Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Sobre el uso de Métodos de Continuación para el tratamiento numérico de MaOPs

Tesis que presenta

Oliver Fernando Cuate González

para obtener el Grado de

Maestro en Ciencias en Computación

Director de la Tesis

Dr. Oliver Steffen Schütze

México, D.F. **Diciembre 2015**

Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Zacatenco Campus

Computer Science Department

On the Use of Continuation Methods for the Numerical Treatment of MaOPs

Submitted by

Oliver Fernando Cuate González

As a fulfillment of the requirement for the degree of

Master in Computer Science

Advisor

Dr. Oliver Steffen Schütze

México, D.F. **December, 2015**

Resumen

El *problema de optimización multiobjetivo* (MOP, por sus siglas en inglés), surge de manera natural en diversas áreas del conocimiento, como Economía, Finanzas y principalmente en la Industria; en las cuales se requiere optimizar simultáneamente más de una función objetivo. Una de las principales características de un MOP es que su conjunto de soluciones, llamado *Conjunto de Pareto*, típicamente forma un objeto de dimensión $(k-1)$, en donde k es el número de objetivos involucrados. En la actualidad, es posible aproximar el conjunto completo de interés para un número relativamente bajo de funciones objetivo (por ejemplo, para $k = 3$ o 4). Sin embargo, para problemas con más de 4 funciones objetivo, los cuales son conocidos como *Many-Objective Optimization Problems* (MaOPs), es necesario diseñar mecanismos específicos para aproximar sus soluciones. Recientemente, los MaOPs han captado el interés de la industria debido al éxito de los métodos existentes y a su impacto en los procesos de toma de decisiones, cada vez más complejos.

En esta tesis se propone el *Pareto Explorer* (PE), una herramienta de naturaleza global/local para el tratamiento numérico de MaOPs. PE consta de dos fases principales: el cálculo de una (o varias) soluciones óptimas globales y la exploración de soluciones óptimas a nivel local a través de un método de continuación numérica multiobjetivo. Este método se adapta al contexto de un MaOP dado y permite dirigir la búsqueda en cualquier dirección dada por el tomador de decisiones. De esta forma, es posible explorar un MaOP ya sea en el espacio de decisión, el de los objetivos, o el de los pesos asociados. Finalmente, se presentan los resultados en algunos problemas de referencia, así como en un problema de 14 objetivos que surge en el diseño de un sistema de lavandería.

Abstract

In many areas such as Economy, Finance,or Industry the problem arises naturally that several objectives have to be optimized concurrently. Mathematically speaking this leads to a *Multiobjective Optimization Problem* (MOP). One important characteristic of MOPs is that its solution set, the *Pareto Set* (PS), typically forms a $(k-1)$-dimensional object where k is the number of objectives involved in the MOP. Thus, it is only possible to approximate the entire set of interest for relatively few number of objectives (say, $k = 3$ or 4). In this work, we address the numerical treatment of MOPs with more than 4 objectives which are also termed as *Many Objective Optimization Problem*s (MaOPs). MaOPs have recently caught the interest in Industry due to the huge success of existing methods for the treatment of MaOPs and since decision making processes are getting more and more complex.

In this thesis, we propose the *Pareto Explorer* (PE), a global/local tool for the numerical treatment of MaOPs. The PE consists of two principal phases: Phase I consists of the computation of one (or several) globally optimal solutions of the given MaOP. In Phase II, the set of optimal solutions is locally explored via a multiobjective continuation method that we tailor to the given context. This continuation method allows to steer the search into any direction given by the *Decision Maker* (DM), e.g., in decision, objective, or weight space of the given problem. We present results on some benchmark models as well as on a 14-objective problem that arises in the design of a laundry system.

A mi madre, Consuelo,
este es un logro de ambos.

Agradecimientos

Agradezco a mi asesor, Dr. Oliver Schütze, por el interés mostrado hacia a este trabajo; sus conocimientos, observaciones y críticas, han sido parte fundamental del mismo y lo han llevado a buen puerto. También por la confianza y la paciencia mostrada a lo largo de este tiempo, así como por abrirme las puertas al mundo de la investigación, lo que me ha permitido conocer nuevos lugares y personas con quienes pude compartir conocimiento y aprender cada día.

De forma muy especial agradezco a mi madre, Consuelo Cuate, por su apoyo incondicional, su amor y su comprensión en todas las etapas de mi vida. Gracias por siempre dar lo mejor de tí, es una motivación constante a hacer lo mismo. Agradezco también al resto de mi familia, siempre presente, en especial a mis tíos, Francisco y María, por permitirme quedarme en su hogar cuando fue necesario.

A CONACyT por el apoyo económico proporcionado durante mis estudios de maestría, además del apoyo otorgado para mi estancia en el extranjero. A CINVESTAV-IPN por todas las facilidades brindadas como una gran institución, así como por los apoyos para estancia en el extranjero y beca terminal.

A mis sinodales, Dr. Amilcar Meneses, por los importantes aportes realizados a este trabajo; y Dra. Adriana Lara, quien de igual forma contribuyó enormemente a este trabajo, y que además ha estado presente desde hace varios años con consejos siempre oportunos, que han repercutido de manera importante en mis decisiones, y mostrando en todo momento su calidez y su grandeza como persona.

A mis amigos, David y Jhonatan, por su compañía en esta etapa, siempre en las buenas y en las malas. A Isaac, un excelente ser humano de quien he aprendido mucho, tanto en el aspecto académico como en el personal, y que me mostró la belleza de un deporte como el tenis. A Alejandro, por toda la ayuda brindada en los primeros meses de la maestría. A Belem y Gabriel, por estar conmigo y darme ánimos cuando más lo necesité. Gracias a todos mis compañeros de generación, me llevo un poco de cada uno de ustedes.

A mi amiga Lulú, por su agradable compañía durante esta etapa, por todas sus aportaciones a este trabajo y por soportarme en mis malos momentos. A mis amigos Carlos y Saúl, por nunca perder la actitud y vernos frecuentemente. A mis amigos en general, son parte importante en mi vida.

A los doctores Debrub Chakraborty, Francisco Rodríguez, Carlos Coello y Gerardo de la Fraga por sus excelentes clases, el esfuerzo puesto en cada una de ellas fue mucho y muy gratificante. Al Dr. Michael Dellnitz, a cargo de mi estancia en la Universidad de Paderborn y a Sebastian, mi compañero de trabajo durante este periodo; gracias por hacer posible una de las mejores experiencias que he vivido.

A Sofía y a Felipa, por estar pendiente de todos los estudiantes y apoyarnos siempre con la mejor disposición. A todo el personal del Departamento de Computación, por hacer aún más agradable mi estancia en este lugar.

Gracias a todos los que de una forma u otra formaron parte de esta experiencia.

Contents

List of Figures

List of Tables

Acronyms

BFGS: *Quasi-Newton method of Broyden, Fletcher, Goldfarb, and Shanno*

DM: *Decision Maker*

DRPP: *Dynamic Reference Point Problem*

DTLZ: *Deb-Thiele-Laumanns-Zitzler*

GUI: *Graphical User Interface*

HV: *Hypervolume*

KKT: *Karush, Kuhn and Tucker*

LP: *Laundering Problem*

MaOP: *Many Objective Optimization Problem*

MOP: *Multiobjective Optmization Problem*

PE: *Pareto Explorer*

PE-N: *Pareto Explorer using Newton method as corrector*

PE-QN: *Pareto Explorer using Quasi-Newton method as corrector*

PF: *Pareto Front*

PS: *Pareto Set*

QN: *Quasi-Newton*

SOP: *Scalar Optmization Problem*

Chapter 1

Introduction

Optimization problems with multiple objectives arise naturally in areas as Economy, Finance and Industry, where it is necessary to obtain the greatest profit of the limited resources. Here, the resources are the decision variables and the objectives are the objectives functions. These functions are generally in conflict with each other. For example, the increment for the quality of certain product usually means an increment of the production costs. Nowadays, there exist a lot of methods used to solve such *Multiobjective Optimization Problem*s (MOPs) [1], each one with advantages and disadvantages, looking increasingly to efficiently solve more general problems. The solution to these problems is not a point rather a set of compromise vectors of the objectives to be optimized. Usually, these methods obtain a set of points whose images have a uniform spread along the whole solution set of the given problem. However, for many applications it is important for a *Decision Maker* (DM) to find points which satisfy certain values for each function. An alternative to solve such problems is to find the closets solutions to a reference points defined by the DM in the objective space, rather than finding the entire set of solutions. This approach is useful when working with *Many Objective Optimization Problem* (MaOP), i.e., problems with more than three objectives.

In this work, we focus on the scenario in which the decision maker wants to follow certain direction, as much in the space of objective functions as in the space of decision variables, starting from an initial optimal solution. To solve this problem, we propose a continuation method for the treatment of MaOPs. A direct application to this method is when the decision maker, after obtaining a solution, constantly changes his/her preferences, leading to a *Dynamic Reference Point Problem* (DRPP), where the reference point is dependant on time. This is only a module of a tool for the treatment of MaOPs called *Pareto Explorer* (PE).

1.1 Motivation

Several strategies for the treatment of continuous MaOPs have been proposed and tested over the past decades. However, we can identify two principal classes of approaches as the most common techniques to solve MaOPs. The former one is the use of evolutionary algorithms in order to compute the entire set of optimal solutions. This approach has the issue that for a MaOP a good approximation to the entire solution set implies to get a lot of points. For example, if we have k objective functions and we want M points for each dimension, then the solution set will have M^{k-1} points, i.e., an exponential growth in k. The above is not very suitable for a DM since he/she must evaluate all the solutions. The latter one is the use of local mathematical algorithms that produces only one point, which is not enough for a DM, e.g., the reference point method.

The PE method raises as a solution for the continuous MaOP and much more. PE is conceived as a global/local exploration tool for the treatment of MaOPs, which consists of two principal phases: *i)* obtaining a global optimal solution for a given MaOP and *ii)* the local exploration of optimal solutions.

1.2 Problem

In certain cases, the DM is interested in a certain region of the solution set of a MaOP. A common way to define this region is with the value of each function, int his way we can obtain an initial optimal solution based on these preferences.

In this work we focus on the scenario where the DM has an initial optimal solution and then he/she wants to search for more optimal solution in certain direction. This direction can be defined both in objective space (the space of the objective functions) and the decision space (the space of the decision variables). Additionally, we treat the DRPP, i.e., we are interested in the scenario where we have a reference point which changes over the time, leads a time depend curve. For this case the continuation method is guided by the time depend curve.

1.3 Aims of the Thesis

Our goal is to design an efficient continuation method for the steering phase of the PE able to find a resulting path which represent the best movement according with the given direction and solving the DRPP, in order to contribute the state-of-the-art.

Our particular aims are the following.

- To contribute with the framework of the PE method.
- To define several ways to steer the local search.
- To solve the DRPP.
- To solve a real world problem with the PE.

1.4 Final Contribution

This thesis contributes to the area of *many-objective optimization*, specifically it provides an efficient way to steer a local search in a given direction. This contribution is a module of a numerical tool called PE.

Our particular contributions are:

- Proposal of PE framework.
- A method for the steering in objective space.
- A method for the steering in decision space.
- A method for the steering in weight space.
- A method for a steering in decision space with a minimal change in objective space.
- An implementation to solve the DRPP.
- A software package for free scientific use.
- A preliminary version of a GUI.
- Collaboration with the University of Paderborn, Germany.

1.5 Organization of the Thesis

This thesis consists of five chapters, including this introductory chapter. The remainder of this document is organized as follows.

Chapter 2 presents the basic concepts of scalar, multi and many objective optimization problems. Further, we review some of the methods for solving a continuous MOP optimization problem along with some methods for MaOPs and *Scalar Optimization Problem*s (SOPs). The PE method is described in detail in Chapter 3, beginning with a general view of the state framework of this method. A whole description of the local exploration using the steering approach is further on given. Results on several benchmark functions and one real work problem are shown in Chapter 4. Finally, Chapter 5 presents our conclusions of this thesis along with some ideas to be considered for future work.

Chapter 2

Basic Concepts

We introduce principal concepts and the necessary theoretical background to understand this work throughout this chapter. The scope of this work contemplates the treatment of a particular class of optimization problems, MaOPs. Thus, we start to define a SOP in Section 2.1 together with three alternatives to solve it. In Section 2.2 we address MOPs and state the definitions and optimality conditions used along this work. We also explain the difference between SOPs and MOPs, as well as the conflict that exists to find a suitable solution for MOPs. Due to the importance of some methods and approaches developed to solve MOPs numerically, we describe the most widely used, related with this work, in Section 2.3. Finally, in Section 2.4 we deal with MaOPs and the most common approaches used to solve them.

2.1 Single Objective Optimization

In a SOP we have a unique objective function which depends on one or more variables, $f : \mathbb{R}^n \to \mathbb{R}$ and it is subject to certain constrains. Then, we can write a continuous SOP in a standard form as:

$$\begin{array}{ll} \min\limits_{x \in \mathbb{R}^n} & f(x), \\ \text{s.t} & g_j(x) = 0, \quad i = 1, ..., m, \\ & h_i(x) = 0, \quad j = 1, ..., p, \end{array} \tag{2.1}$$

where $g_j : \mathbb{R}^n \to \mathbb{R},\ i = 1, \ldots, m$, are the inequality constraints and $h_i : \mathbb{R}^n \to \mathbb{R},\ j = 1, \ldots, p$, are the equality constraints.

Definition 2.1. *The set of points* $\mathcal{X} = \{x \in \mathbb{R}^n \,|\, g(x) \leq 0 \text{ and } h(x) = 0\}$ *is called feasible region 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 inequalities and equalities, respectively.*

We assume along this work that the objective function and the constraints are continuously differentiable. Under this assumption, we define the gradient $\nabla f(x) \in \mathbb{R}^n$ of a multivariable function f as the vector consisting of the function partial derivatives

$$\nabla f(x) = \begin{pmatrix} \frac{\delta f}{\delta x_1}(x) \\ \vdots \\ \frac{\delta f}{\delta x_n}(x) \end{pmatrix} \tag{2.2}$$

and the Hessian matrix $\nabla^2 f(x) \in \mathbb{R}^{n \times n}$ as the square matrix of the second order derivatives

$$\nabla^2 f(x) = \begin{pmatrix} \frac{\delta f}{\delta^2 x_1}(x) & \frac{\delta f}{\delta x_1 \delta x_1}(x) \ldots \frac{\delta f}{\delta x_1 \delta x_n}(x) \\ \frac{\delta f}{\delta x_2 \delta x_1}(x) & \frac{\delta f}{\delta^2 x_2}(x) \quad \ldots \frac{\delta f}{\delta x_2 \delta x_n}(x) \\ \vdots & \\ \frac{\delta f}{\delta x_n \delta x_1}(x) & \frac{\delta f}{\delta x_n \delta x_2}(x) \ldots \quad \frac{\delta f}{\delta^2 x_n}(x) \end{pmatrix}. \tag{2.3}$$

To solve (2.1), the task is to find a vector $x^* \in \mathcal{X}$, such that the function evaluation at x^* gets a minimum value $f(x^*) \in \mathcal{X}$. That is, there is no other $x \in \mathcal{X}$, such that $f(x) < f(x^*)$. Mathematically we have the following.

Definition 2.2. *A point* x^* *is a global minimizer of* f *if* $f(x^*) \leq f(x)$ *for all* $x \in \mathcal{X}$ *and it is a local minimizer of* f *if* $f(x^*) \leq f(x)$ *is satisfied within a feasible neighborhood of* x^*.

Notice that, for some problems, the value $f(x^*)$ could be obtained for more than one vector. This means that, if we have a feasible solution and depending on characteristics thereof, then it is possible to find a set of solution vectors, such that the function takes the minimum value for each one. However, the optimal value of the objective function is unique.

There exist a lot of methods to solve a SOP, each one of which tries to exploit the characteristics of a given SOP. Due to the scope of this work we will not go into detail about these methods. However, we briefly describe three of the most common methods for the continuous case whose concepts are useful for next sections.

2.1.1 Line Search Strategies

The idea of this kind of methods is to find a *descent direction* and then to compute a step size that produces a sufficient decrement along the given direction. In general a iteration of this method is given by

$$x_{i+1} = x_i + t\nu, \tag{2.4}$$

where $t > 0$ is the step size and $\nu \in \mathbb{R}^n$ is the descent direction.

Definition 2.3. *At a point $x \in \mathbb{R}^n$, a descent direction for a function $f : \mathbb{R}^n \to \mathbb{R}$ is defined as a vector $\nu \in \mathbb{R}^n$ such that*

$$\nabla f(x)^T \nu < 0. \tag{2.5}$$

We can solve the following problem to find the best possible step size

$$f_\nu(t) = f(x + t\nu). \tag{2.6}$$

However, the optimization process of (2.6) may be costly. Thus, we need a procedure to find an *acceptable* step size.

Armijo condition allows to get a *sufficient decrease* in the objective function f. This condition is given by

$$f(x + t\nu) \leq f(x) + c_1 t \nabla f(x)^T \nu, \tag{2.7}$$

where $c_1 \in (0, 1)$. On the other hand, a rule that prevents *too small* steps lengths is given by

$$\nabla f(x + t\nu)^T \nu \leq c_2 \nabla f(x)^T \nu, \tag{2.8}$$

where $c_2 \in (0, 1)$. Equations (2.7) and (2.8) together are called the Wolfe conditions.

2.1.2 Newton Method

A Newton method iteration [2] is computed by

$$x_{i+1} = x_i - \nabla^2 f(x)^{-1} \nabla f(x). \tag{2.9}$$

Notice that the structure of (2.9) is very similar to the line search step (2.4). Indeed, we can consider that the direction ν is given by the negative of the gradient, which is a descent direction, and that the step size control is provided by the inverse of the Hessian at the current iteration x_i.

The Newton method possesses some important properties, for instance, it presents typically local quadratic convergence. However, we need the Hessian at each iteration making the method computationally expensive. As an alternative to the above we can use inexact Newton methods, e.g. the Newton-CG [2].

2.1.3 BFGS Method

We can use numerical methods to approximate the Hessians, e.g. finite differences or automatic differentiation [3]. Yet, there exist more suitable methods to update the Hessians when we do not have this information at hand: the *Quasi-Newton* (QN) methods [4]. QN methods builds a quadratic model of the problem to approximate the Hessians at each iterations. These approximations produce at most superlinear convergence.

Some of the QN methods use a line search strategy, where the search direction is given by

$$\nu = -B^{-1}\nabla f(x), \tag{2.10}$$

where $B \approx \nabla^2 f(x)$. Notice that the the only difference with the Newton method is the use of B instead the Hessian.

The most common method update is the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS), and it is given by

$$B_{i+1} = B_i + \frac{B_i s s^T B_i}{s^T B_i s}, \tag{2.11}$$

where $s = x_{i+1} - x_i$ and and $y = \nabla f(x_{i+1}) - \nabla f(x_i)$.

A necessary and sufficient condition to guarantee the positive definiteness of B is the curvature condition which is satisfied by imposing the Wolfe condition (2.8) on the step size control.

2.2 Multiobjective Optimization

The continuous MOP is defined as

$$\begin{array}{ll} \min\limits_{x\in\mathbb{R}^n} & F(x), \\ \text{s.t} & g_j(x) = 0, \quad i = 1, ..., m, \\ & h_i(x) = 0, \quad j = 1, ..., p. \end{array} \tag{2.12}$$

where $F : \mathbb{R}^n \to \mathbb{R}^k$, $F(x) = (f_1(x), \ldots, f_k(x))^T$, $f_i(x) : \mathbb{R}^n \to \mathbb{R}$ $i = 1, \ldots, k$, with a feasible region as in Definition 2.1.

A special type of inequality constraints are so called *box constraints*, which define limits for each component of the vector $x \in \mathbb{R}^n$, so box constraints have the form $a_i \leq x_i \leq b_i$, with $a, b \in \mathbb{R}^n$.

The main difference between a SOP and a MOP is the nature of the solution. As we explained before, the solution for a SOP, if it exists, is a unique optimum value that can be achieved by more than one point in the function domain. On the other hand, the solution of a MOP implies finding a trade-off among all the objective functions, and consequently, it leads to find a set of vectors instead of a single value.

The above occurs when functions are in conflict with each other (see e.g. Figure 2.1); thus, while we minimize one of the objective functions, some of the others increase their value and vice versa. For example, if we minimize simultaneously two monotone decreasing functions the solution $x^* \in \mathcal{X}$ (we assume that the minimum exists within the feasible region $\mathcal{X}$) is a unique point instead a set, due to there are not functions in conflict.

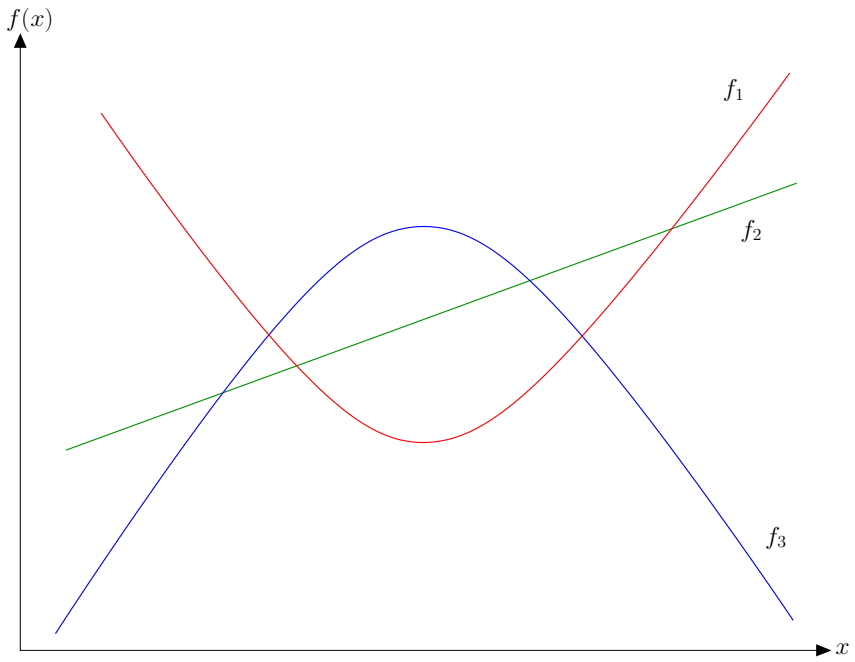


Figure 2.1: Functions in conflict, values of functions f_1, f_2 and f_3 have different behavior with respect to x.

One important characteristic of MOPs is that its solution set, the *Pareto Set* (PS), typically forms a $(k-1)$-dimensional object where k is the number of objectives involved in the MOP. Thus, it is only possible to approximate the entire set of compromise solutions for relatively few number of objectives (say, $k = 3$ or 4). However, the vast majority of the time it is not easy to get such approximation. Therefore, it is important to provide numerical methods to obtain subsets of representative solutions.

Further, in a real world problem, the objective functions commonly have distinct units making them incomparable with each other. For example, we can not compare a cost function versus a quality function. Even if all functions have the same measurements, there is not a *total order* for vectors but only a *partial order*.

A *total order* in $\mathbb{R}$ refers to that, for all $a, b \in \mathbb{R}$, it is always possible to know if $a \leq b$. We can define a *partial order* for vectors as follows, given $c, d \in \mathbb{R}^k$ we say that $c \leq d \iff c_i \leq d_i \ \forall i \in \{1, \ldots, m\}$. Therefore, we need a different way to compare two vectors based on the values of the objectives.

For multiobjective optimization, the most commonly adopted method to compare solutions is the one called *Pareto dominance relation*. This notion of optimality takes into account the aspects that we have considered intuitively in this section. It was originally proposed by Francis Ysidro Edgeworth in 1881 [5] and was later generalized by Vilfredo Pareto in 1896 [6].

In order to formalize the above, we will introduce some definitions.

2.2.1 Definitions

We have two principal spaces when considering MOPs. The first one is called *decision space*. This is the space formed by the variables of the problem; according to our notation this space is within the $\mathbb{R}^n$. The second one is called *objective space* and it is formed by the objective functions. The image of a decision vector is in $\mathbb{R}^k$. When we solve a MOP, we must do comparisons between images of decision vectors, namely, we must do its comparison in objective space.

Definition 2.4 (Pareto dominance)**.** *A point $y \in \mathcal{X}$ is dominated by a point $x \in \mathcal{X}$ if $f_i(x) \leq f_i(y) \ \forall \ i \in \{1, \ldots, k\}$ and $f_j(x) < f_j(y)$ for some $j \in \{1, \ldots, k\}$. In this case we use the notation $x \prec y$, otherwise we say that y is non dominated by x.*

Definition 2.5. *Let $x^* \in \mathcal{X}$ be a feasible point of* (2.12)*, x^* is called* ***weakly Pareto optimal*** *if $\nexists \ x \in \mathcal{X}$ s.t. $f_i(x) < f_i(x^*) \ \forall \ i = 1, \ldots, k$.*

Definition 2.6. *A decision vector $x^* \in \mathbb{X}$ is* ***Pareto optimal*** *with respect to* (2.12) *if there does not exist another decision vector $x \in \mathcal{X}$ such that $x \prec x^*$.*

Definition 2.7. *A point $x^* \in \mathcal{X}$ is locally (weak) optimal if it is (weak) optimal in a feasible neighborhood of x.*

In Figure 2.2 we show a representation of a set of *compromise solutions* for two functions, $f_i(x)$ and $f_2(x)$, which depend of the variable $x \in \mathbb{R}$; each function f_i takes its minimum value at x_i^*. As we can see, the line segment connecting x_1^* and x_2^* consist of the optimal values.

We can see in Figure 2.3 examples of Pareto dominance, weakly Pareto optimality, and Pareto optimality. This plot represents a surface in objective space for two objective functions, $f_1(x)$ and $f_2(x)$. The points p_2 and p_3 are Pareto optimal. As we can see, there is not another point with a less value in both components; the point

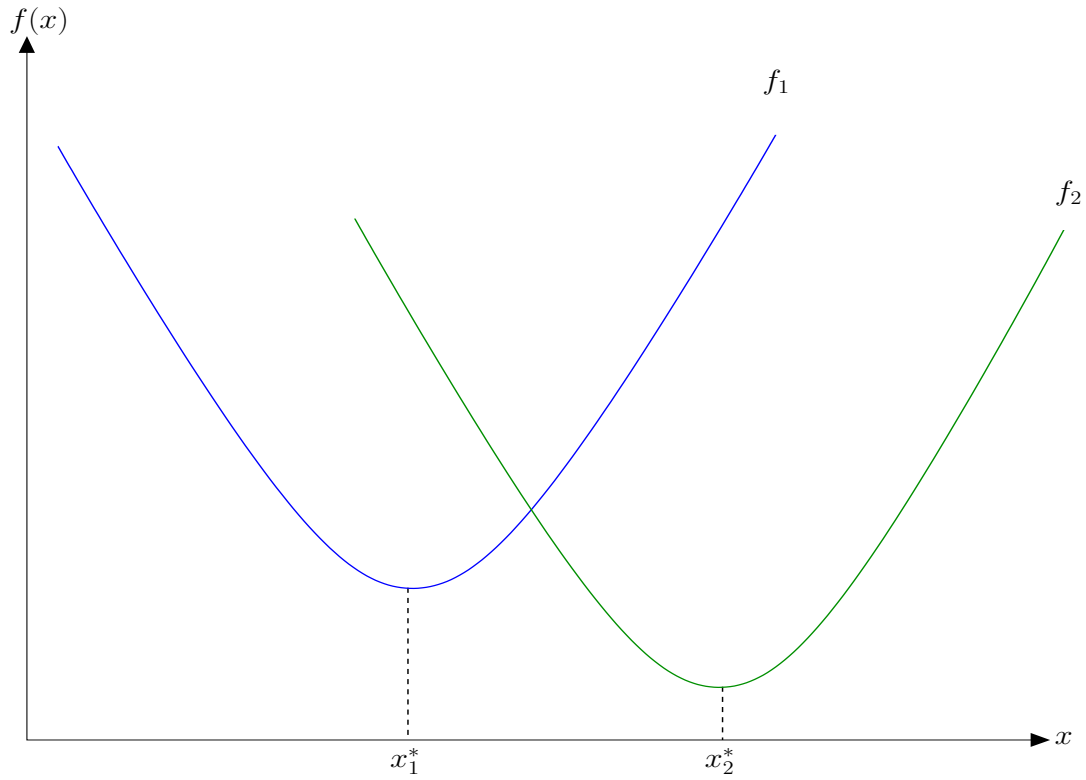


Figure 2.2: We illustrate the idea of compromise solutions. In this graph, we have optimal values between x_1^* and x_2^*, as we can see, values of functions from x_1^* to x_2^* present a trade-off for both.

p_1 is weakly Pareto optimal because the point p_2 dominates it, even so, p_1 is not dominated by the point p_3; for its part, the point p_4 is dominated by the other three points.

Definition 2.8. *The set of optimal points* $\mathcal{P}$ *for* (2.12)*,*

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

is called PS.

Definition 2.9. *The set of images* $\mathcal{F}$ *for* (2.12)*,*

$$\mathcal{F} = \{F(x) \in \mathbb{R}^k \mid x \in \mathcal{P}\}$$

is called Pareto Front (PF).

Examples of PS and PF are shown in Figure 2.2 and Figure 2.3, respectively. In Figure 2.2 the PS is the line segment connecting x_1^* and x_2^*. Whereas that, in Figure 2.3, the PF is represented by the line from $F(p_2)$ to $F(p_3)$.

A large variety of methods has been developed to solve MOPs. These methods try to get a set of optimal solutions and they focus on two principal aspects. On one

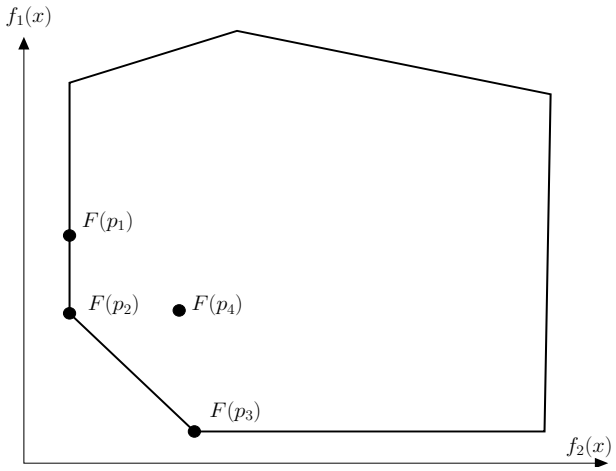


Figure 2.3: We show in this figure a $2D$ plot in objective space with four points, p_1 is a weakly Pareto optimal point, p_2 and p_3 are Pareto optimal points, and finally, p_4 is a dominated point in Pareto sense.

hand, it is important to generate points in all the PF, that is, a good extension; on the other hand, an uniform distribution along the PF is desirable. Nevertheless, as we will show in Section 2.3, according to the problem, it is sometimes necessary to adopt a different approach to solve a MOP.

2.2.2 Optimality Conditions

A first order condition of optimality for differentiable MOPs is given by the *Karush-Kuhn-Tucker* (KKT) equations, named after the work of Karush [7] and Kuhn and Tucker [8].

Theorem 2.1. *Suppose that x^* is a local solution of (2.12). Then, there exist Lagrange multipliers $\alpha \in \mathbb{R}^k$, $\lambda \in \mathbb{R}^p$ and $\gamma \in \mathbb{R}^m$ such that the following conditions are satisfied*

$$\sum_{i=1}^{k} \alpha_i \nabla f_i(x^*) + \sum_{i=1}^{p} \lambda_i \nabla h_i(x^*) + \sum_{i=1}^{m} \gamma_i g_i(x^*) = 0 \tag{2.13a}$$

$$h_i(x^*) = 0, \quad i = 1 \dots p, \tag{2.13b}$$

$$g_i(x^*) \leq 0, \quad i = 1 \dots m, \tag{2.13c}$$

$$\alpha_i \geq 0, \quad i = 1 \dots k, \tag{2.13d}$$

$$\sum_{i=1}^{k} \alpha_i = 1, \tag{2.13e}$$

$$\gamma_i \geq 0, \quad i = 1 \dots m, \tag{2.13f}$$

$$\gamma_i g_i(x^*) = 0, \quad i = 1 \dots m. \tag{2.13g}$$

One important aspect to outline is that given a KKT point x^* its associated weight vector α is normal to the linearization (tangent) of the Pareto front at $F(x^*)$ [9]. The above together with the following definition are a fundamental concepts for this work.

Definition 2.10. *A point $\hat{x}$ is is a critical point of F if rank$(J(\hat{x})) < k$, where J is the Jacobian of F at an arbitrary point x defined as*

$$J(x) = \begin{pmatrix} \nabla f_1(x)^T \\ \vdots \\ \nabla f_n(x) \end{pmatrix}. \tag{2.14}$$

Then, non critical points are those where the Jacobian has full rank. Note that, this criteria applies for both minimization and maximization problems.

Finally, an important aspect in the context of our work is that Pareto points typically form a $(k-1)$-dimensional differentiable manifold [9]. This and subsequent applications will be subject to an in-depth discussion throughout the thesis.

2.3 Solving a MOP

In this section we describe the most important methods related to the continuation method proposed in this work. We focus on continuation methods, interactive methods, and reference point methods for MOPs.

2.3.1 Continuation Methods

Continuation methods have been used to solve MOPs. These methods have the advantage that they perform a movement along a set of interest. To achieve this, we need an initial optimal solution. Starting from this point we compute a predictor, which is a movement according to certain criteria, and then we correct this point to a new optimal solution. The consideration of both the predictor and in the corrector, gives rise to different methods, e.g, Hillermeier method [9], the Directed Search Predictor-Corrector method [10], and the Pareto Tracer refer a continuous case; and the direct Zigzag method [11] for the discrete case, no consider in this work.

Method by Hillermeier

Many predictor-corrector methods are based on the *Implicit Function Theorem* [12]. We briefly state the main steps of classical predictor-corrector methods for

tracing one-dimensional solution. According to this theorem if x is a solution of

$$H(x) = 0, \tag{2.15}$$

where $H : \mathbb{R}^{N+1} \to \mathbb{R}^N$ and $rank(H'(x)) = N$, then there exists a value $\epsilon > 0$ and a curve $c : (-\epsilon, \epsilon) \to \mathbb{R}^{N+1}$ such that $c(0) = x$ and

$$H(c(s)) = 0 \quad \forall\, s \in (-\epsilon, \epsilon). \tag{2.16}$$

Differentiating we obtain

$$H'(c(s))c'(s) = 0. \tag{2.17}$$

This means that we can get the tangent vectors $c'(s)$ computing the kernel vectors of $H'(x)$, This can be done via a QR factorization of the matrix $H'(x)^T$, i.e.

$$H'(x)^T = QR \tag{2.18}$$

for an orthogonal matrix $Q \in \mathbb{R}^{(N+1)\times(N+1)} = (q_1, \ldots . q_{N+1})$ and a right upper triangular matrix $R \in \mathbb{R}^{(N+1)\times N}$. Doing so, the last column vector q_{N+1} is a kernel vector.

Finally, the orientation of the curve can be control led be monitoring the sign of

$$(det) \begin{pmatrix} H'(x) \\ q_{N+1}^T \end{pmatrix}. \tag{2.19}$$

Thus, we can compute a *predictor* point p along the linearized solution curve following the same orientation. Now we can get back to a curve $c(x)$ using (2.15) and p as starting point e.g. via a Gauss-Newton or a Levenberg-Marquardt method [13].

A method was developed by Hillermeier in 2001 [9], for the multiobjective optimization context by considering the auxiliary function $\hat{F} : \mathbb{R}^{n+k} \to \mathbb{R}^{n+1}$,

$$\hat{F}(x, \alpha, \lambda) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla f_i(x) + \sum_{i=1}^{p} \lambda_i \nabla h_i(x) \\ h(x) \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix} = 0. \tag{2.20}$$

The set of KKT points of a non linear equality constrained is contained in the zero set of $\hat{F}$, which motivates the continuation along $\hat{F}^{-1}(0)$. We show a geometrical idea of this method in Figure 2.4.

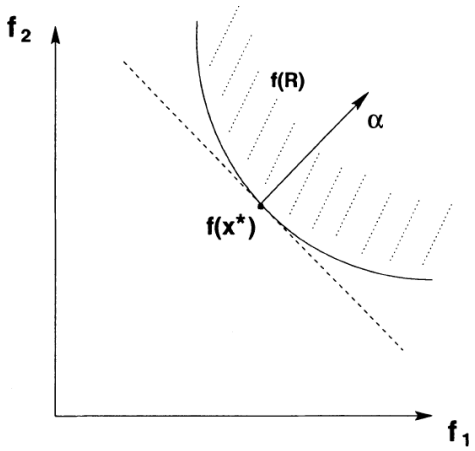


Figure 2.4: We show in this figure the α vector, which is orthogonal to the linearization PF at the point $F(x^*)$ (dotted line).

The Hillermeier method proceeds as the general technique described above, but instead of computing the determinant given in (2.19), authors check the following condition for two consecutive solutions

$$[x_i - x_{i+1}]q \geq 0, \tag{2.21}$$

where $x_i, x_{i+1} \in \mathbb{R}^{n+k+p}$ and q is the tangent vector.

Also, the author suggest a suitable step size which guarantees a uniform spread of the solutions on the PF. That is, for two consecutive solutions we want

$$\|F(x_i) - F(x_{i+1})\| \approx \tau, \tag{2.22}$$

where $\tau > 0$ is the desirable spread. The suggested step size is given by

$$t = \frac{\tau}{\|J\nu_d\|}. \tag{2.23}$$

Pareto Tracer Method

The idea of the Pareto Tracer method [14] is to separate the decision and weight space which is not done in the Hillermeier method. In the compose (x, α, λ) space (see Equation 2.20), the non linearity may increase, e.g, if the PS is linear, then the related solution set does not have to be linear in the compose space. Thus, when we separate x and α spaces the non linearity comes to decrease and it implies that a corrector step is not needed for linear PSs. The above also implies a reduction of the total computational cost. However, the most important aspect for this work about this separation is that this make possible to find a relationship between a direction in objective and decision space, which is the fundamental part for this work.

We first consider the following unconstrained MOP

$$\min_{x \in \mathbb{R}^n} F(x), \tag{2.24}$$

and describe the sequel predictor and corrector steps.

Predictor. The typical task for the computation of predictor points in continuation methods is to determine the tangent space to the given set. In order to do this, we take by KKT conditions

$$\hat{F}(x, \alpha) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla f_i(x) \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix} = 0. \tag{2.25}$$

Differentiating we obtain

$$\hat{F}'(x, \alpha) \begin{pmatrix} \nu \\ \mu \end{pmatrix} = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla^2 f_i(x) & \nabla f_1(x) & \dots & \nabla f_k(x) \\ 0 & 1 & \dots & 1 \end{pmatrix} \begin{pmatrix} \nu \\ \mu \end{pmatrix} = 0 \tag{2.26}$$

The second equation of (2.26) yields,

$$\sum_{i=1}^{k} \mu_i = 0. \tag{2.27}$$

Having a $\mu \in \mathbb{R}^k$ that satisfies (2.27) we obtain

$$\sum_{i=1}^{k} \alpha_i \nabla^2 f_i(x) \nu = -\sum_{i=1}^{k} \mu_i \nabla f_i(x) = -J^T \mu, \tag{2.28}$$

and it is possible to find a relationship between ν and μ, i.e., a relationship between the objective space and the variable space:

$$\nu_\mu = -W_\alpha^{-1} J^T \mu, \tag{2.29}$$

where $W_\alpha \in \mathbb{R}^{n \times n}$ is given by

$$W_\alpha := \sum_{i=1}^{k} \alpha_i \nabla^2 f_i(x). \tag{2.30}$$

If $rank(J) = k - 1$, we can compute the set of tangent vectors via a QR factorization of α

$$\alpha = QR = (q_1, q_2, \dots, q_k)R, \tag{2.31}$$

where $q_i \in \mathbb{R}^k$ and $R \in \mathbb{R}^{k\times 1}$. Let Q_2 denote the matrix formed by the last $k-1$ columns vectors of Q

$$Q_2 = (q_2, \dots q_k). \tag{2.32}$$

this matrix is an orthonormal basis of the linearized Pareto front at $F(x)$ (see Figure 2.5).

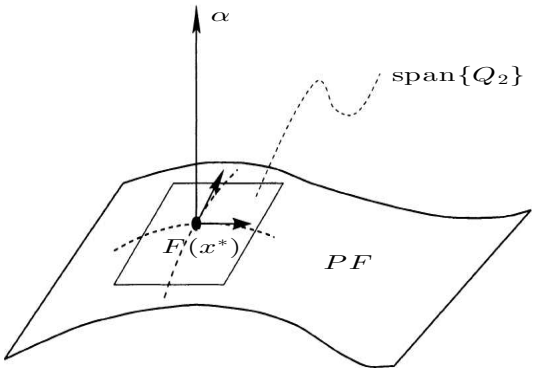


Figure 2.5: We show a plane formed by the tangent vectors of the PF at the point $F(x^*)$ via QR factorization.

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

$$d = J\nu. \tag{2.33}$$

The orientation of the movements along the tangent space is related to (2.33). Thus, the task is to find the proper orientation vector $_d \in \mathbb{R}^k$ that satisfies

$$J\nu_{\mu_d} = d. \tag{2.34}$$

The vector ν_μ can be obtained with the vector μ_d that solves:

$$\begin{pmatrix} -JW_\alpha^{-1}J^T \\ 1\dots 1 \end{pmatrix} \mu_d = \begin{pmatrix} d \\ 0 \end{pmatrix}. \tag{2.35}$$

So, the predictor is given by the following expression,

$$p = x^* + t\nu_\mu. \tag{2.36}$$

where t is the step size given by Equation (2.23) as the Hillermeier method.

Corrector. The goal of the corrector phase is to ensure that the resulting solution is on the efficient set.

We apply the Newton method for MOPs [15] as corrector. The Newton direction is defined as the solution to:

$$\begin{array}{ll} \min\limits_{(\nu,\delta)\in\mathbb{R}^n\times\mathbb{R}} & \delta \\ \text{s.a} & \nabla f_i(x)^T\nu + \frac{1}{2}\nu^T\nabla^2 f_i(x)\nu \leq \delta, \quad i = 1, ..., k, \end{array} \tag{2.37}$$

where δ serves as a measure of the expected decrease in objective space produced by a line search in direction ν in parameter space. An acceptable step size may be decided by a backtracking procedure with a modification of the (componentwise) Armijo condition.

Handling Equality Constrains Now we consider the following MOP:

$$\begin{array}{ll} \min\limits_{x\in\mathbb{R}^n} & F(x), \\ \text{s.t} & h_i(x) = 0, \quad i = 1, ..., p. \end{array} \tag{2.38}$$

In the presence of equality constraints, the follow KKT system has to be considered; let $\tilde{F} : \mathbb{R}^{n+k+p} \to \mathbb{R}^{n+p+1}$,

$$\tilde{F}(x, \alpha, \lambda) = \begin{pmatrix} \sum_{i=1}^k \alpha_i \nabla f_i(x) + \sum_{j=1}^p \lambda_j \nabla h_j(x) \\ h(x) \\ \sum_{i=1}^k \alpha_i - 1 \end{pmatrix} = 0. \tag{2.39}$$

Now we can proceed as in the unconstrained case. We define:

$$W_{\alpha,\lambda} := \sum_{i=1}^k \alpha_i \nabla^2 f_i(x) + \sum_{j=1}^p \lambda_j \nabla^2 h_j(x) \in \mathbb{R}^{n\times n}, \tag{2.40}$$

and

$$H := \begin{pmatrix} \nabla h_1(x)^T \\ \vdots \\ \nabla h_p(x)^T \end{pmatrix} \in \mathbb{R}^{p\times n}. \tag{2.41}$$

Thus, we can write $\tilde{F}'$ as:

$$\tilde{F}'(x, \alpha, \lambda) = \begin{pmatrix} W_{\alpha,\lambda} & J^T & H^T \\ H & 0 & 0 \\ 0 & 1, \ldots, 1 & 0 \end{pmatrix}. \tag{2.42}$$

Predictor. In order to compute a kernel vector of (2.42), we consider $\nu \in \mathbb{R}^n$, $\mu \in \mathbb{R}^k$ and $\xi \in \mathbb{R}^p$, such that:

$$\begin{pmatrix} W_{\alpha,\lambda} & J^T & H^T \\ H & 0 & 0 \\ 0 & 1,\ldots,1 & 0 \end{pmatrix} \begin{pmatrix} \nu \\ \mu \\ \xi \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{2.43}$$

The choice of a vector μ which satisfies (2.35) allows to reduce (2.43) to:

$$\begin{pmatrix} W_{\alpha,\lambda} & H^T \\ H & 0 \end{pmatrix} \begin{pmatrix} \nu_\mu \\ \xi \end{pmatrix} = \begin{pmatrix} -J^T\mu \\ 0 \end{pmatrix}. \tag{2.44}$$

If $\text{rank}(W_{\alpha,\lambda}) = n$ and $\text{rank}(H) = p$, then the matrix on the left hand side is regular and so the solution of (2.44) is unique.

Corrector. For the corrector step, we need a modification of the Newton method for the given MOP. A suggestion is to modify the Newton direction via:

$$\begin{array}{ll} \min\limits_{(\nu,\delta)\in\mathbb{R}^n\times\mathbb{R}} & \delta \\ \text{s.a} & \nabla f_i(x)^T\nu + \frac{1}{2}\nu^T\nabla^2 f_i(x)\nu \leq \delta, \quad i = 1,...,k. \\ & h_i(x) + \nabla h_i(x)^T\nu = 0, \qquad\quad i = 1,...,p. \end{array} \tag{2.45}$$

The additional restriction arises from applying the Newton method to h. The following result shows how we can view this modification as a particular penalization method. The new penalized MOP is given by:

$$\min_{x\in\mathbb{R}^n} F_h : \mathbb{R}^n \to \mathbb{R}^k, \tag{2.46}$$

where $F_h = \left(f_1^h, \ldots, f_k^h\right)^T$, and

$$f_i^h(x) = f_i(x) + CP(x), \tag{2.47a}$$

$$P(x) = \frac{1}{2}\sum_{i=1}^{p} h_i(x)^2 = \frac{1}{2}\|h(x)\|^2. \tag{2.47b}$$

Proposition 2.1. *Let $x \in \mathbb{R}^n$ be given and the $f_i's$ be strictly convex, and (ν^*, δ^*) be a solution of* (2.45)*. Then*

(a) If $\nu^* = 0$, then $\delta^* = 0$ and x is a KKT point of (4.3)*.*

(b) If $\nu^* \neq 0$ and $\delta^* < 0$, then ν^* is a descent direction of (2.46) *for $C = 0$ (i.e. a descent direction of the unconstrained MOP* (2.12)*).*

(c) If $\nu^* \neq 0$ *and* $\delta^* \geq 0$*, then* $\|h(x)\|^2 \neq 0$ *and* ν^* *is a descent direction of* (2.46) *for*

$$C > \frac{\max_{i=1,\ldots,k} \nabla f_i(x)^T \nu^*}{\|h(c)\|^2} \geq 0. \tag{2.48}$$

As we compute descent directions of (2.46), there are 3 possibilities: (i) we can improve F but not P, (ii) we can improve both, and (iii) we can improve P but not F. This is reflected in the step size control. A component-wise Armijo condition is used together with the following function:

$$\tilde{F}_h(x) = \begin{cases} F(x) & \delta < 0 \text{ y } \|h(x)\|^2 = 0 \\ (F(x), P(x))^T & \delta < 0 \text{ y } \|h(x)\|^2 \neq 0 \\ P(x) & \delta \geq 0 \end{cases} \tag{2.49}$$

Notice that if $\delta < 0$, then (i) or (ii) is satisfied. We know, by Proposition 2.1, that ν^* is a descent direction of F. If $\|h(x)\|^2 = 0$, then we can not further improve P, so this is not considered. Now, if $\|h(x)\|^2 \neq 0$, then F and P can be simultaneously decreased through a linear search in direction ν^*. If $\delta \geq 0$, then at least one of the objectives increases its value with the direction ν^*.

By Proposition 2.1, we have $\|h(x)\|^2 \neq 0$ and ν^* is a descent direction of (2.46) for some $C > 0$. The choice of a step size, which produces a *decrement enough* in P, depends of $C \gg 0$. So, we take as an acceptable step size a $t \in \mathbb{R}_+$, which satisfies:

$$\tilde{F}_h(x - t\nu) \leq \tilde{F}_h(x) + ct\Delta\tilde{F}_h(x)\nu. \tag{2.50}$$

Here, δ is a measurement of the expected decrement of F in objective space and the derivative of P in the direction ν^* is given by:

$$h(x)^T H\nu^* = -\|h(x)\|^2 \leq 0.$$

Thus, we can use $-\|h(x)\|^2$ to measure the possible reduction as a penalization. The term $\Delta\tilde{F}_h$ of (2.50) represents the expected decrement of $\tilde{F}_h$ in objective space, and it is given by:

$$\Delta\tilde{F}_h(x) = \begin{cases} \delta e & \delta < 0 \text{ y } \|h(x)\|^2 = 0 \\ (\delta e, -\|h(x)\|^2)^T & \delta < 0 \text{ y } \|h(x)\|^2 \neq 0 \\ -\|h(x)\|^2 & \delta \geq 0, \end{cases} \tag{2.51}$$

where $e \in \mathbb{R}^k$ $e = [1, \ldots, 1]$.

Handling Inequality Constraints We focus on the box constraints case

$$\min_{l \leq x \leq u} \quad F(x), \tag{2.52}$$

where $l,\ u \in \mathbb{R}^n$ are the lower and upper bounds, respectively.

Predictor. We can include the set of active constraints as equality constraints to solve 2.52, i.e.

$$-x_i + l_i = 0, \quad i \in \mathcal{I}_l$$

$$x_i - u_i = 0, \quad i \in \mathcal{I}_u,$$

where

$$\mathcal{I}_l = \{i | -x_i + l_i > -\epsilon, \quad i = 1, \ldots, n\}$$

$$\mathcal{I}_u = \{i | x_i - u_i > -\epsilon, \quad i = 1, \ldots, n\}$$

for some $\epsilon \in \mathbb{R}_+$, and $\mathcal{I}_{l,u} = \{i | i \in \mathcal{I}_l \ \text{o} \ i \in \mathcal{I}_u\}$. Hence $\tilde{F} : \mathbb{R}^{n+k+r} \to \mathbb{R}^{n+r+1}$ is given by

$$\tilde{F}(x, \alpha, \rho, \varrho) = \begin{pmatrix} \sum_{i=1}^{k} \alpha_i \nabla f_i(x) - \sum_{i \in \mathcal{I}_l} \rho_i e_i + \sum_{i \in \mathcal{I}_u} \varrho_i e_i \\ (-x_i + l_i)_{i \in \mathcal{I}_l} \\ (x_i - u_i)_{i \in \mathcal{I}_u} \\ \sum_{i=1}^{k} \alpha_i - 1 \end{pmatrix} = 0, \tag{2.53}$$

where e_i is the $i-th$ canonic vector and $r = |\mathcal{I}_{l,u}|$. We define $\mathcal{I}_{l,u} \in \mathbb{R}^{r \times n}$ as

$$\left[\mathcal{I}_{l,u}\right]_{j_i} = \begin{cases} -e_i^T & i \in \mathcal{I}_l \\ e_i^T & i \in \mathcal{I}_u \end{cases}, \quad j = 1, \ldots, r, \tag{2.54}$$

where $\left[\mathcal{I}_{l,u}\right]_{j_i}$ denotes the $j-th$ row of $\mathcal{I}_{l,u}$. Notice that

$$\tilde{F}'(x, \alpha, \rho, \varrho) = \begin{pmatrix} W_\alpha & J^T & \mathcal{I}_{l,u}^T \\ \mathcal{I}_{l,u} & 0 & 0 \\ 0 & 1, \ldots, 1 & 0 \end{pmatrix}. \tag{2.55}$$

Now we must compute a kernel vector of (2.55). Let $\nu \in \mathbb{R}^n$, $\mu \in \mathbb{R}^k$ and $\eta \in \mathbb{R}^r$ be vectors, such that

$$\begin{pmatrix} W_\alpha & J^T & \mathcal{I}_{l,u}^T \\ \mathcal{I}_{l,u} & 0 & 0 \\ 0 & 1, \ldots, 1 & 0 \end{pmatrix} \begin{pmatrix} \nu \\ \mu \\ \eta \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}. \tag{2.56}$$

A μ vector which satisfies (2.35) reduces (2.56) to

$$\begin{pmatrix} W_\alpha & \mathcal{I}_{l,u}^T \\ \mathcal{I}_{l,u} & 0 \end{pmatrix} \begin{pmatrix} \nu \\ \eta \end{pmatrix} = \begin{pmatrix} -J^T\mu \\ 0 \end{pmatrix}. \tag{2.57}$$

If $\text{rank}(W_\alpha) = n$ and $\text{rank}(I_{l,u}) = r$, then we have a regular matrix and the solution of (2.57) is unique. We obtain by (2.57)

$$W_\alpha \nu + I_{l,u}^T \eta = -J^T \mu \tag{2.58a}$$

$$I_{l,u}\nu = 0, \tag{2.58b}$$

and by (2.58b), we notice that $\nu_i = 0$ for $i \in I_{l,u}$. So, it is enough to compute the jth components of ν, such that $j \notin I_{l,u}$. In addition, we know that $(I_{l,u}^T \nu)_j = 0$ for $j \notin I_{l,u}$, then

$$W_\alpha^{I_c} \nu^{I_c} = -J_{I_c}^T \mu, \quad I_c := \{1, \ldots, n\} \backslash I_{l,u}, \tag{2.59}$$

where $W_\alpha^{I_c}$ comes to remove $i - th$ row and column of W_α $\forall\, i \in I_{l,u}$. Analogy, ν^{I_c} comes to remove $i - th$ element of ν and J_{I_c} comes to remove $i - th$ column of J.

Corrector. Now, the Newton direction is computed to solve

$$\begin{array}{ll} \min\limits_{(\nu,\delta)\in\mathbb{R}^n\times\mathbb{R}} & \delta \\ \text{s.t} & \nabla f_i(x)^T\nu + \frac{1}{2}\nu^T\nabla^2 f_i(x)\nu \leq \delta, \quad i = 1, ..., k. \\ & -\nu_i - x_i + l_i \leq 0, \qquad\qquad i \in I_l. \\ & \nu_i + x_i - u_i \leq 0, \qquad\qquad i \in I_u. \end{array} \tag{2.60}$$

We assume that x_i is not active respect with to both its upper and lower bound at the same time. For the step size control, we use again a component-wise Armijo condition, but in this case, we impose the following upper limit:

$$t_{max} = \min_{i=1,\ldots,n} t_i, \tag{2.61}$$

where

$$t_i = \begin{cases} \frac{l_i - x_i}{\nu_i} & \nu_i < 0 \\ \frac{u_i - x_i}{\nu_i} & \nu_i > 0, \quad i = 1, \ldots, n. \\ +\infty & \nu_i = 0. \end{cases} \tag{2.62}$$

Directed Search Predictor-Corrector Method

This method defines a way to steer the search for continuous problems. We need a direction in objective space and mapping it to parameter space [16]. The main idea of this method is as follows.

Let be $x_0 \in \mathbb{R}^n$ a point in parameter space with $rank(J(x_0)) = k$ and $d \in \mathbb{R}^k$ a given vector which representing a desired search direction in image space. Then, a search direction $\nu \in \mathbb{R}^n$ in decision space is sought such that for $y_0 := x_0 + t\nu$, where $t \in \mathbb{R}_+$ is the step size (i.e., y_0 represents a movement from x_0 in direction v), it holds:

$$\lim_{t \to 0} \frac{f_i(y_0) - f_i(x_0)}{t} = \langle \nabla f_i(x_0), \nu \rangle = d_i, \qquad i = 1, ..., k. \tag{2.63}$$

Using the Jacobian of F, Eq. (2.63) can be stated in matrix vector notation as

$$J(x_0)\nu = d. \tag{2.64}$$

Hence, such a search direction ν can be computed by solving a system of linear equations. Since typically the number of decision variables is (much) higher than the number of objectives for a given MOP, i.e., $n \gg k$, system (2.64) is (probably highly) underdetermined, which implies that its solution is not unique. One possible choice is to take

$$\nu_+ = J(x_0)^+ d, \tag{2.65}$$

where $J(x_0)^+ \in \mathbb{R}^{n \times k}$ denotes the pseudo inverse[1] of $J(x_0)$. A new iterate x_1 can be computed as the following discussion shows: given a candidate solution x_0, a new solution is obtained via $x_1 = x_0 + t\nu$, where $t > 0$ is a step size and $\nu \in \mathbb{R}^n$ is a vector that satisfies (2.64). Among the solutions of system (2.64), ν_+ is the one with the smallest Euclidean norm. Hence, given t, one expects for a step in direction ν_+ (decision space) the largest progress in d-direction (objective space).

Predictor Given a KKT point $x_0 \in \mathbb{R}^n$, it is known that its associated weight vector α is orthogonal to the linearized Pareto front at $F(x_0)$ [9] and hence any direction orthogonal to α could be a promising predictor direction. To compute such a direction a QR factorization on α can be performed:

$$\alpha = QR = (q_1, \ldots, q_k)(r_{11}, 0, \ldots, 0)^T, \tag{2.66}$$

where $Q \in \mathbb{R}^{k \times k}$ is an orthogonal matrix and $R \in R^{k \times 1}$ with $r_{11} \in \mathbb{R} \backslash \{0\}$ is an upper triangular matrix. Since by Eq. (2.66) $\alpha = r_{11} q_1$, it follows that a well spread set of directions can be taken from any of the normalized search directions ν_i such that:

$$J(x_0)\nu_i = q_{i+1}, \quad i = 1, \ldots, k-1. \tag{2.67}$$

To orientate the curve (i.e., to determine the signum of p) the change in one of the objective values can be used. For this, the signum of the according entry of the

[1] If the rank of $J := J(x_0)$ is k (i.e., maximal) the pseudo inverse is given by $J^+ = J^T(JJ^T)^{-1}$.

the direction vector q_2 can be taken. If, for instance, an improvement according to f_2 is sought, then

$$p = x_0 - sgn(q_{22})\frac{t\nu_2}{||\upsilon_2||}, \tag{2.68}$$

where t is the chosen step size.

Corrector Given a predictor p, the subsequent solution along the curve can be computed by solving

$$\begin{aligned} x(0) &= x_0 \in \mathbb{R}^n \\ \dot{\vec{x}}(t) &= J(x(t))^+ d,\ t > 0. \end{aligned} \tag{2.69}$$

Using p as initial value and choosing $d = -\alpha_0$, i.e. the negative of the weight from the previous solution x_0, leading to a new solution x_1. The new associated weight vector α_1 can be updated as follows:

$$\alpha_1 = \min_{\lambda \in \mathbb{R}^k} \left\{ \left\| \sum_{i=1}^{k} \lambda_i \nabla f_i(x) \right\|^2 \text{ s. t. } \lambda_i \geq 0, i = 1, \ldots k, \sum_{i=1}^{k} \lambda_i = 1 \right\}. \tag{2.70}$$

Figure 2.6 displays a single iteration of the DS method, p stands for the predictor direction, while c stands for the corrector direction. It can be seen from the images that even though a movement along a linearization of the Pareto front at $f(x_0)$ is desired, it is not always possible to move in such direction and hence predictors usually end up above the Pareto front, making the use of corrector steps necessary in most cases.

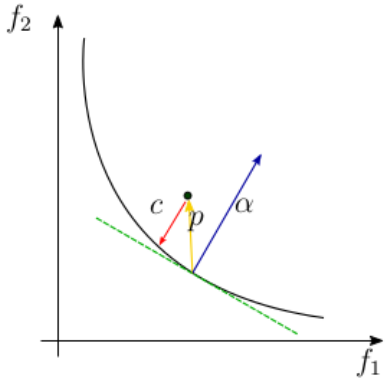


Figure 2.6: Example of the Directed Search predictor-corrector method.

2.3.2 Reference Point Methods

Usually, the solution of a MOP involves to find a set of non dominated vectors, i.e., vectors whose image can not be improved by other vectors in all its components. We commonly want to obtain a set that satisfies certain properties, e.g. that the image of this set has a uniform spread along the whole PF of the given problem.

However, there are real-world problems in which a DM has knowledge about the problem or he/she wants to obtain optimal solutions with certain characteristics instead of solutions in the whole PS. The *reference point methods* are useful for these scenarios. The idea is to get the closest solution to a given vector, usually infeasible, which is a guess of the DM. This kind of methods, where the DM has an active participation in the solution process, are called interactive methods. The difference between a type of interactive method and other ones is the kind of information asked to the DM [1].

Reference Point Problem

We can find different alternatives, which consider one reference point at the same time, in order to get a solution. An example is the classic reference point method, which was presented by Wierzbicki [17] in 1981. This method uses a given reference point, that represents the preferences of the DM, to solve a SOP. The solution of the SOP is presented to the DM and, if the solution is not good enough for the DM, then a new reference point is proposed. The process continues until the DM is in agreement with the solution.

The SOP in the reference point method employs an *achievement function*, which we will define in the next section. Other methods consider different ways to use a reference point, e.g, light beam search [18] and GUESS [19].

Achievement Functions

Most of the *achievement scalarization functions* are based on the Tchebycheff metric. For this work, we use an appropriate achievement scalarizing function defined as follows.

Definition 2.11. *The augmented weighted Tchebycheff scalarizing function is defined by:*

$$s_{z_t}(x) = \max_{i=1,\ldots,k} \{\lambda_i | f_i(x) - z_{ti}|\} + \rho \sum_{i=1}^{k} \lambda_i \left(f_i(x) - z_{ti}\right), \tag{2.71}$$

where z_t is a reference point, $\rho > 0$ is an augmentation coefficient, and $\lambda = (\lambda_1, \ldots, \lambda_k)$ is a vector of weights, such that $\forall i\ \lambda_i \geq 0$ and, for at least one i, $\lambda_i > 0$.

The exploration of the objective space, in most of the reference points methods, is made by moving the reference point at each iteration. That is, weights do not define preferences, but they are mainly used for normalizing each objective function. Usually, the weights are set as:

$$\lambda_i = \frac{1}{z_i^{\text{nad}} - z_i^{\text{utp}}},$$

where z^{nad} and z^{utp} are the *nadir* and the *utopian* point respectively.

The utopian point is a vector formed by the minimum of each objective function, $z_i^{\text{nad}} = \min f_i(x) \mid x \in \mathcal{X} \ \forall i \in \{1, \ldots, k\}$; this point is generally infeasible. The nadir point is a vector formed by the maximum of each objective function on the PF, $z_i^{\text{utp}} = \max f_i(x) \mid x \in \mathcal{P} \ \forall i \in \{1, \ldots, k\}$. Typically, the estimation of the nadir point is more complicated than the estimation of the utopian point.

The weighted Tchebycheff scalarizing function poses some convenient properties over other scalarizing functions. As proved in [1] by using the augmented version of this function we can find any Pareto optimal solution. It is important to mention that the DM can provide both feasible and infeasible reference points.

Dynamic Reference Point Problem

As we have described, reference point methods take an only point at the same time to find a solution. However, the scenario in which a reference point changes over time has been poorly treated.

Let $z(t)$ be a time-dependent curve and consider the MOP in the standard form (2.12). The DRPP is to solve,

$$\min_{x \in \mathcal{X}} s_{z_t}(x), \tag{2.72}$$

for a set of points $z(t_i)$, $i = 0, \ldots, m$. Where $s_{z_t}(x)$ is an achievement scalarizing function.

The continuation method developed in this work is capable of obtaining a sequence of points $x_i^*, \ldots, x_m$, starting from an initial solution x_0^*, such that x_i^* is solution of (2.72) for each $z(t_{i+1})$ respectively. So if we have a solution x_i^* for a reference point $z(t_i)$, then we can compute a predictor for this point with a continuation method and, after that, to obtain the next solution x_{i+1}^* solving the reference point problem for $z(t_{i+1})$, as the corrector. A geometrical idea of the DRPP is depicted in Figure 2.7.

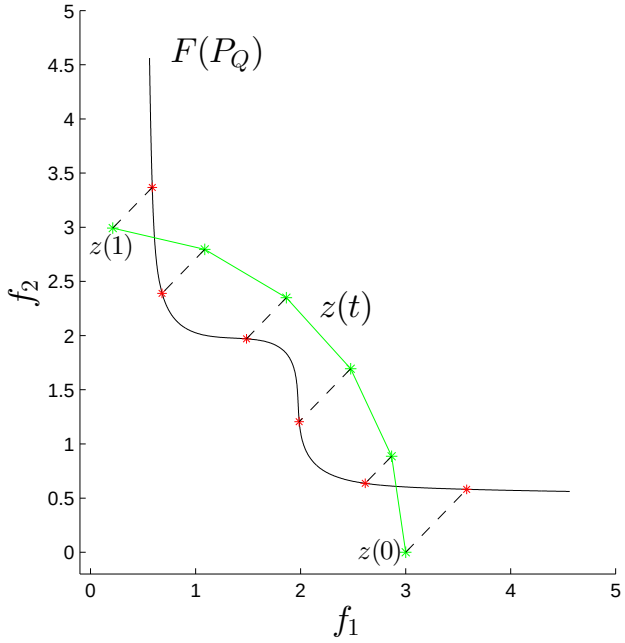


Figure 2.7: Example for the DRPP. The time curve is denoted by $z(t)$ and the PF by $F(P_Q)$. Each solution is joined by a scatter line with the point in the time curve and they are indicated with a dot. The initial point is $z(0)$, whereas, the final point is $z(1)$.

2.3.3 Interactive Methods

For its part, a different class of interactive methods, the learning-oriented methods, exploit the preferences of the DM to direct the search and reduce the number of solutions to consider. Such methods are useful when the set of optimal solutions is very large, for example, MaOPs. A wide variety of these interactive methods have been developed in recent years [20].

Pareto Navigator Method

The Pareto Navigator [21] is an interactive learning-oriented method for nonlinear multiobjective optimization, which uses a set of optimal solutions to create a polyhedral approximation of the PF. The DM can direct the search along this polyhedral approximation according to her/his preferences. Once an interesting region has been identified, the DM can continue with another method to get an optimal solution.

It is important to stand out that the navigation is not along optimal solutions and that we need a set of initial optimal solutions to use this method, which is composed

by two principal phases: the *initialization* and the *navigation*.

1. **Initialization**. Given a set of optimal solutions, we construct a polyhedral in objective space.
2. **Navigation**. According the preferences of the DM, we steer along the polyhedral until reach a good zone for the DM. Now we can use an achievement function to get the closest optimal solution to this zone.

If the DM is not satisfied, we repeat the process, adding the new solution of the initialization phase.

NIMBUS Method

NIMBUS [22] is a system for non-linear optimization problems. It has, as principal characteristic, an on-line GUI to solve MOPs. Te GUI allows the user to easily defines preferences for a search. The method used by this system to solve the MOP is an evolutionary algorithm, so it is a robust system.

Also, this system provides to the user with different kinds of plots to visualize results for problems with many objective functions.

Nautilus Method

This is an interactive method, which is based on the assumption that the DM prefers to gets better solutions in each iteration, instead to sacrifice the value of some function, then the movement stars in the nadir point.

2.4 Many-objective Optimization and Evolutionary Algorithms

The notion of many-criteria optimization was used for the first time in [23]. Mathematically, a MaOP is defined in the same way as Equation 2.12 but with $k > 3$. Thus, described methods in Section 2.3 can be used to it, but these mathematical techniques compute only one solution at each execution.

However, another approach to work with high dimensional problems is the *Evolutionary Optimization*. This approach considers Genetic Algorithms and different population algorithms. A thorough survey on multiobjective evolutionary algorithms

for MaOPs can be found in [24]. There are tables and some graphs with the most important aspects of the considered methods as the maximum number of treated objectives and tests functions used.

We can classify the methods for the treatment of MaOP in two groups. We will describe brief those groups:

i Methods using alternative preference relations:

- Crisp. An example of this kind of method is provided in [25]. Here, authors propose to use the *Preference Ordering*, a generalization of Pareto optimality which uses two more stringent definitions of optimality: *Efficiency of Order* and the *Efficiency of order k with degree z*, as a ranking criterion in the framework of NSGA-II [26]. Tests were made in this paper with problems of up to 8 objective functions.
- Fuzzy. A fuzzy relation is introduced in [23]; it is based on the number of components: bigger, smaller and equal between two vectors. We can find in this paper an expression for the portion e in a M-dimensional criteria domain, such that the dominance concept classifies as equivalent solutions, $e = \frac{2^M - 2}{2^M}$. Thus, Pareto optimal definition is not effective for MOPs.

ii Methods transforming the original MaOP into a SOP:

- Based on scalarization functions,
  - Decomposition based. The most famous method is MOEA/D [27]; this method decomposes the original problem into a set of scalar optimization problems, a scalarization function with different weights for each individual is assigned. In the original paper, the method is tested with until 4 objective functions.
- Indicator based. A method to approximate the value of the *Hypervolume* (HV) indicator was developed in [28]. The idea is to use a Monte Carlo algorithm to estimate values of the HV for a large number of objectives.
- Based on dimensional reduction techniques. Two kinds of objective reduction, *linear objective reduction* and *nonlinear objective reduction*, are presented in [29]. Both are made after to get a set of nondominated solutions with some MOEA, the idea is to consider the correlation of the solution via an eigenvalue analysis to identify the set of important objectives. Tests of this approach include problems with up to 50 objective functions.
- Based on space partitioning. ϵ Ranking-Evolutionary Multiobjective Optimizer (ϵR-EMO) [30]; this method takes the basis of the NSGA-II [26]. It uses a partition strategy to define a schedule of subspace sampling and an adaptive ϵ-ranking procedure to re-rank solutions in each subspace. The

number of solutions to be considered at each partition and the number of generations before creating a new partition is set by the user.

Additionally to the aforementioned methods and classifications, we present the following methods as the most relevant ones.

HV values

- S-metric Selection-EMOA (SMS-EMOA) [31]. The aim of this method is to maximize the HV. The former selection criterion is the non-dominated sorting procedure and the latter one is the HV, if the change of certain individual by a new one improves the HV, then this change is preserved.

- Hypervolume Estimation Algorithm for Multi-objective Optimization (Hype) [32]. As in [28], the idea is to approximate the HV indicator. Hype uses the concept of environmental selection to create a new population from the best solutions in the union set of the parent and offspring populations; this allows us to estimate the HV value by sampling solutions in different fronts. In this work, test problems with up to 50 objectives are considered.

Large Populations

- An survey about MOEA/D and NSGA-II with large pupulation, e.g. 10,000 individuals, is presented in [33].

- Dynamical Multiobjective Evolutionary Algorithm (DMOEA) [34]. This method is based on the principle of the minimal free energy in thermodynamics. The method defines a fitness function, which considers three aspects: the Pareto rank value of the individual, a function analog to the temperature and the crowding distance [26].

- Grid-Based Evolutionary Algorithm (GrEA) [35]. This method tries to strengthen the selection pressure toward the optimal direction while maintaining an extensive and uniform distribution among solutions with the help of a grid. Three grid-based criteria, based on grid dominance and grid difference, are included to compute the fitness of individuals.

Dimension Reduction

- Pareto Corner Search Evolutionary Algorithm (PCSEA) [36]. This algorithm does a dimensionality reduction searching corners of the Pareto front. Authors identify two classes of corners. The minimization is made using the L_2 norm and

solutions that minimize either one of the objectives or the rest of the objectives simultaneously are preferred. The dimensionality analysis for the reduction is performed using a heuristic technique, which considers a rate between the number of non-dominated solutions in a reference set and the number of non-dominated solutions corresponding to the objective set obtained after omitting certain individual from the set.

We considered in this section the mathematical background and the related work in order to present, in Chapter 3, a continuation method for the treatment of MaOPs.

Chapter 3

Pareto Explorer

In this chapter we describe the general framework of the PE which is conceived as a global/local exploration tool for the treatment of MaOPs. We discuss the motivation about this computational tool in Section 3.1.

The PE consists of two principal phases. The first phase is about how to obtain a global optimal solution for a given MaOP, which is explained in Section 3.2. The second phase is concerning to the local exploration of the optimal solutions both in objective and decision spaces.

The local exploration with PE has two different approaches which are detailed in Sections 3.3 and 3.4. On one hand, we consider the search of optimal solutions around an initial optimal solution. On the other hand, we consider the steering along a given direction, which is the main contribution of this work.

3.1 Motivation

As stated in Section 2.3, the solution set of a MOP typically forms a $(k-1)$-dimensional manifold where k is the number of objectives involved in the given MOP. In practice, it is in most cases desirable to to attain a solution set which covers the entire PF uniformly.

There exist different performance indicators use to compare the quality of the solution set obtained by certain algorithm. These indicators usually map the PF to a single value. Examples of performance indicator are the the Generational Distance [37], the Inverted Generational Distance [38], and the Averaged Hausdorff Distance (Δ_p) [39].

In the practice we can obtain good values for the performance indicators with a solution set of few elements of a given MOP, provided that, these elements present an uniform distributions on the PF. However, if we want to obtain a similar accuracy for a problem with more objectives, that is, a MaOP, then we need to increase the number of elements in the solution set.

Notice that, although for a MOP with $k = 2$ a solution set with 50 uniform spread elements can produce good indicator values. For a MOP with three objectives ($k = 3$) this is may be not enough. If we consider that for the case $k = 2$ we have $50 = 50^{k-1}$ solutions, then for a $k = 3$ we need $50^2 = 2500$ solutions. Thus, if we have a MaOP with $k = 10$ this means to compute 50^9 solutions which is computationally costly. In addition, when the MaOP involves a DM the rating of all these solutions it is not feasible.

Reference point methods (see Section 2.3.2) are an alternative to treat a given MaOP. However, a unique solution may be not so useful for the DM. On the other hand, some of the evolutionary algorithms for MaOP (see Section 2.4) allow to indicate the DM preferences but actually the DM has not interaction with the obtained solutions at real time. That is, the algorithm must finish and then the DM evaluates the obtained solutions; if the DM identifies a region of interest between two points, then the algorithm or another method is executed again until the DM is satisfied, which requires time and computer resources.

The two phases of PE arise to provide an efficient tool capable to solve the MaOP considering the DM preferences (computing initial solution) and providing his/her an interaction in real time (local exploration).

3.2 Computing an Initial Solution

The first step of PE consists of computing a global optimal solution. This is necessary to star with the next phase. In this work we compute the initial optimal solution via to different methods.

The former is the Newton method for MOPs, which is used as the corrector step of the Pareto Tracer method (see Section 2.3.1). This approach has the issue that Newton method for MOPs is not a global method and it does not consider the preferences of the DM. However, when we do not have an idea about the given problem, we can use this method as a efficient way to get an initial local solution.

The latter is the reference point problem (see Section 2.3.2) using the Tchebycheff scalarizing function, given by Equation (2.71). Here the preferences of the DM are considered by the chosen reference point used to solve Equation (2.71) which guarantees an staring optimal point close to a region of interest.

Nevertheless, PE contemplates this phase with a global solver , e.g. an evolutionary algorithm, which takes into a consideration the DM preferences and also provides a set with a few elements on different regions of the PF and PS to star with the local search (see Section 2.4). We provide more details of the future work in Chapter 5.

3.3 Local Complete Exploration

The first scenario is that we have an initial optimal solution and we want to find several well spread other solutions around this one. We describe this approach without going into details, even though it is an important part of the PE, due to it is out of the scope of this work.

The neighborhood exploration can be done both in objective and decision spaces. Let be $x^* \in \mathbb{R}^n$ be the initial optimal solution. The idea is to obtain N optimal neighbors x_i $i = 1, \ldots, N$ with a good distribution on the PS or PF, depending on the selected space. In broad strokes the procedure is as follows:

1. Define the best possible distribution around x^* for a search in the decision space or around $F(x^*)$ for a search in the objective space.

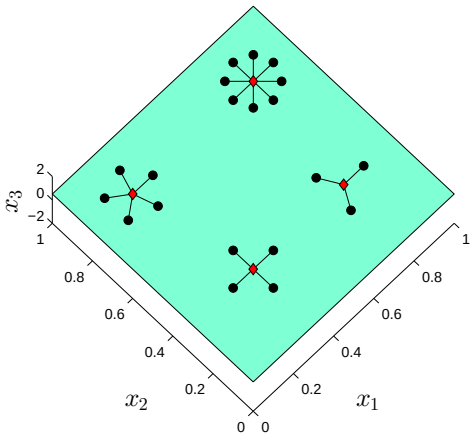
2. Use the appropriate predictor step of the Pareto Tracer (depending on the constraints) to find the corresponding $\mu_i,\ i = 1 \ldots, N$, for each vector on the previous computed distribution.

3. Compute $t_i,\ i = 1, \ldots, N$, using (2.23). Here, τ is the radius of the neighborhood.

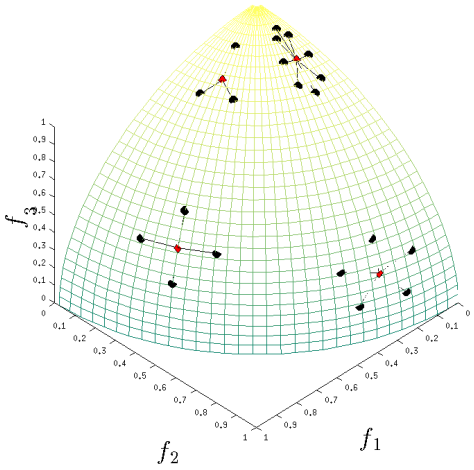
4. Compute $p_i = x^* + t_i\nu_i,\ i = 1, \ldots, N$.

5. Compute $x_i,\ i = 1, \ldots, N$, by using the appropriate Newton method starting at each p_i.

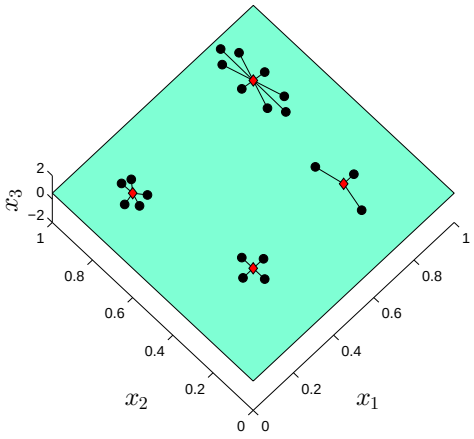
We can see in Figure 3.1 the difference between the neighborhood exploration for a search in decision and in objective space. The search in decision space of is shown in Figures 3.1a and 3.1b, for this case we have a uniform distribution along the PS. On the other hand, in Figures Figures 3.1a and 3.1b we can see the steering in objective space, which presents a uniform distribution along the PF.



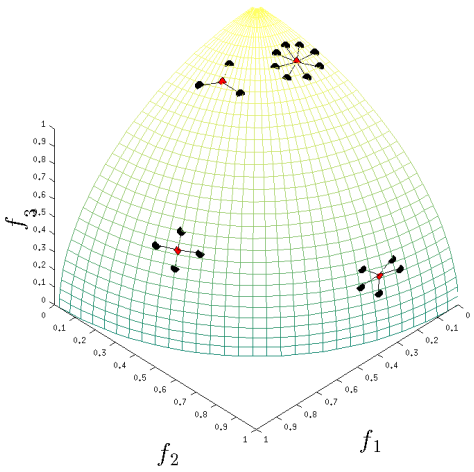
(a) Equidistant solutions in the decision space (PS).



(b) Image (PF) of the equidistant solutions in decision space.



(c) Decision space (PS) of the equidistant solutions in objective space.



(d) Equidistant solutions in objective space (PF).

Figure 3.1: Example of the complete exploration on a MOP with $k = 3$, $n = 3$ and different values of N, $N = 3, 4, 5, 8$. On the top we can see the exploration in decision space and on the bottom we can observe the exploration in objective space. Starting points are the same for both cases in order to show the difference between these approaches.

3.4 Steering the Search into User Defined Directions

As we explained in Section 2.3.1, Pareto Tracer Method [14] provides a relationship between the objective and the decision space. We can use this relationship, given by Equation (2.28), to follow a given direction along the PS/PF of a given MaOP. The way to define the steering depend on the desirable movement but Algorithm 1 provides the pseudocode to illustrate a standard version for the PE steering phase. Hereinafter we use the PE to refer to this phase.

Algorithm 1 General Case of PE Steering

Require: A continuous MaOP.
Ensure: A solution set for the given MaOP.
1: Get an initial optimal solution to the given MaOP
2: Give the obtained solution to the DM.
3: **while** DM is not satisfied **do**
4: The DM provides a direction to steer the search.
5: Ask for the desirable number of solutions and the distance between them.
6: Find solutions according to the given direction.
7: **if** A new solution is found **then**
8: Give the obtained solution set to the DM.
9: **end if**
10: **end while**

3.4.1 Steering in Objective Space

In certain cases, the DM is interested in a certain region of the solution set. A common way to define this region is via the value of each function. The PE starts with an optimal solution and, in this case, the idea is to explore the PF according to a given direction, which represents the preferences of the DM in objective space.

In addition, the DM can specify the number of solutions to get and the distance between them. Once that the method returns a new solution, the DM can stop the process if he/she is satisfied, otherwise, the method continues with the same direction or with a new one.

The DM can easily define a direction in objective space. For example consider a MOP with three objective functions f_1, f_2, and f_3 an the following scenario; the DM has an optimal solution for this problem but he/she is not satisfied. The DM wants to minimize as much as possible the first function, starting from his/her optimal solution. Thus, the direction $d = (-1, 0, 0)^T$ may be considered to get it.

A lot of options can be considered for the above example. The direction $d = (-1, 0, -1)^T$ means a simultaneous reduction in f_1 and f_3, meanwhile, the direction $d = (-1, -5, 0)^T$ implies a reduction of f_1 and f_2 in a bigger amount for the second one (see Example 4). The normalization of the given direction by the DM is not necessary.

For a movement in objective space, it is important to have in mind the notion of Pareto optimality. If we have an optimal solution then it is not possible to improve all the functions simultaneously. However, the DM can define a direction formed only by negative components, or a direction which does not involve an optimal movement. This is not a limitation for the PE.

Once we have obtained an initial optimal solution, the PE method takes any direction and then it seeks for a new direction which guarantees an optimal movement. In order to preserve the preferences of the DM, the new direction is the closest one to the original one as we explain in the following.

Let $d_k \in \mathbb{R}^k$ be a given direction. If x^* is a solution of (2.24) with its corresponding weight vector α, then the best direction to move the point $F(x^*)$ along the PF, according to d_k, is given by the orthogonal projection of d_k on the linearization of PF at the poin $F(x)$, i.e.

$$d = Q_2 Q_2^T d_k, \tag{3.1}$$

where Q_2 is defined as in Equation (2.32).

We can see in Figure 3.2 the geometrical idea of a orthogonal projection. We show a convex PF with two objective functions, the image of an initial optimal solution $F(x^*)$, and a given direction d_k. The linearization of the PF at the point $F(x^*)$ is denoted by $T_{F(x^*)}$, the orthogonal projection of d_k is realized on this space. In this case, the matrix Q_2 is formed by a unique vector in 2D. We can see that the new direction d is in the space $T_{F(x^*)}$.

We consider the normalized direction of d, i.e. $d := d/\|d\|$ and we can now compute the desired vector $\nu_d \in \mathbb{R}^n$ such that $J\nu_d = d$ using (2.29) and (2.35). The normalization of d is useful to compute the step length t, defined as in Equation (2.22), for the predictor. Thus, we can utilize (2.23).

Now we have the direction ν_d in decision space, which produces a movement in objective space to follow d_k. Also, we can set a τ value as in Equation (2.22) in order that the image of the new solution is to an approximate distance of τ from the image of the initial optimal solution.

Finally, the corrector step is applied in the same way that the Pareto Tracer (see Section 2.3.1), i.e. we use the Newton method for MOPs, depending on the constraints. This is because, for this case, the corrector has the same purpose. That is, if the predictor is out the PS, then the corrector returns it again to the PS.

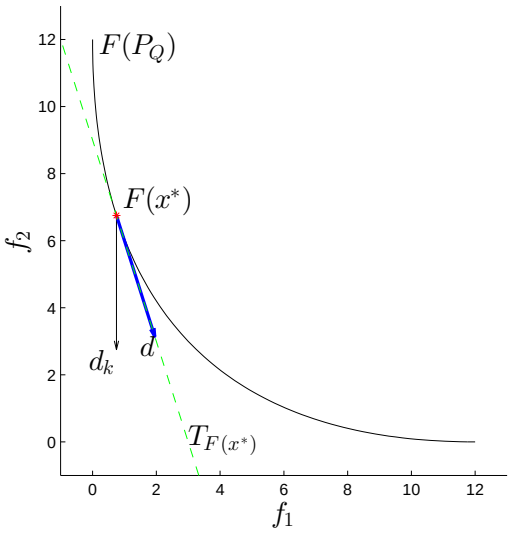


Figure 3.2: Hypothetical example of how to compute the orthogonal direction in objective space. The linearization of the PF at the point $F(x^*)$ is denoted by $T_{F(x^*)}$ and the orthogonal projection of d_k on $T_{F(x^*)}$ is the vector d.

Stopping Criteria

The DM can specify the number of steps for the PE. However, the movement along a given direction may produce a backtrack in the resulting path or there are no more optimal solutions in this direction. Thus, we define different stopping criteria for this case considering the non promising steps in decision space.

We consider three different stopping criteria:

1. $\langle d, d_k \rangle \leq \epsilon_1$. This means that the orthogonal projection of the direction d_k on the linearization of the PF is practically equal to zero vector. So, there not exist a good movement for this point along the PF according d_k. We consider this condition with a tolerance of $\epsilon_1 > 0$ instead of the norm of d in the numerical implementation.

2. $\text{sign}[(J(x_i)\nu_i)_j] = -\text{sign}[(J(x_{i+1})\nu)_{j+1}] \ \forall \ j \in \{1, \ldots, k\}$. This happens when there is a backtrack in the method which means a swing around a point on the PF and no more promising steps.

3. $\|\alpha\|_\infty = 1 - \epsilon_2$. For unconstrained MOPs this condition represents a corner of the PF. If the method reaches a corner, it ends. The tolerance in the numerical implementation is given by ϵ_2.

Examples

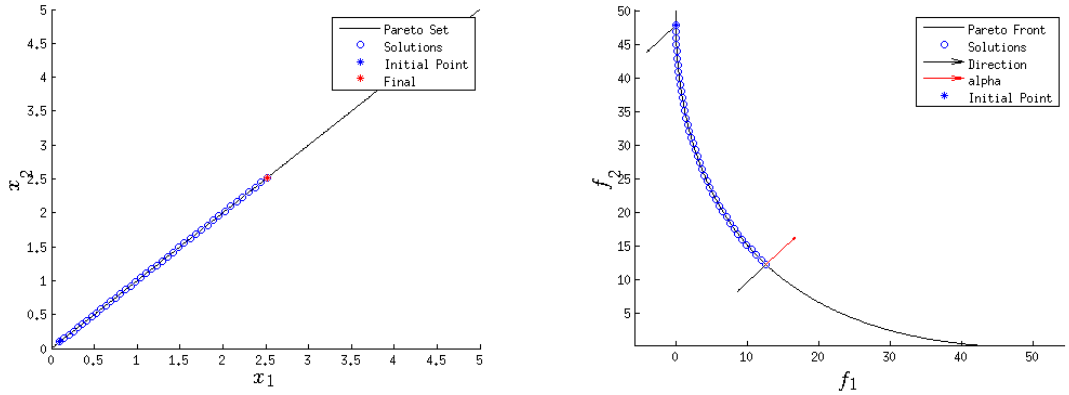
In this section we exemplify the three stopping criteria described in Section 3.4.1. We also include one example which shows a different behavior in a particular direction for convex and non convex functions.

Along this chapter we introduce some test problems. We use those problems with $k = 2, 3$, only as examples. In Chapter 4 we take some of these problems define for $k \geq 4$ and use them for the numerical results for MaOPs.

Example 1. Projection equal to zero For this example we consider Binh problem [40]. This is a convex problem with two objective functions and two decision variables defined as

$$\begin{aligned} f_1(x) &= x_1^2 + x_2^2 \\ f_2(x) &= (x_1 - 5)^2 + (x_2 - 5)^2. \end{aligned} \tag{3.2}$$

For the movement in objective space we take the direction $d_k = (-1, -1)^T$ and the optimal solution $x_0 = (0, 0)^T$ as starting point. We show in Figure 3.3 the PS and PF for this example. We can see in Figure 3.3b that α vector is collinear to the given direction d_k at the last optimal solution.



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

Figure 3.3: Example of the first stopping criterion for the movement in objective space.

We can numerically verify the obtained result for this example. The Jacobian of Equation 3.10 at the point $x^* = (2.5, 2.5)^T$ is given by:

$$J^* = J(x^*) = \begin{pmatrix} \nabla f_1(x^*)^T \\ \nabla f_2(x^*)^T \end{pmatrix} = \begin{pmatrix} 2x_1 & 2x_2 \\ 2(x_1 - 5) & 2(x_2 - 5) \end{pmatrix} = \begin{pmatrix} 5 & 5 \\ -5 & -5 \end{pmatrix}.$$

Solving the KKT system we obtain $\alpha^* = (0.5, 0.5)^T$. Notice that $\alpha_1^* + \alpha_2^* = 1$ and,

$$J^T\alpha^* = (\nabla f_1^*, \nabla f_2^*) \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 5 & -5 \\ 5 & -5 \end{pmatrix} \begin{pmatrix} 0.5 \\ 0.5 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}.$$

Now we observe that $d_k = -2\alpha^*$. Thus, an orthogonal vector to α^* is also orthogonal to d_k. This condition is equivalent to the first stopping criterion of Section 3.4.1. If we compute the QR factorization of α^* we obtain a $Q = (q_1, q_2) \in \mathbb{R}^{2\times 2}$, the first column of this matrix is the normalization of α^* while q_2 is a unit vector orthogonal to α.

According to the procedure of the PE method we must compute the orthogonal projection of d_k on q_2 to get the vector d. As d is in the span of q_2 then d is orthogonal to α^*, and as we described below, it is also orthogonal to d_k, i.e. $< d, d_k >= 0$.

In this example we found a point x^* such that $\nabla f_1(x^*) = -\nabla f_2(x^*)$. Since we have a convex problem, we used the direction $d_k = (-1, -1)^T$ to do this. However, we take a different direction to achieve the same for a concave problem. We show this in the next example.

Example 2. Concave function and corner of the PF For this example we use Fonseca problem [41]. This is a problem with concave PF with two objective functions and two decision variables defined as

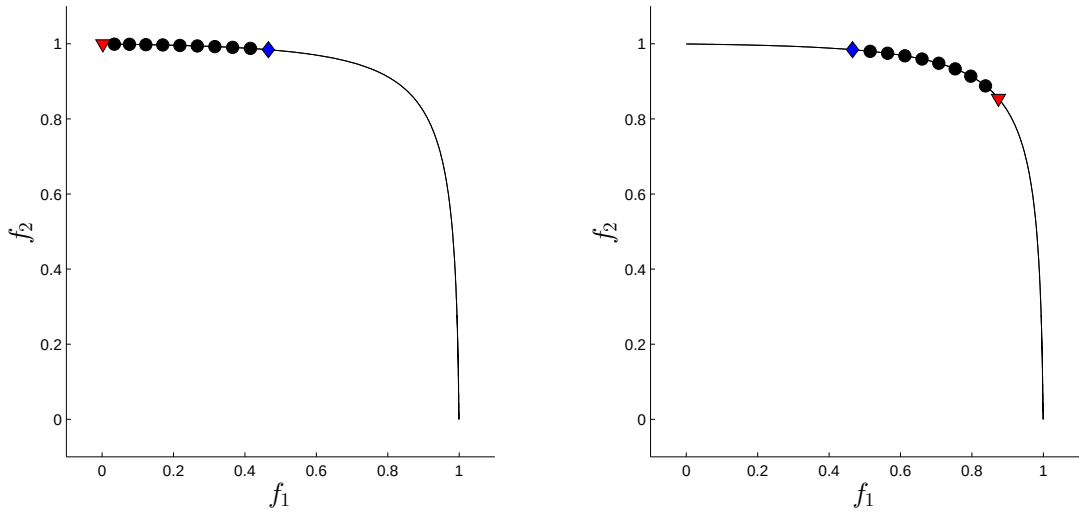
$$\begin{aligned} f_1(x) &= 1 - \exp(1 - (x_1 - 1)^2 - (x_2 + 1)^2) \\ f_2(x) &= 1 - \exp(1 - (x_1 - 1)^2 - (x_2 - 1)^2). \end{aligned} \tag{3.3}$$

Our goal in this example is to get an optimal solution x^* which satisfies $\nabla f_1(x^*) = -\nabla f_2(x^*)$. We show results for these two directions in Figure 3.4, we only plot the PFs.

We take two directions in objective space $d_{k1} = (-1, -1)^T$ and $d_{k2} = (1, 1)^T$, the initial optimal point $x_0 = (-0.44, 0.44)^T$ is the starting point for both directions and the value for the objective function at the point x_0 is $F(x_0) = (0.942, 4659)^T$. We show results for these two directions in Figure 3.4 in objective space.

Equation (3.3) satisfies $\nabla f_1(x^*) = -\nabla f_2(x^*)$ at the point $x^* = (0, 0)^T$. In this point $F(x^*) = (0.8647, 0.8647)^T$, $\nabla f_1(x^*) = (-0.2707, 0.2707)^T$ and $\nabla f_2(x^*) = (0.2707, -0.2707)^T$.

We can see in Figure 3.4b that the direction d_{k2} reaches the desirable solution. This behavior is the opposite to the one obtained for a convex function for the direction $d_{k1} = (-1, -1)^T$ (see Example 1 of this section).



(a) Example with d_{k1}

(b) Example with d_{k2}

Figure 3.4: Problem with concave PF. Plot (a) shows the result for the direction $d_{k1} = (-1, -1)$. Plot (b) shows the result for $d_{k2} = (1, 1)^T$.

Notice that, if we rotate 45° our plots, the direction d_{k1} represents a descent direction while d_{k2} produces an ascent direction. So, if we combine the direction d_{k1} with a convex problem, we can reach the *bottom* of the curve. On the other hand, with d_{k2} and a concave problem we can reach the *top* of the curve. As we notice in Figure 3.4a, when we use the direction d_{k1} in a concave problem we finish in a extreme of the PF. Indeed, if the initial staring point is in the right side of the PF, the final solution will be the right-down corner (see Figure 3.5). We will exploit this fact in Chapter 4.

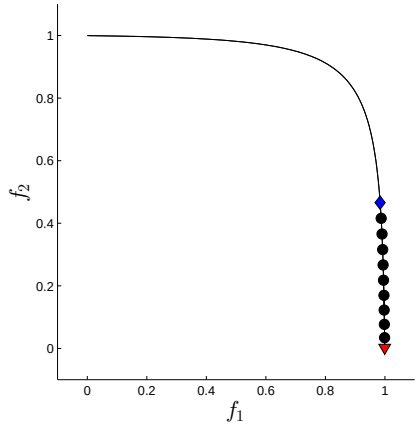


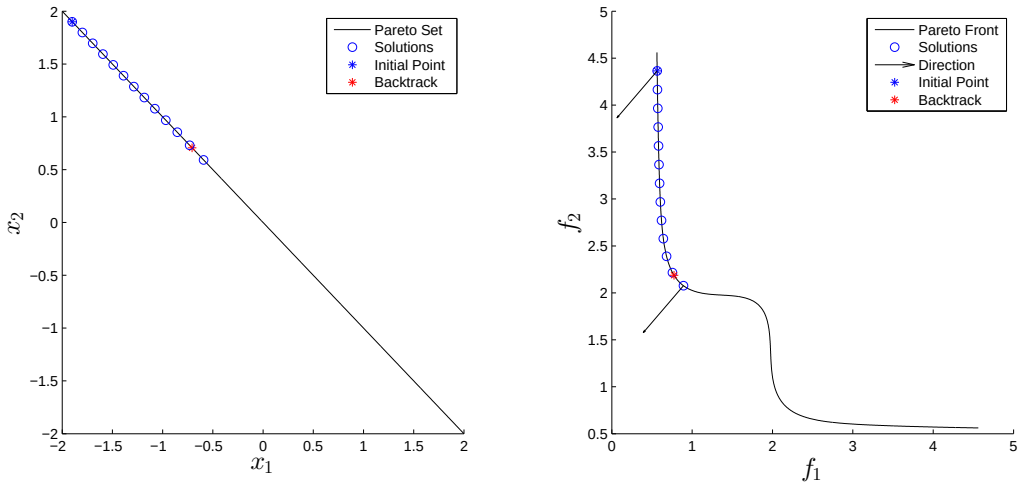
Figure 3.5: Example of Fonseca problem with a starting point on the right side of the PF.

Finally, we see in this example that the stopping criterion used in Figures 3.4a and 3.5 is the third one defined in Section 3.4.1, that is $\|\alpha\|_\infty = 1 - \epsilon_2$.

Example 3. Backtrack in the steps Now we consider the Dent problem [42]

$$\begin{aligned} f_1(x) &= \frac{1}{2}\left(\sqrt{1+(x_1+x_2)^2} + \sqrt{1+(x_1-x_2)^2} + x_1 - x_2\right) + 0.85\,\mathrm{e}^{-(x_1-x_2)^2} \\ f_2(x) &= \frac{1}{2}\left(\sqrt{1+(x_1+x_2)^2} + \sqrt{1+(x_1-x_2)^2} - x_1 + x_2\right) + 0.85\,\mathrm{e}^{-(x_1-x_2)^2} \end{aligned} \tag{3.4}$$

Equation (3.4) is a problem with convex-concave PF defined by two objective functions and two decision variables. For this case we take the vector $x_0 = (-2, 2)^T$ as starting optimal solution and we steer the search in the direction $d_k = (-1, -1)^T$. We show the result of this example in Figure 3.6.



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

Figure 3.6: Example of the third stopping criterion

We can see a backtrack in the steps of the method. We have again the direction $d_k = (-1, -1)^T$, however, in this case we do not obtain an optimal solution x^* with $\nabla f_1(x^*) = -\nabla f_2(x^*)$. This is because although we reach the *top* of the 45° rotate curve, the optimal solution which satisfies the above it is not in this point.

So, the orthogonal projection of d_k on the linearization of the PF is different to the zero vector, even when the solution is near to the *top*. Now, as the method tries to follow the given direction, it produces an oscillation around the final solution showed in Figure 3.6.

Example 4. Different directions Now we consider the DTLZ2 test problem from [43] with three objectives and hundred variables given by

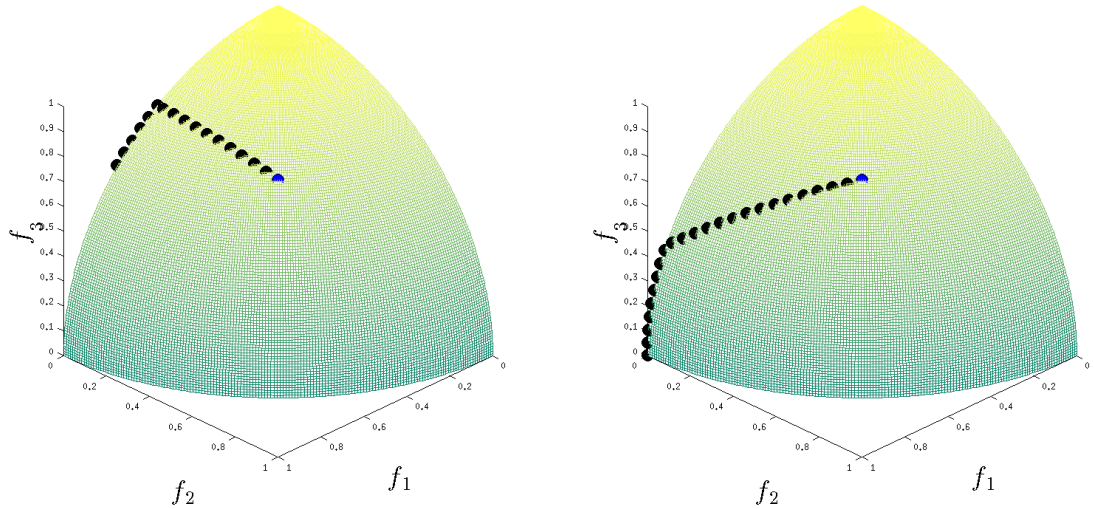
$$\begin{aligned} f_1(x) &= (1+g(X_k))\prod_{i=1}^{k-1}\cos(0.5x_i\pi), \\ f_2(x) &= (1+g(X_k))\sin(0.5x_{k-1}\pi)\prod_{i=1}^{k-2}\cos(0.5x_i\pi), \\ &\vdots \\ f_k(x) &= (1+g(X_k))\sin(0.5x_1\pi), \end{aligned} \tag{3.5}$$

where

$$g(X_k) = \sum_{x_i \in X_k} (x_i - 0.5)^2, \tag{3.6}$$

and X_k is a vector of the remaining attributes $(x_k, \ldots, x_n)$ for $n > k$.

Equation (3.5) is a problem with concave PF. We test the PE method with two directions in $\mathbb{R}^3$, and our goal is to reach the corner of the PF. The first direction is given by the vector $d_{k1} = (0, -1, 0)$ and the second one is the vector $d_{k2} = (0, -1, -1)^T$. We take the same starting point for both cases and we show the results for these cases in Figure 3.7.



(a) Example for $d_{k1} = (0, -1, 0)^T$

(b) Example for $d_{k2} = (0, -1, -1)^T$

Figure 3.7: Example of a movement in objective space for two different directions.

We can see in Figure 3.7a the plot for the direction d_{k2}. For this direction the goal is to reduce the value of f_2. We notice that this happens, the method reduces

the value of f_2 as much as possible and, as the orthogonal projection of d_{k1} on the PF is not the zero vector, then it continues with the steps until a backtrack in the steps is obtained.

On the other hand, we observe in Figure 3.7b the result for the direction d_{k2}. Here we want to reduce both f_2 and f_3, this produces a different kind of movement along the PF. We notice the simultaneous reduction of two functions causing that the method continues the movement in the boundary of the PF until a corner is reached.

We note with this example that is not enough to steer the search only in one direction. If we want to obtain more information of an unknown problem, we must try with different directions. We notice that even when we get the minimum value of certain function, it does not guarantees that the method is in a corner of the PF.

3.4.2 Steering in Decision Space

In a wide variety of applications, the values of decision variables are crucial, since these variables commonly represent the available resources, which are generally limited. Constraints allow us to have a control on these variables in a MaOP.

The DM can suggest a non optimal initial point expressing his/her preferences in decision space. Then staring from this point a particular method can obtain a set of optimal solutions. One of the issues of this approach is that normally methods that solve MaOPs only consider the values of the objective functions. Thus, if an important resource does not have an impact in objective functions, then the solution set could not contain the best value for this resource.

Usually, the DM wishes to improve the value for a certain resource, that are more important than the others, as much as possible without optimality loss. Likewise to the movement in objective space, the PE provides the option to steer the movement along the PS in a given direction, that is, we can get optimal solutions which improve the values of the desired decision variables.

The idea to move from a point $x \in \mathbb{R}^n$ along the PS following a direction $d_n \in \mathbb{R}^n$ is similar to above. We use the orthogonal projection of d_n, but in this case on a linearization of the PS at the point x.

One way to do this is to compute the corresponding vector ν_{d^j} for each vector $d^j,\ j = 1, \ldots, k-1$, which are the columns of the matrix Q_2, using (2.29) and (2.35). We can obtain a basis of the PS at the point x with the matrix formed by the columns ν_{d^j}. Let Q_v denote such a matrix. Once that we have this matrix, we can compute the best direction according d_n via

$$d_v = Q_v Q_v^T d_n. \tag{3.7}$$

Then, for this case, the predictor is given by,

$$p = x + td_v. \tag{3.8}$$

If we want a fixed distance between two solutions in objective space of τ, then we use (2.23). Otherwise, if we want for two consecutive solutions, $\|x_i - x_{i+1}\| \approx \tau$, then we simply set $t = \tau$. The corrector is also given by the Newton method for MOPs.

We can see in Figure 3.2 the geometrical idea of an orthogonal projection. We show a non linear PS with two decision variables, the initial optimal solution x^*, and a given direction d_n. The linearization of the PS at the point x^* is denoted by T_{x^*}, the orthogonal projection of d_n is realized on this space. In this case the matrix Q_v is formed by a unique vector in 2D. We can see that the new direction d is in the space T_{x^*}.

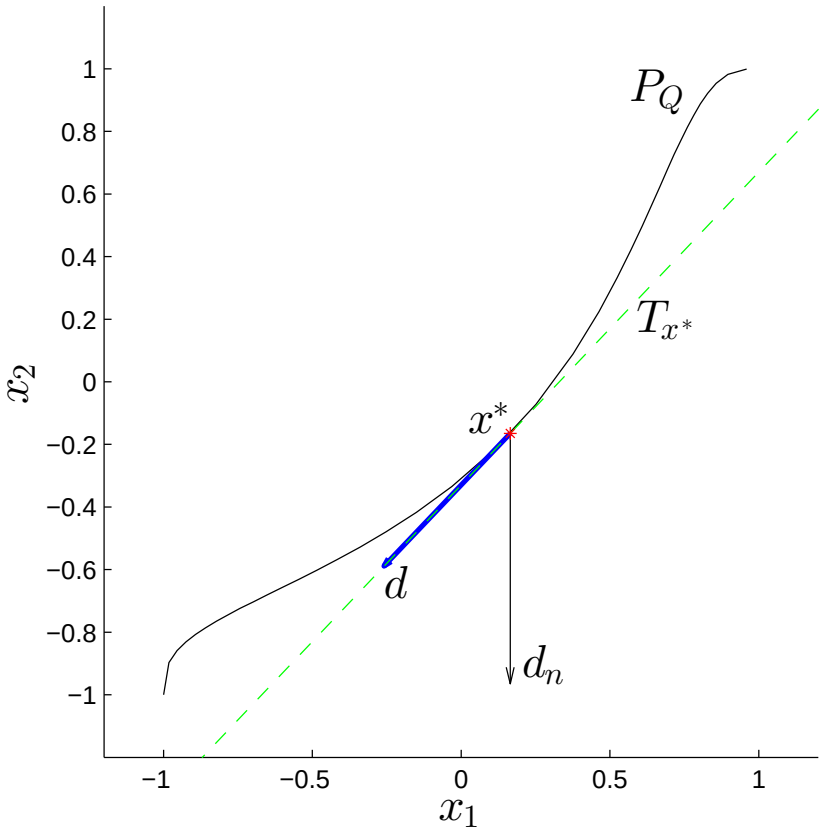


Figure 3.8: Geometrical example of how to compute the orthogonal direction in decision space. The linearization of the PS at the point x^* is denoted by T_{x^*} and the orthogonal projection of d_n on T_{x^*} is the vector d.

Stopping Criteria

For this case, we consider the following three stopping criteria:

1. $\|Q_v Q_v^T d_n\|_\infty < \epsilon_1$. We use this stopping criterion instead of the first one in objective space. This also means the absence of a good movement, but in this case in decision space.

2. $\text{sign}[x_i]_j = -\text{sign}[x_{i+1}]_{j+1}\ \forall\ j \in \{1, \ldots, k\}$. A backtrack in decision space can be detected with this condition.

3. $\|x_{i+1} - x_i\| \leq \epsilon_2$. If the point x_i is equal to x_{i+i} then there is not an improvement in the given direction, so the method stops. We consider a tolerance ϵ_2 for the numerical treatment.

Stopping criteria 1 and 2 are analogous to the first two criteria for a movement in objective space given in Section 3.4.1. Even for unconstrained problems, stopping criterion 3 of Section 3.4.1 has not an equivalent formulation in decision space, due to the fact that there are not a relationship between the KKT multiplier and the geometry of the PS. Indeed, generally speaking there is no correlation between the geometry of PS and PF.

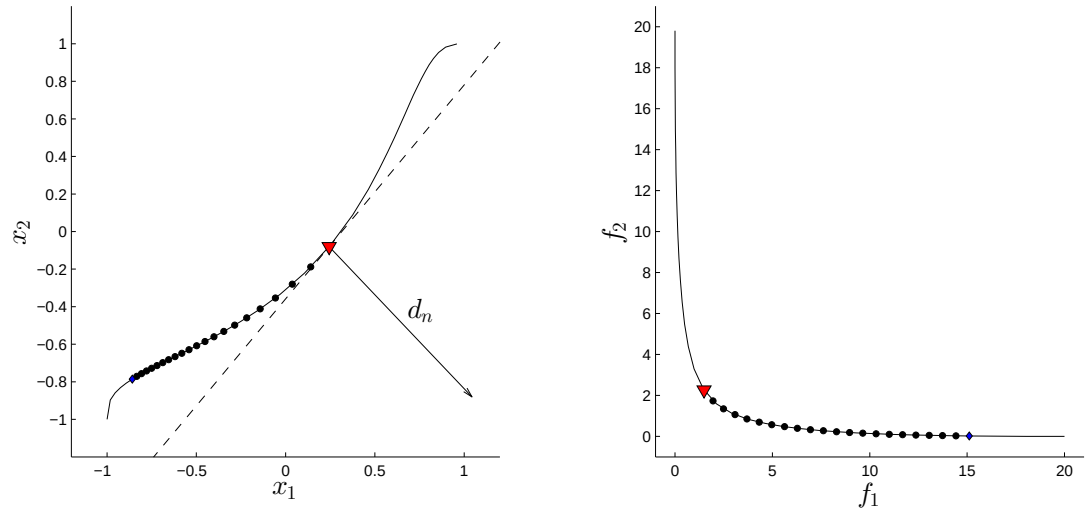
Examples

In this section we illustrate the stopping criteria of Section 3.4.2. As we explained in Section 3.4.1, we use some test problems with $k \leq 3$ for the examples and we take scalable problems with $k > 3$ for the numerical results in Chapter 4.

Example 5. Projection equal to zero For this example we consider the following problem [44]. This is a convex problem with a non linear PS defined as

$$f_j(x) = \sum_{i=1}^{n} (x_i - a_i^j)^2 + (x_j - a_j^j)^4, \qquad j = 1, 2. \tag{3.9}$$

where $a^1 = (1, \ldots, 1)^T \in \mathbb{R}^n$ and $a^2 = -a^1$. In this example we take $n = 2$, $k = 2$ and the direction $d_n = (-1, 1)^T$. The obtained result is shown in Figure 3.9.



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

Figure 3.9: Example of the first stopping criterion for the movement in decision space. Scatter line is the tangent of the PS at the last point which is represented by a triangle, we can see that the direction d_n is orthogonal to the tangent space.

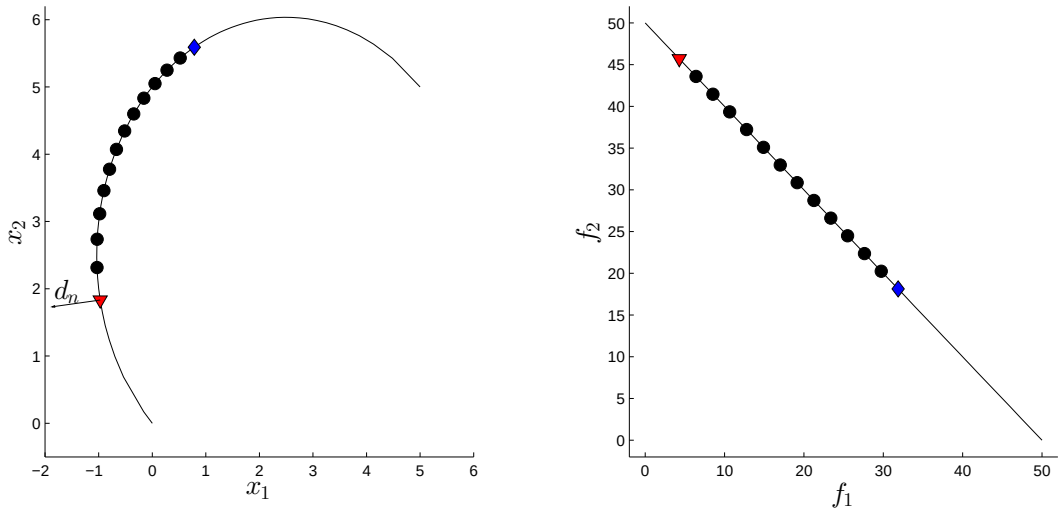
We can see in Figure 3.9a both the direction d_k and the tangent space at the final solution, it is clear that d_k is orthogonal to the tangent space, i.e. the orthogonal projection d is the zero vector, so there is not a possible movement at this point according d_k.

Example 6. Backtrack in the steps For this example we use again the Binh problem [40] with an equality constraint in order to get a non linear PS. Now this problem with two variables and two objectives is defined as

$$\begin{aligned} f_1(x) &= x_1^2 + x_2^2 \\ f_2(x) &= (x_1 - 5)^2 + (x_2 - 5)^2 \\ \text{s.t.} \quad & \|x - c\| = 0 \end{aligned} \tag{3.10}$$

where $c = (2.5, 2.5)^T$. The incorporated equality constraint is a circle which goes through two extremes of the PS of the unconstrained problem.

For this example we take the direction $d_n = (-0.9, -0.1)^T$. Result for this problem is shown in Figure 3.10, PS is in Figure 3.10a and PF is in Figure 3.10b.



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

Figure 3.10: Example of the second stopping criterion for the movement in decision space. We can see the direction d_n at the final solution, we notice that the orthogonal projection is between two solutions, i.e., a backtrack in the steps.

Dots in Figure 3.10 are the steps of the PE method. We can see in Figure 3.10a the direction d_n at the final point, this point is represented by a triangle. We notice that the orthogonal projection of d_k on the linearization of the PS is between the last two solutions, i.e. a backtrack in the steps.

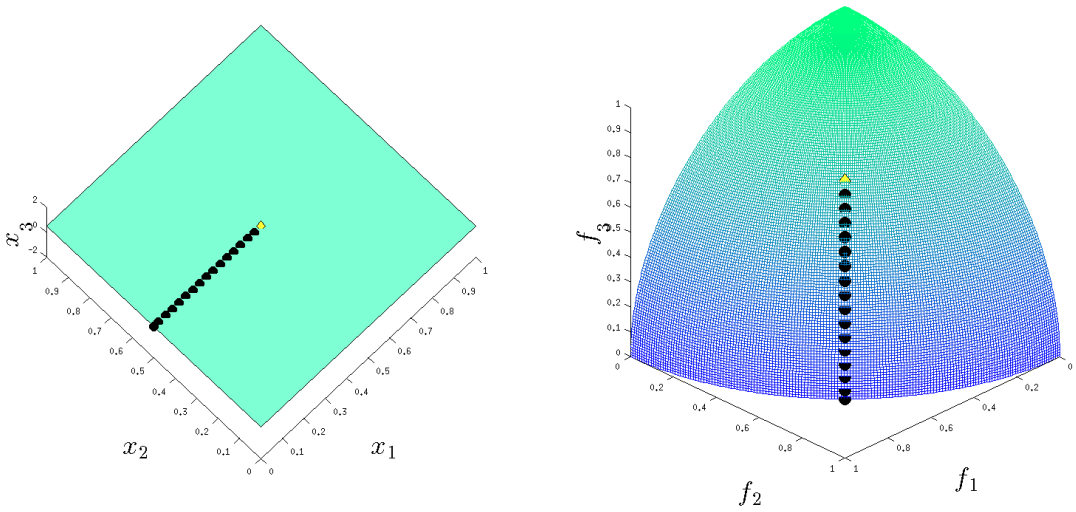
Example 7. No progress in the given direction Now we consider the DTLZ3 test problem from [43] given by

$$\begin{aligned} f_1(x) &= (1 + g(X_k)) \prod_{i=1}^{k-1} \cos(0.5 x_i \pi), \\ f_2(x) &= (1 + g(X_k)) \sin(0.5 x_{k-1} \pi) \prod_{i=1}^{k-2} \cos(0.5 x_i \pi) \\ &\vdots \\ f_k(x) &= (1 + g(X_k)) \sin(0.5 x_1 \pi), \end{aligned} \tag{3.11}$$

where

$$g(X_k) = 100 \left(|x_k| + \sum_{x_i \in x_k} \left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right) \right), \tag{3.12}$$

and X_k is a vector of the remaining attributes $(x_k, \ldots, x_n)$ for $n > k$. For this example we consider $n = 3$, $k = 3$ and the direction $d_n = (-1, 0, 0)^T$. Result of this case is shown in Figure 3.11.



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

Figure 3.11: Example of the third stopping criterion for the movement in decision space. The method stops when there is not a new point in the given direction.

We can see in Figure 3.11a a flat rectangle which is the real PS, starting point is in the center of this rectangle and the steps improve x_1 according d_n. Method stops when steps reach the boundary of the PS, this is because there is no more optimal solutions in the given direction causing a repeated solution.

The above comes from the fact that we have two dimension manifold in the PF (see Figure 3.11b), then a basis of a linearization of this manifold has two vectors in $\mathbb{R}^3$. So, we also have a two vectors basis in $\mathbb{R}^n$.

We focus on the final point x^* at the boundary of the PS of this example. The two vector basis in decision space at x^* produces an orthogonal projection d different to the zero vector. The computed predictor p in the direction d yields a non feasible solution and the corrector returns p to the PS. However, the new point is at a shorter distance than ϵ_2 to the point x^* and the method stops.

3.4.3 Steering in μ Space

As explained in Section 2.3.1, the vector μ provides a way to relate the objective and the decision spaces via (2.35), provided that the sum of its components is zero. Hence, we can use this vector as another alternative to define a direction in objective space.

If we have an optimal solution, then the vector μ represents the trade-off between this one and a new one. That is, in order to define a μ vector, the DM must specify both the improvement functions and the sacrifice functions; after that, the DM assigns values to each one. These values and its signs are in concordance with the preferences of the DM. A negative value means a reduction of certain function, a positive sign is related to a sacrifice function and a zero value can be used for an indifference function. The DM has a μ vector when the sum of these values is equal to zero.

For example, if we have five functions $f_1, \ldots, f_5$ and our goal is to improve the third function, then it is necessary to define the sacrifice functions to get a suitable vector μ. We can choose this vector in many different ways according to the preferences for the sacrifice functions. In this way, the vector $\mu = (0.25, 0.25, -1, 0.25, 0.25)^T$ indicates a sacrifice of the same amount for each function, the vector $\mu = (0, 0, -1, 0, 1)^T$ suggests that the whole sacrifice is taken by f_5, and the vector $\mu = (0.3, 0.2, -1, 0.4, 0.1)^T$ is a representation of a possible configuration with different weights.

Notice that, for the case $k = 2$, the only available options for the μ vector are the vectors: $\mu^{(1)} = (-1, 1)^T$ and $\mu^{(2)} = (1, -1)^T$. These vectors correspond to a left-up and right-down movement, respectively. An exploration in objective space along a biobjective PF may not be very interesting, however, if we use both $\mu^{(1)}$ and $\mu^{(2)}$ vectors we can compute an approximation to all the PF.

The PE procedure for this case is basically the same of Section 3.4.1. The difference is that, as we have a μ vector, then it is not necessary to compute an orthogonal projection. We only need Equation (2.29) to get the predictor. Finally, the step length is computed using Equation (2.23).

Now we consider a change in μ vector. The DM can change μ vector at any time of the exploration, this leads to two possible options. On one hand, the DM can immediately use the new μ vector to steer the search. On the other hand, the DM can take the difference vector as a transition between the current μ vector and the new one. This change can be done because the difference vector is also a μ vector, Lemma 3.1.

Lemma 3.1. *Let $\mu^{(i)}, \mu^{(i+1)} \in \mathbb{R}^k$ be two vectors that satisfy* (2.27)*, then the vector $\mu^d = \mu^{(i+1)} - \mu^{(i)}$ also satisfies* (2.27)*.*

Proof. As $\mu^{(i+1)}$ and $\mu^{(i)}$ satisfy (2.27), then the sum of components of μ^d is given by,

$$\sum_{j=1}^{k} \mu_j^d = \sum_{j=1}^{k} (\mu_j^{(i+1)} - \mu_j^{(i)}) = \sum_{j=1}^{k} \mu_j^{(i+1)} - \sum_{j=1}^{k} \mu_j^{(i)} = 0$$

□

Stopping Criteria We consider the special case in which $\mu_j^d = 0 \; \forall \; j \in \{1, \ldots, k\}$ as a stopping criterion for this problem because it would mean backtrack. This condition replaces the first condition mentioned on the case of objective space, the rest of these criteria are maintained.

Example 8. Different μ vectors Now we consider the DTLZ1 test problem from [43] given by

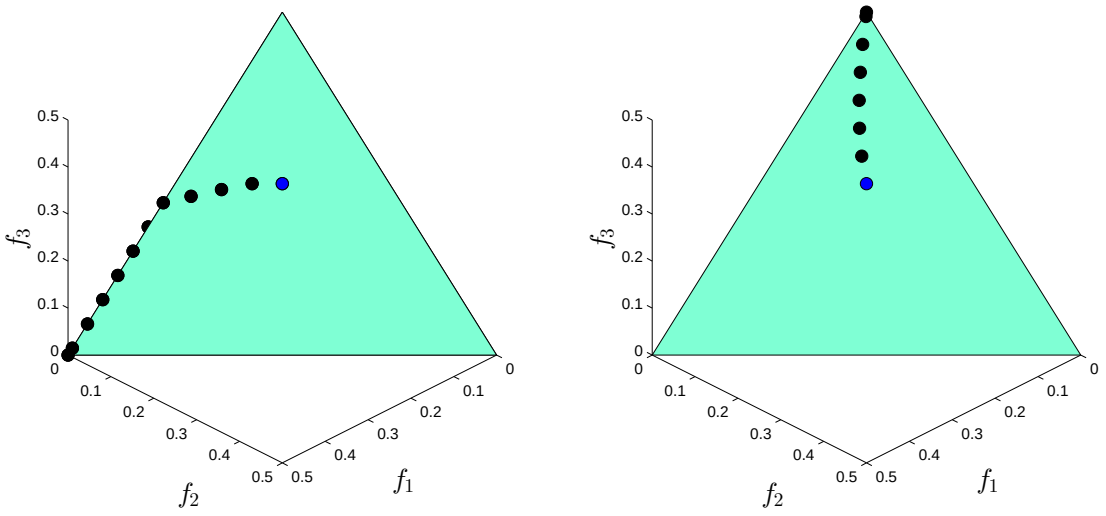
$$\begin{aligned} f_1(x) &= \frac{1}{2}(1 + g(X_k)) \prod_{i=1}^{k-1} x_i, \\ f_2(x) &= \frac{1}{2}(1 + g(X_k))(1 - x_{k-1}) \prod_{i=1}^{k-2} x_i, \\ &\vdots \\ f_{k-1}(x) &= \frac{1}{2}(1 + g(X_k))(1 - x_2)x_1, \\ f_k(x) &= \frac{1}{2}(1 - x_1)(1 + g(X_m)), \end{aligned} \tag{3.13}$$

where

$$g(X_k) = 100 \left(|X_k| + \sum_{x_i \in X_k} \left((x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)) \right) \right). \tag{3.14}$$

and X_k is a vector of the remaining attributes $(x_k, \ldots, x_n)$ for $n > k$. For this example we consider $n = 100$, $k = 3$ and the vectors $\mu^1 = (1, -1, 0)$ and $\mu^1 = (0; -1; 1)$.

In both cases we try to reduce f_2 but the sacrifice function is no the same. For μ^1 we increase f_1 and for μ^2 we increase f_3. We can see the results for this cases in Figure 3.12.



(a) Example with μ^1 **(b)** Example with μ^2

Figure 3.12: Example of a steering in μ space for DTLZ1. Plot (a) shows the result for the direction $\mu^1 = (1, -1, 0)$. Plot (b) shows the result for $\mu^2 = (0, -1, 1)$.

With this example we notice how we can use the vector μ to define a trade-off. We obtain the minimum for f_2 but with two different paths.

3.4.4 Minimal Change in Objective Space

Sometimes the most important aspect for a problem is the exploitation of the resources. Under this assumption, the DM may be interested in optimal solutions that have an equivalent performance in objective space, but with a high discrepancy in decision space. We can also consider the scenario where it is possible to reach an appreciable improvement of some variables with a little change of the function values.

In a MOP the DM may be in agreement with an optimal solution in objective space. Nevertheless, if this solution has not an appropriate value for an important decision variable, then the DM may be willing to sacrifice the function values in order to get a better value for the desired variable.

We can use the procedure of Section 3.4.2 to improve some values in decision space. However, this method does not take into consideration the objective function values, so we can get values that are far from the initial optimal solution. For this kind of problems, we need a different way to compute a direction in objective space, that is close to the given direction and yields a minimal change in objective space.

Let $d_n \in \mathbb{R}^n$ be a given direction in decision space, we can find a new direction $v^* \in \mathbb{R}^n$ as follows:

$$\begin{aligned} \min_{v \in \mathbb{R}^n} \quad & - < v, d_n > \\ \text{s.t.} \quad & Jv = 0 \\ & \|v\|_2^2 = 1. \end{aligned} \tag{3.15}$$

The SOP defined by Equation (3.15) reaches the optimum when the direction v is equal to d_n. The first constraint is due to the relation between the Jacobian and the change in objective space, we got a minimal change when the value of this constraint is near to zero. Finally, the norm of the direction v is considered so that we obtain a different solution to the zero vector.

Stopping Criteria As this is indeed a steering in decision space, the stopping criteria are the same.

Example 9. Difference with the steering in decision space. For this example we consider the following test quadratic problem [44].

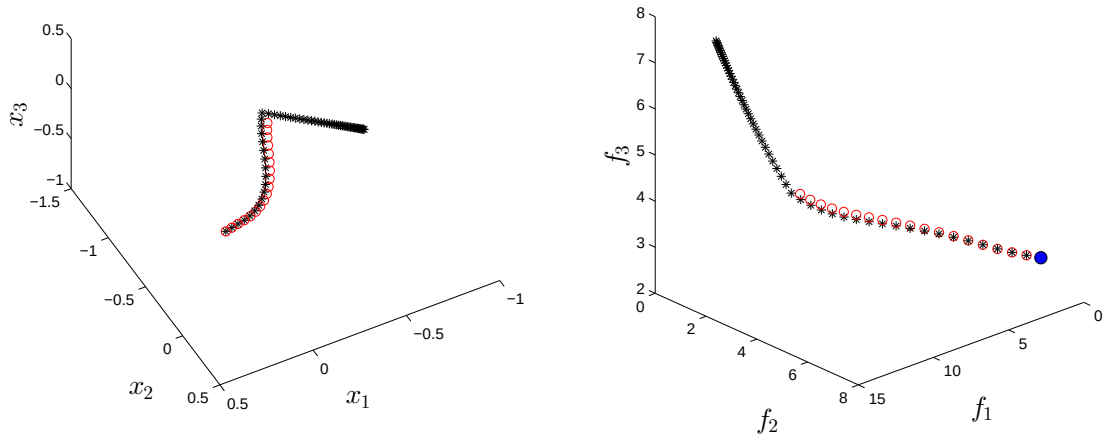
$$f_j(x) = \sum_{i=1}^{n} (x_i - a_i^j)^2 \tag{3.16}$$

where $a^j \in \mathbb{R}^n$ $j = 1, \ldots, k$. We take $k = 3$, $n = 3$ and the vectors $a^1 = (1, 1, 1)^T$, $a^2 = (-1, -1, -1)^T$, and $a_3 = (1, -1, 1)^T$.

We compare the obtained solution for the steering in decision space with the obtained one for the minimal change in objective space under the same initial conditions, the chosen direction is $d_n = (-1, -1, -1)^T$. Results are shown in Figure 3.13, we only plot the path obtained by each kind of steering in order to have a better visualization.

We can see the PF for both methods in Figure 3.13b. we notice that the marks that represent the path obtained by the steering of minimal change in objective space stops when the change become to increase with respect to the first steps. On the other hand, the steering of movement continues its movement.

For the decision space (see Figure 3.13a we notice that at first we practically have the same solution but the minimal change path becomes to differ with the iterations.



(a) PS **(b)** PF

Figure 3.13: Example of minimal change in objective space (o) versus the steering in decision space (∗).

3.4.5 Treatment of Dynamic Reference Point

Now we consider the DRPP defined in Section 2.3.2. Here we have a time dependent curve, which is formed by the DM preferences. Our goal is to obtain a set of optimal solutions, where each solution is the nearest one for each reference point on the curve (if we have a continuous case, then we take a discretization of the curve).

Let x_i^* be solution of Equation (2.72) for the reference point $z(t_i)$. We can solve the DRPP with an embedding method, that is, we simply take x_i^* as the starting point to solve Equation (2.72) for the next reference point $z(t_{i+1})$. Instead of this, we modify the procedure of Section Section 3.4.1 to obtain the next solution x_{i+1}^*.

We define a direction $d_k = z(t_{i+1}) - z(t_i)$ in objective space. After that, we can use the predictor of Section 3.4.1 with a little modification in the step size control. Let d be the orthogonal projection of d_k on the linearization of the PF at the point $F(x_i^*)$. For this case the value of τ in Equation (2.23) is set as the norm of d, so the step size t is given by:

$$t = \frac{\|d\|}{\|J\nu_d\|}. \tag{3.17}$$

Equation (3.17) allows to get a small step size when the change in the PF is low, according d_k, and vice versa. This is because the norm of d has a direct relationship with the orthogonal projection of d_k on the linearization of the PF.

As the goal of this approach is to get the nearest optimal solution for each point on the curve, then we use the point given by the predictor to solve Equation (2.72) for $z(t_{i+1})$. Namely, we have a new corrector. We do not need any stopping criteria, we compute both a predictor and a corrector for each point considered on the time dependent curve.

Example 10. Time dependent curve beyond the PF For this example we consider the quadratic problem defined by Equation 3.16. But for this case we take $k = 2$, $n = 100$ and the vectors $a^1 \in \mathbb{R}^n$ $a^1 := (1, \ldots, 1)$, and $a^2 \in \mathbb{R}^n$ $a^2 = -a^1$.

Results are shown in Figure 3.14. Here we see the time dependent curve and the predictor-corrector steps, we notice that all the predictors (+) are close to the correctors (*).

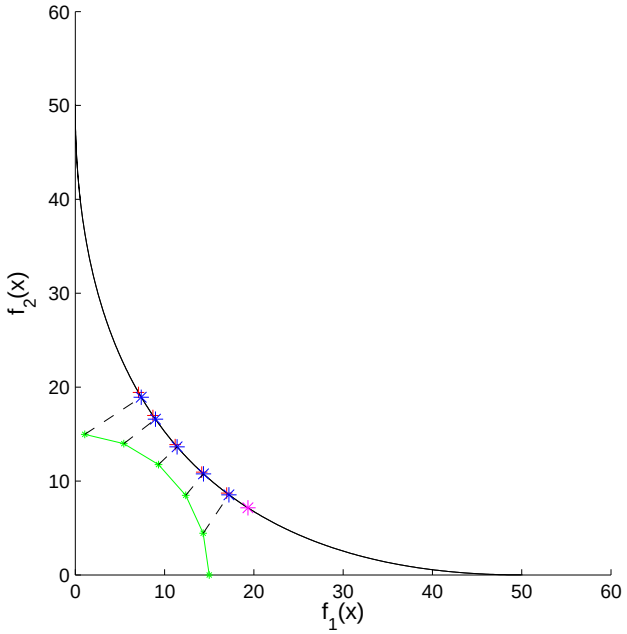
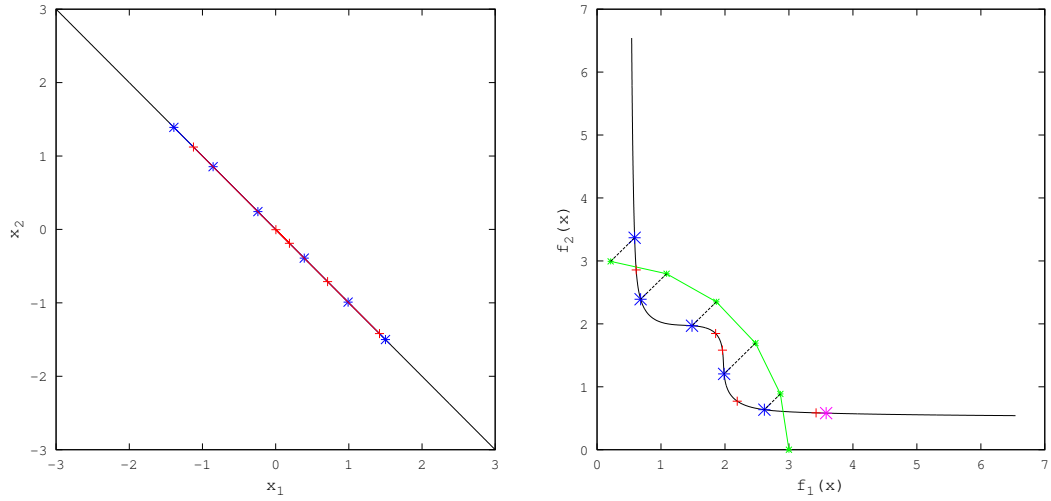


Figure 3.14: Example for the DRPP with an outside time dependent curve.

Example 11. Time dependent curve inside the PF Finally, we consider the Dent problem defined by Equation (3.4).

We present the result of this example in Figure 3.15 both for PS and PF. We notice that the method works for a time dependent curve with point inside the PF due to the Tchebycheff scalarizing function (2.71). However for this case, predictor and correctors is more distant with respect to the Example 10.



(a) PS for a search in decision space.

(b) PF for a search in decision space.

Figure 3.15: Example for the DRPP with an inside time dependent curve.

Chapter 4

Numerical Results

In this chapter, we present numerical results obtained by the PE method. Sections 4.1, 4.2, and 4.3 contain results for some of the benchmark MaOPs defined in Chapter 3. Section 4.4 is completely dedicated to a real world application.

We tested two different versions of the PE method. The first one is the *Newton Pareto Explorer* (PE-N), that is, the classical version of this method as we described in Chapter 3. The second one is the *Quasi-Newton Pareto Explorer* (PE-QN), this is a free Hessian version of the PE method which uses the BFGS method [14].

The results of this chapter are on MaOPs, that is, we can not plot the entire PF solution paths of the method as we did in Chapter 3. Along this chapter we include figures with the radar chart [45] which allows to show iterations that PE takes for the solution of a given problem.

The radar chart representation is useful because we can show a normalization of all the values both objective functions and decision variables, we only need the *utopian* and *nadir* points. If we do not have the exact values for the utopian and the nadir points, we can use approximations of them. Once we have values for the utopian and nadir points, we can compute the normalization of the objectives values as follows.

As our objective is to reduce the values of a given MaOP, we set the utopian on the circle and the nadir on the center of the wheel, i,e., each component of the utopian is set at the unit circle with coordinates $(\cos(\theta_i), \sin(\theta_i))$, with $\theta_i = 2(i-1)\pi/k$, $i = 1, \ldots, k$. The nadir point is used for the normalization.

Let $Y = (y_1, \ldots, y_k)^T$ be an element of the PF of a given MaOP, the normalization r_i for the radar chart is given by

$$r_i = 1 - \frac{y_i - z_i^{utp}}{z_i^{nad} - z_i^{utp}}, \tag{4.1}$$

where z^{nad} and z^{utp} denote the nadir and the utopian point, respectively.

Thus, the coordinates of each component y_i of the Y vector on the radar chart are given by

$$(r_i \cos(\theta_i), r_i \sin(\theta_i)), \quad i = 1, \ldots, k. \tag{4.2}$$

Analogously, we can define a similar representation for the PS taking a normalization in decision space. In order to maintain consistency with the objective space representation, we take the smaller values of x as the furthest ones from the center.

We use lower and upper bounds of each problem in order to do the normalization in decision space for the radar chart results along this chapter. We plot both decision and objective spaces (at the left and right sides, respectively) in four different moments: initial, quarter, middle, and final optimal solution.

An important issue of the radar chart is that we may lose the notion of the magnitude of functions, which is due to the normalization. For example, if the method did a reduction of 10 units for some function, this has a different impact in the char depending of the magnitude of each function, i.e., this change may be very notorious or imperceptible. Hence, in all the tested problems we include a table with the values of the objective functions or the decision variables, according to the type of movement selected, in the four indicated steps.

We have experimentally observed that we have practically the same radar chart and the same quality for the numerical results of PE-N and PE-QN at each iteration. Thus, we include only one of these versions. We indicated the selected version for each case. Additionally, we present one table with computational efforts for each problem. This is in order to compare the performance of the two versions: PE-N and PE-QN.

4.1 DTLZ1

The scalable DTLZ1 problem was defined in Section 3.4.3 in Equation (3.13). Along this section we consider DTLZ1 with 15 decision variables and 10 objective functions.

We tested this problem with four of the five kinds of movement described in Chapter 3. We discard the steer of minimal change in objective space due to the fact that this kind of movement does not produce new solutions for the DTLZ problems (see Section). Likewise, we consider the same four cases for the DTLZ2 problem in Section 4.2 and the DTLZ3 problem in Section 4.2.

4.1.1 DTLZ1 Objective Space Movement

For this first case we assume that it is desire to improve much as possible functions f_8, f_9 and f_{10}; values for others functions are not important for us. Thus, we define the direction $d_k \in \mathbb{R}^{10}$ as $d_k = (0, \ldots, 0, -1, -1, -1)^T$, we take as initial optimal solution the vector $x_0 = (0.5, \ldots, 0.5)^T \in \mathbb{R}^{15}$. Computational efforts for these conditions and a step size of 0.05 are shown in Table 4.1.

Table 4.1: Computational efforts for the PE variants for the objective space movement for DTLZ1.

	PE-N	**PE-QN**
Solutions	16.00	17.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	16.00	18.00
Jacobian evaluations	16.00	18.00
Hessian evaluations	160.00	-

We can see in Table 4.1 that PE-N and PE-QN obtained 16 and 17 solutions, respectively. Stopping criterion was the *backtrack in the steps* in both cases. We present results using PE-QN in Table 4.2 and in Figure 4.1.

Table 4.2: Values of the objective functions for the objective space movement in four different steps for DTLZ1 using the PE-QN.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0010	0.0802	0.1521	0.3757
f_2	0.0010	0.0000	0.0000	0.0000
f_3	0.0020	0.0000	0.0000	0.0000
f_4	0.0039	0.0839	0.1552	0.0000
f_5	0.0078	0.0000	0.0000	0.0000
f_6	0.0156	0.0000	0.0000	0.0000
f_7	0.0312	0.0710	0.1439	0.1243
f_8	0.0625	0.0233	0.0000	0.0000
f_9	0.1250	0.0743	0.0000	0.0000
f_{10}	0.2500	0.1673	0.0488	0.0000

We notice in Figure 4.1 a clear reduction for f_8, f_9 and f_{10}, so the procedure was successful. Indeed, we can in Table 4.2 we reach the minimum for all functions, except for f_1 and f_7.

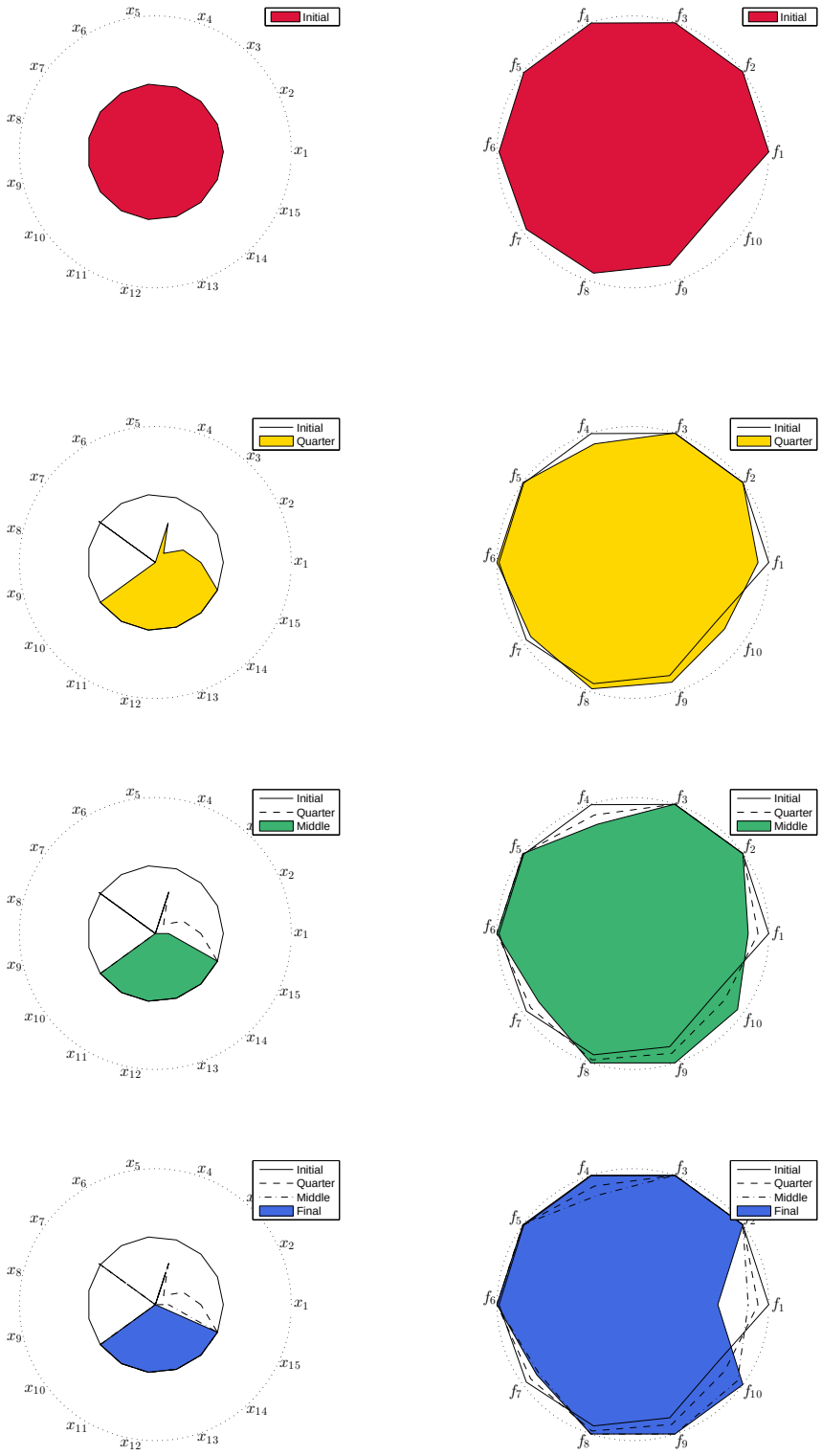


Figure 4.1: Resulting movement in objective space for DTLZ1.

4.1.2 DTLZ1 Decision Space Movement

For this case we assume the direction $d_n \in \mathbb{R}^{15}$ which gets a reduction for the first six decision variables, that is, $d_n = \sum_{i=1}^{6} -e_i$. Our initial optimal solution is the vector $x_0 = (0.5, \ldots, 0.5)^T \in \mathbb{R}^{15}$. Computational efforts for these conditions and a step size of 0.05 are presented in Table 4.3.

Table 4.3: Computational efforts for the PE variants for the decision space movement for DTLZ1.

	PE-N	**PE-QN**
Solutions	5.00	5.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	5.00	5.00
Jacobian evaluations	5.00	5.00
Hessian evaluations	50.00	-

We have for this case a few steps to get the final solution, Table 4.1 indicates five obtained solutions for both versions. We also have *backtrack in the steps* as stopping criterion for both cases. We can see results using PE-N in Table 4.4 and in Figure 4.2.

Table 4.4: Values of the decision variables for the decision space movement in four different steps for DTLZ2 using the PE-N.

	Initial	**Quarter**	**Middle**	**Final**
x_1	0.5000	0.4621	0.3663	0.0000
x_2	0.5000	0.4278	0.2737	0.0000
x_3	0.5000	0.3679	0.1497	0.0000
x_4	0.5000	0.2731	0.0059	0.0000
x_5	0.5000	0.1413	0.0000	0.0000
x_6	0.5000	0.0000	0.0000	0.0000
x_7	0.5000	0.5000	0.5000	0.5000
x_8	0.5000	0.5000	0.5000	0.5000
x_9	0.5000	0.5000	0.5000	0.5000
x_{10}	0.5000	0.5000	0.5000	0.5000
x_{11}	0.5000	0.5000	0.5000	0.5000
x_{12}	0.5000	0.5000	0.5000	0.5000
x_{13}	0.5000	0.5000	0.5000	0.5000
x_{14}	0.5000	0.5000	0.5000	0.5000
x_{15}	0.5000	0.5000	0.5000	0.5000

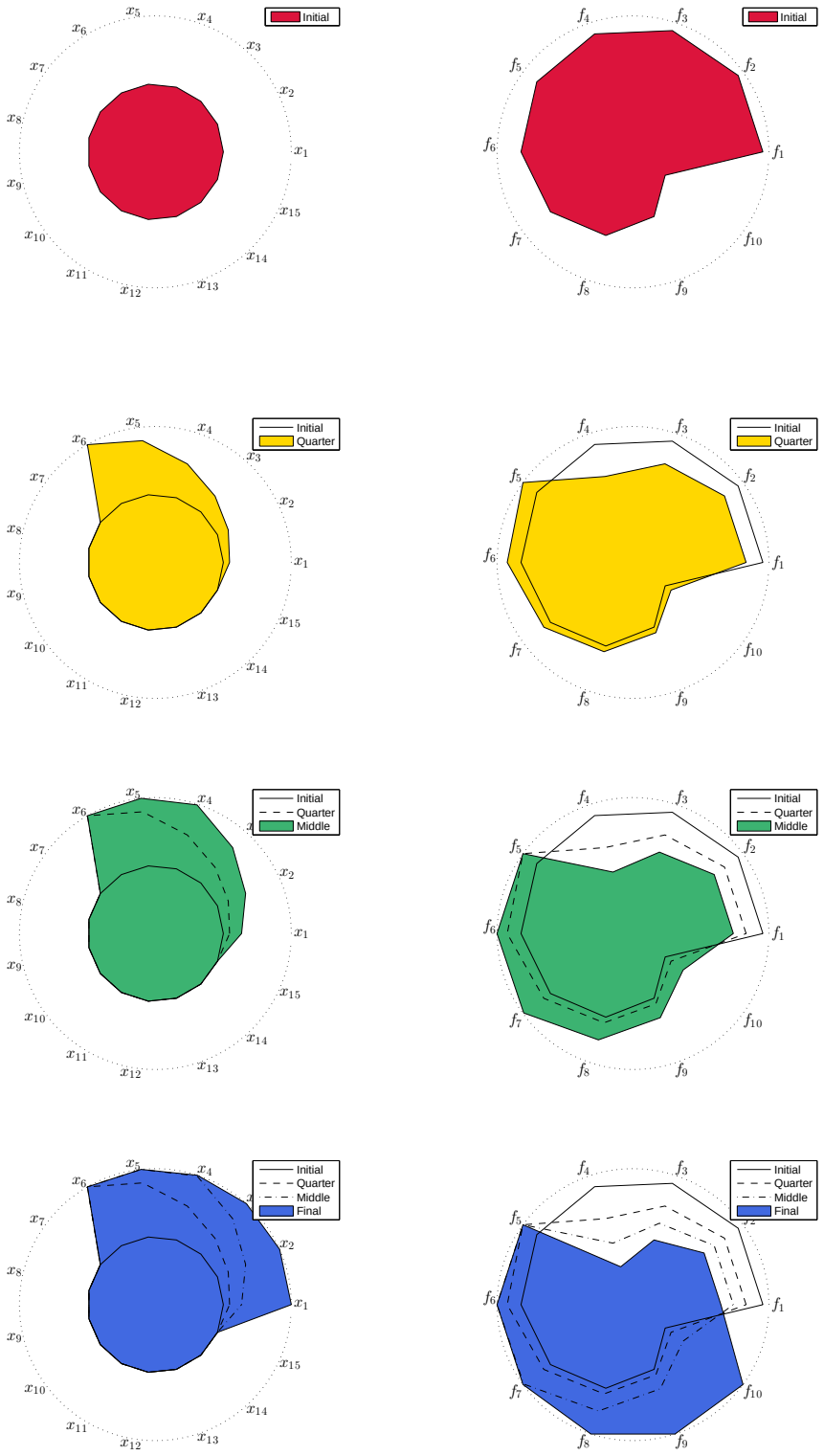


Figure 4.2: Resulting movement in decision space for DTLZ1.

We can observe that PE follows the desirable movement with the radar chart of decision variables in Figure 4.2. We can also see in Figure 4.2 that the movement in decision space produces a reduction on six function in objective space, from f_5 to f_{10}. The values of Table 4.4 confirm that the method reaches the lower bound at the target variables.

4.1.3 DTLZ1 Movement in μ space

For this case we simply chose a vector μ which involves the reduction of f_{10} together with the increment of f_5, that is, $\mu \in \mathbb{R}^{10}$, $\mu_5 = 1, \mu_{10} = -1$, and $\mu_i = \mu_{2i} = 0$, $i = 1, \ldots, 4$. Computational efforts fort this case with a step size of 0.05 are shown in Table 4.5, here we can see that PE-QN needed one more step to complete its solution path. The stopping criteria in this case was *backtrack in the steps*.

Table 4.5: Computational efforts for the PE variants for the μ space movement for DTLZ1.

	PE-N	**PE-QN**
Solutions	17.00	18.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	18.00	19.00
Jacobian evaluations	18.00	19.00
Hessian evaluations	180.00	-

We present the solution path using PE-N in Figure 4.3 and in Table 4.6.

Table 4.6: Values of the objective functions for the μ space movement in four different steps for DTLZ2 using the PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0010	0.0018	0.0040	0.0000
f_2	0.0010	0.0018	0.0040	0.0000
f_3	0.0020	0.0036	0.0080	0.0000
f_4	0.0039	0.0071	0.0161	0.0000
f_5	0.0078	0.0149	0.1073	0.5000
f_6	0.0156	0.0278	0.0410	0.0000
f_7	0.0312	0.0542	0.0559	0.0000
f_8	0.0625	0.1060	0.0889	0.0000
f_9	0.1250	0.2075	0.1747	0.0000
f_{10}	0.2500	0.0754	0.0000	0.0000

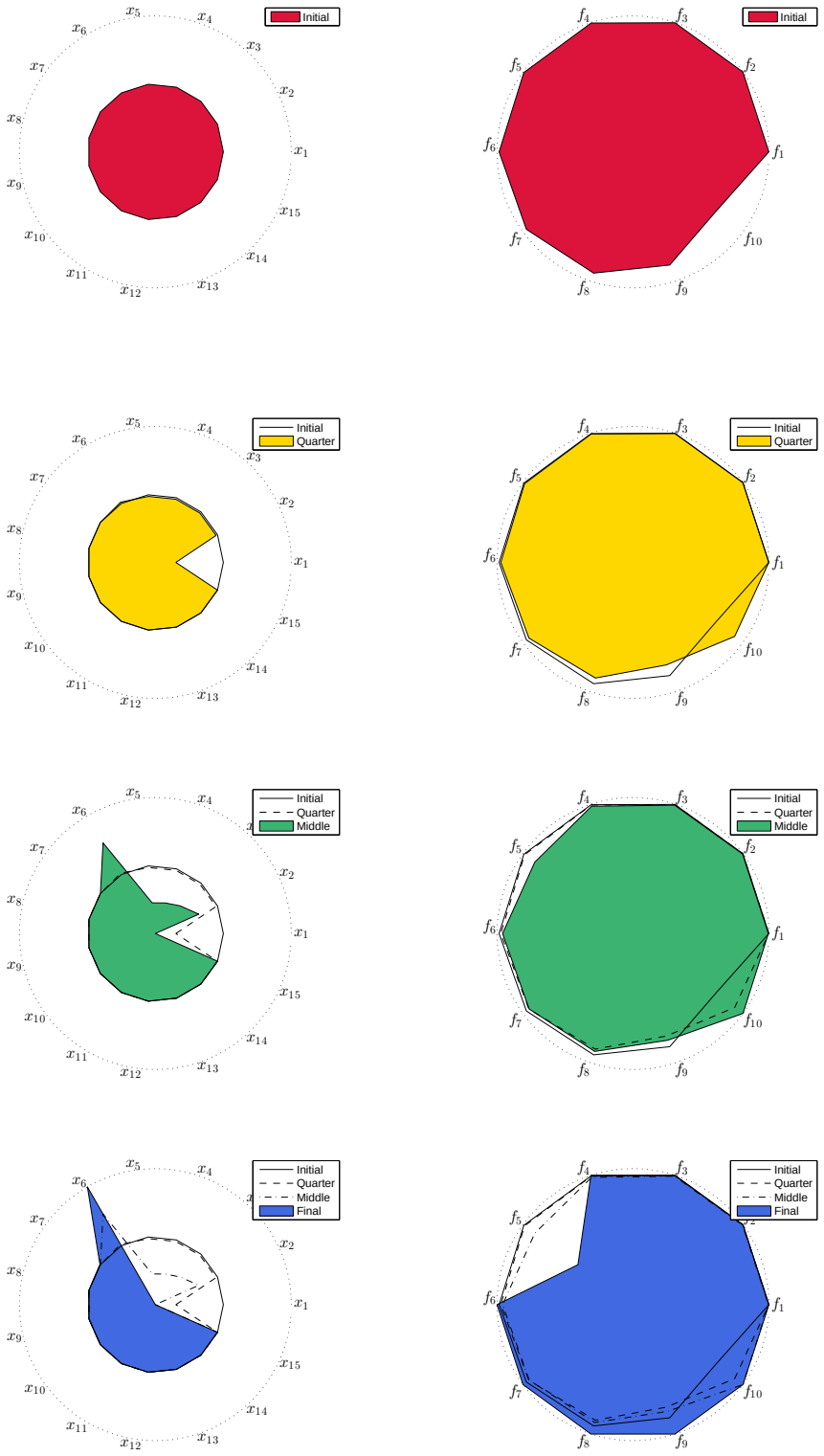


Figure 4.3: Resulting movement in decision space for DTLZ1.

We notice in Figure 4.3 that PE-N follows the desirable path, that is, a reduction of f_{10} together with a increment of f_5. We naturally appreciate a change for the others functions but such changes are more subtle.

Values of Table 4.6 indicate that although PE-N quickly reach the minimum values of f_{10}, it stops until the highest value of f_5 is obtained. Indeed, with this direction we obtain the minimum value for all objective functions except for f_5.

4.1.4 DTLZ1 DDRP

We define a curve $z(t) \in \mathbb{R}^{10}$ as the line segment connecting the points $z^0 \in \mathbb{R}^{10}$, $z_0 = (0, \ldots, 0)^T$ and $z^1 \in \mathbb{R}^{10}$, $z_i^1 = 1,\ i = 1, 3, \ldots, 9$. We divide this segment in 50 equal parts, thus, we have 51 solutions to compute.

The computational efforts for this case are shown in Table 4.7, Here, we notice that there is a considerable increment in the number of functions evaluations with respected to the previous results due to the corrector step. For this case we need at least one corrector by each iteration, that is, we must solve the reference point problem (Equation (2.71)) after each predictor step. According with Table 4.7 the corrector takes in average 33.95 and 31.96 iterations for PE-N and PE-QN, respectively. The above produces the significant increment in the function evaluations (5582 for PE-N and 5453 for PE-QN).

Table 4.7: Computational efforts for the PE variants for DRPP for DTLZ2.

	PE-N	**PE-QN**
Solutions	51.00	51.00
Avg. corrector iterations	33.95	31.96
Avg. backtrack iterations	50.60	50.00
Function evaluations	5582.00	5453.00
Jacobian evaluations	67.00	15.00
Hessian evaluations	670.00	-

We can see the resulting path for this case in Figure 4.4 using PE-N, time depend curve can not plot in our graphical representation. We notice in Figure 4.4 that the method start with similar values for all the functions and in the end PE-N we have a reduction for the even functions.

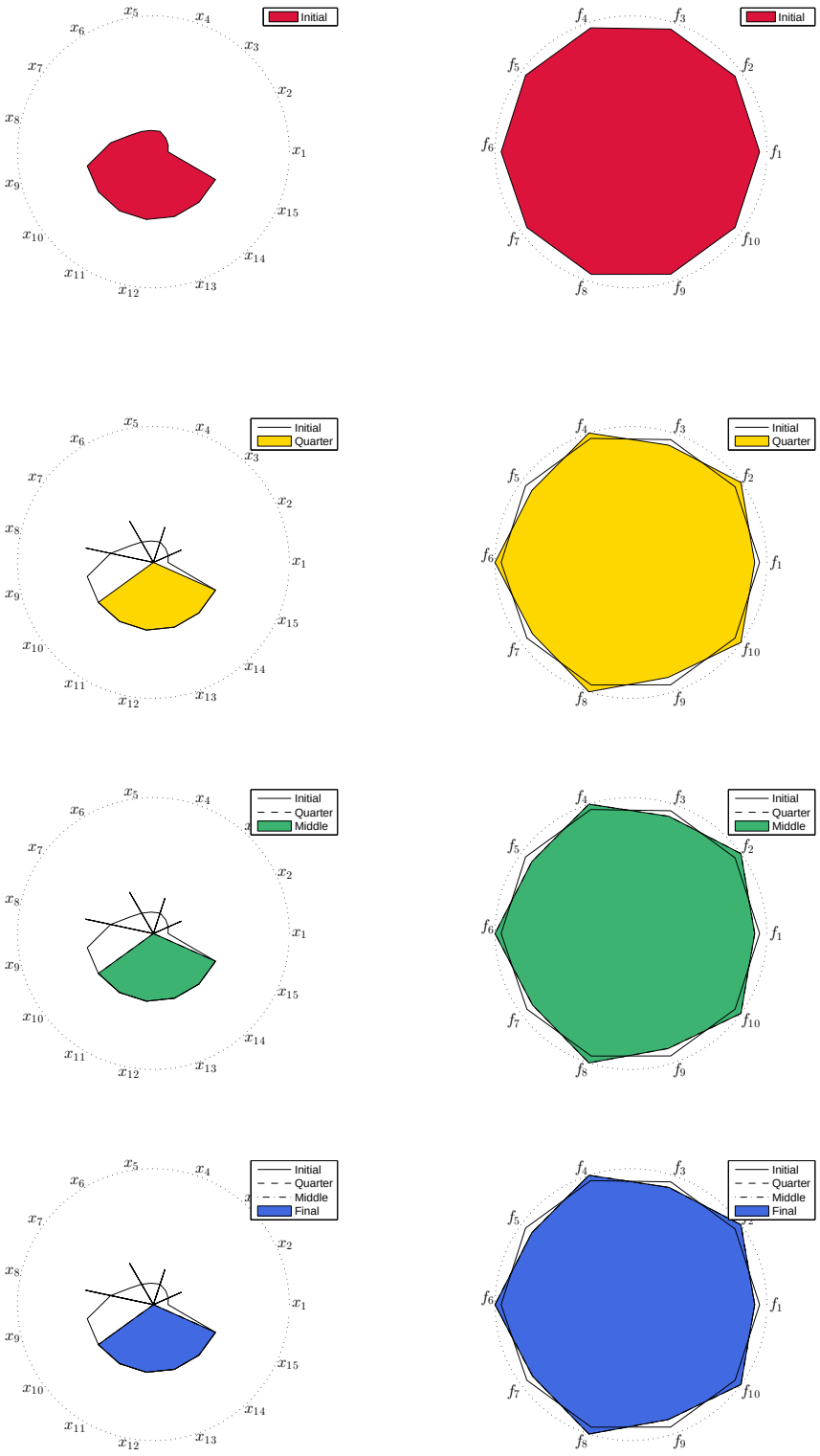


Figure 4.4: Result of DRPP for DTLZ1.

We can see in Table 4.8 the values of the objective functions using PE-N. We notice that a solution is maintained, due to the selected curve can not produce new movements. However, it is unknown until we solve the reference point problem.

Table 4.8: Values of the objective functions for DRPP in four different steps for DTLZ1 using the PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0537	0.0900	0.0900	0.0900
f_2	0.0531	0.0000	0.0000	0.0000
f_1	0.0506	0.0940	0.0940	0.0940
f_1	0.0415	0.0000	0.0000	0.0000
f_1	0.0417	0.0994	0.0994	0.0994
f_1	0.0453	0.0000	0.0000	0.0000
f_1	0.0537	0.1053	0.1053	0.1053
f_1	0.0533	0.0000	0.0000	0.0000
f_1	0.0540	0.1113	0.1113	0.1113
f_1	0.0537	0.0000	0.0000	0.0000

4.2 DTLZ2

The scalable DTLZ2 problem was defined in Section 3.4.1 in Equation (3.5). Along this section we consider the DTLZ2 with 15 decision variables and 10 objective functions. We tested this problem with the same four kinds of steering used in Section 4.1.

4.2.1 DTLZ2 Objective Space Movement

We know that the DTLZ2 problem has a concave PF. Thus, we chose the direction $d_k \in \mathbb{R}^k$ $d_k := (-1, \ldots, -1)^T$ in order to obtain as final result a point on PF which has a similar value at all its components. We take as initial optimal solution the vector $x_0 = (0.5, \ldots, 0.5)^T \in \mathbb{R}^{15}$. Computational efforts for these conditions are shown in Table 4.9.

Table 4.9: Computational efforts for the PE variants for the objective space movement for DTLZ2.

	PE-N	**PE-QN**
Solutions	16.00	16.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	17.00	17.00
Jacobian evaluations	17.00	17.00
Hessian evaluations	170.00	-

Table 4.9 indicates that both versions obtained 16 solutions and spent 17 function evaluations. Also in both cases the method stopped due to a *backtrack in the steps*. We selected PE-N to show the resulting path.

We can see results using PE-N in Table 4.4 and in Figure 4.2.

We notice in Figure 4.1 a regular polygon at the final iteration of the method, that is, the method gets the expected result. While, in Table 4.2 we can see the values of each objective function. We do not have exactly the same values for all functions, however these values are similar with each other.

Table 4.10: Values of the decision variables for the objective space movement in four different steps for DTLZ2 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0442	0.1143	0.1955	0.3233
f_2	0.0442	0.1143	0.1955	0.3233
f_3	0.0625	0.1252	0.2021	0.3228
f_4	0.0884	0.1433	0.2132	0.3221
f_5	0.1250	0.1714	0.2307	0.3210
f_6	0.1768	0.2130	0.2574	0.3192
f_7	0.2500	0.2733	0.2971	0.3164
f_8	0.3536	0.3593	0.3550	0.3121
f_9	0.5000	0.4813	0.4385	0.3055
f_{10}	0.7071	0.6538	0.5578	0.2954

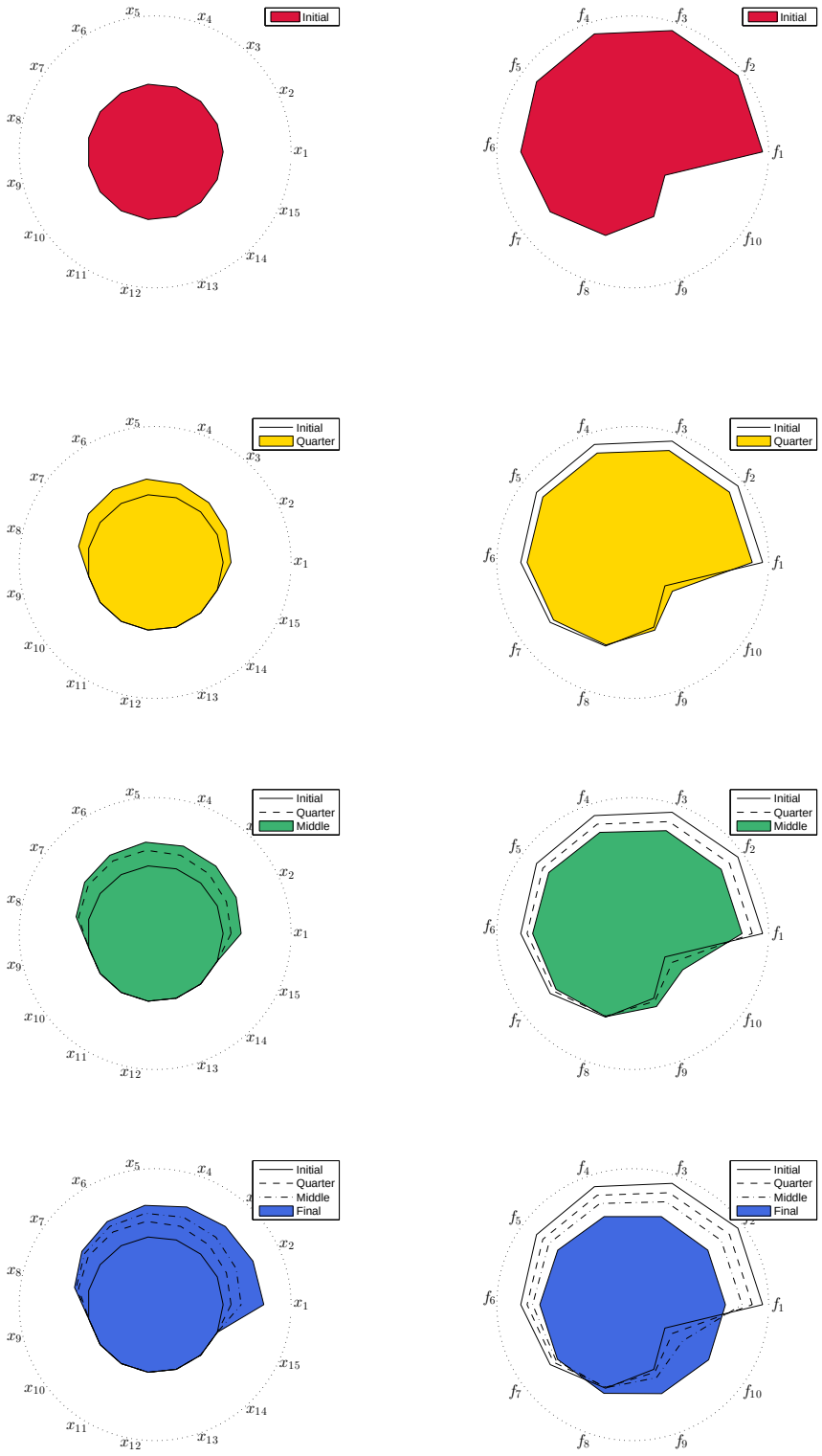


Figure 4.5: Resulting movement in objective space for DTLZ2.

4.2.2 DTLZ2 Decision Space Movement

For this case we take the direction $d_n \in \mathbb{R}^{15}$ as $d_n = -e_5$ in order to reduce x_5. Our initial optimal solution is the vector $x_0 = (0.5, \ldots, 0.5)^T \in \mathbb{R}^{15}$. Computational efforts for these conditions and step size of 0.05 are presented in Table 4.11.

Table 4.11: Computational efforts for the PE variants for the decision space movement for DTLZ2.

	PE-N	**PE-QN**
Solutions	5.00	5.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	5.00	5.00
Jacobian evaluations	5.00	5.00
Hessian evaluations	50.00	-

We have for this case a few steps to get the final solution, Table 4.9 indicates five obtained solutions for both versions. We also have *backtrack in the steps* as stopping criterion for both cases. We can see results using PE-N in Table 4.12 and in Figure 4.6, where is clear that the method follows the direction due to we have a change only for the variable x_5.

Table 4.12: Values of the decision variables for the decision space movement in four different steps for DTLZ2 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
x_1	0.5000	0.5000	0.5000	0.5000
x_2	0.5000	0.5000	0.5000	0.5000
x_3	0.5000	0.5000	0.5000	0.5000
x_4	0.5000	0.5000	0.5000	0.5000
x_5	0.5000	0.3727	0.2454	0.0000
x_6	0.5000	0.5000	0.5000	0.5000
x_7	0.5000	0.5000	0.5000	0.5000
x_8	0.5000	0.5000	0.5000	0.5000
x_9	0.5000	0.5000	0.5000	0.5000
x_{10}	0.5000	0.5000	0.5000	0.5000
x_{11}	0.5000	0.5000	0.5000	0.5000
x_{12}	0.5000	0.5000	0.5000	0.5000
x_{13}	0.5000	0.5000	0.5000	0.5000
x_{14}	0.5000	0.5000	0.5000	0.5000
x_{15}	0.5000	0.5000	0.5000	0.5000

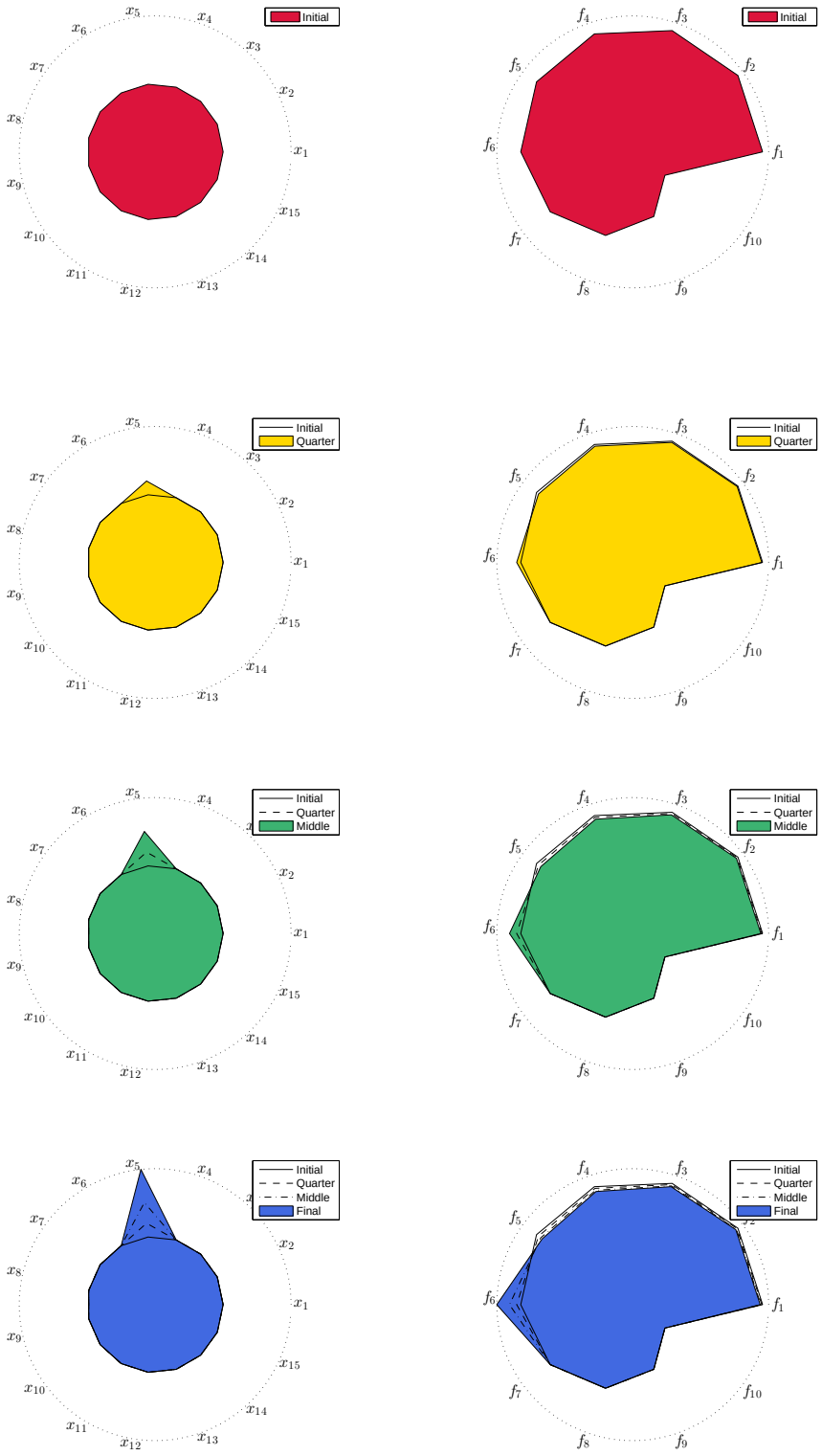


Figure 4.6: Resulting movement in decision space for DTLZ2.

4.2.3 DTLZ2 Movement in μ space

For this case we chose a vector μ which involves the increment of f_9 together with the reduction in the same amount for the others functions, that is, $\mu \in \mathbb{R}^{10}$, $\mu_9 = 1$ and $\mu_i = -1/9$, $i = 1, \ldots, 4$. Computational efforts fort this case with a step size of 0.05 are shown in Table 4.13, here we can see that PE-QN needed one more step to complete its solution path. The stopping criteria in this case is *backtrack in the steps.*

Table 4.13: Computational efforts for the PE variants for the μ space movement for DTLZ2.

	PE-N	**PE-QN**
Solutions	22.00	23.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	23.00	24.00
Jacobian evaluations	23.00	24.00
Hessian evaluations	230.00	-

We notice in Figure 4.7 that PE-N follows the desirable path and we confirm this with the values of Table 4.14. We do not obtain the maximum for f_9 but the considered steps satisfies a considerable decrement for f_9 and a little reduction for the rest of the functions.

Table 4.14: Values of the objective functions for the μ space movement in four different steps for DTLZ2 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0442	0.0394	0.0278	0.0000
f_2	0.0442	0.0394	0.0278	0.0000
f_3	0.0625	0.0558	0.0394	0.0000
f_4	0.0884	0.0791	0.0559	0.0000
f_5	0.1250	0.1123	0.0797	0.0000
f_6	0.1768	0.1598	0.1140	0.0000
f_7	0.2500	0.2283	0.1641	0.0000
f_8	0.3536	0.3278	0.2387	0.0000
f_9	0.5000	0.6610	0.8534	0.9999
f_{10}	0.7071	0.5941	0.4027	0.0163

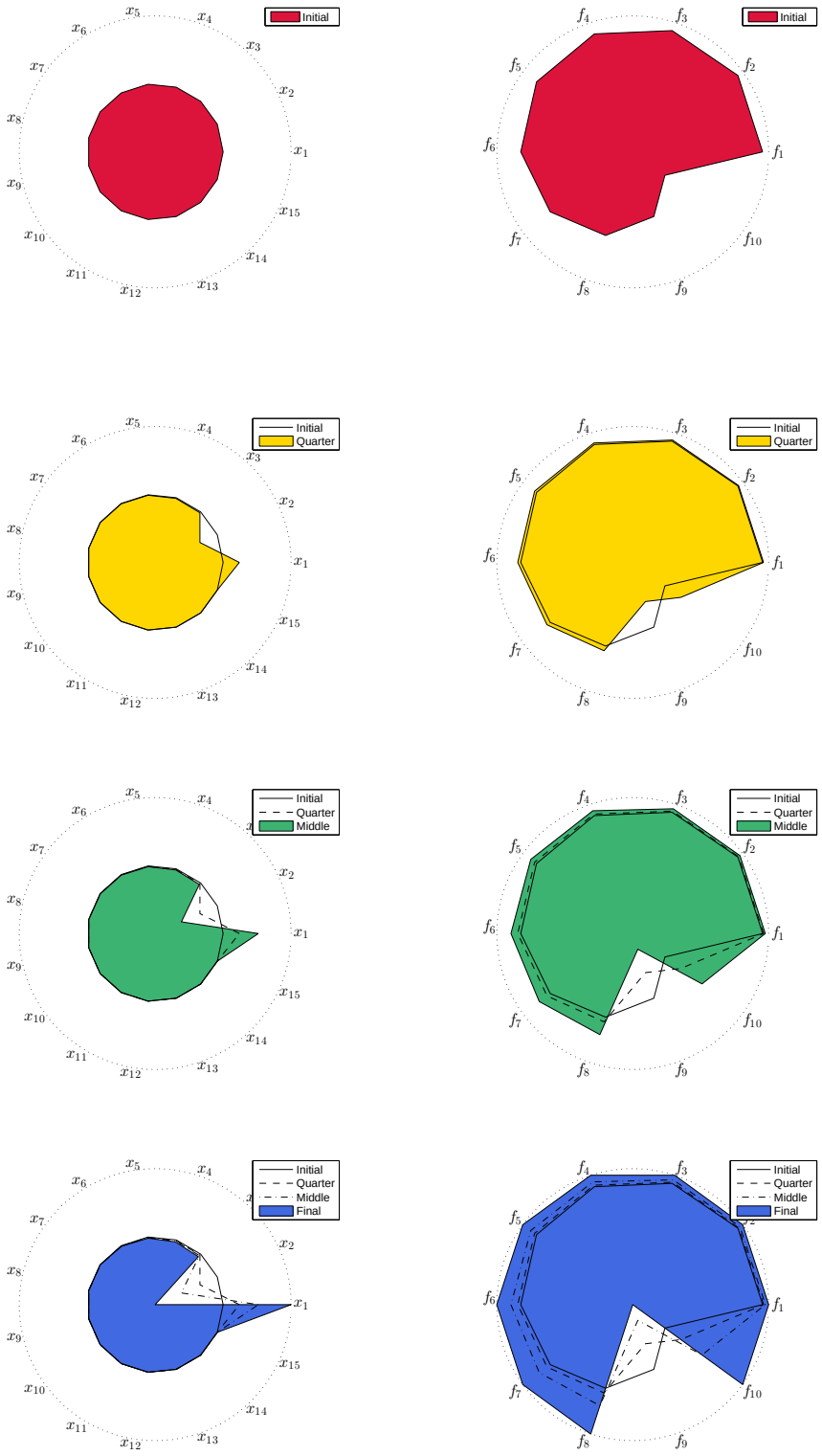


Figure 4.7: Resulting movement in decision space for DTLZ2.

4.2.4 DTLZ2 DDRP

We again consider 51 points on a curve $z(t) \in \mathbb{R}^{10}$ defined as the line segment connecting the points $z^0 \in \mathbb{R}^{10}$, $z_0 = (0, \ldots, 0)^T$ and $z^1 \in \mathbb{R}^{10}$, $z_i^1 = 1,\ i = 1, 3, \ldots, 9$. The computational efforts for this case are shown in Table 4.15.

We also have a significant increment of the functions evaluations with respected to the others movements due to the corrector step, in average 33.95 and 31.96 iterations for PE-N and PE-QN, respectively. As PE-N spends less function evaluations than PE-QN, we show the PE-N resulting path in Figure 4.8 and in Table 4.16.

Table 4.15: Computational efforts for the PE variants for DRPP for DTLZ2.

	PE-N	**PE-QN**
Solutions	51.00	51.00
Avg. corrector iterations	33.95	31.96
Avg. backtrack iterations	0.00	0.00
Function evaluations	16782.00	17987.00
Jacobian evaluations	24.00	53.00
Hessian evaluations	240.00	-

We notice in Figure 4.8 that the method start with similar values for all the functions and at the end it reaches a value according with z^1.

Table 4.16: Values of the objective functions for DRPP in four different steps for DTLZ2 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.3158	0.4126	0.4055	0.3791
f_2	0.3163	0.1726	0.0000	0.0000
f_3	0.3164	0.4126	0.4767	0.4688
f_4	0.3164	0.1726	0.0000	0.0000
f_5	0.3164	0.4126	0.3872	0.3451
f_6	0.3163	0.1726	0.0000	0.0000
f_7	0.3164	0.4126	0.4774	0.4970
f_8	0.3164	0.1726	0.0000	0.0000
f_9	0.3154	0.4126	0.4800	0.5200
f_{10}	0.3165	0.1726	0.0000	0.0000

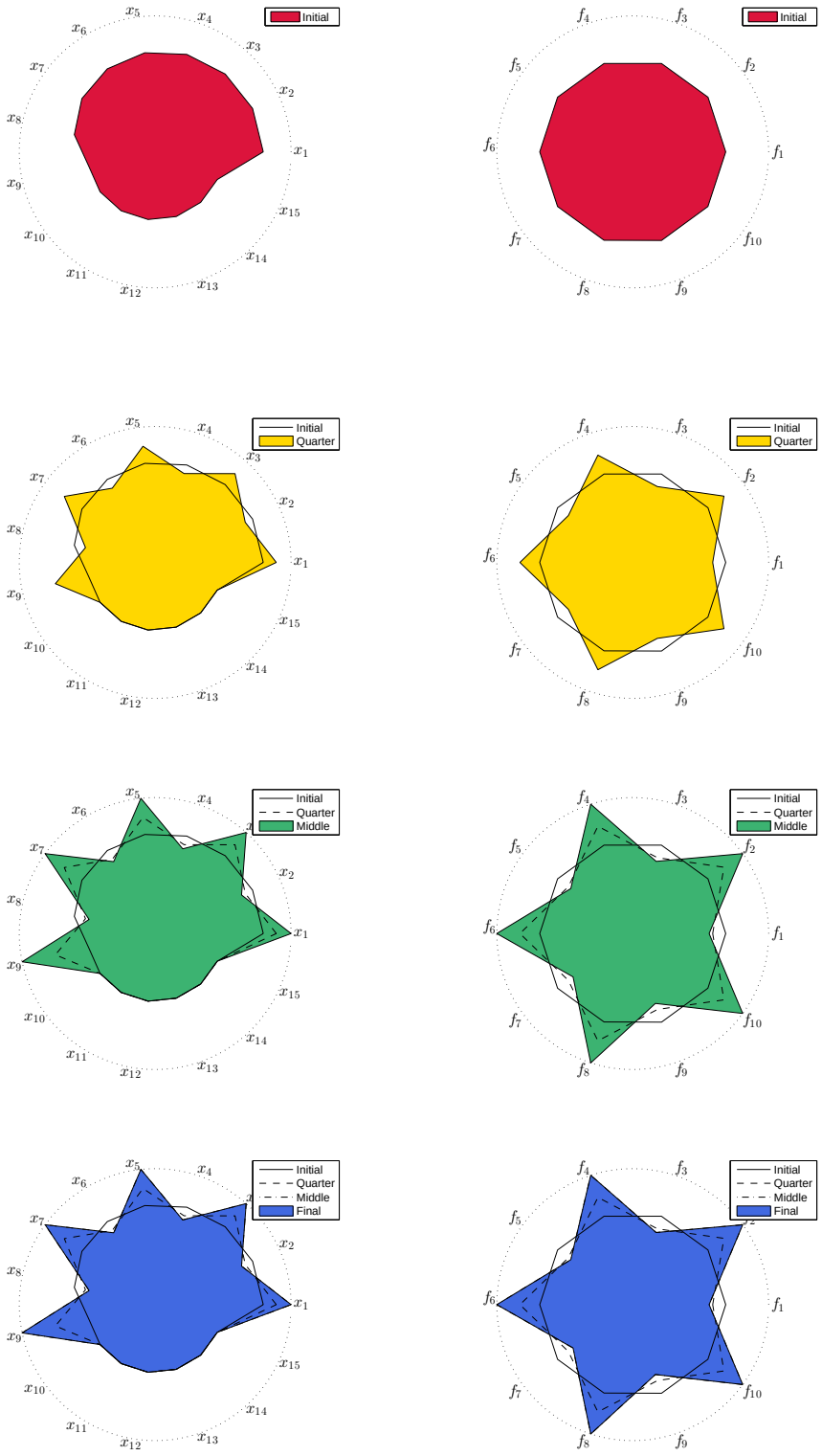


Figure 4.8: Result of DRPP for DTLZ2.

4.3 DTLZ3

The DTLZ3 problem was defined in Section 3.4.2 in Equation (3.11). Along this section we consider the DTLZ3 with 15 decision variables and 10 objective functions. We tested this problem with the same four kinds of steering used in Section 4.1.

4.3.1 DTLZ3 Objective Space Movement

We take as initial optimal solution the vector $x_0 = (0.5, \ldots, 0.5)^T \in \mathbb{R}^{15}$ and the direction $d_k \in \mathbb{R}^{15}$ $d_k := e_{10}$. That is, we want to increase the values of f_{10}. The computational efforts for this case with a step size of 0.05 are shown in Figure 4.17, we can see similar values for PE-N PE-QN. Stop condition was *backtrack in the steps* in both versions.

Table 4.17: Computational efforts for the PE variants for the objective space movement for DTLZ3.

	PE-N	**PE-QN**
Solutions	24.00	24.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	24.00	24.00
Jacobian evaluations	24.00	24.00
Hessian evaluations	240.00	-

We present in Figure 4.9 and in Table 4.18 the resulting path steps for PE-N. PE-N gets the maximum value for f_10 and the minimum for the others.

Table 4.18: Values of the decision variables for the objective space movement in four different steps for DTLZ3 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0442	0.0042	0.0000	0.0000
f_2	0.0442	0.0042	0.0000	0.0000
f_3	0.0625	0.0037	0.0000	0.0000
f_4	0.0884	0.0068	0.0000	0.0000
f_5	0.1250	0.0185	0.0000	0.0000
f_6	0.1768	0.0771	0.0000	0.0000
f_7	0.2500	0.1721	0.0230	0.0000
f_8	0.3536	0.3099	0.1589	0.0000
f_9	0.5000	0.5056	0.4529	0.0000
f_{10}	0.7071	0.7825	0.8770	1.0000

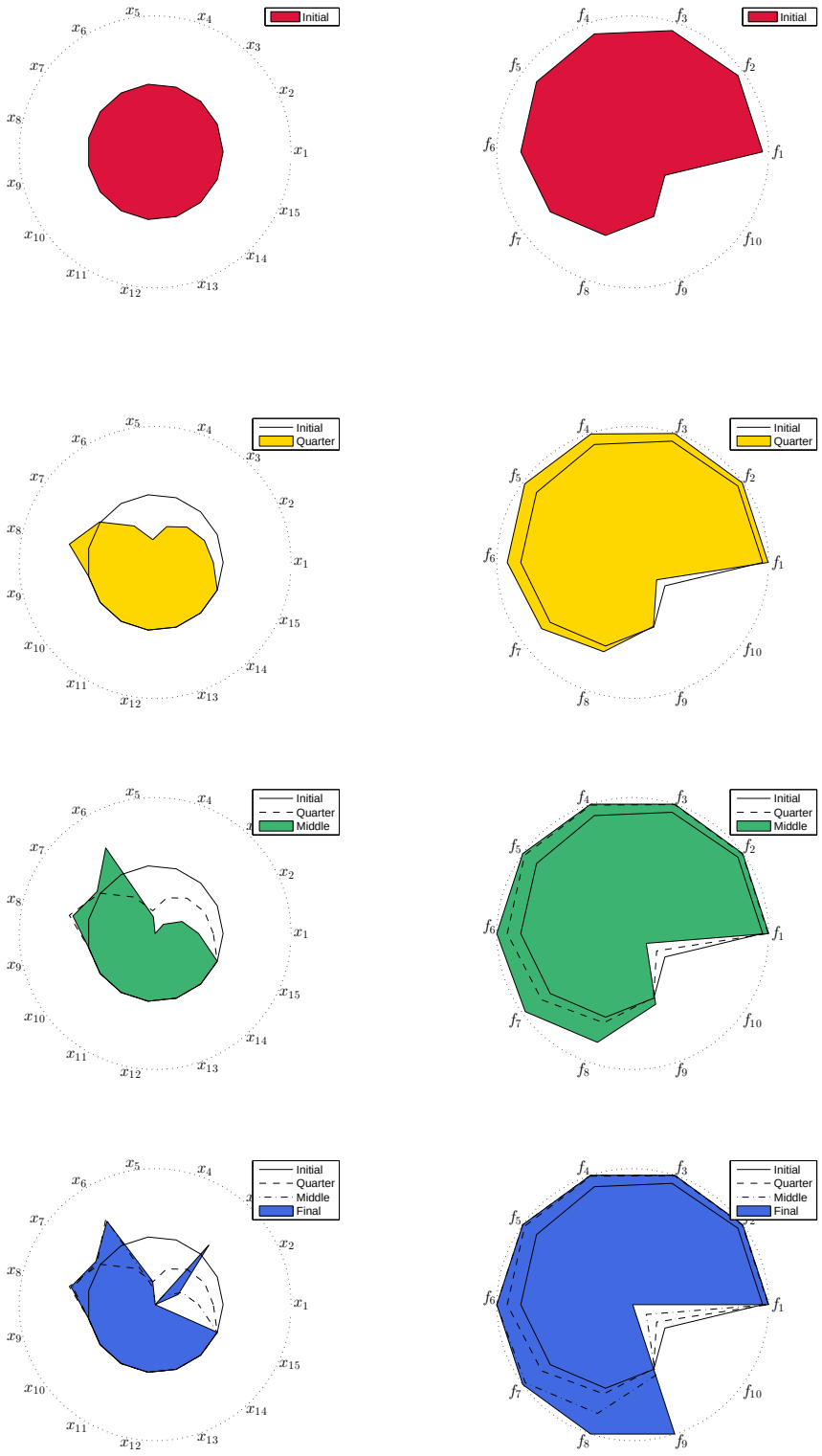


Figure 4.9: Resulting movement in objective space for DTLZ3.

4.3.2 DTLZ3 Decision Space Movement

Now we take the direction $d_n \in \mathbb{R}^{15}$, $d_{ni} = -1$, $i = 1, 3, \ldots, 9$ and $d_{ni} = 0$ otherwise. Our initial optimal solution is the vector $x_0 = (0.5, \ldots, 0.5)^T \in \mathbb{R}^{15}$. Computational efforts for these conditions and a step size of 0.05 are presented in Table 4.19, we notice in this table a similar performance for PE-N and PE-QN. The stopping criterion for both versions is *backtrack in steps*.

Table 4.19: Computational efforts for the PE variants for the decision space movement for DTLZ3.

	PE-N	**PE-QN**
Solutions	27.00	27.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	27.00	27.00
Jacobian evaluations	27.00	27.00
Hessian evaluations	270.00	-

We can see results using PE-N in Table 4.20 and in Figure 4.10. We appreciate that the method follows the given direction due to the indicated decision variables get their minimum value.

Table 4.20: DValues of the decision variables for the decision space movement in four different steps for DTLZ2 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
x_1	0.5000	0.4810	0.3958	0.0000
x_2	0.5000	0.5000	0.5000	0.5000
x_3	0.5000	0.4256	0.1336	0.0000
x_4	0.5000	0.5000	0.5000	0.5000
x_5	0.5000	0.2256	0.0000	0.0000
x_6	0.5000	0.5000	0.5000	0.5000
x_7	0.5000	0.0000	0.0000	0.0000
x_8	0.5000	0.5000	0.5000	0.5000
x_9	0.5000	0.0000	0.0000	0.0000
x_{10}	0.5000	0.5000	0.5000	0.5000
x_{11}	0.5000	0.5000	0.5000	0.5000
x_{12}	0.5000	0.5000	0.5000	0.5000
x_{13}	0.5000	0.5000	0.5000	0.5000
x_{14}	0.5000	0.5000	0.5000	0.5000
x_{15}	0.5000	0.5000	0.5000	0.5000

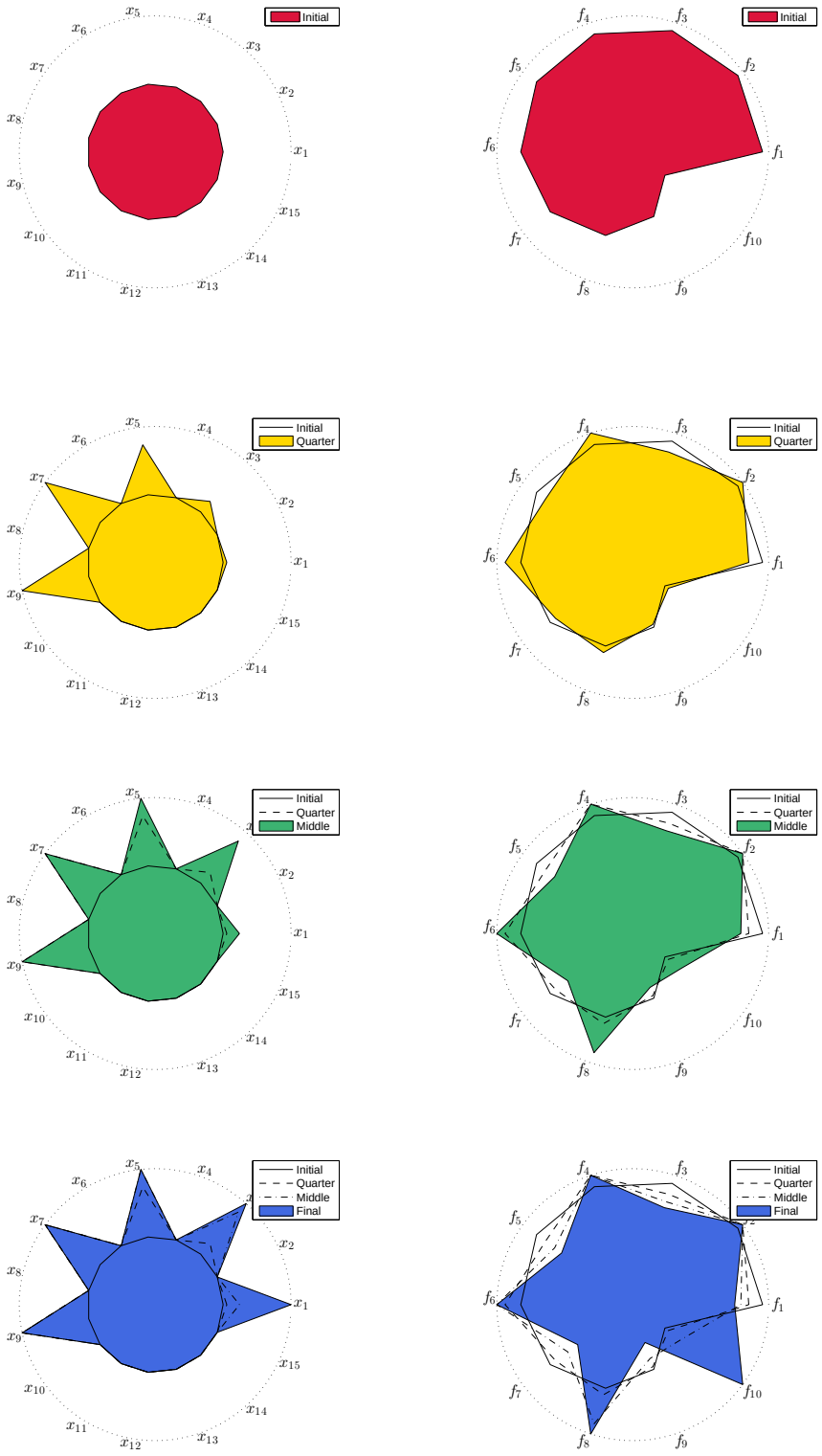


Figure 4.10: Resulting movement in decision space for DTLZ3.

4.3.3 DTLZ3 Movement in μ space

Here we chose a vector μ which involves the decrement of f_5 together with the increment in the same amount for the others functions, that is, $\mu \in \mathbb{R}^{10}$, $\mu_5 = -1$ and $\mu_i = 1/9$, $i = 1, \ldots, 4$. Computational efforts fort this case with a step size of 0.05 are shown in Table 4.21, here we can see that PE-QN needs more steps to complete its solution path. The stopping criteria in this case is *backtrack in the steps*.

Table 4.21: Computational efforts for the PE variants for the μ space movement for DTLZ2.

	PE-N	**PE-QN**
Solutions	3.00	7.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	4.00	8.00
Jacobian evaluations	4.00	8.00
Hessian evaluations	40.00	-

We can see in Figure 4.11 and in Table 4.22 the resulting path using PE-QN. We notice that the expected direction is not exactly satisfied, there are functions besides of f_5 which decrement their value. Thus, we have an approximate movement which depends of each version. It explains the different obtained results in Table 4.21.

Table 4.22: Values of the objective functions for the μ space movement in four different steps for DTLZ3 using PE-Q.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.0442	0.0378	0.0234	0.0159
f_2	0.0442	0.0378	0.0234	0.0159
f_3	0.0625	0.0533	0.0328	0.0222
f_4	0.0884	0.0749	0.0454	0.0304
f_5	0.1250	0.0901	0.0367	0.0171
f_6	0.1768	0.1584	0.1142	0.0894
f_7	0.2500	0.2369	0.1991	0.1751
f_8	0.3536	0.3499	0.3320	0.3186
f_9	0.5000	0.5081	0.5224	0.5361
f_{10}	0.7071	0.7203	0.7474	0.7552

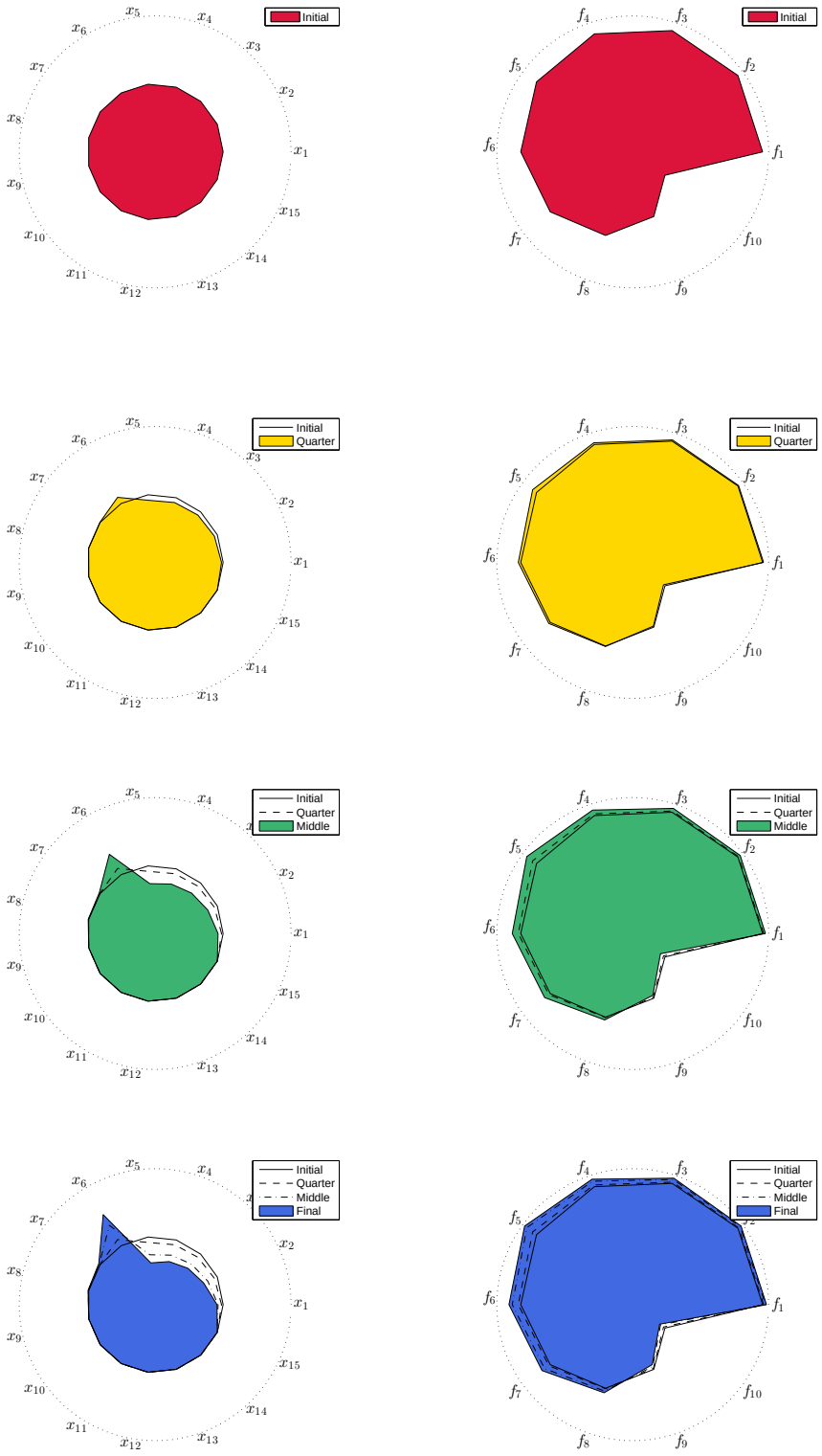


Figure 4.11: Resulting movement in decision space for DTLZ3.

4.3.4 DTLZ3 DDRP

We again consider 51 points on a curve $z(t) \in \mathbb{R}^{10}$ defined as the line segment connecting the points $z^0 \in \mathbb{R}^{10}$, $z_0 = (0, \ldots, 0)^T$ and $z^1 \in \mathbb{R}^{10}$, $z_i^1 = 1,\ i = 1, 3, \ldots, 9$. The computational efforts for this case are shown in Table 4.23.

We have that PE-N and PE-QN require 47.64 and 31.96 correctors in average, respectively. This produces a significant increment of the functions evaluations with respected to the others movements.

Table 4.23: Computational efforts for the PE variants for DRPP for DTLZ3.

	PE-N	**PE-QN**
Solutions	51.00	51.00
Avg. corrector iterations	47.64	31.96
Avg. backtrack iterations	0.00	0.00
Function evaluations	18628.00	21399.00
Jacobian evaluations	28.00	63.00
Hessian evaluations	280.00	-

We show the PE-N resulting path in Figure 4.8 and in Table 4.16, due to PE-N spends less function evaluations. We notice the expected result, that is similar to the observed in Figure 4.10. This suggest that we can obtain similar results with different kinds of movements.

Table 4.24: Values of the objective functions for DRPP in four different steps for DTLZ3 using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	0.3302	0.4126	0.4382	0.4382
f_2	0.3302	0.1726	0.0988	0.0988
f_3	0.3043	0.4126	0.4384	0.4384
f_4	0.2904	0.1726	0.0986	0.0986
f_5	0.3089	0.4126	0.4383	0.4383
f_6	0.2887	0.1726	0.0986	0.0986
f_7	0.3273	0.4126	0.4388	0.4388
f_8	0.3230	0.1725	0.0982	0.0982
f_9	0.3252	0.4126	0.4386	0.4386
f_{10}	0.3302	0.1726	0.0000	0.0000

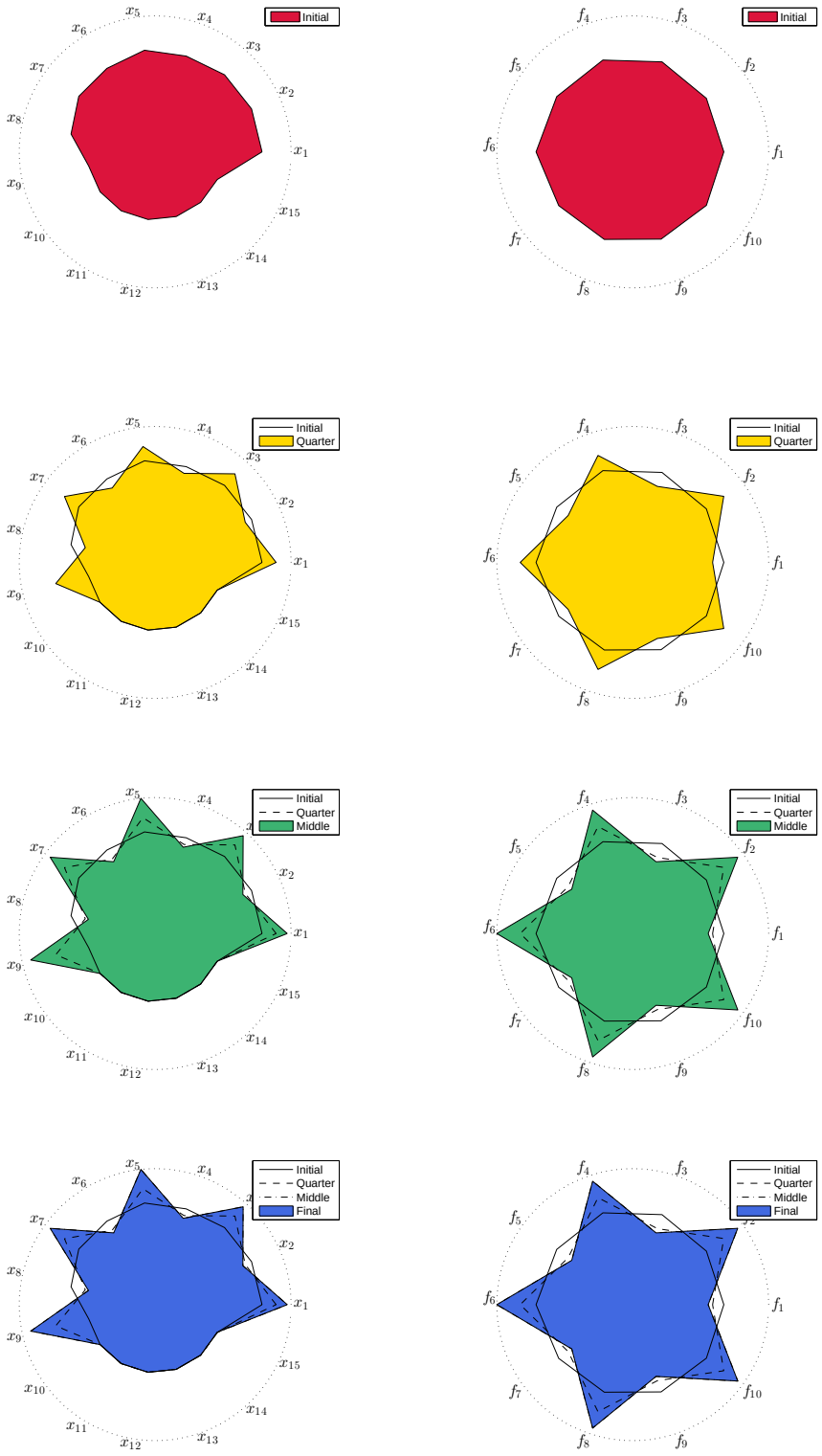


Figure 4.12: Result of DRPP for DTLZ3.

4.4 Real Word Application: Industrial Laundering

In industrial laundering, it is of great importance to achieve high degrees of cleaning for different kinds of contaminations such as oil, egg, curry, etc. for different kinds of fabrics. Additionally, the laundering costs as well as the environmental burden need to be minimized. The laundering process [46] is –among others– influenced by the four parameters:

- x_1: temperature,
- x_2: chemistry (amount of cleaner),
- x_3: time,
- x_4: mechanics (speed of rotation).

In [46] authors developed a model which describes the relation between the factors of and the degree of cleaning for 13 different combinations of fabric and contamination. The model was generated by fitting quadratic ansatz functions to experimental measurements [47]. All parameters are normalized using the reference point $(0, 0, 0, 0)$. The degree of cleaning varies between 0 (no cleaning) and 100 (perfect cleaning). The type of contamination on particular elements considered by objective functions and the cost function are:

- $f_1(x)$: wool grease in cotton,
- $f_2(x)$: wool grease in polyester,
- $f_3(x)$: red in cotton,
- $f_4(x)$: sebum in cotton,
- $f_5(x)$: sebum in polyester,
- $f_6(x)$: currry in cotton,
- $f_7(x)$: motoroel in cotton,
- $f_8(x)$: petroleum in cotton,
- $f_9(x)$: blood in cotton,
- $f_{10}(x)$: egg in cotton,
- $f_{11}(x)$: starch in cotton,
- $f_{12}(x)$: cocoa,
- $f_{13}(x)$: vegetable grease,
- $f_{14}(x)$: cost function.

Thus, the *Laundering Problem* (LP) is given by:

$$\begin{array}{rl} \min\limits_{x\in\mathbb{R}^4} & F(x) = [f_1(x), \ldots, f_{14}(x)], \\ \text{s.t} & -1.5 \leq x_i \leq 1.5,\ i = 1, 2, 4. \\ & 0 \leq x_3 \leq 1.5. \end{array} \tag{4.3}$$

We consider the five different scenarios described in Chapter 3 and the two versions of PE to solve LP. We use an approximation of the Hessians via finite differences for the PE-N.

The first scenario is a *movement in objective space*. For this case the goal is an improvement for the cost function (f_{14}). For the *movement in decision space* we focus on the reduction of the variable which corresponds to time (x_3) and we take the same direction for the case of a *minimal change in objective space*. We define a suitable vector μ for the *movement in μ space*, which represents a bigger progress on f_{10} sacrificing the value for the rest of objective functions in the same amount. Finally, we fixed a curve for the DRPP, such that the method tries to reach the best value for f_{12} and the closest value to the utopian point for the other objective functions.

4.4.1 Movement in Objective Space

For this approach we define the direction in objective space as $d_k = -e_{14}$, i.e. we want to reduce the value of the cost function as much as possible. We take as initial point $x_0 = (1, 1, 1, 1)^T$, which is not an optimal solution.

We can see the computational efforts for each version of the method with a step size of 1 in Table 4.25. Both versions obtained 111 solutions and also in both cases the method stopped due to a *backtrack in the steps*.

We show in Figure 4.13 the initial optimal solution as well as the iterations 27, 55, and 111 obtained by PE-QN. Numerical values of all objective functions in the indicated iterations are presented in Table 4.26.

Table 4.25: Computational efforts for the PE variants for the objective space movement for LP.

	PE-N	**PE-QN**
Solutions	111.00	111.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	112.00	111.00
Jacobian evaluations	112.00	111.00
Hessian evaluations	1568.00	-

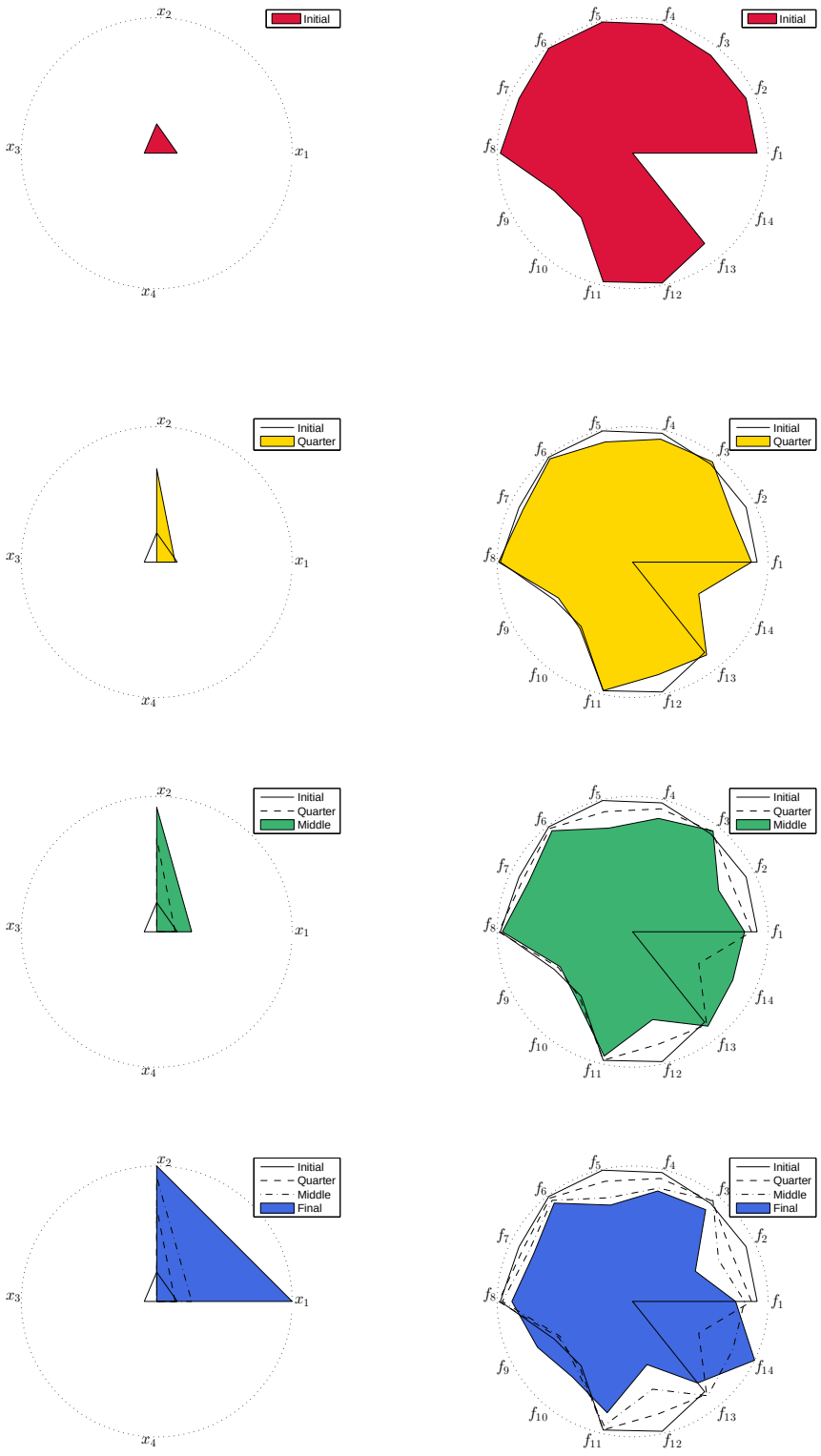


Figure 4.13: Resulting movement in objective space for LP using PE-QN. Reduction of the cost function (f_{14}).

Table 4.26: Values of the objective functions for the objective space movement in four different steps for LP using PE-QN.

	Initial	**Quarter**	**Middle**	**Final**
f_1	-51.7939	-48.4594	-44.1958	-37.1351
f_2	-60.6967	-50.1930	-40.2057	-18.7666
f_3	-64.7622	-66.1990	-66.7195	-60.4359
f_4	-57.0948	-53.8678	-48.3284	-45.0684
f_5	-84.4151	-76.7702	-64.9868	-56.0820
f_6	-71.2207	-69.9971	-68.0270	-64.9441
f_7	-67.3154	-64.1632	-59.7479	-54.3218
f_8	-46.3219	-47.2275	-45.7867	-40.8224
f_9	-65.2682	-61.3069	-59.2949	-83.1116
f_{10}	-32.0226	-33.2730	-35.6052	-41.4375
f_{11}	-45.9450	-46.0133	-44.1074	-36.3988
f_{12}	-71.6009	-60.7151	-44.6800	-22.5289
f_{13}	-40.3162	-41.9827	-43.2332	-33.8284
f_{14}	26.0637	12.9035	6.0533	0.0000

We can see in Figure 4.26 a decrement in the value of the cost function, according with the Table 4.26 the value of f_{14} is reduced from 26.0637 to 0.0000, that is, the method reaches the best possible value. This solution leads to an increment for the majority of the others functions with respect to the initial configuration, only for objectives f_9 and f_{10} we observe a decrement.

Regarding the decision variables, the final solution for this approach implies the reduction of x_1 and x_2 along with the increment of x_3 and x_4 (see Figure 4.13). In other words, we need to reduce the temperature (x_1) and the amount of cleaner (x_2), additionally we need to increase the time (x_3) and the speed of rotation in order to get the minimum possible cost.

4.4.2 Movement in Decision Space

An interesting scenario is to reduce the time of the model without optimality loss. Of course we can provide a configuration with the minimum possible time, but this configuration may correspond to a dominated point. Instead, we perform a steering in decision space in order to know how much we can reduce the time, which is given in the model by x_3. Thus, we defined the direction in decision space as: $d_n = -e_3$ and we take again $x_0 = (1, 1, 1, 1)^T$ as initial point. We show in Figure 4.14 the obtained results for this approach for the PE-QN and present computational efforts for each version of the method with a step size of 0.2 in Table 4.27.

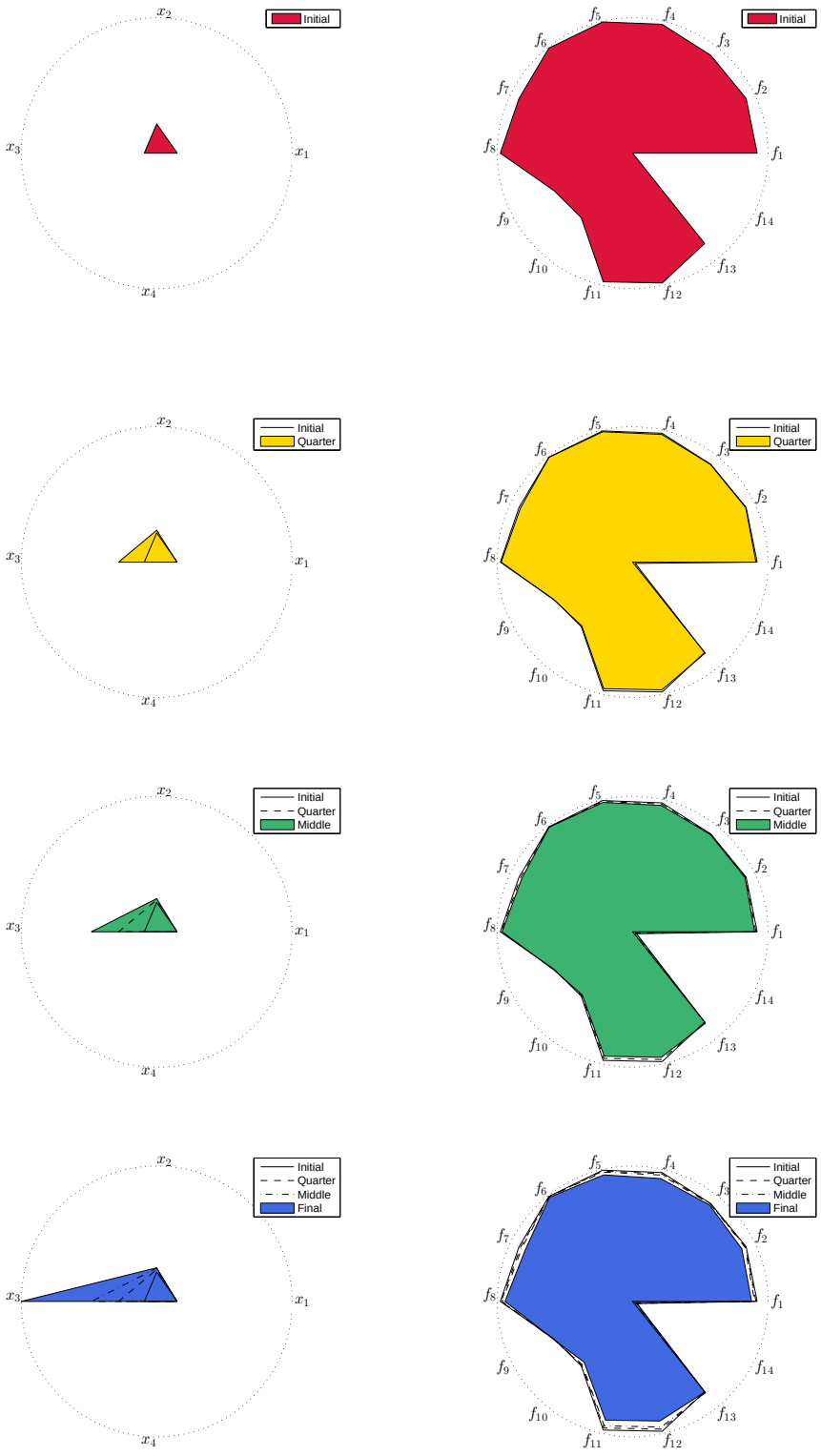


Figure 4.14: Resulting movement in decision space for LP using PE-QN. Reduction of the time variable (x_3).

Table 4.27: Computational efforts for the PE variants for the objective space movement for LP.

	PE-N	**PE-QN**
Solutions	102.00	83.00
Avg. corrector iterations	2.08	0.95
Avg. backtrack iterations	0.03	0.00
Function evaluations	328.00	163.00
Jacobian evaluations	316.00	163.00
Hessian evaluations	4424.00	-

Unlike with the results for benchmark problems, in this case we have a considerable discrepancy on the number of obtained solutions. PE-N obtained 83 solutions while PE-QN obtained 102. The stopping criterion was, for both cases, *backtrack in the steps* but it occurs at different iterations. Thus, PE-N does not reach the minimum time, which seems to be due to the Hessian approximations via finite differences. Therefore, we focus here on the PE-QN results.

We notice in Figure 4.14 the desirable reduction in x_3 and that the rest of the variables do not a significant change. Also, the form of the radar chart for the objective space is very similar at each iteration.

Finally, we present in Table 4.28 the values of the decision variables at iterations 1, 21, 42, and 83 for the PE-QN. The time variable has a reduction from 1.3622 to 0.000, which is a very significant decrement and it is also the minimum possible value for this variable. On the other hand, the value of x_4 is the same at each iteration, while for x_1 and x_2 no a significant changes can be observed.

Table 4.28: Values of the decision variables for the objective space movement in four different steps for LP using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
x_1	1.0429	1.0507	1.0497	1.0504
x_2	0.8521	0.7958	0.7661	0.7549
x_3	1.3622	1.0978	0.7974	0.0000
x_4	1.5000	1.5000	1.5000	1.5000

Similar results obtained for this approach motivate the use of the steer of minimal change in objective space for the same direction $d_n = -e_3$.

4.4.3 Minimal Change in Objective Space

For this approach we take the direction $d_n = -e_3$, but in this case we try to conserve the initial configuration in objective space.

Computational efforts for each version of the method with a step size of 0.2 are shown in Table 4.27. We notice a similar behavior of the PE-N with respect to the movement in objective space, that is, PE-N did less iterations than the PE-QN. The stopping criterion is *backtrack in the steps* for both cases and again PE-N did not reach the best solution because the finite differences.

Table 4.29: Computational efforts for the PE variants for the steer of minimal change in objective space for LP.

	PE-N	**PE-QN**
Solutions	61.00	34.00
Avg. corrector iterations	1.50	0.85
Avg. backtrack iterations	0.03	0.00
Function evaluations	154.00	64.00
Jacobian evaluations	154.00	64.00
Hessian evaluations	2156.00	-

We can see in Figure 4.15 that the form of the radar chart of the objective space is very similar at each iteration and we achieve a reduction for the time variable in decision space at the same time. Table 4.30 presents numerical results obtained by PE-QN at the iterations 1, 15, 30, and 61. Values have similarities with values of Table 4.27, x_4 is the same value at each iteration, x_1 and x_2 have a non significant change, and finally x_3 is reduced but in a lower amount.

Table 4.30: Values of the decision variables for the steer of minimal change in objective space in four different steps for LP using PE-QN.

	Initial	**Quarter**	**Middle**	**Final**
x_1	1.0429	1.0400	1.0349	1.0203
x_2	0.8521	0.8479	0.8470	0.8607
x_3	1.3622	1.1525	0.8698	0.3453
x_4	1.5000	1.5000	1.5000	1.5000

At first we have similar results for this problem approach and the movement in objective space for the same direction. However, the steering with respect to a minimal change in objective space stops when the change in objective space becomes to grow and then it does not reach the minimum value for the time variable.

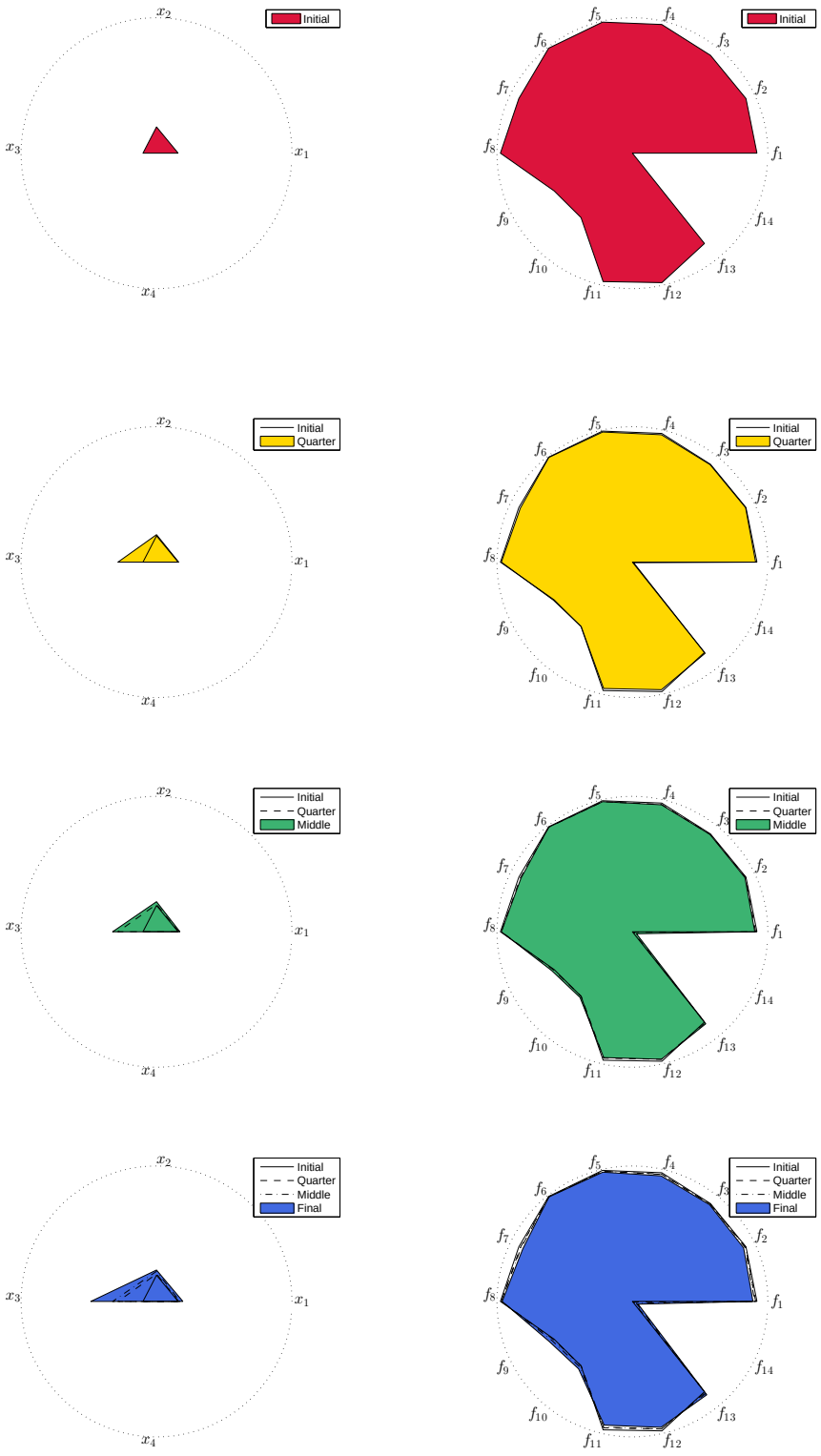


Figure 4.15: Resulting movement of minimal change in objective space for LP.

4.4.4 Movement in μ Space

The μ vector chosen for this example was defined as: $\mu_{10} = -1$ and $\mu_i = 1/13$ otherwise, for $i = 1, \ldots, 14$. The idea is to reduce the value of f_{10} (egg in cotton) sacrificing the value for the rest of objective functions in the same amount.

Computational efforts of the PE variants with a step size of 1 are shown in Table 4.31. PE-QN spends 76 iterations to reach the best value for f_{10}. PE-N only obtains 46 solutions and it does not obtain the best value.

Table 4.31: Computational efforts for the PE variants for the μ space movement for LP.

	PE-N	**PE-QN**
Solutions	46.00	76.00
Avg. corrector iterations	0.00	0.00
Avg. backtrack iterations	0.00	0.00
Function evaluations	47.00	76.00
Jacobian evaluations	47.00	76.00
Hessian evaluations	658.00	-

We show in Table 4.32 the obtained numerical results by PE-QN for iterations 1, 19, 38, and 76. in Figure 4.16 we can see the radar chart for the indicated iterations.

Table 4.32: Values of the decision variables for the μ space movement in four different steps for LP using PE-Q.

	Initial	**Quarter**	**Middle**	**Final**
f_1	-51.7939	-48.3577	-44.6496	-38.0414
f_2	-60.6967	-56.7886	-51.5355	-32.3484
f_3	-64.7622	-63.6687	-62.2982	-58.0583
f_4	-57.0948	-55.8538	-54.4228	-51.0232
f_5	-84.4151	-84.3138	-83.9069	-80.3922
f_6	-71.2207	-70.9236	-70.5037	-68.8194
f_7	-67.3154	-63.8794	-60.2761	-54.8541
f_8	-46.3219	-46.2011	-45.9977	-44.9631
f_9	-65.2682	-77.9235	-91.1811	-111.0770
f_{10}	-32.0226	-41.8877	-51.8563	-63.2592
f_{11}	-45.9450	-44.6885	-43.0823	-37.8752
f_{12}	-71.6009	-71.9252	-71.1836	-59.5752
f_{13}	-40.3162	-44.5083	-48.3741	-49.0434
f_{14}	26.0637	24.4800	22.4533	15.7769

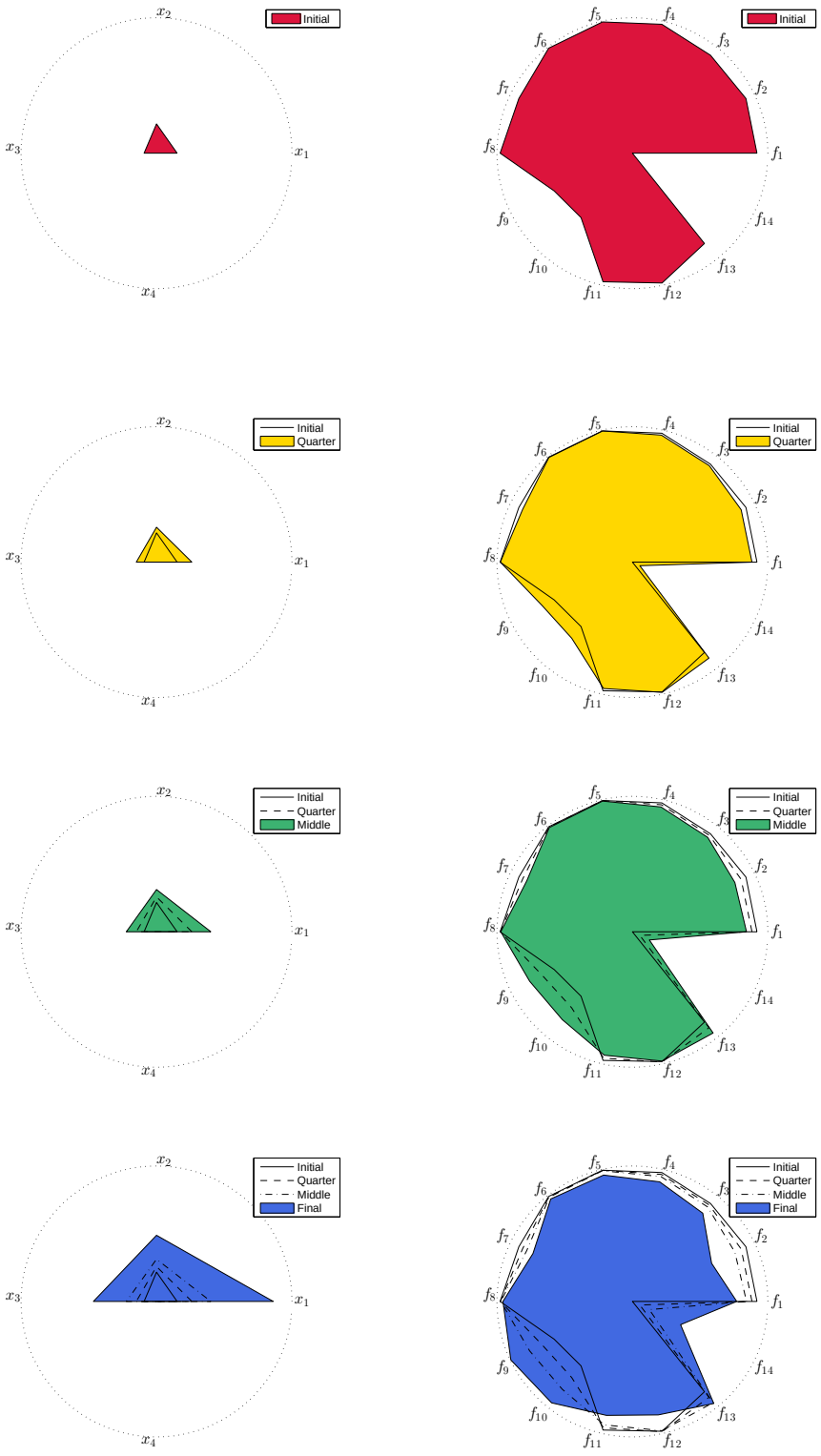


Figure 4.16: Resulting movement in μ space for LP.

4.4.5 Dynamic Reference Point

We define a curve taking 50 intermediate values between the utopian point and the nadir point for each function and changing the value of f_{12} until reach the utopian point value. We expect to get a better value for f_{12} in for each step that the method does. Computational efforts of the PE variants are shown in Table 4.33.

Table 4.33: Computational efforts for the PE variants for the DRPP for LP.

	PE-N	**PE-QN**
Solutions	50.00	50.00
Avg. corrector iterations	23.10	23.36
Avg. backtrack iterations	0.00	0.00
Function evaluations	99.00	99.00
Jacobian evaluations	50.00	99.00
Hessian evaluations	700.00	-

We can see the numerical results given by the PE-N in Figure 4.17 and in Table 4.34. We show the solution for the initial point and the iterations $13th$, $25th$ and 50. Our goal to improve f_{12} is done but the final configuration is not well distributed as in the above case.

Table 4.34: Values of the objective functions for the objective space movement in four different steps for LP using PE-N.

	Initial	**Quarter**	**Middle**	**Final**
f_1	-46.1828	-46.6207	-47.1140	-48.4010
f_2	-47.6121	-49.9058	-52.2351	-57.2613
f_3	-66.7666	-66.3993	-66.0581	-65.4715
f_4	-61.1284	-62.6389	-64.1311	-67.1606
f_5	-58.7567	-62.2110	-65.5399	-71.9140
f_6	-78.5294	-79.1683	-79.7865	-80.9844
f_7	-53.7768	-54.4855	-55.2614	-57.1881
f_8	-35.4564	-36.0470	-36.5950	-37.5522
f_9	-61.8950	-65.1489	-68.0795	-72.6893
f_{10}	-67.5941	-69.8967	-71.9398	-75.0143
f_{11}	-51.7789	-52.3751	-52.9650	-54.1711
f_{12}	-61.4944	-66.1612	-70.7075	-79.6468
f_{13}	-45.6760	-46.6680	-47.5503	-48.8988
f_{14}	-11.7027	-9.9782	-8.1332	-3.7573

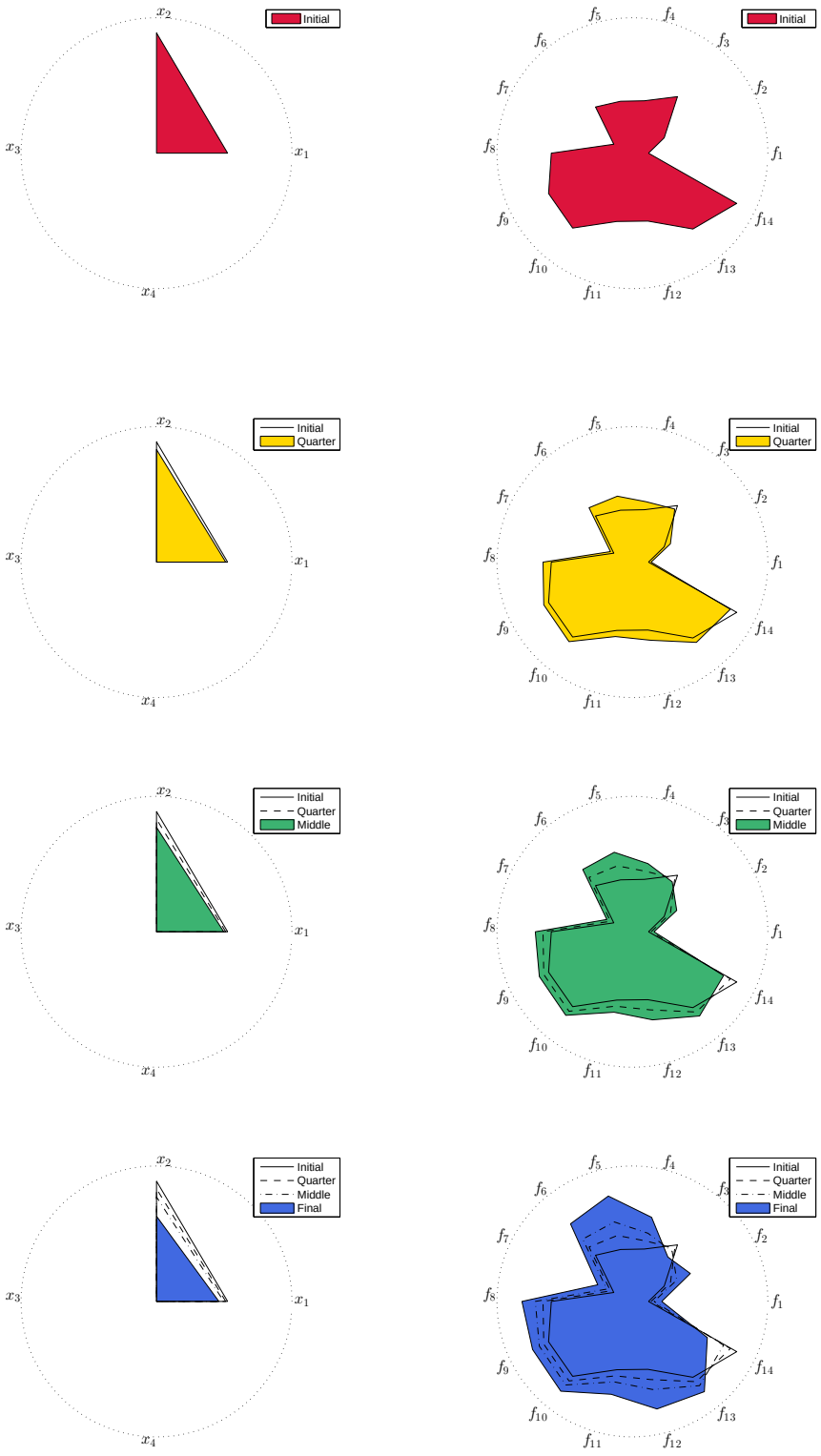


Figure 4.17: Result of DRPP for LP.

Chapter 5

Conclusions and Future Work

In this chapter, we first summarize the thesis work in Section 5.1. After that we discuss our findings and contributions, and point out its limitations in Section 5.2. Finally, in Section 5.3, we outline directions for future research.

5.1 Obtained Results

In this section we briefly present the principal contributions of this work. As described in Chapter 3, *Pareto Explorer* (PE) is a global/local tool for the treatment of *Many Objective Optimization Problem*s (MaOPs). On this framework, we developed a continuation method for the local steering phase of the PE, which is capable to follow the Pareto landscapes (*Pareto Set* (PS) and *Pareto Front* (PF)) in a given direction.

The first kind of movement is the steering in objective space, presented in Section 3.4.1. For this case we have a given direction in objective space and an initial optimal solution. PE method allows to obtain a path of optimal solutions which represents the best possible movement according to the given direction. We also provide stopping criteria that avoid to continue with the search in case of non promising steps.

The second approach is the steering in decision space, described in Section 3.4.2. Here, we also have an initial optimal solution, but the given direction is in decision space. We achieve for this scenario, with the PE method, a continuation focuses along the PS instead of the PF, that is, our interest is in the resulting path on the decision space. We define different stopping criteria for this case considering the non promising steps in decision space.

The third movement is made in the μ space (or weight space) which considers the trade-off between optimal solutions, we explain more details about the μ space in Section 3.4.3. Basically we have an initial optimal solution and a direction in objective space given by a vector whose component sum is equal to zero (a vector μ). Although the movement is performed in objective space, the continuation according to a given vector μ implies a procedure simpler than the requires for the steering in objective space. Indeed, we consider the same stopping criteria used for the steering in objective space. The movement along μ space it is important due to its interpretation for the *Decision Maker* (DM).

A variant of the steering in the decision space is the fourth kind of movement, presented in Section 3.4.4, where we have the steering of minimal change in objective space. Here, PE makes the steering in decision space but the resulting path produces the minimal change in objective space. That is, we start with an initial optimal solution and a given direction in decision space, this direction is a guide. The idea is to get the closest vector to the given direction in objective space such that the change in objective space is minimal.

Finally, we use the predictor of the steering in objective space together with a change on the corrector step to solve the *Dynamic Reference Point Problem* (DRPP). PE returns optimal solution even if the reference points are inside the Pareto surface (see Section 3.4.5).

The five kinds to explore the Pareto landscapes provide to the DM a complete tool to evaluate different options and getting information for a given problem. This is the highlight of this work.

5.2 Conclusions

Here we discuss, based on the examples of Section 3.4 and the numerical results of Chapter 4, the impact of this work.

We developed, as our contribution to the steering phase of the PE, five different ways to steer a search of Pareto landscapes. We provided suitable stopping criteria for the steering in objective and decision spaces. These condition can be used for the steering in μ space and the minimal change in objective space, respectively. As we can see, with the examples of Section 3.4, our stopping criteria prevent the continuation along non promising surfaces and also they avoid the swing of the method.

We appreciate with the tables in Chapter 4 that the PE method generally does not require a lot of function evaluations to get a resulting path. Of course it depends of the number of steps that the method produces, but taking into account that we are considering MaOPs with more than 9 functions, the computational cost is reasonable.

The exception is the treatment of the DRPP which presents a considerable increment of the function evaluations due to the corrector step. Though the method stays in the same point, the method needs to compute at least one iteration for the corrector at each iteration (see Section 4.1.4). In addition, solving the reference point problem implies to find the minimum of a non differentiable function (Equation (2.71)) which naturally has an negative impact whit the efficiency of this approach.

The two versions of PE considered for this work, *Newton Pareto Explorer* (PE-N) and *Quasi-Newton Pareto Explorer* (PE-QN), present a similar performance for the majority of the examples when we used the analytical expression for the Hessians and Jacobians. This means a good performance of the *Broyden-Fletcher-Goldfarb-Shanno* (BFGS) method used for the PE-QN. On the other hand, for the cases when we used finite differences to approximate both Hessians and Jacobians, the PE-QN presents the bests results (see the real word application in Section 4.4). This confirms that the BFGS has a good performance and this also suggest the use of the PE-QN for problems where we have not the second order information.

Finally, we can conclude that PE satisfies its principal purpose, to provide a tool which allows to the DM to explore the Pareto landscapes in real time. The *Laundering Problem* (LP) is an example of a successful application of the PE in a real word problem. Along this work we notice that these movements produce the expected results, that is, the method efficiently achieves a resulting path according with the given direction.

5.3 Future Work

As we described in Chapter 3, the PE method is conceived as a numerical tool for the treatment of MaOPs. Though the first results are very promising, there are several lines of research arising from this work that may be pursued in the future.

- As we commented in Section 3, the continuation method used by the steering phase of the PE is based on the Pareto Tracer method which does not handle general inequality constraints. Thus, the PE is also available only for general equality constraints and box constraints. The extension of the Pareto Tracer to handle with general equality constraints can be also included as a natural extension to the PE.
- The proposed stopping criteria have demonstrated good results in the practice, however mathematical proofs of these conditions are required to guarantee their general effectiveness.

- The treatment of the DRPP comes so far with a high number of function evaluations, required to obtain good results. Thus, it is desirable a more efficient way to solve the reference point problem in order to reduce the required function evaluations.

- The use of an evolutionary algorithm or other technique to find global solutions is the first step to develop a memetic algorithm with the PE which is consider as future work.

- We observe that the radar char has some issues. Thus, the implementation of new visualization techniques in a *Graphical User Interface* (GUI) which allows a friendly interaction with the DM is one of the most important aspect to do as future work. Indeed, we has been already developed a first version of a GUI.

- Finally, we know that PE method have been developed for the continuous MaOP context. As future work the implementation of PE method for discrete MaOPs may be consider in order to potentiate its applicability for real word problems.

Bibliography

[1] Kaisa Miettinen. *Nonlinear Multiobjective Optimization, volume 12 of International Series in Operations Research and Management Science*. Kluwer Academic Publishers, Dordrecht, 1999.

[2] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[3] Andreas Griewank and George F Corliss. *Automatic differentiation of algorithms: theory, implementation, and application*. Defense Technical Information Center, 1992.

[4] John E Dennis, Jr and Jorge J Moré. Quasi-newton methods, motivation and theory. *SIAM review*, 19(1):46–89, 1977.

[5] Francis Ysidro Edgeworth. *Mathematical psychics: An essay on the application of mathematics to the moral sciences*. Number 10. CK Paul, 1881.

[6] Vilfredo Pareto. *Cours D'économie politique*. Lausanne, F. Rouge; Paris, Pichon, 1896.

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

[8] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, 1950*, pages 481–492, Berkeley and Los Angeles, 1951. University of California Press.

[9] Claus Hillermeier. *Nonlinear multiobjective optimization: A generalized homotopy approach*, volume 135. Springer, 2001.

[10] Oliver Schütze, Adriana Lara, and Carlos. A. Coello. The directed search method for multi-objective optimization problems. In *EVOLVE - A Bridge between Probability, Set Oriented Numerics and Evolutionary Computation*, volume 2 of *Studies in Computational Intelligence*, pages 153–168. Springer, 2013.

[11] Honggang Wang. Zigzag search for continuous multiobjective optimization. *INFORMS Journal on Computing*, 25(4):654–665, 2012.

[12] Steven George Krantz and Harold R Parks. *The implicit function theorem: History, theory, and applications*. Springer, 2002.

[13] Ake Björck. *Numerical methods for least squares problems*. Siam, 1996.

[14] Adanay Martín and Oliver Schütze. A new predictor corrector variant for unconstrained bi-objective optimization problems. In *EVOLVE-A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, pages 165–179. Springer, 2014.

[15] Joerg Fliege, LM Grana Drummond, and Benar Fux Svaiter. Newton's method for multiobjective optimization. *SIAM Journal on Optimization*, 20(2):602–626, 2009.

[16] Oliver Schütze, Adriana Lara, and CA Coello Coello. The directed search method for unconstrained multi-objective optimization problems. *Proceedings of the EVOLVE–A Bridge Between Probability, Set Oriented Numerics, and Evolutionary Computation*, 2011.

[17] Andrzej P Wierzbicki. *A mathematical basis for satisficing decision making*. Springer, 1981.

[18] Andrzej Jaszkiewicz and Roman Słowiński. The 'Light Beam Search' approach an overview of methodology applications. *European Journal of Operational Research*, 113(2):300–314, 1999.

[19] John Telfer Buchanan. A naive approach for solving MCDM problems: The guess method. *Journal of the Operational Research Society*, 48(2):202–206, 1997.

[20] Jürgen Branke, Kalyanmoy Deb, Kaisa Miettinen, and Roman Slowinski. *Multiobjective optimization: Interactive and evolutionary approaches*, volume 5252. Springer Science & Business Media, 2008.

[21] Petri Eskelinen, Kaisa Miettinen, Kathrin Klamroth, and Jussi Hakanen. Pareto navigator for interactive nonlinear multiobjective optimization. *OR spectrum*, 32(1):211–227, 2010.

[22] Kaisa Miettinen and Marko M Mäkelä. Interactive multiobjective optimization system www-nimbus on the internet. *Computers & Operations Research*, 27(7):709–723, 2000.

[23] Marco Farina and Paolo Amato. On the optimal solution definition for many-criteria optimization problems. In *Proceedings of the NAFIPS-FLINT international conference*, pages 233–238, 2002.

[24] Christian von Lücken, Benjamín Barán, and Carlos Brizuela. A survey on multi-objective evolutionary algorithms for many-objective problems. *Computational Optimization and Applications*, 58(3):707–756, 2014.

[25] Francesco Di Pierro, Soon-Thiam Khu, Dragan Savic, et al. An investigation on preference order ranking scheme for multiobjective evolutionary optimization. *Evolutionary Computation, IEEE Transactions on*, 11(1):17–45, 2007.

[26] Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal, and TAMT Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182–197, 2002.

[27] Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *Evolutionary Computation, IEEE Transactions on*, 11(6):712–731, 2007.

[28] Johannes Bader, Kalyanmoy Deb, and Eckart Zitzler. Faster hypervolume-based search using monte carlo sampling. In *Multiple Criteria Decision Making for Sustainable Energy and Transportation Systems*, pages 313–326. Springer, 2010.

[29] Dhish Kumar Saxena, Joao A Duro, Anish Tiwari, Kaushik Deb, and Qingfu Zhang. Objective reduction in many-objective optimization: Linear and nonlinear algorithms. *Evolutionary Computation, IEEE Transactions on*, 17(1):77–99, 2013.

[30] Hernán Aguirre and Kiyoshi Tanaka. Many-objective optimization by space partitioning and adaptive ε-ranking on mnk-landscapes. In *Evolutionary Multi-Criterion Optimization*, pages 407–422. Springer, 2009.

[31] Nicola Beume, Boris Naujoks, and Michael Emmerich. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, 181(3):1653–1669, 2007.

[32] Johannes Bader and Eckart Zitzler. Hype: An algorithm for fast hypervolume-based many-objective optimization. *Evolutionary computation*, 19(1):45–76, 2011.

[33] Hisao Ishibuchi, Yuji Sakane, Noritaka Tsukamoto, and Yusuke Nojima. Evolutionary many-objective optimization by NSGA-II and MOEA/D with large populations. In *Systems, Man and Cybernetics, 2009. SMC 2009. IEEE International Conference on*, pages 1758–1763. IEEE, 2009.

[34] Xiufen Zou, Yu Chen, Minzhong Liu, and Lishan Kang. A new evolutionary algorithm for solving many-objective optimization problems. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 38(5):1402–1412, 2008.

[35] Shengxiang Yang, Miqing Li, Xiaohui Liu, and Jinhua Zheng. A grid-based evolutionary algorithm for many-objective optimization. *Evolutionary Computation, IEEE Transactions on*, 17(5):721–736, 2013.

[36] Hemant Kumar Singh, Amitay Isaacs, and Tapabrata Ray. A Pareto corner search evolutionary algorithm and dimensionality reduction in many-objective optimization problems. *Evolutionary Computation, IEEE Transactions on*, 15(4):539–556, 2011.

[37] Carlos A Coello Coello, David A Van Veldhuizen, and Gary B Lamont. *Evolutionary algorithms for solving multi-objective problems*, volume 242. Springer, 2002.

[38] Carlos A Coello Coello and Nareli Cruz Cortés. Solving multiobjective optimization problems using an artificial immune system. *Genetic Programming and Evolvable Machines*, 6(2):163–190, 2005.

[39] Oliver Schütze, Xavier Esquivel, Adriana Lara, and Carlos A Coello Coello. Using the averaged hausdorff distance as a performance measure in evolutionary multiobjective optimization. *Evolutionary Computation, IEEE Transactions on*, 16(4):504–522, 2012.

[40] T.T. Binh and Korn. An evolution strategy for the multiobjective optimization. *The Second International Conference on Genetic Algorithms*, 1996.

[41] C.M Fonseca and P.J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 1995.

[42] Oliver Shütze, Adriana Lara López, Erick Mejia, and Carlos A. Coello Coello. The directed search method fo unconstrained multi-objective optimization problems. 2010.

[43] Kalyanmoy Deb, Lothar Thiele, Marco Laumanns, and Eckart Zitzler. Scalable multi-objective optimization test problems. In *Proceedings of the Congress on Evolutionary Computation (CEC-2002),(Honolulu, USA)*, pages 825–830. Proceedings of the Congress on Evolutionary Computation (CEC-2002),(Honolulu, USA), 2002.

[44] Oliver Schütze. *Set Oriented Methods for Global Optimization.* PhD thesis, University of Paderborn, 2004.

[45] Pekka Korhonen and Jyrki Wallenius. Visualization in the multiple objective decision-making framework. In *Multiobjective optimization*, pages 195–212. Springer, 2008.

[46] Adnan Y. Tamime. *Cleaning-in-place: dairy, food and beverage operations*, volume 13. John Wiley & Sons, 2009.

[47] Alexandra Blum. *Erstellung eines Konzepts für die Waschqualitätsüberwachung in Großwäschereien.* PhD thesis, 2013.