NOVA SCHOOL OF SCIENCE & TECHNOLOGY

DEPARTMENT OF MATHEMATICS

TRUST-REGION METHODS FOR MULTIOBJECTIVE DERIVATIVE-FREE OPTIMIZATION

ABOOZAR MOHAMMADI

Master in Applied Mathematics (Operations Research)

DOCTORATE IN MATHEMATICS

NOVA University Lisbon
November, 2023

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DEPARTMENT OF
MATHEMATICS

TRUST-REGION METHODS FOR MULTIOBJECTIVE DERIVATIVE-FREE OPTIMIZATION

ABOOZAR MOHAMMADI

Master in Applied Mathematics (Operations Research)

Adviser: Ana Luísa da Graça Batista Custódio
Associate Professor, NOVA School of Science and Technology

Examination Committee

Chair: João Jorge Ribeiro Soares Gonçalves de Araújo
Full Professor, NOVA School of Science and Technology

Rapporteurs: António Ismael de Freitas Vaz
Associate Professor with Habilitation, University of Minho

José Firmino Aguilar Madeira
Coordinator Professor with Habilitation, Instituto Superior de Engenharia de Lisboa

Adviser: Ana Luísa da Graça Batista Custódio
Associate Professor, NOVA School of Science and Technology

Members: João Jorge Ribeiro Soares Gonçalves de Araújo
Full Professor, NOVA School of Science and Technology

Maria do Carmo Proença Caseiro Brás
Associate Professor, NOVA School of Science and Technology

Paula Alexandra da Costa Amaral
Associate Professor, NOVA School of Science and Technology

DOCTORATE IN MATHEMATICS

NOVA University Lisbon
November, 2023

Trust-region Methods for Multiobjective Derivative-free Optimization

This document was created with the (pdf/Xe/Lua)LaTeX processor and the NOVAthesis template (v7.1.18) [22].

Acknowledgements

I want to extend my heartfelt appreciation to several individuals and organizations whose unwavering support and contributions have been instrumental in the successful completion of this thesis.

First and foremost, I must acknowledge the belief I had in myself and my abilities. I persevered in pursuing my goals, never being disappointed in my efforts.

Next, my family, with a special mention to my wife and parents, whose constant support and encouragement were a steady source of strength during this academic journey.

I would like to express my deep gratitude to my professors, whose guidance, knowledge, and expertise played a crucial role in my academic growth.

The financial support provided by the Fundação para a Ciência e Tecnologia (FCT) deserves special recognition. Their funding made it possible for me to conduct this research and complete this thesis.

I want to thank the Department of Mathematics, NOVA Math Research Centre, and the Advanced Training Support Division at NOVA School of Science and Technology (NOVA FCT). Their provision of essential educational facilities was vital to the progress of my thesis.

My heartfelt appreciation goes to the Ph.D. coordination, the esteemed thesis examination committee, and Comissão de Acompanhamento de Tese (CAT) members for their significant contributions to the fulfillment of this project.

The collective support, belief, and contributions of these individuals and organizations have been indispensable in achieving the successful completion of this thesis.

”

‘You cannot teach a man anything; you can only help him discover it in himself.’ (Galileo)

“

Abstract

Multiobjective optimization is an interdisciplinary field with diverse applications in science, engineering, environment, finance, medicine, and many other domains. As objective function components often conflict with each other, finding a single point that simultaneously minimizes all function components becomes impossible. Instead, the solution lies in the Pareto front, which represents a set of nondominated points.

This thesis proposes an algorithm that employs a trust-region approach to approximate the set of Pareto critical points in multiobjective optimization problems. Initially, the algorithm utilizes derivative information from the objective function to compute Taylor models that provide approximations for the different components of it. Subsequently, the algorithm will be adapted to handle multiobjective derivative-free optimization problems, where derivative information is not accessible.

The primary objective of this algorithm is to achieve a comprehensive, densely populated, and uniformly distributed approximation of the complete Pareto front. This is accomplished through an iterative process consisting of two main steps: the *extreme point step* and the *scalarization step*. These steps are performed alternately throughout the algorithm's execution.

During the extreme point step, the algorithm expands the approximation to the Pareto front by moving towards its extreme points. These extreme points correspond to the individual minimization of each objective function component. On the other hand, the scalarization step focuses on reducing gaps along the Pareto front by solving suitable scalarization problems.

The scalarization step incorporates a pivotal additional step, referred to as the *middle point step*. This step plays an important role in determining the initial points for solving the scalarization problems. The significance of this step in ensuring the high performance of the algorithm is substantiated through numerical results.

The algorithm proposed in this thesis incorporates a meticulous approach to managing the limited evaluations of objective functions. It is specifically designed to address scenarios where the evaluation of objective functions is computationally expensive and the budget for such evaluations is restricted. This approach maximizes the quality of the

obtained approximation to the Pareto front, allowing for a more accurate representation of the optimal trade-off solutions in multiobjective optimization problems with expensive and limited function evaluations.

The thesis provides a detailed and comprehensive analysis of the convergence properties of the proposed method under the scenario where derivative information of the objective function is available and utilized. The analysis aims to thoroughly understand and evaluate the behavior of the algorithm as it iteratively advances toward the Pareto critical points during the optimization process.

The results of numerical experiments are reported, to illustrate the effectiveness and robustness of the proposed approach. These experiments are designed with two main purposes. The first goal is to demonstrate the essentiality of each key algorithmic feature in achieving optimal performance by this approach. Secondly, the performance of this algorithm is compared against a state-of-the-art multiobjective derivative-based optimization solver, which inherently attempts to generate approximations of the complete Pareto front for a given problem.

In the second phase of the thesis, following the initial algorithm, we further modify it to compute an approximation of the complete Pareto front in multiobjective derivative-free optimization problems. In this scenario, the objective functions are assumed to be expensive black-box functions, where derivatives are not available and cannot be estimated. The modified algorithm is specifically adapted to fully accommodate the absence of derivatives.

To approximate the objective function components, the modified algorithm incorporates various strategies to navigate the challenges posed by the absence of derivatives. It employs a technique based on polynomial interpolation and minimum Frobenius norm approaches to compute high-quality models that approximate the objective function components.

The convergence analysis for the derivative-free version of the algorithm is conducted. Extensive efforts are devoted to examining the convergence properties of the algorithm, specifically considering the challenge of lacking derivative information.

Detailed numerical results are presented, demonstrating the notable performance of the modified algorithm compared to state-of-the-art multiobjective derivative-free optimization solvers, which also aim to approximate complete Pareto fronts.

Resumo

A otimização multiobjetivo é um campo interdisciplinar com diversas aplicações em ciência, engenharia, ambiente, finanças, medicina e em muitos outros domínios. Como as componentes da função objetivo frequentemente estão em conflito entre si, encontrar um único ponto que minimize simultaneamente todas as componentes da função torna-se impossível. Em vez disso, a solução do problema consiste na determinação da frente de Pareto, que representa um conjunto de pontos não dominados.

Esta tese propõe um algoritmo que utiliza uma abordagem baseada em regiões de confiança para aproximar o conjunto de pontos de Pareto críticos, em problemas de otimização multiobjetivo. Inicialmente, o algoritmo utiliza informação acerca das derivadas da função objetivo para calcular modelos de Taylor que constituem aproximações das suas diferentes componentes. Posteriormente, o algoritmo será adaptado para lidar com problemas de otimização multiobjetivo sem derivadas, onde a informação acerca das derivadas não está acessível.

O objetivo principal deste algoritmo é determinar uma aproximação completa, densamente populada e uniformemente distribuída da fronteira de Pareto. Tal é alcançado por meio de um processo iterativo composto por dois passos principais: o passo de *ponto extremo* e o passo de *escalarização*. Estes passos são realizados alternadamente, durante a execução do algoritmo.

No passo de ponto extremo, o algoritmo expande a aproximação da frente de Pareto, movendo-se em direção aos seus pontos extremos. Esses pontos correspondem à minimização individual de cada componente da função objetivo. Por outro lado, o passo de escalarização concentra-se na redução de hiatos ao longo da frente de Pareto, por meio da resolução de problemas de escalarização adequados.

O passo de escalarização incorpora uma etapa adicional fundamental, designada por passo de *ponto médio*. Esse passo desempenha um papel importante na determinação dos pontos iniciais para a resolução dos problemas de escalarização.

O algoritmo proposto nesta tese incorpora uma abordagem meticulosa para gerir as avaliações limitadas das funções objetivo. É especificamente desenhado para lidar com cenários em que a avaliação das funções objetivo é computacionalmente dispendiosa e

o orçamento para tais avaliações é limitado. Esta abordagem maximiza a qualidade da aproximação obtida para a frente de Pareto, permitindo uma representação mais precisa das soluções ótimas de compromisso em problemas de otimização multiobjetivo com avaliação dispendiosa e limitada das funções.

A tese providencia uma análise detalhada e abrangente das propriedades de convergência do método proposto, no cenário em que a informação acerca das derivadas da função objetivo está disponível e é utilizada. A análise tem como objetivo compreender e avaliar detalhadamente o comportamento do algoritmo à medida que avança iterativamente, durante o processo de otimização, em direção aos pontos de Pareto críticos.

Os resultados da experimentação numérica são relatados para ilustrar a eficácia e a robustez da abordagem proposta. Essas experiências são delineadas com dois objetivos principais. O primeiro, é demonstrar a necessidade de cada elemento estrutural do algoritmo na obtenção de um desempenho ideal por meio desta abordagem. Em segundo lugar, o desempenho do algoritmo é comparado com um código bem estabelecido de otimização multiobjetivo baseado em derivadas, que inerentemente gera aproximações completas da frente de Pareto para um certo problema.

Na segunda parte da tese, seguindo a estrutura algorítmica inicial, são introduzidas modificações que permitam determinar uma aproximação à frente de Pareto completa em problemas de otimização multiobjetivo sem derivadas. Neste cenário, assume-se que as funções objetivo são dispendiosas, do tipo caixa-preta, não estando as derivadas disponíveis, nem podendo ser estimadas. O algoritmo modificado é especificamente adaptado para acomodar a ausência de derivadas.

Para aproximar as componentes da função objetivo, o algoritmo modificado incorpora diversas estratégias para lidar com os desafios impostos pela ausência de derivadas. Técnicas de interpolação polinomial e abordagens de norma de Frobenius mínima são usadas para calcular modelos de alta qualidade que aproximam as componentes da função objetivo.

A análise de convergência para a versão sem derivadas do algoritmo é conduzida. Esforços extensivos são dedicados a examinar as propriedades de convergência do algoritmo, considerando especificamente o desafio da falta de informação acerca das derivadas.

Resultados numéricos detalhados são apresentados, demonstrando o bom desempenho do algoritmo modificado por comparação com códigos de otimização multiobjetivo sem derivadas bem estabelecidos, que também procuram aproximar frentes de Pareto completas.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

BFGS	Broyden, Fletcher, Goldfarb, and Shanno Quasi-Newton Method
BoostDMS	Boost Direct Multisearch Method
DMS	Direct Multisearch Method
DMultiMADS	Mesh Adaptive Direct Multisearch Method
MOIF	Multiobjective Implicit Filtering Method
MOSQP	Multiobjective Sequential Quadratic Programming
MOTR	Multiobjective Trust-Region algorithm
MOTRDFO	Multiobjective Trust-Region Derivative-Free Optimization algorithm
MOTRep	Multiobjective Trust-Region algorithm without the extreme point step
MOTRmiddle	Multiobjective Trust-Region algorithm without the middle point step
MOTRsc	Multiobjective Trust-Region algorithm without the scalarization step
SQP	Sequential Quadratic Programming

1

Introduction

This chapter provides an overview of the thesis from a general perspective. It sets the stage for the rest of the thesis, providing readers with a comprehensive understanding of its context and objectives.

The chapter begins with a brief introduction to optimization in Section 1.1, providing an overview of the fundamental concepts. The problem statement is then introduced in Section 1.2, outlining the specific research problem to be addressed by the thesis. The contributions of the thesis are discussed in Section 1.3, highlighting the novel findings and advancements made in the field. Finally, Section 1.4 presents the organization of the thesis, guiding readers through the structure and content of the subsequent chapters.

1.1 A brief introduction to optimization

Optimization is a fundamental concept that plays a crucial role in various fields of study, providing a diverse range of approaches for finding the best possible solutions to complex problems. It involves the process of maximizing or minimizing an objective function, subject to certain constraints or conditions. The applications of optimization are widespread, spanning disciplines such as engineering, economics, biology, psychology, computer science, and machine learning.

The goal of optimization is to identify the optimal solution that maximizes efficiency, minimizes cost, or achieves desired outcomes. This involves exploring the feasible solution space and iteratively improving the objective function value until an optimal or near-optimal solution is reached. Optimization methods can be broadly classified into two categories: single-objective and multiobjective optimization.

Single-objective optimization focuses on optimizing a single objective while considering the constraints or limitations imposed by the problem. It aims to find the optimal solution with respect to a specific objective, disregarding other objectives or trade-offs that may exist.

On the other hand, multiobjective optimization addresses situations where multiple objectives need to be simultaneously optimized. These objectives often conflict with

each other, making it challenging to find a single solution that optimizes all objectives simultaneously. In real-world problems, decision-makers often encounter situations where optimizing one objective may lead to the degradation of another one. For example, in engineering design, achieving higher performance may increase costs.

Multiobjective optimization aims to find a set of solutions that represent a trade-off between the competing objectives, known as the Pareto front. These solutions form a frontier, where improving one objective requires sacrificing another. Decision-makers can subsequently select the most appropriate solution from this set, considering their preferences, requirements, or any established priorities.

Widely used optimization methods include gradient descent, Newton method, and quasi-Newton methods, which can be used within the structure of trust-region or line search algorithms to efficiently search the solution space. Additionally, conjugate gradient and interior point methods are popular techniques for optimization problems with specific characteristics. Evolutionary algorithms, such as genetic algorithms, particle swarm optimization, and differential evolution, are also commonly employed for their ability to explore complex search spaces and handle constraints, even if without general theoretical guarantees of success.

Scalarization methods are commonly used in multiobjective optimization to convert a multiobjective problem into a single-objective one. These methods aim to find a single scalar objective function that incorporates multiple objectives, allowing the use of traditional single-objective optimization techniques. These methods provide a range of trade-off solutions along the Pareto front by adjusting weights or constraint levels.

Commonly used scalarization methods include the weighted sum method, where objectives are linearly combined, and the $\epsilon-$constraint method, which introduces constraints on specific objectives while optimizing the others. Other scalarization methods include achievement scalarizing functions, penalty-based methods, and augmented $\epsilon-$constraint methods, each offering different ways to represent and optimize multiple objectives as a single scalar objective.

Derivative-free optimization is a branch of optimization that addresses problems where the derivative information of the objective function is either unavailable or computationally expensive to obtain. In many real-world scenarios, such as when dealing with black-box functions or simulation-based models, it is not possible or practical to access the derivatives explicitly. In such cases, derivative-free optimization methods offer an alternative approach to finding optimal or near-optimal solutions.

Unlike traditional optimization methods that rely on the gradient or Hessian information, derivative-free optimization techniques explore the solution space solely based on the evaluations of the objective function. By adapting and refining the search based on the observed function values, derivative-free optimization methods aim to iteratively improve the solution and converge towards the optimum.

The field of derivative-free optimization has gained significant attention in recent years due to its applicability in various domains, including engineering design, financial

modeling, and machine learning. It provides a flexible and robust framework for solving complex optimization problems where derivative information is unavailable. By leveraging techniques specifically designed for derivative-free optimization, researchers and practitioners can tackle a wide range of challenging problems and obtain high-quality solutions without relying on the gradient or derivative information.

Several widely used techniques have been developed to address derivative-free optimization problems. Pattern search methods, including methods such as the Hooke-Jeeves method, involve systematic pattern-based exploration of the search space. Direct search methods, including techniques such as the mesh adaptive direct search algorithm, systematically explore the solution space based on the evaluations of the objective function. Trust-region algorithms have also been commonly used. Evolutionary algorithms, such as genetic algorithms and differential evolution, employ stochastic search and selection mechanisms to explore the search space and refine solutions.

Trust-region methods have been successfully applied in various optimization problems and have gained significant attention. They focus on improving the solution within a trust region around the current point. Trust-region methods employ local models to approximate the objective functions and explore the solution space efficiently. The solution is updated iteratively within a trust region, ensuring convergence towards an optimal solution. The trust-region radius is also updated iteratively as a safety region, limiting the step size and ensuring that the objective function improves within a certain radius.

1.2 Problem statement

This thesis addresses the multiobjective optimization problem

$$\begin{aligned} \min F(x) &= (f_1(x), ..., f_q(x)) \\ \text{s.t. } x &\in \mathbb{R}^n, \end{aligned}$$

where $F : \mathbb{R}^n \to \mathbb{R}^q$, $n, q \in \mathbb{N}$, and $q \geq 2$.

The objective function components $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, q$, are mutually conflicting, posing a challenge in finding a single optimal solution that simultaneously minimizes all of them. The aim is to identify a set of solutions that represent a trade-off among the competing objectives, known as the Pareto front of the problem, which comprises a set of nondominated points.

Additionally, it will be taken into consideration that the evaluation of the objective function is highly expensive. Hence, a crucial criterion for the designed algorithms will be their ability to effectively control the number of function evaluations, thereby enhancing the overall performance of the methods.

Initially, the objective function components f_i, $i = 1, \ldots, q$, are assumed to be twice continuously differentiable, with gradients and Hessians available. These gradients and Hessians will be utilized in the solving strategy, as discussed in Chapter 5.

Subsequently, Chapter 6 will concentrate on addressing the multiobjective derivative-free optimization problem. In this scenario, the objective function components f_i, $i = 1, \ldots, q$, are characterized by the absence of available derivatives and the inability to approximate them. Consequently, a modified technique will be introduced, employing numerical methods to compute models that approximate the true objective function components.

1.3 Contributions of this thesis

This thesis delves into two prominent and demanding domains in the field of optimization. Firstly, it tackles the realm of multiobjective optimization, addressing the challenge of optimizing multiple conflicting objectives simultaneously. Subsequently, it narrows its focus to the more specialized field of multiobjective derivative-free optimization, which involves optimizing multiple conflicting objectives without the availability or possibility of using derivative information.

The aim of this thesis is to explore the fundamentals and methodologies of multiobjective optimization. This research will cover various aspects of multiobjective optimization, including optimality conditions, algorithmic approaches, and evaluation metrics.

This thesis seeks to contribute to the existing body of knowledge by presenting novel methodologies or improvements to existing approaches in multiobjective optimization. In addition to the theoretical analysis, through numerical experimentation, we aim to demonstrate the effectiveness and applicability of the proposed methods.

The contributions of this thesis are significant in addressing the limitations of existing algorithms based on trust-region methods for approximating the complete Pareto front in multiobjective optimization problems. Unlike previous trust-region algorithms that only provide a single critical point as a solution, this thesis presents a novel algorithm that effectively approximates the complete Pareto front using trust-region approaches.

Furthermore, the thesis modifies the original derivative-based framework to address the multiobjective derivative-free optimization. It makes a significant contribution by addressing the limitations of existing trust-region algorithms in the literature for approximating the complete Pareto front in a general multiobjective derivative-free problem.

To overcome the absence of derivative information, innovative strategies based on interpolation techniques are employed to build accurate models approximating the true objective function components. These strategies have the potential to significantly advance derivative-free optimization and can be effectively applied in future research and practical applications.

Overall, the findings of this research contribute to the field of multiobjective optimization by advancing our understanding of both derivative-based and derivative-free optimization techniques. The insights gained from this study provide valuable guidance for addressing the challenges associated with expensive, complex, and conflicting objectives in real-world problems. By developing robust and effective algorithms and strategies,

this thesis opens up new avenues for optimizing complex systems and decision-making processes and paves the way for future advancements in multiobjective optimization. The practical implications of this work can extend to various real-world domains, where efficient and robust optimization methods are in high demand.

1.4 Organization of this thesis

The thesis is organized and outlined in the following structure.

Chapter 2 delves into the mathematical background of single-objective optimization, covering fundamentals and optimality conditions. This chapter also explores techniques for building models that approximate functions.

Chapter 3 is dedicated to single-objective numerical optimization methods. This chapter categorizes algorithms into two groups:

- Derivative-based methods, including line search methods and trust-region derivative-based methods;
- Derivative-free methods, encompassing direct search methods and trust-region derivative-free methods.

Chapter 4 focuses on multiobjective optimization. It begins by discussing the fundamentals of multiobjective optimization, subsequently delving into the study of optimality conditions. A brief review of some derivative-based and derivative-free algorithms is also provided. The chapter concludes with a discussion of some multiobjective metrics and performance evaluation tools that will be used in the numerical sections.

In Chapter 5, a comprehensive framework for multiobjective optimization based on trust-region methods is presented. The chapter starts by introducing the algorithmic structure, followed by a discussion on convergence analysis. Numerical results are then reported to evaluate the efficiency and robustness of the algorithm.

In Chapter 6, the proposed algorithm is modified to a multiobjective derivative-free optimization setting. Initially, the algorithmic structure is introduced, highlighting its adaptation to derivative-free optimization scenarios where derivatives are unavailable. Next, the chapter establishes convergence analysis. Finally, numerical results are reported to assess the efficiency and robustness of the modified algorithm.

To sum up, in Chapter 7, some concluding remarks and discussion on some potential avenues for research are provided.

2

Mathematical background in single-objective optimization

In this chapter, our focus is on introducing the fundamental concepts and terminology related to single-objective optimization. Readers who are familiar with this background may skip it. We will introduce definitions, theorems, and relevant discussions. For a comprehensive understanding, we recommend the references [2, 8, 20, 27] that provide deeper insights into the concepts, techniques, and theoretical foundations.

In Section 2.1, the fundamental terminology of optimization is introduced, providing a solid foundation for understanding the subject. Section 2.2 focuses on optimality conditions, which play a crucial role in analyzing and solving optimization problems. These conditions help to identify the optimal solutions and guide the development of optimization algorithms. Finally, in Section 2.3, the concept of models is explored in detail, emphasizing their significance in optimization algorithms. Adequate models can provide good quality approximations to objective functions, enabling efficient exploration and exploitation of the search space.

2.1 Fundamentals of optimization

Optimization is a topic that appears in almost all scientific areas. Solving an optimization problem means minimizing or maximizing an objective function, often subject to some constraints that need to be satisfied:

$$\begin{aligned} &\min/\max f(x), \\ &\text{s.t. } x \in X \end{aligned}$$

where $f : \mathbb{R}^n \to \mathbb{R}^m$, $n, m \in \mathbb{N}$ and $X \subseteq \mathbb{R}^n$.

A problem is defined as being single-objective if $m = 1$. Otherwise, the problem is said to be multiobjective. In this chapter and in Chapter 3, we will focus on single-objective optimization. The multiobjective case will be addressed in Chapter 4.

Throughout this thesis, we will only consider minimization problems, since a maximization problem can be easily converted into a minimization one, by considering the

symmetry of the objective function. The optimum point will be the same, with a symmetric value for the objective function.

If $X = \mathbb{R}^n$, the problem is said to be unconstrained. When in presence of constraints, $X \subset \mathbb{R}^n$ is defined as:

$$X = \{x \in \mathbb{R}^n | c_i(x) = 0,\ i \in \mathcal{E}; c_j(x) \geq 0,\ j \in \mathcal{I}\},$$

where $\mathcal{E}$ and $\mathcal{I}$ denote the index sets for equality and inequality constraints, respectively. The set X is called the feasible set or the feasible region and $x \in X$ is said to be a feasible point.

A single-objective optimization problem is **linear** when the objective function and all the constraints are linear. Otherwise, it is said to be **nonlinear**. We will address the nonlinear case.

2.2 Optimality conditions

Let us now consider the minimization problem:

$$\min_{x \in X \subseteq \mathbb{R}^n} f(x). \tag{2.1}$$

The following definition formalizes the concept of solution.

Definition 2.1. *A point $x^* \in X$ is a global solution of Problem (2.1) if*

$$\forall x \in X,\ \ f(x^*) \leq f(x).$$

Often it is very difficult to compute a global solution for a given problem. Even if we have access to a point corresponding to the global solution, it may be difficult to verify it [20], which justifies that many optimization methods focus on local solutions.

Definition 2.2. *A point $x^* \in X$ is a local solution of Problem (2.1) if there is $\epsilon > 0$ such that*

$$\forall x \in B(x^*, \epsilon),\ \ f(x^*) \leq f(x),$$

where $B(x^*, \epsilon) = \{x \in X : \|x - x^*\| \leq \epsilon\}$.

However, the previous definitions are not practical for the identification of minimizers, not even if the problem is unconstrained. For this purpose, optimality conditions are derived.

In the subsequent part of this section, our focus will shift toward discussing optimality conditions specifically for unconstrained optimization problems. For readers interested in exploring the topic of constrained optimization, we recommend referring to [20, 27]. These references delve into the subject of constrained optimization, providing comprehensive insights and discussions on the theory and methodologies involved.

Theorem 2.1 states a basic result from mathematical analysis, corresponding to a first-order necessary condition for optimization [20, 27].

Theorem 2.1. *Let $X = \mathbb{R}^n$ and assume that the first-order derivatives of f exist. If x^* is a local solution of Problem (2.1), then*

$$\nabla f(x^*) = 0. \tag{2.2}$$

Points at which the vector gradient is null are said to be stationary points. Although, a stationary point is not always a minimizer [20, 27]. Second-order optimality conditions, which will require the concept of positive definiteness, allow the selection of minimizers from stationary points. A symmetric matrix $H \in \mathbb{R}^{n\times n}$ is said to be positive definite if

$$\forall\, d \in \mathbb{R}^n \setminus \{0\},\ d^\top H d > 0,$$

or, equivalently, if all of its eigenvalues are strictly positive. A matrix $H \in \mathbb{R}^{n\times n}$ is said to be positive semidefinite if

$$\forall\, d \in \mathbb{R}^n,\ d^\top H d \geq 0$$

or equivalently, if all of its eigenvalues are nonnegative.

Theorem 2.2 formalizes the second-order necessary and sufficient conditions for optimality.

Theorem 2.2. *Let $X = \mathbb{R}^n$ and f be a second-order continuously differentiable function in an open neighborhood of x^*. If x^* is a local solution of Problem (2.1), then $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive semidefinite.*

If $\nabla f(x^*) = 0$ and $\nabla^2 f(x^*)$ is positive definite, then x^* is a strict local solution of Problem (2.1).

It was already mentioned that identifying global optimization solutions is, in general, a difficult process. However, convexity simplifies the task. A set $X \subseteq \mathbb{R}^n$ is convex if and only if

$$\forall \lambda \in [0,1],\ \forall x, y \in X,\ \lambda x + (1-\lambda) y \in X.$$

A function f on a convex set X is convex if and only if

$$\forall \lambda \in [0,1],\ \forall x, y \in X,\ f(\lambda x + (1-\lambda) y) \leq \lambda f(x) + (1-\lambda) f(y).$$

A function f on a convex set X is strictly convex if and only if

$$\forall \lambda \in [0,1], \forall x, y \in X,\ x \neq y \Rightarrow\ f(\lambda x + (1-\lambda) y) < \lambda f(x) + (1-\lambda) f(y).$$

An optimization problem is convex if and only if X is a convex set and f is convex on X.

Assume that the Hessian matrix of f is defined. Convexity is equivalent to having the Hessian of f positive semidefinite at every point $x \in X$. Equivalently, strict convexity can be defined as having the Hessian of f positive definite at every point $x \in X$. So, we can relate the above theorems to the concept of convexity [27].

Theorem 2.3. *Every local solution of a convex optimization problem is a global solution.*

Theorem 2.4. *Every local solution of a convex optimization problem with a strictly convex objective function is a unique global solution.*

It is worth noting that information about convexity is not always easy to obtain, especially if derivatives are not available, and often functions to optimize are nonconvex.

2.3 Models in optimization

Some optimization algorithms are model-based methods. In this case, models corresponding to approximations to the objective function are used, because they are less computationally expensive and easier to manipulate. Models with higher quality, that have less error when compared to the original function, result in higher-quality solutions. In this section, we will discuss some techniques for building high-quality linear or quadratic polynomial models $m : \mathbb{R}^n \to \mathbb{R}$ to approximate a function $f : \mathbb{R}^n \to \mathbb{R}$. Linear models are less expensive but usually present a lower quality when compared to quadratic ones because they can't capture the curvature information of the objective function.

Derivatives provide noticeably useful information about functions and can be used to compute high-quality approximations to the objective function by considering Taylor expansions of it (see Section 2.3.1). However, in a derivative-free optimization setting, derivatives are not available, and it isn't possible to estimate them. In this situation, interpolation, regression, or minimum Frobenius norm models can be extremely useful. In the context of expensive function evaluation, we will try to reduce the number of function evaluations required, while still building high-quality models.

Throughout this chapter, let $\mathcal{P}_n^d$ be the space of polynomials in $\mathbb{R}^n$ of degree less than or equal to $d \in \mathbb{N}$. Let b_1 be the dimension of this space. For example, when $d = 1$, $\mathcal{P}_n^d$ only contains linear polynomials and $b_1 = n + 1$. In the quadratic case, for $d = 2$, $\mathcal{P}_n^d$ contains linear and quadratic polynomials and $b_1 = \frac{(n+1)(n+2)}{2}$ holds.

A basis of $\mathcal{P}_n^d$ is defined as

$$\phi = \{\phi_0(x), \phi_1(x), \ldots, \phi_b(x)\},$$

a set of b_1 polynomials of degree less than or equal to d that spans $\mathcal{P}_n^d$, where $x \in \mathbb{R}^n$ and $b_1 = b + 1$. There are several polynomial bases that are worth considering for various applications. We will focus on the natural basis of monomials and on the basis of Lagrange polynomials, that are relevant to the context of this thesis. The first one is detailed here and the second one will be discussed later in this chapter.

The natural basis is the simplest and the most common polynomial basis in the literature [8], and can be written as

$$\overline{\phi} = \left\{1, x_1, \ldots, x_n, \frac{x_1^2}{2}, x_1x_2, \ldots, \frac{x_{n-1}^{d-1}x_n}{(d-1)!}, \frac{x_n^d}{d!}\right\}.$$

For example, if $d = 2$ and $n = 2$, it is defined as

$$\overline{\phi} = \left\{1, x_1, x_2, \frac{x_1^2}{2}, x_1x_2, \frac{x_2^2}{2}\right\},$$

and if $d = 2$ and $n = 3$, it corresponds to

$$\overline{\phi} = \left\{1, x_1, x_2, x_3, \frac{x_1^2}{2}, x_1x_2, \frac{x_2^2}{2}, x_1x_3, x_2x_3, \frac{x_3^2}{2}\right\}.$$

Any polynomial $m(x) \in \mathcal{P}_n^d$ can be written as

$$m(x) = \sum_{j=0}^{b} \alpha_j \phi_j(x),$$

where $\alpha_j \in \mathbb{R}$ are real coefficients, for $j \in \{0, ..., b\}$.

2.3.1 Taylor models

Taylor models are based on derivatives, so gradients and Hessian matrices are required for their computation. The following theorem, known as the Taylor theorem, is a powerful tool in derivative-based optimization [27].

Theorem 2.5. *Suppose that $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and let $a, x \in \mathbb{R}^n$. Then,*

$$f(a + x) = f(a) + \nabla f(a + tx)^\top x,$$

for some $t \in (0, 1)$. Moreover, if f is twice continuously differentiable, then

$$f(a + x) = f(a) + \nabla f(a)^\top x + \frac{1}{2} x^\top \nabla^2 f(a + tx) x,$$

with $t \in (0, 1)$.

A linear approximation to a function f can be obtained by using Taylor linear models

$$m(a + x) = f(a) + \nabla f(a)^\top x, \tag{2.3}$$

with $a, x \in \mathbb{R}^n$. In this thesis, we will often make use of quadratic Taylor models

$$m(a + x) = f(a) + \nabla f(a)^\top x + \frac{1}{2} x^\top \nabla^2 f(a) x, \tag{2.4}$$

where $a, x \in \mathbb{R}^n$.

Error bounds for Taylor models are well-established for first-order and second-order approximations. Theorems 2.6 and 2.7 state them, respectively.

Theorem 2.6. *Let $a \in \mathbb{R}^n$, $\Delta > 0$, and assume that the function f is continuously differentiable in an open domain Ω containing $B(a, \Delta)$, ∇f is Lipschitz continuous in Ω with constant $v_1 > 0$, and m is the corresponding linear Taylor model defined by (2.3), approximating f. Then, for all points x in $B(a, \Delta)$, we have*

- *the error between the gradient of the model and the gradient of the function satisfies*

$$\|\nabla f(x) - \nabla m(x)\| \leq \kappa_{eg} \Delta,$$

- *the error between the model and the function satisfies*

$$|f(x) - m(x)| \leq \kappa_{ef} \Delta^2,$$

where κ_{eg} and κ_{ef} are strictly positive constants depending on v_1.

Theorem 2.7. *Let $a \in \mathbb{R}^n$, $\Delta > 0$, and assume that the function f is twice continuously differentiable in an open domain Ω containing $B(a, \Delta)$, $\nabla^2 f$ is Lipschitz continuous in Ω with constant $v_2 > 0$, and m is the corresponding quadratic Taylor model defined by (2.4), approximating f. Then, for all points x in $B(a, \Delta)$, we have*

- *the error between the Hessian of the model and the Hessian of the function satisfies*

$$\|\nabla^2 f(x) - \nabla^2 m(x)\| \leq \kappa_{eh}\Delta,$$

- *the error between the gradient of the model and the gradient of the function satisfies*

$$\|\nabla f(x) - \nabla m(x)\| \leq \kappa_{eg}\Delta^2,$$

- *the error between the model and the function satisfies*

$$|f(x) - m(x)| \leq \kappa_{ef}\Delta^3,$$

where κ_{eh}, κ_{eg} and κ_{ef} are strictly positive constants depending on v_2.

Some optimization methods, based on Taylor models, use real gradients and Hessian matrices of functions, like what we will do in this thesis. Due to the high cost of computing Hessians, quasi-Newton methods use special techniques to approximate them, like the case of the BFGS or symmetric rank-one formulas (see [20] for more details).

2.3.2 Determined interpolation models

In the absence of derivatives, interpolation techniques can be used to build models. A sample set of $p + 1$ points

$$Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n,$$

for which the corresponding function values are known, is required to compute the interpolating polynomial model m, modelling f at the points in Y.

Depending on the number of available points and the budget of function evaluations, polynomial interpolation models can be built in a determined, when $p = b$, regression, when $p > b$, or underdetermined, when $p < b$, form. The determined case is discussed here. The regression and underdetermined cases will be discussed in Sections 2.3.4, and 2.3.5, respectively. Regarding each case, Taylor-like error bounds, corresponding to the polynomial interpolation models, will also be provided.

Let us consider a sample set of $p + 1$ interpolation points $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$. The interpolating polynomial model $m(x) \in \mathcal{P}_n^d$, expressed as $m(x) = \sum_{j=0}^{p} \alpha_j \phi_j(x)$, where $\phi = \{\phi_0(x), \phi_1(x), \ldots, \phi_p(x)\}$ is a basis of $\mathcal{P}_n^d$, is built by computing the coefficients $\alpha_0, \alpha_1, ..., \alpha_p$, satisfying the interpolation conditions

$$m(y^i) = \sum_{j=0}^{p} \alpha_j \phi_j(y^i) = f(y^i), \quad i = 0, .., p.$$

This linear system of equations can be rewritten as

$$M(\phi, Y)\alpha = f(Y),$$

where,

$$M(\phi, Y) = \begin{bmatrix} \phi_0(y^0) & \phi_1(y^0) & \dots & \phi_p(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \dots & \phi_p(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \dots & \phi_p(y^p) \end{bmatrix},$$

$$\alpha = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_p \end{bmatrix}, \; f(Y) = \begin{bmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^p) \end{bmatrix}.$$

Building a well-defined model m requires having a poised interpolating sample set Y. The concept of poisedness, which is related to the geometry of Y, illustrates how well the points of Y are spread. The following definition [8] introduces the concept of poisedness in polynomial interpolation, which implies that the previous system of equations has a unique solution.

Definition 2.3. *The set $Y = \{y^0, y^1, \dots, y^p\}$ is poised for polynomial interpolation in $\mathbb{R}^n$ if the matrix $M(\phi, Y)$ is nonsingular for some basis ϕ in $\mathcal{P}_n^d$.*

An interesting aspect of this system is that if $M(\phi, Y)$ is nonsingular for some basis ϕ, then it is nonsingular for any basis of $\mathcal{P}_n^d$. So, the poisedness of Y is independent of the selected basis. The following lemma [8] guarantees the uniqueness of the interpolating polynomial m.

Lemma 2.1. *Given a function $f : \mathbb{R}^n \to \mathbb{R}$ and a poised set $Y = \{y^0, y^1, \dots, y^p\}$, the interpolating polynomial m exists and is unique.*

Although ignoring the geometry of Y may fail to maintain global convergence of model-based algorithms and cause poor quality numerical results, checking and controlling the geometry step by step may be computationally expensive, regarding the high number of required function evaluations [8, 32]. So, an efficient algorithm will require an appropriate strategy regarding this issue. We will return to this topic later in this chapter.

Linear interpolation

The simplest determined interpolation technique is linear interpolation, in which m is a polynomial of degree $d = 1$ and $p = n$. Using the natural basis

$$\overline{\phi} = \{1, x_1, \dots, x_n\},$$

linear interpolation can be rewritten in matrix form as

$$\begin{bmatrix} 1 & y_1^0 & \dots & y_n^0 \\ 1 & y_1^1 & \dots & y_n^1 \\ \vdots & \vdots & \vdots & \vdots \\ 1 & y_1^n & \dots & y_n^n \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix} = \begin{bmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^n) \end{bmatrix},$$

$$M(\overline{\phi}, Y)\alpha = f(Y).$$

The following theorem [8, Theorems 2.11 and 2.12] establishes error bounds for linear polynomial interpolation models. Proofs are also available in [8].

Theorem 2.8. *Assume that $Y = \{y^0, y^1, \dots, y^n\} \subset \mathbb{R}^n$ is a poised set of sample points, in the linear interpolation sense, contained in the ball $B(y^0, \Delta(Y))$ of radius $\Delta = \Delta(Y) = \max_{1 \le i \le n} \|y^i - y^0\|$. Furthermore, assume that the function f is continuously differentiable in an open domain Ω containing $B(y^0, \Delta)$, with ∇f Lipschitz continuous in Ω with constant $v_1 > 0$. Let m be the corresponding linear polynomial interpolation model approximating f. Then, for all points x in $B(y^0, \Delta)$, we have*

- *the error between the gradient of the model and the gradient of the function satisfies*

$$\|\nabla f(x) - \nabla m(x)\| \le \kappa_{eg}\Delta,$$

- *the error between the model and the function satisfies*

$$|f(x) - m(x)| \le \kappa_{ef}\Delta^2,$$

where $\kappa_{eg} = v_1 \left(1 + n^{\frac{1}{2}}\|\hat{L}^{-1}\|/2\right)$, $\kappa_{ef} = \kappa_{eg} + v_1/2$, and

$$\hat{L} = \left[\tfrac{y^1 - y^0}{\Delta}, \dots, \tfrac{y^n - y^0}{\Delta}\right]^\top = \begin{bmatrix} \frac{y_1^1 - y_1^0}{\Delta} & \dots & \frac{y_n^1 - y_n^0}{\Delta} \\ \vdots & \vdots & \vdots \\ \frac{y_1^n - y_1^0}{\Delta} & \dots & \frac{y_n^n - y_n^0}{\Delta} \end{bmatrix}.$$

Quadratic interpolation

Interpolating nonlinear techniques are essential to capture the curvature information of functions, decreasing the approximation error of the models. Quadratic interpolation is the simplest way of introducing it.

For example, considering the natural basis when $n = d = 2$, a quadratic polynomial interpolation model will be computed by solving the system

$$M(\overline{\phi}, Y) = \begin{bmatrix} 1 & y_1^0 & y_2^0 & \frac{(y_1^0)^2}{2} & y_1^0 y_2^0 & \frac{(y_2^0)^2}{2} \\ 1 & y_1^1 & y_2^1 & \frac{(y_1^1)^2}{2} & y_1^1 y_2^1 & \frac{(y_2^1)^2}{2} \\ 1 & y_1^2 & y_2^2 & \frac{(y_1^2)^2}{2} & y_1^2 y_2^2 & \frac{(y_2^2)^2}{2} \\ 1 & y_1^3 & y_2^3 & \frac{(y_1^3)^2}{2} & y_1^3 y_2^3 & \frac{(y_2^3)^2}{2} \\ 1 & y_1^4 & y_2^4 & \frac{(y_1^4)^2}{2} & y_1^4 y_2^4 & \frac{(y_2^4)^2}{2} \\ 1 & y_1^5 & y_2^5 & \frac{(y_1^5)^2}{2} & y_1^5 y_2^5 & \frac{(y_2^5)^2}{2} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_5 \end{bmatrix} = \begin{bmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^5) \end{bmatrix},$$

$$M(\overline{\phi}, Y)\alpha = f(Y).$$

Error bounds for quadratic polynomial interpolation models can also be established, this time also for Hessians. Detailed theoretical analysis is available in [8].

Theorem 2.9. *Assume that $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$, with $p + 1 = \frac{(n+1)(n+2)}{2}$, is a poised set of sample points, in the quadratic interpolation sense, contained in the ball $B(y^0, \Delta(Y))$ of radius $\Delta = \Delta(Y) = \max_{1 \le i \le p} \|y^i - y^0\|$. Furthermore, assume that the function f is twice continuously differentiable in an open domain Ω containing $B(y^0, \Delta)$, with $\nabla^2 f$ Lipschitz continuous in Ω with constant $v_2 > 0$. Let m be the corresponding quadratic polynomial interpolation model approximating f. Then, for all points x in $B(y^0, \Delta)$, we have*

- *the error between the Hessian of the model and the Hessian of the function satisfies*

$$\|\nabla^2 f(x) - \nabla^2 m(x)\| \le \kappa_{eh} \Delta,$$

- *the error between the gradient of the model and the gradient of the function satisfies*

$$\|\nabla f(x) - \nabla m(x)\| \le \kappa_{eg} \Delta^2,$$

- *the error between the model and the function satisfies*

$$|f(x) - m(x)| \le \kappa_{ef} \Delta^3,$$

where

$$\begin{aligned} \kappa_{eh} &= 3\sqrt{2} p^{\frac{1}{2}} v_2 \|\hat{M}^{-1}\|/2, \\ \kappa_{eg} &= 3(1 + \sqrt{2}) p^{\frac{1}{2}} v_2 \|\hat{M}^{-1}\|/2, \\ \kappa_{ef} &= (6 + 9\sqrt{2}) p^{\frac{1}{2}} v_2 \|\hat{M}^{-1}\|/4 + v_2/6, \end{aligned}$$

and

$$\hat{M} = M\left(\overline{\phi}, \left\{0, \frac{y^1 - y^0}{\Delta}, \ldots, \frac{y^p - y^0}{\Delta}\right\}\right).$$

Both for linear and quadratic cases, there is the need for controlling the size of $\|\hat{L}^{-1}\|$ and $\|\hat{M}^{-1}\|$, guaranteeing the quality of the models in sufficiently small regions. In the following, we will discuss the notion of Λ-poisedness, at first in terms of Lagrange polynomials and then in terms of condition number, which will be adequate as measures of poisedness.

2.3.2.1 Lagrange polynomials as measure of poisedness

Lagrange polynomials are defined to simplify the solution of the linear system used to compute the model coefficients, by setting $M(\phi, Y)$ equal to the identity matrix.

Definition 2.4. *Given a set of interpolation points $Y = \{y^0, y^1, \ldots, y^p\}$, a basis of $p+1$ polynomials, $l_i(x)$, $i = 0, \ldots, p$ in $\mathcal{P}_n^d$ is called a basis of Lagrange polynomials if it holds*

$$l_i(y^j) = \begin{cases} 1 & i = j \\ 0 & otherwise. \end{cases} \qquad i, j = 0, 1, \ldots, p,$$

As previously mentioned, if Y is poised, then the basis of Lagrange polynomials exists and is uniquely defined [8]. The following lemma [8] establishes how the unique polynomial interpolation model m can be expressed using Lagrange polynomials.

Lemma 2.2. *For any function $f : \mathbb{R}^n \to \mathbb{R}$ and any poised set $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$, the unique polynomial m that interpolates f on Y can be expressed as*

$$m(x) = \sum_{i=0}^{p} f(y^i) l_i(x),$$

where $\{l_i(x), i = 0, \ldots, p\}$ is the basis of Lagrange polynomials for Y .

A measure of poisedness indicates how well an interpolation set Y spans the region where interpolation is of interest. Obviously, this measure depends on Y and on the region under consideration. For example, in the case of linear interpolation in $\mathbb{R}^2$, a set $Y = \{(0,0), (0,1), (1,0)\}$ is a well-poised set in $B(0,1)$, but it is not well-poised in $B(0,100)$ [8]. Below, we provide the definition of Λ-poised sets, introduced in [8] to measure the poisedness of a set of points.

Definition 2.5. *Let $\Lambda > 0$ and a set $B \subset \mathbb{R}^n$ be given. A poised set $Y = \{y^0, y^1, \ldots, y^p\}$ is said to be Λ-poised in B if for the basis of Lagrange polynomials associated with Y,*

$$\Lambda \geq \max_{0 \leq i \leq p} \max_{x \in B} |l_i(x)|. \tag{2.5}$$

The following lemma states some basic properties of Λ-poisedness.

Lemma 2.3. *The following statements hold:*

- *If B contains a point in Y and Y is Λ-poised in B, then $\Lambda \geq 1$.*

- *If Y is Λ-poised in a given set B, then it is Λ-poised (with the same constant) in any subset of B.*
- *If Y is Λ-poised in B, then it is $\widetilde{\Lambda}$-poised in B for any $\widetilde{\Lambda} \geq \Lambda$.*

When the sample set Y is Λ-poised in a given region B, broadly speaking, a low value of Λ illustrates a high measure of poisedness. So, the quality of the interpolation model remarkably deteriorates as Λ becomes large [8].

2.3.2.2 Condition number as measure of poisedness

Generally, the condition number of $M(\phi, Y)$ can not be considered as an appropriate quantity to measure poisedness. It depends on the choice of the polynomial basis ϕ, but the poisedness of Y doesn't. For a given poised interpolation set Y, the condition number of $M(\phi, Y)$ can be equal to any number between 1 and $+\infty$, depending on the selected basis ϕ. Furthermore, for a fixed ϕ, the condition number of $M(\phi, Y)$ also depends on the scaling of Y. However, the poisedness constant Λ remains unaffected by both the shifting and the scaling of the sample set Y. Indeed, if properly defined, the condition number can be closely related to the concept of Λ-poisedness.

Given a sample set $Y = \{y^0, y^1, \ldots, y^p\}$, let's shift it and scale it as

$$\left\{0, \frac{y^1 - y^0}{\Delta}, \ldots, \frac{y^p - y^0}{\Delta}\right\},$$

where

$$\Delta = \max_{1 \leq i \leq p} \|y^i - y^0\|.$$

Define

$$\hat{Y} = \{0, \hat{y}^1, \ldots, \hat{y}^p\} = \left\{0, \frac{y^1 - y^0}{\Delta}, \ldots, \frac{y^p - y^0}{\Delta}\right\} \subset B(0, 1).$$

It is reasoned in [8, Lemma 3.9] that the poisedness constant Λ of an interpolation set Y, and consequently the quality of the interpolation on Y, doesn't change under a shift of the coordinates. Furthermore, the poisedness constant Λ does not depend on the scaling of Y, but it depends on the region B in which the poisedness is considered [8], as stated in [8, Lemma 3.8]. If Y is Λ-poised in B, then Y/Δ is Λ-poised in B/Δ.

The worthy point is that, if we select the basis ϕ as the natural basis and consider $\hat{Y}$, the shifted and scaled version of Y in the fixed region $B(0, 1)$, the condition number of $M(\overline{\phi}, \hat{Y})$ is a meaningful measure of well poisedness. Let's denote $M(\overline{\phi}, \hat{Y})$ by $\hat{M}$, and consider the condition number of $M(\overline{\phi}, \hat{Y})$, defined by $cond(\hat{M}) = \|\hat{M}\|.\|\hat{M}^{-1}\|$.

The following theorem, which is presented in [8, Theorem 3.14], establishes the relationship between the condition number of $\hat{M}$ and the Λ-poisedness constant.

Theorem 2.10. *If $\hat{M}$ is nonsingular and $\|\hat{M}^{-1}\| \leq \Lambda$, then the set $\hat{Y}$ is $\sqrt{p_1}\Lambda$-poised in the ball B(0,1). Conversely, if the set $\hat{Y}$ is Λ-poised in the ball B(0,1), then*

$$\|\hat{M}^{-1}\| \leq \theta\sqrt{p_1}\Lambda,$$

where $\theta > 0$ depends on n and d, but it is independent of $\hat{Y}$ and Λ.

Since all points in $\hat{Y}$ belong to $B(0,1)$ and at least one of the interpolation points has the norm equal to one, then

$$1 \leq \|\hat{M}\| \leq p_1^{\frac{3}{2}}.$$

Therefore, smaller values of $cond(\hat{M})$ illustrate higher levels of poisedness of $\hat{Y}$, and vice versa. As a result, smaller values of $cond(\hat{M})$ traduce into higher quality interpolation sets.

Thus, it will be possible to use the matrix $\hat{M}$ directly, without computing Lagrange polynomials, and still be able to check and control well poisedness of the interpolation sets by restricting $cond(\hat{M})$ to some small user-defined values.

2.3.3 Fully linear and fully quadratic models

In model-based optimization algorithms, Taylor models are a reference, since they have the right quality as approximations to the true functions, to ensure convergence to stationarity.

For interpolation models, this quality has been formalized into the concepts of fully linear [8, Definition 6.1], and fully quadratic models [8, Definition 6.2], reproduced below.

Definition 2.6. *Assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function, with Lipschitz continuous gradient. A set of model functions $\mathcal{M} = \{m : \mathbb{R}^n \to \mathbb{R}, m \in C^1\}$ is called a fully linear class of models if the following hold:*

1. *There exist positive constants κ_{ef} and κ_{eg} such that for any $x \in \mathbb{R}^n$ and $\Delta \in (0, \Delta^{max}]$ there exists a model function $m(x+s) \in \mathcal{M}$, with Lipschitz continuous gradient, such that*

   - *the error between the gradient of the model and the gradient of the function satisfies*

   $$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_{eg}\Delta, \quad \forall s \in B(0,\Delta),$$

   - *the error between the model and the function satisfies*

   $$|f(x+s) - m(x+s)| \leq \kappa_{ef}\Delta^2, \quad \forall s \in B(0,\Delta).$$

   Such a model m is called fully linear on $B(x,\Delta)$.

2. *For this class $\mathcal{M}$ there exists an algorithm, which we will call a 'model-improvement' algorithm, that in a finite, uniformly bounded (with respect to x and Δ) number of steps can*

   - *either establish that a given model $m \in \mathcal{M}$ is fully linear on $B(x,\Delta)$ (we will say that a certificate has been provided and the model is certifiably fully linear),*
   - *or find a model $\bar{m} \in \mathcal{M}$ that is fully linear on $B(x,\Delta)$.*

The fully quadratic class of models imposes stronger assumptions, enabling to establish the second-order convergence results for algorithms that utilize these models. A fully quadratic model accurately captures the second-order behavior of the objective function, in addition to providing the first-order information.

Definition 2.7. *Assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a twice continuously differentiable function with Lipschitz continuous Hessian. A set of model functions $\mathcal{M} = \{m : \mathbb{R}^n \to \mathbb{R}, m \in C^2\}$ is called a fully quadratic class of models if the following hold:*

1. *There exist positive constants κ_{ef}, κ_{eg}, and κ_{eh} such that for any $x \in \mathbb{R}^n$ and $\Delta \in (0, \Delta^{max}]$ there exists a model function $m(x+s) \in \mathcal{M}$, with Lipschitz continuous Hessian, such that*

   - *the error between the Hessian of the model and the Hessian of the function satisfies*

   $$\|\nabla^2 f(x+s) - \nabla^2 m(x+s)\| \leq \kappa_{eh}\Delta, \qquad \forall s \in B(0,\Delta),$$

   - *the error between the gradient of the model and the gradient of the function satisfies*

   $$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_{eg}\Delta^2, \quad \forall s \in B(0,\Delta),$$

   - *the error between the model and the function satisfies*

   $$|f(x+s) - m(x+s)| \leq \kappa_{ef}\Delta^3, \quad \forall s \in B(0,\Delta).$$

   Such a model m is called fully quadratic on $B(x,\Delta)$.

2. *For this class $\mathcal{M}$ there exists an algorithm, which we will call a 'model-improvement' algorithm, that in a finite, uniformly bounded (with respect to x and Δ) number of steps can*

   - *either establish that a given model $m \in \mathcal{M}$ is fully quadratic on $B(x,\Delta)$ (we will say that a certificate has been provided and the model is certifiably fully quadratic),*
   - *or find a model $\bar{m} \in \mathcal{M}$ that is fully quadratic on $B(x,\Delta)$.*

Linear (resp., quadratic) determined interpolation models, computed from Λ-poised interpolation sets, create a class of fully linear (resp., fully quadratic) models and will be particularly useful in Chapter 6 of this thesis, when we will focus on the derivative-free case.

2.3.4 Regression models

There could be situations in which the number $p+1$ of points in the sample set is greater than the dimension of $\mathcal{P}_n^d$, making the linear system associated with the computation of the model coefficients overdetermined.

Let $\phi = \{\phi_0(x), \phi_1(x), \ldots, \phi_b(x)\}$ be a basis of $\mathcal{P}_n^d$ with dimension $b_1 = b+1$ in $\mathbb{R}^n$, and $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$ be a given set of $p+1$ sample points, but here $p > b$. Consider a polynomial $m(x) \in \mathcal{P}_n^d$, formulated as $m(x) = \sum_{j=0}^{b} \alpha_j \phi_j(x)$, where $\alpha \in \mathbb{R}^{b_1}$ are real coefficients.

The coefficients $\alpha_0, \alpha_1, \ldots, \alpha_b$ can be computed by solving the overdetermined system

$$M(\phi, Y)\alpha \overset{l.s.}{=} f(Y),$$

where

$$M(\phi, Y) = \begin{bmatrix} \phi_0(y^0) & \phi_1(y^0) & \dots & \phi_b(y^0) \\ \phi_0(y^1) & \phi_1(y^1) & \dots & \phi_b(y^1) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_0(y^p) & \phi_1(y^p) & \dots & \phi_b(y^p) \end{bmatrix}, \ \alpha = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_b \end{bmatrix}, \ f(Y) = \begin{bmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^p) \end{bmatrix},$$

using a least-squares approach

$$\min_{\alpha} \|M(\phi, Y)\alpha - f(Y)\|^2.$$

The next definition and lemma, originally stated in [8], declare that Y is poised if $M(\phi, Y)$ has full column rank, independent of the selected basis. In this case, the least-squares regression polynomial m is unique. So, an interesting aspect of this system of equations is that if $M(\phi, Y)$ has full column rank for some basis ϕ, then the same applies to any basis of $\mathcal{P}_n^d$.

Definition 2.8. *The set $Y = \{y^0, y^1, \dots, y^p\}$ is poised for polynomial least-squares regression in $\mathbb{R}^n$ if the matrix $M(\phi, Y)$ has full column rank for some basis ϕ in $\mathcal{P}_n^d$.*

Lemma 2.4. *Given a function $f : \mathbb{R}^n \to \mathbb{R}$ and a poised set Y, the least-squares regression polynomial m exists and is unique.*

So, if Y is poised for polynomial least-squares regression, the least-squares regression polynomial is independent of the choice of ϕ.

Although using a sample set with more points than the required number for determined polynomial interpolation involves additional function evaluations, which may not be suitable for derivative-free optimization, there are instances in optimization problems where utilizing regression techniques becomes essential. Generally, when function evaluations are relatively less expensive to compute but noisy, regression techniques are reasonable options.

2.3.4.1 Lagrange polynomials in the regression sense

The notions of Λ-poisedness and Lagrange polynomials can be extended to the case of polynomial least-squares regression.

Consider a set of sample points $Y = \{y^0, y^1, \dots, y^p\}$ with $p > b$. A set of $p+1$ regression Lagrange polynomials $l_i(x)$, $i = 0, \dots, p$, in $\mathcal{P}_n^d$ is defined as

$$l_i(y^j) \overset{l.s.}{=} \begin{cases} 1 & i = j \\ 0 & otherwise. \end{cases} \qquad i, j = 0, 1, \dots, p,$$

Since $p > b$, these polynomials are no longer linearly independent. In [8], it is stated that if Y is poised for polynomial least-squares regression, then the set of regression Lagrange polynomials exists and is uniquely defined. It is also proved that in this case,

the least-squares regression Lagrange polynomial model m that approximates f can be uniquely expressed as

$$m(x) = \sum_{i=0}^{p} f(y^i) l_i(x).$$

Interested readers can find the Taylor-like error bounds for both linear and quadratic regression models in [8], also depending on a Λ-poised constant of Y, defined similarly to (2.5) but now using regression Lagrange polynomials. Furthermore, linear (resp., quadratic) regression models built from Λ-poised sets create a class of fully linear (resp., fully quadratic) models, since they satisfy the error bounds of Definition 2.6 (resp., Definition 2.7).

2.3.5 Underdetermined interpolating models

In underdetermined interpolating, the number of points in the sample set is smaller than the dimension of $\mathcal{P}_n^d$, $(p+1 < b+1)$, causing the need to solve underdetermined linear systems for computing the model coefficients.

Underdetermined interpolating techniques are very useful when function evaluations are expensive and model-based derivative-free optimization algorithms are being used. Imagine there is a budget of 500 function evaluations to solve a multiobjective derivative-free optimization problem with $n = 30$ variables. For this problem, 496 function evaluations are required to build a determined quadratic polynomial interpolation model. Thus, using determined interpolation models will only allow an iteration for the algorithm.

Efficient derivative-free optimization methods attempt to use a low number of new function evaluations at each iteration, by reusing points and considering underdetermined interpolating methods.

In this situation, linear models are reasonable options in the matter of computational expense, but their quality isn't always acceptable since they do not incorporate information on the curvature of the function. Efficient derivative-free optimization methods use fewer points than what is required by quadratic interpolation approaches, but more points than the linear case, in an attempt to capture some of the curvature information and still having an acceptable computational cost [8]. Here, we will focus on underdetermined quadratic interpolation because it will be relevant to what follows.

Let $\phi = \{\phi_0(x), \phi_1(x), \ldots, \phi_b(x)\}$ be a basis of $\mathcal{P}_n^2$ with dimension $b+1 = \dfrac{(n+1)(n+2)}{2}$, and consider $Y = \{y^0, y^1, \ldots, y^p\}$, with $p < b$. The interpolation polynomial m, defined by

$$m(y^i) = \sum_{j=0}^{b} \alpha_j \phi_j(y^i) = f(y^i), \quad i = 0, \ldots, p,$$

is no longer unique, because $M(\phi, Y)$ has more columns than rows.

A common approach in underdetermined interpolating techniques is based on the minimum Frobenius norm[1]. Minimum Frobenius norm models have been successfully employed in single-objective interpolation-based trust-region methods. These methods require between $n+1$ and $(n+1)(n+2)/2$ points to build a quadratic model. The use of these models provides a more accurate gradient approximation compared to linear interpolation, while the accuracy of the Hessian approximation may be lower than the one of determined quadratic interpolation. This approach strikes a balance between accuracy and computational efficiency. By utilizing minimum Frobenius norm models, interpolation-based trust-region methods can effectively capture the behavior of the objective function, enabling improved convergence toward the optimal solution. The goal is to develop derivative-free algorithms in which fewer expensive function evaluations are required to build models, but the numerical results are still very satisfying.

To discuss the idea of minimum Frobenius norm models, we introduce the following theorem, which states general error bounds for underdetermined quadratic polynomial interpolation [8, Theorem 5.4].

Theorem 2.11. *Assume that $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$, with $p < b$, is a Λ-poised set of sample points in the linear interpolation sense, or in the linear regression sense if $p > n$, contained in the ball $B(y^0, \Delta(Y))$ of radius $\Delta = \Delta(Y) = \max_{1\leq i\leq p} \|y^i - y^0\|$. Furthermore, assume that the function f is continuously differentiable in an open domain Ω containing $B(y^0, \Delta)$, ∇f is Lipschitz continuous in Ω with constant $v_1 > 0$, and m is the corresponding quadratic underdetermined interpolation model approximating f. Then, for all points x in $B(y^0, \Delta)$, we have:*

- *the error between the gradient of the model and the gradient of the function satisfies*

$$\|\nabla f(x) - \nabla m(x)\| \leq C_p \Lambda \left(v_1 + \|H\|\right) \Delta,$$

- *the error between the model and the function satisfies*

$$|f(x) - m(x)| \leq \left(C_p \Lambda + \frac{1}{2}\right) \left(v_1 + \|H\|\right) \Delta^2,$$

where H is the Hessian matrix of m, and C_p is a positive constant depending on p.

Therefore, given a sample set Y, regarding the accuracy of a quadratic underdetermined interpolation model, it is reasonable to look for a model with minimum Frobenius norm of the corresponding Hessian.

We split the natural basis $\overline{\phi}$ into linear and quadratic parts, $\overline{\phi}_L = \{1, x_1, \ldots, x_n\}$ and $\overline{\phi}_Q = \left\{\frac{x_1^2}{2}, x_1 x_2, \ldots, \frac{x_2^n}{2}\right\}$, respectively. By defining α_L and α_Q as the corresponding linear

[1] Frobenius norm is defined for squared matrices by $\|A_{n\times n}\|^2 = \sum_{i=1}^{i=n} \sum_{j=1}^{j=n} a_{ij}^2$. Equivalently, the Frobenius norm can be defined as the trace of a matrix's inner product, $\|A\|^2 = tr(A^\top A)$, remember that the trace of a squared matrix is the sum of its diagonal entries.

and quadratic parts of the coefficient vector α, respectively, the interpolation model can be expressed by

$$m(x) = \alpha_L^\top \overline{\phi}_L(x) + \alpha_Q^\top \overline{\phi}_Q(x).$$

The solution of the problem

$$\min \frac{1}{2} \|\alpha_Q\|^2,$$

$$\text{s.t.} \quad M(\overline{\phi}_L, Y)\alpha_L + M(\overline{\phi}_Q, Y)\alpha_Q = f(Y)$$

is called the minimum Frobenius norm solution. Considering the natural basis and the separation of α, minimizing the norm of α_Q is equivalent to minimizing the Frobenius norm of the Hessian of m [8]. A sample set Y is poised in the minimum Frobenius norm sense, or equivalently, the minimum Frobenius norm polynomial exists and is unique if and only if the matrix $F(\overline{\phi}, Y)$, defined as follows, is nonsingular:

$$F(\overline{\phi}, Y) = \begin{bmatrix} M(\overline{\phi}_Q, Y)M(\overline{\phi}_Q, Y)^\top & M(\overline{\phi}_L, Y) \\ \\ M(\overline{\phi}_L, Y)^\top & 0 \end{bmatrix}.$$

Poisedness in the minimum Frobenius norm sense implies poisedness in the linear interpolation or regression senses [8]. To measure the level of poisedness, interested readers can notice [8, Definition 5.6], defining Λ-poised in the minimum Frobenius norm sense.

According to the following theorem, the Hessians of the minimum Frobenius norm models are also bounded [8, Theorem 5.7].

Theorem 2.12. *Let $Y = \{y^0, y^1, \ldots, y^p\} \subset \mathbb{R}^n$, with $p < b$, be a set of sample points contained in the ball $B(y^0, \Delta)$, where $\Delta = \max_{1 \le i \le p} \|y^i - y^0\|$. Assume that the function f is continuously differentiable in an open domain Ω containing $B(y^0, \Delta)$, ∇f is Lipschitz continuous in Ω with constant $v_1 > 0$, and let m be the corresponding minimum Frobenius norm model approximating f. If Y is Λ-poised in the minimum Frobenius norm sense then*

$$\|H\| \le C_{p,b} v_1 \Lambda,$$

where H is the Hessian matrix of m, and $C_{p,b}$ is a positive constant depending on p and b.

Incorporating this result in Theorem 2.11 allows us to obtain the Taylor-like error bounds for minimum Frobenius norm models. Furthermore, minimum Frobenius norm models based on a Λ-poised sample set in the minimum Frobenius norm sense are fully linear, considering Definition 2.6, in which κ_{ef} and κ_{eg} depend on Λ, p, b, and on the Lipschitz constant of ∇f [8].

There is an alternative kind of minimum Frobenius norm models, suggested by Powell [28], in which the model with the Hessian matrix H closest to a previous Hessian, in the Frobenius norm sense, is selected.

The topic of minimum Frobenius norm models is a recurrent theme in derivative-free optimization. We suggest [8, 10] for more comprehensive information.

3

Single-objective numerical optimization

In this chapter, we present the basic concepts and methodologies for single-objective optimization. Readers who are familiar with these techniques may skip it.

Throughout this chapter, consider the following minimization problem

$$\min_{x \in \mathbb{R}^n} f(x),$$

with $f : \mathbb{R}^n \to \mathbb{R}$.

The chapter will be organized separately into two main parts: derivative-based and derivative-free optimization. Section 3.1 will be dedicated to derivative-based methods. Derivative-free methods are presented in Section 3.2. Line search, trust-region, and direct search methods will be proposed in a brief review, given their role in optimization algorithms. Comprehensive details can be found in [20, 27] and [2, 8], for derivative-based and derivative-free optimization, respectively.

3.1 Derivative-based methods

In this section, we assume that derivatives are available and we can use quadratic Taylor models, described in Subsection 2.3.1. Two well-known approaches will be introduced: line search and trust-region methods, which will be discussed in Subsections 3.1.1 and 3.1.2, respectively.

Line search and trust-region methods have gained widespread usage in optimization. Both are fundamental techniques employed by various optimization algorithms, including gradient descent, Newton method, and quasi-Newton methods, among others.

One of the advantages of line search and trust-region methods is their ease of implementation, making them accessible to a wide range of users and applications. These methods offer effective strategies for solving optimization problems in various fields, including engineering, finance, machine learning, and many others.

3.1.1 Line search methods

Line search methods have indeed been extensively utilized in various optimization algorithms [20, 27]. These methods offer a straightforward and practical approach to optimizing objective functions.

Let x_k be the current iterate and $d_k \in \mathbb{R}^n$ be the search direction selected at x_k. The new point is given by

$$x_{k+1} = x_k + \alpha_k d_k,$$

where α_k is a positive scalar, called the step length, and it is required to satisfy

$$f(x_k + \alpha_k d_k) < f(x_k).$$

The ideal choice for α_k would be the global solution of a one-dimensional minimization problem given by

$$\min_{\alpha>0} \Phi(\alpha), \tag{3.1}$$

where $\Phi(\alpha) = f(x_k + \alpha d_k)$, for $\alpha > 0$. Usually, it is computationally too expensive, or even impossible, to find this minimizer. So, in most line search algorithms, an approximation to the solution is also accepted.

In order to ensure that the function f is reduced along the search direction d_k, it is necessary for d_k to be a descent direction. This means that d_k must satisfy the condition:

$$d_k^\top \nabla f(x_k) < 0.$$

The search direction d_k usually has the form

$$d_k = -B_k^{-1} \nabla f(x_k),$$

where B_k is some nonsingular and symmetric matrix of order n. When B_k is positive definite, then d_k is a descent direction, since

$$d_k^\top \nabla f(x_k) = -\nabla f(x_k)^\top B_k^{-1} \nabla f(x_k) < 0.$$

The flexibility in choosing the matrix B_k in optimization algorithms allows for a wide variety of line search methods. Some of the most commonly used line search algorithms include:

- Steepest descent method, where B_k is simply the identity matrix I;
- Newton method, where B_k is exactly the Hessian matrix $\nabla^2 f(x_k)$;
- Quasi-Newton methods, where B_k is an approximation to the Hessian matrix $\nabla^2 f(x_k)$.

To guarantee convergence, additional requirements should be applied to the search direction. As mentioned, d_k is required to be a descent direction, but this is not practically enough. Because d_k may be a descent direction but very close to being orthogonal to $\nabla f(x_k)$ and thus the algorithm wouldn't make noticeable progress toward a solution. To handle this, d_k is also required to produce sufficient descent that is denoted by

$$-\frac{d_k^\top \nabla f(x_k)}{\|d_k\|.\|\nabla f(x_k)\|} \geq \epsilon > 0,$$

for all k, where ϵ is a positive tolerance.

To ensure convergence, additional assumptions on the step length are also required. At first, α_k should produce a sufficient decrease in the objective function value. To achieve this, the following inequality, called the Armijo condition, should be satisfied

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \mu \alpha_k d_k^\top \nabla f(x_k), \tag{3.2}$$

for all k, where $\mu \in (0, 1)$ is some scalar. A small value of μ means that a small decrease in the function value is enough for the new point to be accepted.

The sufficient decrease condition is satisfied for all sufficiently small values of α_k. So, it is not enough by itself to ensure that the algorithm makes reasonable progress, especially when α_k is too small. Therefore, another condition on α_k is that it shouldn't be too small. A simple approach to handle this condition uses backtracking, in which α_k can be chosen as the first element of the sequence

$$\left\{1, \frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \dots\right\},$$

that satisfies the Armijo condition (3.2).

One notable benefit of employing backtracking techniques in algorithms is their ease of implementation. Moreover, it is possible to establish the convergence analysis [20]. However, the backtracking approaches are not recommended for practical algorithms [20]. Although they can be applied to Newton method, they are less appropriate for quasi-Newton methods [27].

Another strategy to guarantee that the step length α_k is not too small, uses the so-called curvature condition, which is formulated as

$$d_k^\top \nabla f(x_k + \alpha_k d_k) \geq \eta d_k^\top \nabla f(x_k),$$

for all k, where $\eta \in (\mu, 1)$ is some scalar, and μ is the constant from (3.2).

The combination of the Armijo and curvature conditions is known as the Wolfe conditions, which are used in most line search methods, leading to more effective algorithms.

The line search approach is detailed by Algorithm 1.

Theoretical results concerning the convergence analysis of line search methods have been established independently for both the Armijo-backtracking procedure and when

Algorithm 1. Line search algorithm

Input
Initial point x_0.

For $k = 0, 1, 2, \ldots$
1. Direction calculation
Compute $d_k = -B_k^{-1}\nabla f(x_k)$, where B_k is sufficiently positive definite.

2. Line search update
Set $x_{k+1} = x_k + \alpha_k d_k$, where α_k satisfies the Wolfe conditions
or is computed using an Armijo-backtracking approach.
End For

applying the Wolfe conditions. The results are available in the case of the Armijo-backtracking approach in [20, Theorem 11.7]. Here, we consider the analysis for the Wolf conditions, stated by the following theorem, where the detailed proofs are available in [27, Section 3.2].

Theorem 3.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real-valued function and x_0 be some given initial point. Assume that the level set $S = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ is bounded. Suppose that f is continuously differentiable in an open domain Ω containing S, with ∇f Lipschitz continuous on Ω. Let $\{x_k\}_{k\in\mathbb{N}}$ be the sequence of iterates generated by Algorithm 1, where $x_{k+1} = x_k + \alpha_k d_k$, such that*

- *d_k is a descent direction producing sufficient descent. So, there is a positive constant ϵ such that*
$$-\frac{d_k^\top \nabla f(x_k)}{\|d_k\|.\|\nabla f(x_k)\|} \geq \epsilon.$$
- *α_k satisfies the Wolfe conditions.*

Then
$$\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0.$$

3.1.2 Trust-region derivative-based methods

Trust-region methods are powerful and efficient techniques that have been efficiently used in optimization [7], and have led to practical implementations and commercial software [25].

The general framework of a trust-region algorithm is relatively straightforward, yet highly versatile and applicable to various optimization problems. Trust-region methods provide a flexible approach by iteratively updating and minimizing a model of the objective function within a limited trust region around the current iterate, where the model is considered to be accurate enough. These methods dynamically adjust the solution and the

size of the trust region based on the performance of the model and the objective function, allowing for adaptive exploration and exploitation of the search space.

At iteration k, the model function m_k is computed around the current iterate x_k and minimized in a trust region, typically a ball around the current iterate, with a given radius. Depending on how well the model reduction predicts the actual decrease in the objective function value, the model minimizer is accepted or rejected, and the trust-region radius is updated.

The algorithmic structure of trust-region and line search algorithms are different. In a trust-region step, we try to compute an appropriate descent direction inside a trust region. On the other hand, a line search step generates a descent direction at first and then tries to calculate the best step length along it.

In this section, the model m_k is assumed to be a quadratic polynomial based on the Taylor expansion of f around x_k, defined as

$$m_k(x_k + d) = f(x_k) + d^\top g_k + \frac{1}{2} d^\top B_k d,$$

where $g_k = \nabla m_k(x_k) = \nabla f(x_k)$ and $B_k = \nabla^2 m_k(x_k)$.

At iteration k, the trust-region subproblem

$$\begin{aligned} \min_{d \in \mathbb{R}^n} \quad & m_k(x_k + d) \\ \text{s.t.} \quad & \|d\| \leq \Delta_k, \end{aligned} \tag{3.3}$$

is solved to compute the search direction d_k, where $\Delta_k > 0$ is the trust-region radius. In other words, we compute the direction $d_k \in \mathbb{R}^n$, where $x_k + d_k$ is inside the current trust region, that is $x_k + d_k \in B(x_k, \Delta_k) = \{x \in \mathbb{R}^n \mid \|x - x_k\| \leq \Delta_k\}$.

Different choices of B_k have led to the development of different trust-region algorithms. Some of the most commonly used algorithms include:

- Trust-region steepest descent method, where $B_k = 0$ and

$$d_k = -\frac{\Delta_k \nabla f(x_k)}{\|\nabla f(x_k)\|};$$

- Trust-region Newton method, where B_k is exactly the Hessian matrix $\nabla^2 f(x_k)$;
- Trust-region quasi-Newton methods, where B_k is an approximation to the Hessian matrix $\nabla^2 f(x_k)$.

The essential aspect of any trust-region algorithm is the trust-region management, involving whether to accept or reject the new point and how to update the trust-region radius. Generally, if the reduction obtained in the true function is not too small compared to the reduction of the model, then the new point is accepted and the trust-region radius can be increased; this is a successful iteration. Otherwise, the new point is rejected and the trust-region radius is decreased to allow more accurate models in the remaining iterations;

this is an unsuccessful iteration. For this purpose, the ratio ρ_k is computed to measure the agreement between the model m_k and the real objective function f, as

$$\rho_k = \frac{f(x_k) - f(x_k + d_k)}{m_k(x_k) - m_k(x_k + d_k)}.$$

A high value of ρ_k means that the model adequately predicts the reduction in the function value, and vice versa.

The trust-region approach is described by Algorithm 2.

Algorithm 2. Trust-region algorithm

Input

Initial point x_0, initial trust-region radius Δ_0,
values for the parameters $0 < \eta_1 \leq \eta_2 < 1$ and $0 < \mu_1 < 1 < \mu_2$.

For $k = 0, 1, 2, \ldots$

1. Step calculation

Obtain d_k by solving the trust-region subproblem (3.3).
Compute $\rho_k = \dfrac{f(x_k) - f(x_k + d_k)}{m_k(x_k) - m_k(x_k + d_k)}$.

2. Trial point acceptance and trust-region update

If $\rho_k \geq \eta_1$ then:
  Set $x_{k+1} = x_k + d_k$.
  If $\rho_k \geq \eta_2$ and $\|d_k\| = \Delta_k$ then set $\Delta_{k+1} = \mu_2 * \Delta_k$.
Else, set $x_{k+1} = x_k$ and $\Delta_{k+1} = \mu_1 * \Delta_k$.

End For

Some techniques to solve subproblem (3.3) are detailed in [7, 27]. Indeed, in most trust-region algorithms, finding an exact solution to the trust-region subproblem is computationally expensive or even impossible, especially for large-scale problems. Therefore, most trust-region algorithms only find an approximate solution to this subproblem. However, to guarantee convergence, it is required to compute d_k, an approximate solution of the subproblem (3.3), that lies within the trust region and provides a sufficient reduction in the model function m_k:

$$m_k(x_k) - m_k(x_k + d_k) \geq \kappa \, \|g_k\| \, \min\left\{\frac{\|g_k\|}{\|B_k\|}, \Delta_k\right\}, \tag{3.4}$$

for some constant $\kappa \in (0, 1]$.

Let's quantify the model reduction along the steepest descent direction

$$d_k^S = -\frac{\Delta_k}{\|g_k\|} g_k.$$

Define the Cauchy point as the minimizer of m_k along d_k^S by

$$x_k^C = x_k + d_k^C,$$

where $d_k^C = \alpha_k d_k^S$ and α_k is computed by solving

$$\min_{\alpha \geq 0} \left\{ m_k(x_k + \alpha d_k^S) \ : \ \|\alpha d_k^S\| \leq \Delta_k \right\}.$$

The minimizer α_k depends on the value of $g_k^\top B_k \, g_k$, as

$$\alpha_k = \begin{cases} 1 & g_k^\top B_k \, g_k \leq 0, \\ \min\left\{ \frac{\|g_k\|^3}{\Delta_k \, g_k^\top B_k \, g_k}, 1 \right\} & \text{otherwise.} \end{cases}$$

It can be proved that d_k^C satisfies (3.4) with $\kappa = \frac{1}{2}$, [27, Lemma 4.3], that is

$$m_k(x_k) - m_k(x_k + d_k^C) \geq \frac{1}{2} \, \|g_k\| \, \min\left\{ \frac{\|g_k\|}{\|B_k\|}, \Delta_k \right\}.$$

It is clear that the exact solution d_k^* of (3.3) also satisfies (3.4) with $\kappa = \frac{1}{2}$, because

$$m_k(x_k) - m_k(x_k + d_k^*) \geq m_k(x_k) - m_k(x_k + d_k^C).$$

However, for approximately solving the subproblem (3.3), a fixed positive fraction of the Cauchy decrease is enough.

In this section, we consider that $B_k = \nabla^2 f(x_k)$. The case of quasi-Newton methods is also well-discussed in [7, 27].

The convergence analysis for trust-region methods is stated by the following theorem. The proof of this theorem is available in [20, Theorem 11.11].

Theorem 3.2. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real-valued function, x_0 be some given initial point, and $\{x_k\}_{k \in \mathbb{N}}$ be the sequence generated by Algorithm 2. Assume that the level set S, defined by $S = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ is bounded. Furthermore, suppose that $\nabla^2 f$ exists and is continuous for all $x \in S$. Then*

$$\lim_{k \to +\infty} \|\nabla f(x_k)\| = 0.$$

3.2 Derivative-free optimization

In this section, we focus on optimization problems where derivative information is not available and cannot be estimated. In derivative-free optimization, the use of Taylor models, which rely on derivative information, is not possible. Thus, alternative techniques need to be employed to make progress toward the optimal solution.

Derivative-free optimization methods often rely on strategies such as direct search or surrogate modeling to optimize the objective function without explicitly calculating derivatives. These approaches aim to explore the search space efficiently, make use of limited function evaluations, and iteratively refine the solution without relying on gradient information. We consider two popular and widely used approaches for solving derivative-free problems: directional direct search and trust-region methods.

In Subsection 3.2.1, we introduce directional direct search methods that explore the search space by evaluating the objective function at a set of candidate points, without relying on derivative information.

In Subsection 3.2.2, we discuss trust-region methods for derivative-free optimization. These methods aim to find a solution inside a trust region that provides a sufficient reduction in the objective function, without requiring derivative information.

Both direct search and trust-region methods are effective in handling optimization problems when derivative information is unavailable or expensive to compute. They provide robust and efficient approaches for exploring and optimizing the objective function in the absence of derivatives.

3.2.1 Directional direct search methods

Directional direct search methods proceed with sampling the objective function at a finite number of points along specified directions, scaled by a step size, from the current iterate.

During each iteration, directional direct search methods generate candidate points by moving the current iterate along predetermined directions. The objective function is then evaluated at these candidate points to gain insight into its local behavior. Based on these evaluations, the algorithm determines how to update the current iterate and the step size parameter in order to potentially improve the solution.

The selection of the search directions is crucial, as it influences the algorithm's ability to explore and exploit the search space effectively. Common choices include random directions, coordinate directions, or other systematic patterns. By iteratively sampling the objective function at different candidate points, directional direct search methods aim to converge toward an optimal or near-optimal solution, without relying on derivative information.

One of the simplest directional direct search methods is the coordinate search method, also known as compass search. The corresponding algorithm is outlined in Algorithm 3.

At iteration k, coordinate search evaluates the function f at the set of poll points P_k, defined by

$$P_k = \{x_k + \alpha_k d : \ d \in D_\oplus\} \,,$$

where x_k is the current iterate, α_k is the current step size, and $D_\oplus$ is the maximal positive basis given by

$$D_\oplus = [I_n \ \ -I_n] \,,$$

with I_n as the identity matrix of order n.

If some points in P_k lead to a decrease in the value of f, the iteration is declared successful and the step size remains unchanged. Otherwise, the iteration is unsuccessful and the step size is decreased.

The convergence analysis for the presented coordinate search algorithm is stated by the following theorem. A detailed proof of this theorem can be found in [2, Theorem 3.4].

Algorithm 3. Coordinate search algorithm

Input

Initial point x_0 and initial step size α_0.

For $k = 0, 1, 2, \ldots$

1. Poll step

Evaluate f at the poll points $P_k = \{x_k + \alpha_k d : d \in D_\oplus\}$,
following a pre-determined order, until a point $x_k + \alpha_k d_k \in P_k$ is found
such that $f(x_k + \alpha_k d_k) < f(x_k)$.

If such a point $x_k + \alpha_k d_k$ is found, then:
Set $x_{k+1} = x_k + \alpha_k d_k$.
Declare the iteration and the poll step as successful.
Stop poll step.

Otherwise:
Set $x_{k+1} = x_k$.
Declare the iteration and the poll step as unsuccessful.

2. Parameter update

If the iteration is successful, set $\alpha_{k+1} = \alpha_k$.
Otherwise, set $\alpha_{k+1} = \frac{1}{2}\alpha_k$.

End For

Theorem 3.3. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a real-valued function and x_0 be some given initial point. Let $\{x_k\}_{k\in\mathbb{N}}$ be the sequence of points generated by Algorithm 3. Assume that the level set $S = \{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$ is bounded. Furthermore, suppose that f is continuously differentiable in an open domain Ω containing S. Then*

$$\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0.$$

For a comprehensive understanding of directional direct search methods, interested readers can refer to the details provided in [2, 8].

3.2.2 Trust-region derivative-free methods

The trust-region approach was discussed in Section 3.1.2, and the corresponding algorithmic structure was presented by Algorithm 2. In Section 3.1.2, the quadratic model m_k was assumed to be based on the Taylor expansion of f around x_k which here is no longer possible, because derivatives are not available.

In this section, models will be built using polynomial interpolation, regression, or minimum Frobenius norm approaches, discussed in Section 2.3.

The model m_k is assumed to be a quadratic polynomial model based on interpolation techniques, approximating f around x_k, defined as

$$m_k(x_k + d) = f(x_k) + d^\top g_k + \frac{1}{2} d^\top B_k d,$$

where $g_k = \nabla m_k(x_k)$ and $B_k = \nabla^2 m_k(x_k)$.

A sample set

$$Y_k = \{y^0, y^1, \dots, y^p\} \subset \mathbb{R}^n,$$

for which the corresponding function values are known, is required to compute the interpolating polynomial model m_k, modeling f at the points in Y_k. Generally, the first element is equal to the current iterate, that is $y^0 = x_k$.

At each iteration k, Y_k should be updated, depending on whether the iteration is successful ($x_{k+1} = x_k + d_k$) or unsuccessful ($x_{k+1} = x_k$). To update Y_k, most trust-region derivative-free algorithms consider x_{k+1} as the first element in Y_{k+1} and then discard the furthest point from x_{k+1} or don't discard any point.

Generally, there is a restriction on the points in the sample set Y_k considering their distance from x_k. In this case, a sample point $y \in Y_k$ should satisfy

$$\|y - x_k\| \leq r\Delta_k,$$

where $\Delta_k > 0$ is the trust-region radius, and $r > 0$ is a user-defined parameter. All points outside the ball $B(x_k, r\Delta_k)$ are discarded.

The important issue here is whether the poisedness of Y_k, discussed in Section 2.3, should be checked and controlled or not. Appropriately poised sample sets result in high-quality models and consequently good solutions. On the other hand, controlling the geometry of sample sets at each iteration is computationally very expensive and may finally lead to inadequate numerical results, especially in derivative-free optimization algorithms with a restricted budget of function evaluations.

Regarding the poisedness of sample sets, different strategies have been used. In [16], the authors present a trust-region algorithm and state that ignoring the control of the geometry of sample sets does not harm the efficiency and robustness of their algorithm, which is competitive with other model-based algorithms that use a geometry phase. On the other hand, the role of geometry in model-based derivative-free optimization algorithms is emphasized in [32], in which they believe that when the model gradient is small, the sufficient poisedness of the model is necessarily required.

In the literature [8], interested readers can find algorithms that address the completion of nonpoised sample sets or improve their poisedness using techniques such as Lagrange polynomials or LU factorization. These algorithms provide strategies to enhance the quality and properties of sample sets in optimization problems. By completing or improving the poisedness of the sample sets, the algorithms aim to improve the accuracy and efficiency of optimization algorithms that rely on these sample sets.

Derivative-free trust-region algorithms are introduced in both first-order and second-order cases with convergence analysis in [8, Chapter 10].

4

Multiobjective numerical optimization

A multiobjective optimization problem is defined by

$$\min_{x \in X} F(x) = \left(f_1(x), ..., f_q(x)\right), \tag{4.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^q$, with $n, q \in \mathbb{N}$, $q \geq 2$, and the feasible region $X \subseteq \mathbb{R}^n$.

The objective function components $f_i : \mathbb{R}^n \to \mathbb{R}$, where $i \in \{1, ..., q\}$, are considered to be conflicting among each other. This means that improving one objective function component leads to the deterioration of at least another one, resulting in a trade-off between different optimization goals.

The conflicting nature of the objective function components makes the multiobjective optimization problem challenging and requires specialized algorithms to efficiently explore and approximate the solution, providing a set of points that represent different trade-offs among the conflicting objectives.

In Section 4.1, we will delve into the fundamental aspects of multiobjective optimization. Afterward, in Section 4.2, we will outline the optimality conditions associated with multiobjective optimization problems. Section 4.3 will provide an overview of derivative-based algorithms, while Section 4.4 will present derivative-free methods. Finally, Section 4.5 will address the topic of multiobjective metrics.

4.1 Fundamentals of multiobjective optimization

In multiobjective optimization, due to the conflicting objective function components, the optimal solution can no longer be represented by a single point. Instead, it is represented by a set of points known as the Pareto front. Any point on the Pareto front is considered to be an efficient or Pareto optimal solution. A point is called efficient if it is not possible to improve one objective function component, without deteriorating the value of at least another one.

Definition 4.1. *A point $\overline{x} \in X$ is called efficient or Pareto optimal for Problem (4.1) if and only if there exists no point $x \in X$, such that $F(x) \leq F(\overline{x})$, and $F(x) \neq F(\overline{x})$.*

In equivalent terms, an efficient point is not dominated by any other point, so it is called nondominated. In practical applications of multiobjective optimization, finding an efficient solution can be computationally expensive, if not impossible. Therefore, some algorithms aim to provide solutions that are weakly efficient.

Definition 4.2. *A point $\overline{x} \in X$ is called weakly efficient for Problem (4.1) if and only if there exists no point $x \in X$, such that $F(x) < F(\overline{x})$.*

Every efficient point is also weakly efficient. However, the reverse is not always true.

In multiobjective optimization, similar to the single-objective case, obtaining a global solution can be challenging, computationally expensive, or even impossible. In such cases, it may be more practical to focus on finding and evaluating local solutions as an alternative approach.

Definition 4.3. *A point $\overline{x} \in X$ is called locally (weakly) efficient for Problem (4.1) if and only if there exists a neighborhood $\Omega \subset X$ of $\overline{x}$ such that $\overline{x}$ is (weakly) efficient for the problem $\min_{x \in \Omega} F(x)$.*

4.2 Optimality conditions

In this section, we explore and generalize the optimality conditions for the following unconstrained multiobjective optimization problem,

$$\min_{x \in \mathbb{R}^n} F(x) = \left(f_1(x), ..., f_q(x)\right), \tag{4.2}$$

with $F : \mathbb{R}^n \to \mathbb{R}^q$, $n, q \in \mathbb{N}$, $q \geq 2$, and $f_i : \mathbb{R}^n \to \mathbb{R}$, $i \in \{1, ..., q\}$.

First, let us generalize the concept of descent directions in the context of multiobjective optimization. Next, we will establish a connection between these generalized descent directions and the concept of Pareto criticality.

Definition 4.4. *A vector $d \in \mathbb{R}^n$ is called a descent direction for the function F at $x \in \mathbb{R}^n$ if there exists a scalar $t_0 > 0$ such that $f_i(x + td) < f_i(x)$ for all $t \in (0; t_0]$, and for all $i \in \{1, ..., q\}$.*

The following lemma establishes sufficient and necessary conditions for descent directions in the context of multiobjective optimization.

Lemma 4.1. *Let $f_i : \mathbb{R}^n \to \mathbb{R}$, $i \in \{1, ..., q\}$, be continuously differentiable functions. The vector $d \in \mathbb{R}^n$ is a descent direction for the function F at $x \in \mathbb{R}^n$ if and only if $\nabla f_i(x)^\top d < 0$, for all $i \in \{1, ..., q\}$.*

The concept of optimality conditions in multiobjective optimization is indeed more intricate compared to the single-objective case, as it involves trade-offs and conflicts between multiple objectives. However, it provides a generalization of the optimality conditions from single-objective optimization. Pareto criticality is formalized in the following definition.

Definition 4.5. *Let $f_i : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable functions, for $i \in \{1, ..., q\}$. A point $x^* \in \mathbb{R}^n$ is called Pareto critical for Problem (4.2) if*

$$\forall d \in \mathbb{R}^n, \exists i \in \{1, ..., q\} : \nabla f_i(x^*)^\top d \geq 0.$$

Pareto criticality and descent directions are connected concepts in multiobjective optimization. If $\overline{x}$ is not Pareto critical for Problem (4.2), then there must exist a descent direction for F at $\overline{x}$. On the other hand, there exists no descent direction for F at Pareto critical points. Obviously, the minimizer of each objective function component f_i, $i \in \{1, ..., q\}$, is also a Pareto critical point for Problem (4.2).

Pareto criticality serves as a necessary condition for locally weak efficiency, as stated by the following lemma [21].

Lemma 4.2. *If $\overline{x}$ is locally weakly efficient for Problem (4.2), then it is Pareto critical for this problem.*

The following lemma, originally stated in [18], indicates a criticality measure for multiobjective optimization.

Lemma 4.3. *Let $f_i : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable functions, for $i \in \{1, ..., q\}$. Define*

$$\omega(x) = -\min_{\|d\| \leq 1} \max_{i=1,...,q} \nabla f_i(x)^\top d.$$

The following statements hold:

- *The mapping $x \mapsto \omega(x)$ is continuous;*
- *$\omega(x) \geq 0$ for all $x \in \mathbb{R}^n$;*
- *A point $x^* \in \mathbb{R}^n$ is Pareto critical for Problem (4.2) if and only if $\omega(x^*) = 0$.*

The following lemma, introduced in [33], states some useful characteristics of the function ω. It also indicates the connection between this function and descent directions.

Lemma 4.4. *Let $\overline{x} \in \mathbb{R}^n$ be given, and d_ω be a minimizer for the optimization problem*

$$-\min_{\|d\| \leq 1} \max_{i=1,...,q} \nabla f_i(\overline{x})^\top d,$$

denoted by $\omega(\overline{x})$. There exist scalars $\alpha_i \in [0, 1]$ for $i \in \{1, 2, ..., q\}$ with $\sum_{i=1}^q \alpha_i = 1$ and $\mu \geq 0$, such that

$$d_\omega = -\mu \sum_{i=1}^{q} \alpha_i \nabla f_i(\overline{x})$$

and

$$\omega(\overline{x}) \leq \left\| \sum_{i=1}^{q} \alpha_i \nabla f_i(\overline{x}) \right\|$$

hold.

- *If $\overline{x}$ is not Pareto critical for Problem (4.2), then d_ω is a descent direction, and $\|d_\omega\| = 1$.*
- *If $\overline{x}$ is Pareto critical for Problem (4.2), then $d_\omega = \sum_{i=1}^q \alpha_i \nabla f_i(\overline{x}) = 0$ holds.*

4.3 Derivative-based algorithms

A wide variety of techniques have been established to solve multiobjective optimization problems. Typical approaches to this class of problems include the use of aggregation techniques, that combine the different objectives into a single function, which is then optimized, generating a single point in the Pareto front of the problem [15, 23]. Regarding these techniques, if the goal is to compute an approximation to the complete Pareto front, care must be taken in the selection of an adequate scalarization technique, since the Pareto front cannot be simply retrieved with linear combinations of the corresponding objectives [11, 12]. Recently, some algorithms based on SQP techniques have also been proposed for this task [1, 19] which use derivative information of functions.

In the literature, there are several methods that just attempt to compute a single Pareto point. These techniques include Steepest Descent [18], Newton method [17], Quasi-Newton approaches [26], among many others, which are applicable to derivative-based problems. Often, in numerical experiments, algorithms are run from different initializations in an attempt to generate different points in the Pareto front. However, there is no guarantee of success and the algorithms do not incorporate any explicit mechanism for that purpose.

In [29], an algorithm based on trust-region methods for solving multiobjective optimization problems is proposed, assuming the positive definiteness of the Hessians of the models. In [5], the authors present a trust-region globalization strategy for nonconvex unconstrained multiobjective optimization problems, which is a generalization of the algorithm proposed by [17] for convex problems. An additional set of linear inequality constraints is considered in the scalarization problem to address the nonconvex case. Recent work is a trust-region algorithm based on a nonmonotone technique [30], where the extra set of linear inequality constraints of [5] is also used. However, all algorithms presented in the mentioned papers [5, 17, 29, 30] are designed to obtain a single stationary point of the problem.

In Chapter 5, a strategy based on trust-region methods will be presented to generate an approximation to the complete Pareto front of a general derivative-based multiobjective optimization problem [24]. Neither additional assumptions nor any extra conditions are considered by this method. Convergence analysis will be stated. Additionally, it will be indicated in the numerical results that this technique is numerically competitive.

4.4 Derivative-free algorithms

Multiobjective derivative-free optimization is a challenging area, where objectives are black-box functions, which are not given analytically and are the result of some time-consuming experiments. In this scenario, the evaluation of these functions is numerically expensive. Their derivatives would not exist and could be impossible to be numerically

approximated. The use of derivative-based techniques is not applicable and derivative-free methods are required.

A common approach to handle multiobjective derivative-free optimization problems is based on direct search techniques [8, 9]. In this approach, only function values are required. The direct multisearch framework (DMS) [9] is a well-known class of multiobjective optimization methods. Polynomial interpolation models were already incorporated in this multiobjective framework under the denomination of BoostDMS [4]. A recent work in this class is based on a mesh adaptive direct search approach, called DMultiMADS [3]. Implicit filtering methods have also been generalized for multiobjective derivative-free optimization [6]. However, trust-region methods have not yet been properly addressed.

Although trust-region algorithms with a well-established convergence analysis have already been proposed for multiobjective derivative-based optimization [5, 29, 30], this approach has not yet been properly generalized to multiobjective derivative-free optimization problems. A trust-region algorithm is proposed in [31] for biobjective derivative-free problems, which approximates the Pareto front. However, the extension to a general number of objectives is not straightforward. A trust-region algorithm is also presented for multiobjective heterogeneous optimization in which only one of the objective function components is expensive, without derivative information [33]. All other objective function components are given analytically, where derivatives can easily be computed. The algorithm presented in [33] is designed to obtain a single Pareto stationary point for a multiobjective problem.

In Chapter 6, a modified technique based on trust-region methods will be proposed to generate an approximation to the complete Pareto front of a general derivative-free multiobjective optimization problem. This algorithm is an adaptation of the derivative-based multiobjective method. The proposed algorithm uses a technique based on quadratic polynomial interpolation or minimum Frobenius norm approaches to build models that approximate the objective function components. Convergence analysis will be established. Furthermore, the numerical results will indicate that the algorithm is numerically competitive against other derivative-free multiobjective optimization methods.

4.5 Multiobjective metrics

To compare different algorithms, we consider the performance profiles proposed by Dolan and Moré [14], which allow assessing the numerical performance of different solvers, considering different metrics. The performance of solver $s \in S$ on a given set of problems P is represented by a cumulative function

$$\rho_s(\tau) = \frac{1}{|P|} \left| \{ p \in P : r_{p,s} \leq \tau \} \right|,$$

where $\tau \geq 1$ and the performance ratio is defined by

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,\bar{s}} : \bar{s} \in S\}}.$$

Here $t_{p,s}$ represents the value of the selected metric, obtained by solver $s \in S$ when solving problem $p \in P$. Larger values of $\rho_s(\tau)$ indicate a better numerical performance of solver s. In particular, the solver with the largest value of $\rho_s(1)$ is the most efficient. Indeed, the value of $\rho_s(1)$ indicates the probability of the solver s winning over the remaining solvers. On the other hand, the solver with the largest value of $\rho_s(\tau)$ for large values of τ is the most robust.

Selecting a single metric to compare the performance of multiobjective optimization solvers isn't always fair. To have a comprehensive evaluation, considering that a good multiobjective optimization solver should be able to generate a large percentage of non-dominated points and should also be able to capture the extent of the Pareto front of the multiobjective optimization problem, we decided to consider four metrics that attempt to quantify these features, namely purity, hypervolume, and spread metrics Γ and Δ.

Purity measures the percentage of nondominated points generated by a given solver

$$\bar{t}_{p,s} = Pur_{p,s} = \frac{|F_{p,s} \cap F_p|}{|F_{p,s}|},$$

where $F_{p,s}$ represents the approximation to the Pareto front of problem p computed by solver s and F_p is a reference Pareto front for problem p, computed by considering the union of the Pareto approximations corresponding to all solvers, $\cup_{s \in S} F_{p,s}$, and discarding from it all the dominated points [9].

Hypervolume [35], in addition to nondominance, attempts to capture spread, by measuring the volume of the region dominated by the current approximation to the Pareto front and a reference point $U_p \in \mathbb{R}^q$, that is dominated by all points belonging to the different approximations computed for the Pareto front of problem $p \in P$ by all solvers tested. Mathematically, it can be formalized as

$$\bar{t}_{p,s} = HV_{p,s} = Vol\{y \in \mathbb{R}^q | \, y \leq U_p \wedge \exists x \in F_{p,s} : x \leq y\} = Vol\left(\bigcup_{x \in F_{p,s}} [x, U_p] \right),$$

where $Vol(.)$ denotes the Lebesgue measure of a q-dimensional set of points and $[x, U_p]$ denotes the interval box with lower corner x and upper corner U_p.

To compute the performance profiles for purity and hypervolume metrics, since larger values indicate better performance, the inverse value of each one of the metrics is used ($t_{p,s} = 1/\bar{t}_{p,s}$).

Finally, to directly assess the spread across the Pareto front, two additional metrics were considered: the Γ metric, that measures the size of the largest gap in the approximation to the Pareto front computed, and the Δ metric, that assesses how uniformly the nondominated points are distributed along the approximation generated. In a simplified way, consider that solver $s \in S$ has computed, for problem $p \in P$, an approximated Pareto front with points $y_1, y_2, \ldots, y_N$, to which we add the so-called extreme points, y_0 and y_{N+1}, corresponding to the points with the best and worst values for each objective function

component. Then

$$\Gamma_{p,s} = \max_{j\in\{1,\ldots,q\}} \left(\max_{i\in\{0,\ldots,N\}} \{\delta_{j,i}\} \right), \tag{4.3}$$

where $\delta_{j,i} = f_j(y_{i+1}) - f_j(y_i)$, assuming that the objective function values have been sorted by increasing order for each objective function component j. The metric Δ [13] is computed by

$$\Delta_{p,s} = \max_{j\in\{1,\ldots,q\}} \left(\frac{\delta_{j,0} + \delta_{j,N} + \sum_{i=1}^{N-1} |\delta_{j,i} - \bar{\delta}_j|}{\delta_{j,0} + \delta_{j,N} + (N-1)\bar{\delta}_j} \right), \tag{4.4}$$

where $\bar{\delta}_j$, for $j = 1, \ldots, q$, indicates the average of the distances $\delta_{j,i}$, $i = 1, \ldots, N-1$.

5

A general framework for multiobjective optimization

In this chapter, we address the multiobjective optimization problem defined as

$$\begin{aligned} \min F(x) &= \big(f_1(x), \ldots, f_q(x)\big) \\ \text{s.t. } x &\in \mathbb{R}^n, \end{aligned} \tag{5.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^q$, $n, q \in \mathbb{N}$, and $q \geq 2$. The objective function components $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, q$, are assumed to be twice continuously differentiable, with available gradients and Hessians, and conflicting with each other, meaning that it is not possible to find a single point that simultaneously minimizes all objective function components. The problem solution will be the so-called Pareto front of the problem, namely a set of efficient or nondominated points.

In Section 5.1, we will present an algorithm for solving the multiobjective optimization problem by approximating the Pareto front of Problem (5.1). We will then conduct a convergence analysis of this algorithm in Section 5.2. By Section 5.3, we will provide the numerical results demonstrating the efficiency and robustness of this algorithm.

The primary goal of this thesis addresses multiobjective derivative-free optimization. Our intention was to develop an algorithm capable of generating approximations to the complete Pareto front of multiobjective derivative-free optimization problems. During the initial formulation of the algorithm, we opted to assess its performance using Taylor models due to their highly precise approximations of the true objective function components. Once we established confidence in the foundational algorithmic structure, we proceeded to modify it for derivative-free problems by incorporating quadratic polynomial interpolation or minimum Frobenius norm models. Remarkably, the algorithm demonstrated impressive performance in resolving problems that rely on derivatives, leading to a publication [24]. The derivative-based algorithm is comprehensively presented in this chapter.

5.1 Algorithmic structure

Two key goals should be considered when developing an algorithm to approximate the Pareto front of a multiobjective optimization problem:

- The algorithm should aim to capture the extent of the Pareto front by being able to approximate its extreme points. As previously mentioned, these extreme points correspond to the individual minimization of each component of the objective function.
- The algorithm should focus on the density of the Pareto front. This involves the ability of the algorithm to fill in the gaps between points in the computed approximation, resulting in a more comprehensive representation of the Pareto front.

The proposed Multiobjective Trust-Region (MOTR) algorithm achieves these goals through two distinguished steps: the *extreme point step* for trying to capture the extent of the Pareto front and the *scalarization step* for enhancing the density of the approximation [24].

Within the scalarization step, an additional step called the *middle point step* is utilized to determine the initial points at which scalarization problems will be solved. It will be demonstrated through numerical results that this intermediate step, which was developed for the first time in the context of MOTR, plays an important role in the algorithm's process.

MOTR keeps a list of nondominated points and associated quantities, defined as

$$L = \{(x^j, F(x^j), \Delta_{ep}^j, \Delta_{sc}^j) \mid j \in J\}, \tag{5.2}$$

where $J \subset \mathbb{N}$ is the set of indexes of the points in the list, Δ_{ep}^j is a $q \times 1$ vector storing at each component the trust-region radius associated with x^j and the corresponding component of the objective function, to be used at the extreme point step, and Δ_{sc}^j represents the trust-region radius to be used at the scalarization step.

Each time that a new point is added to the list, all dominated points are removed from it. Through this thesis, in a clear abuse of notation but to enhance the clarity and ease of presentation, it will often be stated $x^j \in L$, meaning that $(x^j, F(x^j), \Delta_{ep}^j, \Delta_{sc}^j) \in L$.

In any of the two main steps, points are selected from this list and quadratic Taylor models are built centered at the selected points, to replace the components of the objective function. For $i = 1, \ldots, q$, the quadratic model m_i approximating f_i around a given point x_{step} is defined as follows,

$$m_i(x) = f_i(x_{step}) + \nabla f_i(x_{step})^\top (x - x_{step}) + \frac{1}{2}(x - x_{step})^\top \nabla^2 f_i(x_{step})(x - x_{step}), \tag{5.3}$$

where $\nabla f_i(x_{step})$ and $\nabla^2 f_i(x_{step})$ represent the gradient vector and the Hessian matrix of f_i computed at x_{step}, respectively.

Algorithm 4 formalizes the main procedure, where the extreme point and scalarization steps are executed in alternating iterations.

Sections 5.1.1 and 5.1.2 detail the extreme point and the scalarization steps, respectively.

Algorithm 4. MOTR

Input
Initial list of nondominated points L_0 defined by (5.2).

For $k = 0, 1, 2, \ldots$
If $mod(k, 2) = 0$, then go to the *Extreme Point Step*.
Else go to the *Scalarization Step*.
If some stopping criterion is met, then return.
End For

5.1.1 Extreme point step

The main goal of the extreme point step is to expand the approximation to the Pareto front by moving towards the extreme points of it, corresponding to the individual minimization of each objective function component.

At each iteration k associated with the extreme point step, for each component of the objective function f_i, $i = 1, ..., q$, the point $x_{ep}^{i,k}$, corresponding to the minimum value of f_i for the points in the list, is selected. Ties are broken by the largest extreme point trust-region radius corresponding to f_i, encouraging successful iterations by favoring larger trust-region radii, which indicate points that have not been selected or have been successful in their exploration so far.

Once that $x_{ep}^{i,k}$ is selected, the extreme point trust-region radius corresponding to f_i is set equal to zero for all the other points in the list. This strategy causes the other points in the list to no longer be candidates for the extreme point step corresponding to the selected objective function component.

The quadratic Taylor model m_i^k (5.3), centered at $x_{ep}^{i,k}$, is computed and then one iteration of a single objective trust-region algorithm is performed. The model is minimized in $B(x_{ep}^{i,k}, \Delta_{ep}^{i,k}(i))$, the closed ball centered at $x_{ep}^{i,k}$ with radius $\Delta_{ep}^{i,k}(i)$, to compute the minimizer $x_{ep}^{i,k*}$.

Then, $\rho_{ep}^{i,k}$ is computed, denoting the ratio of agreement between the decrease obtained in the model m_i^k and the variation obtained in the corresponding objective function component. Based on the value of the ratio $\rho_{ep}^{i,k}$, the decision to accept or reject the model minimizer $x_{ep}^{i,k*}$, as well as the update strategy for the trust-region radius, are determined according to the identical rules applied in single-objective trust-region methods.

Having $m_i^k(x_{ep}^{i,k}) - m_i^k(x_{ep}^{i,k*}) = 0$ means that the current model center $x_{ep}^{i,k}$ is the model minimizer. In this situation, $\rho_{ep}^{i,k}$ is set equal to zero, which results in the decrease of the corresponding trust-region radius. Consequently, this will lead to an increase in the agreement between the Taylor model and the objective function component.

Otherwise, $m_i^k(x_{ep}^{i,k}) - m_i^k(x_{ep}^{i,k*}) > 0$ and we set

$$\rho_{ep}^{i,k} = \frac{f_i(x_{ep}^{i,k}) - f_i(x_{ep}^{i,k*})}{m_i^k(x_{ep}^{i,k}) - m_i^k(x_{ep}^{i,k*})}.$$

A high value of $\rho_{ep}^{i,k}$ indicates that the model is adequately predicting the reduction in the function value. In such cases, the model minimizer will be accepted and the trust-region radius will be increased, considering a successful iteration. Consequently, the point $x_{ep}^{i,k*}$ is added to the list, and all dominated points are removed from it. In fact, when $\rho_{ep}^{i,k} > 0$, the value of the objective function $f_i(x_{ep}^{i,k*})$ is smaller than $f_i(x_{ep}^{i,k})$. Thus, it is clear that $x_{ep}^{i,k*}$ is nondominated by all points in the list L_k.

In successful iterations, there is the possibility of $x_{ep}^{i,k}$ continuing to be a nondominated point and still remaining in the list. In this situation, it will no longer be a candidate in any extreme point step corresponding to the objective function component f_i. Consequently, $\Delta_{ep}^{i,k}(i)$ is set equal to zero.

If $\rho_{ep}^{i,k}$ is low, then $x_{ep}^{i,k*}$ will not be accepted and the trust-region radius $\Delta_{ep}^{i,k}(i)$ will be decreased, considering an unsuccessful iteration. By aiming to enhance the quality of the Taylor model, as an approximation to the real function component, this approach is undertaken.

The extreme point step is outlined within Algorithm 5.

5.1.2 Scalarization step

In the scalarization step, there is an attempt to fill the gaps in the current approximation to the Pareto front. An initial point associated with the largest gap, corresponding to the objective function component under analysis, is selected. Then, an appropriate scalarization problem is solved and the corresponding minimizer is added to the list if it is nondominated.

In each scalarization step iteration k, for each component of the objective function, f_i, $i = 1, ..., q$, an initial point $x_{sc}^{i,k}$ is selected or computed. Once that $x_{sc}^{i,k}$ is obtained, models are built for each objective function component, centered at $x_{sc}^{i,k}$. A joint minimization of the models is performed, by solving the following scalarization problem, computing the new point $x_{sc}^{i,k*}$:

$$\begin{aligned} \min \quad & t \\ \text{s.t.} \quad & m_l^k(x) - m_l^k(x_{sc}^{i,k}) \leq t, \ l = 1, ..., q, \\ & x \in B(x_{sc}^{i,k}, \Delta_{sc}^{i,k}), \\ & t \in \mathbb{R}. \end{aligned} \tag{5.4}$$

A similar scalarization approach was considered in [5] and [29]. In the first case, it was incorporated with an extra set of linear inequality constraints. The algorithm proposed by [29] requires positive definite Hessians for the theoretical analysis. In both cases, the algorithms aim to generate a single point as a solution, without explicitly attempting to compute any approximation of the complete Pareto front of the problem.

Algorithm 5. Extreme point step

Input

Current list of nondominated points L_k, defined by (5.2).
The minimum trust-region radius value Δ_{ep}^{min}.
Values for the parameters $0 < \eta_{ep}^1 \leq \eta_{ep}^2 < 1$ and $0 < \mu_1 < 1 < \mu_2$.

$L = L_k$
For $i = 1, ..., q$

1. Selection of an iterate point

Select $(x_{ep}^{i,k}, F(x_{ep}^{i,k}), \Delta_{ep}^{i,k}, \Delta_{sc}^{i,k}) \in L$ such that $x_{ep}^{i,k} \in \arg\min_{x \in L} f_i(x)$. Break ties by selecting the point with the maximum extreme point trust-region radius corresponding to f_i.
For all points x^j in L, except the selected one, set $\Delta_{ep}^{j}(i) = 0$.
If $\Delta_{ep}^{i,k}(i) < \Delta_{ep}^{min}$, stop the procedure and continue to the next i.

2. Step calculation

Compute the model function m_i^k, centered at $x_{ep}^{i,k}$, and $x_{ep}^{i,k*} \in \arg\min_{x \in B(x_{ep}^{i,k}, \Delta_{ep}^{i,k}(i))} m_i^k(x)$.
If $m_i^k(x_{ep}^{i,k}) - m_i^k(x_{ep}^{i,k*}) = 0$ then set $\rho_{ep}^{i,k} = 0$. Else, set $\rho_{ep}^{i,k} = \frac{f_i(x_{ep}^{i,k}) - f_i(x_{ep}^{i,k*})}{m_i^k(x_{ep}^{i,k}) - m_i^k(x_{ep}^{i,k*})}$.

3. Trial point acceptance and trust-region update

If $\rho_{ep}^{i,k} \geq \eta_{ep}^1$ then:
Set $\Delta_{ep}^{i,k*} = \Delta_{ep}^{i,k}$ and $\Delta_{sc}^{i,k*} = \Delta_{sc}^{i,k}$.
If $\rho_{ep}^{i,k} \geq \eta_{ep}^2$ then set $\Delta_{ep}^{i,k*}(i) = \mu_2 * \Delta_{ep}^{i,k}(i)$.
Set $\Delta_{ep}^{i,k}(i) = 0$.
Add the new point to the list, by setting $L = L \cup \{(x_{ep}^{i,k*}, F(x_{ep}^{i,k*}), \Delta_{ep}^{i,k*}, \Delta_{sc}^{i,k*})\}$ and delete the dominated points from it.
Else, set $\Delta_{ep}^{i,k}(i) = \mu_1 * \Delta_{ep}^{i,k}(i)$.

End For
$L_{k+1} = L$

After computing the new point in the scalarization step, it is evaluated to determine if it should be accepted or rejected. The trust-region radius associated with the scalarization step is then updated accordingly. Algorithm 6 provides the details of the scalarization step.

5.1.2.1 Initial point at the scalarization step

The procedure for selecting or computing an initial point at the scalarization step depends on the number of nondominated points in the list with scalarization step trust-region radii greater than or equal to the minimum associated value Δ_{sc}^{min}. In scalarization step iteration k, for each objective function f_i, $i = 1, ..., q$, one of the following steps are taken:

- If there are no points in the list that satisfy the minimum trust-region radius criterion,

Algorithm 6. Scalarization step

Input

Current list of nondominated points L_k, defined by (5.2).
The minimum trust-region radius value Δ_{sc}^{min}.
Values for the parameters $0 < \eta_{sc}^1 \leq \eta_{sc}^2 < 1$ and $0 < \mu_1 < 1 < \mu_2$.

$L = L_k$
For $i = 1, ..., q$

1. Selection of an iterate point

Compute $p = |\{x \in L : \Delta_{sc} \geq \Delta_{sc}^{min}\}|$.
If $p = 0$ then stop the procedure and move on to the next extreme point step.
If $p = 1$ then set $x_{sc}^{i,k}$ equal to the point in L satisfying $\Delta_{sc} \geq \Delta_{sc}^{min}$.
If $p \geq 2$ then go to the *middle point step* to compute $x_{sc}^{i,k}$.

2. Step calculation

Compute the model functions m_l^k, $l = 1, ..., q$, centered at $x_{sc}^{i,k}$, and $x_{sc}^{i,k*}$ by solving the scalarization Problem (5.4).
Define $\phi(x) = \max_{j=1,...,q} f_j(x)$ and $\phi_m^k(x) = \max_{j=1,...,q} m_j^k(x)$.
If $\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) = 0$ then set $\rho_{sc}^{i,k} = 0$. Else compute $\rho_{sc}^{i,k} = \frac{\phi(x_{sc}^{i,k}) - \phi(x_{sc}^{i,k*})}{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*})}$.

3. Trial point acceptance and trust-region update

If $\rho_{sc}^{i,k} \geq \eta_{sc}^1$ and $x_{sc}^{i,k*}$ is nondominated then:
If $\rho_{sc}^{i,k} \geq \eta_{sc}^2$ then set $\Delta_{sc}^{i,k} = \mu_2 * \Delta_{sc}^{i,k}$.
Set $\Delta_{ep}^{i,k*} = \Delta_{ep}^{i,k}$ and $\Delta_{sc}^{i,k*} = \Delta_{sc}^{i,k}$.
Add the new point to the list, by setting $L = L \cup \{(x_{sc}^{i,k*}, F(x_{sc}^{i,k*}), \Delta_{ep}^{i,k*}, \Delta_{sc}^{i,k*})\}$ and delete the dominated points from it.
Else, set $\Delta_{sc}^{i,k} = \mu_1 * \Delta_{sc}^{i,k}$.

End For
$L_{k+1} = L$

the scalarization step is discarded, and we move on to the next extreme point step;

- If there is only one point in the list that meets the minimum trust-region radius criterion, that point is selected as the initial point $x_{sc}^{i,k}$;
- If there are two or more points in the list that satisfy the minimum trust-region radius criterion, the point $x_{sc}^{i,k}$ is computed using a procedure called the middle point step, described in Algorithm 7.

The middle point step is a straightforward and efficient procedure. First, all points in the list L are sorted based on their f_i values. Then, the gaps between consecutive points are computed and sorted. In cases where there are ties between gaps, priority is given to gaps associated with a larger scalarization step trust-region radius. This prioritization promotes the progress and success of the algorithm.

Starting from the largest gap, if at least one of the two consecutive points defining the current gap satisfies the minimum trust-region radius criterion, the middle point between these two points is computed in the variable space. However, if none of these two consecutive points satisfies the minimum trust-region radius criterion, the algorithm proceeds to the next gap.

If the computed middle point already exists in the list and satisfies the minimum trust-region radius criterion, it is selected as the point $x_{sc}^{i,k}$, and the algorithm returns to the scalarization step.

On the other hand, if the computed middle point does not already belong to the list and is nondominated, it is added to the list as the point $x_{sc}^{i,k}$, and dominated points are deleted from the list. The algorithm then returns to the scalarization step.

If none of these conditions occur for any of the gaps, the algorithm proceeds to the next objective function component, and the scalarization step for the current objective function component is discarded.

To the best of our knowledge, this strategy has not been previously presented in the literature of multiobjective or single-objective optimization. As it will be illustrated in the numerical results, reported in Section 5.3, this strategy stands out as one of the key features of MOTR.

5.1.2.2 New point acceptance and trust-region updates at the scalarization step

The criterion for accepting $x_{sc}^{i,k*}$ and updating the scalarization step trust-region radius requires computing the ratio $\rho_{sc}^{i,k}$. For this purpose, the auxiliary functions $\phi(x)$ and $\phi_m^k(x)$ are defined by

$$\phi(x) = \max_{j=1,\ldots,q} f_j(x),$$

and

$$\phi_m^k(x) = \max_{j=1,\ldots,q} m_j^k(x).$$

When $\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*})$ is equal to zero, $\rho_{sc}^{i,k}$ is set equal to zero. This forces a decrease in the trust-region radius, leading to an increase in the agreement between the models and the objective function components. Otherwise, $\rho_{sc}^{i,k}$ is computed by

$$\rho_{sc}^{i,k} = \frac{\phi(x_{sc}^{i,k}) - \phi(x_{sc}^{i,k*})}{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*})}.$$

In [33] it is established that when $\rho_{sc}^{i,k} > 0$, descent is guaranteed for at least one component of the objective function.

After computing the value of $\rho_{sc}^{i,k}$, the strategy for accepting or rejecting the new point and updating the trust-region radius follows the standard approach used in any trust-region method. The key difference is that the new point must be nondominated with respect to the list of existing nondominated points. If $\rho_{sc}^{i,k}$ is sufficiently large, but the new point is dominated, the trust-region radius is reduced to improve the quality of the models' approximation.

Algorithm 7. Middle point step

Input

Current list of nondominated points L, defined by (5.2).
Current objective function component $i \in \{1, ..., q\}$.
The minimum trust-region radius value Δ_{sc}^{min}.
The initial trust-region radius values Δ_{ep}^{init} and Δ_{sc}^{init}.

1. Compute and sort the gaps

Sort $\{f_i(x) \mid x \in L\}$ by increasing value.
Compute the gaps between consecutive values of the sorted f_i
and order them by decreasing value.
Break ties according to the largest scalarization step trust-region radius,
corresponding to the points associated with the gaps.

For all gaps, starting from the largest one

2. Compute the middle point

If for at least one point of the pair associated with the gap, the corresponding trust-region radius associated with the scalarization step satisfies $\Delta_{sc} \geq \Delta_{sc}^{min}$ then:
Compute x_{middle}, the middle point of the pair in the variable space.
Else, continue to the next gap.

3. Test the middle point

If $x_{middle} \in L$, with the corresponding trust-region radius associated to the scalarization step satisfying $\Delta_{sc} \geq \Delta_{sc}^{min}$ then:
Set $x_{sc}^{i,k} = x_{middle}$ and return.
Else if $x_{middle} \notin L$ and is nondominated then:
Set $x_{sc}^{i,k} = x_{middle}$,
add it to the list by setting $L = L \cup \{(x_{sc}^{i,k}, F(x_{sc}^{i,k}), \Delta_{ep}^{init}, \Delta_{sc}^{init})\}$,
delete the dominated points from the list, and return.
Else, continue to the next gap.

End For

5.2 Convergence analysis

To analyze the theoretical behavior of MOTR, we will consider that no stopping criteria are defined. In particular, $\Delta_{ep}^{min} = \Delta_{sc}^{min} = 0$. Convergence will be established for linked sequences of points $\{x^k\}_{k \in K}$ generated by the algorithm.

Definition 5.1. *Consider $\{L_k\}_{k \in \mathbb{N}}$ as the sequence of sets of nondominated points generated by Algorithm 4. A linked sequence is a sequence $\{x^k\}_{k \in K}$, where $K \subseteq \mathbb{N}$ denotes the indexes of the points belonging to the linked sequence, such that for any $k \in K$, the element $(x^k, F(x^k), \Delta_{ep}^k, \Delta_{sc}^k) \in L_k$ is generated from the element $(x^{k-1}, F(x^{k-1}), \Delta_{ep}^{k-1}, \Delta_{sc}^{k-1}) \in L_{k-1}$.*

An initial point of a linked sequence can be one of the following:

- A point in the initial list, provided by the user;

- Middle points generated by MOTR in the middle point step, such that these points are not currently in the list and are added to the list by the algorithm.

Every time that a new middle point is generated a linked sequence is initialized. We are going to establish that every linked sequence generated by MOTR converges to a Pareto critical point, a necessary condition for being a solution of Problem (5.1), formalized in Definition 4.5.

The following lemma provides a criticality measure for multiobjective optimization. This lemma was also presented in Section 4.2. Due to its importance in our theoretical analysis and to maintain the coherence of this chapter, we present it again here.

Lemma 5.1. *Let $f_i : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable functions, for $i \in \{1, ..., q\}$. Define*

$$\omega(x) = -\min_{\|d\| \leq 1} \max_{i=1,...,q} \nabla f_i(x)^\top d. \tag{5.5}$$

The following statements hold:

- *The mapping $x \mapsto \omega(x)$ is continuous;*
- *$\omega(x) \geq 0$ for all $x \in \mathbb{R}^n$;*
- *A point $x^* \in \mathbb{R}^n$ is Pareto critical if and only if $\omega(x^*) = 0$.*

Under reasonable assumptions, we are going to establish that for every linked sequence $\{x^k\}_{k \in K}$ generated by MOTR,

$$\lim_{k \to +\infty ; k \in K} \omega(x^k) = 0.$$

For each $i \in \{1, ..., q\}$, we assume that the objective function component f_i is lower bounded and twice continuously differentiable. Consequently, function $\phi(x) = \max_{i=1,...,q} f_i(x)$ is also lower bounded.

Assumption 5.1. *For $i \in \{1, ..., q\}$, the Hessian matrix of the objective function component f_i is uniformly bounded, meaning that there is a constant $\kappa_h > 0$ such that*

$$\|\nabla^2 f_i(x)\| \leq \kappa_h$$

for all $x \in \mathbb{R}^n$ and for all $i \in \{1, ..., q\}$.

Assumption 5.1 implies that ∇f_i is Lipschitz continuous, for each $i \in \{1, ..., q\}$. From it, we can deduce that function ω, defined by (5.5), is uniformly continuous [34].

At each iteration k, for $i \in \{1, ..., q\}$, model m_i^k, centered at x^k, is a quadratic Taylor model, defined by (5.3), again twice continuously differentiable and satisfying:

$$m_i^k(x^k) = f_i(x^k),$$

$$\nabla m_i^k(x^k) = \nabla f_i(x^k),$$

$$\nabla^2 m_i^k(x^k) = \nabla^2 f_i(x^k).$$

In this chapter, B_k denotes the current iteration ball, defined as follows

$$B_k = B(x^k, \Delta_k) = \left\{x \in \mathbb{R}^n \mid \|x - x^k\| \leq \Delta_k\right\},$$

where x^k is the current iterate and Δ_k represents the current trust-region radius.

Assumption 5.1 allows us to establish the well-known error bounds for Taylor models.

Lemma 5.2. *Let Assumption 5.1 hold. At every iteration k, the model m_i^k is valid for f_i in B_k, for all $i \in \{1, ..., q\}$, that is, there exists a constant $\kappa_{fm} > 0$ such that*

$$|f_i(x) - m_i^k(x)| \leq \kappa_{fm}\Delta_k^2$$

holds for all $x \in B_k$.

Function ω, defined by equation (5.5), can be generalized to models through

$$\omega_{m^k}(x) = -\min_{\|d\|\leq 1} \max_{i=1,...,q} \nabla m_i^k(x)^\top d.$$

The use of quadratic Taylor models guarantees that $\omega_{m^k}(x^k) = \omega(x^k)$, at each iteration k, where x^k represents the point where the model was built. This equality ensures that when the iteration point x^k is Pareto critical or close to criticality for the model, the same applies to the objective function.

To prove convergence, model minimization should provide a sufficient decrease at each iteration. Following [34], for the scalarization step, we quantify the best model reduction obtained along a direction belonging to $\mathcal{D}(x)$, the set of directions associated with the solution of (5.5), within the trust-region B_k. For $i \in \{1, ..., q\}$, let $d_k^* \in \mathcal{D}(x_{sc}^{i,k})$ and compute α_k by solving

$$\min_{\alpha\geq 0}\{\phi_m^k(x_{sc}^{i,k} + \alpha d_k^*) : x_{sc}^{i,k} + \alpha d_k^* \in B_k\}. \tag{5.6}$$

The Pareto-Cauchy point is defined as

$$x_k^C = x_{sc}^{i,k} + d_k^C, \tag{5.7}$$

where $d_k^C := \alpha_k d_k^*$ and $B_k = B(x_{sc}^{i,k}, \Delta_{sc}^{i,k})$.

Lemma 5.3. *Let Assumption 5.1 hold. For $i \in \{1, ..., q\}$ and $k \in \mathbb{N}$, the Pareto-Cauchy point $x_k^C = x_{sc}^{i,k} + d_k^C$, defined by (5.7), satisfies*

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C) \geq \frac{1}{2}\omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}. \tag{5.8}$$

Proof. Since $d_k^* \in \mathcal{D}(x_{sc}^{i,k})$, we have $\|d_k^*\| \leq 1$, which implies that, for all $\alpha \in [0, \Delta_{sc}^{i,k}]$, $x_{sc}^{i,k} + \alpha d_k^* \in B_k$. Problem (5.6) has the same solution as

$$\max_{0\leq\alpha\leq\Delta_{sc}^{i,k}}\{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*)\}.$$

On the other hand, for all $\alpha \geq 0$, we have

$$\begin{aligned}\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*) &= \max_{j=1,\dots,q} m_j^k(x_{sc}^{i,k}) - \max_{j=1,\dots,q} m_j^k(x_{sc}^{i,k} + \alpha d_k^*) \\ &\geq - \max_{j=1,\dots,q} \alpha \nabla m_j^{k\top}(x_{sc}^{i,k}) d_k^* - \max_{j=1,\dots,q} \frac{1}{2}\alpha^2 d_k^{*\top} \nabla^2 m_j^k(x_{sc}^{i,k}) d_k^*.\end{aligned}$$

According to (5.5), Assumption 5.1, $\|d_k^*\| \leq 1$, and the Cauchy-Schwarz inequality we have

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*) \geq \alpha\omega(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h.$$

Then, it is clear that

$$\max_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*)\} \geq \max_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \left\{\alpha\omega(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h\right\}.$$

In other words,

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + d_k^C) \geq \max_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \left\{\alpha\omega(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h\right\}.$$

Let us consider the concave function g, defined by $g(\alpha) = \alpha\omega(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h$, with unconstrained maximizer $\alpha^* = \frac{\omega(x_{sc}^{i,k})}{\kappa_h} \geq 0$, corresponding to the optimum value $g(\alpha^*) = \frac{1}{2}\frac{\omega(x_{sc}^{i,k})^2}{\kappa_h} \geq 0$. Two cases can occur:

- If $0 \leq \alpha^* \leq \Delta_{sc}^{i,k}$ then $\max_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \left\{\alpha\omega(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h\right\} = \frac{1}{2}\frac{\omega(x_{sc}^{i,k})^2}{\kappa_h}$;
- If $\alpha^* > \Delta_{sc}^{i,k}$ then $\max_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \left\{\alpha\omega(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h\right\} = \Delta_{sc}^{i,k}\omega(x_{sc}^{i,k}) - \frac{1}{2}(\Delta_{sc}^{i,k})^2\kappa_h$.

In this last case, since $\alpha^* = \frac{\omega(x_{sc}^{i,k})}{\kappa_h} > \Delta_{sc}^{i,k}$, we have $\Delta_{sc}^{i,k}\omega(x_{sc}^{i,k}) - \frac{1}{2}(\Delta_{sc}^{i,k})^2\kappa_h \geq \frac{1}{2}\Delta_{sc}^{i,k}\omega(x_{sc}^{i,k})$, resulting in

$$\begin{aligned}\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C) &\geq \min\left\{\frac{1}{2}\frac{\omega(x_{sc}^{i,k})^2}{\kappa_h}, \frac{1}{2}\Delta_{sc}^{i,k}\omega(x_{sc}^{i,k})\right\} \\ &= \frac{1}{2}\omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}.\end{aligned}$$

□

Let $(x_{sc}^{i,k*}, t^*)$ be the solution of Problem (5.4). The following lemma states an important property of t^*.

Lemma 5.4. *At each iteration k and for each $i \in \{1, \dots, q\}$, $x_{sc}^{i,k}$ is not a Pareto critical point for $\min_{x\in B_k}(m_1^k(x), \dots, m_q^k(x))$, if and only if $t^* < 0$.*

Proof. Since $(x_{sc}^{i,k}, 0)$ is feasible for Problem (5.4), $t^* \leq 0$. It is clear that $x_{sc}^{i,k}$ is not a Pareto critical point when $t^* < 0$.

Now, assume that $x_{sc}^{i,k}$ is not a Pareto critical point. Then, it is not a weakly efficient point, meaning that there exists a point $x^{'} \in B_k$ such that for all $j \in \{1, ..., q\}$, $m_j^k(x^{'}) < m_j^k(x_{sc}^{i,k})$. So,

$$m_j^k(x^{'}) - m_j^k(x_{sc}^{i,k}) < 0, \quad \forall j \in \{1, ..., q\}.$$

Considering $t^{'} = \max_{j=1,...,q}(m_j^k(x^{'}) - m_j^k(x_{sc}^{i,k}))$, we have

$$m_j^k(x^{'}) - m_j^k(x_{sc}^{i,k}) \leq t^{'} < 0, \quad \forall j \in \{1, ..., q\}.$$

Hence, t^* should be strictly negative because $(x^{'}, t^{'})$ is feasible for Problem (5.4). □

Now, we can obtain a more precise understanding of the descent that occurs for ϕ_m^k.

Lemma 5.5. *Let Assumption 5.1 hold. At each iteration k and for each $i \in \{1, ..., q\}$, there exists $j \in \mathbb{N}$ such that*

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \left(\frac{1}{2}\right)^j \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\},$$

where $(x_{sc}^{i,k*}, t^*)$ is the solution of Problem (5.4).

Proof. Two different cases need to be analyzed. Assume that $x_{sc}^{i,k}$ is not Pareto critical. According to Lemma 5.4, t^* is strictly negative. So, for each $l \in \{1, ..., q\}$, we have

$$m_l^k(x_{sc}^{i,k}) - m_l^k(x_{sc}^{i,k*}) \geq -t^* > 0.$$

By considering $\phi_m^k(x) = \max_{l=1,...,q} m_l^k(x)$, it results

$$\phi_m^k(x_{sc}^{i,k}) - m_l^k(x_{sc}^{i,k*}) \geq m_l^k(x_{sc}^{i,k}) - m_l^k(x_{sc}^{i,k*}), \text{ for all } l \in \{1, ..., q\}.$$

Let j be the index such that $\phi_m^k(x_{sc}^{i,k*}) = m_j^k(x_{sc}^{i,k*})$. Then

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq m_j^k(x_{sc}^{i,k}) - m_j^k(x_{sc}^{i,k*}).$$

Hence,

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq -t^* > 0. \tag{5.9}$$

Since $\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C) \geq 0$, there must exist $j \in \mathbb{N}$ such that

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \left(\frac{1}{2}\right)^{j-1} (\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C)).$$

Hence, considering (5.8), it implies

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \left(\frac{1}{2}\right)^j \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}.$$

If $x_{sc}^{i,k}$ is Pareto critical, then $\omega(x_{sc}^{i,k}) = 0$. So, the right side of this inequality is equal to zero, and since $\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq 0$, the inequality holds. □

The last three lemmas motivate us to consider the following assumption, stating that, at each scalarization step, a sufficient reduction in the model space is ensured.

Assumption 5.2. *There is a constant $\kappa_\phi \in (0,1)$ such that at each iteration k, where the scalarization step is performed, for all $i \in \{1,...,q\}$, we have*

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \kappa_\phi \, \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}. \tag{5.10}$$

As long as $x_{sc}^{i,k}$ is not Pareto critical, the left side of (5.10) is strictly positive.

The following lemma plays a key role in establishing convergence and provides an error bound for ϕ_m^k as an approximation of ϕ.

Lemma 5.6. *Let Assumption 5.1 hold. At every iteration k, the model ϕ_m^k is valid for ϕ at $x_{sc}^{i,k*}$, for all $i \in \{1,...,q\}$, that is*

$$|\phi(x_{sc}^{i,k*}) - \phi_m^k(x_{sc}^{i,k*})| \leq \kappa_{fm}(\Delta_{sc}^{i,k})^2,$$

where κ_{fm} is defined in Lemma 5.2.

Proof. Two situations should be analyzed. Assume that $\phi(x_{sc}^{i,k*}) \geq \phi_m^k(x_{sc}^{i,k*})$. Consider $j \in \{1,...,q\}$ such that $\phi(x_{sc}^{i,k*}) = f_j(x_{sc}^{i,k*})$. Using Lemma 5.2 and the fact that $\phi_m^k(x_{sc}^{i,k*}) \geq m_j^k(x_{sc}^{i,k*})$, we have

$$|\phi(x_{sc}^{i,k*}) - \phi_m^k(x_{sc}^{i,k*})| = f_j(x_{sc}^{i,k*}) - \phi_m^k(x_{sc}^{i,k*}) \leq f_j(x_{sc}^{i,k*}) - m_j^k(x_{sc}^{i,k*}) \leq \kappa_{fm}(\Delta_{sc}^{i,k})^2.$$

Assume now that $\phi(x_{sc}^{i,k*}) < \phi_m^k(x_{sc}^{i,k*})$. Consider $j \in \{1,...,q\}$ such that $\phi_m^k(x_{sc}^{i,k*}) = m_j^k(x_{sc}^{i,k*})$. Again, Lemma 5.2 and the fact that $\phi(x_{sc}^{i,k*}) \geq f_j(x_{sc}^{i,k*})$ allow us to conclude that

$$|\phi(x_{sc}^{i,k*}) - \phi_m^k(x_{sc}^{i,k*})| = m_j^k(x_{sc}^{i,k*}) - \phi(x_{sc}^{i,k*}) \leq m_j^k(x_{sc}^{i,k*}) - f_j(x_{sc}^{i,k*}) \leq \kappa_{fm}(\Delta_{sc}^{i,k})^2.$$

□

In the remaining of the analysis, we categorize the successful scalarization step iterations:

- The set of indexes of *successful scalarization step iterations* is denoted by

$$S = \{(k,i),\ k \in \mathbb{N}, i \in \{1,...,q\} : k \text{ is a scalarization step iteration and } \rho_{sc}^{i,k} \geq \eta_{sc}^1\}.$$

- The set of indexes of *very successful scalarization step iterations* corresponds to

$$V = \{(k,i),\ k \in \mathbb{N}, i \in \{1,...,q\} : k \text{ is a scalarization step iteration and } \rho_{sc}^{i,k} \geq \eta_{sc}^2\}.$$

For each linked sequence of points $\{x^k\}_{k\in K}$ generated by MOTR, one of the two different scenarios will occur, as outlined below:

1. There exists $i \in \{1, ..., q\}$, such that for each $k \in \mathbb{N}$,

$$\Delta_{ep}^{i,k} > 0.$$

2. For each $i \in \{1, ..., q\}$, there exists $k_i \in \mathbb{N}$, such that for all $k > k_i$,

$$\Delta_{ep}^{i,k} = 0.$$

Remark 5.1. *In the first scenario, the linked sequence, updated at the extreme point step, matches the set of iterates generated by a single-objective trust-region method, when applied to the objective function component f_i. Stationarity is then guaranteed for f_i and the corresponding limit point is a Pareto critical point. The proof is similar to the single-objective trust-region case (see [7, 20, 27]).*

Remark 5.2. *In the second scenario, define $k_{ep} = \max\{k_i \mid i = 1, ..., q\}$. For $k > k_{ep}$, it holds $\Delta_{ep}^{i,k} = 0,\ \forall i = 1, ..., q$. Therefore, for $k > k_{ep}$, all points of the linked sequence have been generated in the scalarization step. The remaining analysis focuses on this situation.*

The following two lemmas clarify the behavior of MOTR when the current point is not Pareto critical.

Lemma 5.7. *Let Assumptions 5.1 and 5.2 hold. Suppose that at the scalarization step iteration k, for $i \in \{1, ..., q\}$, $x_{sc}^{i,k}$ is not a Pareto critical point and*

$$\Delta_{sc}^{i,k} \leq \frac{\kappa_\phi \omega(x_{sc}^{i,k})(1 - \eta_{sc}^2)}{\kappa_v}, \tag{5.11}$$

with $\kappa_v = \max\{\kappa_{fm}, \kappa_h\}$. Then the pair (k, i) corresponds to a very successful scalarization step iteration, and $\Delta_{sc}^{i,k*} > \Delta_{sc}^{i,k}$.

Proof. According to Lemma 5.1, $\omega(x_{sc}^{i,k}) > 0$, because $x_{sc}^{i,k}$ is not a Pareto critical point. On the other hand, $\kappa_\phi, \eta_{sc}^2 \in (0, 1)$. Thus, $\kappa_\phi(1 - \eta_{sc}^2) \in (0, 1)$ and

$$\Delta_{sc}^{i,k} \leq \frac{\kappa_\phi \omega(x_{sc}^{i,k})(1 - \eta_{sc}^2)}{\kappa_v} < \frac{\omega(x_{sc}^{i,k})}{\kappa_v}.$$

From Assumption 5.2, we have

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \kappa_\phi \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\} = \kappa_\phi \omega(x_{sc}^{i,k}) \Delta_{sc}^{i,k}.$$

Considering this inequality, the equation $\rho_{sc}^{i,k} = \frac{\phi(x_{sc}^{i,k}) - \phi(x_{sc}^{i,k*})}{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*})}$, Lemma 5.6, and (5.11), we have

$$|\rho_{sc}^{i,k} - 1| = \left|\frac{\phi_m^k(x_{sc}^{i,k*}) - \phi(x_{sc}^{i,k*})}{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*})}\right| \leq \frac{\kappa_{fm}}{\kappa_\phi \omega(x_{sc}^{i,k})} \Delta_{sc}^{i,k} \leq \frac{\kappa_v}{\kappa_\phi \omega(x_{sc}^{i,k})} \Delta_{sc}^{i,k} \leq (1 - \eta_{sc}^2).$$

Consequently, $\rho_{sc}^{i,k} \geq \eta_{sc}^2$. So, the pair (k, i) corresponds to a very successful scalarization step iteration and $\Delta_{sc}^{i,k*} > \Delta_{sc}^{i,k}$. □

The following lemma states that, as long as $x_{sc}^{i,k}$ is not Pareto critical, the scalarization step trust-region radius can not be too small. In fact, it should be lower bounded by a strictly positive constant.

Lemma 5.8. *Let Assumptions 5.1 and 5.2 hold, and consider the constant $\sigma > 0$. If $\omega(x_{sc}^{i,k}) \geq \sigma$ holds for the pair (k, i), with k a scalarization step iteration and $i \in \{1, ..., q\}$, then there is a constant $\Delta > 0$, depending on σ, such that $\Delta_{sc}^{i,k} \geq \Delta$.*

Proof. Assume, as a mean of contradiction, that for each $\Delta > 0$ there is a pair (k, i), with k a scalarization step iteration and $i \in \{1, ..., q\}$, satisfying $\omega(x_{sc}^{i,k}) \geq \sigma > 0$, such that

$$\Delta_{sc}^{i,k} < \Delta.$$

In particular, consider

$$\Delta = \frac{\mu_1 \sigma \kappa_\phi (1 - \eta_{sc}^2)}{\kappa_v},$$

with $\kappa_v = \max\{\kappa_{fm}, \kappa_h\}$. Let $(\overline{k}, i)$ be the first pair such that $\omega(x_{sc}^{i,\overline{k}}) \geq \sigma > 0$ and

$$\Delta_{sc}^{i,\overline{k}^*} < \frac{\mu_1 \sigma \kappa_\phi (1 - \eta_{sc}^2)}{\kappa_v}.$$

Then, it holds $\Delta_{sc}^{i,\overline{k}^*} < \Delta_{sc}^{i,\overline{k}}$. Thus,

$$\Delta_{sc}^{i,\overline{k}} = \frac{\Delta_{sc}^{i,\overline{k}^*}}{\mu_1} < \frac{\sigma \kappa_\phi (1 - \eta_{sc}^2)}{\kappa_v} \leq \frac{\omega(x_{sc}^{i,\overline{k}}) \kappa_\phi (1 - \eta_{sc}^2)}{\kappa_v}.$$

Since point $x_{sc}^{i,\overline{k}}$ is not Pareto critical, according to Lemma 5.7, the pair $(\overline{k}, i)$ corresponds to a very successful scalarization step iteration, and $\Delta_{sc}^{i,\overline{k}^*} > \Delta_{sc}^{i,\overline{k}}$. This contradicts $\Delta_{sc}^{i,\overline{k}^*} < \Delta_{sc}^{i,\overline{k}}$ and the initial assumption. □

Considering Remarks 5.1 and 5.2, the following lemma states the first convergence result for linked sequences of MOTR, having only a finite number of successful iterations performed at the scalarization step.

Lemma 5.9. *Suppose that Assumptions 5.1 and 5.2 hold. Let $\{x^k\}_{k \in K}$ be a linked sequence generated by MOTR at the scalarization step, with finitely many successful iterations at the scalarization step. Then this linked sequence converges to a Pareto critical point.*

Proof. Assume that $x^{k_0+l} = x^*$ for all $l \in \mathbb{N}$, where $k_0 = \max\{k_s, k_{ep}\}$, k_s is the index of the last successful scalarization step iteration and k_{ep} is defined as in Remark 5.2. So, $\Delta_{sc}^{k_0+l}$ converges to zero, because in the scalarization step all iterations are unsuccessful for sufficiently large l.

Suppose that x^* is not a Pareto critical point. According to Lemma 5.7, there must be a very successful scalarization step iteration, with an index larger than k_0, which is a contradiction because all iterations after k_0 are unsuccessful. Therefore, x^* is a Pareto critical point. □

The following lemma clarifies the behavior of MOTR when a linked sequence has an infinite number of distinct points generated at the scalarization step.

Lemma 5.10. *Suppose that Assumptions 5.1 and 5.2 hold. Let $\{x^k\}_{k\in K}$ be a linked sequence of points generated by MOTR, with infinitely many successful iterations at the scalarization step. Then*

$$\liminf_{k\to+\infty;\, k\in K} \omega(x^k) = 0.$$

Proof. Suppose that $\liminf_{k\to+\infty;\, k\in K} \omega(x^k) \neq 0$. So, there must exist a constant $\epsilon > 0$ such that for all $k \in K$,

$$\omega(x^k) \geq \epsilon.$$

Lemma 5.8 guarantees the existence of $\Delta > 0$ such that $\Delta_{sc}^k \geq \Delta$, for all $k \in K$.

Consider S, the set of indexes of successful scalarization step iterations, $k \in S \cap K$, and $k > k_{eps}$, where k_{eps} is the index of the first successful scalarization step iteration after k_{ep}, and k_{ep} is defined as in Remark 5.2. Thus, $\rho_{sc}^k \geq \eta_{sc}^1$. According to Assumption 5.2, we have

$$\begin{aligned}
\phi(x^k) - \phi(x^{k+1}) &\geq \eta_{sc}^1(\phi_m^k(x^k) - \phi_m^k(x^{k+1})) \\
&\geq \eta_{sc}^1 \kappa_\phi \omega(x^k) \min\left\{\frac{\omega(x^k)}{\kappa_h}, \Delta_{sc}^k\right\} \\
&\geq \eta_{sc}^1 \kappa_\phi \epsilon \min\left\{\frac{\epsilon}{\kappa_h}, \Delta\right\}.
\end{aligned}$$

Summing over all successful iterations at the scalarization step, from k_{eps} to k results in

$$\begin{aligned}
\phi(x^{k_{eps}}) - \phi(x^{k+1}) &= \sum_{i=k_{eps}, i\in S\cap K}^{k} \phi(x^i) - \phi(x^{i+1}) \\
&\geq \sigma_k\, \eta_{sc}^1\, \kappa_\phi\, \epsilon\, \min\left\{\frac{\epsilon}{\kappa_h}, \Delta\right\},
\end{aligned}$$

where σ_k represents the number of successful scalarization step iterations in the linked sequence from k_{eps} to k. It is clear that $\lim_{k\to+\infty;\, k\in K} \sigma_k = +\infty$, because there are infinitely many such iterations. Thus, $\phi(x^{k_{ep}}) - \phi(x^{k+1})$ is unbounded. So, $\phi(x)$ can not be bounded from below and this is a contradiction. Therefore, $\liminf_{k\to+\infty;\, k\in K} \omega(x^k) = 0$. □

We are now ready to prove the main result, for the linked sequences generated by MOTR.

Theorem 5.1. *Let Assumptions 5.1 and 5.2 hold. For every linked sequence of points $\{x^k\}_{k\in K}$ generated by MOTR, we have*

$$\lim_{k\to+\infty;\, k\in K} \omega(x^k) = 0.$$

Proof. Let $\{x^k\}_{k\in K}$ be a linked sequence generated by MOTR. If there exists $i \in \{1,...,q\}$, such that for each $k \in \mathbb{N}, \Delta_{ep}^{i,k} > 0$ or if for each $i \in \{1,...,q\}$, there exists $k_i \in \mathbb{N}$, such that for all $k > k_i, \Delta_{ep}^{i,k} = 0$, but there are only finitely many successful iterations at the scalarization step, Remarks 5.1, 5.2, and Lemma 5.9 guarantee $\lim\limits_{k\to+\infty;\, k\in K} \omega(x^k) = 0$ by Lemma 5.1.

Now, assume that for all $k > k_i, \Delta_{ep}^{i,k} = 0$ and there is an infinite number of successful scalarization step iterations. Suppose that there exists a subsequence of successful scalarization step iterations, indexed by $\{t_j > k_{ep}\} \subset S \cap K$, such that

$$\omega(x^{t_j}) \geq 2\epsilon > 0, \tag{5.12}$$

for some $\epsilon > 0$ and for all j, where k_{ep} is defined as in Remark 5.2.

From Lemma 5.10, for each t_j, there exists a first successful scalarization step iteration $l_j > t_j$ such that $\omega(x^{l_j}) < \epsilon$. Thus, there exists another subsequence of $S \cap K$, indexed by $\{l_j\}$, such that

$$\omega(x^k) \geq \epsilon \text{ for } t_j \leq k < l_j \text{ and } \omega(x^{l_j}) < \epsilon. \tag{5.13}$$

Consider the successful iterates whose indexes are in

$$\mathcal{K} = \{k \in S \cap K \,|\, \exists j \in \mathbb{N} : t_j \leq k < l_j\},$$

where t_j and l_j belong to the two subsequences defined above.

Assumption 5.2, the fact that $\mathcal{K} \subset S \cap K$, and inequalities (5.13) guarantee that, for $k \in \mathcal{K}$,

$$\begin{aligned}
\phi(x^k) - \phi(x^{k+1}) &\geq \eta_{sc}^1(\phi_m^k(x^k) - \phi_m^k(x^{k+1})) \\
&\geq \eta_{sc}^1 \kappa_\phi \omega(x^k) \min\left\{\frac{\omega(x^k)}{\kappa_h}, \Delta_{sc}^k\right\} \\
&\geq \eta_{sc}^1 \kappa_\phi \epsilon \min\left\{\frac{\epsilon}{\kappa_h}, \Delta_{sc}^k\right\}.
\end{aligned} \tag{5.14}$$

The sequence $\{\phi(x^k)\}_{k\in K}$ is convergent, since it is monotonically decreasing and bounded from below. Thereby, $\lim\limits_{k\to+\infty;\, k\in K} \left(\phi(x^k) - \phi(x^{k+1})\right) = 0$. Consequently, considering the minimum part in the last term of (5.14), for $k \in \mathcal{K}$ sufficiently large, it implies that

$$\Delta_{sc}^k \leq \frac{1}{\eta_{sc}^1 \kappa_\phi \epsilon}(\phi(x^k) - \phi(x^{k+1})).$$

Therefore, for j sufficiently large, it holds that

$$\begin{aligned}
\|x^{t_j} - x^{l_j}\| &\leq \sum_{i=t_j, i\in\mathcal{K}}^{l_j-1} \|x^i - x^{i+1}\| \\
&\leq \sum_{i=t_j, i\in\mathcal{K}}^{l_j-1} \Delta_{sc}^i \\
&\leq \frac{1}{\eta_{sc}^1 \kappa_\phi \epsilon}(\phi(x^{t_j}) - \phi(x^{l_j})).
\end{aligned}$$

Again, the convergence of $\{\phi(x^k)\}_{k\in K}$ implies

$$\lim_{j\to+\infty} \|x^{t_j} - x^{l_j}\| = 0.$$

Assumption 5.1 and the uniform continuity of ω allow us to conclude that

$$\lim_{j\to+\infty} |\omega(x^{t_j}) - \omega(x^{l_j})| = 0,$$

contradicting the fact that $|\omega(x^{t_j}) - \omega(x^{l_j})| \geq \epsilon$, a consequence of the definition of sequences $\{t_j\}$ and $\{l_j\}$, in (5.12) and (5.13).

So, no subsequence of successful iterations satisfying (5.12) can exist, and subsequently

$$\lim_{k\to+\infty;\, k\in K} \omega(x^k) = 0.$$

□

5.3 Numerical results

The numerical experiments were performed with two main goals. The first was to illustrate the importance of each of the key algorithmic features of MOTR, namely the extreme point step, the scalarization step, and the middle point strategy, corresponding to Algorithms 5, 6, and 7, respectively.

To achieve this purpose, three different versions of MOTR were implemented, with each version omitting one of the aforementioned strategies:

- MOTRep: MOTR without the extreme point step;
- MOTRsc: MOTR without the scalarization step;
- MOTRmiddle: MOTR using a different strategy than the one described in Algorithm 7 to select the point where to solve the scalarization problem.

In the MOTRmiddle algorithm, instead of using the middle point step, a different strategy is employed to select a point from the current list of nondominated points for solving the scalarization problem. By this strategy, all points are first sorted according to the value of the objective function component under analysis. Average distances are then computed for each point by considering the distances to its two closest neighboring points from the sorted set. The average distances of the first and last points correspond to the distances of these points to the next or previous points, respectively. The selected point is the one with the largest average distance, given that its scalarization step trust-region radius is greater than or equal to the minimum allowed value.

The second goal of the numerical section was to compare the performance of MOTR against other derivative-based multiobjective optimization solvers that intrinsically attempt to generate approximations to the complete Pareto front of a multiobjective optimization problem. With this purpose, the Multiobjective Sequential Quadratic Programming (MOSQP) algorithm [19] was selected.

All codes were implemented in MATLAB (version R2021b was considered). The minimization subproblems of MOTR, defined at the extreme point and scalarization steps, were solved with the MATLAB function `fmincon.m`.

MOTR was initially designed and analyzed for unconstrained multiobjective optimization. However, it can be easily adapted to incorporate bound constraints, by including these constraints in the subproblems to be solved. Notice that for convex feasible regions, like the case of bound constraints, the middle point, computed by using Algorithm 7, will remain feasible.

Test problems

As a test set, we utilized 54 twice continuously differentiable bound constrained multiobjective optimization problems, available at

`https://docentes.fct.unl.pt/algb/pages/problems-collections`,

encompassing a range of variables from 1 to 30, involving 2 or 3 objective function components. Table 5.1 contains a comprehensive list of the problems along with their respective dimensions.

Table 5.1: The set of problems considered in the numerical experiments. For each problem, n represents the number of variables and q is the number of components of the objective function.

Problem	n	q	**Problem**	n	q	**Problem**	n	q
BK1	2	2	CL1	4	2	Deb41	2	2
Deb513	2	2	Deb521b	2	2	DG01	1	2
DPAM1	10	2	DTLZ1	7	3	DTLZ1n2	2	2
DTLZ2	12	3	DTLZ2n2	2	2	DTLZ3	12	3
DTLZ3n2	2	2	DTLZ4	12	3	DTLZ4n2	2	2
DTLZ6	22	3	DTLZ6n2	2	2	ex005	2	2
Far1	2	2	Fonseca	2	2	IKK1	2	3
IM1	2	2	Jin1	2	2	Jin3	2	2
L2ZDT2	30	2	L3ZDT2	30	2	lovison1	2	2
lovison2	2	2	lovison3	2	2	lovison4	2	2
lovison5	3	3	lovison6	3	3	LRS1	2	2
MHHM1	1	3	MHHM2	2	3	MLF1	1	2
MLF2	2	2	MOP1	1	2	MOP2	4	2
MOP3	2	2	MOP5	2	3	MOP6	2	2
MOP7	2	3	SK1	1	2	SK2	4	2
SP1	2	2	SSFYY1	2	2	SSFYY2	1	2
TKLY1	4	2	VFM1	2	3	VU1	2	2
VU2	2	2	ZDT2	30	2	ZLT1	10	3

Numerical settings of MOTR

MOTR was run with the parameters $\mu_1 = 0.5$, $\mu_2 = 2$, $\eta_{ep}^1 = \eta_{sc}^1 = 0.001$, $\eta_{ep}^2 = \eta_{sc}^2 = 0.9$, $\Delta_{ep}^{init} = (1, ..., 1)^\top \in \mathbb{R}^q$, and $\Delta_{sc}^{init} = 1$. Regarding the update of the trust-region radius at very successful iterations, it was only increased if the boundary of the trust region was reached. In this case, a maximum value of $\Delta^{max} = \|u - l\|/2$ is allowed, where u and l represent the upper and lower bounds of the problem variables. The algorithm was always initialized with a single point, namely the centroid of the feasible region.

As stopping criteria, the minimum trust-region radius values $\Delta_{ep}^{min} = \Delta_{sc}^{min} = 10^{-5}$ were allowed, componentwise in the case of Δ_{ep}. Three different budgets were taken into account in terms of function evaluations, with values of 500, 5000, and 20000. For each problem, the approximation to the Pareto front generated by each algorithm corresponds to all the current feasible nondominated points, stored in the list L.

5.3.1 Performance assessment and metrics

The performance profiles proposed by Dolan and Moré [14], described in Section 4.5, were utilized as a performance tool. These profiles enable the simultaneous evaluation of the numerical performance of various solvers across different metrics.

In a simplified way, the performance of solver s on the given set of problems is denoted by $\rho_s(\tau)$, where $\tau \geq 1$. Higher values of $\rho_s(\tau)$ indicate better numerical performance for solver s. Specifically, a solver with a larger value of $\rho_s(1)$ is considered to be more efficient, while a solver with a larger value of $\rho_s(\tau)$ for large τ values is deemed to be more robust.

To evaluate the effectiveness and robustness of a multiobjective optimization solver, it is important to assess its ability to generate a large percentage of well-distributed nondominated points and also be able to capture the extent of the Pareto front of the multiobjective optimization problem. For this purpose, we considered four metrics that aim to quantify these characteristics, called purity, hypervolume, and the spread metrics Γ and Δ, described in Section 4.5.

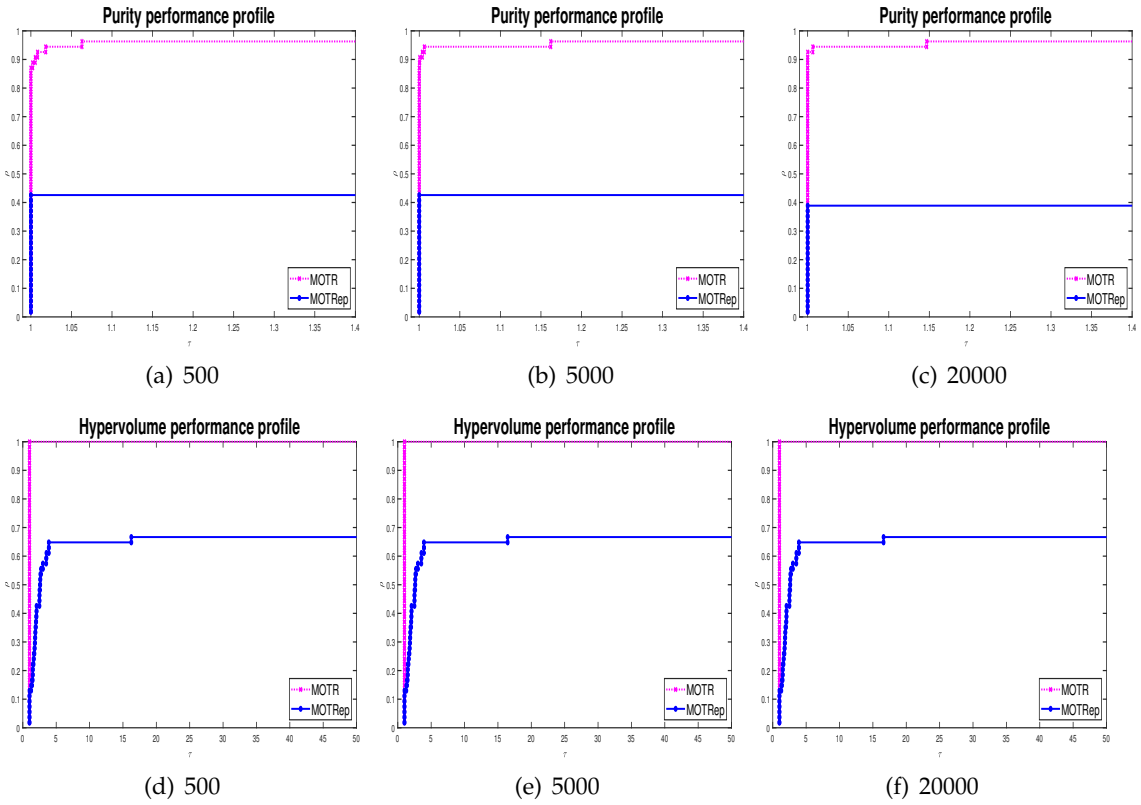
5.3.2 Adequacy of the algorithmic structure of MOTR

Figures 5.1 and 5.2 depict performance profiles comparing MOTR and MOTRep, where MOTRep is a version of MOTR that excludes the extreme point step.

Figures 5.3 and 5.4 report the comparison between MOTR and MOTRsc, where the latter algorithm corresponds to a version of MOTR omitting the scalarization step.

Finally, MOTR numerical performance is evaluated against MOTRmiddle, where, in MOTRmiddle, the use of Algorithm 7 is replaced by the strategy described in Section 5.3 to select points to solve the scalarization problems. The results of this assessment can be found in Figures 5.5 and 5.6.

It is clear the advantage of MOTR over each one of its variants, both in terms of efficiency and robustness, for each one of the metrics considered, independently of the



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 5.1: Comparing MOTR and MOTRep based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

budget of function evaluations allowed. The exception appears in the results for the Δ metric, when comparing MOTR with MOTRsc or MOTRmiddle, where MOTR still continues to present a better performance in terms of efficiency, but is not competitive in the case of robustness.

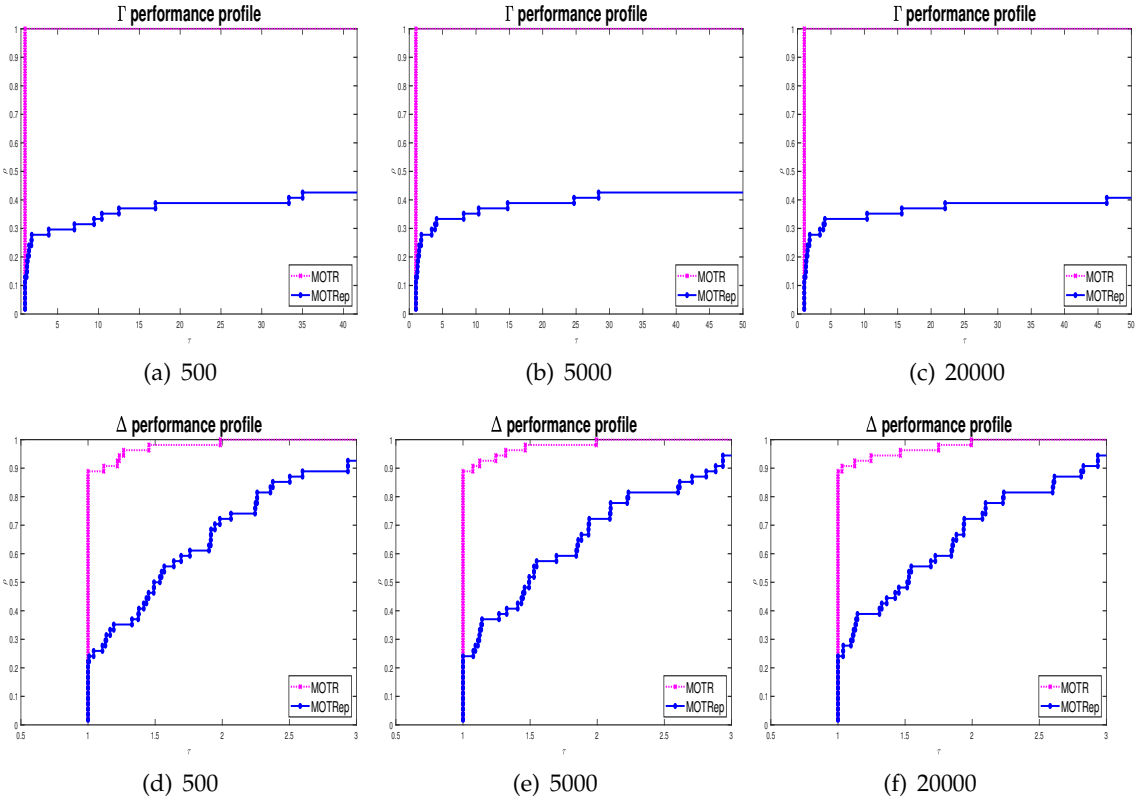
Considering these results, all parts are extremely essential and crucial for the good numerical performance of MOTR. Based on the obtained results, we are not only confident about MOTR but also encouraged to compare it with another state-of-the-art solver.

5.3.3 Comparing MOTR with MOSQP

MOSQP was proposed in [19], based on sequential quadratic programming techniques, incorporating in its algorithmic structure strategies to compute approximations to the complete Pareto front of a given multiobjective optimization problem. The solver also keeps a list of points that is updated at each iteration by solving single-objective constrained optimization problems derived as SQP problems.

In [19], the authors compared the solver against a classical scalarization approach for biobjective problems and also genetic algorithms. The reported numerical results establish the superiority of MOSQP over the remaining solvers tested.

A MATLAB implementation of MOSQP is distributed by the authors, providing



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

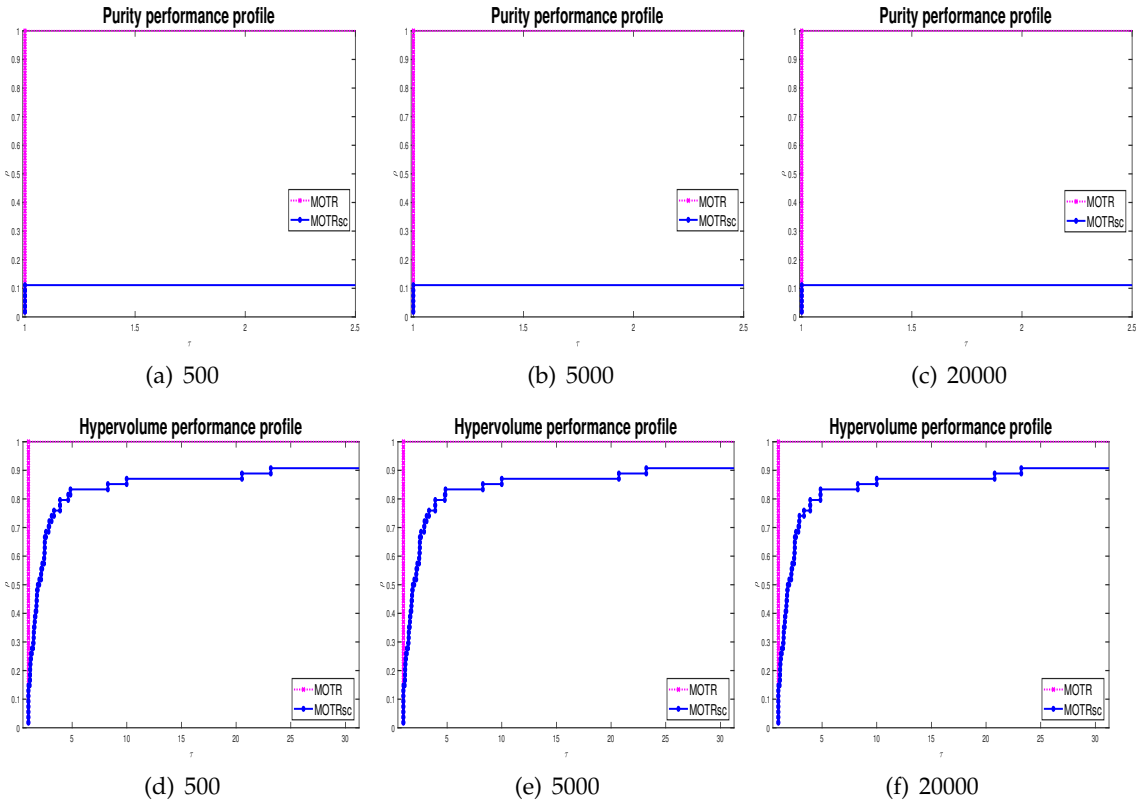
Figure 5.2: Comparing MOTR and MOTRep based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

different algorithmic choices. We selected MOSQP $(H = (I, \nabla^2 f), line)$, which corresponds to a line initialization strategy, by computing 200 initial points evenly spaced in the line segment joining the lower and upper bounds of the variables, and where the identity matrix and the true Hessians are used in the second and third algorithmic stages, respectively [19]. This version is reported in [19] as the one that presents the best computational performance. In regard to stopping criteria, we kept all the default values but experimented with the three different budgets for function evaluations.

For each problem, the approximation to the Pareto front generated by each one of the solvers corresponds to all current feasible nondominated points, stored in the corresponding lists. Figures 5.7 and 5.8 display the performance profiles for purity, hypervolume, and the spread metrics, showcasing the comparison between MOTR and MOSQP.

MOTR is clearly competitive, with remarkably good results in terms of efficiency and robustness for purity, hypervolume, and Γ metrics. Regarding the uniformity of the distribution of points across the approximation to the Pareto front, MOSQP presents a better performance. These conclusions hold, independently of the budget of function evaluations considered.

Table 5.2 reports the number of feasible nondominated points obtained by each solver, for a maximum budget of 5000 function evaluations, after joining the lists of nondominated points corresponding to both solvers for each problem and removing the dominated points.

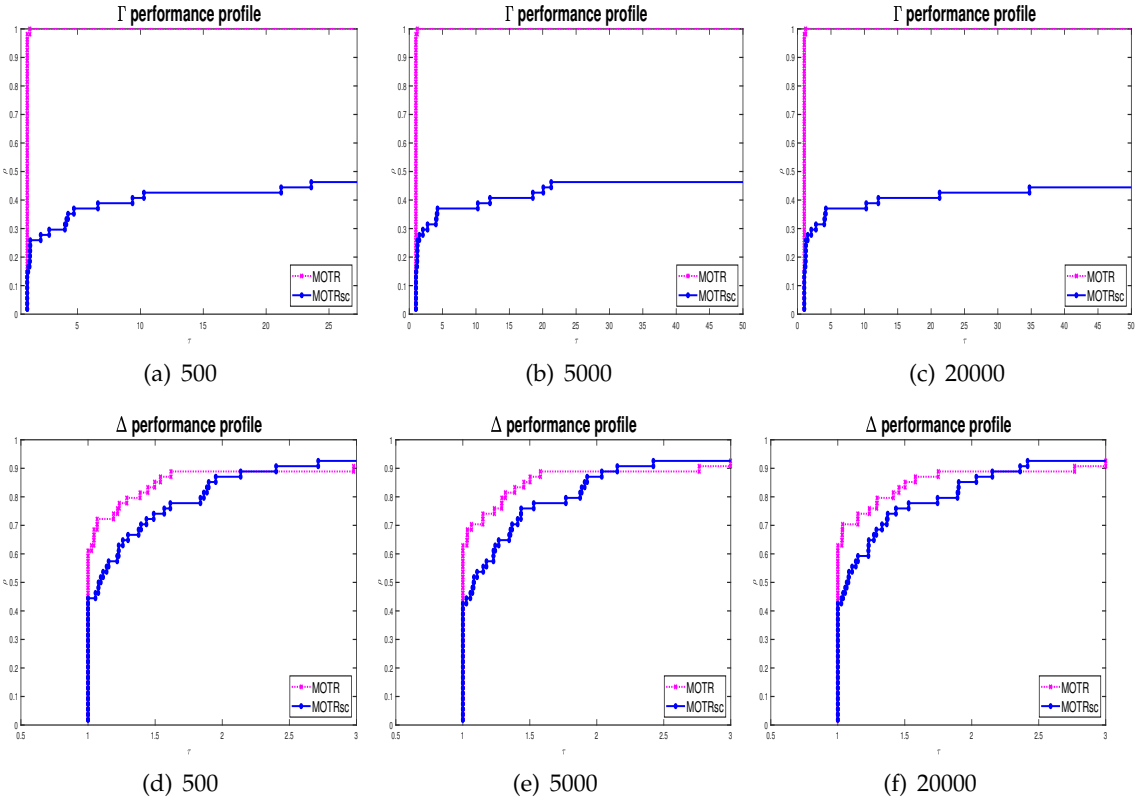


(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 5.3: Comparing MOTR and MOTRsc based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

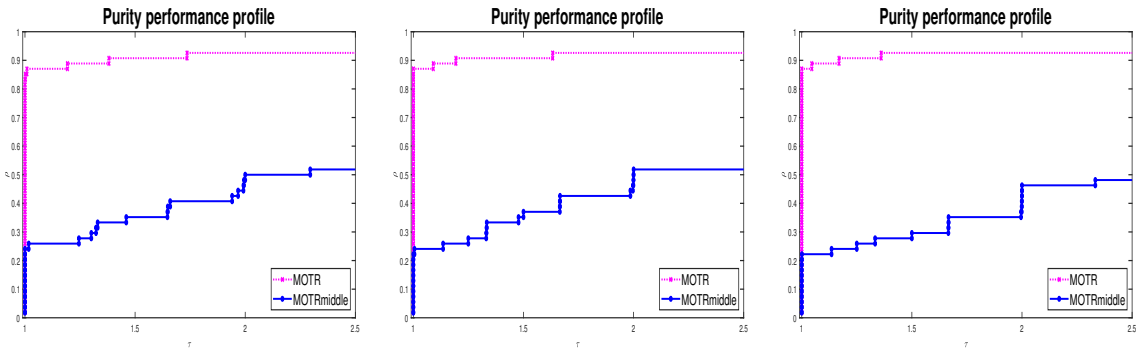
Figures 5.9 and 5.10 illustrate the final approximations to the Pareto fronts obtained by MOTR and MOSQP on two biobjective and two triobjective problems, respectively.



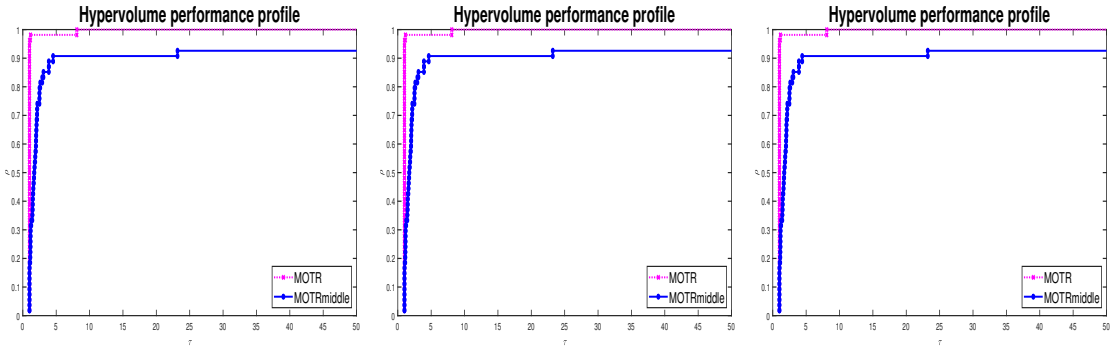
(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 5.4: Comparing MOTR and MOTRsc based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

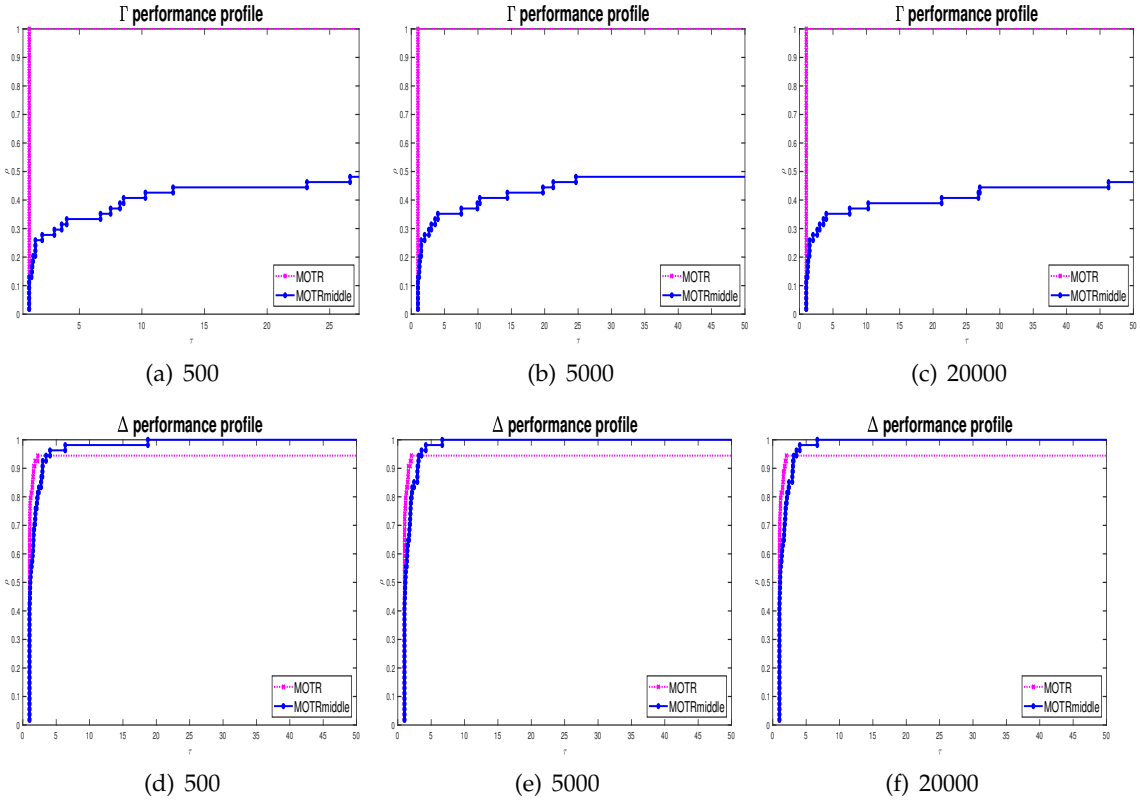


(a) 500 (b) 5000 (c) 20000



(d) 500 (e) 5000 (f) 20000

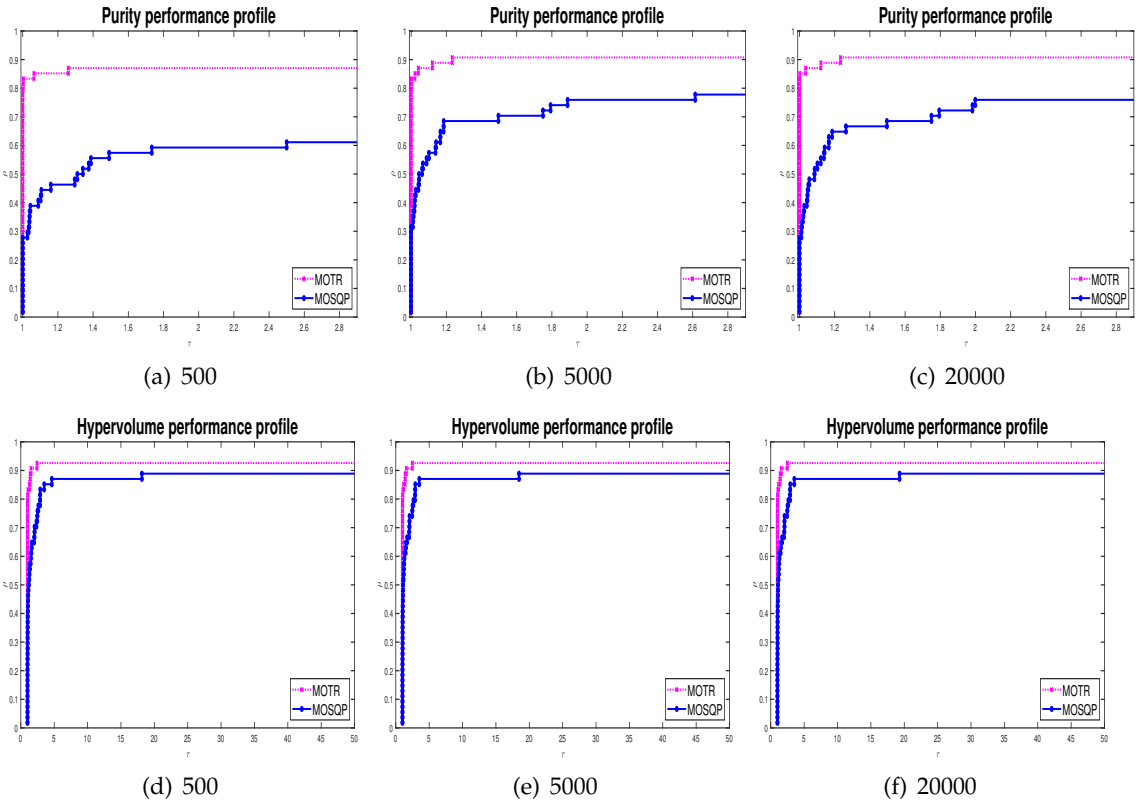
Figure 5.5: Comparing MOTR and MOTRmiddle based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

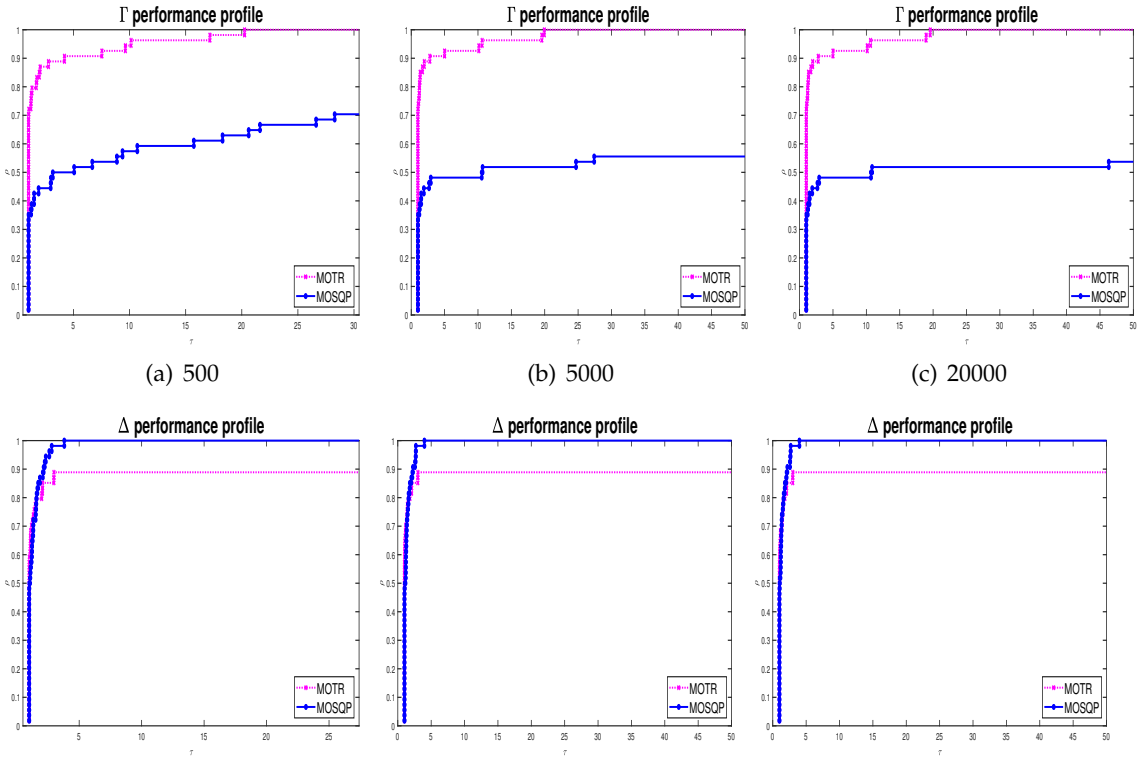
Figure 5.6: Comparing MOTR and MOTRmiddle based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

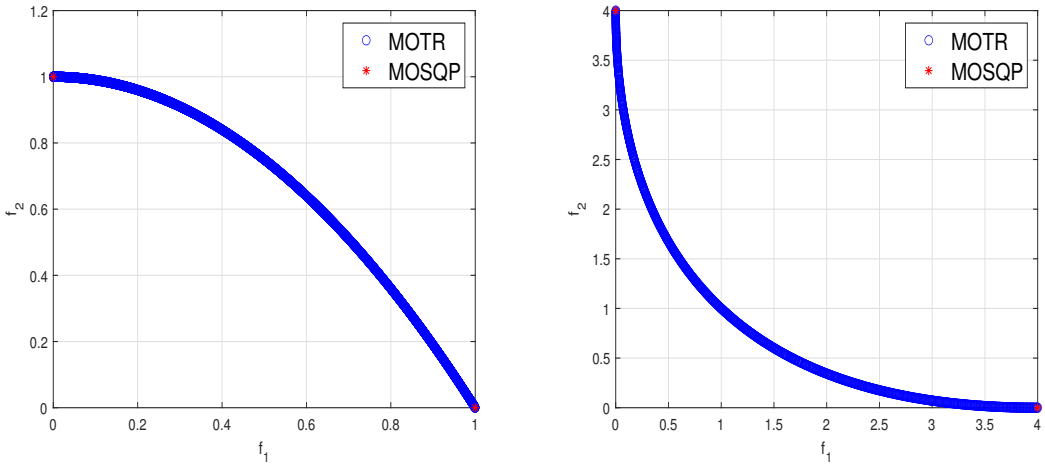
Figure 5.7: Comparing MOTR and MOSQP based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 5.8: Comparing MOTR and MOSQP based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

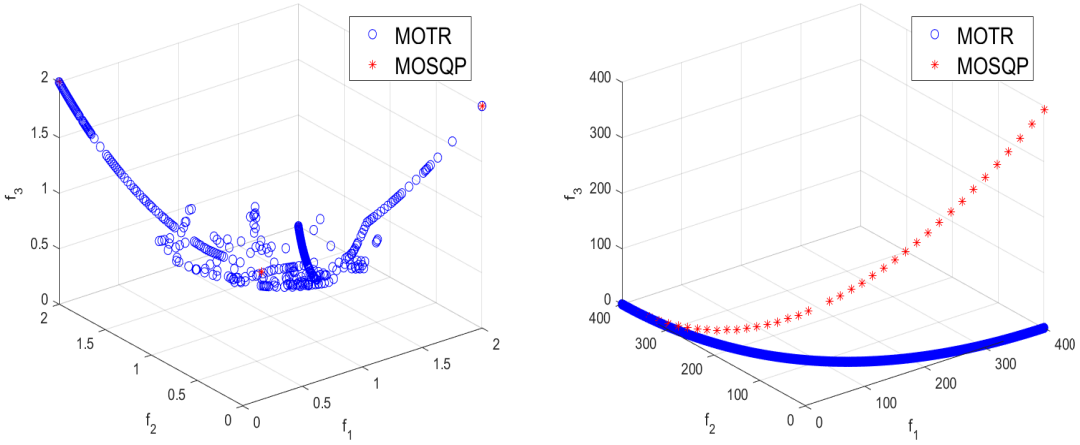


(a) Problem: ZDT2 (b) Problem: MOP1

Figure 5.9: Approximations to the Pareto fronts of problems ZDT2 and MOP1, obtained by solvers MOTR and MOSQP, for a budget of 5000 function evaluations.

Table 5.2: Number of feasible nondominated points in the final approximation of the Pareto front, generated for each problem by MOTR and MOSQP, considering a budget of 5000 function evaluations.

Problem	**MOTR**	**MOSQP**	**Problem**	**MOTR**	**MOSQP**
BK1	5000	66	CL1	4827	89
Deb41	2197	64	Deb513	1	2
Deb521b	4477	1	DG01	77	25
DPAM1	2496	7	DTLZ1	4999	1
DTLZ1n2	4998	0	DTLZ2	4991	16
DTLZ2n2	4989	16	DTLZ3	4989	1
DTLZ3n2	4988	2	DTLZ4	1	8
DTLZ4n2	1	6	DTLZ6	950	1
DTLZ6n2	2497	1	ex005	4976	67
Far1	1726	52	Fonseca	4996	0
IKK1	3922	36	IM1	4982	90
Jin1	4997	87	Jin3	4834	2
L2ZDT2	0	2	L3ZDT2	752	1
lovison1	4942	93	lovison2	1	8
lovison3	4871	97	lovison4	4373	81
lovison5	412	18	lovison6	269	9
LRS1	4998	1	MHHM1	3936	8
MHHM2	4994	0	MLF1	0	17
MLF2	4938	34	MOP1	5000	0
MOP2	5000	24	MOP3	1610	19
MOP5	2535	12	MOP6	1	2
MOP7	4302	1	SK1	3960	88
SK2	1632	6	SP1	4952	59
SSFYY1	5000	1	SSFYY2	2502	1
TKLY1	595	68	VFM1	4997	49
VU1	0	90	VU2	2498	24
ZDT2	4936	2	ZLT1	383	1



(a) Problem: ZLT1

(b) Problem: IKK1

Figure 5.10: Approximations to the Pareto fronts of problems ZLT1 and IKK1, obtained by solvers MOTR and MOSQP, for a budget of 5000 function evaluations.

6

A class of trust-region methods for multiobjective derivative-free optimization

In this chapter, we focus on solving the multiobjective derivative-free optimization problem defined as follows

$$\begin{aligned} \min F(x) &= \big(f_1(x), \ldots, f_q(x)\big) \\ \text{s.t. } x &\in \mathbb{R}^n, \end{aligned} \tag{6.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^q$, $n, q \in \mathbb{N}$, and $q \geq 2$. The objective function components $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, q$, are assumed to be expensive to evaluate and conflicting with each other, meaning that it is not possible to find a single point that simultaneously minimizes all objective function components. Derivatives of the components of the objective function are not available and also it is not possible to approximate them. The problem solution is referred to as the Pareto front, which consists of a set of nondominated points.

In Section 6.1, a modified algorithm will be presented for solving the multiobjective derivative-free optimization problem. This algorithm aims to approximate the Pareto front of Problem (6.1) and incorporates new techniques for building models approximating the objective function components. The convergence analysis of this algorithm will be discussed in Section 6.2. Furthermore, Section 6.3 will present numerical results that showcase the efficiency and robustness of the proposed method.

6.1 Algorithmic structure

In this chapter, we introduce the Multiobjective Trust-Region Derivative-Free Optimization (MOTRDFO) algorithm as a modified approach. The original algorithmic structure was thoroughly explained in Chapter 5, focusing on multiobjective trust-region methods. MOTRDFO maintains exactly the same algorithmic framework as the one described in Chapter 5, except for its adjustment to derivative-free problems. In this context, it entirely disregards derivative information associated with the objective function components, in contrast to the utilization of Taylor-based models in Chapter 5.

To obtain comprehensive details about the algorithm, please refer to Chapter 5. However, to maintain the coherence of this chapter, here we provide a brief overview of the algorithmic structure. Subsequently, we will provide a comprehensive discussion of our methods for building models when derivatives are not available. In this context, new techniques based on polynomial interpolation and minimum Frobenius norm approaches are proposed to build models which approximate the objective function components.

The purpose is to approximate the complete Pareto front of Problem (6.1). The algorithm aims to achieve a comprehensive, dense, and uniformly spread approximation by iteratively performing two main steps: the extreme point step and the scalarization step. These steps continue until certain stopping criteria are met. The extreme point step expands the approximation by moving towards the extreme points of the Pareto front, delineated by Algorithm 5. On the other hand, the scalarization step focuses on reducing gaps in the Pareto front approximation, outlined by Algorithm 6. The scalarization step incorporates an additional step known as the middle point step, detailed by Algorithm 7. This intermediate step is employed to determine the initial points for solving the scalarization problems.

In MOTRDFO, a list of nondominated points and their associated quantities is maintained throughout the algorithm. This list, denoted as L, is defined as follows:

$$L = \{(x^j, F(x^j), \Delta^j_{ep}, \Delta^j_{sc}) \mid j \in J\}, \tag{6.2}$$

where, J represents the set of indexes of the points in the list, Δ^j_{ep} is a $q \times 1$ vector that stores, for each component, the trust-region radius associated with x^j and the corresponding component of the objective function, used during the extreme point step of the algorithm. Additionally, Δ^j_{sc} represents the trust-region radius utilized in the scalarization step.

Algorithm 8 outlines the main procedure of the MOTRDFO algorithm. In any of the two main steps, points are either selected or computed from the list L, and quadratic polynomial models are constructed around these points. These models serve as replacements for the components of the objective function.

Algorithm 8. MOTRDFO

Input
Initial list of nondominated points L_0 defined by (6.2).

For $k = 0, 1, 2, \ldots$
If $mod(k, 2) = 0$, then go to the *Extreme Point Step*.
Else go to the *Scalarization Step*.
If some stopping criterion is met, then return.
End For

At iteration k, for each i, $i \in \{1, \ldots, q\}$, the quadratic model m^k_i approximating the objective function component f_i around a given point x_{step} is defined as:

$$m_i^k(x) = f_i(x_{\text{step}}) + (x - x_{\text{step}})^\top g_i^k + \frac{1}{2}(x - x_{\text{step}})^\top H_i^k (x - x_{\text{step}}),$$

where g_i^k represents the gradient vector and H_i^k represents the Hessian matrix of the model m_i^k.

6.1.1 Building models

In this context, we provide clarification on the process of building model m_i^k to approximate the objective function component f_i, where $k \in \mathbb{N}$ and $i \in \{1, \ldots, q\}$, during the extreme point and scalarization steps. In a broader perspective, we present a method for computing a quadratic polynomial model $m : \mathbb{R}^n \to \mathbb{R}$ that serves as an approximation for a general single-objective function $f : \mathbb{R}^n \to \mathbb{R}$.

To build the model m, we use a set of sample points Y. All points in Y should be inside the ball $B(x_{\text{step}}, r\Delta_{\text{step}})$ defined as

$$B(x_{\text{step}}, r\Delta_{\text{step}}) = \left\{ x \in \mathbb{R}^n \mid \|x - x_{\text{step}}\| \leq r\Delta_{\text{step}} \right\}, \tag{6.3}$$

in which x_{step} is the current iterate, $r \geq 1$ is a user-defined parameter to control the distance of the sample points from the current iterate, and Δ_{step} represents the trust-region radius associated with the current step. Therefore, any points located outside this ball are disregarded during the current iteration.

Let $Y = \{y^0, y^1, \ldots, y^p\}$ be the sample set of $p_1 = p + 1$ points satisfying (6.3), in which the first point y^0 is equal to the current iterate point x_{step}, where we want to build model m around it. As a general rule, the first point of each sample set is always the current iterate. Furthermore, a distinct sample set is assigned to each point in the list of nondominated points.

At the beginning of the algorithm, for each initial point x^0, the initial sample set is a set of $2n + 1$ points defined by $\left\{x^0, x^0 + \Delta[I_n \ -I_n]\right\}$, in which $\Delta = \frac{1}{2}\min\left\{\Delta_{ep}^{init}, \Delta_{sc}^{init}\right\}$ and I_n denotes the $n \times n$ identity matrix.

In both the extreme point and scalarization steps, the new evaluated points are added to the corresponding sample sets without discarding any of the available points. In the middle point step, every time that a new middle point is computed and added to the list, the initial sample set is the union of the sample sets of two points associated with the corresponding gap.

The fact that points are never discarded may be surprising. However, in the context of expensive function evaluation, we should take all possible advantages from previously evaluated points. We also ensure the sample sets are computed in a way that the sample points are not very far from the current iterate, an important feature when building a local model.

The quadratic model m with gradient g and Hessian H is defined in the form

$$m(x) = f(y^0) + (x - y^0)^\top g + \frac{1}{2}(x - y^0)^\top H (x - y^0),$$

where y^0 is the current iterate.

To build this model, g and H are required to be computed. The specific technique used in this chapter for building m depends on p_1, which represents the number of points in Y.

Before delving further into this topic, interested readers can refer to Section 2.3 for additional background information. We will only review and discuss the necessary topics here as required to adequately formalize our technique.

Throughout this chapter, let $\mathcal{P}_n^2$ be the space of polynomials of degree less than or equal to 2 in $\mathbb{R}^n$, and $b_1 = \frac{(n+1)(n+2)}{2}$ be the dimension of this space.

Consider the natural basis that can be written as

$$\begin{aligned}\overline{\phi} &= \{\phi_0(x), \phi_1(x), \ldots, \phi_b(x)\} \\ &= \left\{1, x_1, \ldots, x_n, \frac{x_1^2}{2}, x_1x_2, \ldots, x_{n-1}x_n, \frac{x_n^2}{2}\right\}.\end{aligned}$$

where $x \in \mathbb{R}^n$ and $b_1 = b + 1$.

The interpolating polynomial $m(x) \in \mathcal{P}_n^2$ can be formulated as

$$m(x) = \sum_{j=0}^{b} \alpha_j \phi_j(x),$$

where $\alpha \in \mathbb{R}^{b_1}$ is a vector of real coefficients.

If $p_1 = b_1$, the interpolating polynomial model m is built by computing the coefficients $\alpha_0, \alpha_1, ..., \alpha_p$, satisfying the interpolation conditions

$$m(y^i) = f(y^i), \qquad i = 0, \ldots, p.$$

It can be rephrased as follows

$$m(y^i) = \sum_{j=0}^{p} \alpha_j \phi_j(y^i) = f(y^i), \quad i = 0, .., p.$$

This system of linear equations can be rewritten in matrix form as

$$M(\overline{\phi}, Y)\alpha = f(Y),$$

where,

$$M(\overline{\phi}, Y) = \begin{bmatrix} 1 & y_1^0 & \ldots & y_n^0 & \frac{(y_1^0)^2}{2} & y_1^0 y_2^0 & \ldots & y_{n-1}^0 y_n^0 & \frac{(y_n^0)^2}{2} \\ 1 & y_1^1 & \ldots & y_n^1 & \frac{(y_1^1)^2}{2} & y_1^1 y_2^1 & \ldots & y_{n-1}^1 y_n^1 & \frac{(y_n^1)^2}{2} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ 1 & y_1^p & \ldots & y_n^p & \frac{(y_1^p)^2}{2} & y_1^p y_2^p & \ldots & y_{n-1}^p y_n^p & \frac{(y_n^p)^2}{2} \end{bmatrix},$$

$$\alpha = \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \vdots \\ \alpha_p \end{bmatrix}, \; f(Y) = \begin{bmatrix} f(y^0) \\ f(y^1) \\ \vdots \\ f(y^p) \end{bmatrix}.$$

If more than $\frac{(n+1)\times(n+2)}{2}$ points are available, indicated by $p > b$, MOTRDFO utilizes an efficient selection strategy, outlined by Algorithm 9.

Let $Y = \{y^0, y^1, \ldots, y^b, y^{b+1}, \ldots, y^p\}$, where $p > b$, be the set of sample points, in which all points are sorted according to their distance from y^0. At first, we select the first $b+1$ points, $Y^b = \{y^0, y^1, \ldots, y^b\}$, as the sample set, then the geometry of Y^b is checked, using the notion of poisedness which illustrates how well the interpolation set Y^b spans the region where interpolation is of interest.

To check and control the geometry of Y^b, we use the notion of condition number. Recall from Section 2.3 that if we select the natural basis and if $\widehat{Y^b}$ denotes the shifted and scaled version of Y^b in the ball $B(0,1)$, then the condition number of $M(\overline{\phi}, \widehat{Y^b})$ is a meaningful measure of poisedness. Smaller values of the condition number of $M(\overline{\phi}, \widehat{Y^b})$ illustrate higher levels of poisedness of Y^b, associated with higher quality of interpolation models, and vice versa.

In this chapter, $cond(M(\overline{\phi}, \widehat{Y^b}))$ remarks the condition number of matrix $M(\overline{\phi}, \widehat{Y^b})$ and the set $\widehat{Y^b}$ is expressed by

$$\widehat{Y^b} = \{0, \widehat{y}^1, ..., \widehat{y}^b\} = \left\{0, \frac{y^1 - y^0}{\Delta}, ..., \frac{y^b - y^0}{\Delta}\right\} \subset B(0,1),$$

where $B(0,1)$ is the ball centered at the origin, of radius equal to 1, and

$$\Delta = \max_{1 \le i \le b} \|y^i - y^0\|.$$

To improve the geometry of Y^b, we use the basis of quadratic Lagrange polynomials $\{l_i(x),\ i = 0, \ldots, b\}$, expressed by

$$l_i(y^j) = \begin{cases} 1 & i = j \\ 0 & \text{otherwise.} \end{cases}$$

The basis of Lagrange polynomials is commonly used for measuring poisedness in the multivariate polynomial interpolation literature [8, Chapter 3].

The geometry of Y^b is controlled by a parameter called $condtol$. If $cond(M(\overline{\phi}, \widehat{Y^b}))$ is equal to or smaller than $condtol$, then we accept the geometry of Y^b and build the model m, using Y^b. Otherwise, we improve the level of poisedness by updating Y^b, via using the Lagrange polynomials, as it is illustrated in Algorithm 9.

On the other hand, if $n + 1 < p_1 < b_1$, meaning that there are more points than the ones necessary for linear interpolation but fewer than the ones required for determined

Algorithm 9. Checking and improving the level of poisedness of an interpolation set

Input

The list of sample points $Y = \{y^0, y^1, \ldots, y^b, y^{b+1}, \ldots, y^p\}$ in which all points are sorted according to their distance from y^0, where $b+1$ is the dimension of $\mathcal{P}_n^2$, and $p > b$.
Value for the parameter $condtol > 1$.

1. Check the geometry

Select the first $b+1$ points, $Y^b = \{y^0, y^1, \ldots, y^b\}$, as the sample set.
If $cond(M(\overline{\phi}, \widehat{Y^b})) \leq condtol$, then return.

2. Improve the geometry

For $i = b+1, ..., p$

Define $Y^b_{new} = Y^b \backslash \{y^s\} \cup \{y^i\}$, where $s \in \arg\max_{j=1,\ldots,b} |l_j(y^0)|$,
and $\{l_j(x),\ j = 0, \ldots, b\}$ is the basis of Lagrange polynomials for Y^b.

If $cond(M(\overline{\phi}, \widehat{Y^b_{new}})) < cond(M(\overline{\phi}, \widehat{Y^b}))$, then set $Y^b = Y^b_{new}$.
If $cond(M(\overline{\phi}, \widehat{Y^b})) \leq condtol$, then return.

End For

quadratic interpolation, we compute a minimum Frobenius norm model by solving

$$\begin{aligned} &\min \frac{1}{2}\|H\|^2, \\ &\text{s.t.} \\ &m(y^j) = f(y^j), \quad j = 1, ..., p. \end{aligned}$$

Finally, if fewer than $n+2$ points are available, new points are added to the sample set by considering the extreme points or mid-points of the set $\left\{y^0 + \frac{1}{2}\Delta_{\text{step}}[I_n \ -I_n]\right\}$, to reach a total of $n+2$ points, in which again y^0 is the current iterate, Δ_{step} represents the trust-region radius of the current step and I_n is $n \times n$ identity matrix.

We assume that at each iteration, every model is built by using a Λ-poised sample set that holds a sufficient level of poisedness. Furthermore, at each iteration, every model that we build is at least a minimum Frobenius norm model. As a result, the models are fully linear [8, 10], detailed in Section 2.3, allowing to establish the following error bounds:

$$\|\nabla f(x+s) - \nabla m(x+s)\| \leq \kappa_{eg}\Delta, \quad \forall s \in B(0,\Delta), \tag{6.4}$$

$$|f(x+s) - m(x+s)| \leq \kappa_{ef}\Delta^2, \quad \forall s \in B(0,\Delta), \tag{6.5}$$

where κ_{eg} and κ_{ef} are positive constants, depending on Λ, p, b, and the Lipschitz constant of the gradient of f.

To achieve convergence to a first-order critical point in model-based optimization algorithms, it is necessary for the models to be fully linear [8]. Consequently, we will incorporate this characteristic when conducting our convergence analysis.

6.2 Convergence analysis

In order to analyze the theoretical behavior of MOTRDFO, we will assume that no stopping criteria are defined, specifically $\Delta_{ep}^{min} = \Delta_{sc}^{min} = 0$. Convergence will again be established for linked sequences of points $\{x^k\}_{k\in K}$ generated by MOTRDFO. A linked sequence is defined as follows.

Definition 6.1. *Consider $\{L_k\}_{k\in\mathbb{N}}$ as the sequence of sets of nondominated points generated by Algorithm 8. A linked sequence is a sequence $\{x^k\}_{k\in K}$, where $K \subseteq \mathbb{N}$ denotes the indexes of the points belonging to the linked sequence, and such that for any $k \in K$, the element $(x^k, F(x^k), \Delta_{ep}^k, \Delta_{sc}^k) \in L_k$ is generated from the element $(x^{k-1}, F(x^{k-1}), \Delta_{ep}^{k-1}, \Delta_{sc}^{k-1}) \in L_{k-1}$.*

Like the derivative-based case, an initial point of a linked sequence can take one of the following forms:

- A point in the initial list, provided by the user;
- Middle points generated by MOTRDFO in the middle point step, such that these points are not currently in the list and are added to the list by the algorithm.

We are going to establish that every limit point of a linked sequence generated by MOTRDFO is a Pareto critical point, defined by Definition 4.5.

According to Lemma 4.3, by reasonable assumptions, we are going to establish that for every linked sequence $\{x^k\}_{k\in K}$ generated by MOTRDFO, the following statement holds:

$$\lim_{k\to+\infty;\, k\in K} \omega(x^k) = 0,$$

where

$$\omega(x) = -\min_{\|d\|\leq 1} \max_{i=1,\ldots,q} \nabla f_i(x)^\top d. \tag{6.6}$$

For each $i \in \{1, \ldots, q\}$, we assume that the objective function component f_i is lower bounded and twice continuously differentiable. In each iteration k, all models m_i^k are quadratic polynomials, implying that they are twice continuously differentiable for $i \in \{1, \ldots, q\}$.

Assumption 6.1. *The Hessians of the objective function f_i and model function m_i^k are uniformly bounded. So, there exists a constant $\kappa_h > 0$ satisfying*

$$\|\nabla^2 m_i^k(x)\| \leq \kappa_h$$

and

$$\|\nabla^2 f_i(x)\| \leq \kappa_h$$

for all $k \in \mathbb{N}$, $i \in \{1, \ldots, q\}$, and $x \in \mathbb{R}^n$.

Assumption 6.1 implies that ∇f_i is Lipschitz continuous, for each $i \in \{1, ..., q\}$. From it, we can deduce that function ω, defined by (6.6), is uniformly continuous [34].

As usual, B_k denotes the current iteration ball, defined as follows:

$$B_k = B(x^k, \Delta_k) = \left\{x \in \mathbb{R}^n \mid \|x - x^k\| \leq \Delta_k\right\},$$

where x^k is the current iterate and Δ_k represents the current trust-region radius.

In a model-based trust-region algorithm, it is important to have good local accuracy of model functions compared to real functions in every iteration. So, the following lemma ensures the accuracy of models.

Lemma 6.1. *Assume that Assumption 6.1 holds. In every iteration k, the model m_i^k is valid for f_i in B_k, for all $i \in \{1, ..., q\}$. That is, there exists a constant $\kappa_{fm} > 0$ such that*

$$|f_i(x) - m_i^k(x)| \leq \kappa_{fm}\Delta_k^2$$

holds for all $k \in \mathbb{N}$, $i \in \{1, ..., q\}$, and $x \in B_k$.

Proof. For each $i \in \{1, ..., q\}$, it is clear that f_i is continuously differentiable with Lipschitz continuous gradient. Furthermore, for each $k \in \mathbb{N}$, m_i^k is at least a minimum Frobenius norm model built from a poised set of sample points in $B(x^k, r\Delta_k)$ defined by (6.3). So, m_i^k is at least fully linear at $B(x^k, r\Delta_k)$. Therefore, according to (6.5), for each $i \in \{1, ..., q\}$ there exists a positive constant κ_{ef}^i such that

$$|f_i(x) - m_i^k(x)| \leq \kappa_{ef}^i r^2 \Delta_k^2, \quad \forall x \in B(x^k, r\Delta_k).$$

Hence, with $\kappa_{fm} = r^2 \max_{i=1,...,q} \kappa_{ef}^i$ we have

$$|f_i(x) - m_i^k(x)| \leq \kappa_{fm}\Delta_k^2, \quad \forall\, i \in \{1, ..., q\} \text{ and } x \in B(x^k, r\Delta_k).$$

Considering the fact that $B_k \subseteq B(x^k, r\Delta_k)$, the proof is over. □

Following the same reasoning, we can prove the accuracy of the gradients of models, formalized in the following lemma.

Lemma 6.2. *Suppose that Assumption 6.1 holds. There exists a constant $\kappa_{grad} > 0$ such that*

$$\|\nabla f_i(x) - \nabla m_i^k(x)\| \leq \kappa_{grad}\Delta_k \tag{6.7}$$

holds for all $k \in \mathbb{N}$, $i \in \{1, ..., q\}$, and $x \in B_k$.

Function ω, defined by (6.6), is again generalized to incorporate models using the following formulation:

$$\omega_{m^k}(x) = -\min_{\|d\| \leq 1} \max_{i=1,...,q} \nabla m_i^k(x)^\top d. \tag{6.8}$$

The following lemma ensures the accuracy of ω_{m^k}, as an approximation to ω, when the trust-region radius is small enough.

Lemma 6.3. *In every iteration k,*

$$|\omega(x) - \omega_{m^k}(x)| \leq \kappa_{grad}\Delta_k$$

holds for all $x \in B_k$, where κ_{grad} is defined in Lemma 6.2.

Proof. For $x \in B_k$, according to (6.6), (6.7), and the fact that $\|d\| \leq 1$ we have

$$\begin{aligned}
-\omega(x) &= \min_{\|d\|\leq 1} \max_{i=1,\dots,q} \nabla f_i(x)^\top d \\
&= \min_{\|d\|\leq 1} \max_{i=1,\dots,q} \left\{(\nabla f_i(x) - \nabla m_i^k(x))^\top d + \nabla m_i^k(x)^\top d\right\} \\
&\leq \min_{\|d\|\leq 1} \max_{i=1,\dots,q} \left\{\|\nabla f_i(x) - \nabla m_i^k(x)\|\, \|d\| + \nabla m_i^k(x)^\top d\right\} \\
&\leq \min_{\|d\|\leq 1} \max_{i=1,\dots,q} \left\{\|\nabla f_i(x) - \nabla m_i^k(x)\| + \nabla m_i^k(x)^\top d\right\} \\
&\leq \min_{\|d\|\leq 1} \max_{i=1,\dots,q} \left\{\kappa_{grad}\Delta_k + \nabla m_i^k(x)^\top d\right\} \\
&= \kappa_{grad}\Delta_k + \min_{\|d\|\leq 1} \max_{i=1,\dots,q} \left\{\nabla m_i^k(x)^\top d\right\} \\
&= \kappa_{grad}\Delta_k - \omega_{m^k}(x).
\end{aligned}$$

Therefore $\omega_{m^k}(x) - \omega(x) \leq \kappa_{grad}\Delta_k$ holds. Equivalently, we can prove that $\omega(x) - \omega_{m^k}(x) \leq \kappa_{grad}\Delta_k$. So, $|\omega(x) - \omega_{m^k}(x)| \leq \kappa_{grad}\Delta_k$ holds. □

In particular, in every iteration k, we have

$$|\omega(x^k) - \omega_{m^k}(x^k)| \leq \kappa_{grad}\Delta_k.$$

This inequality inspires us to have the following assumption. This assumption is also presented by [33, Assumption 4.8]. What we insist on demonstrating is that when we are approaching a Pareto critical point, the attitudes of $\omega(x^k)$ and $\omega_{m^k}(x^k)$ are the same.

Assumption 6.2. *There exists a constant $\kappa_\omega > 0$ such that, for every $k \in \mathbb{N}$,*

$$|\omega(x^k) - \omega_{m^k}(x^k)| \leq \kappa_\omega \omega_{m^k}(x^k).$$

This assumption ensures that when the iteration point x^k is Pareto critical or close to criticality in the model space, the same applies to the main optimization problem.

Similar to Chapter 5, the functions $\phi(x)$ and $\phi_m^k(x)$ are defined by

$$\phi(x) = \max_{j=1,\dots,q} f_j(x),$$

and

$$\phi_m^k(x) = \max_{j=1,\dots,q} m_j^k(x).$$

To prove convergence, model minimization should provide a sufficient decrease at each iteration, here quantified using the function ϕ_m^k. Following [34], for the scalarization

step, we quantify the best model reduction obtained along a direction belonging to $\mathcal{D}(x)$, the set of directions associated with the solution of (6.6), within the trust-region B_k. For $i \in \{1, ..., q\}$, let $d_k^* \in \mathcal{D}(x_{sc}^{i,k})$ and compute α_k by solving

$$\min_{\alpha \geq 0} \{\phi_m^k(x_{sc}^{i,k} + \alpha d_k^*) : x_{sc}^{i,k} + \alpha d_k^* \in B_k\}. \tag{6.9}$$

The Pareto-Cauchy point is defined as

$$x_k^C = x_{sc}^{i,k} + d_k^C, \tag{6.10}$$

where $d_k^C := \alpha_k d_k^*$ and $B_k = B(x_{sc}^{i,k}, \Delta_{sc}^{i,k})$.

Lemma 6.4. *Let Assumptions 6.1 and 6.2 hold. There exists a constant $\bar{\kappa}_\phi \in (0,1)$ such that for $i \in \{1, ..., q\}$ and $k \in \mathbb{N}$, the Pareto-Cauchy point $x_k^C = x_{sc}^{i,k} + d_k^C$, defined by (6.10), satisfies*

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C) \geq \frac{1}{2}\bar{\kappa}_\phi \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}. \tag{6.11}$$

Proof. Since $d_k^* \in \mathcal{D}(x_{sc}^{i,k})$, we have $\|d_k^*\| \leq 1$, which implies that, for all $\alpha \in [0, \Delta_{sc}^{i,k}]$, $x_{sc}^{i,k} + \alpha d_k^* \in B_k$. Problem (6.9) can be reformulated as follows:

$$\max_{0 \leq \alpha \leq \Delta_{sc}^{i,k}} \{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*)\}$$

On the other hand, for all $\alpha \geq 0$, we have

$$\begin{aligned}\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*) &= \max_{j=1,...,q} m_j^k(x_{sc}^{i,k}) - \max_{j=1,...,q} m_j^k(x_{sc}^{i,k} + \alpha d_k^*) \\ &\geq -\max_{j=1,...,q} \alpha \nabla m_j^{k\top}(x_{sc}^{i,k}) d_k^* - \max_{j=1,...,q} \frac{1}{2}\alpha^2 d_k^{*\top} \nabla^2 m_j^k(x_{sc}^{i,k}) d_k^*.\end{aligned}$$

According to (6.8), Assumption 6.1, $\|d_k^*\| \leq 1$, and the Cauchy-Schwarz inequality we have

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*) \geq \alpha \omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2 \kappa_h.$$

Then, it is clear that

$$\max_{0 \leq \alpha \leq \Delta_{sc}^{i,k}} \{\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + \alpha d_k^*)\} \geq \max_{0 \leq \alpha \leq \Delta_{sc}^{i,k}} \left\{\alpha \omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2 \kappa_h\right\}.$$

In other words,

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k} + d_k^C) \geq \max_{0 \leq \alpha \leq \Delta_{sc}^{i,k}} \left\{\alpha \omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2 \kappa_h\right\}.$$

Let us consider the concave function g, defined by $g(\alpha) = \alpha \omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2 \kappa_h$, with unconstrained maximizer $\alpha^* = \dfrac{\omega_{m^k}(x_{sc}^{i,k})}{\kappa_h} \geq 0$, corresponding to the optimum value $g(\alpha^*) = \dfrac{1}{2}\dfrac{\omega_{m^k}(x_{sc}^{i,k})^2}{\kappa_h} \geq 0$. Two cases can occur:

- If $0 \leq \alpha^* \leq \Delta_{sc}^{i,k}$ then $\max\limits_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \left\{\alpha\omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h\right\} = \frac{1}{2}\frac{\omega_{m^k}(x_{sc}^{i,k})^2}{\kappa_h}$;
- If $\alpha^* > \Delta_{sc}^{i,k}$ then $\max\limits_{0\leq\alpha\leq\Delta_{sc}^{i,k}} \left\{\alpha\omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}\alpha^2\kappa_h\right\} = \Delta_{sc}^{i,k}\omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}(\Delta_{sc}^{i,k})^2\kappa_h$.

In this last case, since $\alpha^* = \frac{\omega_{m^k}(x_{sc}^{i,k})}{\kappa_h} > \Delta_{sc}^{i,k}$, we have $\Delta_{sc}^{i,k}\omega_{m^k}(x_{sc}^{i,k}) - \frac{1}{2}(\Delta_{sc}^{i,k})^2\kappa_h \geq \frac{1}{2}\Delta_{sc}^{i,k}\omega_{m^k}(x_{sc}^{i,k})$, resulting in

$$\begin{aligned}\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C) &\geq \min\left\{\frac{1}{2}\frac{\omega_{m^k}(x_{sc}^{i,k})^2}{\kappa_h}, \frac{1}{2}\Delta_{sc}^{i,k}\omega_{m^k}(x_{sc}^{i,k})\right\} \\ &= \frac{1}{2}\omega_{m^k}(x_{sc}^{i,k})\min\left\{\frac{\omega_{m^k}(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}.\end{aligned}$$

According to assumption 6.2, we have

$$\omega_{m^k}(x_{sc}^{i,k}) \geq \frac{1}{1+\kappa_\omega}\omega(x_{sc}^{i,k})$$

So for every iteration k,

$$\phi_m(x_{sc}^{i,k}) - \phi_m(x_k^C) \geq \frac{1}{2}\bar{\kappa}_\phi\omega(x_{sc}^{i,k})\min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}$$

holds, with $\bar{\kappa}_\phi = \frac{1}{(1+\kappa_\omega)^2} \in (0,1)$. □

Let $(x_{sc}^{i,k*}, t^*)$ represent the solution of Problem (5.4), with the quadratic polynomial models computed by numerical interpolation or using a minimum Frobenius norm approach. The subsequent lemma highlights an important characteristic of t^*. The proof is omitted since it is equal to the one of Lemma 5.4.

Lemma 6.5. *At each iteration k and for each $i \in \{1,\ldots,q\}$, $x_{sc}^{i,k}$ is not a Pareto critical point for $\min_{x\in B_k}(m_1^k(x),\ldots,m_q^k(x))$, if and only if $t^* < 0$.*

Now, we can gain a more precise understanding of the descent that occurs in ϕ_m^k.

Lemma 6.6. *Let Assumptions 6.1 and 6.2 hold. At each iteration k and for each $i \in \{1,\ldots,q\}$, there exists $j \in \mathbb{N}$ such that*

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \left(\frac{1}{2}\right)^j \bar{\kappa}_\phi\,\omega(x_{sc}^{i,k})\min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\},$$

where $(x_{sc}^{i,k*}, t^*)$ is the solution of Problem (5.4), considering quadratic polynomial interpolation or minimum Frobenius norm models, and $\bar{\kappa}_\phi$ is defined in Lemma 6.4.

Proof. Two different cases need to be analyzed. Assume that $x_{sc}^{i,k}$ is not Pareto critical. According to Lemma 6.5, t^* is strictly negative. So, for each $l \in \{1, ..., q\}$, we have

$$m_l^k(x_{sc}^{i,k}) - m_l^k(x_{sc}^{i,k*}) \geq -t^* > 0.$$

By considering $\phi_m^k(x) = \max_{l=1,...,q} m_l^k(x)$, it results

$$\phi_m^k(x_{sc}^{i,k}) - m_l^k(x_{sc}^{i,k*}) \geq m_l^k(x_{sc}^{i,k}) - m_l^k(x_{sc}^{i,k*}), \text{ for all } l \in \{1, ..., q\}.$$

Let j be the index such that $\phi_m^k(x_{sc}^{i,k*}) = m_j^k(x_{sc}^{i,k*})$. Then

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq m_j^k(x_{sc}^{i,k}) - m_j^k(x_{sc}^{i,k*}).$$

Hence,

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq -t^* > 0.$$

Since $\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C) \geq 0$, there must exist $j \in \mathbb{N}$ such that

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \left(\frac{1}{2}\right)^{j-1} (\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_k^C)).$$

Hence, considering (6.11), it implies

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \left(\frac{1}{2}\right)^{j} \bar{\kappa}_\phi \, \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}.$$

If $x_{sc}^{i,k}$ is Pareto critical, then $\omega(x_{sc}^{i,k}) = 0$. So, the right side of this inequality is equal to zero, and since $\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq 0$, the inequality holds. □

Subsequently, the convergence analysis follows a similar pattern and reasoning to that discussed in Chapter 5. To avoid unnecessary repetition and keep this chapter concise, we only present here the general structure of the analysis for readers specifically interested in the derivative-free case. For detailed proofs, we refer interested readers to Chapter 5.

The last three lemmas motivate us to consider the following assumption, stating that, at each scalarization step, a sufficient reduction in the model space is ensured.

Assumption 6.3. *There is a constant $\kappa_\phi \in (0, 1)$ such that at each iteration k, where the scalarization step is performed, for all $i \in \{1, ..., q\}$, we have*

$$\phi_m^k(x_{sc}^{i,k}) - \phi_m^k(x_{sc}^{i,k*}) \geq \kappa_\phi \, \omega(x_{sc}^{i,k}) \min\left\{\frac{\omega(x_{sc}^{i,k})}{\kappa_h}, \Delta_{sc}^{i,k}\right\}. \tag{6.12}$$

The subsequent lemma plays a crucial role in establishing convergence and presents an error bound for ϕ_m^k as an approximation of ϕ.

Lemma 6.7. *Let Assumption 6.1 hold. At every iteration k, the model ϕ_m^k is valid for ϕ at $x_{sc}^{i,k*}$, for all $i \in \{1, ..., q\}$, that is*

$$|\phi(x_{sc}^{i,k*}) - \phi_m^k(x_{sc}^{i,k*})| \leq \kappa_{fm} (\Delta_{sc}^{i,k})^2,$$

where κ_{fm} is defined in Lemma 6.1.

In the remaining analysis, we continue by categorizing the iterations of successful scalarization steps:

- The set of indexes of *successful scalarization step iterations* is denoted by

$$S = \{(k, i),\ k \in \mathbb{N}, i \in \{1, ..., q\} : k \text{ is a scalarization step iteration and } \rho_{sc}^{i,k} \geq \eta_{sc}^{1}\}.$$

- The set of indexes of *very successful scalarization step iterations* corresponds to

$$V = \{(k, i),\ k \in \mathbb{N}, i \in \{1, ..., q\} : k \text{ is a scalarization step iteration and } \rho_{sc}^{i,k} \geq \eta_{sc}^{2}\}.$$

For each linked sequence of points $\{x^k\}_{k\in K}$ generated by MOTRDFO, one of the two different scenarios will occur, as outlined below:

1. There exists $i \in \{1, ..., q\}$, such that for each $k \in \mathbb{N}$,

$$\Delta_{ep}^{i,k} > 0.$$

2. For each $i \in \{1, ..., q\}$, there exists $k_i \in \mathbb{N}$, such that for all $k > k_i$,

$$\Delta_{ep}^{i,k} = 0.$$

Remark 6.1. *In the first scenario, the linked sequence, updated at the extreme point step, matches the set of iterates generated by a single-objective derivative-free trust-region method, when applied to the objective function component f_i. Stationarity is then guaranteed for f_i and the corresponding limit point is a Pareto critical point. The proof is similar to the single-objective derivative-free trust-region case (see [8]).*

Remark 6.2. *In the second scenario, define $k_{ep} = \max\{k_i \mid i = 1, ..., q\}$. For $k > k_{ep}$, it holds $\Delta_{ep}^{i,k} = 0,\ \forall i = 1, ..., q$. Therefore, for $k > k_{ep}$, all points of the linked sequence have been generated in the scalarization step. The remaining analysis focuses on this situation.*

Two lemmas are presented below to clarify the behavior of MOTRDFO in cases where the current point is not Pareto critical.

Lemma 6.8. *Let Assumptions 6.1 and 6.3 hold. Suppose that at the scalarization step iteration k, for $i \in \{1, ..., q\}$, $x_{sc}^{i,k}$ is not a Pareto critical point and*

$$\Delta_{sc}^{i,k} \leq \frac{\kappa_\phi \omega(x_{sc}^{i,k})(1 - \eta_{sc}^{2})}{\kappa_v}, \tag{6.13}$$

with $\kappa_v = \max\{\kappa_{fm}, \kappa_h\}$. Then the pair (k, i) corresponds to a very successful scalarization step iteration, and $\Delta_{sc}^{i,k*} > \Delta_{sc}^{i,k}$.

The following lemma establishes that when $x_{sc}^{i,k}$ is not Pareto critical, the trust-region radius of the scalarization step cannot be arbitrarily small. Specifically, it must be bounded from below by a strictly positive constant.

Lemma 6.9. *Let Assumptions 6.1 and 6.3 hold, and consider the constant $\sigma > 0$. If $\omega(x_{sc}^{i,k}) \geq \sigma$ holds for the pair (k, i), with k a scalarization step iteration and $i \in \{1, ..., q\}$, then there is a constant $\Delta > 0$, depending on σ, such that $\Delta_{sc}^{i,k} \geq \Delta$.*

Taking into account Remarks 6.1 and 6.2, the subsequent lemma presents a convergence result for linked sequences of MOTRDFO, having only a finite number of successful iterations executed during the scalarization step.

Lemma 6.10. *Suppose that Assumptions 6.1 and 6.3 hold. Let $\{x^k\}_{k\in K}$ be a linked sequence generated by MOTRDFO at the scalarization step, with finitely many successful iterations at the scalarization step. Then this linked sequence converges to a Pareto critical point.*

The following lemma elucidates the convergence analysis when a linked sequence encompasses an infinite number of distinct points generated during the scalarization step.

Lemma 6.11. *Suppose that Assumptions 6.1 and 6.3 hold. Let $\{x^k\}_{k\in K}$ be a linked sequence of points generated by MOTRDFO, with infinitely many successful iterations at the scalarization step. Then*

$$\liminf_{k\to+\infty;\, k\in K} \omega(x^k) = 0.$$

We are now prepared to establish the main result for the linked sequences generated by MOTRDFO.

Theorem 6.1. *Let Assumptions 6.1 and 6.3 hold. For every linked sequence of points $\{x^k\}_{k\in K}$ generated by MOTRDFO, we have*

$$\lim_{k\to+\infty;\, k\in K} \omega(x^k) = 0.$$

6.3 Numerical results

To demonstrate the efficiency and robustness of MOTRDFO, we conducted a comparative analysis of its performance against other multiobjective derivative-free optimization solvers that intrinsically attempt to generate approximations to the complete Pareto front of problems. Our goal is to illustrate that MOTRDFO is competitive. For this purpose, three solvers, which are based on a wide variety of techniques in multiobjective derivative-free optimization, were selected as follows:

- MOIF which proposes an implicit filtering algorithm for multiobjective derivative-free optimization [6];
- BoostDMS which uses polynomial models in the multiobjective framework of directional direct search [4];
- DMultiMADS which is based on mesh adaptive direct multisearch for black-box multiobjective optimization [3].

The codes for MOTRDFO, MOIF, and BoostDMS were implemented in MATLAB (version R2021b was considered). The codes for DMultiMADS were implemented in Julia (version 1.7.2 was considered).

The minimization subproblems of MOTRDFO, defined at the extreme point and scalarization steps, were solved with the MATLAB function `fmincon.m`.

While MOTRDFO was initially outlined and analyzed for unconstrained multiobjective optimization, the algorithmic description can be readily adapted to accommodate bound constraints. This can be achieved by incorporating these constraints into the subproblems that need to be solved.

Test problems

As a test set, we utilized 54 twice continuously differentiable bound constrained multiobjective optimization problems, available at

`https://docentes.fct.unl.pt/algb/pages/problems-collections`,

encompassing a range of variables from 1 to 30, involving 2 or 3 objective function components. A complete list of the problems along with their respective dimensions is presented in Table 5.1.

Numerical settings

MOTRDFO was run with the parameters $\mu_1 = 0.5$, $\mu_2 = 2$, $\eta_{ep}^1 = \eta_{sc}^1 = 0.001$, $\eta_{ep}^2 = \eta_{sc}^2 = 0.9$, $\Delta_{ep}^{init} = (1, \ldots, 1)^\top \in \mathbb{R}^q$, and $\Delta_{sc}^{init} = 1$. Regarding the update of the trust-region radius during very successful iterations, it was only increased if the trust region's boundary was reached. In this case, a maximum value of $\Delta^{max} = \|u - l\|/2$ was allowed, where u and l represent the upper and lower bounds of the problem variables, respectively. The algorithm was initialized with a single point, specifically the centroid of the feasible region. For each problem, the approximation of the Pareto front generated by MOTRDFO corresponds to all the current feasible nondominated points, which are stored in the list L.

The parameters which were used to build models, as described in Section 6.1.1, were configured with $r = 10$ and $condtol = 1000$.

All default settings were retained for MOIF, BoostDMS, and DMultiMADS solvers.

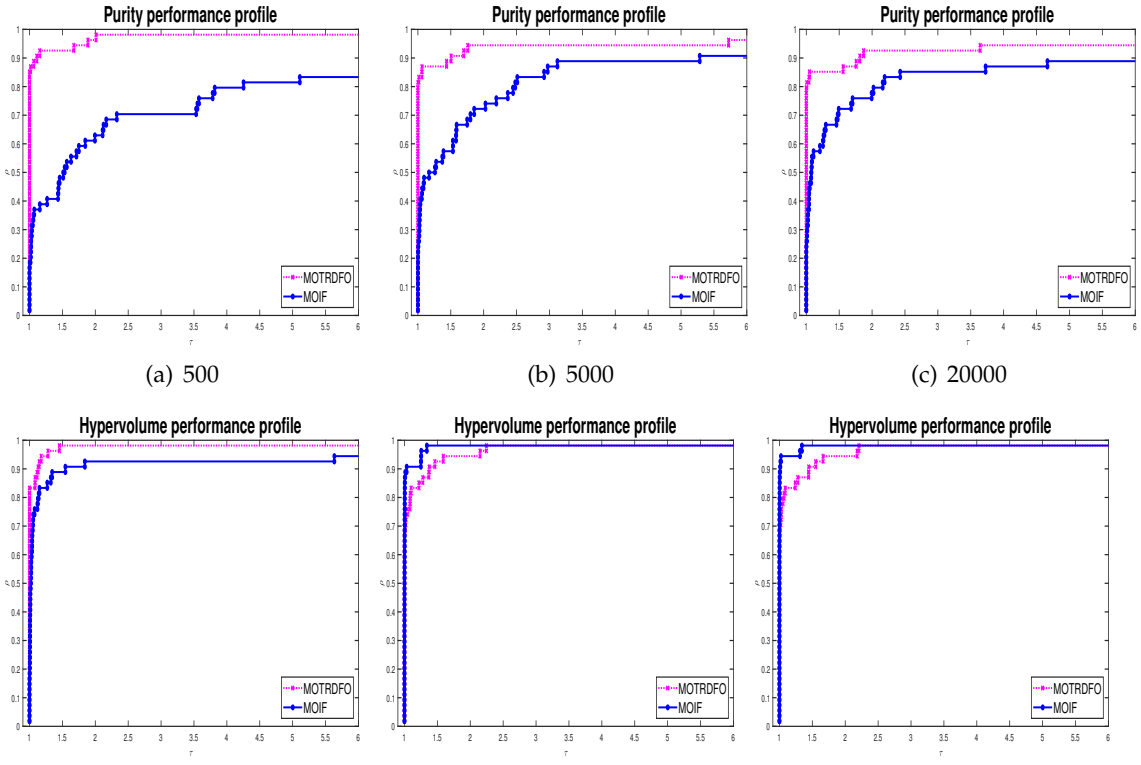
Regarding MOTRDFO, the minimum trust-region radius values $\Delta_{ep}^{min} = \Delta_{sc}^{min} = 10^{-3}$ were set as stopping criterion, with componentwise consideration for Δ_{ep}. Three distinct budgets were considered in terms of function evaluations, with values of 500, 5000, and 20000.

In terms of stopping criteria, we kept all the default values for the MOIF, BoostDMS, and DMultiMADS solvers. However, we experimented with three different budgets of function evaluations.

6.3.1 Comparing MOTRDFO with other algorithms

In order to demonstrate the competitiveness of MOTRDFO in comparison to other solvers, we utilized performance profiles, described in Section 4.5, by considering purity, hypervolume, and spread (Γ and Δ) metrics.

The results comparing MOTRDFO and MOIF can be found in Figures 6.1 and 6.2. It is clear the advantage of MOTRDFO over MOIF in terms of purity and Γ metrics, for both efficiency and robustness, independent of the allowed budget of function evaluations. Regarding hypervolume, for smaller budgets of function evaluations, MOTRDFO is also competitive both for efficiency and robustness. When this budget grows up, MOIF performs more efficiently, considering the hypervolume metric, although both solvers are comparably as robust as each other. MOTRDFO also outperforms MOIF, with remarkably good results, in terms of efficiency, considering the uniformity of the distribution of points across the approximation to the Pareto front.

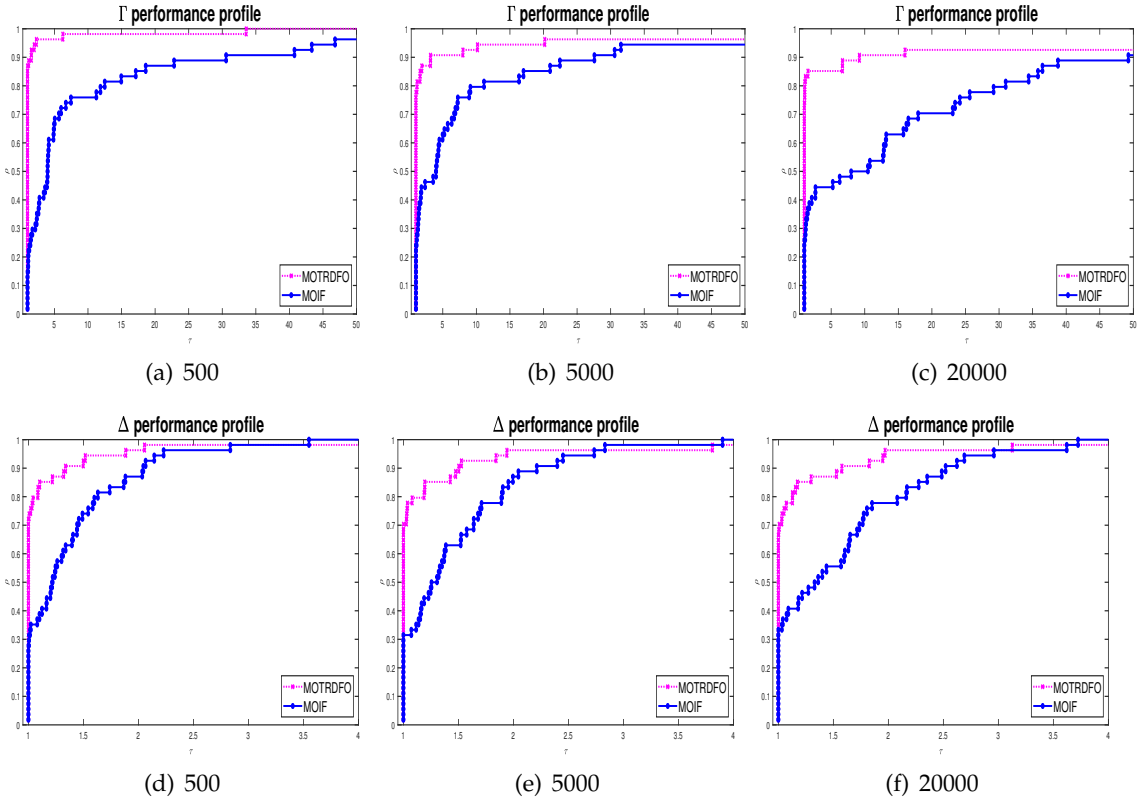


(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 6.1: Comparing MOTRDFO and MOIF based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

Figures 6.3 and 6.4 report the comparison between MOTRDFO and BoostDMS. Clearly, MOTRDFO outperforms BoostDMS in terms of efficiency for purity and spread metrics. The performance profiles are also very close in terms of robustness for purity and Γ metrics, with the exception of purity for small budget of function evaluations, where MOTRDFO again presents a better performance against BoostDMS. Considering the hypervolume



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

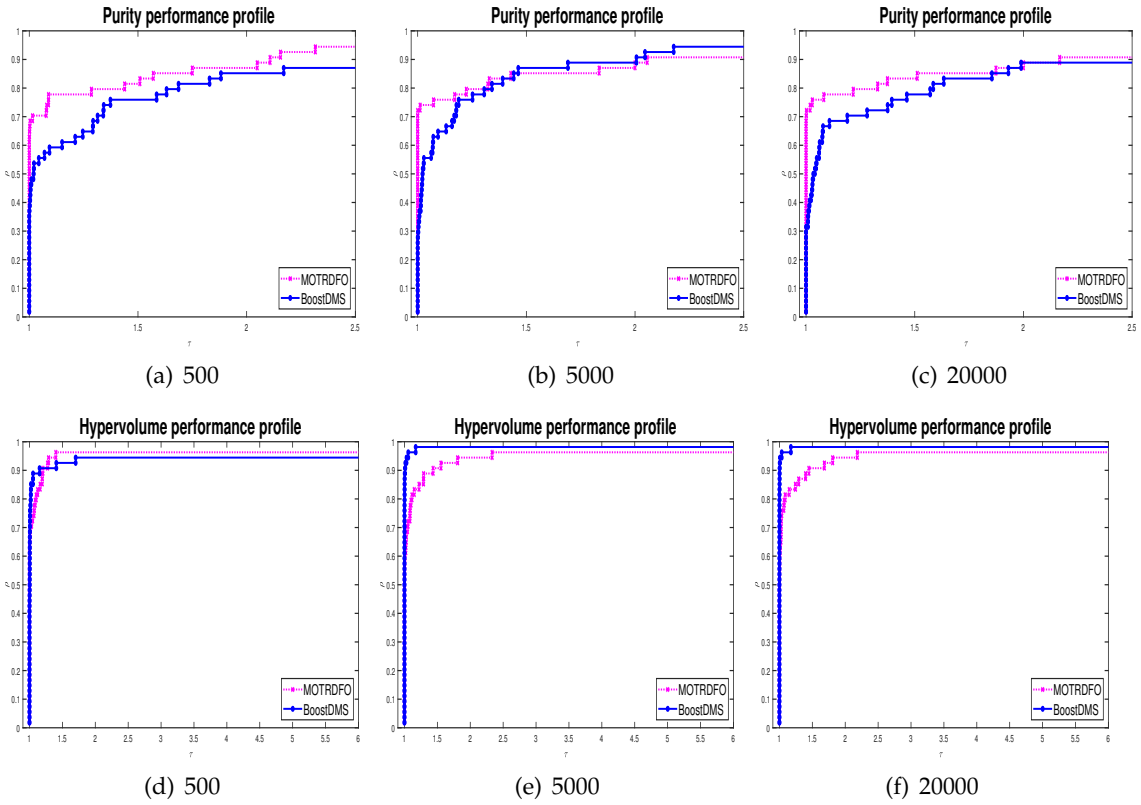
Figure 6.2: Comparing MOTRDFO and MOIF based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

metric, BoostDMS is more efficient, where the results are close in terms of robustness.

Finally, MOTRDFO's numerical performance was assessed against DMultiMADS, where Figures 6.5 and 6.6 report the obtained results. It is clear the advantage of MOTRDFO over DMultiMADS, in terms of efficiency for purity and spread metrics. Regarding hypervolume, DMultiMADS is more efficient, where the results are close in terms of robustness. With reference to robustness in the purity case, for the smaller budget of function evaluations, MOTRDFO is also competitive, but when this budget grows up DMultiMADS performs better. The results are very close in terms of robustness corresponding to the Γ metric, the exception appears in the case of large budget of function evaluations where MOTRDFO performs better. With respect to the Δ metric, the advantage of MOTRDFO is also clear.

Table 6.1 provides the number of feasible nondominated points obtained by MOTRDFO and MOIF for a maximum budget of 5000 function evaluations. The table combines the lists of feasible nondominated points generated by both solvers for each problem, with dominated points being removed from the final count. Tables 6.2 and 6.3 present the equivalent results for MOTRDFO compared to BoostDMS and DMultiMADS, respectively, using the same budget of function evaluations.

Figures 6.7 and 6.8 depict the final approximations of the Pareto fronts achieved by MOTRDFO and MOIF for two biobjective and two triobjective problems, respectively. A

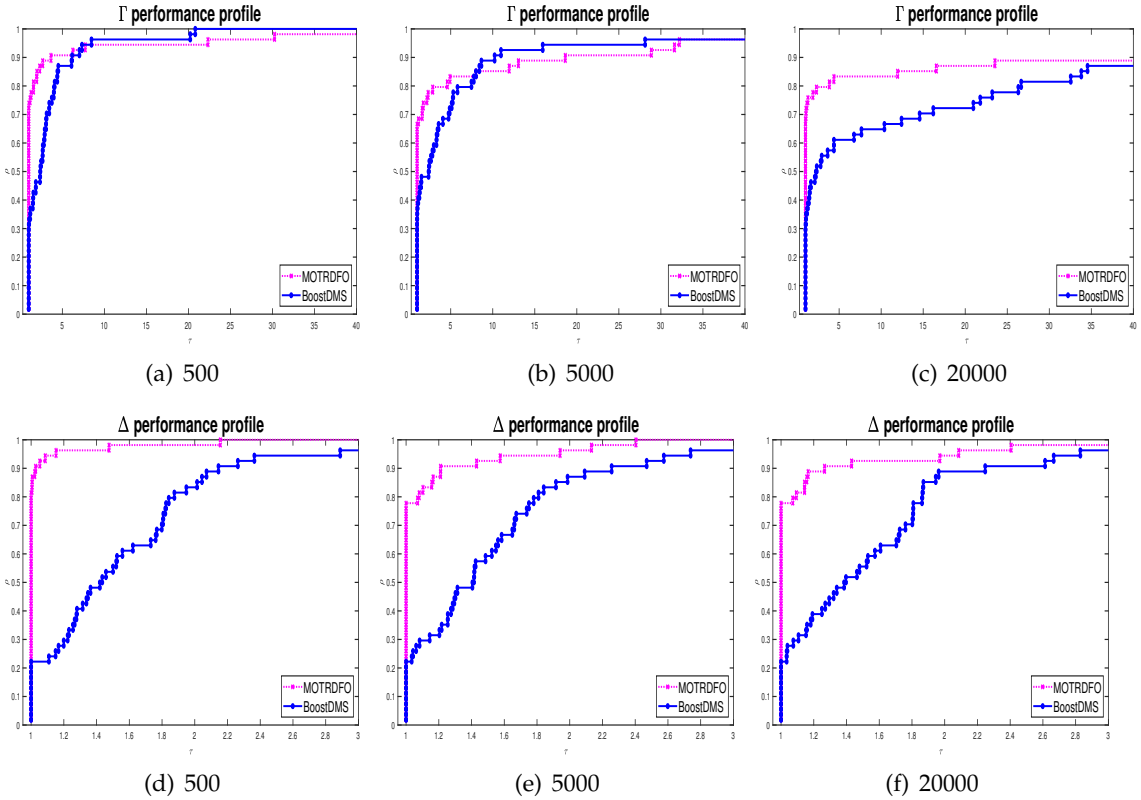


(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 6.3: Comparing MOTRDFO and BoostDMS based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.

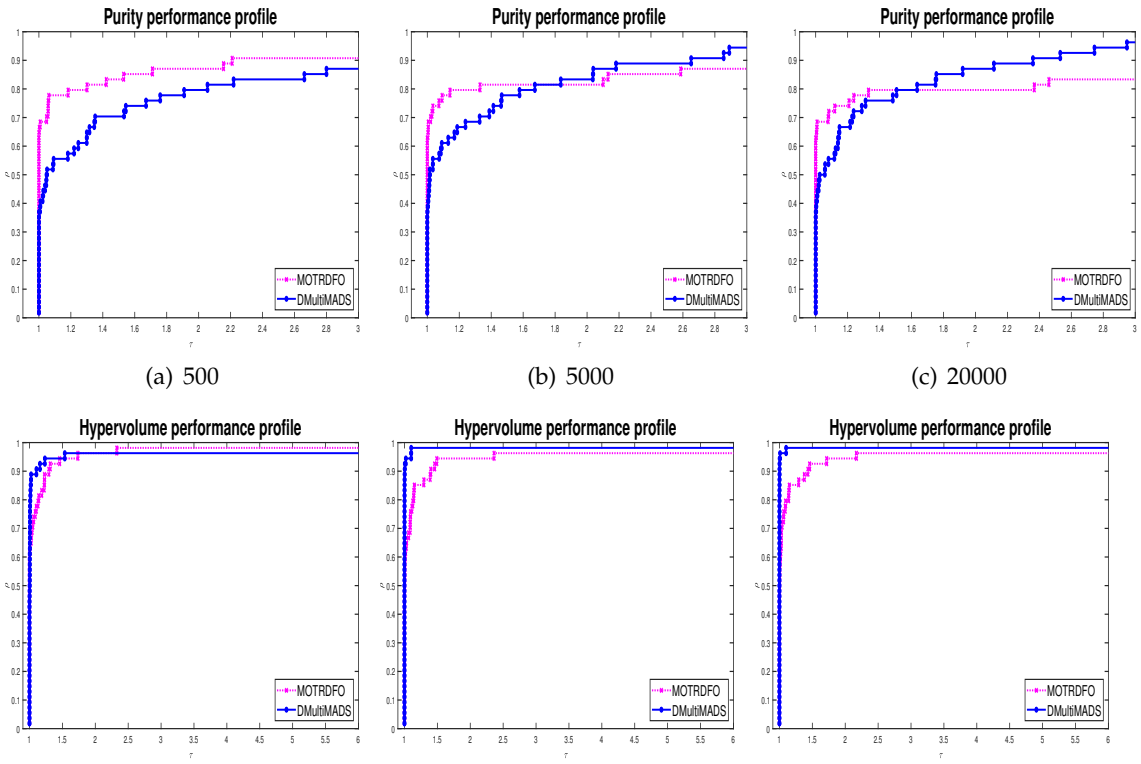
maximum budget of 5000 function evaluations was taken into account. Similarly, Figures 6.9 and 6.10 present the corresponding results obtained by MOTRDFO and BoostDMS. Additionally, we provide the results for the Pareto fronts obtained by MOTRDFO and DMultiMADS through Figures 6.11 and 6.12.



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

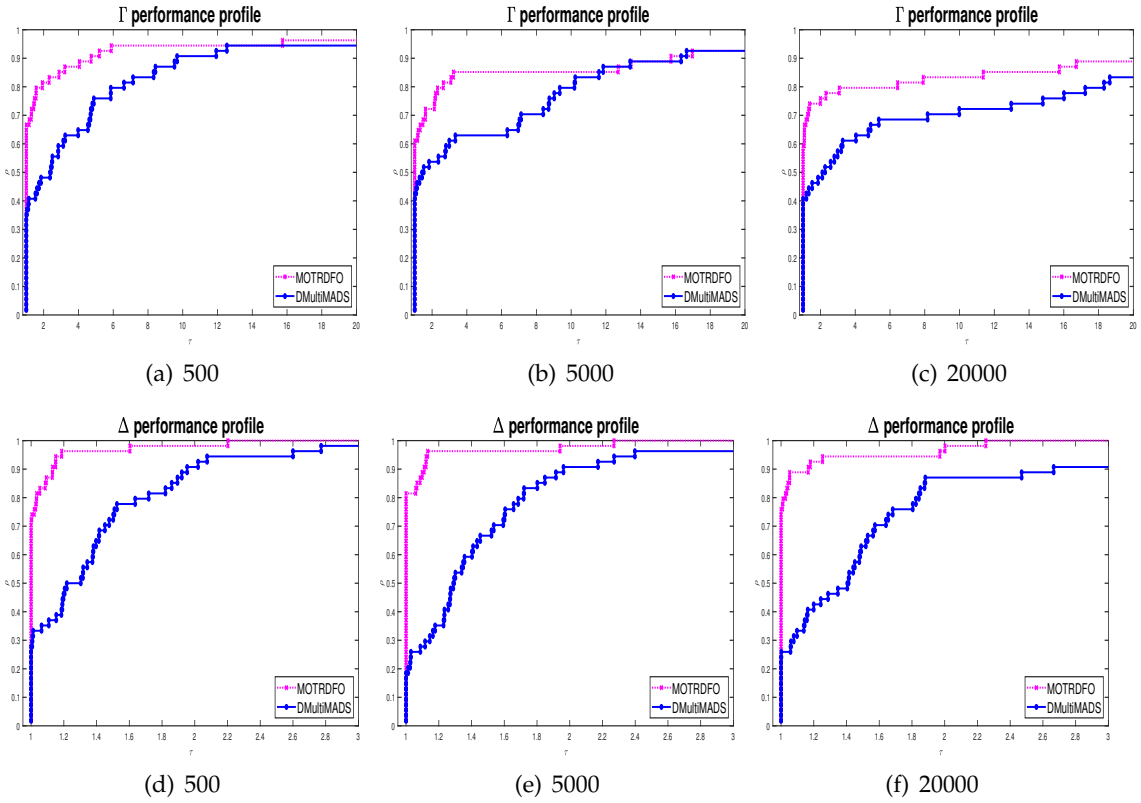
Figure 6.4: Comparing MOTRDFO and BoostDMS based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

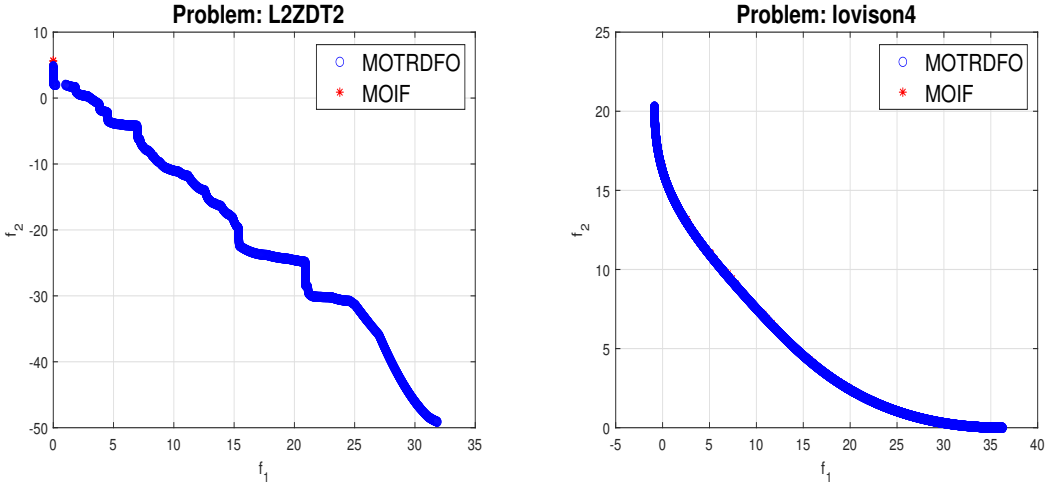
Figure 6.5: Comparing MOTRDFO and DMultiMADS based on performance profiles of purity and hypervolume metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.



(a) 500 (b) 5000 (c) 20000

(d) 500 (e) 5000 (f) 20000

Figure 6.6: Comparing MOTRDFO and DMultiMADS based on performance profiles of Γ and Δ metrics. Budgets of 500, 5000, and 20000 function evaluations were allowed.



(a) Problem: L2ZDT2 (b) Problem: lovison4

Figure 6.7: Approximations to the Pareto fronts of problems L2ZDT2 and lovison4, obtained by solvers MOTRDFO and MOIF, for a budget of 5000 function evaluations.

Table 6.1: Number of feasible nondominated points in the final approximation of the Pareto front, generated for each problem by MOTRDFO and MOIF, considering a budget of 5000 function evaluations.

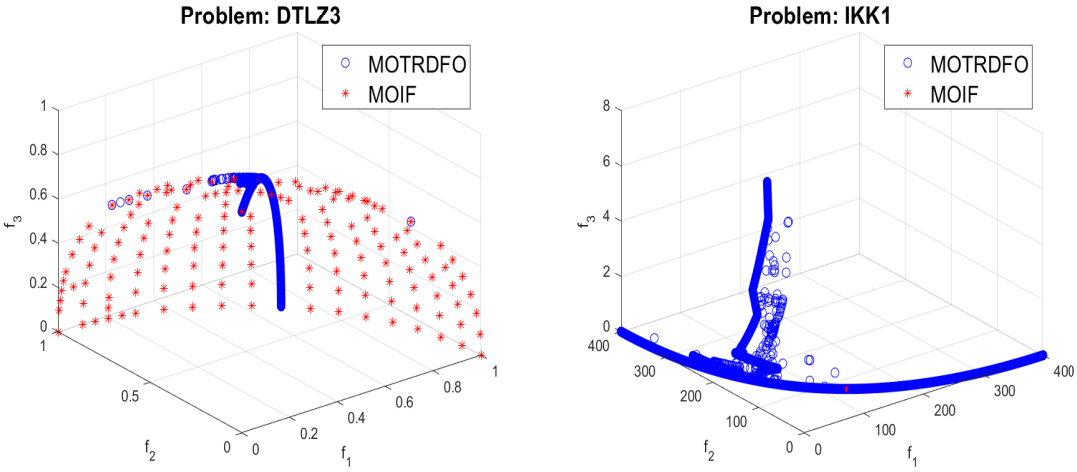
Problem	**MOTRDFO**	**MOIF**	**Problem**	**MOTRDFO**	**MOIF**
BK1	3478	159	CL1	1649	151
Deb41	2830	415	Deb513	1245	427
Deb521b	4674	355	DG01	2455	915
DPAM1	1	76	DTLZ1	2136	249
DTLZ1n2	2909	16	DTLZ2	1445	150
DTLZ2n2	4126	307	DTLZ3	2155	146
DTLZ3n2	2294	33	DTLZ4	0	18
DTLZ4n2	1155	248	DTLZ6	1505	79
DTLZ6n2	2183	129	ex005	4749	740
Far1	991	262	Fonseca	3515	13
IKK1	4607	615	IM1	4699	747
Jin1	3078	203	Jin3	4921	491
L2ZDT2	1620	0	L3ZDT2	0	1
lovison1	3354	227	lovison2	2321	510
lovison3	2093	281	lovison4	3553	146
lovison5	302	100	lovison6	395	46
LRS1	4003	108	MHHM1	3698	94
MHHM2	4358	860	MLF1	4239	389
MLF2	4025	470	MOP1	4137	16
MOP2	1063	63	MOP3	968	575
MOP5	2535	531	MOP6	1245	427
MOP7	2579	762	SK1	4988	1252
SK2	931	40	SP1	3652	98
SSFYY1	3533	132	SSFYY2	2389	630
TKLY1	318	228	VFM1	3339	1442
VU1	3008	164	VU2	4945	835
ZDT2	903	24	ZLT1	608	113

Table 6.2: Number of feasible nondominated points in the final approximation of the Pareto front, generated for each problem by MOTRDFO and BoostDMS, considering a budget of 5000 function evaluations.

Problem	**MOTRDFO**	**BoostDMS**	**Problem**	**MOTRDFO**	**BoostDMS**
BK1	3476	1947	CL1	524	1163
Deb41	2820	1559	Deb513	1283	171
Deb521b	4914	680	DG01	2483	471
DPAM1	1	364	DTLZ1	2135	254
DTLZ1n2	3200	185	DTLZ2	1456	578
DTLZ2n2	4181	1368	DTLZ3	2170	42
DTLZ3n2	2536	236	DTLZ4	0	56
DTLZ4n2	1164	26	DTLZ6	1212	193
DTLZ6n2	2183	341	ex005	4747	997
Far1	940	1622	Fonseca	2781	2054
IKK1	4625	2972	IM1	4751	1853
Jin1	3125	2160	Jin3	4931	690
L2ZDT2	1510	1	L3ZDT2	0	1
lovison1	3352	2041	lovison2	2321	697
lovison3	2092	1549	lovison4	3550	913
lovison5	175	1150	lovison6	262	723
LRS1	3781	212	MHHM1	3761	16
MHHM2	4299	3608	MLF1	4959	1054
MLF2	4013	961	MOP1	4958	1277
MOP2	464	733	MOP3	1220	733
MOP5	2539	603	MOP6	1283	171
MOP7	2555	1652	SK1	3750	4787
SK2	957	532	SP1	3649	882
SSFYY1	3527	1493	SSFYY2	2484	1030
TKLY1	360	813	VFM1	3418	4253
VU1	2987	1615	VU2	4945	1516
ZDT2	863	137	ZLT1	610	850

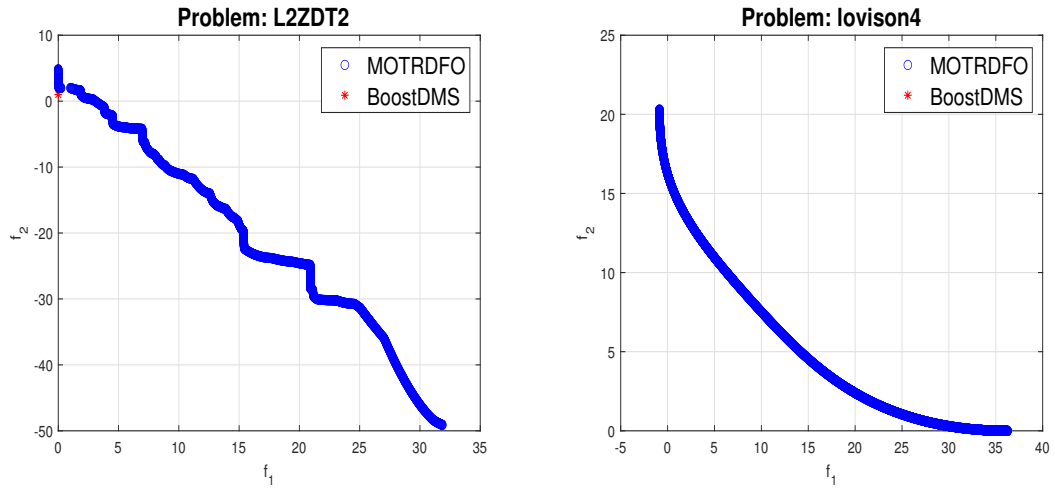
Table 6.3: Number of feasible nondominated points in the final approximation of the Pareto front, generated for each problem by MOTRDFO and DMultiMADS, considering a budget of 5000 function evaluations.

Problem	**MOTRDFO**	**DMultiMADS**	**Problem**	**MOTRDFO**	**DMultiMADS**
BK1	3481	397	CL1	702	651
Deb41	0	526	Deb513	1283	1582
Deb521b	4912	1062	DG01	2319	4742
DPAM1	2	162	DTLZ1	2152	54
DTLZ1n2	3398	830	DTLZ2	1456	111
DTLZ2n2	4331	405	DTLZ3	2173	3
DTLZ3n2	2760	760	DTLZ4	0	43
DTLZ4n2	1119	755	DTLZ6	1423	224
DTLZ6n2	2181	858	ex005	4749	1090
Far1	767	811	Fonseca	3228	576
IKK1	4625	2020	IM1	4757	936
Jin1	3137	586	Jin3	4930	988
L2ZDT2	1510	1	L3ZDT2	0	1
lovison1	3355	369	lovison2	2317	1196
lovison3	2092	1138	lovison4	3554	145
lovison5	172	813	lovison6	185	693
LRS1	4380	920	MHHM1	1326	1953
MHHM2	4338	2259	MLF1	0	4962
MLF2	4016	1038	MOP1	4906	4889
MOP2	472	712	MOP3	981	1347
MOP5	2535	1198	MOP6	1283	1582
MOP7	2600	1918	SK1	3750	4930
SK2	104	794	SP1	3652	352
SSFYY1	3531	345	SSFYY2	2481	4926
TKLY1	324	217	VFM1	3450	3232
VU1	3005	1209	VU2	4942	2071
ZDT2	909	121	ZLT1	611	70



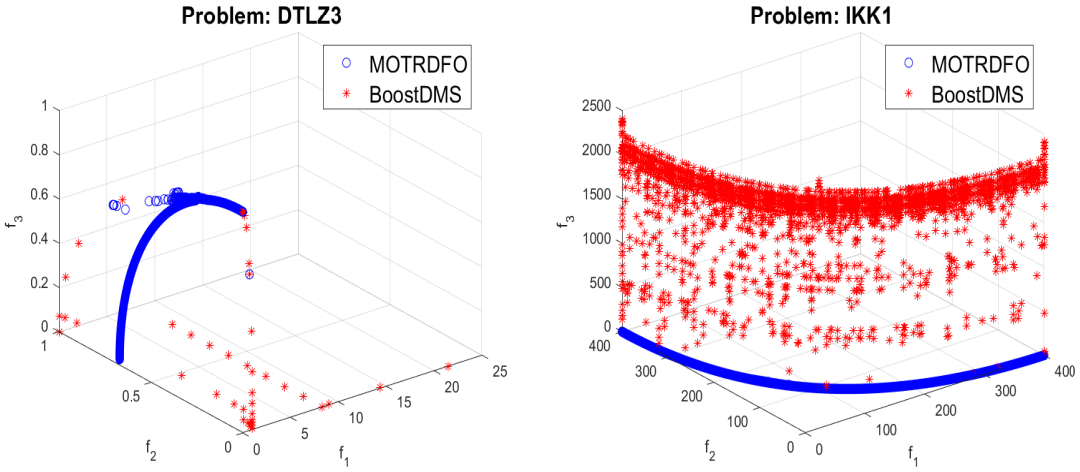
(a) Problem: DTLZ3 (b) Problem: IKK1

Figure 6.8: Approximations to the Pareto fronts of problems DTLZ3 and IKK1, obtained by solvers MOTRDFO and MOIF, for a budget of 5000 function evaluations.



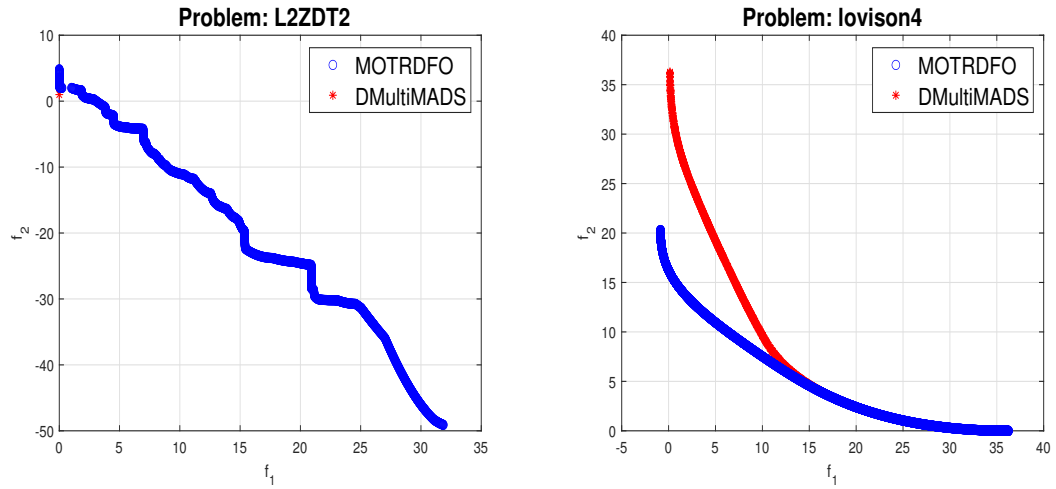
(a) Problem: L2ZDT2 (b) Problem: lovison4

Figure 6.9: Approximations to the Pareto fronts of problems L2ZDT2 and lovison4, obtained by solvers MOTRDFO and BoostDMS, for a budget of 5000 function evaluations.



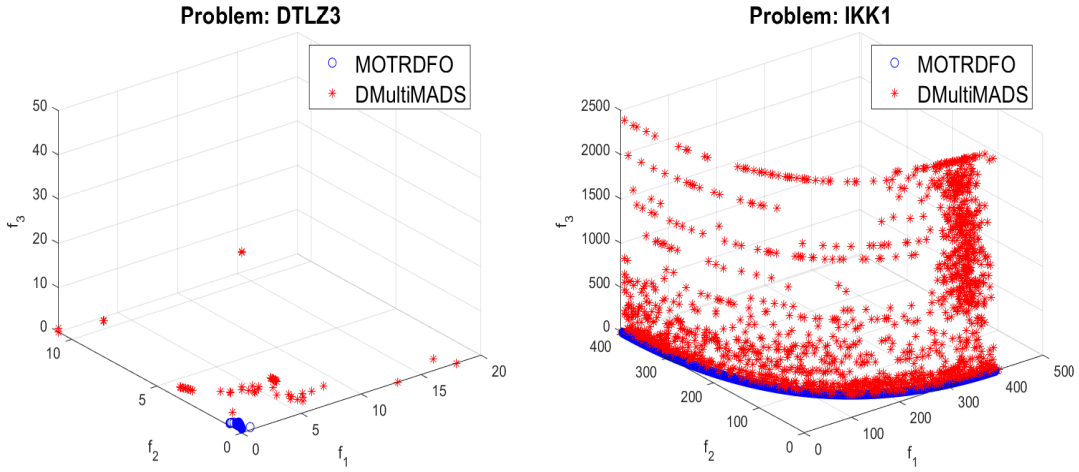
(a) Problem: DTLZ3 (b) Problem: IKK1

Figure 6.10: Approximations to the Pareto fronts of problems DTLZ3 and IKK1, obtained by solvers MOTRDFO and BoostDMS, for a budget of 5000 function evaluations.



(a) Problem: L2ZDT2 (b) Problem: lovison4

Figure 6.11: Approximations to the Pareto fronts of problems L2ZDT2 and lovison4, obtained by solvers MOTRDFO and DMultiMADS, for a budget of 5000 function evaluations.



(a) Problem: DTLZ3

(b) Problem: IKK1

Figure 6.12: Approximations to the Pareto fronts of problems DTLZ3 and IKK1, obtained by solvers MOTRDFO and DMultiMADS, for a budget of 5000 function evaluations.

7

Conclusions and open questions

This thesis introduced the MOTR algorithm, a novel approach based on trust-region methods, aimed at computing approximations of the complete Pareto front for multiobjective optimization problems. The algorithm comprises two key steps: the extreme point step and the scalarization step, executed alternately. In the extreme point step, the algorithm strives to reach the extreme points of the Pareto front, while the scalarization step focuses on reducing large gaps within the Pareto front.

To address the task of reducing the size of gaps within the Pareto front, a novel strategy was implemented within the scalarization step. This strategy involved the computation of middle points, which were carefully computed to effectively close the gaps and improve the overall coverage of the Pareto front, by solving appropriate scalarization problems.

The algorithm MOTR effectively addressed the challenges associated with conflicting objective function components by individually considering each objective function component in each extreme point or scalarization step. This approach allowed the algorithm to handle the conflicting nature of the objectives and make informed decisions based on their individual contributions.

By harnessing the derivative information of the objective function, the algorithm employed the computation and minimization of Taylor models throughout its execution. This approach allowed for the generation of high-quality models that were closely aligned with the true objective function components. The utilization of Taylor models enabled the algorithm to capture and incorporate valuable information regarding the behavior and characteristics of the objective function, enhancing the accuracy and fidelity of the computed approximations. This utilization of derivative information played a critical role in improving the overall quality and reliability of the algorithm's results.

Considering the computationally expensive nature of the objective functions, special attention was given to managing the number of function evaluations in order to optimize the algorithm's performance. Efforts were made to implement control mechanisms that effectively limited the frequency of function evaluations without compromising the accuracy or quality of the results. By carefully managing the number of function evaluations, the algorithm could strike a balance between computational efficiency and

achieving satisfactory approximations of the complete Pareto front. This optimization strategy played a vital role in enhancing the overall performance of the algorithm, making it more efficient and effective in solving multiobjective optimization problems with expensive objective functions.

A comprehensive convergence analysis was conducted on linked sequences of points generated by the MOTR algorithm. This analysis provided valuable insights into the behavior and properties of the algorithm's solution process. The analysis established a significant result, demonstrating that any limit point attained by these linked sequences of points corresponds to a Pareto critical point. In other words, as the algorithm iteratively progresses and generates these linked sequences of points, the convergence toward Pareto critical points is guaranteed. This finding reinforces the reliability and effectiveness of the MOTR algorithm in identifying and approximating points on the Pareto front, contributing to its credibility as a robust solution approach for multiobjective optimization problems.

The significance and effectiveness of each step within the MOTR algorithm were substantiated through numerical experiments. These experiments shed light on the crucial role played by each component of MOTR in ensuring robust numerical performance. The findings underscored that the algorithm's overall performance heavily relies on the successful execution and integration of these individual steps. Consequently, the numerical experiments reinforced the notion that the completeness and proper functioning of each component are indispensable in guaranteeing the overall efficacy of MOTR in approximating the complete Pareto front of multiobjective optimization problems.

Furthermore, to assess the competitiveness of the MOTR algorithm, a thorough performance comparison was conducted against a state-of-the-art multiobjective optimization solver. The selected benchmark solver inherently aims to generate approximations of the complete Pareto front for a given problem. The results of this comparison showcased the algorithm's competitiveness, demonstrating its ability to produce comparable or superior approximations when compared to the established benchmark solver. This validation further validates the robustness and efficiency of the MOTR algorithm as a cutting-edge solution approach for multiobjective optimization problems.

Subsequently, the original algorithm was modified, resulting in MOTRDFO, a comprehensive approach based on trust-region methods for computing approximations of complete Pareto fronts in multiobjective derivative-free optimization problems. In this case, the availability of derivatives within the algorithm was assumed to be nonexistent, and estimation was not performed. Consequently, Taylor models could not be employed. Instead, a novel strategy based on polynomial interpolation and minimum Frobenius norm approaches was applied to build accurate and reliable models, even in the absence of derivative information.

The convergence analysis for MOTRDFO was conducted, taking into account the challenge of lacking derivative information for the objective function, which added complexity to the analysis. However, despite this challenge, it was successfully demonstrated that each limit point derived from the algorithm's linked sequences of points represents a Pareto

critical point. This result highlights the algorithm's ability to converge towards critical points on the Pareto front, even in the absence of derivative information. The rigorous convergence analysis solidifies the credibility and effectiveness of MOTRDFO as a robust solution method for multiobjective derivative-free optimization problems, reaffirming its capability to provide accurate approximations of the complete Pareto front.

The numerical competitiveness of MOTRDFO was evaluated through computational experiments, comparing its performance to other well-established state-of-the-art solvers for multiobjective derivative-free optimization. The selected solvers, like MOTRDFO, aim to generate approximations of the complete Pareto front for a given problem. The results of the comparative analysis provided strong evidence of MOTRDFO's ability to deliver competitive results and established its position as a competitive solver for multiobjective derivative-free optimization problems, capable of generating comprehensive approximations of the complete Pareto fronts.

In terms of future research directions, there are several avenues to be explored. One potential area of study is the extension of the proposed algorithm to handle constrained multiobjective derivative-based or derivative-free optimization problems, encompassing both linear and nonlinear constraints. The incorporation of constraints adds another layer of complexity and requires effective techniques to efficiently handle the trade-off between conflicting objectives while satisfying the imposed constraints.

The exploration of quasi-Newton approaches in model computation is also worth being investigated. In derivative-based optimization, these approaches would avoid the use of the Hessian matrix and could be particularly relevant in a derivative-free setting, only requiring a linear model, computed through numerical interpolation.

Another promising avenue is the application of the developed methods to solve practical multiobjective derivative-based or derivative-free optimization problems in diverse domains of science and industry. By adapting the algorithms to specific problem domains, valuable insights can be gained, and real-world challenges can be effectively addressed. Exploring these future directions holds great potential for advancing the field of multiobjective optimization and broadening the practical applicability of the proposed algorithms.

Bibliography

[1] M. A. T. Ansary and G. Panda. "A globally convergent SQCQP method for multiobjective optimization problems". In: *SIAM J. Optim.* 31 (2021), pp. 91–113 (cit. on p. 36).

[2] C. Audet and W. Hare. *Derivative-Free and Blackbox Optimization*. Cham, Switzerland: Springer, 2017 (cit. on pp. 6, 23, 30, 31).

[3] J. Bigeon, S. Le Digabel, and L. Salomon. "DMulti-MADS: Mesh adaptive direct multisearch for bound-constrained blackbox multiobjective optimization". In: *Comput. Optim. Appl.* 79 (2021), 301–338 (cit. on pp. 37, 83).

[4] C. P. Brás and A. L. Custódio. "On the use of polynomial models in multiobjective directional direct search". In: *Comput. Optim. Appl* 77 (2020), pp. 897–918 (cit. on pp. 37, 83).

[5] G. A. Carrizo, P. A. Lotito, and M. C. Maciel. "Trust region globalization strategy for the nonconvex unconstrained multiobjective optimization problem". In: *Math. Program., Ser. A* 159 (2016), pp. 339–369 (cit. on pp. 36, 37, 43).

[6] G. Cocchi, G. Liuzzi, A. Papini, and M. Sciandrone. "An implicit filtering algorithm for derivative-free multiobjective optimization with box constraints". In: *Comput. Optim. Appl.* 69 (2018), pp. 267–296 (cit. on pp. 37, 83).

[7] A. R. Conn, N. I. M. Gould, and Ph. L. Toint. *Trust-Region Methods*. MOS-SIAM Ser. Optim. Philadelphia, USA: SIAM, 2000 (cit. on pp. 26, 28, 29, 53).

[8] A. R. Conn, K. Scheinberg, and L. N. Vicente. *Introduction to Derivative-free Optimization*. MPS-SIAM Series on Optimization. Philadelphia, USA: SIAM, 2009 (cit. on pp. 6, 9, 12–17, 19–23, 31, 32, 37, 74–76, 82).

[9] A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente. "Direct multisearch for multiobjective optimization". In: *SIAM J. Optim.* 21 (2011), pp. 1109–1140 (cit. on pp. 37, 38).

[10] A. L. Custódio, H. Rocha, and L. N. Vicente. "Incorporating minimum Frobenius norm models in direct search". In: *Comput. Optim. Appl.* 46 (2010), pp. 265–278 (cit. on pp. 22, 75).

[11] I. Das and J. E. Dennis. "A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems". In: *Struct. Multidiscip. Optim.* 14 (1997), pp. 63–69 (cit. on p. 36).

[12] I. Das and J. E. Dennis. "Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems". In: *SIAM J. Optim.* 8 (1998), pp. 631–657 (cit. on p. 36).

[13] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. "A fast and elitist multiobjective genetic algorithm: NSGA-II". In: *IEEE T. Evolut. Comput.* 6 (2002), pp. 182–197 (cit. on p. 39).

[14] E. D. Dolan and J. J. Moré. "Benchmarking optimization software with performance profiles". In: *Math. Program.* 91 (2002), pp. 201–213 (cit. on pp. 37, 59).

[15] G. Eichfelder. *Adaptive Scalarization Methods in Multiobjective Optimization*. Vector Optimization. Heidelberg, Germany: Springer, 2008 (cit. on p. 36).

[16] G. Fasano, J. L. Morales, and J. Nocedal. "On the geometry phase in model-based algorithms for derivative-free optimization". In: *Optim. Methods Softw.* 24 (2009), 145–154 (cit. on p. 32).

[17] J. Fliege, L. M. G. Drummond, and B. F. Svaiter. "Newton's method for multiobjective optimization". In: *SIAM J. Optim.* 20 (2009), pp. 602–626 (cit. on p. 36).

[18] J. Fliege and B. F. Svaiter. "Steepest descent methods for multicriteria optimization". In: *Math. Methods Oper. Res.* 51 (2000), pp. 479–494 (cit. on pp. 35, 36).

[19] J. Fliege and A. I. F. Vaz. "A method for constrained multiobjective optimization based on SQP techniques". In: *SIAM J. Optim.* 26 (2016), pp. 2091–2119 (cit. on pp. 36, 57, 60, 61).

[20] I. Griva, S. G. Nash, and A. Sofer. *Linear and Nonlinear Optimization, Second Edition*. Philadelphia, USA: SIAM, 2009 (cit. on pp. 6–8, 11, 23–26, 29, 53).

[21] J. Jahn. *Vector Optimization*. Berlin: Springer, 2011 (cit. on p. 35).

[22] João M. Lourenço. *The NOVAthesis LaTeX Template User's Manual*. NOVA University Lisbon. 2021. URL: https://github.com/joaomlourenco/novathesis/raw/main/template.pdf (cit. on p. i).

[23] D. T. Luc, T. Q. Phong, and M. Volle. "Scalarizing functions for generating the weakly efficient solution set in convex multiobjective problems". In: *SIAM J. Optim.* 15.4 (2005), pp. 987–1001 (cit. on p. 36).

[24] A. Mohammadi and A. L. Custódio. "A trust-region approach for computing Pareto fronts in multiobjective optimization". In: *Comput. Optim. Appl.* 87 (2024), pp. 149–179 (cit. on pp. 36, 40, 41).

[25] J. J. Moré. "Recent developments in algorithms and software for trust region methods". In: *Mathematical Programming The State of the Art: Bonn 1982*. Ed. by A. Bachem, B. Korte, and M. Grötschel. Berlin, Heidelberg: Springer Berlin Heidelberg, 1983, pp. 258–287 (cit. on p. 26).

[26] V. Morovati, H. Basirzadeh, and L. Pourkarimi. "Quasi-Newton methods for multi-objective optimization problems". In: *4OR* 16 (2018), pp. 261–294 (cit. on p. 36).

[27] J. Nocedal and S. J. Wright. *Numerical Optimization*. Second Edition, New York, USA: Springer, 2006 (cit. on pp. 6–8, 10, 23–26, 28, 29, 53).

[28] M. J. D. Powell. "Least Frobenius norm updating of quadratic models that satisfy interpolation conditions". In: *Math. Program.* 100 (2004), pp. 183–215 (cit. on p. 22).

[29] S. Qu, M. Goh, and B. Liang. "Trust region methods for solving multiobjective optimisation". In: *Optim. Methods Softw.* 28 (2013), 796–811 (cit. on pp. 36, 37, 43).

[30] V. A. Ramirez and G. N. Sottosanto. "Nonmonotone trust region algorithm for solving the unconstrained multiobjective optimization problems". In: *Comput. Optim. Appl.* 81 (2022), pp. 769–788 (cit. on pp. 36, 37).

[31] J.-H. Ryu and S. Kim. "A derivative-free trust-region method for biobjective optimization". In: *SIAM J. Optim.* 24 (2014), pp. 334–362 (cit. on p. 37).

[32] K. Scheinberg and Ph. L. Toint. "Self-correcting geometry in model-based algorithms for derivative-free unconstrained optimization". In: *SIAM J. Optim.* 20 (2010), pp. 3512–3532 (cit. on pp. 12, 32).

[33] J. Thomann and G. Eichfelder. "A trust region algorithm for heterogeneous multiobjective optimization". In: *SIAM J. Optim.* 29 (2019), pp. 1017–1047 (cit. on pp. 35, 37, 46, 78).

[34] K. D. V. Villacorta, P. R. Oliveira, and A. Soubeyran. "A trust region method for unconstrained multiobjective problems with applications in satisficing processes". In: *J. Optim. Theory Appl.* 160 (2014), pp. 865–889 (cit. on pp. 48, 49, 77, 78).

[35] E. Zitzler, L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca. "Performance assessment of multiobjective optimizers: An analysis and review". In: *IEEE T. Evolut. Comput.* 7 (2003), pp. 117–132 (cit. on p. 38).