1 - 2x_2 - 3x_3 = 0, \\ \sin(2x_1) - x_2 \leq 0. \end{aligned} \tag{4.8}$$

Table 4.15: Computation efforts for the proposed test problem (4.8).

	PT	NBI	ε-constraint	NSGA-II
Solutions	378	0	N/A	0
Function Evaluations	1923	2290	N/A	50000
Jacobian Evaluations	756	1641	N/A	-
Hessian Evaluations	-	1431	N/A	-
Total of Evaluations	4947	40336	N/A	50000
Δ_2	2.0658	-	N/A	-

Table 4.16: Parameters used by NSGA-II for the proposed test problem (4.8).

Population size	100
Number of generations	500
Probability of crossover	0.9
Probability of mutations	0.5

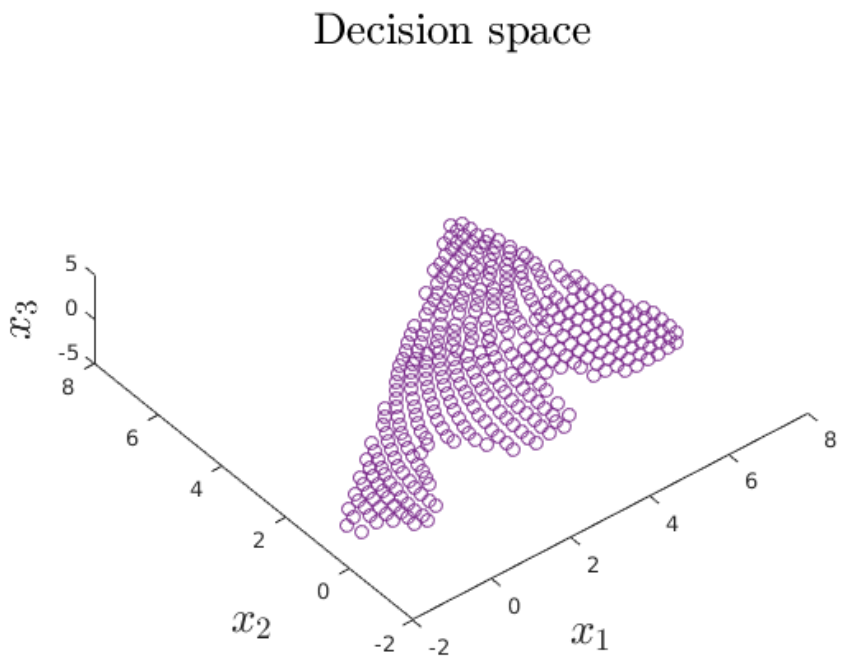


Figure 4.19: Result in the decision space for the proposed test problem (4.8).

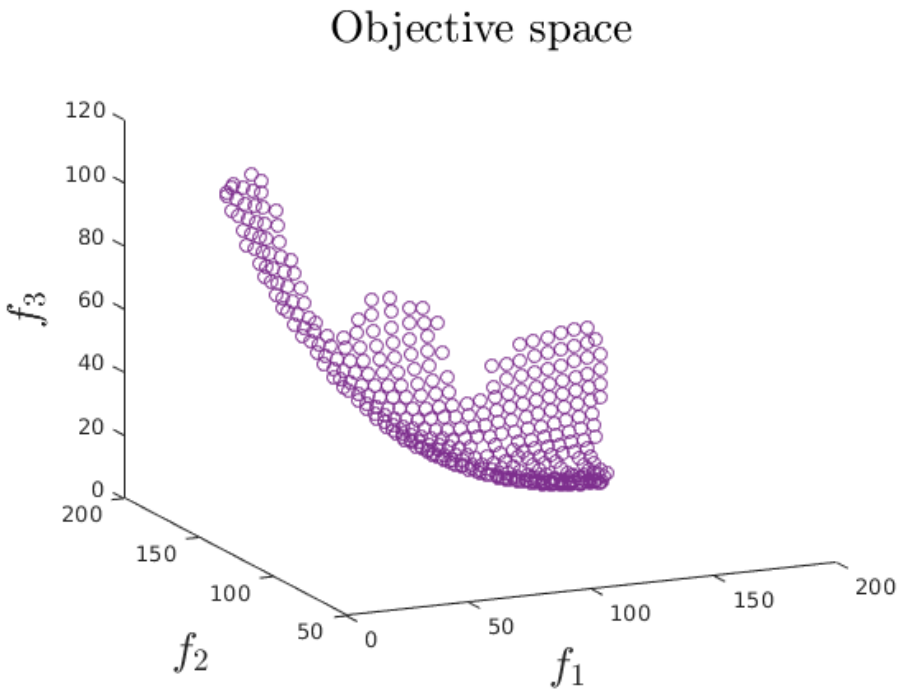


Figure 4.20: Result in objective space for the proposed test problem (4.8).

Chapter 5

Conclusions and Future Work

In this chapter we summarize the work developed in this thesis, discuss its findings and contributions, and point out its limitations. We also outline directions for future research.

First, we shortly describe the principal contributions of this work. As stated in Chapter 3, PT is a predictor corrector method for solving MOPs. PT can handle equality constraints and box constrains but is not able to handle general inequality constraints. In this tesis work we modified PT so that it can reliably, leading to a new continuation method that is state of the art.

For the algorithm presented in this work we modified the corrector step of the existing PT. More precisely, we consider particular inequality constraints as equality ones. In every corrector step of the method we evaluate all the inequality constraints and we decide which of the equality constraints we need to consider as well.

We tested our algorithm on academic benchmark problems and we observed a considerably better performance of our algorithm compared with the other ones. It is important to remark that with less function evaluations our algorithm is able to compute solutions sets as good as the solution sets computed by the other algorithms. We also observed that improvements over the other algorithms increase as the number of objectives in the MOP increases.

As conclusion we can consider PT as a highly competitive algorithm for the treatment of constrained MOPs.

Future Work

There are several lines of research arising from this work that may be pursued in the future. Following we list some of them.

- Hybridization of PT with multi-objective evolutionary algorithms (MOEAs): PT is of local nature, and is hence in need of suitable starting points. Further, for each given solution, the PT is restricted to the connected component of the Pareto set/front that contains this starting point. A hybridization with MOEAs that are of global nature is a possible remedy.

- Application to real world problems: finally, we would like to demonstrate the strength of PT (and/or the related hybrid that uses as local search engine) on more realistic problems. We conjecture that the novel method will be highly advantageous in particular for more complex problems that contains several constraints, and are defined in higher dimensional search spaces.

Bibliography

[1] Eugene Allgower and Kurt Georg, *Introduction to numerical continuation methods*, Springer, 1990.

[2] To Thanh Binh and Ulrich Korn, *Mobes: A multiobjective evolution strategy for constrained optimization problems*, The Third International Conference on Genetic Algorithms (Mendel 97), vol. 25, 1997, p. 27.

[3] Johan Bogoya, Andrés Vargas, and Oliver Schütze, *The averaged hausdorff distances in multi-objective optimization: A review*, Mathematics **7** (2019), no. 10, 1–35.

[4] Jürgen Branke, Kalyanmoy Deb, Kaisa Mietten, and Roman Slowinski, *Multiobjective optimziation*, Springer, 2008.

[5] R. Burachik, C. Kaya, and M. Rizvi, *A new scalarization technique to approximate pareto fronts of problems with disconnected feasible sets*, Journal of Optimization Theory and Applications **162** (2014), 428–446.

[6] Edmund Kieran Burke and Graham Kendall, *Search methodologies*, Springer, 2014.

[7] Vira Chankong and Yacov Haimes, *Multiobjective decision making: theory and methodology*, Courier Dover Publications, 2008.

[8] Kenneth Chircop and David Zammit-Mangion, *On epsilon-constraint based methods for the generation of Pareto frontiers*, David Publishing, 2012.

[9] Carlos Coello Coello and Gary Lamount, *Applications of multi-objective evolutionary algorithms*, World Scientific Publishing, 2004.

[10] Antonio Conejo, Enrique Castillo, Roberto Mínguez, and Raquel García-Bertrand, *Decomposition techniques in mathematical programming*, Springer, 2006.

[11] George Corliss, Christele Faure, Andreas Griewank, Laurent Hascoet, and Uwe Naumann, *Automatic differentiation of algorithms: from simulation to optimization*, Springer Science & Business Media, 2002.

[12] Oliver Cuate, Lourdes Uribe, Antonin Ponsich, Adriana Lara, Fernanda Beltran, Alberto Rodríguez Sánchez, and Oliver Schütze, *A new hybrid metaheuristic for equality constrained bi-objective optimization problems*, International Conference on Evolutionary Multi-Criterion Optimization, Springer, 2019, pp. 53–65.

[13] Indraneel Das and John Dennis, *Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems*, SIAM Journal on Optimization **8** (1996), 631–657.

[14] Kalyanmoy Deb, *Multi-objective optimization using evolutionary algorithms*, vol. 16, John Wiley & Sons, 2001.

[15] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Transactions on Evolutionary Computation **6** (2002), no. 2, 182–197.

[16] Kalyanmoy Deb, Amrit Pratap, and T. Meyarivan, *Constrained test problems for multi-objective evolutionary optimization*, International conference on evolutionary multi-criterion optimization, Springer, 2001, pp. 284–298.

[17] J. Dennis and R. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equations*, SIAM, 1196.

[18] Matthias Ehrgott, *Multicriteria optimization*, Springer, 2005.

[19] Matthias Ehrgott and David Ryan, *Constructing robust crew schedules with bicriteria optimization*, Journal of Multi-Criteria Decision Analysis **11** (2002), no. 3, 139–150.

[20] Gabriele Eichfelder, *Adaptive scalarization methods in multiobjective optimization*, Springer, 2008.

[21] J. Fliege, L. Drummond, and B. Svaiter, *Newton's method for multi-objective optimization*, SIAM Journal on Optimization **20** (2009), no. 2, 602–626.

[22] Jörg Fliege, *Gap-free computation of Pareto-points by quadratic scalarizations*, Mathematical Methods of Operations Research **59** (2004), no. 1, 69–89.

[23] Jörg Fliege and Benar Fux Svaiter, *Steepest descent methods for multicriteria optimization*, Mathematical Methods of Operations Research **51** (2000), no. 3, 479–494.

[24] Saul Gass and Thomas Saaty, *The computational algorithm for the parametric objective function*, Naval Research Logistics **1-2** (1995), no. 6, 39–45.

[25] Arthur Geoffrion, *Proper efficiency and the theory of vector maximization*, Journal of Mathematical Analysis and Applications **22** (1968), no. 3, 618 – 630.

[26] Gene Golub and Charles Van Loan, *Matrix Computations*, The Johns Hopkins Studies in the Mathematical Sciences, 2013.

[27] Y. Haimes, Leon Lasdon, and D. Wismer, *On a bicriterion formulation of the problems of integrated system identification and system optimization*, IEEE Transactions on Systems, Man, and Cybernetics - TSMC **1** (1971), 296–297.

[28] Juha Heinonen, *Lectures on analysis on metric spaces*, Springer, 01 2001.

[29] Claus Hillermeier, *Nonlinear Multiobjective optimization*, Birkhauser, 2001.

[30] Julia, *JuMP — Julia for Mathematical Optimization*, http://www.juliaopt.org/JuMP.jl/v0.14/, Accessed: 2020-01-12.

[31] William Karush, *Minima of functions of several variables with inequalities as side constraints*, PhD thesis, Masters thesis, Dept. of Mathematics, Univ. of Cicago, 1939.

[32] Harold William Kuhn and Albert William Tucker, *Nonlinear programming*, Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, 1951.

[33] Mahamat Maimos, *Alienor method for nonlinear multi-objective optimization*, Applied Mathematics **02** (2011), 217–224.

[34] R. Marler and Jasbir Singh Arora, *The weighted sum method for multi-objective optimization: New insights*, Structural and Multidisciplinary Optimization **41** (2010), no. 6, 853–856.

[35] Benjamin Martin, Alexandre Goldsztejn, Laurent Granvilliers, and Christophe Jermann, *On continuation methods for non-linear bi-objective optimization: Certified interval-based approach*, Journal of Global Optimization **64** (2014), 1–16.

[36] Adanay Martín and Oliver Schütze, *Pareto Tracer: a predictor–corrector method for multi-objective optimization problems*, Engineering Optimization **50** (2018), no. 3, 516–536.

[37] Jorge Nocedal and Sthephen Junius Wright, *Numerical Optimization*, Springer, 2006.

[38] Andrzej Osyczka and Sourav Kundu, *A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm*, Structural optimization **10** (1995), no. 2, 94–99.

[39] Victor Pereyra, Michael Saunders, and Jose Castillo, *Equispaced pareto front construction for constrained bi-objective optimization*, Mathematical and Computer Modelling **57** (2013), no. 9-10, 2122–2131.

[40] Oliver Schutze, Xavier Esquivel, Adriana Lara, and Carlos Coello Coello, *Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization*, IEEE Transactions on Evolutionary Computation **16** (2012), no. 4, 504–522.

[41] Oliver Schütze, Adriana Lara, and Carlos Coello Coello, *The directed search method for unconstrained multi-objective optimization problems*, EVOLVE — A bridge between probability, set orientednumerics, and evolutionary computation (2011), 1–24.

[42] Masahiro Tanaka, Hikaru Watanabe, Yasuyuki Furukawa, and Tetsuzo Tanino, *Ga-based decision support system for multicriteria optimization*, IEEE International Conference on Systems Man and Cybernetics, vol. 2, INSTITUTE OF ELECTRICAL ENGINEERS INC (IEEE), 1995, pp. 1556–1561.

[43] Honggang Wang, *Zigzag search for continuous multiobjective optimization*, INFORMS Journal on Computing **25** (2013), 654–665.

[44] Lotfi Zadeh, *Optimality and non-scalar-valued performance criteria*, IEEE Transactions on Automatic Control **8** (1963), no. 1, 59–60.