EVOLUTIONARY ALGORITHMS FOR OPTIMIZING PESSIMISTIC BILEVEL PROBLEMS AND MIN-MAX PROBLEMS

Margarita Antoniou

Doctoral Dissertation
Jožef Stefan International Postgraduate School
Ljubljana, Slovenia

Supervisor: Prof. Dr. Gregor Papa, IPS and Jožef Stefan Institute, Ljubljana, Slovenia

Evaluation Board:
Asst. Prof. Dr. Tea Tušar, Chair, IPS and Jožef Stefan Institute, Ljubljana, Slovenia
Prof. Dr. Aleš Zamuda, Member, Faculty of Electrical Engineering and Computer Science, University of Maribor, Slovenia
Prof. Dr. Ankur Sinha, Member, Indian Institute of Management Ahmedabad, India

MEDNARODNA PODIPLOMSKA ŠOLA JOŽEFA STEFANA
JOŽEF STEFAN INTERNATIONAL POSTGRADUATE SCHOOL

Margarita Antoniou

EVOLUTIONARY ALGORITHMS FOR OPTIMIZING PESSIMISTIC BILEVEL PROBLEMS AND MIN-MAX PROBLEMS

Doctoral Dissertation

EVOLUCIJSKI ALGORITMI ZA OPTIMIZACIJO PESIMISTIČNIH DVONIVOJSKIH PROBLEMOV IN MINIMAKS PROBLEMOV

Doktorska disertacija

Supervisor: Prof. Dr. Gregor Papa

Ljubljana, Slovenia, February 2025

Acknowledgments

I am grateful for the guidance I received throughout this PhD journey. First and foremost, I would like to sincerely thank my supervisor, Prof. Dr. Gregor Papa, for his thoughtful mentorship and continuous support over these years. I am especially thankful to Prof. Dr. Ankur Sinha for his academic input, insightful discussions, and our valuable collaboration. I would also like to thank the other members of my evaluation committee—Asst. Prof. Dr. Tea Tušar and Prof. Dr. Aleš Zamuda—for their constructive feedback and suggestions, which enhanced the quality of this thesis.

The Marie Curie ITN UTOPIAE project has been instrumental in initiating and funding this research, and I'm grateful to have been part of the network and to have collaborated with the other early-stage researchers. I also want to thank my secondment hosts at ESTECO and the University of Ghent. Additional support was provided by other EU-funded projects, including iRel40 and CONDUCTOR. I am grateful to Prof. Dr. Gregor Papa for giving me the opportunity to work on these projects, and to the Jožef Stefan Institute and the Department of Computer Systems for providing the resources that made this research possible.

I would also like to thank my diploma and master's thesis supervisors in Greece, Prof. Konstantinos Katsifarakis and Prof. Nikolaos Theodossiou, for sparking my interest in optimization and evolutionary algorithms.

To all my friends and loved ones, near and far, thank you for being there and for your support. To my extended family and my grandmother, thank you for your care and encouragement. To my parents, Eleni and Christoforos, and my sister, Penelope, thank you for your constant support and everything you've done for me over the years. Μαμά, μπαμπά, σας ευχαριστώ.

Abstract

This thesis investigates two distinct but related optimization problems: bilevel and min-max problems, in the context of evolutionary algorithms (EAs). Bilevel optimization involves hierarchical decision-making, where decisions at the upper level are subject to constraints defined by the solutions of an optimization problem at the lower level. These problems are particularly challenging due to their nested structure, the need to solve multiple optimization problems simultaneously and hierarchically, and the ill-posedness that arises when the lower level admits multiple optimal solutions. To address this ill-posedness, bilevel problems can be analyzed through two approaches: optimistic and pessimistic. The optimistic approach assumes that the lower-level decision-maker (follower) will select a solution that is most favorable to the upper-level decision-maker (leader). In contrast, the pessimistic approach assumes the follower selects the least favorable solution for the leader, requiring the leader to optimize for the best achievable outcome under worst-case conditions. This thesis introduces a δ-perturbed formulation to tackle this ill-posedness, ensuring a unique approximate solution in the lower level. The method is applied to a differential evolution (DE), a type of EA. EAs, including DE, are particularly suited for solving bilevel optimization problems due to their ability to efficiently explore complex, high-dimensional, and non-convex spaces. Theoretical properties of the δ-perturbed formulation, including error bounds and proofs, are validated through computational studies. Extensive experiments on benchmark test functions demonstrate the approach's ability to find both optimistic and pessimistic solutions. The methodology is then extended to multiobjective bilevel problems, where the upper level involves multiple objectives, and the lower level retains a single objective. In these problems, the ill-posedness of the lower level leads to two distinct Pareto fronts in the upper level. The optimistic Pareto front arises when the follower selects solutions that maximize the leader's benefit, while the pessimistic Pareto front represents the best achievable trade-offs under the follower's worst-case behavior. These distinct Pareto fronts highlight the impact of lower-level ambiguities on upper-level objectives. The method is employed in the m-BLEAQ algorithm, an evolutionary multiobjective bilevel algorithm, and computational experiments on test problems validate its ability to accurately approximate both fronts. Min-max optimization is a special case of bilevel optimization, where the leader seeks solutions against the worst-case responses of the follower. These problems model robust optimization scenarios, where the objective is to find solutions that perform reliably under the most unfavorable conditions imposed by uncertainty. A differential evolution with estimation of distribution algorithm (MMNDE-EDA) is proposed to solve min-max problems. This algorithm combines global search capabilities with probabilistic modeling to efficiently estimate best worst-case solutions. Experiments on standard min-max test functions and an engineering application demonstrate the algorithm's comparable performance to a state-of-the-art method. These contributions advance theoretical understanding and provide novel methods for solving hierarchical optimization problems, validated through computational experiments.

Povzetek

Disertacija obravnava dva različna, vendar povezana optimizacijska problema: dvonivojske in minimaks probleme v kontekstu evolucijskih algoritmov (EA). Dvonivojska optimizacija vključuje hierarhično odločanje, pri katerem so odločitve na zgornjem nivoju podvržene omejitvam, določenim z rešitvami optimizacijskega problema na spodnjem nivoju. Ti problemi so posebej zahtevni zaradi svoje ugnezdene strukture, potrebe po sočasnem in hierarhičnem reševanju več optimizacijskih problemov ter zaradi slabo-pogojenosti, ki nastane, kadar spodnji nivo dopušča več optimalnih rešitev. Za obravnavo te slabo-pogojenosti je mogoče dvonivojske probleme analizirati z dvema pristopoma: optimističnim in pesimističnim. Optimistični pristop predpostavlja, da bo odločevalec na spodnjem nivoju (sledilec) izbral rešitev, ki je najbolj ugodna za odločevalca na zgornjem nivoju (vodjo). Nasprotno pa pesimistični pristop predpostavlja, da sledilec izbere najmanj ugodno rešitev za vodjo, kar od vodje zahteva optimizacijo za najboljši možni izid v najslabših okoliščinah. Disertacija uvaja δ-perturbirano formulacijo za reševanje slabo-pogojenosti, kar zagotavlja približno enolično rešitev na spodnjem nivoju ob ustrezni izbiri parametra δ. Metoda se uporablja v kombinaciji z diferencialno evolucijo (DE), vrsto EA. Evolucijski algoritmi, vključno z DE, so posebej primerni za reševanje dvonivojskih optimizacijskih problemov zaradi svoje sposobnosti učinkovitega raziskovanja kompleksnih, visokodimenzionalnih in nekonveksnih prostorov. Teoretične lastnosti δ-perturbirane formulacije, vključno z mejami napak in dokazi, so potrjene s pomočjo računalniških študij. Obsežni eksperimenti na testnih funkcijah potrjujejo sposobnost pristopa pri iskanju tako optimističnih kot pesimističnih rešitev. Metodologija se razširi tudi na večkriterijske dvonivojske probleme, kjer zgornji nivo vključuje več ciljev, spodnji nivo pa ostaja enokriterijski. V teh problemih slabo-pogojenost spodnjega nivoja vodi do dveh različnih Pareto front na zgornjem nivoju. Optimistična Pareto fronta nastane, kadar sledilec izbere rešitve, ki maksimizirajo koristi za vodjo, medtem ko pesimistična Pareto fronta predstavlja najboljše možne kompromise ob najslabšem vedenju sledilca. Te različne Pareto fronte poudarjajo vpliv nejasnosti spodnjega nivoja na cilje zgornjega nivoja. Metoda se uporablja v algoritmu m-BLEAQ, evolucijskem večkriterijskem dvonivojskem algoritmu, računalniški eksperimenti na testnih problemih pa potrjujejo sposobnost algoritma za natančno približevanje rešitvam obeh front. Minimaks optimizacija je poseben primer dvonivojske optimizacije, kjer vodja išče rešitve ob najslabšem odzivu sledilca. Ti problemi modelirajo scenarije robustne optimizacije, kjer je cilj najti rešitve, ki ostanejo učinkovite tudi v najslabših pogojih, ki jih povzroča negotovost. Za reševanje minimaks problemov je predlagan diferencialni evolucijski algoritem z oceno porazdelitve (MMNDE-EDA). Ta algoritem združuje globalne iskalne sposobnosti s probabilističnim modeliranjem za učinkovito oceno najboljših rešitev ob najslabših odzivih. Poskusi na standardnih testnih funkcijah min-max in inženirski aplikaciji dokazujejo primerljivo učinkovitost algoritma v primerjavi s sodobno metodo. Ti prispevki izboljšujejo teoretično razumevanje in predstavljajo nove metode za reševanje hierarhičnih optimizacijskih problemov, potrjene s pomočjo računalniških eksperimentov.

Contents

List of Figures

List of Tables

List of Algorithms

Abbreviations

AE	... Absolute Error
ASGA	... Aging Sampled Genetic Algorithm
BiGA	... Bilevel Genetic Algorithm
BLEA-Krig	... Bilevel Evolutionary Algorithm with Kriging
BLEAQ	... Bilevel Evolutionary Algorithm based on Quadratic Approximations
BLEAs	... Bilevel Evolutionary Algorithms
BLP	... Bilevel Linear Programming
BOP	... Bilevel Optimization Problem
CODBA-LS	... Co-evolutionary Decomposition Algorithm with Local Search
CoBRA	... Co-evolutionary Bilevel Optimization Algorithm
CVPP	... Commercial Virtual Power Plant
DE	... Differential Evolution
EDA	... Estimation of Distribution Algorithms
EAs	... Evolutionary Algorithms
E-CODBA	... Energy-based Co-evolutionary Decomposition Algorithm
GA	... Genetic Algorithm
GD	... Generational Distance
HV	... Hypervolume
IBEA	... Indicator-based Evolutionary Algorithm
IGD	... Inverted Generational Distance
KKT	... Karush-Kuhn-Tucker
KTLBO	... Kriging-assisted Teaching-Learning-based Optimization
LL	... Lower Level
MBOP	... Multiobjective Bilevel Optimization Problem
MIBLPP	... Mixed-Integer Bilevel Linear Programming Problem
MMDE	... Min-Max Differential Evolution
MMNDE-EDA	... Min-Max Nested Differential Evolution Estimation of Distribution Algorithm
MOEA/D	... Multiobjective Evolutionary Algorithm based on Decomposition
MOEAs	... Multiobjective Evolutionary Algorithms
MOPs	... Multiobjective Optimization Problems
m-BLEAQ	... Multiobjective BLEAQ
NSGA-II	... Non-dominated Sorting Genetic Algorithm II
N-BLEA	... Nested Bilevel Evolutionary Algorithm
PSO	... Particle Swarm Optimization
SABLA	... Surrogate-Assisted Bilevel Algorithm
SAEAs	... Surrogate-assisted Evolutionary Algorithms
SBSO	... Surrogate-based Bilevel Shape Optimization
SPEA2	... Strength Pareto Evolutionary Algorithm 2
TLEA	... Transfer Learning-based Parallel Evolutionary Algorithm
UL	... Upper Level

Symbols

F ... Upper-level objective function
G ... Upper-level constraints
f ... Lower-level objective function
g ... Lower-level constraints
Ψ ... Lower level reaction set
S ... Lower level feasible region (admissible set)
I ... Inducible region
ψ ... Choice function
φ ... Lower level optimal value function
k ... Upper-level pessimistic function
M ... Maximum value of the upper-level objective function
m ... Minimum value of the upper-level objective function
δ ... Perturbation value, a small non-negative perturbation parameter
λ ... Dual variable for the constraint $f(x, y) \leq \varphi(x, y)$
y^o ... Optimistic lower-level solution
y_o^δ ... Perturbed optimistic lower-level solution
y^p ... Pessimistic lower-level solution
y_p^δ ... Perturbed pessimistic lower-level solution
ψ^o ... Optimistic choice function
ψ_o^δ ... Perturbed optimistic choice function
ψ^p ... Pessimistic choice function
ψ_p^δ ... Perturbed pessimistic choice function

Glossary

Bilevel Optimization A class of optimization problems where two levels of decision-making are involved. The upper-level decision maker (the leader) is influenced by the optimal decisions of the lower-level (the follower). The lower-level solution, which is itself the result of an optimization problem, depends on the decisions made at the upper level.

Estimation of Distribution Algorithms (EDAs) A class of evolutionary algorithms that replace traditional crossover and mutation operators with probabilistic models to estimate and sample new candidate solutions, using the distribution of previously successful solutions.

Evolutionary Algorithms (EAs) A class of optimization algorithms inspired by natural selection, where candidate solutions evolve over generations through processes like selection, mutation, and crossover to improve the objective function.

Follower(Lower level decision maker) The decision-maker in a bilevel optimization or Stackelberg game who responds to the upper-level (leader)'s decision, optimizing their own objective based on the leader's choices.

Min-Max Optimization An optimization approach where the objective is to minimize the worst-case scenario (the maximum value of a given objective function), often used in robust optimization to account for uncertainty in the problem.

Multiobjective Bilevel Problems Bilevel optimization problems that involve multiple objectives at one or both the upper and lower level.

Optimistic Bilevel Problem A type of bilevel optimization where the follower (at the lower level) is assumed to make decisions that lead to the best possible outcome for the leader (at the upper level), reflecting an optimistic scenario for the leader.

Optimistic Pareto Set The set of solutions in a multiobjective bilevel problem where the leader assumes the best-case scenario for the follower's responses, resulting in a Pareto set that reflects the optimistic decision-making perspective.

Pareto Set A set of Pareto optimal solutions in a multiobjective optimization problem. A solution is Pareto optimal if no other solution improves one objective without worsening at least one other objective.

Pareto Front The image of the Pareto set in the objective space.

Pessimistic Bilevel Problem A type of bilevel optimization where the lower-level (follower) is assumed to make decisions that minimize the upper-level (leader)'s objective, reflecting a worst-case scenario for the leader.

Pessimistic Pareto Set The set of solutions in a multiobjective bilevel problem where the leader assumes the worst-case scenario for the follower's responses, resulting in a Pareto set that reflects the pessimistic decision-making perspective.

Perturbation Techniques Methods used to introduce controlled changes in optimization problems, typically to explore solution spaces more effectively or to address issues like non-uniqueness in solutions.

Robust Optimization An type of optimization that seeks solutions effective under a range of possible scenarios, ensuring performance despite variability in data or model parameters.

Stackelberg Game A strategic game where the leader makes the first move, and the followers choose their strategies based on the leader's decision.

Leader (Upper level decision maker) The decision-maker in a bilevel optimization or Stackelberg game who makes the first move, setting the conditions for the follower to respond.

Chapter 1

Introduction

"The science of operations, as derived from mathematics more especially, is a science of itself, and has its own abstract truth and value"
Ada Lovelace, Notes on the Analytical Engine, 1843

This thesis proposes new methods for addressing ill-posed bilevel problems, including both pessimistic and optimistic bilevel problems, and min-max problems, all within the framework of evolutionary algorithms. In this chapter, we outline the background and motivation, identify the research gap, provide an overview of the methodology, present the hypotheses, and outline the objectives of the thesis. Additionally, we present the study's scope and thesis structure.

1.1 Background and Motivation

Every day, we make choices to achieve the best possible result—and that is what optimization is all about. Derived from the Latin word optimum, meaning 'the best' or 'most favorable', optimization is about finding the best solutions. Whether it is planning the quickest route to work, prioritizing tasks based on urgency, or selecting the shortest line at the grocery store by assessing both the cashier's speed and the number of people in line, we are constantly making small optimizations.

When we move from simple, everyday decisions to more complex problems, a clear mathematical model becomes necessary. Before applying optimization techniques, we need to define the objective function (what needs to be optimized), the decision variables (the choices we can make), and the problem's constraints (its limits and requirements). This step is crucial because it provides a structured representation of the problem, enabling effective optimization. Without a solid model, we risk making decisions based on incomplete information, which could lead to less effective or even incorrect solutions.

Many real-world applications are hierarchical by nature, making it necessary to model these problems in such a way. This leads to a specific class of optimization problems known as *bilevel optimization*. In bilevel optimization, an upper-level problem depends on the solution of a lower-level problem, creating a nested structure. These problems are particularly challenging, with even the simplest forms classified as NP-hard [1]. From the perspective of game theory, bilevel optimization can be seen as a Stackelberg game, where the upper-level decision-maker (the leader) chooses their strategy first, anticipating that the lower-level decision-maker (the follower) will respond by optimizing their own objectives given the leader's choice. The leader takes this anticipated response into account in their planning, using a first-mover advantage.

Basically, modeling an optimization problem means recognizing that decision-making

in many circumstances is not usually straightforward, due to uncertainty. This uncertainty can come from limited data, dynamic conditions, or unpredictable behaviors of the parties involved. Even with a carefully articulated model, one cannot always exhibit a system's full complexity for all situations. To address this, robust optimization aims to find solutions that perform well across a range of possible scenarios, particularly under worst-case conditions [2]. This approach is often framed as a *min-max problem*, a type of bilevel optimization [3], where the objective is to ensure good performance even in the least favorable situations.

Classical optimization algorithms are well-established methods designed to solve specific types of problems, often with convergence guarantees under certain conditions. For example, when the objective and constraints are all linear, then the problem can be solved using the now-called linear-programming techniques [4]. Similarly, when the objective function is quadratic, quadratic programming methods can be applied directly to solve the problem. Gradient-based algorithms, such as gradient descent and the Newton method, use derivative information to iteratively converge to optimal points, performing well for smooth and differentiable functions [5]. Linear constrained problems are well-handled by methods like the simplex algorithm and Lagrange multipliers, which effectively address equality and inequality constraints [4]. Classical approaches are particularly useful for problems with properties like convexity, which guarantees that any local minimum is also a global minimum. However, these methods are less effective for complex, non-convex, or non-differentiable objective functions, where alternative methods are required.

In this context, Evolutionary Algorithms (EAs) provide an alternative approach. Inspired by natural selection, EAs use mechanisms such as selection, mutation, and crossover to improve candidate solutions over time. These population-based heuristics are especially suitable for complex optimization problems with nonlinearity, nonconvexity, and non-standard properties. EAs are also applicable to black-box optimization, where the objective function's mathematical properties are unknown or difficult to define [6]. A key advantage of EAs is their ability to optimize the objective function directly without requiring derivative information. Additionally, the flexible representation of solutions allows EAs to adapt to a wide range of problem types, enabling effective optimization even when the problem's structure is not explicitly defined. Their population-based nature also helps maintain solution diversity, reducing the risk of getting stuck in local optima. Although EAs do not guarantee optimality due to their stochastic nature, they can adequately approximate solutions in many cases where conventional optimization techniques cannot even be applied [7].

1.2 Research Gap and Problem Statement

As mentioned earlier, EAs are well-suited for complex optimization problems due to their ability to handle nonlinearity, nonconvexity, and discontinuities, all of which are common in bilevel optimization problems. Bilevel Evolutionary Algorithms (BLEAs) are evolutionary algorithms specifically designed to solve bilevel optimization problems, in which one level interacts with the other. EAs also do not require derivatives; thus, they are especially suitable for bilevel problems whose gradients are difficult or impossible to compute because of the hierarchical structure. Besides, bilevel optimization problems often involve continuous and discrete variables, which EAs handle effectively through their flexible representation of solutions and their ability to apply the same operators, such as crossover and mutation, to both variable types. They also excel at navigating complex, high-dimensional solution spaces, incorporating constraints to ensure that solutions are feasible and aligned with the problem's structure. Examples of BLEAs include BlDE [8], NBLEA [9], a memetic

approach [10], BLEAQ [11], and BL-CMA-ES [12].

Most existing BLEAs assume either a single lower-level optimal solution or adopt an *optimistic approach* to solve bilevel optimization problems. When the lower-level problem is multimodal—having several global optima—the problem becomes ill-posed, resulting in a lack of a clear optimal response from the lower level, complicating the interactions. The optimistic approach makes finding a solution easier because it assumes, in its solution process, that the lower-level decision maker will also act optimally to favor the upper-level. In contrast, a *pessimistic approach* optimizes for the worst-case scenario, accounting for potential opposition from the lower level. Although more tractable, the optimistic approach is based on the highly unrealistic assumption of cooperation among decision-makers, especially without some form of reward for the lower level. While this pessimistic approach is theoretically more conservative in trying to avoid risks and giving more robust solutions under problem conditions, it deserves to hold a place of special worth in real-world applications. Due to its much higher computational complexity, the BLEAs have driven less attention to this approach despite its advantages. More efficient methods are necessary to address pessimistic bilevel optimization, especially in applications where cooperation cannot be guaranteed.

Meanwhile, EAs are also utilized for solving min-max problems, which represent special cases of a bilevel optimization problem. In such problems, the upper level (designer) optimizes against the worst-case response of the lower level (environment or nature), and both levels commonly share the same objective function. The most widely adapted methods in solving the min-max problems with EAs is the co-evolutionary strategy. This is achieved with the simultaneous evolution of the populations in both the design and scenario spaces. For example, a co-evolutionary genetic algorithm was introduced in [13], and particle swarm optimization was used in [14]. While effective for symmetric min-max problems, these methods struggle with asymmetric cases, and nested methods, which solve the lower-level problem repeatedly during optimization, increase computational costs, highlighting the need for more efficient approaches.

1.3 Overview of Methodology

This thesis adopts a structured methodology that combines a review of related work, development of novel methods, theoretical analysis, and empirical validation to address the challenges of bilevel and min-max optimization using evolutionary algorithms. We begin by reviewing key methods and approaches in bilevel and min-max optimization, focusing on evolutionary algorithms. This review helps identify research gaps, such as the need for methods that ensure solution uniqueness in bilevel optimization and the high computational cost of solving min-max problems.

Based on these findings, we propose a mathematical method called δ-perturbation for bilevel optimization, which ensures a unique approximate solution at the lower level. This method is integrated with evolutionary algorithms to develop a perturbation-based bilevel Differential Evolution (DE) algorithm. These methods allow the identification of both optimistic and pessimistic solutions, enabling robust decision-making under ambiguity. The δ-perturbation method is extended to multiobjective bilevel optimization and applied within the m-BLEAQ evolutionary algorithm, enabling the identification of optimistic and pessimistic Pareto fronts at the upper level. These fronts represent the boundaries of feasible solutions, considering cooperative and worst-case lower-level responses. For min-max problems, we introduce a hybrid algorithm combining nested DE with Estimation of Distribution Algorithms (EDAs), termed MMNDE-EDA. This approach uses probabilistic modeling to reduce computational costs while maintaining solution quality.

Mathematical proofs are provided to support the δ-perturbation method, including error bounds for optimistic and pessimistic solutions. The proposed algorithms are validated through computational experiments and benchmarking against state-of-the-art algorithms when possible, demonstrating their performance.

1.4 Hypotheses

The hypotheses of this thesis are:

H1: EAs can be enhanced with perturbation methods to approximate both optimistic and pessimistic solutions for ill-posed single-objective bilevel optimization problems.

H2: The integration of perturbation methods within EAs will enable them to effectively identify both optimistic and pessimistic Pareto Fronts for multiobjective bilevel optimization problems.

H3: Implementing probabilistic Estimation of Distribution within nested EAs can reduce the computational costs associated with solving min-max problems without sacrificing solution quality.

1.5 Objectives

The main objectives of this research are as follows:

O1: Establish a foundational understanding of bilevel and min-max optimization methods using evolutionary algorithms.

- O1.1: Review key methods and approaches in bilevel optimization.
- O1.2: Examine existing evolutionary algorithm techniques in min-max optimization.
- O1.3: Identify gaps in the current research related to evolutionary algorithms in these contexts.

O2: Develop perturbation methods and their theoretical framework for bilevel optimization problems.

- O2.1: Investigate perturbation techniques within evolutionary algorithms for single-objective bilevel problems, including proofs and theoretical error bounds for optimistic and pessimistic solutions.
- O2.2: Extend perturbation techniques to multiobjective bilevel optimization problems for obtaining optimistic and pessimistic Pareto fronts.

O3: Design and implement evolutionary algorithms for bilevel and min-max optimization.

- O3.1: Extend evolutionary algorithms with perturbation techniques to solve single-objective bilevel problems effectively.
- O3.2: Integrate perturbation techniques into evolutionary algorithms to support multiobjective bilevel optimization, focusing on optimistic and pessimistic Pareto fronts.
- O3.3: Implement a probabilistic Estimation of Distribution within nested evolutionary algorithms to reduce computational costs in min-max optimization while maintaining solution quality.

O4: Evaluate and validate the proposed algorithms and perturbation methods.

O4.1: Validate the theoretical error bounds for perturbation techniques in single-objective bilevel optimization through computational studies.

O4.2: Conduct benchmarking analyses of the proposed algorithms to assess their effectiveness across different bilevel and min-max optimization problems.

1.6 Scope of the Study

This study focuses on continuous bilevel optimization problems with non-unique lower-level solutions, as well as multiobjective bilevel problems characterized by a multiobjective upper level and a single-objective lower level with non-unique solutions. The main algorithms used are nested DE and m-BLEAQ; however, the proposed techniques are applicable to other evolutionary algorithms as well. Additionally, it addresses unconstrained continuous min-max problems, for which the algorithms employed are nested DE and EDAs. The probabilistic method is EA-independent and can be applied to other nested evolutionary algorithms as well.

We treat all problems as black-box optimization problems, where only the input-output evaluations are available, without access to the internal structure of the functions. Optimization problems can be categorized by their solution methods, the number of objectives they have, the types of their variables, such as integer or continuous, the number of levels of decision-making, the presence of constraints or not, and the nature of their objective function, such as if it is convex, nonconvex, or black-box. These categories are shown in Figure 1.1. The grayed parts are the categories relevant to our thesis.

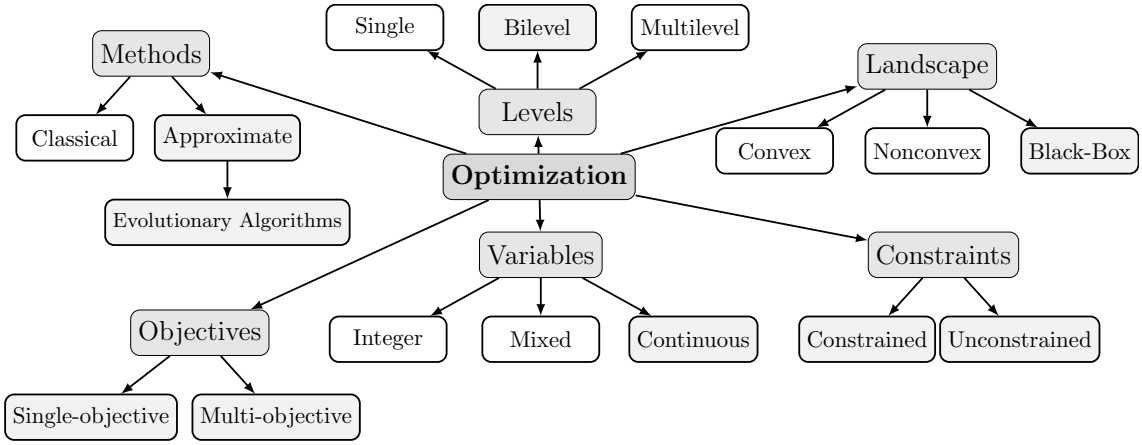


Figure 1.1: Categories of optimization problems based on problem characteristics and methods.

1.7 Structure of the Thesis

This thesis consists of seven chapters, including the current, introductory one. Figure 1.2 presents the chapter outline diagram.

Chapter 2 addresses bilevel optimization, covering definitions, mathematical formulations, and real-world applications. It outlines the two approaches for handling ill-posedness, namely the optimistic and pessimistic approaches, along with their mathematical defini-

tions. The chapter also reviews classical methods for both optimistic and pessimistic problems and discusses the different categories of bilevel evolutionary algorithms.

In Chapter 3, we propose the δ-perturbed formulation of bilevel optimization problems, useful for efficiently managing multiple lower-level optimal solutions. This approach ensures a single approximate optimal solution for any upper-level decision by employing an appropriate perturbation strategy. The chapter includes proofs and theoretical error bounds. We then apply the perturbation to a nested DE, and the computational studies validate the error bounds and showcase the efficacy of the approach on various test functions.

In Chapter 4, we extend the δ-perturbation method to approximate the optimistic and pessimistic Pareto fronts in a certain type of multiobjective bilevel optimization problems. These problems have multiple objectives at the upper level and a single objective at the lower level. To assess the method, we incorporate the pertubation into the existing m-BLEAQ algorithm. Computational experiments on test functions test its ability to determine both optimistic and pessimistic fronts.

Chapter 5 transitions to min-max optimization, defining its fundamental concepts and mathematical formulation. We show how it is used in robust optimization and review the existing evolutionary algorithms that are developed for min-max problems.

Chapter 6 introduces a differential evolution with an estimation of distribution algorithm for min-max optimization (MMNDE-EDA). This approach utilizes a nested structure that combines differential evolution with a probabilistic model to estimate the best worst-case solutions. The probabilistic model represents the distribution of the best solutions found during the optimization process, incorporating a key characteristic of EDAs. By sampling from this distribution, the algorithm uses information from previous iterations to generate new candidate solutions in the lower level, saving function evaluations. We then assess the influence of the model on the algorithm and compare the performance of the proposed algorithm against a state-of-the-art algorithm on benchmark problems.

Chapter 7 provides a summary of the thesis, along with the scientific contributions and the validation of the hypotheses. It also suggests potential directions for future work.

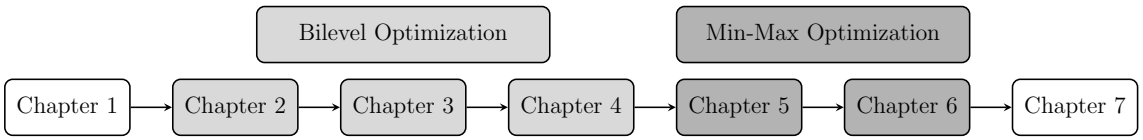


Figure 1.2: Chapter outline diagram.

Chapter 2

Bilevel Optimization: Definitions and Existing Methods

In this chapter, we introduce bilevel optimization, starting with a historical overview and its connections to game theory. We present key applications to demonstrate their practical relevance. A formal mathematical definition, along with an example using linear functions, clarifies core principles. The chapter also distinguishes between bilevel and biobjective problems and examines optimistic and pessimistic formulations, each supported by a mathematical example. Additionally, we review methods for solving these problems, covering both classical and evolutionary approaches.

Connection to Thesis Objectives

This chapter addresses O1.1, O1.3.

2.1 A Little Bit of History

Although we may recognize optimization as a formal mathematical field today, the roots of optimization problems can be traced back in time. Ancient civilizations were already dealing with problems in resource allocation, land division, and trade logistics—decisions that required finding the best use of limited resources. Though these early decision makers were not working within a formalized field, they were solving optimization problems in some way. For example, Euclid's development of geometry in ancient Greece provided fundamental tools for spatial reasoning that would later support more structured mathematical methods.

A more systematic approach to these types of problems began to emerge in the 17th century with the development of calculus by Newton and Leibniz [16], [17]. Calculus provided the mathematical tools necessary to rigorously study continuous change, and Pierre de Fermat [18] soon applied these methods to finding maxima and minima, marking an early step toward the optimization field as we know it today.

The 18th and 19th centuries brought further mathematical advances foundational to optimization. Joseph-Louis Lagrange's introduction of the method of Lagrange multipliers made it possible to find extrema of functions under constraints—a technique that bridged basic calculus with more structured mathematical approaches [19]. This work laid princi-

Parts of this chapter have been published in:
M. Antoniou and P. Korošec, "Multilevel optimisation," in *Optimization Under Uncertainty with Applications to Aerospace Engineering*, Springer, 2021, pp. 307–331

ples that later became essential for applications in economics and engineering. Similarly, Karl Friedrich Gauss made significant contributions through his work on least squares, which laid mathematical groundwork that would influence fields like convex analysis [20]. Linear programming[1] saw its first major breakthrough during World War II, with George Dantzig's development of the Simplex Method [21]. This method provided an efficient, systematic way to solve linear optimization problems, giving rise to operations research as a formal field and leading to widespread applications in industrial planning.

As optimization challenges grew more complex, new branches developed to handle nonlinear and integer problems. In this period, the Karush-Kuhn-Tucker (KKT) conditions were developed, providing necessary conditions for optimality in constrained nonlinear problems [22]. Around the same time, Richard Bellman introduced dynamic programming, a framework that enabled complex problems to be broken into simpler, manageable subproblems, greatly expanding the potential applications of optimization [23]. These advances helped broaden optimization's reach across diverse fields, from economics to engineering and logistics.

During the late 20th century evolutionary algorithms (EAs) were first developed, which took inspiration from biological evolution. John Holland's work on genetic algorithms in 1975 was particularly influential [6]. His approach—modeling optimization as a process of natural selection—provided a novel method for handling high-dimensional, nonlinear problems that traditional techniques struggled to solve. Around this time, convex optimization progressed significantly, with notable advancements in designing efficient algorithms capable of handling large-scale problems [24].

As the 21st century progresses, the field of optimization has further diversified to address increasingly complex and uncertain real-world problems. Stochastic and robust optimization have gained attention, especially in areas like finance and supply chain management, where variability and unpredictability are inherent [25]. Additionally, the integration of optimization with machine learning has led to new methods for training models and making predictions, underscoring the role of optimization in modern data science. Advances in high-performance computing have made it possible to tackle large-scale optimization problems through distributed and parallel processing, accelerating solution times.

Among the branches of optimization, bilevel optimization has emerged as a uniquely challenging area. Unlike traditional single-level optimization, bilevel problems are defined by a hierarchical structure in which two interconnected levels of decision-making occur. The upper level, often called the "leader," seeks to optimize an objective while anticipating the optimal response of the lower level, or "follower."[2] This interplay between levels requires accounting for the follower's reaction, introducing a dependency that greatly complicates the solution process.

The origins of bilevel optimization's hierarchical structure trace back to the leader-follower dynamics introduced by Stackelberg in 1934 [26]. Stackelberg's work in game theory provided an early framework for modeling these interactions, particularly within economic and competitive contexts. However, it was only in 1973 that bilevel optimization was formally defined as a mathematical problem [27]. The term "Bilevel Programming" was coined in 1977 [28], and since then, the field has evolved rapidly.

By the early 1990s, researchers had shown that bilevel optimization problems are NP-hard [1], with strong NP-hardness established in 1994 [29]. This complexity implies that

[1]The term "programming" in optimization was first introduced in operations research during the 1940s. Here, "programming" refers to the structured approach to formulating and solving optimization problems under constraints, distinct from programming in computer science, which typically involves writing code to execute tasks on a computer.

[2]In this thesis, *upper level* (for upper-level decision maker) and *leader* are used interchangeably, as are *lower level* (for lower-level decision maker) and *follower*.

even approximate solutions can be computationally challenging to obtain. Around this same period, genetic algorithms were applied to bilevel optimization, signaling the potential for evolutionary approaches in tackling the hierarchical nature of these problems [30]. Figure 2.1 shows the timeline of these early key developments in bilevel optimization.

The difficulty of solving bilevel problems lies in the need to solve two interconnected optimization problems simultaneously, with the upper-level decision-maker needing to anticipate the optimal reaction of the lower-level. This dependency renders standard optimization methods ineffective, leading to the development of specialized algorithms. In recent decades, bilevel optimization has benefited from advancements in computational power, along with heuristic and metaheuristic approaches, which have provided new ways to explore the solution space. Methods such as particle swarm optimization offer an alternative to traditional approaches, enabling researchers to tackle increasingly complex bilevel problems [31], [32]. Today, bilevel optimization techniques are applied across diverse fields, from economics and engineering design to complex network and transportation planning, where decision-making is inherently hierarchical.

The chapter is structured as follows: Section 2.2 explores connections to game theory, highlighting the relevance of leader-follower dynamics. In Section 2.3, motivating examples demonstrate practical applications of bilevel optimization across various fields. Section 2.4 addresses the mathematical foundations and complexity, covering formal definitions, linear examples, distinctions between bilevel and biobjective problems, optimistic and pessimistic formulations, computational complexity, and connections to other domains. Section 2.5 provides an overview of solution methods, discussing both classical approaches and evolutionary algorithms for bilevel optimization. Finally, Section 2.6 summarizes the chapter's key points.

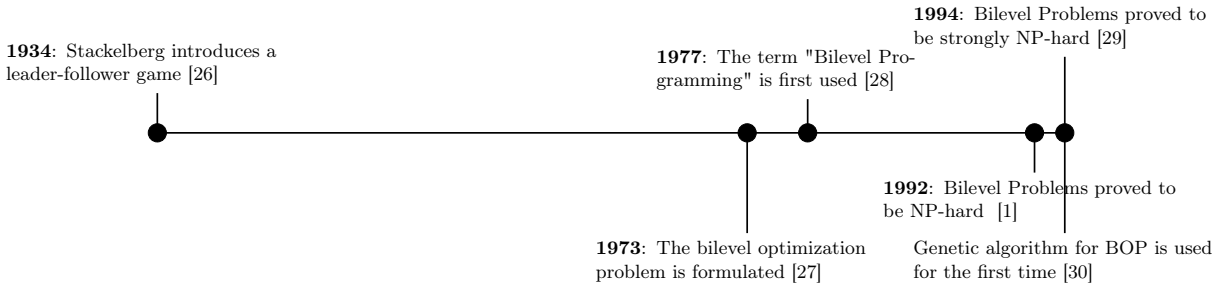


Figure 2.1: Timeline of early developments in bilevel optimization.

2.2 Connections to Game Theory

Game theory, a subfield of applied mathematics, studies scenarios where decision-makers, referred to as players, make interdependent decisions. In such settings, each player's strategy depends on the possible actions of others, as the outcome for one participant is shaped by the combined decisions of all [33]. Game theory aims to identify optimal strategies for participants, whether in cooperative or competitive environments, while analyzing potential outcomes.

The Stackelberg game is a key class within this framework, introduced by the German economist Heinrich von Stackelberg in his 1934 work "Market Structure and Equilibrium" [34]. In a Stackelberg game, one player, the leader, makes the first move, and the follower(s) respond according to their own objectives. This sequential structure leads to a Stackelberg equilibrium, where both the leader's and followers' strategies are optimized

based on the leader's decision. Stackelberg games can be classified as either static—where the interaction occurs at a single point in time—or dynamic, where decisions evolve over multiple stages.

Bilevel optimization problems can be viewed as static, non-cooperative Stackelberg games, as shown in Figure 2.2. In these problems, the leader, an upper-level decision-maker, makes a decision x, which influences the strategies of the lower-level decision-maker, the follower. The follower optimizes their strategy $y^*(x)$ in response to the leader's choice, and this strategy is fed back into the leader's objective function $F(x, y^*(x))$. The solution $(x^*, y^*(x^*))$ represents a Stackelberg equilibrium, where the leader chooses the optimal strategy x^*, anticipating the follower's best response $y^*(x^*)$. In this hierarchical setup, the leader's strategy accounts for the follower's optimal response, and the follower, having no ability to improve their outcome given the leader's decision, plays a reactionary role.

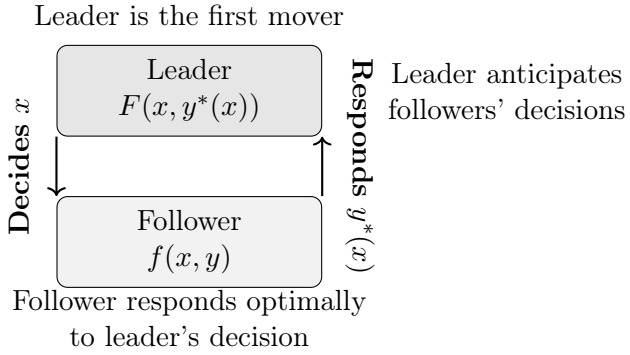


Figure 2.2: Stackelberg Game Model. The leader makes the first decision, anticipating the follower's optimal response. The follower reacts accordingly, reflecting the hierarchical structure of bilevel optimization.

While the standard bilevel optimization problem involves a single leader and a single follower, the model can be extended to handle multiple followers [35]. In this case, each follower makes their own decision based not only on the leader's choice but also considering the decisions of other followers. This results in a game-theoretic context where each follower's optimal strategy depends on the leader's choice and the strategies chosen by the other followers. In such cases, the concept of Nash equilibrium becomes relevant. A Nash equilibrium occurs when each follower's strategy is the best response to the strategies of others. For a Stackelberg game with multiple followers, given the leader's decision x, the followers' strategies form a Nash equilibrium if for each follower i, their optimal decision satisfies $y_i^*(x) = \underset{y_i}{\operatorname{argmin}} f_i(x, y_i, y_{-i}^*(x)), \forall i$. Here, $y_i^*(x)$ represents the optimal decision of follower i, while $y_{-i}^*(x)$ denotes the strategies of all other followers except follower i. A Nash equilibrium among followers occurs when no follower can improve their outcome by changing their strategy unilaterally, given the strategies of other followers. The leader, however, remains unaffected by the competition among followers and optimizes their decision based on the followers' Nash equilibrium response. The structure of the game, including the leader's decision and the followers' responses, is illustrated in Figure 2.3.

This framework can be further extended to cases with multiple leaders, multiple followers, or even more complex hierarchical structures, depending on the specific problem being modeled. Such extensions allow for more realistic representations of real-world decision-making, where multiple agents (leaders or followers) may interact in different ways [36] [37].

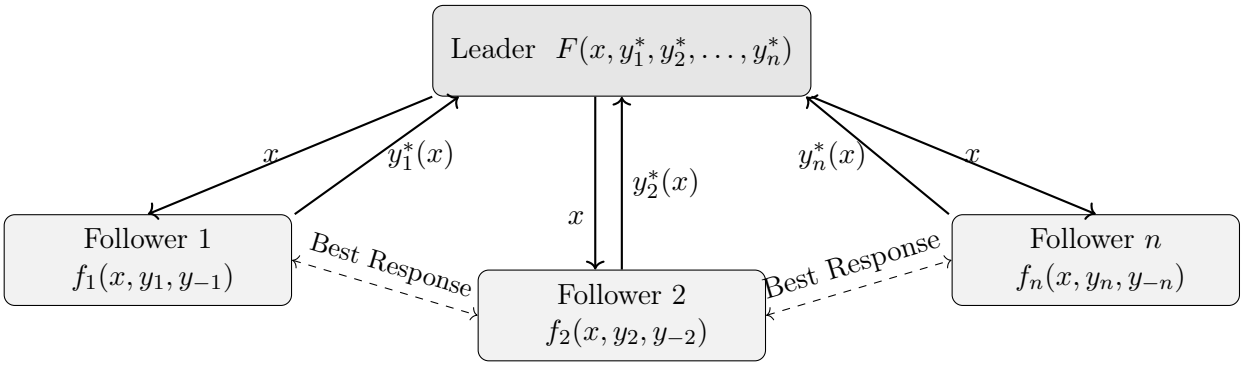


Figure 2.3: Stackelberg game with Nash equilibrium among followers. The leader chooses x, and each follower optimizes their objective $f_i(x, y_i, y_{-i})$ while treating other followers' strategies as fixed. A Nash equilibrium is reached when no follower can improve their outcome unilaterally. The equilibrium strategies $y^*(x)$ depend on the leader's decision and feed back into the leader's objective $F(x, y_1^*, y_2^*, \ldots, y_n^*)$.

2.3 Applications of Bilevel Optimization: Motivating Examples

In this section, we will present a few motivating applications that have benefited from the bilevel formulation. This list is, of course, not exhaustive but a nice starting point to understand the importance of the field. In Figure 2.4, we show examples of applications and their upper and lower levels (UL, LL). More specifically, bilevel problems can be found in:

1. **Transportation Systems**: In transportation bi-level problems, the government or network operator typically acts as the leader, making decisions on network design and price setting. Meanwhile, network users function as followers, choosing specific routes within the framework set by the operator. Such an approach is already well studied, and one can find a review about bilevel programming in traffic planning back in 1995 [38]. The study in [39] addresses a bi-level problem in transportation networks. At the upper level, the network manager aims to enhance global efficiency by mitigating traffic bottlenecks, while at the lower level, passengers minimize their travel costs by selecting the shortest path. The study in [40] formulates a bilevel vehicle routing problem with selective backhauls. At the upper level, the shipper determines minimum-cost delivery routes and offers incentives to carriers for integrated transportation. At the lower level, carriers decide which incentives to accept and optimize backhaul routes. The overall goal is to maximize transportation efficiency through coordinated inbound and outbound routes. Additional review papers and relevant publications can be found in [41],[42],[43] and [44].

2. **Toll setting**: Closely related to transportation networks and price setting, the toll setting problem is a classic example of bilevel modeling, allowing explicit consideration of user behavior within the framework. A prominent study in this area is [45], which examines toll optimization on a multicommodity transportation network with arc capacity constraints. In this context, a highway authority, acting as the "leader," sets tolls on selected road segments. Meanwhile, network users, or "followers," select their routes, choosing the shortest (least costly) paths to connect their origins and destinations. The leader's aim is to maximize toll revenue, leading to the formulation

of a bilevel program. Reviews and comparisons of methods for the bilevel toll setting problem are available in [46], and an overview of its dynamic variant can be found in [47].

3. **Engineering Design**: Bilevel problems naturally form in numerous engineering design applications. In structural optimization, the truss topology optimization (shape) can be the upper level and the size optimization the lower level [48], [49]. A study in aerospace engineering [50], solved the stacking sequence of laminated composite wing structures as a bilevel problem. At the upper level, global optimization defines lamination parameters or assumes homogeneous material properties to satisfy blending and manufacturing constraints. At the lower level, techniques such as ply shuffling or layer permutations (using HyperShuffle or a genetic algorithm) alter the stacking sequences based on these parameters to achieve optimal performance, as demonstrated by tests on an 18-panel wing box. More about optimal engineering design as bilevel problems can be found in [51], [52].

4. **Interdiction Problems**: The interdiction problem is a central focus in discrete bilevel optimization, often involving a leader who acts as a defender and seeks to prevent a follower from accessing specific resources [53]. Many interdiction problems are framed on graphs, where the follower tries to determine an optimal path, like the shortest route from an origin to a destination, while the leader can interdict, or disable, certain arcs in the graph to disrupt this path. The leader's interdiction capability is constrained by a limited budget, allowing only a select number of arcs to be interdicted. This approach is applied in various fields, including cyber network defense, where a risk-averse defender at the upper level optimally deploys security countermeasures to minimize the expected maximum loss across attack scenarios and mitigate the risk of significant losses from the most damaging attacks. In this case, lower-level attackers adjust their strategies to maximize their impact within their budget constraints, dynamically responding to the defender's actions [54]. This structure effectively captures the interaction between defensive and offensive strategies, making it a valuable model for scenarios where resources must be protected under budget constraints.

5. **Enviromental Policy**: Environmental policy includes rules and efforts aimed at conserving natural resources, minimizing pollution, and promoting sustainable development. It combines legislation, incentives, and public participation to address concerns such as climate change and biodiversity loss. Bilevel optimization models frequently frame these policies, with a central authority as the upper-level leader setting environmental objectives and followers (such as producers or power system operators) adjusting their actions to meet these goals while pursuing their own objectives [55]. For example, in a power system, the upper level could be an environmental agency that establishes climate policies, while the lower level is the independent system operator who runs the electricity market according to these policies [56]. Another application uses a bilevel Stackelberg model for green procurement, with manufacturers at the top level lowering costs under carbon tax rules by selecting suppliers, and suppliers at the lower level optimizing purchasing volumes. This strategy demonstrates that varied sourcing boosts producers' flexibility, while reliable, substantial orders help suppliers cut emissions, supporting a greener supply chain [57].

6. **Energy Systems**: Energy systems are the infrastructure, technology, and processes involved in the production, distribution, and consumption of energy. These systems

are critical for addressing societal demands, promoting economic activity, and ensuring sustainability. Bilevel optimization can be applied to energy systems in a variety of ways, including energy market design [58], resource allocation [59], demand-size management [60], policy regulation [61], and hydrothermal scheduling to minimize fuel costs and emissions [62]. For example, in [58], the authors develop a bilevel optimization model for a commercial virtual power plant (CVPP) active in the day-ahead electricity market. At the upper level, the CVPP focuses on maximizing its day-ahead profit while minimizing anticipated costs from real-time imbalances in production and consumption. The lower level represents the independent system operator's task of clearing the market by balancing supply and demand, all while factoring in uncertainties like variations in distributed energy resource production, consumer load, and real-time market prices. The CVPP incorporates demand response strategies to effectively adapt to market fluctuations and manage its risk exposure.

7. **Water Management**: Water management is the process of planning and organizing how we use and protect our water resources so that they can be used in the future as well as today. When it comes to water management issues, there are naturally different levels of decision-makers. These include the government, utilities, businesses, communities, and non-governmental organizations. Each of these groups affects and interacts with the others within a set of rules and guidelines. In this kind of research, a bilevel programming model is used to figure out how to divide up regional water resources based on who owns the water rights in a river area [63]. At the upper level, the model aims to maximize the best overall value for society. At the lower level, it does the same for each subarea's economy. The bilevel optimization seeks to strike a balance between decision-makers to ensure equitable allocation of water resources while also taking into account fuzzy random variables to manage uncertainties. In [64], the authors introduce a bilevel optimization model designed to allocate water resources among various stakeholders over a set time period. At the upper level, a central authority is responsible for distributing water to different demand points, keeping environmental and sustainability goals in mind while balancing supply and demand. Meanwhile, at the lower level, managers of these demand points make water allocation choices based on economic considerations. To help guide these managers, the central authority can use tools like fees to encourage decisions that align with broader goals, even for actions beyond its direct control. Additional studies exploring similar bilevel approaches in water resource management are available in [65], [66], and [67].

8. **Emergency Planning and Disaster Response**: Efficient supply chains are vital for delivering food, water, medicines, clothes, and other shelter materials to the community effectively in the event of natural disasters. Bilevel optimization has been applied to model strategic and operational levels of decision-making in various contexts, such as relief and humanitarian logistics. For instance, [68] develops a bilevel programming model to optimize the deliverability of relief supplies given damaged infrastructure, very high demand over supply, and limited transport capacity. In this model, an upper level minimizes distribution time while, at a lower level, allocation fairness across multiple supply and demand sites is maximized. In fact, it was used on the Wenchuan earthquake in 2008 by showing its practical effects in disasters. Besides, [69] suggests a scheduling model for coordinating emergency workers on rescue missions as a bilevel game. The upper level is dedicated to the coordination of disaster areas' emergency responses, while the lower level takes into account the po-

tential limited rationality of the behavior of victims so that fairness and effectiveness in scheduling can be ensured for minimal loss while enhancing satisfaction. Other works that are related to this topic are [70], [71] and [72], which provide more insight into the applications of bilevel optimization in humanitarian logistics.

9. **Principal–Agent**: From the management control perspective, this is a classic problem of how to design compensation systems that align the principal's objectives–usually profit maximization–with the agent's incentives, typically focused on minimizing effort and risk. Bilevel optimization provides a strategic framework for modeling such interactions. In this setting, the principal at the upper level optimizes the compensation contract to create desired behaviors, while the agent at the lower level chooses their effort level in response to the principal's incentives. Such a hierarchical decision process resolves conflicts and creates equilibria because it aligns the organization's goals with the actions taken. More relevant studies can be found in [73], [74], and [75].

10. **Machine Learning**: In recent years, bilevel optimization has attracted considerable attention for solving different machine learning problems, among which the most popular is hyperparameter tuning. The work in [76] presents a bilevel optimization-based hyperparameter tuning method that overcomes naive techniques like random search and grid search for hyperparameters. This algorithm optimizes the hyperparameters at the upper level to minimize the loss of the validation set, while at the lower level, it minimizes the loss of the training set by approximating the optimal values for some hyperparameter settings. Most adversarial learning methods also consider unsupervised learning as a bilevel game between two competitors, including one generator synthesizing samples from a distribution and one discriminator classifying samples as real or fake [77], [78]. Parameter control in EAs has also benefited from bilevel approaches, as explored in [79], which applied a bilevel optimization framework to tune parameters for two specific evolutionary algorithms: Differential Evolution (DE) and the Nelder-Mead algorithm. By formulating the parameter tuning process as a bilevel problem—where the upper level optimizes the algorithm's parameters and the lower level solves the actual optimization task—the study demonstrated that this approach leads to more efficient and effective parameter settings, enhancing the performance of these algorithms on various optimization problems. Additionally, parameter control has been addressed in [80], [81], where adaptive mechanisms dynamically adjust parameters during the optimization process. These works focus on integrating surrogate models and feedback-driven strategies within evolutionary frameworks, leading to improved convergence rates and computational efficiency in bilevel optimization. Readers may refer to [82], [83], and [84] for further bilevel optimization applications in machine learning.

2.4 Mathematical Foundations and Complexity

2.4.1 Mathematical Definition of Bilevel Optimization Problems

To understand bilevel optimization problems (BOPs), let us first define a standard optimization problem:

Definition 2.1 (Standard Optimization Problem). A standard optimization problem

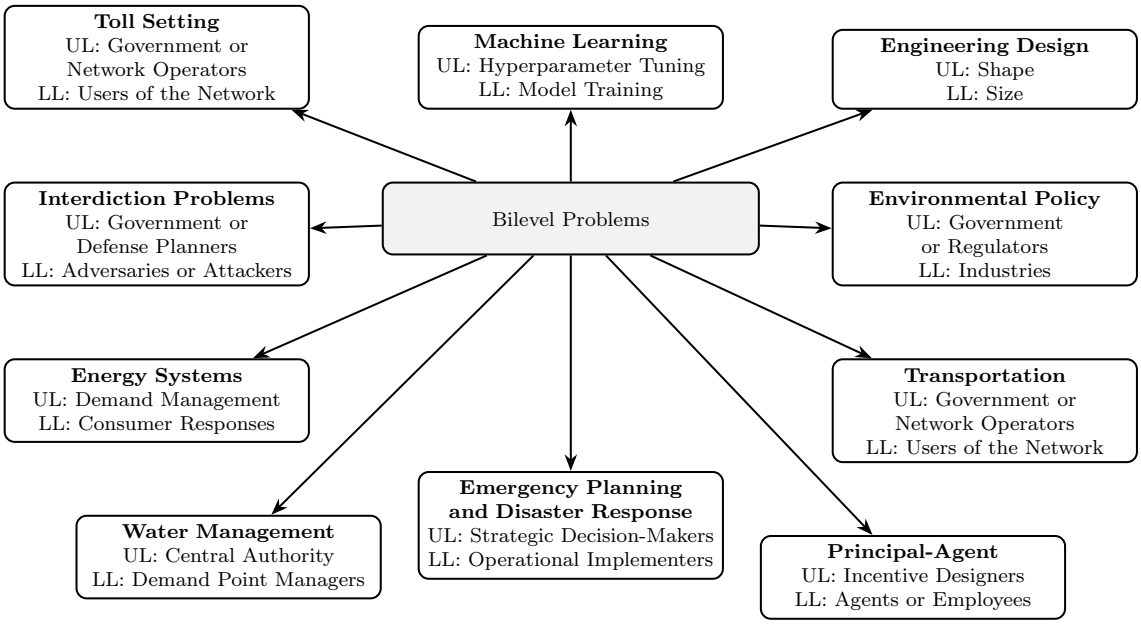


Figure 2.4: Examples of applications with bilevel optimization formulation.

is formulated as follows:

$$\begin{aligned} &\min_{x \in X} \quad F(x) \\ &\text{subject to} \\ &\qquad G(x) \leq 0, \\ &\qquad H(x) = 0, \end{aligned} \tag{2.1}$$

where $F(x)$ is the objective function to be minimized, $G(x) \leq 0$ represents inequality constraints (with the notation $G(x)$ used to denote one or more constraints for brevity), $H(x) = 0$ represents equality constraints, and X denotes the domain of the decision variables.

In a BOP, we extend this setup to include two levels, where the *leader* must take into account the optimal reaction of the *follower*. The leader's feasible decisions are thus constrained by the follower's response to any given choice. For simplicity, equality constraints are omitted from the following bilevel formulation.

Definition 2.2 (Bilevel Optimization Problem). A bilevel optimization problem is formulated as:

$$\begin{aligned} &\text{“}\min\text{”}_{x \in X, y \in Y} \quad F(x, y) \\ &\text{subject to:} \quad G(x, y) \leq 0, \\ &\qquad\qquad \min_{y \in Y} f(x, y) \\ &\qquad\qquad \text{subject to} \quad g(x, y) \leq 0, \end{aligned} \tag{2.2}$$

where $F(x, y)$ is the leader's objective function, $G(x, y) \leq 0$ represents the leader's constraints, $f(x, y)$ is the follower's objective function, $g(x, y) \leq 0$ represents the follower's constraints, and X and Y are the respective domains of the leader's and follower's decision variables. The notations $G(x, y)$ and $g(x, y)$ are used to denote one or more constraints for brevity. The quotation marks in the upper-level minimization indicate ambiguity, as explained later in this chapter.

Figure 2.5 illustrates this structure, where the optimal leader decision x^* and the corresponding follower response y^* depend on each other through their respective objectives and feasible regions.

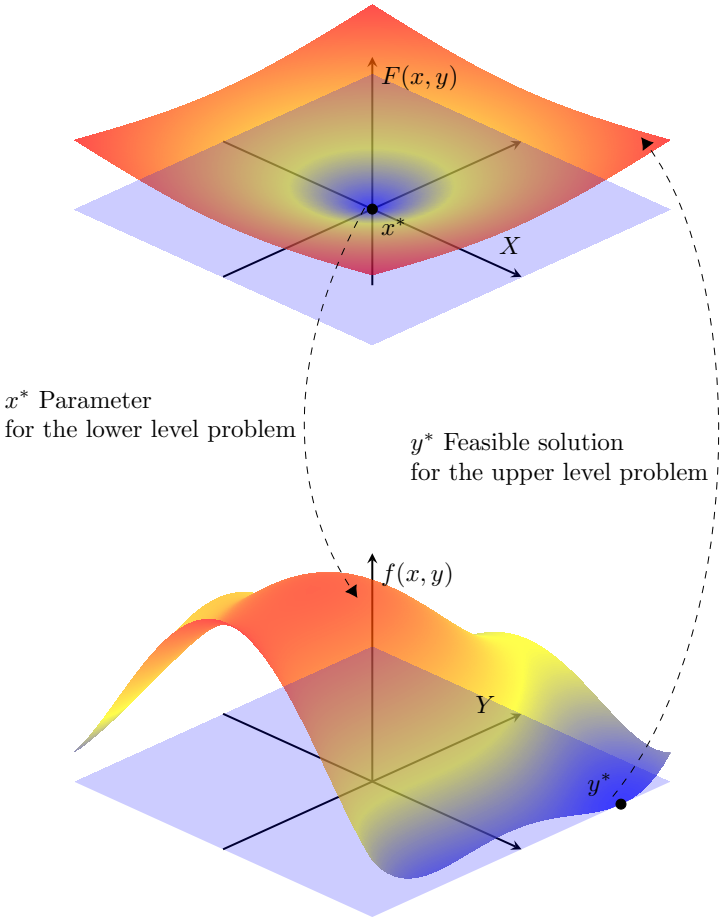


Figure 2.5: A sketch of a bilevel problem showing the inter-linkage between the upper and lower levels.

Reaction Set and Lower-Level Optimal Value Function: In BOPs, the follower's optimal response to each leader decision x is represented by the *reaction set* $\Psi(x)$, which includes all optimal values of y for a given x that are *feasible*. A solution is considered *feasible* if it satisfies all the constraints of the optimization problem. Formally, the reaction set is defined as:

$$\Psi(x) = \{y \in Y \mid y \in \underset{y' \in Y}{\operatorname{argmin}} f(x, y') \text{ and } g(x, y) \leq 0\}. \tag{2.3}$$

When $\Psi(x)$ is a singleton, the follower has a unique optimal response for each leader decision x, simplifying the bilevel optimization structure. However, when $\Psi(x)$ contains multiple optimal responses for a given x, this means that $|\Psi(x)| \geq 2$, indicating that $\Psi(x)$ is a set rather than a singleton, the leader faces uncertainty about the follower's choice within the reaction set, expressed in Definition 2.2 by the quotations in "min". To address this ambiguity, two main approaches can be adopted, the optimistic and the pessimistic approaches, which we will explain in detail in Section 2.4.4.

The *lower-level optimal value function* $\varphi(x)$ provides the minimum optimal value of

$f(x, y)$ for each leader decision x:

$$\varphi(x) = \min_{y \in \Psi(x)} f(x, y). \tag{2.4}$$

If these mappings $\psi(x)$ and $\phi(x)$ are known, the bilevel optimization problem can be rewritten as a single-level problem, simplifying the optimization.

Admissible and Reaction Sets, and the Inducible Region: The *Lower-Level Admissible Set* $S(x)$ represents all follower decisions that satisfy the follower's constraints for a given leader decision x:

$$S(x) = \{y \in Y \mid g(x, y) \leq 0\}. \tag{2.5}$$

Therefore we can say that the *Reaction Set* $\Psi(x)$, already defined above, includes all admissible, optimal responses of the follower for each leader decision x:

$$\Psi(x) = \{y \in S(x) \mid y \in \operatorname*{argmin}_{y' \in S(x)} f(x, y')\}. \tag{2.6}$$

The *Inducible Region* I captures all feasible pairs (x, y) where $y \in \Psi(x)$ (follower optimality) and $G(x, y) \leq 0$ (leader constraints):

$$I = \{(x, y) \mid y \in \Psi(x), G(x, y) \leq 0\}. \tag{2.7}$$

Constraints in Bilevel Problems: Bilevel problems can have constraints at both the upper and lower level. Wiesemann et al. [85] proposed a categorization of bilevel problems into dependent and independent bilevel problems. In dependent bilevel problems, the lower-level constraints are characterized by the relationship $g(x, y) \leq 0$, where g is a function of both x and y. In this case, the follower's feasible region is influenced by the leader's decisions, leading to the condition $S(x) \neq S(x')$ for some $x, x' \in X$, and the upper-level variables that appear in the lower-level constraints are called linking variables. In contrast, if the lower-level constraints are independent of the leader's decision x, the lower-level admissible set $S(x)$ remains unaffected, meaning $S(x) = S(x')$ for all $x, x' \in X$. However, the follower's optimal reaction set $\Psi(x)$, which depends on the lower-level objective function $f(x, y)$, may still vary if $f(x, y)$ includes x.

In the same vein, upper-level constraints can be non-connecting or connecting (also called coupling constraints). Non-connecting constraints are expressed as $G(x) \leq 0$ for $x \in X$ and do not involve the lower-level decision variable y. Consequently, y has no impact on the feasibility of the upper-level problem, as these constraints depend only on x; however, y still affects the upper-level problem through the objective function. In contrast, connecting constraints, also referred to as coupling constraints, take the form $G(x, y) \leq 0$ for $x \in X$ and $y \in \Psi(x)$. These constraints now depend also on the lower-level decisions and integrate the optimal reaction of the lower-level problem. The presence of connecting constraints has raised interesting questions in the bilevel optimization community. Some researchers argue that these constraints can be simply shifted to the lower-level problem, while others, such as Mersha et al. [86], contend that doing so undermines the fundamental modeling concept of hierarchy in bilevel programming. In this thesis, we deal with the general case where both levels can have any type of constraints.

2.4.2 Bilevel Optimization Problem: A Linear Example

To show the structure of a bilevel optimization problem, let us consider a linear example, where both the objectives and constraints are linear. A general linear bilevel problem can be written as:

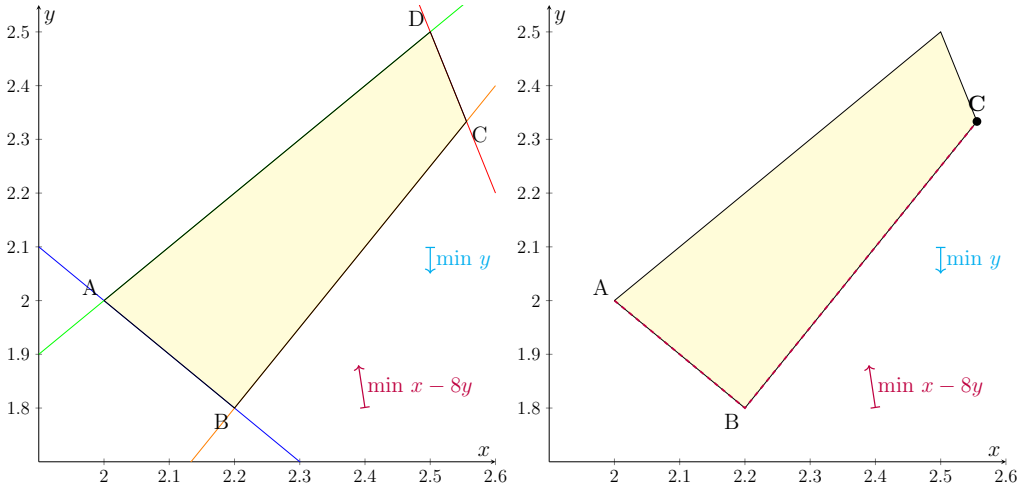
$$\begin{aligned} \min_{x\in X, y\in Y} \quad & F(x,y) = c^T x + d^T y \\ \text{subject to} \quad & Ax + By \leq b, \\ & y \in \operatorname{argmin} y \in Y \left\{ f(x,y) = e^T y + h^T x \mid Cx + Dy \leq q \right\}, \end{aligned} \tag{2.8}$$

where the leader seeks to minimize $F(x, y)$ over their constraints, while the follower minimizes $f(x, y)$ within their own feasible region, determined by the constraints $Cx+Dy \leq q$.

Now, let's look at a specific numerical example:

$$\begin{aligned} \min_{x\geq 0, y\geq 0,} \quad & F(x,y) = x - 8y \\ \text{subject to} \quad & y \in \operatorname{argmin}\{ f(x,y) = y \mid \\ & -x - y \leq -4, \\ & -x + y \leq 0, \\ & 3x + y \leq 10, \\ & -3x + 2y \geq -3\}. \end{aligned} \tag{2.9}$$

In this example, the leader's objective is $F(x, y) = x - 8y$, while the follower's objective is $f(x, y) = y$. Figure 2.6a illustrates the follower's feasible region defined by these constraints, along with the directions in which each level minimizes its objective. In Figure 2.6b, the inducible region—the set of feasible solutions respecting the bilevel structure—is represented by the line segments connecting points A, B, and C. Among these points, point C, located at (2.56, 2.33), is the optimal solution to the bilevel problem, yielding objective values of $F = -16.08$ for the leader and $f = 2.33$ for the follower. Although points A and B are also feasible, they do not minimize the leader's objective, demonstrating the importance of the hierarchical structure in determining the bilevel solution.



(a) Follower's feasible region. (b) Inducible region and optimal solution.

Figure 2.6: Example of a linear bilevel problem showing the follower's feasible region and inducible region.

2.4.3 Bilevel vs. Biobjective Problems

Optimization problems can have more than one objective, leading to what is known as a multiobjective optimization problem. We refer to a biobjective optimization problem when there are two objectives, and we can formulate it as follows:

$$\begin{aligned} \min_{x,y} \quad & (F(x,y), f(x,y)) \\ \text{subject to} \quad & G(x,y) \leq 0, \\ & g(x,y) \leq 0, \end{aligned} \tag{2.10}$$

where F and f are the two objectives to be optimized, x and y represent the decision variables, and G and g are the constraints.

In single-objective optimization, comparing solutions is straightforward: one solution is better or worse than another based on their objective values. However, in biobjective optimization, comparisons are more complex and rely on the concept of dominance. As defined by [87], a solution x_1 is said to dominate another solution x_2 if x_1 is no worse than x_2 in all objectives and strictly better in at least one objective. Based on this principle, the non-dominated set of solutions within the feasible region is called the Pareto set, and its image in the objective spaces of these solutions is known as the Pareto front.

Researchers have explored the connections between bilevel and biobjective optimization problems [88], [89]. While in certain specific cases a BOP can be reformulated as a biobjective problem, enabling solutions to be found using biobjective methods, this approach does not apply universally. Notably, there is no guarantee that an optimal solution to a BOP will lie on the Pareto-optimal front of a corresponding biobjective formulation.

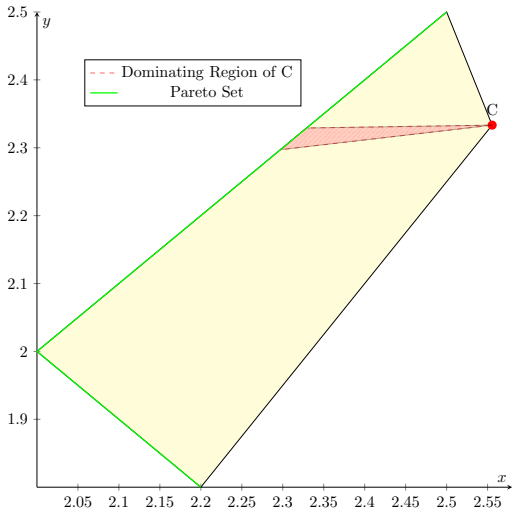
To better understand the differences between BOPs and biobjective optimization problems, we present a comparison inspired by Talbi in [90]. Let us reformulate the previous bilevel problem into a biobjective one as follows:

$$\begin{aligned} \min_{x\geq 0, y\geq 0} \quad & (F(x,y) = x - 8y,\ f(x,y) = y) \\ \text{subject to} \quad & -x - y \leq -4, \\ & -x + y \leq 0, \\ & 3x + y \leq 10, \\ & -3x + 2y \geq -3. \end{aligned} \tag{2.11}$$

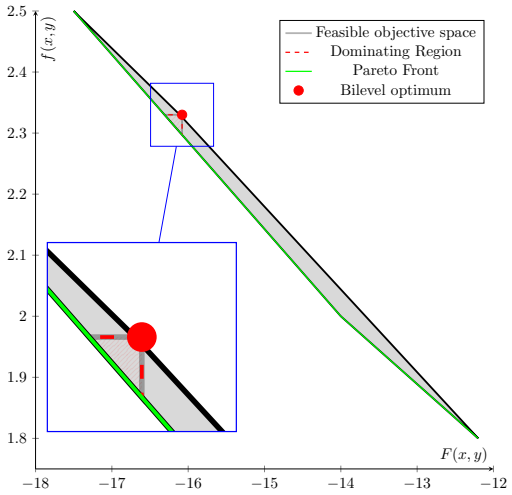
In this reformulation, F and f represent the upper and lower level objectives, respectively. However, we now optimize them independently within a single-level framework. Figure 2.7 compares the feasible decision spaces for the bilevel and biobjective formulations, both of which yield the same trapezoidal region. Figure 2.7b displays the objective space for the biobjective problem, where the red point marks the bilevel optimal solution. Notably, this solution is dominated by several alternatives—particularly those located within the red-filled triangle shown in Figure 2.7a, which lie on the Pareto front. These observations illustrate that a BOP does not necessarily correspond to a biobjective formulation with the same upper- and lower-level objectives. Consequently, the bilevel optimal solution is not always Pareto-optimal in the biobjective problem, and vice versa. Therefore, reformulating a bilevel problem as a biobjective one and applying Pareto dominance criteria may not always lead to equivalent solutions.

2.4.4 Optimistic vs. Pessimistic Problems

In (2.2), the upper-level minimization problem is in quotation marks to indicate the indeterminacy that arises when there is more than one optimal solution at the lower level for a



(a) Feasible Decision Space and Pareto Set. The yellow-shaded region represents the feasible decision space defined by the problem constraints. The green line denotes the Pareto set (the set of non-dominated decision vectors). The red dot highlights the bilevel optimum, which is dominated relative to the Pareto set.



(b) Feasible Objective Space and Pareto Front. The grey area is the projection of the entire feasible decision space into the objective space. The green line is the Pareto front, corresponding to the Pareto set in decision space. The red patterned region indicates the domination area associated with the bilevel optimum, and the red dot marks the bilevel optimal solution.

Figure 2.7: Comparison of feasible spaces in bilevel and biobjective optimization.

given upper-level decision vector. This creates ambiguity about which specific lower-level solution should be taken into account in the upper-level optimization, leading to an ill-posed problem due to the non-uniqueness of lower-level solutions. An ill-posed problem, as defined by Hadamard, is one that violates at least one of the following conditions for well-posedness: existence of a solution, uniqueness of the solution, or continuous (stable) dependence of the solution on the input data (i.e., small changes in the input should not cause disproportionately large changes in the solution) [91]. In this case, the violation is the uniqueness condition. To address this, the upper-level decision maker can adopt one of two positions: the optimistic or pessimistic stance. From a game theory perspective, the optimistic case is referred to as a strong Stackelberg game, while the pessimistic case is known as a weak Stackelberg game [92]. Let us consider the following example [93] to demonstrate this:

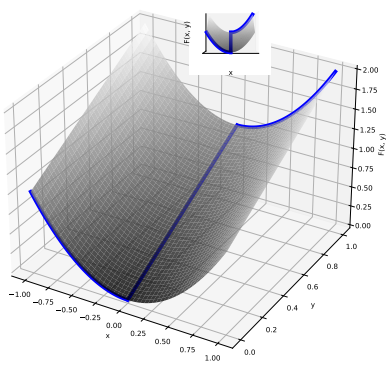
$$\begin{aligned} \min_{x\in X, y\in\Psi(x)} \quad & x^2 + y \\ \text{subject to} \quad & x \in [-1, 1] \\ \text{where} \quad & \Psi(x) = \operatorname*{argmin}_{y\in[0,1]} (-xy) \end{aligned}$$

The best response of the follower is given by:

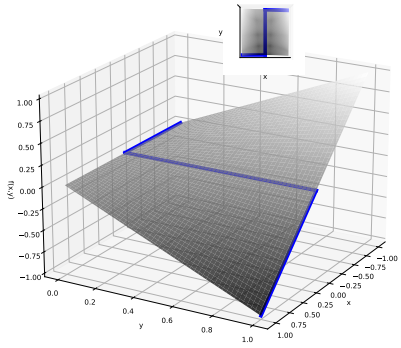
$$\Psi(x) = \begin{cases} [0,1], & \text{if } x = 0, \\ 0, & \text{if } x < 0, \\ 1, & \text{if } x > 0. \end{cases}$$

When $x = 0$, the follower has a range of optimal responses and can choose any value of y within this range. This results in multiple possible objective values for the upper-level decision-maker.

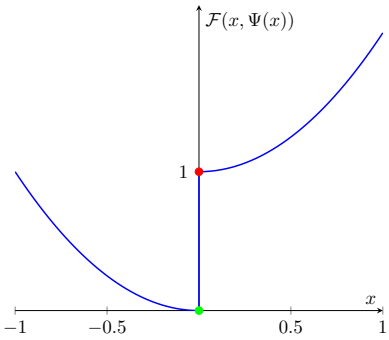
Figures 2.8a and 2.8b show the 3-D plots of the upper and lower-level optimization problems. The left subplot displays the upper-level objective, indicating the region where the upper-level decision maker seeks to minimize $F(x, y)$. The right subplot represents the lower-level objective, where the follower's best responses are identified. With blue lines, the optimal lower-level reaction set $\Psi(x)$ is depicted in both plots. At first glance, the bilevel optimal solution might appear to be $(\hat{x}, \hat{y}) = (0, 0)$. When examining $\Psi(x)$, we find that for $x = 0$, $\Psi(0) = \{y : 0 \leq y \leq 1\}$. At $x = 0$, the optimal solution value for the lower level is 0 for any y within that range. If the lower-level decision maker chooses $y = 0$, the upper-level decision maker can achieve the optimal solution $F(0, 0) = 0$. Instead, if the lower-level decision maker selects $y = 1$, the upper-level decision maker results in the less favorable outcome $F(0, 1) = 1$. Figures 2.8c shows the mapping of x to $F(x, \Psi(x))$, indicating the region where the upper-level decision maker seeks to minimize their objective function. Figure 2.8d depicts the mapping of x to $\Psi(x)$, illustrating the follower's reaction set. In both figures the optimistic and the pessimistic solutions are shown in green and red, respectively.



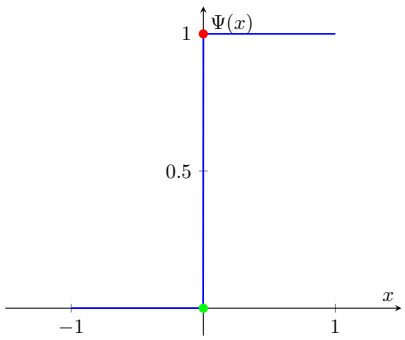
(a) Upper level Objective.



(b) Lower level Objective.



(c) x-F(x,$\Psi(x)$) mapping.



(d) x-$\Psi(x)$ mapping.

Figure 2.8: Example of an ill-posed bilevel problem.

Let us now provide the formal definitions of the two positions. As discussed, in the optimistic scenario, the upper-level decision maker assumes that the lower-level decision maker will select the best solution from the feasible solutions in $\Psi(x)$. One can formulate the problem in the following way, where $x \in X$ and $y \in Y$, with X and Y denoting the

respective domains of the upper-level and lower-level decision variables:

$$\begin{aligned} \min_{x} \quad & F(x,y) \\ \text{subject to} \quad & G(x,y) \leq 0, \\ & y \in \Psi(x), \\ & g(x,y) \leq 0. \end{aligned} \tag{2.12}$$

The lower-level decision maker's choice function in this context (best-case for the upper level) can be defined as:

$$\psi_{\mathrm{o}}(x) = \underset{y}{\operatorname{argmin}}\{F(x,y) : y \in \Psi(x)\}$$

In the pessimistic scenario, the upper-level decision maker assumes that the lower-level decision maker will choose the least favorable solution from the feasible set. One can rewrite the bilevel problem as follows:

$$\begin{aligned} \min_{x} \quad & \max_{y \in \Psi(x)} F(x,y) \\ \text{subject to} \quad & G(x,y) \leq 0, \\ & y \in \Psi(x), \\ & g(x,y) \leq 0. \end{aligned} \tag{2.13}$$

The lower-level decision maker's choice function in this context (worst-case for the upper level) can be expressed as:

$$\psi_{\mathrm{p}}(x) = \underset{y}{\operatorname{argmax}}\{F(x,y) : y \in \Psi(x)\}$$

This in essence is a trilevel optimization problem, formulated as follows:

$$\begin{aligned} \min_{x} \quad & F(x,y) \\ \text{subject to} \quad & G(x,y) \leq 0, \\ & \min_{y} f(x,y) \\ & \text{subject to:} \quad \max_{y} F(x,y) \\ & \qquad\qquad\qquad\quad g(x,y) \leq 0, \\ & \qquad\qquad\qquad\quad G(x,y) \leq 0. \end{aligned} \tag{2.14}$$

It is essential to recognize that the pessimistic position is generally less tractable than the optimistic one. This is often due to the reformulations that can introduce non-convex functions, resulting in problems that are not well-behaved, and because the pessimistic position involves this trilevel structure [94].

2.4.5 Computational Complexity

Bilevel optimization problems are difficult to solve, even when F, f, X, Y, and $\Psi(x)$ are convex. This complexity persists even in cases involving two linear levels. In fact, the authors in [95] and [96] proved that these problems are NP-hard. Later research [97], [1] proved that linear bilevel problems are not only NP-hard but also strongly NP-hard. Additionally, it has been demonstrated in [98] and [29] that determining local optimality for a specific solution—essentially checking whether a given point is a local minimum of a bilevel problem—is also NP-hard. Furthermore, the number of local optimal solutions may grow exponentially, as noted in [99]. Verifying bilevel feasibility becomes strongly NP-hard when one loses the convexity property at either level.

2.4.6 Connections with Other Domains

Bilevel optimization is not limited to one area of optimization; it intersects with various fields and methods. By reformulating problems or exploring specific instances of bilevel structures, we can apply it to different domains. The following are some of these key areas:

- Stackelberg Games: As previously mentioned, bilevel optimization can be viewed through the lens of Stackelberg games, where the decision-making hierarchy between the leader and follower reflects similar strategic interactions. This connection allows for the application of game-theoretic concepts to analyze and solve bilevel problems, and vice versa.

- Min-Max Problems: Min-max problems are a type of bilevel problem where $F(x, y) = -f(x, y)$. These problems often arise in optimization under uncertainty, where the goal is to prepare for the worst-case scenario, instead of focusing on the most likely conditions. This approach is closely related to robust optimization, and is discussed further in Chapter 5.

- Generalized Semi-Infinite Problems: Generalized semi-infinite problems involve an infinite number of constraints or variables. Bilevel optimization and semi-infinite problems are related, particularly in two cases: (1) the optimistic case and (2) when the lower-level rational set $\Psi(x)$ is a singleton. In these situations, the bilevel program can be reformulated as a semi-infinite program using the lower-level optimal value function $\varphi(x) = \min\{f(x, y) \mid g(x, y) \leq 0\}$. This transformation facilitates decomposition while preserving dependence through infinite constraints. More about semi-infinite programming can be found in [100]–[102].

- Set-Valued Optimization Problems: Set-valued optimization problems involve objective and constraint mappings that assign a set of possible values rather than a single value to each decision variable [103]. These problems are closely linked to bilevel optimization because the follower's response set, $\Psi(x)$, is naturally a set-valued mapping. For a given leader's decision x, $\Psi(x)$ represents the set of all optimal solutions to the follower's problem. When $\Psi(x)$ contains multiple solutions, the leader faces ambiguity and must adopt either an optimistic approach (assuming the follower selects the leader-favorable solution) or a pessimistic approach (assuming the follower chooses the least favorable solution). This relationship introduces an additional layer of complexity to bilevel optimization, and the connections between these two fields have been explored in [104], [105].

- Mathematical Problems with Equilibrium Constraints: Mathematical problems with equilibrium constraints are characterized by optimization problems, where some constraints are defined as variational equalities that must satisfy equilibrium conditions. Such problems can be expressed as follows:

$$\begin{aligned} \min_{x \in X, y \in Y} \quad & F(x, y) \\ \text{subject to} \quad & G(x, y) \leq 0 \\ & \nabla_y f(x, y) = 0 \end{aligned}$$

  When the function $f(x, y)$ is convex, these problems can often be reformulated as bilevel optimization problems. If the response set $\Psi(x)$ is a singleton for every leader decision x, the equilibrium condition $\nabla_y f(x, y) = 0$ can be replaced by the follower's optimal response $y = \text{argmin}\{f(x, y) \mid g(x, y) \leq 0\}$.This transformation highlights

the close connection between bilevel programming and mathematical programs with equilibrium constraints. It provides a simpler way to formulate these problems, which has proven useful in many studies. For a deeper look into these connections and the related optimality conditions, Dempe's annotated bibliography offers a great starting point [106].

- Multilevel Problems vs. Multilevel Approaches: In the literature, "multilevel optimization" is sometimes used to refer to a specific type of optimization problem and other times to an approach in optimization process. To avoid confusion, it's important to distinguish between the two: multilevel optimization problems refer to hierarchical problems with multiple levels, with bilevel optimization being a specific case of two levels. Multilevel approaches, on the other hand, refer to methods that integrate information from different scales or levels of a problem, such as varying model details, adjusting simulation fidelity, or considering spatial and temporal factors. The terms "multilevel," "multi-fidelity," "multi-scale," "multi-grid," and "multistage" are often used interchangeably in this context. Examples include [107], [108], [109], [110], [111], [112], and [113].

2.5 Overview of Methods

This section provides an overview of classical and evolutionary algorithms for solving single-objective bilevel problems. Classical methods work well for problems where assumptions include linearity and convexity. On the other hand, EAs are better at solving more challenging bilevel problems, such as with non-convexity, discrete variables, and nonlinear objective functions.

2.5.1 Classical Methods for BOPs

Classical algorithms in bilevel optimization are developed to address either the optimistic or pessimistic bilevel problem. Most research has focused on linear or convex functions, where assumptions like differentiability and lower semicontinuity simplify analysis. This has led to many classical methods that deal with the linear bilevel problem. The following subsections provide an overview of methods for both optimistic and pessimistic approaches.

2.5.1.1 Methods for the Optimistic Approach

Methods Based on KKT Conditions: The Karush-Kuhn-Tucker (KKT) conditions are essential for solving constrained optimization problems, extending the method of Lagrange multipliers to include inequality constraints. These conditions encompass essential requirements, such as the gradients of the objective and constraint functions, feasibility conditions, and complementary slackness. In bilevel optimization, KKT conditions are applied on the lower-level problem to turn it into a set of constraints for the upper-level problem. This turns the bilevel problem into a single-level problem, which works effectively when the lower-level problem is smooth and convex [114], [115]. Further details, including an example, are provided in Appendix A. Chenggen Shi et al. [116] extended the KKT method for linear bilevel programming (BLP), particularly for problems with random linear upper-level constraints, addressing cases that standard methods cannot solve. In addition to this, a later study [117] combined goal programming, KKT conditions, and penalty functions to provide a more complete method for linear BLPs. In a study on mixed-integer bilevel linear programming (MIBLPP) [118], the problem is reformulated as a single-level

one using KKT conditions when the lower-level continuous functions are convex and differentiable with fixed integer values. This method splits the MIBLPP into a mixed-integer program with complementarity constraints by separating the continuous and integer variables. This ensures the global and local minimizers align under specific conditions and broadens the range of MIBLPP solutions, particularly in non-convex problems.

Descent methods: Descent methods iteratively adjust the current solution to reduce the objective function. They work by calculating the gradient of the objective and updating the solution in the opposite direction, continuing this process until a stopping criterion is satisfied to approach a local minimum. In bilevel optimization, a descent direction must not only decrease the upper-level objective but also maintain feasibility by ensuring the point remains optimal for the lower level. This requirement makes finding a descent direction more complex, as it must balance improving the upper-level objective while preserving lower-level optimality. For example, [119] used a gradient descent method to find the minimum of a smooth convex function while following the constraints set by another smooth convex function. This avoided the more complicated penalization subproblems and projections that are common in constrained optimization. In another study, [29] introduced two descent methods tailored for bilevel problems where the lower-level problem involves strictly convex quadratic optimization, proposing both a pivot-based algorithm and a modified steepest descent method. They addressed local optimality challenges and demonstrated that verifying local optimality in bilevel programming is NP-hard.

Penalty function methods: Penalty function methods are optimization techniques that transform constrained problems into unconstrained ones by embedding constraints within the objective function through penalty terms. These methods involve solving a series of penalized problems and adjusting penalty parameters iteratively to increasingly enforce the constraints. In bilevel optimization, penalty function methods incorporate terms from the lower-level problem's constraints and objectives into the upper-level objective function. This reformulation simplifies the bilevel structure by converting it into a single-level problem, making it more tractable for optimization. In particular, such methods work quite effectively for linear and quadratic bilevel problems where both levels have well-defined structures that enable systematic penalty adjustments. They also behave well for optimistic bilevel problems where the lower-level problem is strongly convex, which guarantees a unique and well-behaved solution of the lower level that can be integrated smoothly into the upper-level problem using penalties. For example, the objective penalty method suggested in [120] works well for optimistic bilevel problems because it takes into account the hierarchical structure by correctly including the lower-level solution in the optimization at the upper level. Additional exact and inexact penalty methods in [121] facilitate feasibility and optimality in optimistic bilevel cases. Further, [122] explores exact penalization and necessary optimality conditions for generalized bilevel problems, focusing on cases where the lower level is formulated as a variational inequality and the upper-level objective is uniformly Lipschitz continuous.

Trust region: Trust region methods are a class of iterative techniques for nonlinear optimization that require, in each step, the construction of a so-called "trust region" around the current estimate of the optimum within which a simpler model—usually, quadratic—is an adequate approximation of the objective function. The solution to this subproblem defines, within the trust regions, the step to take. The region is expanded if the model is good and reduced otherwise. In bilevel optimization, trust region methods are valuable for handling the nested complexity of upper and lower-level objectives. They ensure each step remains within a region where both objective models are reliable, effectively managing the dependencies between the two levels. This can result in the convergence and balancing of optimizations at both levels. It has been shown that trust region algorithms work well for

nonlinear bilevel problems. They do this by improving the lower level solution with each iteration, which then helps with more accurate optimization at the upper level [123] [124].

2.5.1.2 Methods for the Pessimistic Approach

Various solution methods have addressed pessimistic bilevel optimization. For a detailed overview, readers may refer to two key resources: [125], which surveys methods and solution approaches, and [126], which covers foundational models, definitions, and applications. Key methods include:

Penalty Methods: In pessimistic bilevel programming, penalty methods simplify the bilevel problem by transforming it into a single-level optimization problem, adding a penalty term to the leader's objective function. The general form of the penalized problem is:

$$\min_{x \in X} \sup_{y \in \Psi(x)} \{F(x, y) + k \cdot \xi(x, y)\},$$

where $F(x, y)$ is the leader's objective, $\Psi(x)$ denotes the set of the follower's optimal solutions, and $\xi(x, y)$ represents a penalty function that enforces the follower's constraints or optimality conditions. The penalty parameter k is iteratively adjusted to ensure that the solutions of the penalized problem converge to those of the original pessimistic bilevel problem.

The method by Aboussoror and Mansouri [127] introduces an exact penalty approach for weak linear bilevel problems by incorporating a penalty term directly into the leader's objective. Their method addresses the duality gap in the follower's problem by solving a series of penalized problems that converge to the original solution as the penalty parameter increases. Although this approach guarantees solution existence under specific conditions, such as a polyhedral feasible region for the leader, it is primarily theoretical and does not include computational results. Building on this, [128] extends the framework with a new exact penalty method that integrates both upper and lower-level constraints and optimality conditions. Unlike Aboussoror and Mansouri's approach, which focuses on the upper-level problem, this method directly incorporates both levels into a unified framework. The authors provide theoretical proofs for solution existence and demonstrate the method with numerical examples. Further refining this approach, [129] proposes a penalty method for weak linear bilevel problems that, like earlier methods, adds a penalty term to the objective. They ensure that the resulting penalized problem forms a finite polyhedral constraint region, allowing the bilevel problem to be reformulated as a single-level optimization problem using Kuhn-Tucker conditions. This paper includes theoretical proofs and numerical examples to demonstrate practical implementation.

Kth-Best Algorithm: The Kth-Best algorithm is a method used to systematically explore and evaluate multiple feasible solutions in an optimization problem, starting with the best (optimal) solution. This approach is particularly useful in situations where the optimal solution might depend on various scenarios or responses from a secondary decision-maker, as in bilevel programming. In [130], Zheng, Fang, and Wan adapt this algorithm to solve weak linear bilevel programming problems, where the leader needs to consider different potential responses from the follower. Their method involves generating a set of vertices representing feasible solutions for the leader's problem and evaluating the corresponding follower responses for each vertex. The algorithm ranks these vertices by solving the leader's problem under the assumption that the follower will choose the worst possible response at each step. This iterative process continues by selecting the next-best vertex that has not been dominated by previous solutions, ensuring that the final decision is robust. The authors provide theoretical validation for their approach and include a numerical

example to demonstrate its application.

Approximation and Relaxation Methods: Approximation and relaxation methods are commonly used in optimization to simplify problems. By modifying constraints, these techniques make the problems more tractable and improve computational efficiency. A study by [85] examines the independent pessimistic bilevel problem, providing conditions for the existence of solutions, analyzing its computational complexity, and proposing convergent approximation methods and iterative algorithms. Their approach handles non-convex instances with specific independence properties, using semi-infinite programming discretization techniques to manage these complex cases, with computational results demonstrated on benchmark instances. Dempe et al. expand the two-level optimal value function method to include non-smooth data and give specific optimality conditions for pessimistic bilevel problems. This approximation method effectively manages the challenges posed by non-smoothness [131]. Another approach in [132] applies the Scholtes relaxation technique to pessimistic bilevel problems, extending its success from the optimistic case. This method uses Slater's qualification and differentiability assumptions to rewrite the KKT problem and includes a relaxation algorithm that makes sure it converges to the best solutions to the original problem. Theoretical results confirm the algorithm's robustness and parametric optimization feasibility. In pessimistic bilevel optimization, Lignola and Morgan suggest viscosity solutions instead of exact solutions. These solutions use inner regularizations to get closer to the lower-level solutions. This method allows the leader to attain a security value under suitable conditions, providing a viable solution where exact solutions are unattainable [133].

Perturbation Techniques: Perturbation techniques are often applied in bilevel optimization, particularly to address ill-posed bilevel problems. Molodtsov's seminal work [134] from 1976 on non-antagonistic two-player games introduces a method for simplifying decision processes in weak Stackelberg games by introducing a perturbation parameter ϵ. In this framework, the leader maximizes their payoff function $F(x, y)$ by selecting a strategy x, while anticipating the follower's response to maximize their own payoff function $G(x, y)$. Molodtsov's approach modifies the follower's payoff, reformulating it as $G_\epsilon(x, y) = G(x, y) - \epsilon F(x, y)$, with the follower's response set defined as $S_\epsilon(x)$, which consists of the follower's optimal responses to the perturbed payoff. The leader's problem can then be expressed as:

$$u_\epsilon = \max_{x \in X} \min_{y \in S_\epsilon(x)} F(x, y)$$

With continuity and compactness conditions, solutions to the perturbed problem converge to the leader's guaranteed payoff as $\epsilon \to 0$. Since then, many bilevel optimization applications have extended the concept. For example, Bialas [135] used the lower-level perturbation in resource allocation problems [136]. Cao [137] extends the perturbation approach to solve linear bilevel problems under partial cooperation, resulting in optimistic, pessimistic, and intermediate solutions. Loridan [138] uses Molodtsov's approach to tackle strong Stackelberg (optimistic) bilevel problems,while also addressing some relations to pessimistic cases.

Expanding on the principles underlying these perturbation techniques, in Chapter 3, we present a δ-perturbation approach specifically designed for bilevel optimization problems with multiple lower-level solutions. By applying a perturbation in both optimistic and pessimistic formulation, this method ensures an (approximate) unique solution at the lower level, making optimistic and pessimistic solutions tractable for iterative algorithms [139].

Reduction Methods: Reduction methods reformulate the bilevel problem into a single-level form. A reduction method is suggested in [140] to solve pessimistic bilevel optimization problems. It works by first creating a relaxation of the original problem

that keeps its main structure. The method applies a two-step "Relaxation-and-Correction" scheme: initially relaxing the problem to simplify its structure and then reintroducing constraints to ensure the solution meets the pessimistic criteria of the original problem. Specifically, the author reformulates the pessimistic bilevel problem as a regular optimistic bilevel programming problem, enabling solutions to be computed more readily. Numerical examples demonstrate the effectiveness of this approach on linear pessimistic bilevel problems. In the same way, [141] talks about a reduction method for pessimistic bilevel optimization problems where the upper-level objective function is convex quadratic with respect to the leader's variables and concave quadratic with respect to the follower's variables, but the lower-level objective function stays linear. This method reformulates the problem by incorporating the follower's optimal response into the leader's objective or constraints, resulting in a single-level nonconvex problem dependent solely on the leader's decision variables. The transformed problem is then solved using global search techniques such as difference-of-convex programming. Despite the strong theoretical foundations of this approach, its practical evaluation remains limited due to the limited availability of computational experiments.

2.5.2 Evolutionary Algorithms for BOPs

Evolutionary Algorithms (EAs) have gained popularity for solving BOPs due to their flexibility in handling complex search spaces. Unlike traditional methods, which rely on assumptions like linearity or convexity, EAs do not require such assumptions and can effectively tackle real-world problems.

Conducting a search on Scopus using the terms "hierarchical optimization", "two-level optimization", "bi-level programming", "bi-level optimization", "evolutionary algorithms", "genetic algorithms", "evolutionary strategies", "differential evolution", or "evolutionary computation" reveals an increasing amount of research on these topics. The growth in the number of publications, as represented in Figure 2.9, indicates the increase in interest in EAs for solving problems related to BOP.

Sinha et al. [142] and Talbi [90] categorized BLEAs into four groups: single-level reduction, coevolutionary, nested, and surrogate-assisted. Building on these categories, we adopt the same classification and introduce additional subcategories for the nested and surrogate-assisted approaches, as shown in Figure 2.10, which are discussed further. The references provided are illustrative examples and not exhaustive.

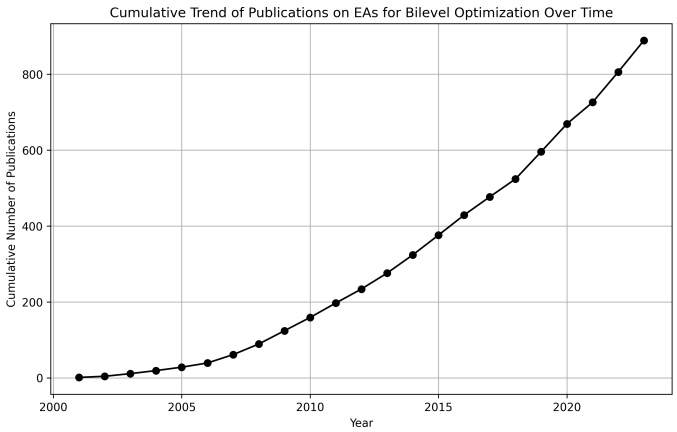


Figure 2.9: Number of cumulative publications on Evolutionary Algorithms for Bilevel Optimization Problems over time in Scopus.

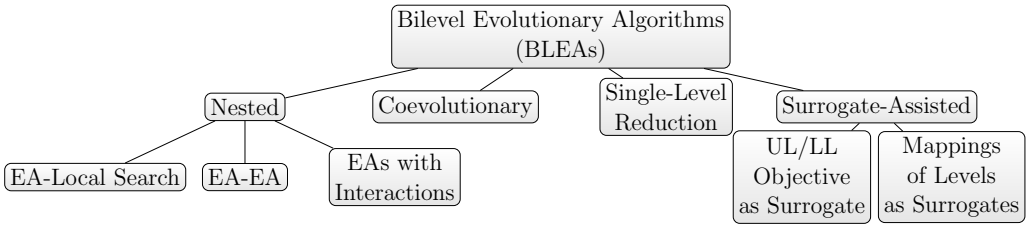


Figure 2.10: Classification of Bilevel Evolutionary Algorithms (BLEAs).

Single-Level Reduction: This technique simplifies a bilevel optimization problem by transforming it into a single-level optimization problem, making it solvable with traditional evolutionary algorithms. The general approach is illustrated in Figure 2.11. Typically, KKT conditions are applied to the lower-level problem, while the upper-level problem may vary in complexity. An early example of this reduction is in [143], where a two-level mixed zero-one programming problem is reformulated into a single-level problem using genetic algorithms. In this case, the leader's zero-one decision variables are encoded as bit strings, and the follower's continuous decision variables are optimized through linear programming to compute Stackelberg solutions. Similarly, in [144], the bilevel programming problem is transformed by applying KKT conditions to the lower-level problem, resulting in a single-level formulation that is solved using a genetic algorithm. To improve computational efficiency, the problem is further decomposed, and theorems are applied to reduce the search space. A practical application of single-level reduction is presented in [145], where the KKT method is used to optimize the sizing of battery energy storage systems and microgrid operations, employing a hybrid binary particle swarm optimization and quadratic programming method for both binary and continuous decision variables. Further advancements in [146] and [147] tackle nonlinear bilevel programming problems by converting them into single-objective nonlinear programming problems. These methods utilize customized crossover operators and constraint-handling schemes to address both linear and nonlinear constraints, working well even when the leader's objectives are non-differentiable. While single-level reduction methods are versatile and widely applicable, they do have limitations. A common assumption is that the lower-level problem is well-behaved (e.g., convex and continuous), which may not always hold in real-world applications. Moreover, these approaches can still require a high number of function evaluations, with methods like those in [146] and [147] needing up to 100,000 evaluations, even for small-scale bilevel problems.

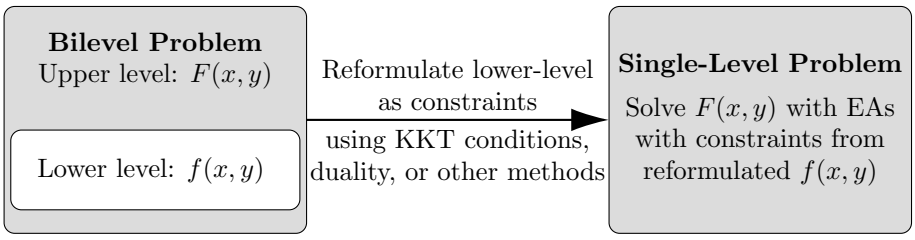


Figure 2.11: Singe level reduction method.

Co-evolutionary approaches: A co-evolutionary algorithm simulates the natural process of species evolving together, where the evolution of each species depends on the others. Unlike most evolutionary algorithms that operate with a single population, co-evolutionary algorithms involve multiple populations evolving in parallel. Each population represents a distinct part of the problem, and they interact by sharing information about

the progress made by the other populations. This interaction can either be competitive, where the populations compete with each other to outperform each other, or cooperative, where the populations collaborate to construct a single, globally superior solution. In the context of bilevel optimization, co-evolutionary methods employ two disparate evolutionary algorithms: one for the upper level and one for the lower level. During the optimization process, the two algorithms interact, with the upper-level algorithm using feedback from the lower level to optimize its decisions at each step, and the lower-level algorithm updating its solution refinements based on changes in the upper level. Through such a process, evolution balances the important interdependencies between the two levels, often improving overall solution quality. Figure 2.12 illustrates the core concept of this approach.

A co-evolutionary method for solving bilevel optimization problems was introduced in [148] with the development of BiGA (Bilevel Genetic Algorithm). This algorithm iteratively tackled two optimization tasks: the leader problem, which involved optimizing all the x variables and a subset of the y variables associated with the optimal basis of the follower's problem, and the follower problem, where the x variables were fixed while optimizing the remaining y variables. A basic genetic algorithm with full operators was employed in a dual-population environment to solve both the leader and follower problems, demonstrating robust performance across different classes of bilevel optimization problems. Additionally, [149] proposed CoBRA, a parallel co-evolutionary algorithm for bilevel optimization, expanding on the ideas behind BiGA. CoBRA is a co-evolutionary metaheuristic consisting of two evolving sub-populations, each corresponding to one level, with periodic information exchange between them. Applied to a bilevel transportation problem, CoBRA achieved better results than those obtained using a classical hierarchical approach. A further development in co-evolutionary bilevel optimization is seen in [150], where the Energy-based Co-evolutionary Decomposition Algorithm (E-CODBA) combines co-evolution, decomposition, and energy management principles to tackle the challenges of bilevel optimization in discrete settings, particularly for complex problems like the bilevel multi-depot vehicle routing problem. While E-CODBA reduces function evaluations through its energy-based structure, its effectiveness is heavily dependent on the precise calibration of energy management parameters, which can make it challenging to adapt to different problem landscapes. To address some of these limitations, an extended version called CODBA-LS incorporates a variable neighborhood search at the lower level to improve convergence rates by enhancing the local search process [151]. CODBA-LS achieves faster and often higher-quality solutions, especially in production-distribution applications within supply chain management. However, with the addition of local search, CODBA-LS also inherits dependencies on heuristic tuning and structure-specific configurations, which may restrict its adaptability to diverse bilevel problems and scalability for larger instances.

Despite these advancements, co-evolutionary approaches in bilevel optimization still face limitations. Managing multiple evolving populations increases both the complexity and computational cost, especially as the problem scale grows. Additionally, ensuring proper synchronization and interaction between populations is crucial, as poor coordination can lead to suboptimal solutions or slow convergence. These challenges necessitate careful design and tuning, which can reduce the flexibility of co-evolutionary approaches across varied bilevel landscapes and larger-scale applications.

Nested methods: The nested method for bilevel optimization involves solving the lower-level problem for each candidate solution of the upper-level problem. Essentially, for every possible set of upper-level decision variables, the corresponding lower-level problem is fully optimized. The solution of the lower-level problem is then used to evaluate the upper-level objective function, with the lower-level problem embedded within the upper-level

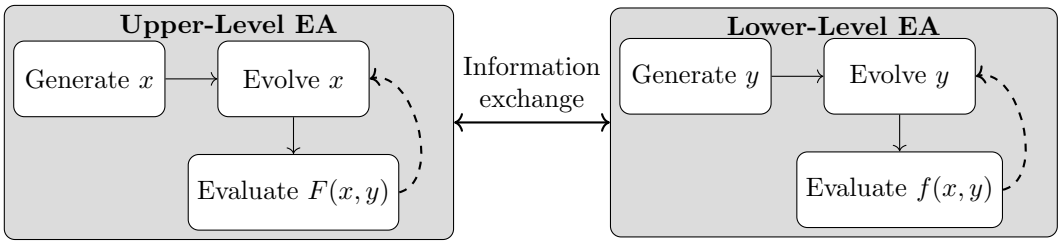


Figure 2.12: Co-evolutionary approach for solving BOPs. The two metaheuristics evolve in parallel and cooperate via information exchange [90].

optimization (see Figure 2.13). This process ensures that upper-level decisions account for the optimal reactions of the lower level, but it can be computationally expensive, as the lower-level problem must be solved for every evaluation of the upper-level decision variables.

Nested evolutionary approaches adapt this concept by integrating evolutionary algorithms at one or both levels, resulting in three main variations:

- **EA-Local Search:** In this variation, an evolutionary algorithm optimizes the upper-level problem, while a simpler local search or exact solver handles the lower-level problem. This approach works well when the lower-level problem can be solved with exact methods, such as linear programs. A classic example is GABBA, which employs a genetic algorithm for the upper-level decision vector while directly solving the follower's linear problem [152]. Another example is the genetic-algorithm-based approach for bilevel transportation system planning models, which uses genetic algorithms at the upper level and handles the lower-level problem with sensitivity analysi [153]. More recently, [154] introduced a memetic algorithm at the upper level, combined with global or local search techniques for the lower level, offering a flexible solution when the lower level can be efficiently solved through exact or analytical methods.

- **EA-EA:** In this form, evolutionary algorithms are applied to both the upper and lower levels, making it suitable for bilevel problems where both levels are complex and cannot be simplified with exact methods. Here, an evolutionary algorithm optimizes the upper level, and a separate evolutionary algorithm solves the lower level for each candidate solution. For example, a differential evolution (DE) method is applied at both levels in [8] to handle bilevel programming problems, while the nested bilevel evolutionary algorithm (N-BLEA) employs a steady-state genetic algorithm at both levels [155]. Although EA-EA is effective for complex bilevel problems, it requires considerable computational resources and scales poorly as the number of lower-level optimizations grows with the upper-level variables, making it impractical for large or highly complex problems.

- **Evolutionary Algorithms with Interactions:** These approaches employ some kind of interaction between the upper and lower-level evolutionary algorithms, which share information during the optimization process for better search exploration. As an example, [12] proposed a covariance matrix adaptation evolution strategy-based evolutionary bilevel optimization algorithm, in which the information from the upper-level optimizer is shared with the lower-level optimizer to assist with the algorithm's search directions. Similarly, [156] proposed an evolutionary multitasking bilevel optimization algorithm where an implicit information-sharing mechanism was inbuilt into

the algorithm architecture. This form can lead to improved convergence rates and solution accuracy. Moreover, the authors in [157] propose a transfer learning-based parallel evolutionary algorithm (TLEA) framework for bilevel optimization, designed to enhance lower-level search processes through a transfer learning strategy. They implement and test two versions of TLEA—using covariance matrix adaptation and differential evolution operators—demonstrating improved performance compared to existing evolutionary bilevel optimization algorithms on benchmark problems.

In general, nested evolutionary methods can be quite useful for complex, real-world bilevel optimization problems and surely serve as a baseline for more advanced techniques. However, they are computationally demanding and require careful tuning of the parameters for the involved EAs.

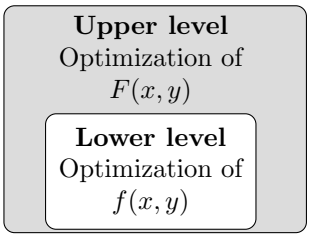


Figure 2.13: An illustration of the nested evolutionary approach, where lower level is optimized inside the upper level.

Surrogate-assisted methods: Surrogate-assisted evolutionary algorithms (SAEAs) integrate surrogate models into the evolutionary process to reduce computational costs by approximating expensive objective function evaluations. SAEAs use surrogate models, also called meta-models, to approximate with statistical models the objective functions or constraints. By training first the surrogates on limited number of real evaluations, they then use them during the optimization. These models then are much faster than directly solving the original problem. This makes them particularly useful for large-scale problems and problems that require expensive simulations. The general flow of these algorithms is illustrated in Figure 2.14, showing the classical EA loop enhanced with steps for training, updating (optional), and refining a surrogate model. In bilevel optimization, surrogate-assisted methods tackle the hefty computational burden of solving these nested problems. By making evaluations more efficient and cutting down on the number of function evaluations (FEs), these approaches streamline the process without losing sight of the critical interactions between the upper and lower levels. This preservation of essential information often translates into better overall optimization performance. Surrogate models, in particular, can be integrated into the optimization framework in two key ways:

- **Replacing the Upper-Level Objective Function and/or Lower-Level Objective Function:** Many applications replace either the upper-level or lower-level objective function using surrogate models, especially when they depend on expensive simulations common in engineering design. For instance, [158] proposed a surrogate-based bilevel shape optimization (SBSO) method that uses surrogate models to approximate the upper-level objective of maximizing the lift-to-drag ratio in blended-wing body underwater gliders, thereby minimizing the need for costly CFD simulations. This approach integrates Kriging-assisted teaching-learning-based optimization (KTLBO) with the non-dominated sorting genetic algorithm II (NSGA-II) at the upper level, combining surrogate modeling with heuristic techniques. Different surrogate models can be combined to improve optimization efficiency at both levels.

For example, [159] presents a bilevel surrogate-assisted evolutionary algorithm that integrates two separate Kriging models: one as a surrogate for the upper-level objective and another for the Ψ-mapping at the lower level. This approach minimizes the need for repeated, expensive function evaluations by approximating both the upper-level objective and the lower-level response, allowing the optimization to proceed efficiently without exhaustive evaluations at each stage. Surrogate models are also used to replace the lower-level objective function $f(x, y)$, which can be especially beneficial due to the nested nature of bilevel problems. Since the lower-level evaluations occur within each upper-level iteration, computational demands can escalate significantly when $f(x, y)$ relies on costly simulations. This lets the EAs run faster without having to do full evaluations at every iteration.

- **Approximating Mappings:** Surrogate models can also approximate mappings between the upper and lower levels, such as the reaction set mapping Ψ and the lower-level optimal value function ϕ. In [160], the authors present a DE approach assisted by a surrogate model for bilevel programming. This method employs nested DE at both levels, using a similarity-based surrogate model to approximate the Ψ-mapping at the lower level. The surrogate reduces the need for expensive lower-level optimization, enabling faster convergence while retaining solution quality. Similarly, [161] proposes an approximation method that uses Kriging with the Ψ-mapping, applied to the engineering design of a thin-walled pressure vessel, achieving improved solutions compared to traditional methods. Additionally, kernel interpolation as a surrogate model within a DE framework facilitates efficient exploration by approximating the interactions between decision variables across both levels [162]. The Surrogate-Assisted Bilevel Algorithm (SABLA) proposed in [163] uses the Ψ-mapping to approximate lower-level optimal responses, employing multiple surrogates—specifically, Response Surface Models (RSM) of orders 1 and 2 and Kriging—to model the lower-level objectives and constraints. This method combines DE at both levels, with the lower-level DE further refined by a local search. SABLA's use of selective re-evaluation and periodic updates of the surrogate models improves both optimization accuracy and computational efficiency. The Bilevel Evolutionary Algorithm based on Quadratic Approximations (BLEAQ), proposed in [164], employs quadratic approximations (QA) at the lower level, combined with a steady-state Genetic Algorithm (GA) at the upper level to improve the optimization process. BLEAQ uses the Ψ-mapping to capture the lower-level response in terms of the upper-level variables. Later, [165] extended this approach to BLEAQ2, which iteratively approximates both the rational reaction mapping Ψ and the optimal value function mapping ϕ, dynamically selecting one based on the problem characteristics to enhance efficiency and adaptability. In a similar fashion, the BLEA-Krig method proposed in [166] uses Kriging models with the Ψ-mapping to approximate the lower-level response, providing a flexible alternative to quadratic approximations for bilevel optimization problems.

An extensive analysis of these surrogate-assisted strategies, where different nested variants have been systematically tested was given by [167]. This included DE-DE (DE at both levels), DE-SA (DE at the upper level and Surrogate-Assisted at the lower level), SA-SA (Surrogate-Assisted at both levels), and SA-DE (Surrogate-Assisted at the upper level and DE at the lower level). They noted that the choice of variant often depends on the relative computational cost at each level, demonstrating how combinations of surrogate models and evolutionary algorithms can be adjusted to deal with different bilevel optimization problems. Later, in [168], a performance analysis was conducted comparing similar variants

against a surrogate-free (1+1)-CMA-ES baseline, with results indicating that replacing the lower-level objective with a surrogate yields the most significant performance gains across benchmark problems. A list with all of the surrogate-assisted bilevel optimization algorithms we discussed can be found in Table 2.1.

Table 2.1: Examples of Surrogate-Assisted Bilevel Optimization Algorithms.

Algorithm	**Surrogate Model**	**Evolutionary Algorithm**	**Approximation Type**
SBSO [158]	Kriging	NSGA-II, KTLBO	Surrogates at upper level
BL-SAEA [159]	Kriging	CMA-ES	Surrogates at upper level, Lower-level Ψ mapping
SABLA [163]	RSM, Kriging	DE	Lower-level Ψ-mapping
SA-BlDE [160]	k-NN	DE	Lower-level Ψ-mapping
OSFAKM [161]	Kriging	GA	Lower-level Ψ-mapping, applied to thin-walled pressure vessel design
BLEAQ [164]	QA	GA	Lower-level Ψ-mapping
BLEAQ2 [165]	QA	GA	Adaptive Ψ- and ϕ- mappings
BLEA-Krig [166]	Kriging	GA	Ψ- and ϕ-mappings
SABO [162]	RBF	CMA-ES	Surrogates at upper level, Lower-level Ψ-mapping
Singh et al. [167]	RSM, Kriging	DE	Systematic Study: Variants for both levels
Moradi et al. [168]	GP, RBF	CMA-ES	Systematic Study: Variants for both levels

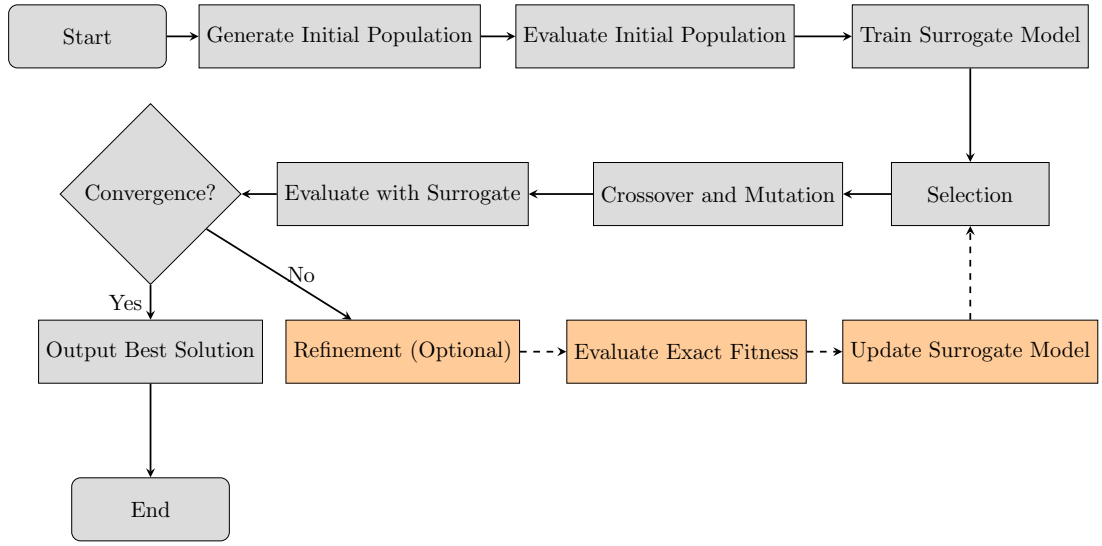


Figure 2.14: General flow of a SAEA.

After presenting the various categories and subcategories of BLEAs, one notable commonality across all these approaches is that they either make no specific assumption or focus on the optimistic formulation of BOPs. To date, there appears to be no research specifically applying EAs to the pessimistic formulation of single-objective BOPs. The majority of studies that have been done so far have focused on related types of problems, such as multiobjective bilevel problems [169] semivectorial bilevel problems [170], and others [171].

2.6 Summary

In this chapter, we introduced the bilevel optimization problem, covering its definitions, mathematical formulation, and example applications. We discussed challenges related to ill-posedness due to lower-level non-unique solutions, and discussed both optimistic and pessimistic formulations. We also reviewed both optimistic and pessimistic classical solution methods and EAs specifically designed for bilevel optimization. While research on EAs has largely focused on optimistic solutions, relatively few approaches address pessimistic formulations, motivating the contributions in the next chapter.

The following chapter proposes a δ-perturbation method to handle both optimistic and pessimistic bilevel problems, establishing theoretical proofs and error bounds. We then integrate this method within EAs and validate it through computational testing on benchmark problems.

Chapter 3

δ-Perturbation Method for Optimistic and Pessimistic Bilevel Optimization

Building on the foundational concepts and solution approaches for bilevel optimization introduced in the previous chapter, this chapter addresses the challenge of ill-posedness due to non-unique lower-level solutions. A perturbed formulation of bilevel optimization problems, referred to as the δ-perturbed formulation, is analyzed, which is particularly useful for cases with multiple lower-level optima. By employing a targeted perturbation strategy for either the optimistic or pessimistic formulation, the lower-level problem yields only a single (approximate) optimal solution for any given upper-level decision. This adjustment enables evolutionary algorithms (EAs) and other iterative methods that solve the lower level repeatedly to efficiently reach the desired bilevel optimal solution. The δ-perturbed formulation is derived by adding the upper-level objective to the lower-level objective, scaled by a small positive or negative δ. It is established that this formulation approximates the original problem, providing a proof and error bound for the approximation, and demonstrating the application of the δ-perturbation scheme within EAs, specifically with a nested Differential Evolution (DE) algorithm. This chapter is based on the work published in [139].

Connection to Thesis Objectives

This chapter addresses O2.1, O3.1, O3.2., O4.1, O4.2

3.1 Problem Definition

In bilevel optimization, decision-making happens in a hierarchical setup where the upper-level decision-maker (the leader) must account for how the lower-level decision-maker (the follower) will react to their choices [172], [173]. This framework is common in real-world scenarios across fields like economics, engineering, and transportation [174]–[176], where one party governs a process and another adjusts in response. However, solving bilevel optimization problems is challenging, particularly when the lower level has multiple optimal solutions for a given upper-level decision. In this case, the leader's problem involves imperfect information due to this multiplicity of solutions, as they cannot fully predict

The content of this chapter have been published in **M. Antoniou**, A. Sinha, and G. Papa, "δ-perturbation of bilevel optimization problems: An error bound analysis," *Operations Research Perspectives*, p. 100 315, 2024

the follower's response. To resolve this, two standard approaches are commonly used: the optimistic approach, which assumes the follower will choose the solution most favorable to the leader, and the pessimistic approach, which assumes the worst-case scenario for the leader [126], [177].

Imagine a city's traffic management system where the City Traffic Manager (Leader) wants to minimize overall traffic congestion by adjusting traffic light timings and managing road usage, while Drivers (Followers) aim to minimize their own travel time by choosing the fastest available routes. For any given signal timing, drivers may have several routes that take the same amount of time, giving them multiple equally good options. In the optimistic approach, the Traffic Manager assumes that drivers will evenly distribute themselves across these equally good routes, which would help balance the traffic and reduce congestion. For example, the manager might adjust the green light timings on a busy road, expecting that drivers will choose alternative routes, easing the overall congestion. This assumption is based on the best-case scenario, where everything goes as planned, and drivers cooperate to reduce traffic. On the other hand, in the pessimistic approach, the Traffic Manager assumes the worst-case scenario: that drivers will concentrate on the same routes, leading to more congestion on certain roads. To account for this, the Traffic Manager optimizes under the worst-case assumption, meaning they plan and make decisions to minimize congestion even if drivers do not spread out as expected. This could involve adding extra lanes, implementing adaptive traffic signals, or providing real-time updates to drivers to prevent overload on certain routes. The key difference here is that the optimistic approach assumes drivers will cooperate and help reduce congestion, while the pessimistic approach plans for the worst-case situation, ensuring that the system remains efficient no matter how drivers behave.

Both the optimistic and pessimistic approaches make bilevel optimization computationally expensive, with the pessimistic approach in fact resulting in a trilevel problem. The pessimistic formulation requires evaluating the leader's and follower's strategies, as well as considering the worst-case scenario, which significantly increases the problem's complexity. Efficiently solving these problems, particularly in non-convex settings, requires advanced strategies to reduce the complexity and make the problem tractable for evolutionary and population-based algorithms. More about these issues the reader can find in the previous Chapter 2.

This chapter is organized as follows: Section 3.2 reviews the related work. In Section 3.3, we define the bilevel optimization problem, discuss its mathematical properties, and introduce the notation used throughout this work. Section 3.4 analyzes the proposed δ-perturbation method for ill-posed bilevel problems from both optimistic and pessimistic perspectives, discussing error bounds and providing illustrative examples. In Section 3.5, we describe the nested Differential Evolution (DE) approach in which the method was employed. Section 3.6 demonstrates how this nested Evolutionary Algorithm (EA) effectively solves both optimistic and pessimistic formulations using the δ-perturbation method, with computational validation of the error bounds. Finally, Section 3.7 concludes the chapter.

3.2 Related Work

As discussed in Chapter 2, bilevel problems are known to be NP-hard, even in their simplest formulations [1]. Classical methods employed to tackle these problems include the KKT method [114], [178], gradient descent [179], [180], trust region methods [123], [181], [182], and penalty methods [183]–[185]. For complex bilevel problems that cannot be simplified mathematically, evolutionary algorithms (BLEAs) are often preferred. EAs, such as genetic algorithms [186]–[188], differential evolution [189], [190], Bayesian optimization

[191], and hybrid methods [192], [193], utilize efficient sampling strategies to converge towards a bilevel optimal solution [163], [186]–[189], [192], [194]–[196]. However, many of these EA-based methods can be computationally intensive due to their nested structure, as they require solving lower-level optimization problems multiple times for each upper-level solution. To address this, researchers have proposed methods to simplify the process by approximating the reaction set mapping and lower-level optimal value function during intermediate generations [160], [166], [197].

Moreover, most of existing BLEAs either rely on a single solution from the lower level or adopt an optimistic approach, which assumes cooperative behavior from the follower. Research on the pessimistic formulation is limited because this approach introduces additional complexity, effectively transforming the problem into a trilevel structure as we mentioned above. To our knowledge, little research has specifically addressed the pessimistic formulation for single-objective bilevel problems. Existing studies have focused on related bilevel problems, such as multiobjective bilevel problems [169], semivectorial bilevel problems [170], [198], and other variants [171].

A thorough review of pessimistic bilevel problems is provided by [126], with relevant classical methods discussed in [125] and optimality conditions explored by [177]. Classical techniques, such as the k-th best algorithm [130] and penalty-based methods [127], [128], have been applied to specific kinds of pessimistic bilevel problems. Approximation methods have also been explored in [133], [199]. Perturbation-based approaches, originally proposed by Molodtsov [134] for weak Stackelberg games, have been applied to resource control [135], linear bilevel problems [136], and to find optimistic, pessimistic, and intermediate solutions [137].

This thesis extends these approaches specifically to bilevel problems in both optimistic and pessimistic perspectives and, for the first time [139], applies them within EAs, enabling the approximation of solutions for either optimistic or pessimistic bilevel problems. Furthermore, this thesis presents a formal error bound analysis for the perturbation-based formulation. By transforming lower-level problems with multiple solutions into problems with unique approximate solutions, the perturbation-based method makes the optimization process tractable for both optimistic and pessimistic bilevel problems.

3.3 Mathematical Definition

For the sake of clarity, we revisit some of the definitions regarding the bilevel optimization problem provided in Section 2.4. In a bilevel problem, each level operates with its own distinct objectives, constraints, and variables. There are two categories of decision variables (vectors): upper-level decision vectors and lower-level decision vectors. The lower-level optimization is essentially a parametric optimization problem, solved with respect to the lower-level decision vectors while considering the upper-level decision vectors as parameters. Importantly, the lower-level optimization problem acts as a constraint for the upper-level optimization problem. For the upper-level objective function $F : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ and the lower-level objective function $f : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$, the bilevel problem is given by:

$$\underset{x \in X, y \in Y}{\text{“min”}} \quad F(x, y) \tag{3.1}$$

$$\text{subject to} \tag{3.2}$$

$$y \in \underset{y}{\operatorname{argmin}} \{ f(x, y) : g_i(x, y) \leq 0, i = 1, \ldots, I \} \tag{3.3}$$

$$G_j(x, y) \leq 0, \quad j = 1, \ldots, J \tag{3.4}$$

where $g_i : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ denotes the lower level constraints, and $G_j : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ denotes the upper level constraints. For brevity,from now on, we let g and G respectively denote the

total number of lower-level and upper-level constraints (although there could be multiple constraints in each category). Equality constraints may also exist but have been omitted for simplicity. Note that $x \in X$ and $y \in Y$ in the above definition denote that variables can be real or integers, depending on X being $\mathbb{R}^n$ or $\mathbb{Z}^n$ and Y being $\mathbb{R}^m$ or $\mathbb{Z}^m$. However, in most of our discussions, we will avoid using the sets X and Y, and we will consider the variables to be real and continuous unless explicitly stated to be integers. When the feasible set of the lower level does not depend on the upper level decision variables, i.e., $g(x, y)$ in (3.3) is reduced to $g(y)$, the bilevel problem is referred to as an *independent* bilevel problem [199].

Figure 3.1 illustrates a general bilevel problem involving nested optimization tasks. The figure illustrates that for any given upper level decision vector, there exists a corresponding lower level optimization problem to be addressed. This lower level problem, which is parametric in nature, yields the rational and optimal response of the lower level participant in response to any decision by the upper level participant. In this context, the upper level decision vector is denoted as x, while the lower level decision vector is denoted as y. Pairs $(x^{(1)}, y^{(1)})$ and $(x^{(2)}, y^{(2)})$, where $y^{(1)}$ and $y^{(2)}$ represent optimal responses to $x^{(1)}$ and $x^{(2)}$, respectively, are considered feasible solutions to the bilevel optimization problem, provided that they also adhere to the other constraints of the problem. In bilevel problems, the leader typically has knowledge of the follower's optimization problem , while the follower observes the leader's decisions and subsequently optimizes its own strategies accordingly.

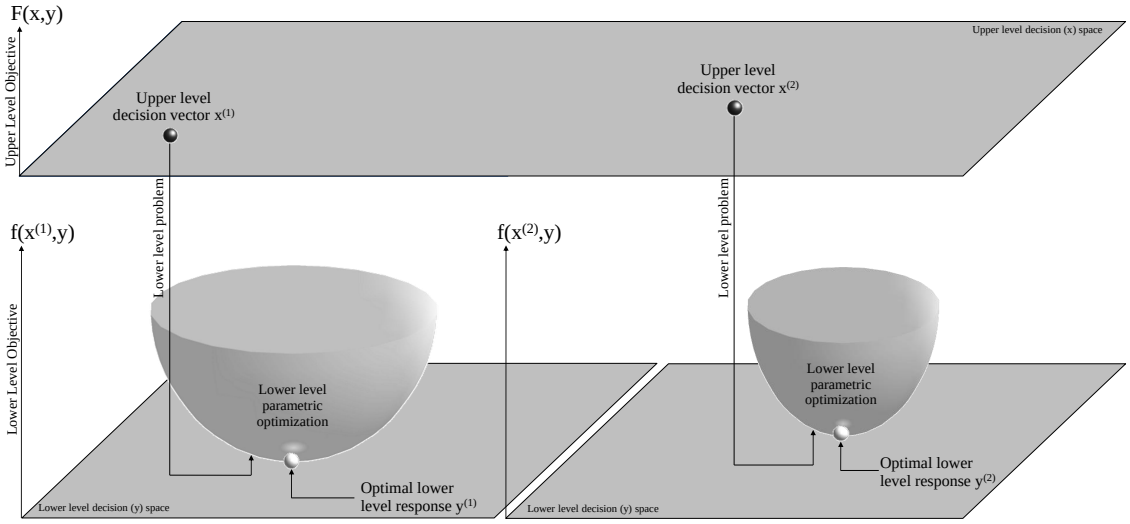


Figure 3.1: A sketch of a bilevel problem showing the inter-linkage between the upper and lower levels.

An equivalent formulation of the above problem can be stated in terms of set-valued mappings, where $\Psi : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$ is a set-valued mapping.

$$\Psi(x) = \underset{y}{\operatorname{argmin}}\{f(x, y) : g(x, y) \leq 0\}$$

The bilevel optimization problem, in terms of $\Psi(x)$, can be expressed as a constrained

optimization problem as follows:

$$\begin{aligned} &\text{“}\min_{x,y}\text{”} \quad F(x,y) \\ &\text{subject to} \\ &\quad y \in \Psi(x) \\ &\quad G(x,y) \leq 0 \end{aligned}$$

A summary of the terminologies commonly used in the context of bilevel optimization is given in Table 3.1.

In the preceding two definitions, quotation marks are employed when describing the upper level minimization problem due to an inherent ambiguity that rises when multiple lower level optimal solutions exist for a given upper level decision vector. When confronted with several lower level optimal solutions, it becomes unclear at the upper level which specific optimal solution from the lower level should be employed. To address this ambiguity, different positions that the leader can adopt can be defined. Two frequently explored positions are the optimistic and pessimistic positions, which we discuss next.

Table 3.1: Important terminology and notations in bilevel optimization.

Name	Symbol	Description
Lower level reaction set	$\Psi : \mathbb{R}^n \rightrightarrows \mathbb{R}^m$	$\Psi(x) = \{y : y \in \underset{y}{\operatorname{argmin}}\{f(x,y) : g(x,y) \leq 0\}\}$, represents the lower level optimal solution(s) for an upper level decision vector
Lower level feasible region (admissible set)	$S(x)$	$S(x) = \{y : g(x,y) \leq 0\}$ represent the lower level feasible region corresponding to a given upper level decision vector.
Inducible region	$I = \operatorname{gph}\Psi$	$I = \{(x,y) : (x,y) \in \Phi, y \in \Psi(x)\}$, represents the set of upper level decision vector and corresponding lower level optimal solution(s)
Choice function	$\psi : \mathbb{R}^n \to \mathbb{R}^m$	$\psi(x)$ represents the solution chosen by the follower for any upper level decision vector. It becomes important in case of multiple lower level optimal solutions.
Lower level optimal value function	$\varphi : \mathbb{R}^n \to \mathbb{R}$	$\varphi(x) = \min_y\{f(x,y) : g(x,y) \leq 0\}$ represents the minimum lower level function value corresponding to a given upper level decision vector.
Upper level pessimistic function	$k : \mathbb{R}^n \to \mathbb{R}$	$k(x) = \max_y\{F(x,y) : g(x,y) \leq 0, f(x,y) \geq \varphi(x)\}$ represents the maximum Upper level function value corresponding to lower level decisions for a given upper level decision vector.

3.3.1 Optimistic and Pessimistic Approches

In an optimistic position, when faced with multiple lower level optimal solutions, the leader anticipates that the follower will select the solution within the optimal set $\Psi(x)$ that yields the best objective function value at the upper level. The follower's choice function in this context (best-case for leader) can be defined as follows:

$$\psi_{\mathrm{o}}(x) = \underset{y}{\operatorname{argmin}}\{F(x,y) : y \in \Psi(x)\}$$

This formulation assumes a degree of cooperation between both participants. The bilevel optimization problem, operating under an optimistic position, is outlined below:

$$\min_{x,y} \quad F(x,y) \tag{3.5}$$

subject to

$$y = \psi_{\mathrm{o}}(x) \tag{3.6}$$

$$G(x,y) \leq 0 \tag{3.7}$$

The optimistic position is more tractable as compared to the pessimistic position; therefore, most of the studies handle the optimistic version of the bilevel optimization problem. An alternative way to write the optimistic position is through the use of the lower level optimal value function φ mapping, defined as:

$$\varphi(x) = \min_{y}\{f(x,y) : g(x,y) \leq 0\}$$

Lower level optimal value function mapping gives the optimal lower level function value corresponding to any given upper level decision vector. In terms of φ-mapping, the bilevel problem can be written as follows:

$$\min_{x,y} \quad F(x,y) \tag{3.8}$$

subject to

$$f(x,y) \leq \varphi(x) \tag{3.9}$$

$$g(x,y) \leq 0 \tag{3.10}$$

$$G(x,y) \leq 0 \tag{3.11}$$

In a pessimistic position, when faced with multiple optimal solutions at the lower level, the leader seeks to optimize for the worst-case scenario. In other words, she assumes that the follower might choose the solution from the optimal set that results in the least favorable objective function value at the upper level. We can express the follower's choice function in this context (worst-case for leader) as follows:

$$\psi_{\mathrm{p}}(x) = \operatorname*{argmax}_{y}\{F(x,y) : y \in \Psi(x)\}$$

This formulation does not assume any form of cooperation. The bilevel optimization problem under a pessimistic position has been defined below:

$$\min_{x,y} \quad F(x,y) \tag{3.12}$$

subject to

$$y = \psi_{\mathrm{p}}(x) \tag{3.13}$$

$$G(x,y) \leq 0 \tag{3.14}$$

The pessimistic position [177], [200], [201] is relatively less tractable when compared to the optimistic position. The pessimistic formulation is guaranteed to have optimal solutions under stronger assumptions compared to the optimistic formulation. In the case of the optimistic position, we provide a formulation based on φ mapping, which is a popular formulation and is commonly used in algorithms to solve the optimistic formulation. We argue that it is possible to write an alternative formulation of the pessimistic position in

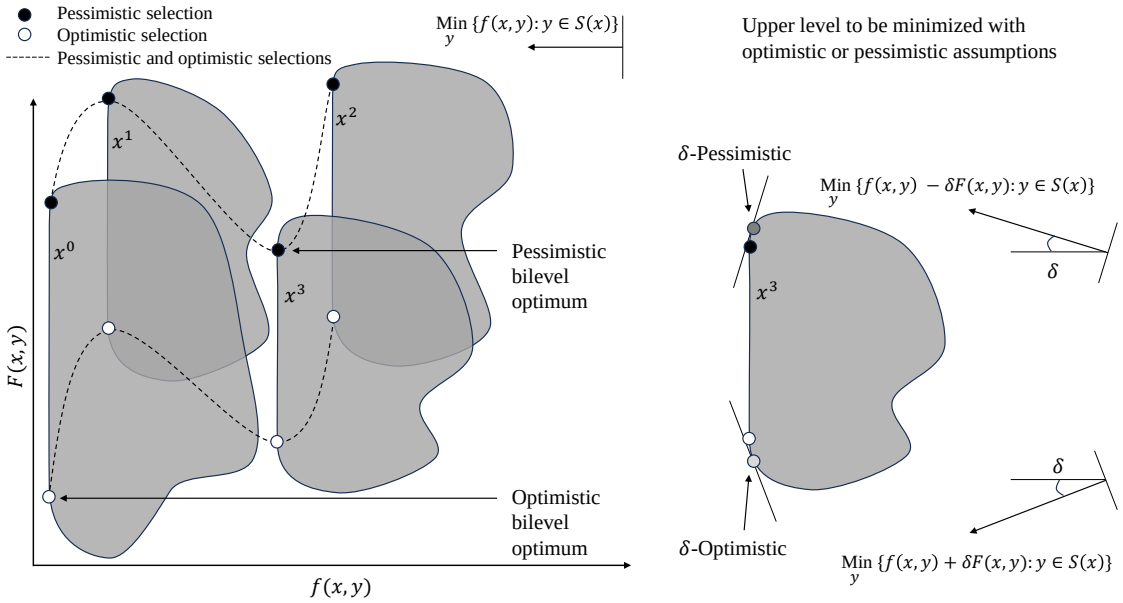


Figure 3.2: Each shaded region in the left figure shows the feasible region of the lower level optimization problem in the F-f space for a given x_0, x_1, x_2 and x_3. The left vertical part of each of the shaded regions represent multiple lower level optimal solutions. The right figure gives a graphical representation for the errors induced by δ-perturbed lower level objective function corresponding to x_3.

terms of the φ-mapping. However, it requires an additional constraint to ensure that the upper level chooses the worst solution from the lower level's optimal set for any given x.

$$\min_{x,y} \quad F(x,y) \tag{3.15}$$

subject to

$$f(x,y) \leq \varphi(x) \tag{3.16}$$

$$F(x,y) \geq k(x) \tag{3.17}$$

$$g(x,y) \leq 0 \tag{3.18}$$

$$G(x,y) \leq 0 \tag{3.19}$$

where $k(x) = \max_y \{F(x,y) : g(x,y) \leq 0, f(x,y) \leq \varphi(x)\}$.

3.4 Mathematical Method: δ-Perturbation for Bilevel Problems and Error Bound Analysis

Consider the bilevel problem (3.1)-(3.4) for which one may take an optimistic or a pessimistic position while solving it. It is proposed that the δ-perturbed bilevel formulation

for the optimistic position is given as follows:

$$\begin{aligned} &\min_{x,y} \quad F(x,y) \\ &\text{subject to} \\ &\quad y \in \operatorname*{argmin}_{y}\{f(x,y) + \delta F(x,y) : g(x,y) \leq 0\} \\ &\quad G(x,y) \leq 0 \end{aligned}$$

Similarly, the δ-perturbed bilevel formulation for the pessimistic position is given as follows:

$$\begin{aligned} &\min_{x,y} \quad F(x,y) \\ &\text{subject to} \\ &\quad y \in \operatorname*{argmin}_{y}\{f(x,y) - \delta F(x,y) : g(x,y) \leq 0\} \\ &\quad G(x,y) \leq 0 \end{aligned}$$

In both the above formulations, δ is a small positive constant that leads to a minor perturbation in the optimistic and the pessimistic bilevel problems. We explain the perturbations visually through Figure 3.2. This figure shows a bilevel optimization problem with multiple lower level optimal solutions on the left hand side. The feasible region of the lower level is shown in gray for four decision vectors of the upper level, x_0, x_1, x_2 and x_3. Note that there are multiple lower level optimal solutions for all of these four upper level decision vectors. Among the multiple lower level optimal solutions, the optimistic and pessimistic selections for x_0, x_1, x_2 and x_3 are marked in white and black circles, respectively. A dotted line marks the paths traced by the optimistic and pessimistic lower level selections for all upper level decision vectors, making it clear that x_0 represents the optimistic bilevel optimum (as x_0 corresponds to minimum of $F(x,y)$ for optimistic selection), while x_3 represents the pessimistic bilevel optimum (as x_3 corresponds to minimum of $F(x,y)$ for pessimistic selection).

The perturbation of the lower level problem with δ slightly changes the optimistic and pessimistic selections that have been shown on the right-hand side of Figure 3.2. Solving this perturbed problem, therefore, leads to an approximate optimistic and pessimistic bilevel optimum. In this work, we show the relationship between the choice of δ and the maximum error that it leads to when the δ-perturbed formulation is solved instead of the true bilevel (optimistic or pessimistic) formulation.

A summary of the terminology used for δ-perturbation is provided in Table 3.2. In the following subsections, we explore the δ-perturbed formulation for the pessimistic position first followed by the δ-perturbed formulation for the optimistic position. The reason for this is that the results obtained for the pessimistic case, can be easily extended to arrive at the results for the optimistic case.

3.4.1 δ-Perturbation of the Pessimistic Bilevel Problem

We consider the δ-perturbation of the pessimistic bilevel problem in (3.15)-(3.19). We make the assumption that all functions in the bilevel problem are continuous and the relaxed feasible set $\Phi = \{(x,y) : G(x,y) \leq 0, g(x,y) \leq 0\}$ is closed and bounded. We also assume that there exists a feasible and bounded lower level optimal solution for any given upper level decision vector. Let M be the maximum value of $F(x,y)$ and m be the minimum value of $F(x,y)$ in the set Φ. Note that (3.15)-(3.19) contains

$$k(x) = \max_{y}\{F(x,y) : G(x,y) \leq 0, f(x,y) \leq \varphi(x)\} \text{ and } \varphi(x) = \min_{y}\{f(x,y) : g(x,y) \leq 0\}$$

Table 3.2: Terminology and notation in δ- perturbation for bilevel optimization.

Name	Symbol	Description
Maximum Value of upper level function	$M = \max\{F(x,y) \in \Phi\}$	Represents the maximum value of the upper level objective function $F(x,y)$ over the feasible set Φ.
Minimum Value of upper level function	$m = \min\{F(x,y) \in \Phi\}$	Represents the minimum value of the upper level objective function $F(x,y)$ over the feasible set Φ.
Perturbation value δ	$\delta \geq 0$	δ is a small non-negative perturbation number.
Dual variable λ	λ for $f(x,y) \leq \varphi(x,y)$	Represents the dual variable for the constraint $f(x,y) \leq \varphi(x,y)$.
Optimistic lower-level solution	y^o	y^o represents the specific follower decision that leads to the optimistic upper-level solution for a given leader decision.
Perturbed Optimistic lower-level solution	y_o^δ	y_o^δ represents the perturbed specific follower decision that leads to an approximate optimistic upper-level solution for a given leader decision.
Pessimistic lower-level solution	y^p	y^p represents the specific follower decision that leads to the pessimistic upper-level solution for a given leader decision.
Perturbed Pessimistic lower-level solution	y_p^δ	y_p^δ represents the perturbed specific follower decision that leads to an approximate pessimistic upper-level solution for a given leader decision.
Optimistic Choice Function	$\psi^o : \mathbb{R}^n \to \mathbb{R}^m$	$\psi^o(x)$ represents the function that maps a leader decision x to the follower decision that leads to the optimistic upper-level solution.
Perturbed Optimistic Choice Function	$\psi_o^\delta : \mathbb{R}^n \to \mathbb{R}^m$	$\psi_o^\delta(x)$ represents the perturbed function that maps a leader decision x to the follower decision leading to an approximate optimistic upper-level solution.
Pessimistic Choice Function	$\psi^p : \mathbb{R}^n \to \mathbb{R}^m$	$\psi^p(x)$ represents the function that maps a leader decision x to the follower decision that leads to the pessimistic upper-level solution.
Perturbed Pessimistic Choice Function	$\psi_p^\delta : \mathbb{R}^n \to \mathbb{R}^m$	$\psi_p^\delta(x)$ represents the perturbed function that maps a leader decision x to the follower decision leading to an approximate pessimistic upper-level solution.

that are unknown *a priori*. From the definition of $k(x)$ and $\varphi(x)$, for any feasible pessimistic bilevel solution (x,y) the following holds:

$$f(x,y) \leq \varphi(x) \tag{3.20}$$

$$F(x,y) \geq k(x) \tag{3.21}$$

Introducing $\delta > 0$, from (3.20) and (3.21) we can write the following:

$$f(x,y) - \delta F(x,y) \leq \varphi(x) - \delta k(x) \tag{3.22}$$

Now let us consider δ-perturbation of the lower level problem in (3.1)-(3.4):

$$\begin{aligned} \min_{y} \quad & f(x,y) - \delta F(x,y) \\ \text{subject to} \quad & \\ & g(x,y) \leq 0 \end{aligned}$$

and we denote the optimal solution of the above perturbed problem as:

$$y^p \in \underset{y}{\operatorname{argmin}}\{f(x,y) - \delta F(x,y) : g(x,y) \leq 0\}$$

Since $f(x,y^p) - \delta F(x,y^p)$ is the optimal function value of the above problem for any given x, we can write the following for any y that is lower level feasible for the given x:

$$f(x,y^p) - \delta F(x,y^p) \leq f(x,y) - \delta F(x,y) \tag{3.23}$$

From (3.22) and (3.23), the following holds:

$$f(x,y^p) - \delta F(x,y^p) \leq \varphi(x) - \delta k(x) \tag{3.24}$$

From the definition of $\varphi(x)$, it is obvious that:

$$\varphi(x) \leq f(x,y^p) \tag{3.25}$$

Therefore, from (3.24) and (3.25), we can write:

$$k(x) \leq F(x,y^p) \tag{3.26}$$

From (3.24) and (3.25), we can also write the following:

$$\varphi(x) \leq f(x,y^p) \leq \varphi(x) + \delta[F(x,y^p) - k(x)] \tag{3.27}$$

$$\varphi(x) \leq f(x,y^p) \leq \varphi(x) + \delta[M - m] \tag{3.28}$$

Note that as $\delta \to 0$, $f(x,y^p) \to \varphi(x)$. Therefore, the error in $\varphi(x)$ is bounded by $\delta[M-m]$ for all x.

Let us write the following problems for the ease of discussion:

$$\begin{aligned} F(x,y^p) = \max_{y} \quad & F(x,y^p) \\ \text{subject to} \quad & \\ & g(x,y^p) \leq 0 \\ & f(x,y) \leq f(x,y^p) \end{aligned}$$

$$\begin{aligned} k(x) = \max_{y} \quad & F(x,y) \\ \text{subject to} \quad & \\ & g(x,y) \leq 0 \\ & f(x,y) \leq \varphi(x,y) \end{aligned}$$

$$\begin{aligned} l(x) = \max_{y} \quad & F(x,y) \\ \text{subject to} \quad & \\ & g(x,y) \leq 0 \\ & f(x,y) \leq \varphi(x) + \delta(M-m) \end{aligned}$$

From (3.28), we can write $k(x) \leq F(x, y^p) \leq l(x)$.

From the assumptions made at the beginning of the section, $k(x)$ and $l(x)$ will always be feasible and bounded. Note that $l(x)$ can be obtained by perturbing the constraint $f(x, y) \leq \varphi(x, y)$ in $k(x)$ by $\delta(M - m)$. If λ is the dual[1] for constraint $f(x, y) \leq \varphi(x, y)$ and the right-hand side of the constraint is perturbed by $\delta(M - m)$, then for $\delta \to 0$, we can write $l(x) - k(x) = \delta(M - m)\lambda$ with first-order approximation.

Since $F(x, y)$ is assumed to be continuous and bounded in Φ, $k(x)$ and $l(x)$ are feasible and bounded, it implies that λ will always be finite. Therefore, we can state the following:

$$\begin{aligned} & l(x) = k(x) + \delta(M - m)\lambda \\ \Rightarrow\ & k(x) \leq F(x, y^p) \leq k(x) + \delta(M - m)\lambda \\ \Rightarrow\ & \text{as } \delta \to 0\ , F(x, y^p) \to k(x) \end{aligned}$$

We have shown that as $\delta \to 0$,

$$\begin{aligned} & f(x, y^p) \to \phi(x) \quad \text{with error } \delta[M - m] \\ & F(x, y^p) \to k(x) \quad \text{with error } \delta\lambda[M - m] \end{aligned} \tag{3.29}$$

This implies that the errors in the lower and upper objective level function values on solving a δ-perturbed pessimistic bilevel problem are bounded by $\delta[M-m]$ and $\delta\lambda[M-m]$, respectively.

3.4.2 δ-Perturbation of the Optimistic Bilevel Problem

The δ-perturbation of the optimistic bilevel problem may also be necessary at times to ensure that whenever the lower level optimization problem is solved, it returns an (approximate) optimistic selection for the algorithm to work efficiently. The error bounds for the δ-perturbed optimistic bilevel problem can be obtained in a similar manner as in the pessimistic case. Note that the formulation based on the lower level optimal value function (3.8)-(3.11) for the optimistic bilevel problem does not involve $k(x)$. However, for the sake of finding the error bounds in a similar manner as that for the δ-perturbed pessimistic formulation, we will introduce a slightly modified $k(x)$ and write the lower level optimal value function-based formulation as follows:

$$\min_{x,y} \quad F(x, y) \tag{3.30}$$

subject to

$$f(x, y) \leq \varphi(x) \tag{3.31}$$

$$F(x, y) \leq k(x) \tag{3.32}$$

$$g(x, y) \leq 0 \tag{3.33}$$

$$G(x, y) \leq 0 \tag{3.34}$$

where $\varphi(x) = \min_y\{f(x, y) : g(x, y) \leq 0\}$ and $k(x) = \min_y\{F(x, y) : g(x, y) \leq 0, f(x, y) \leq \varphi(x)\}$. Obviously, (3.32) is a redundant constraint, and once again, from the definition of $k(x)$ and $\varphi(x)$, the following holds:

$$f(x, y) \leq \varphi(x) \tag{3.35}$$

$$F(x, y) \leq k(x) \tag{3.36}$$

[1] A dual is an alternative formulation of an optimization problem (known as the primal) that provides bounds on the optimal value of the original problem. Often, the dual is derived by forming a Lagrangian—assigning multipliers to the primal constraints—to obtain dual variables. However, duality can also be established through other frameworks, such as Fenchel or conic duality.

Introducing $\delta > 0$, from (3.35) and (3.36) we can write the following:

$$f(x,y) + \delta F(x,y) \leq \varphi(x) + \delta k(x) \tag{3.37}$$

The δ-perturbation of the lower level problem in (3.1)-(3.4) in the optimistic case is given as follows:

$$\begin{aligned} \min_{y} \quad & f(x,y) + \delta F(x,y) \\ \text{subject to} \quad & \\ & g(x,y) \leq 0 \end{aligned}$$

with the optimal solution of the above perturbed problem denoted as:

$$y^o \in \underset{y}{\operatorname{argmin}}\{f(x,y) + \delta F(x,y) : g(x,y) \leq 0\}$$

Since $f(x,y^o) + \delta F(x,y^o)$ is the optimal function value of the above problem for any given x, we can write the following for any y that is lower level feasible for the given x:

$$f(x,y^o) + \delta F(x,y^o) \leq f(x,y) + \delta F(x,y) \tag{3.38}$$

Building up along the same lines as in the pessimistic case, one can write the following:

$$f(x,y^o) + \delta F(x,y^o) \leq \varphi(x) + \delta k(x) \tag{3.39}$$
$$\varphi(x) \leq f(x,y^o) \tag{3.40}$$
$$F(x,y^o) \leq k(x) \tag{3.41}$$

From (3.39) and (3.40),:

$$\varphi(x) \leq f(x,y^o) \leq \varphi(x) + \delta[k(x) - F(x,y^o)] \tag{3.42}$$
$$\varphi(x) \leq f(x,y^o) \leq \varphi(x) + \delta[M - m] \tag{3.43}$$

Note that as $\delta \to 0$, $f(x,y^o) \to \varphi(x)$. Therefore, the error in $\varphi(x)$ is bounded by $\delta[M-m]$ for all x.

Now let us write the following minimization problems, noting that in the pessimistic case, similar problems were defined with maximization of the objective function:

$$\begin{aligned} F(x,y^o) = \min_{y} \quad & F(x,y^o) \\ \text{subject to} \quad & \\ & g(x,y^o) \leq 0 \\ & f(x,y) \leq f(x,y^o) \end{aligned}$$

$$\begin{aligned} k(x) = \min_{y} \quad & F(x,y) \\ \text{subject to} \quad & \\ & g(x,y) \leq 0 \\ & f(x,y) \leq \varphi(x,y) \end{aligned}$$

$$\begin{aligned} l(x) = \min_{y} \quad & F(x,y) \\ \text{subject to} \quad & \\ & g(x,y) \leq 0 \\ & f(x,y) \leq \varphi(x) + \delta(M-m) \end{aligned}$$

From (3.43), we can write $l(x) \leq F(x, y^o) \leq k(x)$. Next, following similar arguments as in the pessimistic case, the error bounds for the lower and upper objective functions again turn out to be $\delta[M - m]$ and $\delta\lambda[M - m]$, respectively, with λ being the dual variable with a finite value for the constraint $f(x, y) \leq \varphi(x, y)$.

3.4.3 Illustrative Example

Let us revisit the example from Section 2.4.4. Without any perturbation, the follower's optimal response varies depending on the leader's choice of x. When $x = 0$, the follower's objective $f(y) = -xy = 0$ becomes independent of y, resulting in a range of optimal responses where any $y \in [0, 1]$ minimizes $f(y)$. This range introduces ambiguity for the leader, as they cannot take into account a single outcome for y.

In Figure 3.3, we observe the effect of introducing a perturbation $\pm\delta F$, which removes this illposedness. With the added term, the follower's response at $x = 0$ shifts to a unique solution. Specifically, when $f(y) - \delta F$ is used, the follower has $y = 0$ as the unique minimum. When $f(y) + \delta F$ is used, the follower's optimal response becomes $y = 1$. Thus, the perturbation transforms the follower's response from a range of possible values to a single, fixed outcome, thereby giving an optimistic (green dot) or pessimistic (red dot) solution to the leader.

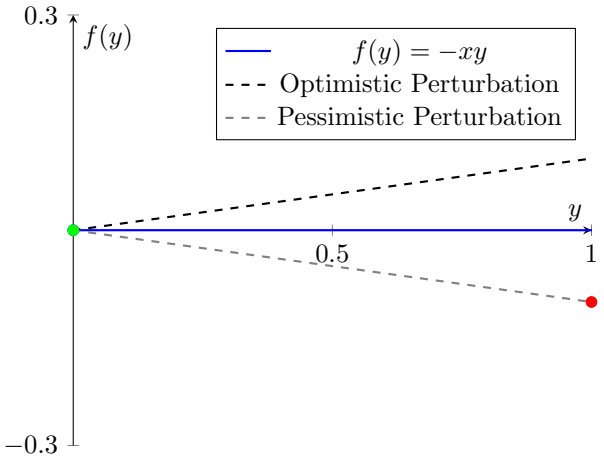


Figure 3.3: Effect of perturbation on the follower's optimal response, for x=0 for the example in Section 2.4.4.

We present another example, taken from [202], to illustrate a more general case with a multimodal landscape at the lower level, where the optimistic and pessimistic solutions correspond to different upper-level decisions (x). This contrasts with the previous example, where both solutions were associated with the same upper-level decision. The example is as follows:

$$\begin{aligned} &\min_{x,y} \quad x^2 - y \\ &\text{subject to} \\ &\qquad y \in \operatorname*{argmin}_y \{((y - 1 - x/10)^2 - x/2 - 1/2)^2 : 0 \leq y \leq 3\} \\ &\qquad 0 \leq x \leq 1 \end{aligned} \tag{3.44}$$

For a fixed upper level decision x, the set of optimal lower level decisions is given by:

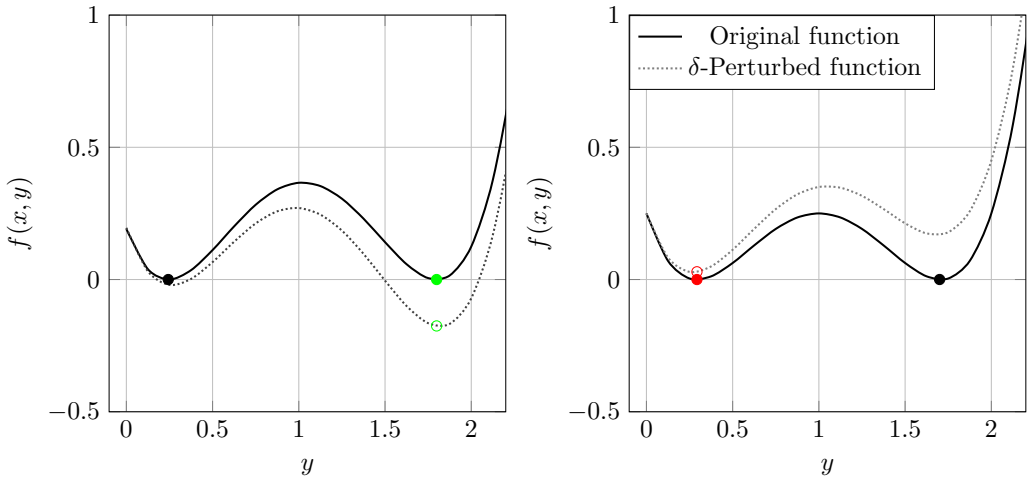
$$\Psi(x) = \{1 + x/10 \pm \sqrt{2 + 2x}/2\} \tag{3.45}$$

In Table 3.3, the known optima for the problem are shown under three different scenarios: the global optimistic solution, the pessimistic solution for the optimistic x, and the global pessimistic solution. We can see that when the leader optimizes under the optimistic assumption, the expected outcome is F = -1.755 . However, when faced with the pessimistic reality, the outcome turns out to be F = -0.1987 . Had the leader instead optimized under the assumption of the worst-case scenario, the outcome would have been F = -0.2929, ensuring a better pessimistic result. In Figure 3.4a, we show the lower-level objective

Table 3.3: Optimistic and pessimistic solutions for the example problem [202].

Solutions	x	y	F
Optimistic	0.2106	1.799	-1.755
Optimistic-Pessimistic	0.2106	0.243	-0.1987
Pessimistic	0	0.2929	-0.2929

function for $x = 0.2106$, which has two lower-level optimal points shown with filled circles. A leader solving this problem from an optimistic perspective will prefer the green-filled circle based on the upper-level objective function. When using the δ-perturbation, the lower level objective function is slightly modified so that an approximate optimistic solution (green circle) is selected automatically by solving the lower-level problem. In Figure 3.4b, the same is shown from a pessimistic perspective, where the δ-perturbation allows the selection of an approximate pessimistic solution (red circle) automatically. In Figure 3.5, we show the actual choice functions $\psi_o(x)$ and $\psi_p(x)$ for the optimistic and pessimistic cases, respectively. When the optimistic and the pessimistic problems are δ-perturbed, the actual functions change to $\psi_o^\delta(x)$ and $\psi_p^\delta(x)$. The difference between the actual choice function and the approximate choice function represents the error induced due to the δ-perturbed formulation.



(a) Lower level f for $x = 0.2106$ (b) Lower level f for $x = 0.0$

Figure 3.4: δ-perturbation plots for the example problem showing true lower level optimal solutions and the δ-perturbed lower level solutions for optimistic and pessimistic cases for $\delta = 0.1$.

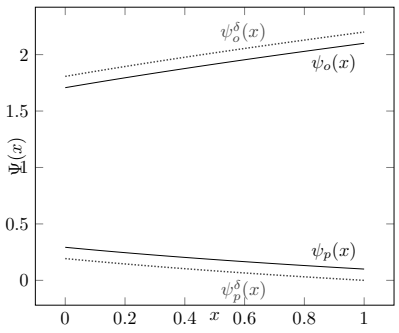


Figure 3.5: $\Psi(x)$-x plot for the original and δ-perturbed example problem.

3.5 Algorithmic Method

In this section, we outline the general steps of Differential Evolution (DE). Next, we describe the nested DE (NDE) approach, focusing in particular on the perturbation-based NDE method, which is later used in our experiments.

3.5.1 Differential Evolution (DE)

DE [203] is a population-based metaheuristic search algorithm and falls under the category of evolutionary algorithm methods. Following the standard schema of such methods, it is based on an evolutionary process, where a population of candidate solutions goes through mutation, crossover, and selection operations. The main steps of the algorithm can be seen below. While DE can employ different strategies, the approach described here follows the commonly used `rand/1/bin` strategy, where mutation is performed using a randomly selected base vector, one difference vector, and binomial crossover.

1. Initialization: A population of $NPop$ individuals is randomly initialized. Each individual is represented by a D-dimensional parameter vector, $X_{i,g} = (x^1_{i,g}, x^2_{i,g}, \ldots, x^D_{i,g})$, where $i = 1, 2, \ldots, NPop$ and $g = 1, 2, \ldots,$ MaxGen, where MaxGen is the maximum number of generations. Each vector component is subject to upper and lower bounds $X_{\min}$ and $X_{\max}$. The initial values of the ith individual are generated as:

$$X_i = X_{\min} + \text{rand}(0,1) \times (X_{\max} - X_{\min}) \tag{3.46}$$

   where rand$(0, 1)$ is a random number uniformly distributed between 0 and 1.

2. Mutation: The new individual is generated by adding the weighted difference vector between two randomly selected population members to a third member. This process is expressed as:

$$V_{i,G} = X_{\text{r1},G} + F_{\text{mut}} \times (X_{\text{r2},G} - X_{\text{r3},G}) \tag{3.47}$$

   where $V_{i,G}$ is the mutant vector, X is an individual, r1, r2, and r3 are randomly chosen distinct integers within the range $[1, NPop]$ such that r1 $\neq$ r2 $\neq$ r3 $\neq i$, G corresponds to the current generation, and F_{mut} is the scale factor, usually a positive real number between 0.2 and 0.8, controlling the rate at which the population evolves.

3. Crossover: After mutation, the binomial crossover operation is applied. The mutant individual $V_{i,G}$ is recombined with the parent vector $X_{i,G}$ to generate the offspring $U_{i,G}$. The vector elements of the offspring are inherited from $X_{i,G}$ or $V_{i,G}$ depending on a parameter called crossover probability, $C_r \in [0, 1]$, as follows:

$$U_{i,t,G} = \begin{cases} V_{i,t,G}, & \text{if rand} \leq C_r \text{ or } t = \text{random}(i), \\ X_{i,t,G}, & \text{otherwise.} \end{cases} \tag{3.48}$$

   where $rand \in (0, 1)$ is a uniformly generated number, $random(i) \in \{1, \ldots, D\}$ is a randomly chosen index that assures $V_{i,t,G}$ contributes at least one gene to $U_{i,G}$, and $t = 1, \ldots, D$ denotes the t-th gene of the individual's vector.

4. Evaluation: The fitness (or objective value) of the offspring $U_{i,G}$ is evaluated using the objective function $f(x)$. This step is essential to assess how well the offspring performs relative to the current population.

5. Selection: The selection operation is a competition between each individual $X_{i,G}$ and its offspring $U_{i,G}$ and defines which individual will prevail in the next generation.

The winner is the one with the best fitness value. The operation is expressed by the following equation:

$$X_{i,G+1} = \begin{cases} U_{i,G}, & \text{if } f(U_{i,G}) \leq f(X_{i,G}) \\ X_{i,G}, & \text{otherwise.} \end{cases} . \tag{3.49}$$

The above steps of mutation, crossover, and selection are repeated for each generation until a certain set of termination criteria has been met. Figure 3.6 shows the basic flowchart of the DE. The pseudocode for the DE is shown in Algorithm 3.1.

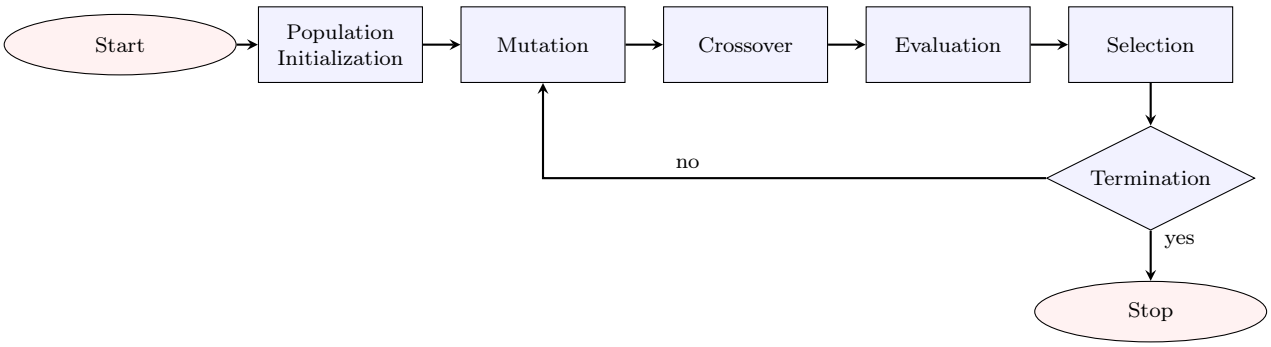


Figure 3.6: Basic flowchart of the differential evolution algorithm.

Algorithm 3.1: Differential Evolution

Input: Population size $NPop$, mutation factor F_{mut}, crossover probability C_r, maximum generations $G_{\max}$, bounds $[X_{\min}, X_{\max}]$, problem definition
Output: Best solution found X_{best}
Initialize population $P(0)$ within bounds $[X_{\min}, X_{\max}]$
Evaluate each individual X_i by computing fitness $f(X_i)$
Set X_{best} as the individual with the best fitness
$g \leftarrow 0$
while (stopping criteria not met) **do**
  for each individual X_i in the population **do**
    Select random individuals X_{r1}, X_{r2}, X_{r3} from $P(g)$, distinct from X_i
    Perform mutation: $V_i \leftarrow X_{\text{r1}} + F_{\text{mut}} \cdot (X_{\text{r2}} - X_{\text{r3}})$
    Perform crossover to generate trial vector U_i
    Evaluate fitness $f(U_i)$
    if fitness $f(U_i)$ is better than $f(X_i)$ **then**
      $X_i \leftarrow U_i$
    end if
  end for
  Update X_{best} if a better individual is found
  $g \leftarrow g + 1$
end while
return X_{best}

3.5.2 Bilevel Nested Differential Evolution

A nested approach requires solving a lower-level optimization problem for each given upper-level vector. We sample the upper-level vectors using an upper-level DE algorithm and use a lower-level DE to solve the lower-level optimization problem corresponding to each upper-level vector. This nested DE approach for bilevel optimization has been previously introduced [8].To ensure that the problem constraints are respected, we implement a constraint handling mechanism based on the following criteria [204]: any feasible solution is preferred over any infeasible solution; among two feasible solutions, the one with a better objective function value is preferred; among two infeasible solutions, the one with the smaller overall constraint violation is preferred. The pseudocode for the upper-level DE is provided in Algorithm 3.2, while the pseudocode for the lower-level DE can be found in Algorithm 3.3. The steps are as follows:

1. Upper-Level Initialization: The algorithm begins by initializing the upper-level population $P_U(0)$, where each candidate solution X_i is generated within specified bounds. This population serves as the starting point for the optimization process.

2. Upper-Level Evolution: The upper-level population undergoes evolution until a stopping criterion is met, such as reaching the maximum number of generations. During each generation, mutation and crossover operators are applied to each upper-level vector X_i to produce trial vectors U_i. For each U_i, the algorithm proceeds to the lower-level optimization to evaluate the corresponding lower-level solution.

3. Lower-Level Optimization: For each upper-level solution X, the algorithm performs a lower-level DE optimization to solve the lower-level problem. This involves evolving a population of lower-level candidate solutions Y_i using the DE process. Fitness evaluation includes both the lower-level objective function $f(X, Y)$ and an upper-level perturbation $\delta \cdot F(X, Y)$. The best lower-level solution Y_{best} is then identified and returned to the upper level.

4. Upper-Level Fitness Evaluation: Once the lower-level optimization provides the best lower-level solution Y_{best}, the fitness of the upper-level trial vector U_i is evaluated. The fitness is compared to that of the current solution X_i, and constraint handling is applied if necessary. If the trial vector U_i has better fitness or better satisfies the constraints, it replaces the current solution X_i.

5. Update and Iteration: After evaluating all trial vectors, the algorithm updates the best solution found in the upper-level population. The process of evolution continues with the next generation, applying mutation and crossover to further explore the search space, until the stopping criterion is satisfied.

6. Final Solution: Once the stopping criterion for the upper-level optimization is met, the algorithm terminates. The best solution X_{best} from the upper-level population and its corresponding lower-level solution Y_{best} are returned as the final results of the nested optimization process.

Algorithm 3.2: Upper-Level DE

Input: Population size $NPop_U$, mutation factor F_{mut}, crossover probability C_r, maximum generations $G_{\max}$, bounds $[X_{\min}, X_{\max}]$, problem definition
Output: Best solution found X_{best}
Initialize upper-level population $P_U(0)$ within bounds $[X_{\min}, X_{\max}]$
$g_U \leftarrow 0$
while (stopping criteria not met for the upper-level) **do**
  for each upper-level individual X_i in the population **do**
    Select random individuals X_{r1}, X_{r2}, X_{r3} from $P_U(g_U)$, distinct from X_i
    Perform upper-level mutation: $V_i \leftarrow X_{\text{r1}} + F_{\text{mut}} \cdot (X_{\text{r2}} - X_{\text{r3}})$
    Perform upper-level crossover to generate trial vector U_i
    Call Lower-Level Optimization for each U_i:
    $Y_{\text{best}} \leftarrow$ **Lower-Level DE**(U_i)
    Compute upper-level fitness $F(U_i)$
    Compute upper-level fitness $F(X_i)$
    Constraint Handling:
    if U_i is feasible and X_i is infeasible **then**
      $X_i \leftarrow U_i$
    else if U_i and X_i are both feasible **then**
      if $F(U_i) < F(X_i)$ **then**
        $X_i \leftarrow U_i$
      end if
    else if U_i and X_i are both infeasible **then**
      if Constraint violation of $U_i <$ Constraint violation of X_i **then**
        $X_i \leftarrow U_i$
      end if
    end if
  end for
  Update X_{best} if a better upper-level individual is found
  $g_U \leftarrow g_U + 1$
end while
return X_{best}

Algorithm 3.3: Lower-Level DE

Input: Upper-level vector X, population size $NPop_L$, mutation factor F_{mut}, crossover probability C_r, maximum generations $G_{\max}$, bounds $[Y_{\min}, Y_{\max}]$, problem definition, perturbation factor $\pm\delta$
Output: Best solution found Y_{best}
Initialize lower-level population $P_L(0)$ within bounds $[Y_{\min}, Y_{\max}]$ for the given upper-level vector X
Evaluate each individual Y_i by computing fitness $f(X, Y_i) - \delta \cdot F(X, Y_i)$
Set Y_{best} as the individual with the best fitness
$g_L \leftarrow 0$
while (stopping criteria not met for lower-level) **do**
  for each lower-level individual Y_i in the population **do**
    Select random individuals Y_{r1}, Y_{r2}, Y_{r3} from $P_L(g_L)$, distinct from Y_i
    Perform lower-level mutation: $V_i \leftarrow Y_{\text{r1}} + F_{\text{mut}} \cdot (Y_{\text{r2}} - Y_{\text{r3}})$
    Perform lower-level crossover to generate trial vector U_i
    Evaluate fitness $f(X, U_i) - \delta \cdot F(X, U_i)$
    Constraint Handling:
    if U_i is feasible and Y_i is infeasible **then**
      $Y_i \leftarrow U_i$
    else if U_i and Y_i are both feasible **then**
      if fitness of $f(X, U_i) - \delta \cdot F(X, U_i)$ is better than $f(X, Y_i) - \delta \cdot F(X, Y_i)$ **then**
        $Y_i \leftarrow U_i$
      end if
    else if U_i and Y_i are both infeasible **then**
      if Constraint violation of U_i is less than that of Y_i **then**
        $Y_i \leftarrow U_i$
      end if
    end if
  end for
  Update Y_{best} if a better lower-level individual is found
  $g_L \leftarrow g_L + 1$
end while
return Y_{best}

3.6 Numerical Experiments

In this section, we provide an experimental study on various bilevel test problems from the literature to computationally demonstrate the workings of the δ-perturbation approach when a nested algorithm is used to solve the test problems. We describe the experimental setup and the results.

3.6.1 Experimental Setup

The values of the parameters are stated in Table 3.4 and are selected to be close to values from the existing literature [8]. For every test problem, we use the nested DE to solve the δ-perturbed formulations of optimistic and pessimistic variants separately. The only difference in the two variants is the choice of δ, which is expected to be a small positive number for the optimistic position and a small negative number for the pessimistic position.

Table 3.4: Parameters used in DE.

	Upper Level	Lower Level
Crossover rate	0.9	0.9
Mutation rate	0.8	0.8
Population size	50	50
Number of generations	30	50

We test the approach on six test problems, the definitions of which are provided in Appendix B. Five of these problems are taken from the *mb*-suite [202], while the remaining problem, a principal-agent problem, is taken from [205]. For all test problems, we performed 20 independent runs on an Intel(R) Core(TM) i5-8365U CPU @ 1.60GHz with 16 GB of RAM on the Windows 10 operating system. The algorithm is implemented in Matlab R2023b without any parallelization.

To evaluate the performance of our method, we use the absolute error (AE) as a metric in the upper-level objective function and the lower-level objective function. If the optimal upper and lower-level objective function values for the optimistic bilevel problem are F_o^* and f_o^*, and the bilevel optimal solution obtained by solving the δ-perturbed formulation are F_o^δ and f_o^δ, then the absolute errors at the upper and lower levels are given as follows:

$$\text{AE}_o^F = |F_o^\delta - F_o^*|$$
$$\text{AE}_o^f = |f_o^\delta - f_o^*|$$

Similarly, for the pessimistic case, the absolute errors are as follows:

$$\text{AE}_p^F = |F_p^\delta - F_p^*|$$
$$\text{AE}_p^f = |f_p^\delta - f_p^*|$$

For values below 10^{-6}, the table indicates them as 10^{-6}, as precision below this value is usually impractical.

3.6.2 Results for bilevel test problems

First, we report the results of the bilevel problems taken from [202]. In the following Tables 3.5-3.9 we report the statistical results from 20 independent runs of the algorithm

for different values of δ, encompassing both the optimistic and pessimistic variants. We present the minimum, maximum, median, mean, and standard deviation of the accuracy of the objective functions.

In Figures 3.7, 3.9, 3.11, 3.13, and 3.15, we present the boxplots of the upper level and lower level AE obtained by the algorithm for different δ while solving the δ-perturbed formulations of the optimistic and pessimistic variants for each test problem. It is obvious that in almost all cases the higher the δ, the AE is expected to be higher. However, this may not always generalize computationally as can be seen in Fig.3.9b for Problem 2, where for $\delta = 0.5$, the AE is lower compared to $\delta = 0.05, 0.1$. This anomaly likely occurs in this test function due to its strongly multimodal landscape and tight coupling between x and y, where the perturbation temporarily shifts the solution into a different basin of attraction, a behavior not observed in simpler or less sensitive functions. Furthermore, in Figure 3.11b and Problem 3, the cases of $\delta = 0.001$ and 0.1 once again seems to be an anomaly, but it should be noted that the AE for both these cases have a magnitude of 10e-6, which is sufficiently low. In addition, we note that in Problem 1 in the optimistic case, the upper-level error is low despite the higher lower-level error for higher δ. This is because the lower-level objective includes a dominant quartic term, making the landscape highly sensitive to perturbation and causing a significant shift in the optimistic solution. However, the linear upper-level objective remains stable, leading to smaller upper-level error. In contrast, in Problem 3 in the opimistic case, the lower-level error is low while the upper-level error is high. This occurs because the lower-level objective has a smoother landscape around the optimum, resulting in smaller shifts under perturbation, while the quadratic upper-level objective amplifies changes in the optimistic solution, leading to higher upper-level error.

In Figures 3.8, 3.10, 3.12, 3.14 and 3.15 we have plotted the lower level absolute error achieved for each δ for all runs, along with the theoretical bounds proposed in the previous section. The AE of the lower level function for different δ is shown to be always below the theoretical error bound (i.e. within the gray area).

Table 3.5: Statistical results of Absolute Errors (AE) for Problem 1.

δ	AE_o^F					AE_o^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	6.56e-06	4.19e-04	1.57e-04	1.76e-04	1.27e-04
0.005	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	8.03e-05	2.70e-03	5.55e-04	1.03e-03	8.23e-04
0.01	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	2.44e-04	4.32e-03	1.51e-03	1.91e-03	1.24e-03
0.05	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	8.79e-05	2.66e-02	1.41e-02	1.31e-02	8.21e-03
0.10	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	2.92e-03	5.10e-02	2.42e-02	2.28e-02	1.32e-02
0.50	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	1.43e-03	2.61e-01	4.88e-02	8.48e-02	7.94e-02
1.00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	7.98e-03	5.00e-01	2.06e-01	2.40e-01	1.70e-01
1.50	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	8.29e-03	4.56e-01	2.30e-01	2.29e-01	1.40e-01
5.00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	2.48e-02	1.11e+00	3.20e-01	4.99e-01	4.14e-01
5.50	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	6.25e-02	1.11e+00	4.30e-01	5.80e-01	4.34e-01
10.00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	0.00e+00	9.71e-03	1.11e+00	4.80e-01	5.30e-01	4.04e-01
δ	AE_p^F					AE_p^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	1.44e-05	6.40e-05	2.44e-05	2.83e-05	1.20e-05	1.00e-06	9.01e-02	1.00e-06	1.17e-02	2.79e-02
0.005	6.39e-05	1.11e-04	8.28e-05	8.43e-05	1.16e-05	1.00e-06	1.79e-02	1.00e-06	2.76e-03	5.59e-03
0.01	1.82e-04	2.37e-04	2.16e-04	2.12e-04	1.72e-05	1.00e-06	2.70e-02	1.00e-06	2.17e-03	6.39e-03
0.05	1.24e-03	1.29e-03	1.27e-03	1.27e-03	1.22e-05	2.93e-05	3.18e-05	3.09e-05	3.09e-05	5.91e-07
0.10	2.56e-03	2.60e-03	2.58e-03	2.58e-03	9.25e-06	1.26e-04	1.29e-04	1.27e-04	1.27e-04	9.13e-07
0.50	1.27e-02	1.27e-02	1.27e-02	1.27e-02	1.02e-05	3.11e-03	4.28e-03	3.13e-03	3.18e-03	2.59e-04
1.00	2.46e-02	2.46e-02	2.46e-02	2.46e-02	1.68e-05	1.20e-02	1.20e-02	1.20e-02	1.20e-02	1.68e-05
1.50	3.58e-02	3.59e-02	3.59e-02	3.59e-02	1.54e-05	2.60e-02	2.61e-02	2.60e-02	2.60e-02	2.31e-05
5.00	1.02e-01	1.02e-01	1.02e-01	1.02e-01	2.93e-05	2.34e-01	2.35e-01	2.35e-01	2.34e-01	1.46e-04
5.50	1.10e-01	1.10e-01	1.10e-01	1.10e-01	3.54e-05	2.76e-01	2.77e-01	2.77e-01	2.77e-01	1.94e-04
10.00	1.73e-01	1.73e-01	1.73e-01	1.73e-01	7.22e-05	7.61e-01	7.63e-01	7.62e-01	7.62e-01	7.21e-04

Table 3.6: Statistical results of Absolute Errors (AE) for Problem 2.

δ	AE_o^F					AE_o^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.0001	2.90e-03	4.64e-02	4.64e-02	1.51e-02	3.94e-02	1.45e-03	2.31e-02	2.31e-02	1.96e-02	7.50e-03
0.001	9.95e-02	9.54e-01	1.00e-01	1.43e-01	1.91e-01	3.09e-05	4.90e-02	4.90e-02	4.65e-02	1.09e-02
0.005	1.71e-01	1.71e-01	1.71e-01	1.71e-01	7.17e-05	8.04e-02	8.05e-02	8.05e-02	8.05e-02	2.96e-05
0.01	2.15e-01	2.15e-01	2.15e-01	2.15e-01	8.61e-05	9.76e-02	9.77e-02	9.77e-02	9.77e-02	3.11e-05
0.05	3.68e-01	3.68e-01	3.68e-01	3.68e-01	5.85e-05	1.34e-01	1.34e-01	1.34e-01	1.34e-01	5.44e-06
0.10	4.64e-01	4.64e-01	4.64e-01	4.64e-01	7.51e-05	1.32e-01	1.32e-01	1.32e-01	1.32e-01	1.10e-05
0.50	7.93e-01	7.94e-01	7.94e-01	7.94e-01	8.39e-05	1.03e-01	1.03e-01	1.03e-01	1.03e-01	1.17e-04
1.00	1.00e+00	1.00e+00	1.00e+00	1.00e+00	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
1.50	1.00e+00	1.00e+00	1.00e+00	1.00e+00	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
5.00	1.00e+00	1.00e+00	1.00e+00	1.00e+00	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
5.50	1.00e+00	1.00e+00	1.00e+00	1.00e+00	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
10.00	1.00e+00	1.00e+00	1.00e+00	1.00e+00	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
δ	AE_p^F					AE_p^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	1.00e-06	5.52e-04	1.00e-06	2.85e-05	1.23e-04	1.00e-06	1.38e-03	1.00e-06	6.98e-05	3.08e-04
0.005	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
0.01	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
0.05	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
0.10	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
0.50	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
1.00e	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
1.50	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
5.00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
5.50	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
10.00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00

Table 3.7: Statistical results of Absolute Errors (AE) for Problem 3.

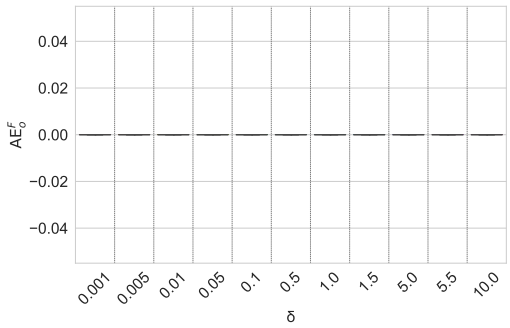
δ	AE_o^F					AE_o^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	9.26e-04	9.96e-04	9.83e-04	9.80e-04	1.93e-05	1.00e-06	4.84e-06	1.00e-06	1.50e-06	9.23e-07
0.005	4.88e-03	4.97e-03	4.96e-03	4.95e-03	2.14e-05	1.00e-06	2.74e-06	1.00e-06	1.09e-06	3.89e-07
0.01	9.86e-03	9.90e-03	9.89e-03	9.89e-03	1.08e-05	1.00e-06	4.77e-06	1.00e-06	1.37e-06	1.15e-06
0.05	4.75e-02	4.75e-02	4.75e-02	4.75e-02	1.29e-05	1.00e-06	3.67e-06	1.00e-06	1.13e-06	5.98e-07
0.10	9.00e-02	9.00e-02	9.00e-02	9.00e-02	7.70e-06	1.00e-06	1.10e-06	1.00e-06	1.01e-06	2.25e-08
0.50	2.50e-01	2.50e-01	2.50e-01	2.50e-01	2.10e-12	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
1.00	2.50e-01	2.50e-01	2.50e-01	2.50e-01	7.50e-15	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
1.50	2.50e-01	2.50e-01	2.50e-01	2.50e-01	1.54e-15	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
5.00	2.50e-01	2.50e-01	2.50e-01	2.50e-01	2.14e-15	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
5.50	2.50e-01	2.50e-01	2.50e-01	2.50e-01	1.41e-15	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
10.00	2.50e-01	2.50e-01	2.50e-01	2.50e-01	4.05e-15	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
δ	AE_p^F					AE_p^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	1.66e-06	7.65e-05	1.87e-05	2.25e-05	1.75e-05	1.00e-06	1.52e-06	1.00e-06	1.03e-06	1.16e-07
0.005	4.03e-03	4.11e-03	4.04e-03	4.04e-03	1.92e-05	1.00e-06	1.29e-06	1.00e-06	1.04e-06	9.22e-08
0.01	9.10e-03	9.14e-03	9.12e-03	9.12e-03	1.09e-05	1.00e-06	1.98e-06	1.00e-06	1.10e-06	2.67e-07
0.05	5.15e-02	5.16e-02	5.15e-02	5.15e-02	1.68e-05	1.00e-06	1.40e-05	1.00e-06	1.70e-06	2.90e-06
0.10	1.09e-01	1.09e-01	1.09e-01	1.09e-01	2.18e-05	1.00e-06	3.78e-06	1.00e-06	1.31e-06	8.05e-07
0.50	7.49e-01	7.49e-01	7.49e-01	7.49e-01	3.07e-09	5.00e-01	5.00e-01	5.00e-01	5.00e-01	4.71e-05
1.00	7.49e-01	7.49e-01	7.49e-01	7.49e-01	3.05e-09	5.00e-01	5.00e-01	5.00e-01	5.00e-01	3.80e-05
1.50	7.49e-01	7.49e-01	7.49e-01	7.49e-01	4.81e-09	5.00e-01	5.00e-01	5.00e-01	5.00e-01	4.38e-05
5.00	7.49e-01	7.49e-01	7.49e-01	7.49e-01	2.21e-09	5.00e-01	5.00e-01	5.00e-01	5.00e-01	3.12e-05
5.50	7.49e-01	7.49e-01	7.49e-01	7.49e-01	6.12e-10	5.00e-01	5.00e-01	5.00e-01	5.00e-01	2.13e-05
10.00	7.49e-01	7.49e-01	7.49e-01	7.49e-01	1.47e-09	5.00e-01	5.00e-01	5.00e-01	5.00e-01	3.15e-05

Table 3.8: Statistical results of Absolute Errors (AE) for Problem 4.

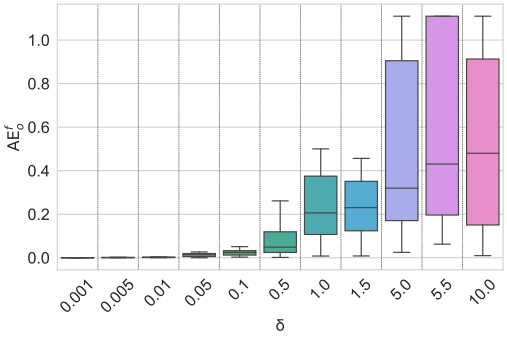
δ	AE_o^F					AE_o^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	2.65e-04	2.65e-04	2.65e-04	2.65e-04	5.64e-08	2.13e-04	2.66e-04	2.43e-04	2.42e-04	1.54e-05
0.005	1.03e-03	1.03e-03	1.03e-03	1.03e-03	8.00e-08	7.00e-04	7.84e-04	7.41e-04	7.41e-04	1.80e-05
0.01	2.00e-03	2.00e-03	2.00e-03	2.00e-03	5.69e-08	1.35e-03	1.44e-03	1.38e-03	1.38e-03	2.20e-05
0.05	1.04e-02	1.04e-02	1.04e-02	1.04e-02	9.56e-08	6.78e-03	6.88e-03	6.83e-03	6.83e-03	2.37e-05
0.10	2.32e-02	2.32e-02	2.32e-02	2.32e-02	1.22e-07	3.03e-03	3.13e-03	3.09e-03	3.09e-03	2.29e-05
0.50	4.01e-01	4.01e-01	4.01e-01	4.01e-01	2.44e-06	1.40e-01	1.40e-01	1.40e-01	1.40e-01	1.22e-06
1.00	6.96e-01	6.96e-01	6.96e-01	6.96e-01	7.52e-06	3.53e-01	3.53e-01	3.53e-01	3.53e-01	7.52e-06
1.50	8.69e-01	8.69e-01	8.69e-01	8.69e-01	1.57e-05	5.67e-01	5.67e-01	5.67e-01	5.67e-01	2.36e-05
5.00	1.24e+00	1.24e+00	1.24e+00	1.24e+00	0.00e+00	1.50e+00	1.50e+00	1.50e+00	1.50e+00	0.00e+00
5.50	1.24e+00	1.24e+00	1.24e+00	1.24e+00	0.00e+00	1.50e+00	1.50e+00	1.50e+00	1.50e+00	0.00e+00
10.00	1.24e+00	1.24e+00	1.24e+00	1.24e+00	0.00e+00	1.50e+00	1.50e+00	1.50e+00	1.50e+00	0.00e+00
δ	AE_p^F					AE_p^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	2.00e-03	2.00e-03	2.00e-03	2.00e-03	2.28e-08	1.00e-06	1.00e-06	1.00e-06	1.00e-06	2.28e-11
0.005	1.00e-02	1.00e-02	1.00e-02	1.00e-02	2.03e-08	2.51e-05	2.51e-05	2.51e-05	2.51e-05	1.01e-10
0.01	2.01e-02	2.01e-02	2.01e-02	2.01e-02	1.66e-08	1.01e-04	1.01e-04	1.01e-04	1.01e-04	1.66e-10
0.05	1.04e-01	1.04e-01	1.04e-01	1.04e-01	7.16e-08	2.62e-03	2.62e-03	2.62e-03	2.62e-03	3.58e-09
0.10	2.14e-01	2.14e-01	2.14e-01	2.14e-01	1.88e-07	1.09e-02	1.09e-02	1.09e-02	1.09e-02	1.88e-08
0.50	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
1.00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
1.50	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
5.00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
5.50	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00
10.00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00	5.00e-01	5.00e-01	5.00e-01	5.00e-01	0.00e+00

Table 3.9: Statistical results of Absolute Errors (AE) for Problem 5.

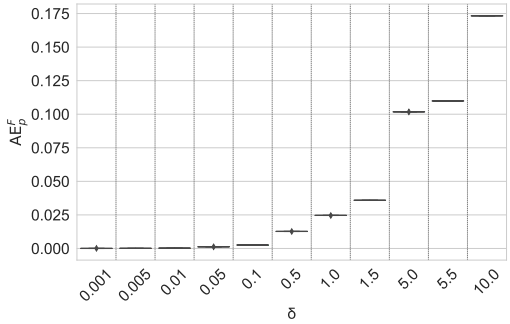
δ	AE_o^F					AE_o^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	7.56e-05	7.57e-05	7.56e-05	7.56e-05	1.76e-08	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
0.005	7.49e-04	7.49e-04	7.49e-04	7.49e-04	7.56e-08	2.57e-06	2.58e-06	2.58e-06	2.58e-06	6.14e-10
0.01	1.78e-03	1.78e-03	1.78e-03	1.78e-03	2.02e-07	1.03e-05	1.03e-05	1.03e-05	1.03e-05	3.10e-09
0.05	9.86e-03	9.86e-03	9.86e-03	9.86e-03	8.35e-07	2.52e-04	2.53e-04	2.52e-04	2.52e-04	9.28e-08
0.10	1.97e-02	1.97e-02	1.97e-02	1.97e-02	1.73e-06	9.87e-04	9.89e-04	9.88e-04	9.88e-04	5.73e-07
0.50	8.84e-02	8.84e-02	8.84e-02	8.84e-02	8.15e-06	2.12e-02	2.13e-02	2.12e-02	2.12e-02	2.58e-05
1.00	1.58e-01	1.58e-01	1.58e-01	1.58e-01	2.23e-05	7.31e-02	7.34e-02	7.32e-02	7.33e-02	7.09e-05
1.50	2.16e-01	2.17e-01	2.16e-01	2.16e-01	2.94e-05	1.46e-01	1.47e-01	1.47e-01	1.47e-01	2.02e-04
5.00	4.92e-01	4.92e-01	4.92e-01	4.92e-01	7.52e-05	1.00e+00	1.00e+00	1.00e+00	1.00e+00	8.95e-04
5.50	5.22e-01	5.22e-01	5.22e-01	5.22e-01	1.05e-04	1.16e+00	1.16e+00	1.16e+00	1.16e+00	1.57e-03
10.00	7.38e-01	7.39e-01	7.38e-01	7.38e-01	2.65e-04	2.81e+00	2.82e+00	2.81e+00	2.81e+00	4.83e-03
δ	AE_p^F					AE_p^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	2.57e-04	2.57e-04	2.57e-04	2.57e-04	1.16e-08	1.00e-06	1.00e-06	1.00e-06	1.00e-06	0.00e+00
0.005	1.25e-03	1.25e-03	1.25e-03	1.25e-03	2.08e-08	3.11e-06	3.11e-06	3.11e-06	3.11e-06	1.04e-10
0.01	2.49e-03	2.49e-03	2.49e-03	2.49e-03	4.68e-08	1.24e-05	1.24e-05	1.24e-05	1.24e-05	4.67e-10
0.05	1.22e-02	1.22e-02	1.22e-02	1.22e-02	1.09e-07	3.02e-04	3.02e-04	3.02e-04	3.02e-04	5.44e-09
0.10	2.38e-02	2.38e-02	2.38e-02	2.38e-02	3.39e-07	1.17e-03	1.17e-03	1.17e-03	1.17e-03	3.39e-08
0.50	1.02e-01	1.02e-01	1.02e-01	1.02e-01	8.44e-07	2.39e-02	2.39e-02	2.39e-02	2.39e-02	4.22e-07
1.00	1.78e-01	1.78e-01	1.78e-01	1.78e-01	1.72e-06	7.99e-02	7.99e-02	7.99e-02	7.99e-02	1.72e-06
1.50	2.40e-01	2.40e-01	2.40e-01	2.40e-01	2.95e-06	1.57e-01	1.57e-01	1.57e-01	1.57e-01	4.42e-06
5.00	2.93e-01	2.93e-01	2.93e-01	2.93e-01	0.00e+00	2.50e-01	2.50e-01	2.50e-01	2.50e-01	0.00e+00
5.50	2.93e-01	2.93e-01	2.93e-01	2.93e-01	0.00e+00	2.50e-01	2.50e-01	2.50e-01	2.50e-01	0.00e+00
10.00	2.93e-01	2.93e-01	2.93e-01	2.93e-01	0.00e+00	2.50e-01	2.50e-01	2.50e-01	2.50e-01	0.00e+00



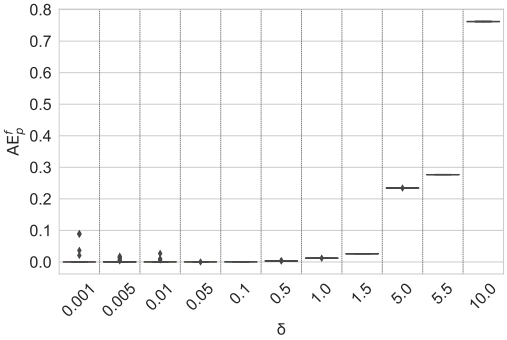
(a) Absolute error (optimistic) for upper level objective.



(b) Absolute error (optimistic) for lower level objective.

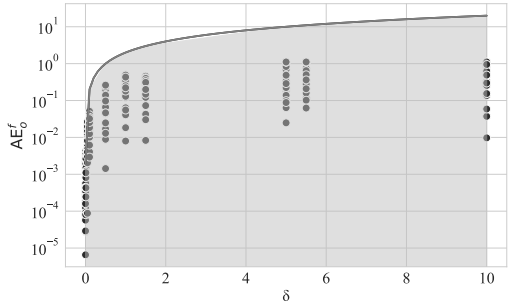


(c) Absolute error (pessimistic) for upper level objective.

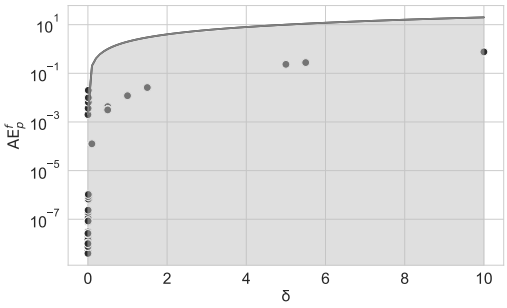


(d) Absolute error (pessimistic) for lower level objective.

Figure 3.7: Boxplots of absolute error (AE) for various values of δ for Problem 1 from 20 runs.

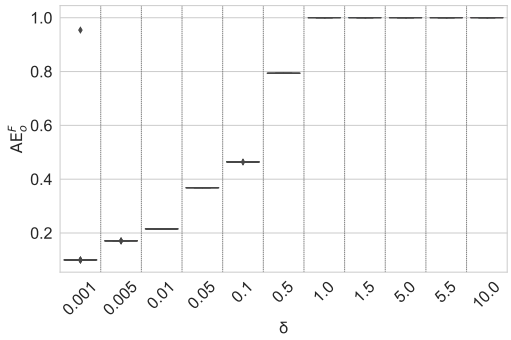


(a) Optimistic case

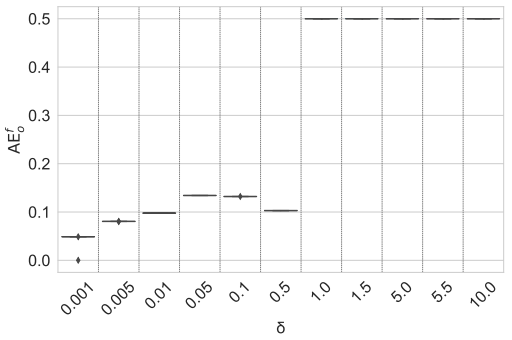


(b) Pessimistic case

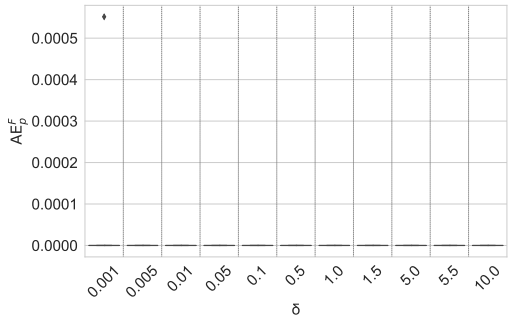
Figure 3.8: Theoretical error bound (shaded) and absolute error for lower level objective from 20 runs of Problem 1 for various values of δ.



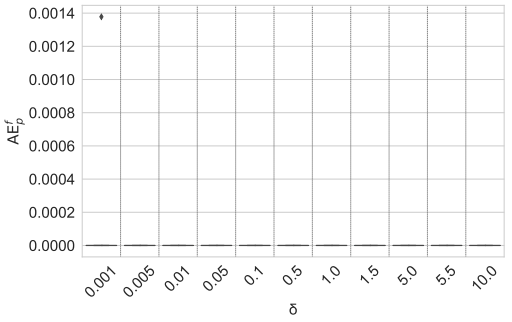
(a) Absolute error (optimistic) for upper level objective.



(b) Absolute error (optimistic) for lower level objective.

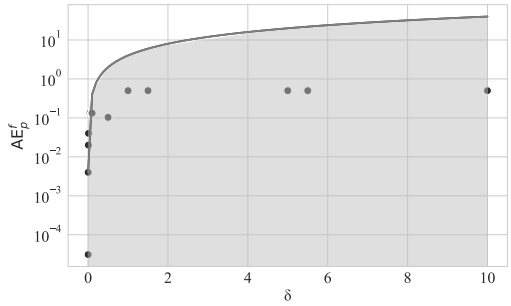


(c) Absolute error (pessimistic) for upper level objective.

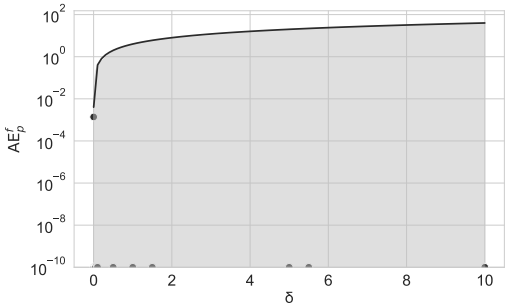


(d) Absolute error (pessimistic) for lower level objective.

Figure 3.9: Boxplots of absolute error (AE) for various values of δ for Problem 2 from 20 runs.

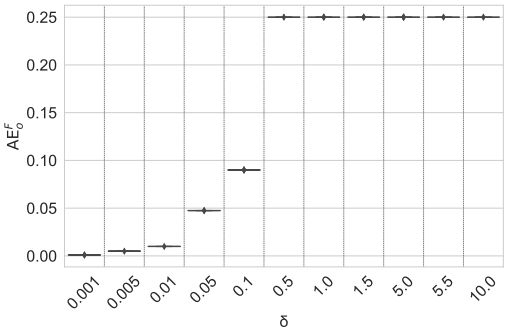


(a) Optimistic case

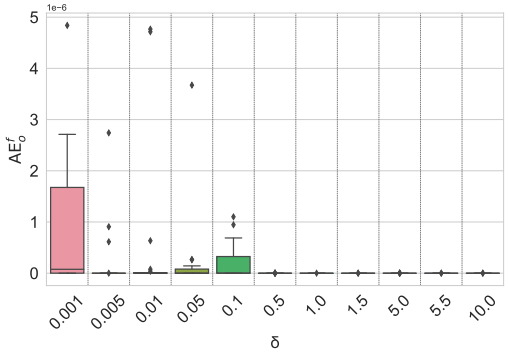


(b) Pessimistic case

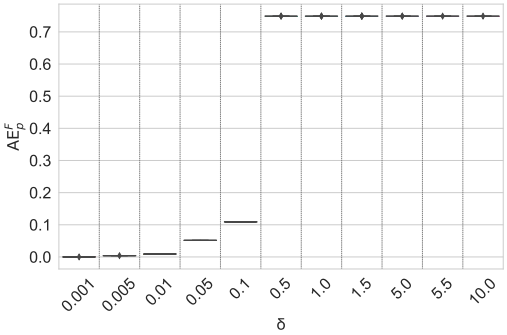
Figure 3.10: Theoretical error bound (shaded) and absolute error for lower level objective from 20 runs of Problem 2 for various values of δ.



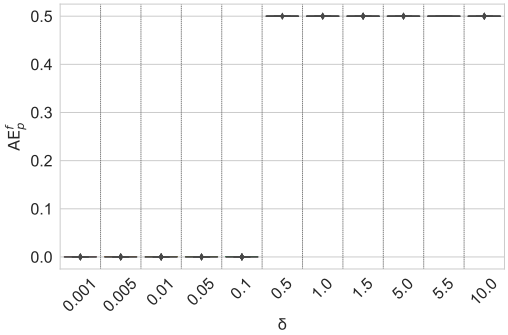
(a) Absolute error (optimistic) for upper level objective.



(b) Absolute error (optimistic) for lower level objective.

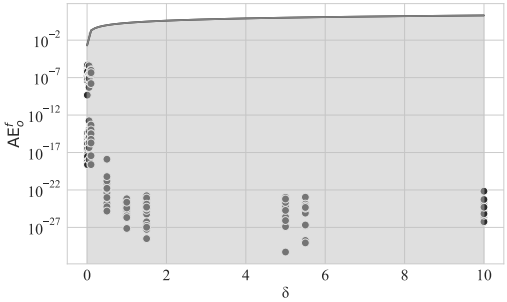


(c) Absolute error (pessimistic) for upper level objective.

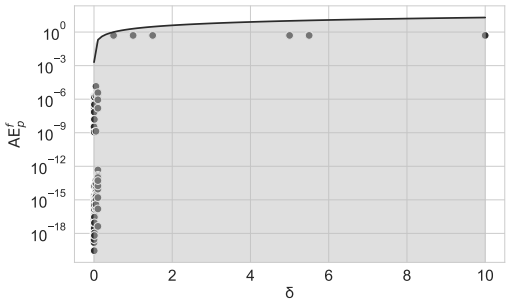


(d) Absolute error (pessimistic) for lower level objective.

Figure 3.11: Boxplots of absolute error (AE) for various values of δ for Problem 3 from 20 runs.

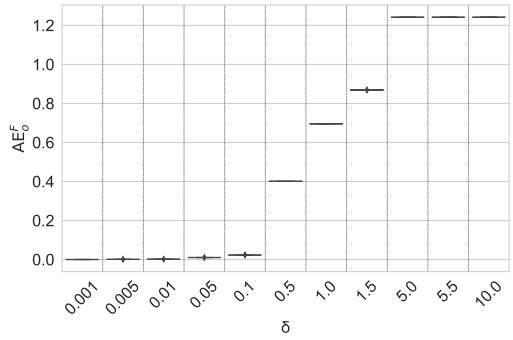


(a) Optimistic case

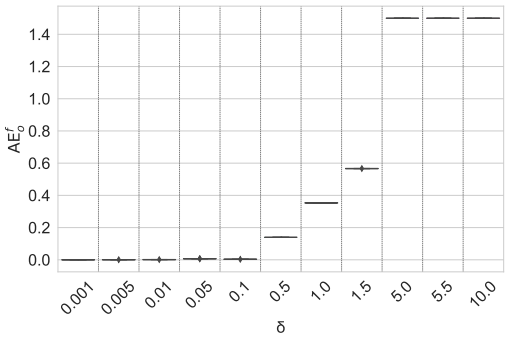


(b) Pessimistic case

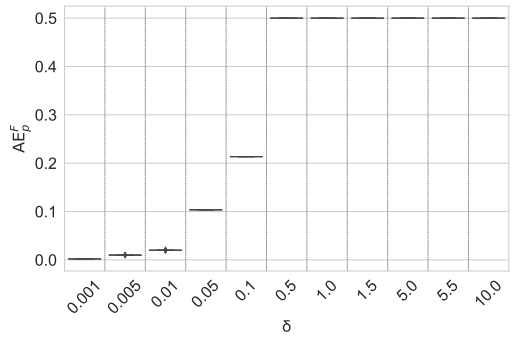
Figure 3.12: Theoretical error bound (shaded) and absolute error for lower level objective from 20 runs of Problem 3 for various values of δ.



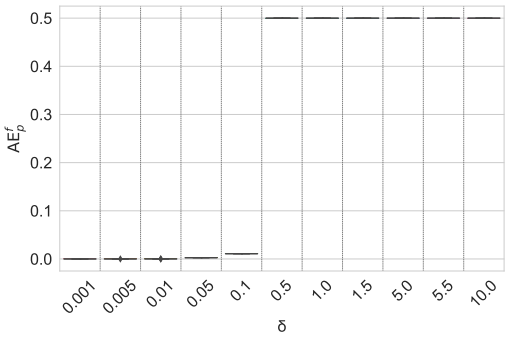
(a) Absolute error (optimistic) for upper level objective.



(b) Absolute error (optimistic) for lower level objective.

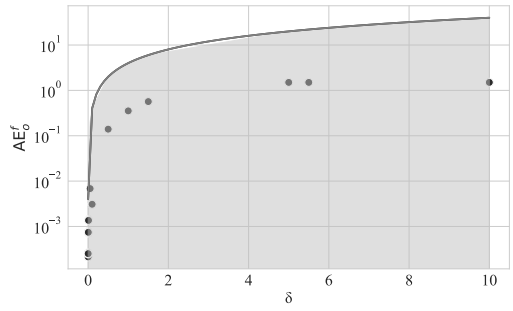


(c) Absolute error (pessimistic) for upper level objective.

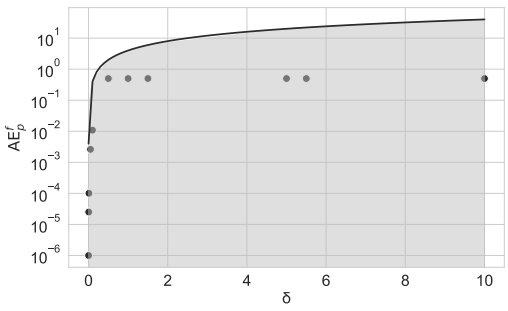


(d) Absolute error (pessimistic) for lower level objective.

Figure 3.13: Boxplots of absolute error (AE) for various values of δ for Problem 4 from 20 runs.

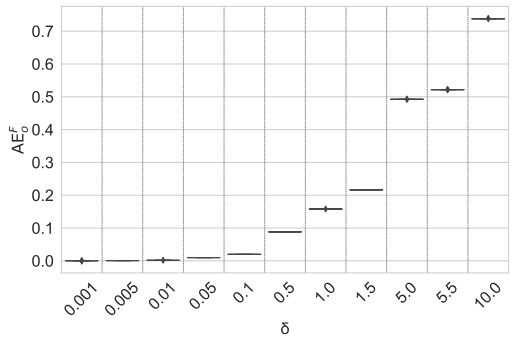


(a) Optimistic case

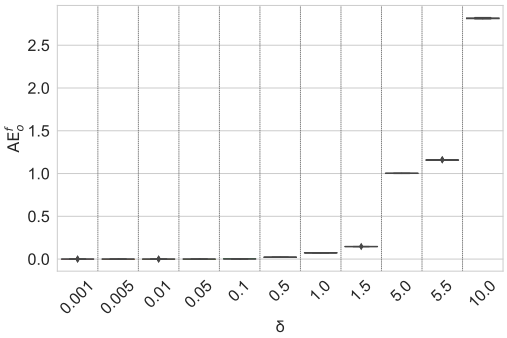


(b) Pessimistic case

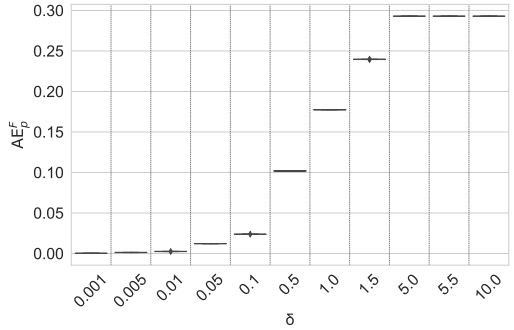
Figure 3.14: Theoretical error bound (shaded) and absolute error for lower level objective from 20 runs of Problem 4 for various values of δ.



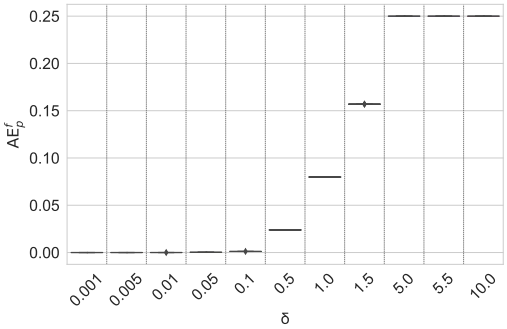
(a) Absolute error (optimistic) for upper level objective.



(b) Absolute error (optimistic) for lower level objective.

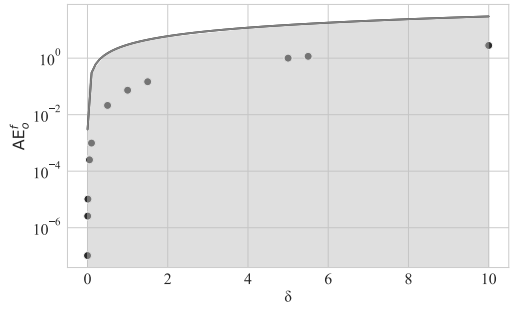


(c) Absolute error (pessimistic) for upper level objective.

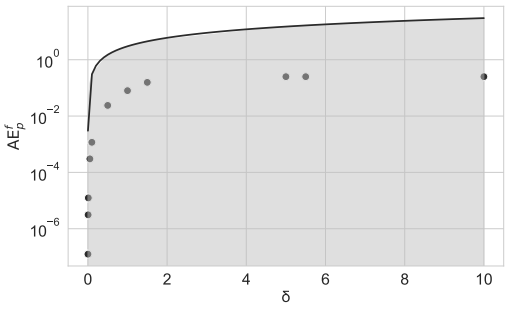


(d) Absolute error (pessimistic) for lower level objective.

Figure 3.15: Boxplots of absolute error (AE) for various values of δ for Problem 5 from 20 runs.



(a) Optimistic case



(b) Pessimistic case

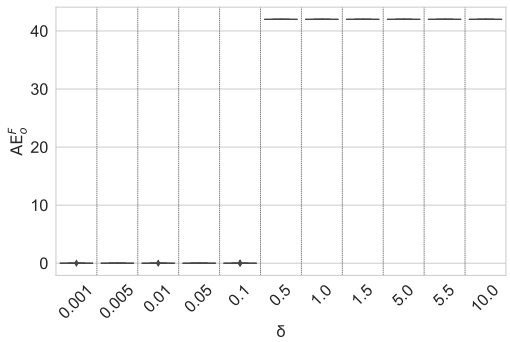
Figure 3.16: Theoretical error bound (shaded) and absolute error for lower level objective from 20 runs of Problem 5 for various values of δ.

3.6.3 Results for a Principal-Agent Problem

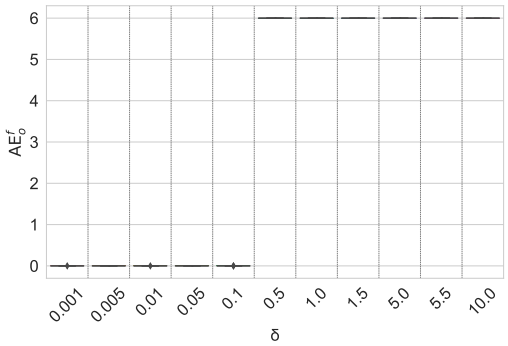
In the domain of game theory, the principal-agent (PA) problem surfaces when a principal delegates a task to an agent in return for compensation. In this scenario, the principal devises a contract, where the decision of the principal is x, while the agent selects an effort or action, denoted as y, to execute the task. The agent's objective is to maximize their utility function, denoted as $f(x, y)$, whereas the principal aims to maximize its utility function, denoted as $F(x, y)$. The definition of the problem considered in the paper can be found in Appendix B. The problem is taken from [205]. In Table 3.10, the statistical results from 20 runs of the algorithm for various values of δ are reported for the optimistic and pessimistic variants. More specifically, we report the minimum, maximum, median, mean, and standard deviation of the accuracy of the objective functions as above. The results are also shown in Figure 3.17. It is noted that the results demonstrate robustness, as the algorithm converges to similar solutions in all 20 runs for various values of δ. For small values of δ we observe the solution obtained from the δ-perturbed formulation is close to the true solution for both optimistic and pessimistic case. However, the algorithm converges to an inaccurate upper-level solution when $\delta \geq 0.5$ for optimistic case, and $\delta \geq 1.5$ for pessimistic case. In Figure 3.18 we have once again plotted the (median) lower-level accuracy achieved for each δ, along with the theoretical limits that we have reported in the previous section. The behavior is found to be the same, such that the median error is well below the theoretical line for all the cases.

Table 3.10: Statistical results of Absolute Errors (AE) for PA.

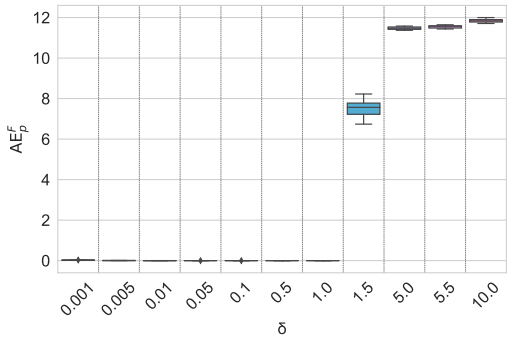
δ	AE_o^F					AE_o^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	1.73e-09	2.84e-07	6.70e-08	8.74e-08	7.57e-08	1.00e-03	1.00e-03	1.00e-03	1.00e-03	0.00e+00
0.005	3.74e-09	1.43e-07	2.62e-08	4.71e-08	4.75e-08	1.00e-03	1.00e-03	1.00e-03	1.00e-03	0.00e+00
0.01	2.50e-09	1.42e-07	2.58e-08	3.17e-08	3.06e-08	1.00e-03	1.00e-03	1.00e-03	1.00e-03	0.00e+00
0.05	9.95e-14	2.36e-11	3.50e-12	7.09e-12	7.12e-12	1.00e-03	1.00e-03	1.00e-03	1.00e-03	0.00e+00
0.10	0.00e+00	5.40e-12	6.96e-13	1.00e-12	1.41e-12	1.00e-03	1.00e-03	1.00e-03	1.00e-03	0.00e+00
0.50	4.20e+01	4.20e+01	4.20e+01	4.20e+01	0.00e+00	6.00e+00	6.00e+00	6.00e+00	6.00e+00	0.00e+00
1.00	4.20e+01	4.20e+01	4.20e+01	4.20e+01	0.00e+00	6.00e+00	6.00e+00	6.00e+00	6.00e+00	0.00e+00
1.50	4.20e+01	4.20e+01	4.20e+01	4.20e+01	0.00e+00	6.00e+00	6.00e+00	6.00e+00	6.00e+00	0.00e+00
5.00	4.20e+01	4.20e+01	4.20e+01	4.20e+01	0.00e+00	6.00e+00	6.00e+00	6.00e+00	6.00e+00	0.00e+00
5.50	4.20e+01	4.20e+01	4.20e+01	4.20e+01	0.00e+00	6.00e+00	6.00e+00	6.00e+00	6.00e+00	0.00e+00
10.00	4.20e+01	4.20e+01	4.20e+01	4.20e+01	0.00e+00	6.00e+00	6.00e+00	6.00e+00	6.00e+00	0.00e+00
δ	AE_p^F					AE_p^f				
	Min	Max	Median	Mean	Std	Min	Max	Median	Mean	Std
0.001	1.77e-02	2.97e-02	2.09e-02	2.19e-02	3.38e-03	2.71e-07	9.83e-06	1.34e-06	2.17e-06	2.85e-06
0.005	4.96e-03	7.11e-03	5.65e-03	5.72e-03	6.78e-04	5.58e-07	1.50e-05	4.72e-06	6.72e-06	5.52e-06
0.01	1.36e-03	2.95e-03	1.97e-03	2.17e-03	5.40e-04	5.76e-07	5.55e-06	2.56e-06	2.74e-06	1.83e-06
0.05	2.99e-07	6.58e-06	9.53e-07	1.62e-06	1.91e-06	1.08e-08	2.99e-07	2.92e-08	6.31e-08	8.73e-08
0.10	9.53e-09	5.81e-08	1.96e-08	2.29e-08	1.39e-08	9.20e-11	2.29e-08	1.11e-08	1.07e-08	7.30e-09
0.50	5.84e-10	1.40e-09	8.78e-10	9.02e-10	2.42e-10	5.84e-10	1.40e-09	8.78e-10	9.02e-10	2.42e-10
1.00	2.26e-09	7.58e-09	4.05e-09	4.30e-09	1.80e-09	2.26e-09	7.58e-09	4.05e-09	4.30e-09	1.80e-09
1.50	6.74e+00	8.23e+00	7.57e+00	7.52e+00	4.52e-01	5.84e+00	8.68e+00	7.57e+00	7.53e+00	7.73e-01
5.00	1.14e+01	1.16e+01	1.15e+01	1.15e+01	7.32e-02	1.15e+01	1.21e+01	1.17e+01	1.17e+01	1.72e-01
5.50	1.14e+01	1.16e+01	1.16e+01	1.15e+01	7.06e-02	1.15e+01	1.20e+01	1.17e+01	1.17e+01	1.31e-01
10.00	1.17e+01	1.20e+01	1.18e+01	1.18e+01	1.04e-01	1.18e+01	1.20e+01	1.19e+01	1.19e+01	7.89e-02



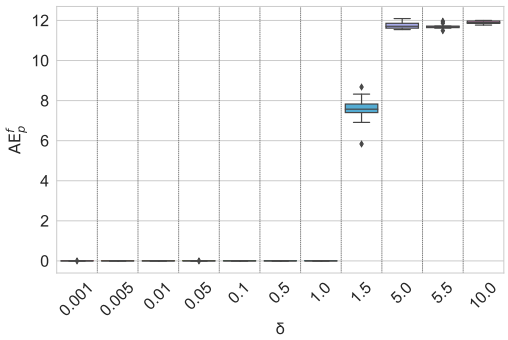
(a) Absolute error (optimistic) for upper level objective.



(b) Absolute error (optimistic) for lower level objective.

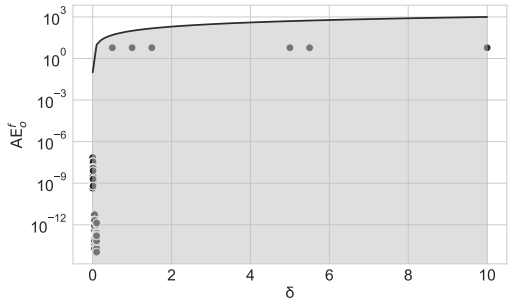


(c) Absolute error (pessimistic) for upper level objective.

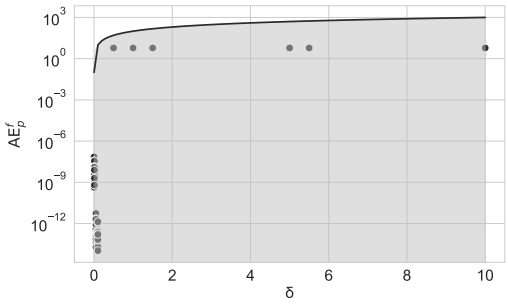


(d) Absolute error (pessimistic) for lower level objective.

Figure 3.17: Boxplots of Absolute Error (AE) for each different δ for PA, for Optimistic and Pessimistic Upper and Lower Level Functions over 20 runs.



(a) Optimistic case.



(b) Pessimistic case.

Figure 3.18: Theoretical error bound (shaded) and absolute error for lower level objective from 20 runs of PA for various values of δ.

3.7 Summary

This chapter examined the δ-perturbed formulation for addressing ill-posed bilevel optimization problems, where there are multiple optimal solutions at the lower level. Our analysis shows that this formulation successfully manages to approximate solutions by introducing a tailored perturbation strategy, ensuring only a single optimal solution in the lower level for any given upper-level decision. This adjustment facilitates an efficient search for both optimistic and pessimistic solutions using EAs. We also provide a proof of the approximate equivalence between the δ-perturbed formulation and the original optimistic and pessimistic formulations, supported by an error bound analysis. Computational tests, using DE across a range of test problems, further confirmed these theoretical bounds.

In the following chapter, we extend this approach to multiobjective bilevel optimization problems, presenting a proof of concept for applying the δ-perturbed formulation in a specific category of multiobjective bilevel problems and examining its effectiveness on approximating the optimistic and pessimistic Pareto fronts.

Connection to Thesis Hypothesis

This chapter supports Hypothesis 1 by demonstrating that the δ-perturbation method enhances evolutionary algorithms to efficiently approximate both optimistic and pessimistic solutions for ill-posed single-objective bilevel optimization problems, validated through theoretical proofs and numerical experiments on selected problems.

Chapter 4

Optimistic and Pessimistic Pareto-fronts in Multiobjective Bilevel Optimization via δ-Perturbation

In the previous chapter, we introduced the δ-perturbation approach for addressing ill-posed bilevel optimization problems. This chapter extends that approach to explore optimistic and pessimistic Pareto fronts in a multiobjective bilevel optimization problem, where the upper level has multiple objectives and the lower level has a single objective with multiple optimal solutions. Without specific knowledge of the lower level's choice function, the upper level must account for both optimistic and pessimistic scenarios, resulting in two distinct Pareto-optimal fronts. To approximate these fronts, we employ the existing m-BLEAQ evolutionary algorithm, which, combined with the δ-perturbation method, efficiently identifies both optimistic and pessimistic fronts. This proof-of-concept study demonstrates the method's effectiveness on two test problems, illustrating its capacity to accurately determine both Pareto-optimal fronts. This chapter is based on the work published in [206].

Connection to Thesis Objectives

This chapter addresses O2.1, O2.2, O3.2., O4.2.

4.1 Problem Definition

A bilevel optimization problem (BOP) is a distinctive category of optimization problems characterized by two interconnected levels of optimization: the upper level and the lower level. Each level controls a different set of variables and has its own objective function and constraints. Due to its hierarchical structure, the lower-level problem forms part of the constraints for the upper-level problem, requiring that any feasible solution for the upper level satisfies the optimality conditions of the lower-level problem. Extensive overviews of bilevel optimization can be found in [15], [93], [207]. When one or both levels involve

Parts of this chapter have been published in:
M. Antoniou, A. Sinha, and G. Papa, "A study on optimistic & pessimistic pareto-fronts in multiobjective bilevel optimization via δ-perturbation (accepted)," in *Proceedings of the 2025 International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, 2025

multiple objectives, then we have a multiobjective bilevel optimization problem (MBOP). For a detailed review of MBOPs, refer to [208].

When there are multiple lower-level solutions for each upper-level decision in a bilevel optimization problem, it means that there is a set of efficient lower-level decision vectors for each upper-level vector. This implies that the optimal lower-level solution for a given upper-level decision is not a singleton but rather a set, creating ambiguity about which lower-level solution the upper level should utilize when searching for the bilevel optimal solution(s). To address this, two approaches are commonly taken: the optimistic and pessimistic approaches [207]. In the optimistic approach, the upper level assumes that the lower level will choose the solution from its optimal set that is most beneficial for the upper level. In the pessimistic approach, the upper level assumes that the lower level may choose the option that is least favorable for the upper level.

In most situations, the upper level does not have knowledge of the choice function at the lower level, meaning that the follower could select any solution from its set of optimal solutions for a given upper-level decision. In multiobjective bilevel optimization, this can lead to the following two different Pareto- optimal fronts for the upper level: the optimistic (best-case) and the pessimistic (worst-case) fronts of the upper-level objectives. These fronts introduce additional decision-making challenges, and identifying these optimal fronts becomes the focus of this study.

We organize the remainder of this chapter as follows: Section 4.2 reviews the related work. Section 4.3 provides preliminaries on multiobjective optimization. In Section 4.4, we define the MBOP and its optimistic and pessimistic Pareto-optimal fronts. Section 4.5 introduces the proposed δ-perturbation approach for MBOPs. Section 4.6 briefly describes the m-BLEAQ algorithm. Section 4.7 presents an evaluation of the proposed method on two test problems. Finally, Section 4.8 concludes the chapter and discusses potential directions for future research.

4.2 Related Work

In bilevel optimization, the existence of multiple optimal solutions at the lower level introduces significant complexity, particularly in distinguishing between optimistic and pessimistic formulations. While the majority of bilevel optimization methods have concentrated on the optimistic approach due to its relative simplicity, the pessimistic approach is more difficult to solve yet is essential for achieving robust solutions [207].

In recent years, evolutionary approaches have been developed to address the multiobjective bilevel problem, such as [209]–[211]. In their paper [211], the authors suggested an evolutionary local-search algorithm for multiobjective bilevel problems that can handle more than one objective at the upper level. This work was an improvement over earlier versions of their algorithm [212], [213]. Further advancements in multiobjective bilevel solution methods were made in [214] by approximating the lower-level Pareto-optimal set.

The multiobjective bilevel evolutionary algorithm (m-BLEAQ) was proposed in [215], [216] to solve problems with multiple objectives at the upper level and a single objective at the lower level. A value function, representing the preferences of the lower-level decision maker, models lower-level optimization when both levels have multiple objectives. This value function selects a single preferred solution from the lower-level Pareto front, effectively reducing the multiobjective lower-level problem to a single-objective one. This reduction assumes that the preference structure of the lower-level decision maker is known to the upper-level decision maker. The algorithm's principle relies on estimating lower-level decisions as a function of upper-level decisions, utilizing local quadratic approximations to reduce computational costs. Similar bilevel mapping-approximation ideas are used to solve

single-objective bilevel optimization problems as well [217], [218]. When there is a single objective at the lower level and the lower level returns a singleton, neither an optimistic nor a pessimistic position emerges.

Most multiobjective methods [209], [211] assume an optimistic approach, primarily due to the relative ease of finding the Pareto-optimal front compared to the pessimistic case. Under a pessimistic approach, additional complexity arises from the intricate decision-making interactions between the upper level and lower level. However, solutions on the optimistic Pareto-optimal front may not always be realistic, as the upper level cannot control the choices of the lower level. The pessimistic Pareto front is essential as it provides a bound for worst-case lower-level responses. The decision maker ensures robustness by identifying this front; by selecting a point from the pessimistic front, the upper-level decision maker's realized solution remains unaffected by choices made by the lower-level decision maker within the front's dominated region.

In [219], the authors talked about semivectorial bilevel problems[1]. They also mentioned that an upper-level decision maker might look for intermediate (moderate) solutions in addition to optimistic and pessimistic ones, depending on how much risk they are willing to take. Recently, Deb et al. [169] introduced an evolutionary approach for such hierarchical problems, enabling upper-level decision makers to minimize deviations from their expectations due to independent lower-level decisions by identifying a set of pseudopessimistic solutions. To our knowledge, no study has approximated both the optimistic and pessimistic Pareto-optimal fronts in multiobjective bilevel optimization problems where the upper level has multiple objectives and the lower level has a single objective with multiple optimal solutions.

In classical bilevel optimization literature, approximation methods exist for problems with specific mathematical structures—such as linearity or convexity—where pessimistic formulations are perturbed and reformulated to facilitate solvability [138], [220]. It is possible to make sure that the lower-level problem has a unique (approximately) optimal solution for any given upper-level decision by using the right perturbation strategy for either optimistic or pessimistic formulations. In our work [139], as described in Chapter 3, we introduced the δ-perturbed formulation for bilevel optimization, designed to address the challenge of multiple optimal solutions at the lower level. This approach modifies the lower-level objective by incorporating the upper-level objective through a small perturbation parameter δ. We provided theoretical proofs establishing error bounds and computational results validating these findings. The perturbation approach enforces a unique optimal solution for the lower-level problem, enabling the application of standard bilevel algorithms to both optimistic and pessimistic formulations.

Building on this foundation, we extend the δ-perturbed approach to multiobjective bilevel optimization problems involving multiple objectives at the upper level and a single objective with multiple optimal solutions at the lower level [206]. Our contributions include reformulating such problems so that the lower level returns a unique solution for both optimistic and pessimistic scenarios. This reformulation enables us to identify both the optimistic and pessimistic Pareto-fronts. In this proof-of-concept study, we apply the m-BLEAQ algorithm [215] to the reformulated problem, demonstrating its effectiveness on two test cases where we successfully identify the optimistic and pessimistic Pareto optimal fronts.

[1] In semivectorial bilevel optimization, the upper-level problem has a single objective, while the lower-level problem has multiple objectives.

4.3 Preliminaries on Multiobjective Optimization Problems

In this section, we briefly define the multiobjective optimization problem, the concept of the Pareto front, multiobjective evolutionary algorithms, and their performance metrics.

4.3.1 Multiobjective Optimization Problems (MOPs)

Multiobjective optimization problems arise in many real-world scenarios where multiple conflicting objectives must be optimized simultaneously. These problems require trade-offs between competing objectives, leading to a solution space that reflects these trade-offs. Formally, an MOP can be represented as:

$$\min_{x \in X} \{f_1(x), f_2(x), \ldots, f_K(x)\}, \tag{4.1}$$

where X is the decision space, and $f_i(x)$ are the objective functions, each representing a criterion to be optimized [221]. In contrast to single-objective optimization, where a single optimal solution exists, MOPs with conflicting objectives yield a set of solutions that balance the objectives without any single solution optimizing all objectives simultaneously [222]. This concept of trade-offs is critical in multiobjective optimization, where solutions are assessed by their quality relative to multiple objectives rather than by absolute optimality in any single criterion.

4.3.2 Pareto Optimality

The solutions to MOPs are often evaluated based on the concept of Pareto optimality, named after Vilfredo Pareto, who introduced this idea in economic contexts. A solution is Pareto optimal if improving one objective cannot occur without compromising at least one other objective. Formally, a solution $x^* \in X$ is Pareto optimal if no other solution $x \in X$ exists such that:

$$f_i(x) \leq f_i(x^*) \quad \text{for all } i, \tag{4.2}$$

and

$$f_i(x) < f_i(x^*) \quad \text{for at least one } i. \tag{4.3}$$

The collection of all Pareto-optimal solutions forms the Pareto set, and its image to the objectives space forms the Pareto front that shows the trade-offs among the objectives [223]. Identifying or approximating this Pareto front is often the goal in solving MOPs, as it provides decision-makers with a spectrum of equally valid solutions that capture the inherent trade-offs among conflicting criteria [221].

4.3.3 Multiobjective Evolutionary Algorithms and Performance Metrics

Multiobjective Evolutionary Algorithms (MOEAs) are widely used for solving MOPs, particularly in cases where objectives conflict, the problem is high-dimensional, or traditional optimization methods are computationally infeasible [224]. MOEAs differ from single-objective evolutionary algorithms by seeking a set of solutions—rather than a single optimal point—that represents trade-offs among conflicting objectives. MOEAs are designed to approximate Pareto fronts by evolving a diverse population of solutions over multiple generations.

The general structure of an MOEA follows a few common steps. First, an initial population of solutions is generated, either randomly or by using problem-specific heuristics. Then, solutions are evaluated based on multiple objective functions, and non-dominated sorting is typically used to rank solutions according to their Pareto dominance. Solutions

are selected, combined, and mutated to form new solutions, promoting both convergence towards the Pareto front and diversity along it [223]. For instance, the Non-dominated Sorting Genetic Algorithm II (NSGA-II) [224] is a very popular MOEA that employs a fast non-dominated sorting approach combined with a crowding distance mechanism. This approach allows NSGA-II to maintain a spread of solutions across the Pareto front while ensuring convergence. Another popular algorithm, the Strength Pareto Evolutionary Algorithm 2 (SPEA2) [225], uses an external archive to store non-dominated solutions across generations, combined with a density estimation technique to maintain diversity. The Multiobjective Evolutionary Algorithm based on Decomposition (MOEA/D) [226] divides the MOP into multiple single-objective subproblems, each associated with a different weight vector. By solving these subproblems concurrently and sharing information among them, MOEA/D promotes a well-distributed Pareto front, particularly suitable for high-dimensional problems. The indicator-based evolutionary algorithm (IBEA) [227] uses a quality indicator, such as hypervolume, directly in its selection process, guiding the algorithm towards solutions that balance convergence and diversity based on that indicator.

There are various performance metrics, which are generally used to evaluate the solution quality of solutions produced by MOEAs. Generational Distance (GD) measures the average distance from each solution in the obtained front to the closest point on the true Pareto front. It ascertains how close a solution set is to the optimal trade-offs [228]. However, GD is not Pareto compliant, as dominated solutions can still reduce its value. Inverted Generational Distance (IGD) complements GD by measuring the average distance from points on the true Pareto front to the nearest solution in the obtained front, capturing the extent of Pareto front coverage [229]. Yet, IGD also lacks Pareto compliance since dominated solutions can influence the metric. IGD+ improves upon IGD by ensuring Pareto compliance; specifically, it incorporates Pareto dominance relationships such that solutions in the obtained front that are dominated do not contribute to the metric, while nondominated solutions closer to the true Pareto front positively influence the result [230]. The hypervolume (HV) metric captures both convergence and diversity by calculating the volume of the space between a reference point and the obtained Pareto front, providing a combined measure of solution quality [222]. Each of these metrics has certain requirements: while both GD and IGD (and IGD+) require knowledge of the true Pareto front, HV requires a reference point, whose choice might affect its value.

4.4 Mathematical Definition

In this section, we present the mathematical formulation of the multiobjective bilevel optimization problem. Moreover, we define the optimistic and pessimistic Pareto-optimal fronts.

4.4.1 Multiobjective Bilevel Optimization Problem

The general mathematical definition of a multiobjective bilevel optimization problem is as follows:

$$\text{“}\min_{x,y}\text{”}\, F(x,y) = (F_1(x,y),\ldots,F_K(x,y)) \tag{4.4}$$

$$\text{subject to} \tag{4.5}$$

$$y \in \operatorname*{argmin}_{y} \{f(x,y) = (f_1(x,y),\ldots,f_N(x,y)) : g(x,y) \leq 0\} \tag{4.6}$$

$$G(x,y) \leq 0 \tag{4.7}$$

where $G(x, y)$ represents the upper-level constraints, and $g(x, y)$ represents the lower-level constraints. Any bounds on x and y are included in the inequality constraint set.

In this study, we are interested in the case of multiobjective bilevel optimization problems, where the upper level has multiple objectives while the lower level has a single objective:

$$\text{“}\min_{x,y}\text{”}\, F(x, y) = (F_1(x, y), \ldots, F_M(x, y)) \tag{4.8}$$

subject to

$$y \in \operatorname*{argmin}_y \left\{ f(x, y) : g_(x, y) \leq 0 \right\} \tag{4.9}$$

$$G(x, y) \leq 0 \tag{4.10}$$

The problem under consideration in this paper can also be expressed in terms of set-valued mapping, Ψ as follows, where $g(x, y)$ represents vector-valued constraints:

$$\Psi(x) = \operatorname*{argmin}_y \{ f(x, y) : g(x, y) \leq 0 \} \tag{4.11}$$

The multiobjective bilevel optimization problem, in terms of $\Psi(x)$, can be formulated as a constrained optimization problem in the following manner:

$$\text{“}\min_{x,y}\text{”}\, F(x, y) = (F_1(x, y), \ldots, F_M(x, y)) \tag{4.12}$$

$$\text{subject to} \tag{4.13}$$

$$y \in \Psi(x) \tag{4.14}$$

$$G(x, y) \leq 0 \tag{4.15}$$

where the feasible sets X and Y have been omitted for brevity. Note that we are using quotations in our formulation as we have not defined the position that we take, i.e. optimistic or pessimistic, while solving the problem. Without stating the position, the above formulations are ill-posed.

4.4.2 Optimistic vs. Pessimistic Pareto Front

As discussed, when the lower level contains multiple optimal solutions for a given upper-level decision, we have to make some assumptions. If the upper level anticipates that the lower level will select the optimal solution from its set that best suits the upper level, we state that the upper level is taking an optimistic position. In such cases, the upper level aims to identify an optimistic Pareto-optimal front. The optimistic Pareto front corresponds to the solution of the following problem.

$$\min_{x,y \in \Psi^O(x)} F(x, y) = (F_1(x, y), \ldots, F_M(x, y)) \tag{4.16}$$

$$\text{subject to} \tag{4.17}$$

$$G(x, y) \leq 0 \tag{4.18}$$

$$\text{where } \Psi^O(x) = \arg\min_{y \in \Psi(x)} \{ (F_1(x, y), \ldots, F_M(x, y)) \} \tag{4.19}$$

Optimistic Pareto Front Definition:
The optimistic Pareto front P^O consists of all solutions x within the feasible region X and y within the subset of the optimal reaction set $\Psi^O(x)$ for which there is no other solution that simultaneously satisfies the following conditions for the objectives $F_i(x, y)$:

$$\forall i = 1 \text{ to } M,\ F_i(x', y) \leq F_i(x, y) \quad \text{for } x, x' \in X, y, y' \in \Psi^O(x) \tag{4.20}$$

$$\exists j \text{ such that } F_j(x', y') < F_j(x, y) \quad \text{for } x, x' \in X, y, y' \in \Psi^O(x) \tag{4.21}$$

If the upper level anticipates the worst-case from the lower level, we take the pessimistic approach and converge to the pessimistic Pareto front. The solutions here are:

$$\min_{x, y \in \Psi^P(x)} F(x, y) = (F_1(x, y), \ldots, F_M(x, y)) \tag{4.22}$$

$$\text{subject to} \tag{4.23}$$

$$G(x, y) \leq 0 \tag{4.24}$$

$$\text{where } \Psi^P(x) = \arg \max_{y \in \Psi(x)} \{(F_1(x, y), \ldots, F_M(x, y))\} \tag{4.25}$$

where $\Psi^P(x)$ is a subset of $\Psi(x)$, which provides the feasible pessimistic solutions of the upper level.

Pessimistic Pareto Front Definition:
The pessimistic Pareto front P^P consists of all solutions x within the feasible region X and y within the subset of the optimal reaction set $\Psi^P(x)$ for which there is no other solution that simultaneously satisfies the following conditions for the objectives $F_i(x, y)$:

$$\forall i = 1 \text{ to } M,\ F_i(x', y) \leq F_i(x, y) \quad \text{for } x, x' \in X, y, y' \in \Psi^P(x)$$

$$\exists j \text{ such that } F_j(x', y') < F_j(x, y) \quad \text{for } x, x' \in X, y, y' \in \Psi^P(x)$$

The relationship between the objective space (F-space) and the decision space $x - \Psi(x)$ for a multiobjective bilevel optimization problem (two upper-level objectives and one lower-level objective with multiple optimal solutions) is shown through Figure 4.1. For simplicity, assume that there are no upper-level constraints, $G(x, y) \leq 0$. On the left, the decision space $x - \Psi(x)$ is shown, where the shaded gray region represents the full set of feasible y for each x. The green curve represents a subset of optimistic feasible solutions, $\Psi^O(x)$, that contribute to the optimistic Pareto front, while the red curve represents a subset of pessimistic feasible solutions, $\Psi^P(x)$, that contribute to the pessimistic Pareto front. Optimistic and pessimistic solutions for a few x are shown as orange and red circles, respectively. The gray points denote solutions that are part of $\Psi(x)$ but do not lie on either the optimistic or the pessimistic Pareto fronts.

On the right, the objective space is depicted where functions F_1 and F_2 are the upper-level objectives. The green curve again represents the optimistic Pareto front (P^O) and in red, the pessimistic Pareto front is depicted (P^P) in the context of a minimization problem. The red, green, and gray points from the left-hand side figure map onto the corresponding points in the right-hand side figure. The left figure also shows a specific upper-level solution, x^0. In orange are all the lower-level optimal solutions. The green point on this orange line corresponds to a solution on the optimistic Pareto front. The red point corresponds to a solution on the pessimistic Pareto front. The gray point corresponds to neither of the two. All the solutions along the orange line in the left figure get mapped to the orange curve in the right figure.

It is important to note that in some sense the pessimistic Pareto front can be regarded as an upper bound (in the case of minimization) for the upper-level decision maker. If a upper-level decision maker chooses a decision (say (x^0, y')) from the pessimistic Pareto front, then no matter which solution among the multiple optimal solutions is picked up by the lower-level decision maker (say y'' for x^0), the solution (x^0, y'') will lie somewhere between the pessimistic and optimistic fronts. The figure illustrates one such scenario using green lines. However, note that it is not necessary that there will always be an optimistic Pareto point corresponding to a upper-level decision made from the pessimistic Pareto front (as in the case of upper-level decision x^1 for which lower-level decisions are shown with yellow lines). Since the realized solution may lie anywhere between the two fronts, the problem considered in this paper raises a number of decision making difficulties.

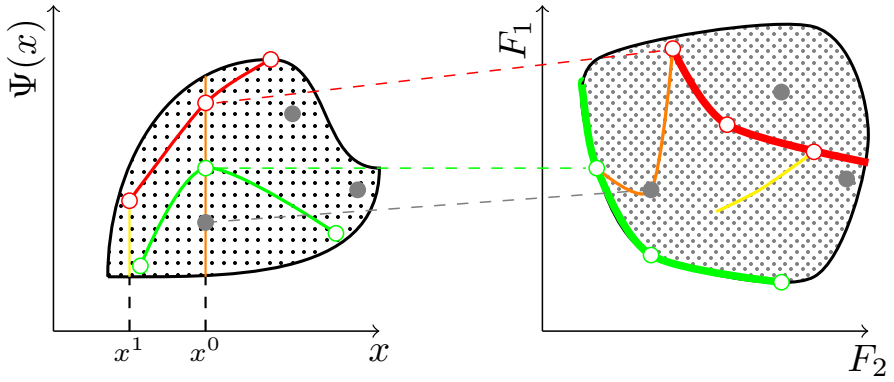


Figure 4.1: Illustration of the upper level decision space $x - \Psi(x)$ (left) and the upper level objective space F (right).

4.5 Mathematical Method: δ-perturbation of MBOPs

In this section, we explain the δ-perturbed formulation of the MBOP. For better understanding, let us first recall the δ-perturbed formulation of the single-objective bilevel problem,

$$\begin{aligned} \text{“}\min_{x,y}\text{”} \quad & F(x,y) \\ \text{subject to} \quad & G(x,y) \leq 0 \\ & y \in \operatorname{argmin}\{f(x,y) \,|\, g(x,y) \leq 0\} \end{aligned}$$

which has been thoroughly analyzed in [139] and in Chapter 3. For a small positive δ, the perturbed optimistic and pessimistic bilevel formulation is obtained by adding or subtracting $\delta F(x,y)$, respectively, as follows:

$$\min_{x,y} \quad F(x,y) \tag{4.26}$$

$$\text{subject to} \quad G(x,y) \leq 0 \tag{4.27}$$

$$y \in \operatorname*{argmin}_{y}\{f(x,y) \pm \delta F(x,y) \,|\, g(x,y) \leq 0\} \tag{4.28}$$

When δ is extremely small, the above formulation leads to an approximate optimistic/ pessimistic solution depending on the sign that is chosen. In this paper, we extend this idea for multiobjective bilevel problems where the upper level has multiple objectives,

while the lower level has a single objective but multiple optimal solutions corresponding to a upper-level decision.

Theorem 4.1. *For a bilevel optimization problem with multiple objectives at the upper level and a single objective at the lower level, approximate optimistic and pessimistic Pareto fronts can be obtained by solving the following bilevel problem with a '+' and '−' sign, respectively:*

$$\min_{x,w,y} \quad F(x,y) = (F_1(x,y), \ldots, F_M(x,y)) \tag{4.29}$$

$$\text{subject to} \quad G(x,y) \leq 0 \tag{4.30}$$

$$y \in \underset{y}{\operatorname{argmin}} \{ f(x,y) \pm \delta V(w, F(x,y)) \mid g(x,y) \leq 0 \} \tag{4.31}$$

Where $V(\cdot)$ represents a scalarization function and the vector w represents the preference vector.

The scalarization function $V(\cdot)$ should be a suitable scalarization method, such as a weighted sum scalarization [231], weighted Chebyshev scalarization [232], conic scalarization [233], or Benson's scalarization [234]. The preference vector w is considered an upper-level decision variable.

Proof. Let us write the problem in (4.8)-(4.10) into a single objective problem at the upper level for a given preference vector w.

$$\text{“}\min_{x,y}\text{”} \quad V(w, F(x,y)) \tag{4.32}$$

$$\text{subject to} \quad G(x,y) \leq 0 \tag{4.33}$$

$$y \in \underset{y}{\operatorname{argmin}} \{ f(x,y) : g(x,y) \leq 0 \} \tag{4.34}$$

$$\tag{4.35}$$

Using (4.26)-(4.28), one can find the approximated optimistic and the pessimistic solutions for (4.32)-(4.35) by solving the following problem.

$$\min_{x,w,y} \quad V(w, F(x,y)) \tag{4.36}$$

$$\text{subject to} \quad G(x,y) \leq 0 \tag{4.37}$$

$$y \in \underset{y}{\operatorname{argmin}} \{ f(x,y) \pm \delta V(w, F(x,y)) \mid g(x,y) \leq 0 \} \tag{4.38}$$

$$\tag{4.39}$$

If the above problem is solved for all possible values of w, it leads to all the (approximate) optimistic and (approximate) pessimistic solutions corresponding to w. Therefore, one can solve the multiobjective problem in (4.29)-(4.31) to get an approximation of the entire optimistic and pessimistic Pareto-fronts by choosing the appropriate signs $(+/-)$. □

To gain a clearer insight into how this perturbation works, let us look at the visual explanation in Figure 4.2. It demonstrates the link between upper-level and lower-level solutions, with the upper level having two objectives and the lower level being perturbed by the upper-level function. The plot on the left shows the upper-level objective space, where the objectives $F_1(x^*, y)$ and $F_2(x^*, y)$ are represented for a fixed x^* and a varying lower-level decision variable y and different weights w. The points $F(x^*, y^o)$ (green) and $F(x^*, y^p)$ (red) represent the two distinct solutions—optimistic and pessimistic respectively—for different values of y. The right plot represents the lower-level optimization

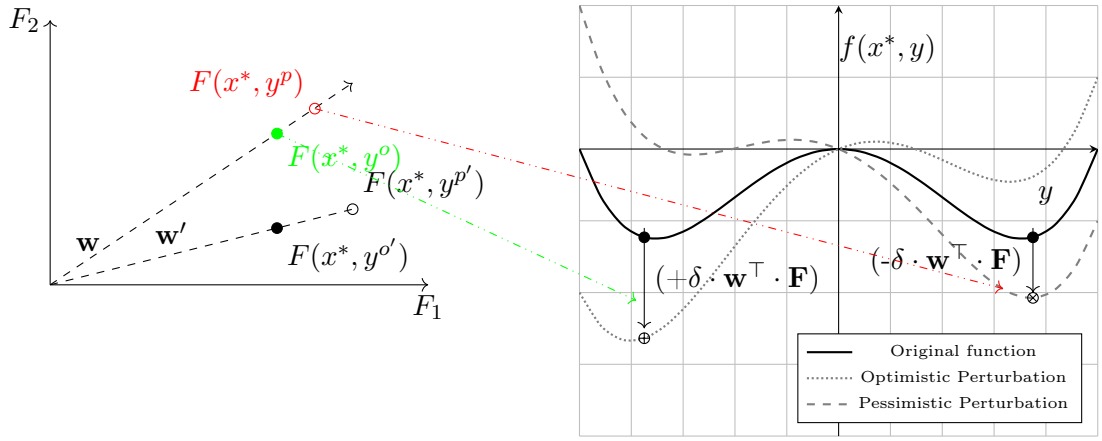


Figure 4.2: The left diagram shows the different solutions $F(x^*, y^o)$ and $F(x^*, y^p)$ in the upper level space, and the right diagram shows the corresponding perturbations of the lower level solutions in $f(x^*, y)$.

problem, $f(x^*, y)$, and the effects of perturbation on the lower-level objective. The black solid curve represents the original lower-level objective without perturbation, indicating the existence of two equal solutions. The gray dashed and dotted curves illustrate the pessimistic and optimistic perturbations, respectively. The points in the lower-level graph correspond to the same lower-level solutions, y^o and y^p, as depicted in the upper-level graph, showing how perturbations shift the solutions, changing the lower-level landscape to favor one or the other. The arrows between the two plots indicate the correspondence between the lower-level solutions and their impact on the upper-level objective space. This figure demonstrates how changes in the lower-level decision variable y, and associated perturbations, affect outcomes at the upper level and lower-level of the optimization problem.

4.6 Algorithmic Method: m-BLEAQ

The multiobjective BiLevel Evolutionary Algorithm based on Quadratic approximations, known as m-BLEAQ [215], is an evolutionary method that utilizes insights from parametric optimization theory to reduce computational costs. It is an extension of BLEAQ [217], which was developed for solving single-objective bilevel problems and was later extended to BLEAQ-II [218]. BLEAQ utilizes only the reaction set mapping in its implementation, while BLEAQ-II utilizes both the reaction set mapping and value function mapping to solve bilevel problems. In this study, we employ both the reaction set mapping and value function mapping in our m-BLEAQ implementation, following BLEAQ-II's approach. The reaction set mapping, denoted by $\psi(x)$, maps the upper-level decision variables x to the lower-level reaction set, approximating the optimal responses y at the lower level for a given x. The value function mapping, denoted by $\phi(x)$, approximates the optimal value function at the lower level by mapping the upper-level objective to the corresponding optimal lower-level objective value for a given upper-level objective. With approximations of the mappings in bilevel optimization, the algorithm quickly converges to the Pareto-optimal solutions of the multiobjective bilevel problem. The pseudocode for m-BLEAQ is provided in Algorithm 4.1. In the algorithm, the process begins by randomly initializing the population (step 2). For each individual, lower-level solutions are computed (step 3), and individuals are tagged as 1 if the lower-level optimization is successful, or as 0 if it is not (step 4). The fitness of each individual is assigned based on their non-domination rank and crowding distance

(step 5).Offspring are generated using genetic operations such as crossover and mutation (step 6). Lower-level solutions for the offspring are approximated using either the $\psi(x)$-mapping or $\phi(x)$-mapping through quadratic approximations (step 7). If the number of tag-1 individuals is less than half of the population size (step 8), lower-level optimization is performed for the offspring, and tags are updated accordingly (step 9). The fitness of the offspring is then assigned (step 10), and the population is updated by selecting the best individuals based on fitness (step 11). This process continues until the stopping criteria are met (step 12). Finally, all tag-1 individuals, representing the non-dominated solutions, are archived and returned as the output (step 13). In this work, m-BLEAQ is adapted with the perturbation method to enhance its performance in handling optimistic and pessimistic Pareto fronts (PFs).

Algorithm 4.1: m-BLEAQ

1: **Input:** Maximum upper-level evaluations $T_{\max}$, population size N.
2: Randomly initialize the upper-level (UL) population of size N.
3: Perform lower-level (LL) optimization for each UL member to compute LL solutions.
4: Tag UL members as *1* if LL optimization is successful, otherwise as *0*.
5: Assign fitness to UL members based on their non-domination rank and crowding distance.
6: **while** stopping criteria not met (e.g., $T_{\max}$ reached) **do**
7: Generate UL offspring using genetic operations.
8: Approximate LL solutions for offspring using either $\psi(x)$- or $\phi(x)$-mapping through quadratic approximations.
9: **if** Number of tag-1 UL members $< N/2$ **then**
10: Perform LL optimization for the UL offspring.
11: Update tags based on LL optimization results.
12: **end if**
13: Assign fitness to the offspring.
14: Update the UL population by selecting the best individuals based on fitness.
15: **end while**
16: Archive all tag-1 UL members to preserve the non-dominated solutions.
17: **Output:** Non-dominated solutions in the archive.

4.7 Numerical Experiments

In this section, we demonstrate the proposed approach, by applying it to two test problems, with two objectives at the upper level. The parameters used in m-BLEAQ were maintained at their default settings [213], with the exception of the population size and generation size, both adjusted to 500. The value of δ was set at 0.01. The weighted-sum scalarization technique is used, and the weight parameter w is sampled from $[0, 1]$. These parameters were selected to demonstrate the functionality of the proposed method, without explicit parameter tuning. A sufficiently large population size is used to assess the accuracy of the perturbation method, ensuring any inaccuracies are mainly due to the perturbation, not population or generation size. For each test problem, we conducted 20 independent runs on an Intel(R) Core(TM) i5-8365U CPU @ 1.60GHz with 16 GB of RAM running Windows 10 and MATLAB R2023b.

4.7.1 Test Problems

The first problem in our study, referred to as Test Problem 1, is taken from [212], where it originally includes two objectives at both the upper level and lower level. In our formulation, we modify the lower-level problem to have a single objective as follows:

$$\begin{aligned}
\min_{x,y} \quad & F_1 = (y_1 - 1)^2 + y_2^2 + x^2 \\
& F_2 = (y_1 - 1)^2 + y_2^2 + (x - 1)^2 \\
\text{subject to} \quad & y \in \arg\min_y \{\sqrt{y_1^2 + y_2^2} + \sqrt{(y_1 - x)^2 + y_2^2} : -1 \leq y_1, y_2 \leq 2\} \\
& -1 \leq x \leq 2
\end{aligned}$$

The optimistic PF is obtained when $x \in [0.5, 1]$ and $y_1 = x, y_2 = 0$. In contrast, the pessimistic PF is derived for $x \in [0, 1]$ and $y_1 = y_2 = 0$ [219]. We illustrate the characteristics of Test Problem 1 through Figure 4.3, which shows the relationship between optimistic and pessimistic solutions in the upper-level objective space. If, for any fixed x, the lower level has multiple optimal solutions to choose from, then we refer to the non-dominated solution(s) with respect to the upper-level objectives as optimistic solution(s). Conversely, for that x, the solutions in the subset of pessimistic feasible space are referred to as pessimistic solution(s), with the non-dominated solutions among them forming the pessimistic Pareto front. For certain problems, for any given x, the optimistic/pessimistic solution(s) may coincide. Our objective is to find the optimistic and pessimistic fronts with respect to upper-level objectives, i.e., the optimistic and pessimistic Pareto fronts. Figure 4.3a shows the optimistic and pessimistic feasible solutions in the upper-level objective space. The optimistic Pareto front is shown as a solid line, while the dashed line represents the pessimistic Pareto front. The black circles represent pessimistic-dominated solutions, while the crosses denote optimistic-dominated solutions. Naturally, all pessimistic solutions, including both the Pareto front and the dominated solutions, are dominated by the optimistic Pareto front. Figure 4.3b illustrates the range of lower-level optimal solutions for a fixed value of x in the upper-level decision space. The multiple lower-level optimal solutions for any x can be generated by fixing $y_2 = 0$ and varying y_1.

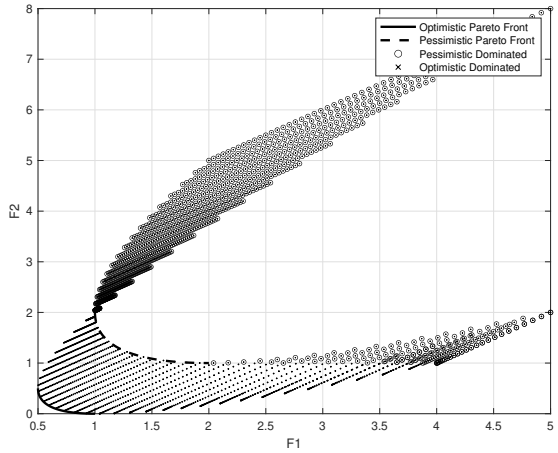
The second problem considered in our study is Test Problem 2, which combines a modified SMD1 at the upper level and SMD6 at the lower level [155]. The SMD6 is selected for the lower level due to its multimodal structure with multiple global optima. The description of Test Problem 2 is as follows:

$$\begin{aligned}
\min_{x} \quad & F_1 = x_1^2 + y_1^2 + y_2^2 + x_2^2 + (x_2 - y_3)^2 \\
& F_2 = 1 - |x_1| + x_2^2 + y_1^2 + y_2^2 + (x_2 - y_3)^2 \\
\text{subject to} \quad & y \in \arg\min_y \{x_1^2 + (y_2 - y_1)^2 + (x_2 - y_2)^2 : -5 \leq y_1, y_2 \leq 10\} \\
& -5 \leq x_1, x_2, x_3 \leq 10
\end{aligned}$$

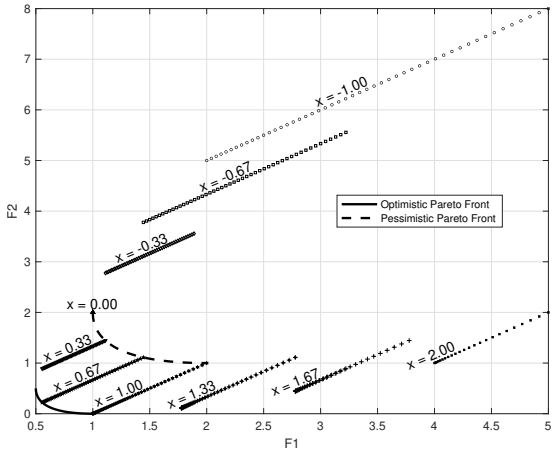
The optimistic PF is obtained when $x_1 \in [-5, 10]$ and $x_2 = y_1 = y_2 = y_3 = 0$, while the pessimistic PF is derived for $x_1 \in [-5, 10]$ and $x_2 = y_3 = 0, y_1 = y_2 = 10$.

4.7.2 Results and Discussion

To evaluate the performance of the algorithm, it is essential to choose an appropriate performance indicator to assess the quality and diversity of the solutions obtained. Commonly used indicators in multiobjective EAs include hypervolume (HV) [222], inverted generational distance (IGD) [235], and others. While using EAs with a perturbation strategy, the Pareto-optimal solutions obtained are an approximation of the true optima. Therefore, a



(a) Optimistic and Pessimistic Feasible Solutions in the upper level Objective Space.



(b) Solutions for fixed x^* and varying y_1 ($y_2 = 0$).

Figure 4.3: Illustration of the optimistic and pessimistic feasible solutions in the upper level objective space for Test Problem 1, highlighting the different sets of solutions and their corresponding Pareto fronts.

better Pareto front in terms of HV may not necessarily indicate superior algorithm performance, as an inaccurate solution at the LL may provide an artificially better UL solution. Instead, IGD calculates the sum of distances between each point of the true Pareto front and the nearest non-dominated set found by the algorithm. Smaller IGD values indicate that the approximated points are closer to the Pareto front of the problem. IGD+ [230], a Pareto-compliant alternative, modifies IGD by incorporating dominance relationships to exclude dominated solutions, ensuring a more accurate assessment. Both IGD and IGD+ are used in this study to analyze the quality of the approximated Pareto front.

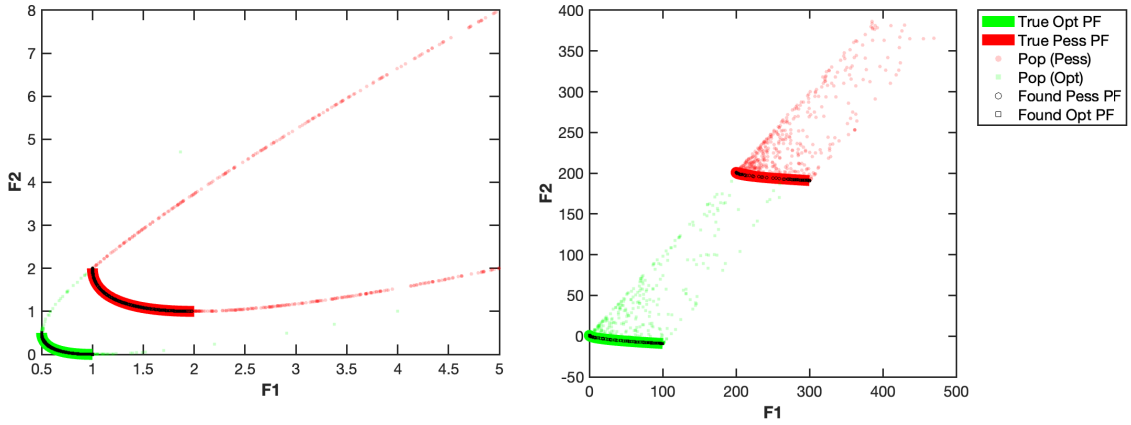
We report four IGD values, IGD_O, IGD_{O+}, IGD_P, and IGD_{P+}, for the optimistic and pessimistic Pareto fronts in Table 4.1, calculated as the median of 20 runs. The values were computed using 1000 uniformly distributed reference points. For the first problem, all values are of the order 0.001, while for the second problem, they are of the order 0.01. This difference can be attributed to the higher dimensionality and complexity of the second problem. In all cases, $IGD+$ values are slightly higher than IGD due to the exclusion of dominated solutions, which can artificially improve IGD by reducing the apparent distance to the true Pareto front. Both results demonstrate that the algorithm effectively approximates the true Pareto fronts.

Table 4.1: Median IGD and IGD+ values for the Test Problems over 20 runs.

	IGD_O	IGD_{O+}	IGD_P	IGD_{P+}
Test Problem 1	2.218e-03	2.699e-03	2.652e-03	2.653e-03
Test Problem 2	6.097e-02	6.30e-02	5.930e-02	6.091e-02

To further illustrate the performance, we display in Figure 4.4 the true optimistic (green line) and pessimistic (red line) Pareto fronts, alongside the final population solutions (green and red) and the Pareto fronts derived from these solutions for the two Test Problems after one run. The black circles indicate the pessimistic-nondominated solutions from the final population, while the black crosses show the optimistic-nondominated solutions. In Figure

4.4a, which corresponds to Test Problem 1, the solutions found are densely distributed and spread well, closely approximating the true Pareto fronts. In Figure 4.4b, which corresponds to Test Problem 2, the solutions are less densely distributed, which aligns with the slightly higher IGD values observed above.



(a) Test Problem 1. (b) Test Problem 2.

Figure 4.4: True and algorithm-obtained Pareto fronts for optimistic and pessimistic cases and algorithm's final population.

For both test problems, the optimistic and pessimistic Pareto fronts can become valuable tools for decision-makers. For example, in both test problems, the optimistic and pessimistic solutions deviate significantly. From a decision-maker's perspective, the risk of adopting the optimistic assumption may result in surprises and substantial costs to the UL decision maker. This makes such a kind of analysis critically important. Also note that identifying the optimistic and pessimistic Pareto fronts does not completely solve the decision making challenges for these problems.

As a final remark, it is important to highlight that, as previously mentioned, we used the weighted sum scalarization, and it worked well since the specific Pareto fronts are convex. In the case of nonconvexity, other suitable methods should be selected.

4.8 Summary

In this chapter, we investigated a multiobjective bilevel optimization problem, where the upper level has multiple objectives and the lower level has a single objective with multiple optimal solutions for any given upper-level decision. We began by defining the problem and introducing both optimistic and pessimistic Pareto fronts in the context of multiobjective bilevel optimization. To approximate these fronts, we developed a perturbation technique that adjusts the lower-level objective by a small δ, based on the weighted values of the upper-level objectives. The reformulated bilevel problem was then addressed using m-BLEAQ, an evolutionary algorithm capable of managing multiple objectives at the upper level. The effectiveness of the proposed method was demonstrated through tests on two benchmark problems, successfully approximating both the optimistic and the pessimistic Pareto fronts. One of the main issues of using EAs in bilevel optimization is their computational demand; however, their study provides valuable insights into the intricate ways in which both level problems can mutually influence each other in optimistic and pessimistic

approaches. This can provide a new aspect to solving multiobjective bilevel problems and offer more informed potential solutions to decision-makers.

In the next chapter, we shift our focus to a specific kind of bilevel problem, the min-max problem, and review existing evolutionary methods relevant to robust optimization.

Connection to Thesis Hypothesis

This chapter supports Hypothesis 2 by demonstrating that the integration of the δ-perturbation approach within evolutionary algorithms (EAs) enables effective identification of both optimistic and pessimistic Pareto fronts in multiobjective bilevel optimization problems (MBOPs).

Chapter 5

Min-Max Optimization: Key Concepts and Evolutionary Algorithms

Building on our previous discussion of bilevel optimization, this chapter focuses on min-max optimization, a type of bilevel optimization in which both levels have the same objective. The purpose of min-max optimization is to optimize under the worst-case scenario. This chapter defines the underlying principles and mathematical formulations of min-max problems and presents an example of robust optimization. It also examines the connections to game theory, specifically zero-sum games, and describes known evolutionary methods developed for this kind of problems.

Connection to Thesis Objectives

This chapter addresses O1.2, O1.3.

5.1 Introduction

Uncertainty is present in real-world decision-making. Whether in construction, supply chain management, or aerospace engineering, decision-makers in every field often must account for worst-case scenarios, as overlooking them can compromise the success and reliability of their plans. For example, in supply chain management, unexpected demand could result in lost profits or unplanned delays. Similarly, in aerospace engineering, systems must be designed not for average conditions but for worst-case scenarios, such as extreme radiation levels, micrometeorite impacts, or unforeseen atmospheric changes in drag that could jeopardize mission success. In structural engineering, bridges must be designed not only for the typical weather and traffic conditions but also for extreme events such as earthquakes and severe storms. Optimization for these worst-case scenarios ensures that

Parts of this chapter have been published in:
M. Antoniou and G. Papa, "Evaluation of parallel hierarchical differential evolution for min-max optimization problems using scipy," in *International Conference on Bioinspired Optimization Methods and Their Applications*, Springer, 2022, pp. 84–98
M. Antoniou and G. Papa, "Differential evolution with estimation of distribution for worst-case scenario optimization," *Mathematics*, vol. 9, no. 17, p. 2137, 2021
M. Antoniou and G. Papa, "Solving min-max optimisation problems by means of bilevel evolutionary algorithms: A preliminary study," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 2020, pp. 187–188

the systems are designed to function safely and successfully even under the most challenging and/or rare conditions.

To address such challenges, it becomes essential to optimize under uncertainty. The framework of min-max optimization specifically allows for the minimization of maximum possible losses and thereby gives robust solutions that would still stand in worst-case scenarios. This allows decision-makers to design systems that are less prone to uncertainty and variability by focusing on the most adverse scenarios. While the solutions derived from this approach tend to be more conservative, they are crucial in environments where unpredictable variables can cause significant disruptions. By preparing for the worst, decision-makers can ensure that their systems are more resilient and better equipped to handle uncertainty when it arises.

The chapter is structured as follows: Section 5.2 presents the mathematical formulation of min-max optimization. Section 5.4 provides an example of its application in robust optimization. Section 5.5 explores min-max optimization in game theory, particularly in zero-sum games. Section 5.3 discusses the connection between min-max optimization and bilevel optimization. Section 5.6.2 reviews evolutionary algorithms for solving min-max problems. Finally, Section 5.7 concludes the chapter.

5.2 Mathematical Formulation of Min-Max Problems

The general unconstrained min-max problem can be formulated as:

$$\min_{x \in X} \max_{y \in Y} f(x, y) \tag{5.1}$$

where $X \subseteq \mathbb{R}^n$ is the feasible set for the minimization variable x, $Y \subseteq \mathbb{R}^m$ is the feasible set for the maximization variable y, and $f : X \times Y \to \mathbb{R}$ is a continuous function representing the objective. In this formulation, for each choice of $x \in X$, the function $f(x, y)$ is maximized over all $y \in Y$, yielding the inner optimal value $\max_{y \in Y} f(x, y)$. The outer optimization then minimizes this maximum value by selecting the optimal $x^* \in X$ such that

$$x^* = \arg\min_{x \in X} \left(\max_{y \in Y} f(x, y) \right) \tag{5.2}$$

The problem is considered *symmetrical* if the following condition holds:

$$\min_{x \in X} \max_{y \in Y} f(x, y) = \max_{y \in Y} \min_{x \in X} f(x, y) \tag{5.3}$$

This condition guarantees the existence of a saddle point (x^*, y^*) such that [239]:

$$f(x^*, y) \leq f(x^*, y^*) \leq f(x, y^*) \quad \forall x \in X, \forall y \in Y \tag{5.4}$$

When the symmetry condition is satisfied, the min-max problem can be simplified. Specifically, if we can find or approximate the saddle point (x^*, y^*), the problem reduces to a standard optimization problem [240]:

$$\min_{x \in X} f(x, y^*) \tag{5.5}$$

In this case, finding the global minimum with respect to x becomes straightforward, as y^* is known or well-approximated.

5.2.1 Examples of Symmetrical and Asymmetrical Problems

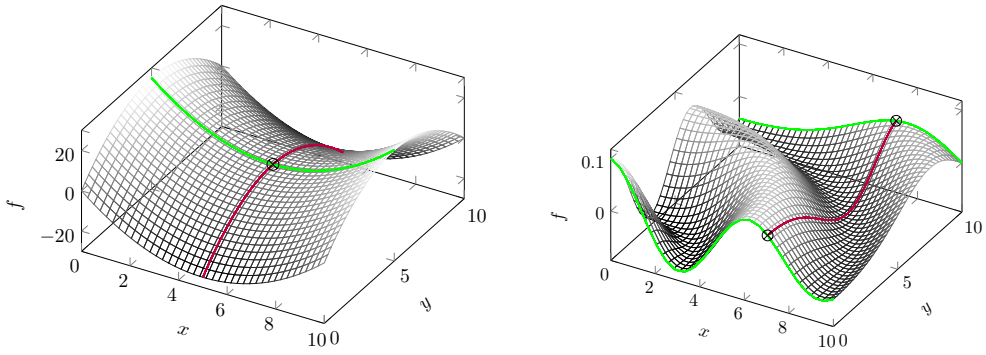
To illustrate these symmetry conditions, we present here an example of a symmetrical and an asymmetrical problem. The first example is a symmetrical function:

$$\min_{x \in X} \max_{y \in Y} f(x, y) = (x - 5)^2 - (y - 5)^2 \tag{5.6}$$

where $X = [0, 10]$ and $Y = [0, 10]$.The known solution is $x^* = 5$ and $y^* = 5$ with an optimal value of $f_1(x^*, y^*) = 0$. This test function is a saddle point function. The function along with the known optimum is plotted in Figure 5.1a and it serves as an example of a symmetric function. The second example is an asymmetrical function:

$$\min_{x \in X} \max_{y \in Y} f(x, y) = \frac{\cos\left(\sqrt{x^2 + y^2}\right)}{\sqrt{x^2 + y^2} + 10} \tag{5.7}$$

where $X = [0, 10]$ and $Y = [0, 10]$. The known optimal solutions are $x^* = 7.0441$ and $y^* = 10$ or $y^* = 0$ and the optimal value is $f_4(x^*, y^*) = 0.042488$. It is a damped cosine wave function, as shown in Figure 5.1b. In both figures, the green line indicates the



(a) f symmetrical (b) f asymmetrical

Figure 5.1: 3D mesh plots of the test functions. The black circle corresponds to the known optimum.

direction in which the outer level is optimized, while the red line shows the direction in which the inner level is optimized. In the symmetrical case, the optimal solutions for both the upper and lower levels converge to the same point, where the min-max is equal to the max-min. However, in the asymmetrical case, it is clear that the optimal solutions of the min-max problem do not correspond to those of the max-min.

5.3 Connections to Bilevel Optimization

The min-max problem is naturally connected to bilevel optimization, as it has a similar mathematical structure and is regarded as special case of bilevel problems. Recent work has shown that robust optimization problems—such as static robust problems, worst-case regret problems, and two-stage robust problems—can often be reformulated as bilevel problems, whereas bilevel structures are generally more complex and cannot always be directly translated into robust forms [241]. Earlier work by the author of this thesis provided empirical evidence of this connection by evaluating evolutionary bilevel algorithms on a series of min-max test functions, helping to establish a foundation for shared methods between

the bilevel and min-max optimization communities [238]. Let us recall the bilevel problem definition given in Chapter 2, as follows:

$$\begin{aligned} \min_{x \in X, y \in Y} \quad & F(x,y) \\ \text{subject to} \quad & G_k(x,y) \leq 0,\ k = 1, ..., K, \\ & \min_{y \in Y} \quad f(x,y) \\ & \text{subject to} \quad g_j(x,y) \leq 0,\ j = 1, ..., J, \end{aligned} \tag{5.8}$$

where K is the number of constraint functions for the upper level, and J is the number of constraint functions for the lower level. Here, y is the solution of the lower-level problem from the set $Y \subseteq \mathbb{R}^n$, based on the solution x from the upper level's set $X \subseteq \mathbb{R}^m$. The objective function F represents the upper level, while f represents the lower level. In the same vein, min-max problems are special instances of bilevel problems, where $f = -F$. In this thesis, we focus on unconstrained test functions; therefore, we provide the definition for the unconstrained min-max problem. Nevertheless, (5.9) can be extended to constrained min-max problems. The general unconstrained min-max problem can be described as bilevel problem as follows:

$$\begin{aligned} & \min_{x \in X} f(x,y) \\ \text{subject to} \quad & \max_{y \in Y} f(x,y) \end{aligned} \tag{5.9}$$

where $X \subseteq \mathbb{R}^m$ represents the set of candidate solutions and $Y \subseteq \mathbb{R}^n$ the set of all possible scenarios. In this formulation, the upper and lower levels share the same objective function $f(x,y)$: the upper level minimizes with respect to the decision variables x in the solution space, while the lower level maximizes with respect to the uncertain parameters y in the scenario space. Figure 5.2 provides a visual representation of this concept.

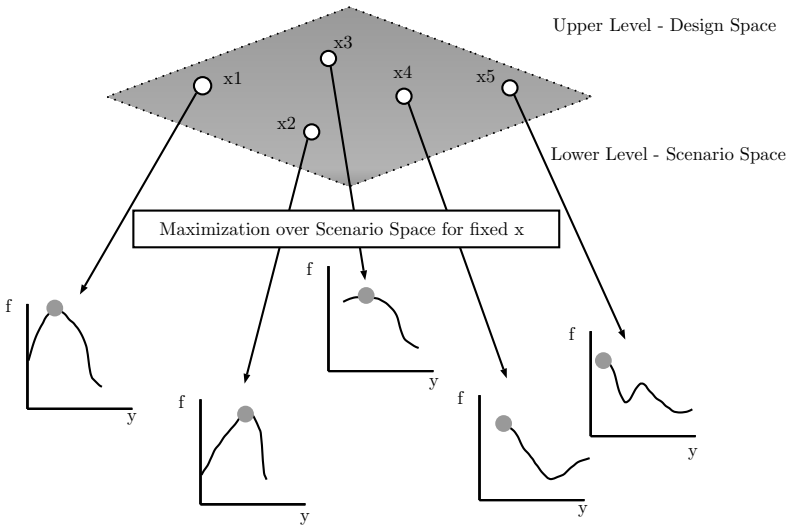


Figure 5.2: A general sketch of the min-max optimization problem as a bilevel problem, inspired by [242].

5.4 Min-Max for Robust Optimization

Robust optimization is an essential approach in the design of systems that have to function reliably under uncertain and variable conditions. Traditional optimization techniques often

assume all parameters are precisely known, which results in solutions that can fail in the real world. In contrast, robust optimization explicitly takes account of uncertainty, seeking to find solutions that perform well across a range of possible scenarios. This is particularly critical in engineering, where a system should be so designed that it could bear the fluctuations in the environmental conditions, material properties, and operational parameters without degradation in performance or safety [243].

Robust optimization can be viewed as a game between two players: the *designer* and *nature*. The *designer* is the decision-maker, who chooses a design d from the design space $\mathcal{D}$. Nature, representing the uncertainties or challenges that the design may face, acts as the opponent, selecting the worst-case scenario s from the scenario space S in response to the designer's choice. In this game, the designer must anticipate the worst possible conditions $s \in S$ to ensure the design remains robust and performs well despite nature's adversities.

The necessity for robust optimization can be contextualized with the help of the idea of *nominal design*. Nominal design is about seeking to optimize a system with regard to a certain, pre-defined scenario—often one which represents the most likely, or average conditions. We optimize under these conditions because they are representative of the most expected or standard operating environment. Nominal design can be mathematically written as:

$$\min_{\mathbf{d}\in D} f(\mathbf{d}, \mathbf{s}_0) \tag{5.10}$$

where $\mathbf{d}$ represents the design variables chosen by the designer, D is the feasible set of designs, and $\mathbf{s}_0$ is a fixed, predefined scenario. In this context, $\mathbf{s}_0$ is not part of the decision variables; rather, it is an assumed condition under which the optimization takes place. The goal is to find a design $\mathbf{d}$ that performs optimally under the specific scenario $\mathbf{s}_0$. However, if the actual conditions deviate significantly from $\mathbf{s}_0$, the design may become vulnerable, as it was optimized for a scenario that does not encompass possible variations in conditions.

In contrast, min-max optimization (or worst-case design) directly addresses the challenge of ensuring that a design remains effective even under the most adverse conditions that could arise within a defined uncertainty set. In this framework, the designer aims to minimize the maximum possible loss or risk that nature could impose. This interaction is captured mathematically as:

$$\min_{\mathbf{d}\in D} \max_{\mathbf{s}\in S} f(\mathbf{d}, \mathbf{s}) \tag{5.11}$$

Here, nature operates as an opponent, selecting the worst-case scenario $\mathbf{s}$ from the set S. The designer's purpose is to select an ideal design $\mathbf{d}$ for this worst-case situation. Minimizing the worst-case impact, min-max optimization assures that the design is robust and reliable regardless of whatever scenario within S occurs. A simple engineering example of this application is provided in the next section.

5.4.1 Illustrative Example of Robust Optimization in Engineering

The following example considers the design of a truss structure, which is composed of straight, linear members connected at their ends by joints. Each member carries tensile or compressive forces depending on the applied load. The design aims to ensure that the maximum stress in any member stays within acceptable limits, regardless of the direction of the acting load. We are going to solve both nominal optimization and robust optimization problems and plot the outcome to highlight how the design result changes when considering optimization under uncertainty.

As discussed, the nominal optimization approach seeks to minimize maximum member stress by determining optimal cross-sectional areas for truss members under a specific

nominal load P, ensuring that internal forces remain within the material's yield strength. This is mathematically represented as:

$$\min_{A_1,\ldots,A_n} \max_i \left| \frac{F_i}{A_i} \right| \tag{5.12}$$

where A_i is the cross-sectional area of the i-th truss member, F_i is the force in the i-th truss member under the nominal load, and n represents the total number of truss members.

In contrast, robust optimization minimizes the maximum stress across all possible load directions, ensuring that the truss design is efficient and reliable under varying conditions. The robust optimization problem is formulated as:

$$\min_{A_1,\ldots,A_n} \max_{\theta\in\Theta} \max_i \left| \frac{F_i(\theta)}{A_i} \right| \tag{5.13}$$

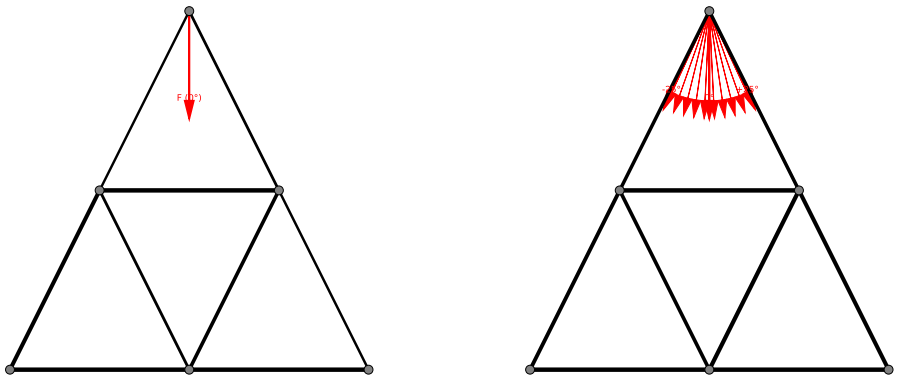
where Θ represents the set of possible load directions, A_i is the cross-sectional area of the i-th truss member, and $F_i(\theta)$ is the force in the i-th truss member under load direction θ. For both nominal and robust optimization, the stress in the i-th truss member is given by:

$$\sigma_i(\theta) = \frac{F_i(\theta)}{A_i} \tag{5.14}$$

where the force $F_i(\theta)$ is computed from the applied load vector $\mathbf{F}(\theta)$ and the direction vector of the member $\mathbf{n}_i$:

$$F_i(\theta) = \mathbf{F}(\theta) \cdot \mathbf{n}_i \tag{5.15}$$

Figure 5.3a shows the cross-sectional areas optimized with respect to a given nominal load direction, whereas Figure 5.3b gives the result for optimization against all possible load directions. While this approach requires larger cross-sectional areas for certain truss members—thus using more material and increasing costs—it ensures the truss performs reliably under varying load conditions, offering greater strength and stability compared to the nominal design.



(a) Nominal truss design optimization (b) Robust truss design optimization

Figure 5.3: Comparison of nominal and robust truss optimization.

In Figure 5.4, the right plot shows the stress levels for both the optimal nominal and robust designs across different load directions, while the left plot displays the density of stress values for each design. The stress-load direction plot (right) shows that the nominal design, optimized for a specific load direction ($\theta = 0$), experiences a sharp increase in stress

as the load direction changes ($-25° \leq \theta \leq 25°$). However, there is a small region where the stress levels for both designs coincide, indicating similar performance in this range of load directions. Outside this region, the robust design maintains more stable stress levels across all directions, ensuring consistent performance. The density plot (left) highlights the robust design's lower variability in stress values, with a narrower distribution indicating more stable performance. In contrast, the nominal design has a wider spread, suggesting greater variability and sensitivity to load direction changes.

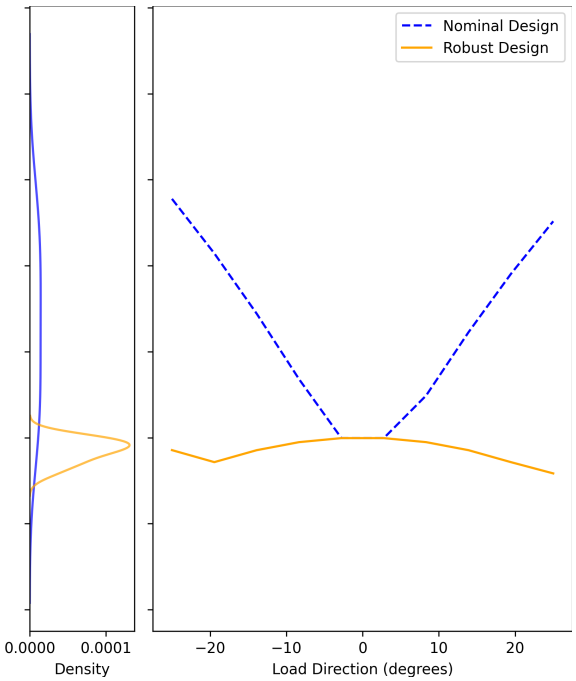


Figure 5.4: Performance Comparison of Nominal vs Robust Design Under Varying Load Directions. The nominal case is where $\theta = 0$, while in the robust one $\theta \in [-25, 25]$.

5.5 Min-Max Optimization in Game Theory: Zero-Sum Games

Game theory is related to min-max optimization, especially when it comes to zero-sum games [244]. In general, in a zero-sum game, two players are competing where the gain of the first player equals the loss of the other. In that case, there can also be seen a direct relationship with min-max problems, where one player tries to minimize his potential loss, taking into account an opponent's strategy that maximizes his gain.

Consider for example a simple two-player game where Player X and Player Y are choosing strategies independently. The outcome is determined via payoff matrix given by Table 5.1 which both the players are aware of. The values in the matrix represent payments from Player X to Player Y: positive values indicate a payment from Player X to Player Y, while negative values represent a payment from Player Y to Player X. For example, if Player X selects strategy A and Player Y selects strategy S, Player X must pay 9 to Player Y. Alternatively, if Player X selects strategy B and Player Y selects strategy Q, Player Y would pay 2 to Player X.

In doing so, Player X is trying to minimize her maximum possible loss, considering the worst-case scenario for every action. By analyzing the payoff matrix, Player X concludes that if she chooses strategy A, her worst-possible loss would be 9; if she chooses strategy B, her worst-possible loss would be 8; and if she chooses strategy C, her worst-possible loss

Player X	Player Y			
	P	**Q**	**R**	**S**
A	3	7	4	9
B	5	-2	8	6
C	-1	4	0	3

Table 5.1: Payoff matrix representing payments from Player X to Player Y.

would be 4. Because Player X wants to reduce her highest possible loss, she will choose strategy C, knowing that in the worst case she could lose 4. Her choice is a min-max strategy because she tries to minimize her biggest possible loss. This approach is crucial in zero-sum games, in which each player assumes that their opponent will choose the strategy that maximizes their own payoff.

The previous example represents a discrete min-max problem, in which there is a finite, countable number of choices of strategies. In many practical problems, however, the set of possible strategies may be continuous, leading to continuous min-max problems. Unlike a finite set of strategies, the decision variables are considered continuous functions, as was presented mathematically in the previous section. The goal remains the same: to find that strategy that minimizes the maximum possible loss, which is often referred to as finding a saddle point in continuous optimization [245], [246].

5.6 Overview of Methods

5.6.1 Classical Methods for Min-Max Optimization

Classical approaches to handling uncertainty in optimization include robust optimization methods from mathematical programming, which aim to find solutions that remain feasible under worst-case conditions. Robust counterpart formulations transform uncertain constraints into deterministic equivalents by ensuring feasibility across all possible realizations within a predefined uncertainty set [247]. Similarly, chance-constrained programming ensures constraint satisfaction with a specified probability, while stochastic programming models uncertainty through multiple scenarios and seeks solutions that perform well across them [248], [249]. While these methods provide strong theoretical guarantees, they typically assume convexity, differentiability, and precise knowledge of uncertainty distributions, making them less effective for complex, nonconvex, or high-dimensional problems. A comprehensive review can be found in [25] and [243].

In classical approaches towards robust optimization problems formulated as min-max problems, the solution of worst-case solutions is the central focus, solved through problem-specific tailored algorithmic approaches. Branch and bound techniques explore the solution space in stages, dividing it into subproblems and using bounds to eliminate suboptimal solutions, but they struggle with scalability as the size of the problem grows [250]. Second-order methods exploit Hessian-based approximations to locate saddle points in nonconvex-nonconcave scenarios, but they fail in black-box problems where gradient information is absent [251]. Cutting plane methods iteratively improve the feasible region using linear approximations and work well when the function is convex-concave, but such structures are uncommon in real-world problems, and nonconvex formulations are computationally expensive [252]. Lagrangian relaxation and other duality-based methods simplify the solution process by reducing the original problem to its dual form, but their applicability relies

on maintaining strong duality conditions, which often fail in nonconvex settings [253].

These classical methods work well for structured problems with well-defined uncertainty sets; however, they have serious limitations in sophisticated, high-dimensional, or dynamically evolving domains. Evolutionary algorithms (EAs), however, provide a flexible solution, especially for black-box problems which are nonconvex and follow an implicit uncertainty. Unlike robust mathematical programming approaches, EAs do not require gradient information or explicit uncertainty modeling, making them effective for problems where uncertainty arises from simulations or empirical data. Although EAs may involve higher computational effort and lack convergence guarantees, their adaptability allows them to explore diverse solution landscapes and identify robust solutions across a wide range of uncertainty scenarios. Therefore, while classical robust optimization and traditional min-max solution methods provide strong theoretical foundations, we assume that the problems resolved in this dissertation are of black-box structure, nonconvexity, and implicitly defined uncertainty. These assumptions make EAs a more suitable choice due to their flexibility, adaptability, and ability to navigate complex, real-world optimization landscapes.

5.6.2 Evolutionary Algorithms for Min-Max Optimization

EAs have gained popularity in min-max problems due to their ability to handle nested structures and non-convex landscapes, using population-based search to find robust solutions without relying on gradients. To explore the connection between EAs and robust optimization or min-max optimization, we searched a query[1] in the Web of Science. Figure 5.5 illustrates the keyword co-occurrence network from the publications found from the query, where the frequency of occurrence between two terms in a document is represented. In this case, the minimum occurrence of a keyword was set to 5, with both author and index keywords considered. This network map highlights the interconnectedness of various optimization techniques, showing how different methods and approaches are related. Colors represent different clusters, with node size indicating term frequency and edges showing co-occurrence associations. We can note three main clusters for EAs, namely "Genetic Algorithms"," Differential Evolution" and "Particle Swarm Optimization". Also, robust design with EAs is very commonly applied for power system stability, composite structures, problems that need simulations and engineering designs. Terms like “reliability-based design optimization,” “Taguchi methods,” “Monte Carlo simulation,” and “sensitivity analysis” also show up frequently. These methods are all used to handle uncertainty and variability in system design, with the goal of improving robustness, performance, and reliability in uncertain environments.

In Chapter 2, we analyzed the usage of EAs to bilevel optimization problems. We categorized these methods into four main approaches: single-level reduction, nested optimization, coevolutionary strategies, and surrogate-assisted methods. For min-max optimization problems, we can define a similar framework, structured into the following analogous categories:

- **Optimization of Scenario Space**: This corresponds to nested optimization, focusing on optimizing each design over the different scenarios.

[1]The search query used was: (“evolutionary algorithms” OR “evolutionary computation” OR “evolutionary optimization” OR “coevolutionary algorithms” OR “coevolutionary optimization” OR “genetic algorithms” OR “genetic programming” OR “differential evolution” OR “particle swarm optimization” OR “ant colony optimization”) AND (“min-max robust optimization” OR “robust design” OR “minimax optimization” OR “min-max optimization” OR “worst-case scenario optimization”).

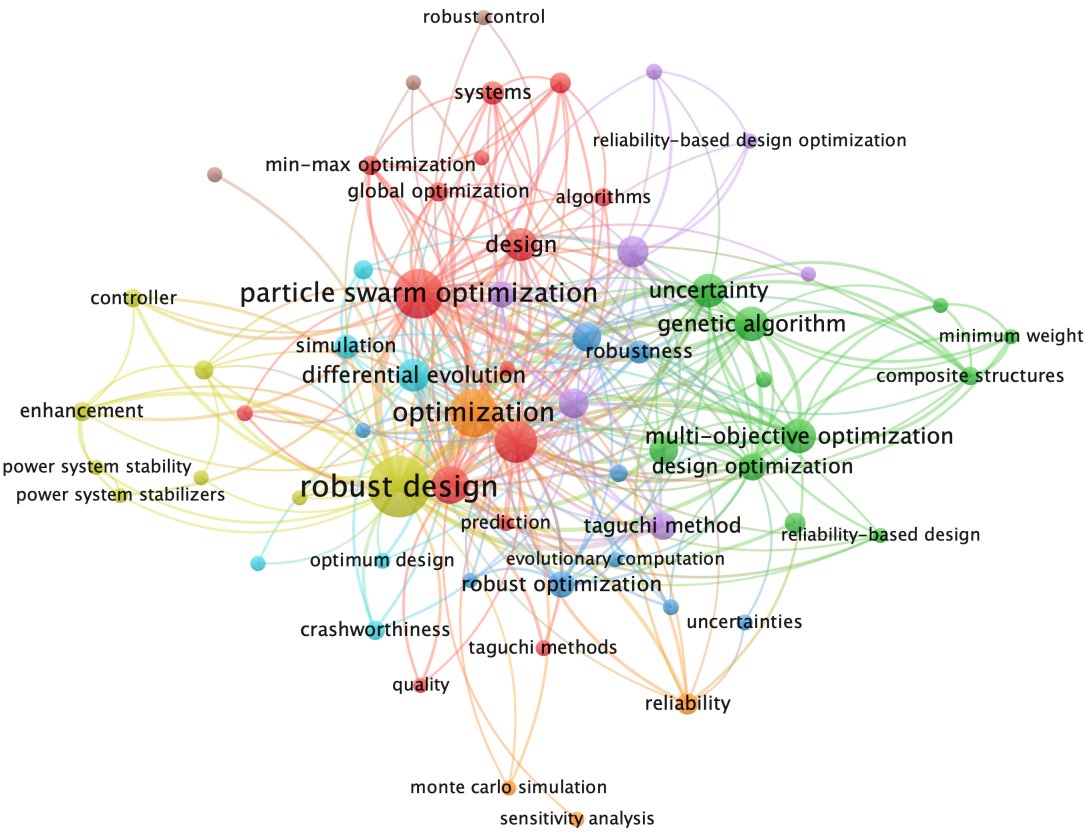


Figure 5.5: VOSviewer network visualization map of keyword co-occurrence. Colors represent different clusters, with node size indicating term frequency and edges showing co-occurrence relationships.

- **Coevolutionary Strategy**: This retains the same principles as in bilevel optimization, where the two populations of design and scenarios are co-evolving in parallel.
- **Sampling Scenarios**: This is similar to the single-level reduction approach, involving optimization over pre-sampled scenarios.
- **Surrogate Models**: This is the same as the surrogate-assisted approach in bilevel optimization, where objective functions or mappings are approximated by surrogate-sor meta-models.

These methods are visually summarized in Figure 5.6. In the following subsections, we discuss them further, with examples from the literature.

Methods Based on the Optimization of Scenario Space: These methods reflect a nested optimization approach. For each potential solution d, its performance is evaluated under all possible scenarios s. Specifically, this involves calculating the objective function value $f(d, s)$ for each scenario, testing how the solution performs under different conditions. This way, we determine the worst-case performance for each potential solution by identifying the maximum value of $f(d, s)$ across all scenarios. This step is crucial as it assesses how each solution performs in the most challenging scenario, thus evaluating its

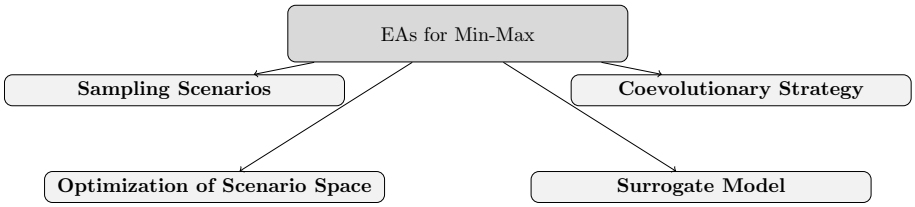


Figure 5.6: Categorization of Evolutionary Algorithms for Minimax Optimization Problems.

robustness. Finally, the method selects the optimal solution by minimizing this worst-case performance.

Evolutionary programming has been employed to design neural net controllers under uncertain conditions, with one of the first applications noted in 1994 [254]. This method used a population-based approach where neural net controllers were evolved through selection, mutation, and recombination. The evolutionary process was nested, focusing on optimizing the controller parameters in the outer loop and then evaluating these controllers across different plant models in the inner loop to minimize the maximum error observed. A modification of Particle Swarm Optimization (PSO) was adapted to address minimax problems, first presented in 2002 [255]. In this approach, PSO incorporated a nested loop where each particle's position (representing a potential solution) was evaluated across multiple scenarios. The algorithm iteratively adjusted the particles' velocities and positions to converge on the solution that minimized the worst-case scenario. A recent proposal introduced a two-phase differential evolution approach for minimax optimization [256]. The first phase focused on scenario exploration by refining selected individuals in the population, while the second phase emphasized solution exploitation, using an archive-based comparison strategy and dynamic resource allocation to balance exploration and exploitation effectively. This two-phase approach demonstrated its effectiveness in minimizing worst-case scenarios across various test functions and real-world applications.

Although these methods can be quite useful, they all share the common problem of high computational cost. The frequent bottleneck—usually the need to conduct extensive numbers of scenario evaluations due to increasing problem dimensionality—tends to impede scalability and practicability in real-time or large-scale applications. This fact has spurred interest in alternatives such as parallelization, surrogate modeling, and approximations.

Methods Based on Coevolutionary Strategy: Coevolutionary strategies involve the parallel evolution of solutions and scenarios. In these methods, solutions and scenarios evolve independently but interactively, often treating one as an evolving adversary to the other. Like in a competitive game, both sides in this dynamic interaction are constantly adjusting their strategy to counter the other's moves. The effectiveness of these strategies in min-max optimization lies in their ability to explore and exploit both the solution and scenario spaces simultaneously, offering robust solutions under varying conditions.

Evolve d (solutions) and s (scenarios) simultaneously with interaction

One of the earliest coevolutionary approaches for constrained optimization was introduced in 1999 by [13], employing a genetic algorithm with two co-evolving populations. Each population encoded values of variables, and fitness evaluation was jointly dependent on the evolutionary progress of both populations. This method reformulated constrained optimization problems into min-max problems through Lagrange multipliers, allowing the algorithm to identify feasible solutions that satisfy constraints. Later, [257] proposed a coevolutionary augmented Lagrangian method in 2000, also for constrained optimization.

Here, the problem was framed as a zero-sum game between a parameter vector and a Lagrange multiplier vector. The coevolutionary process approximated the game's saddle-point by evolving one population for the parameter vector and the other for the multiplier vector, with annealing techniques applied to avoid premature convergence. In 2002, coevolutionary methods for constrained optimization advanced with a coevolutionary particle swarm optimization (PSO) technique, incorporating an augmented Lagrangian approach [14]. This approach utilized two interconnected PSO populations: one for solution variables and the other for Lagrange multipliers, enhancing search efficiency by leveraging swarm intelligence. A Gaussian distribution-based coevolutionary PSO was introduced in 2006 [258], further refining the approach. Here, two PSO populations evolved the variable and Lagrange multiplier vectors, respectively, with Gaussian distribution applied to generate stochastic coefficients, thereby improving the algorithm's robustness against local optima. In 2012, a coevolutionary differential evolution algorithm was introduced for min-max optimization problems [259], utilizing parallel processing on graphical processing units (GPUs) with C-CUDA for simultaneous evaluation of multiple candidate solutions and scenarios, which significantly reduced computational time.

A significant challenge in coevolutionary strategies is the Red Queen effect[2] [260]. In the context of min-max optimization, the Red Queen effect describes the phenomenon where both the solution and scenario populations continually adapt to each other without making significant progress toward an optimal solution, that is very promiment on asymmetrical problems. As one population improves, the other must also evolve to counteract this improvement, potentially leading to an arms race that prevents convergence to a stable, optimal solution. This cyclic process of alternating between minimizing over variable d and maximizing over variable s is illustrated in Figure 5.7.

To address challenges in coevolutionary strategies, including those that can hinder progress, various researchers have proposed different methods. In 2001, a refinement was introduced to tackle symmetry in search spaces [239]. Previous approaches often assumed a symmetric condition between the solution space and the scenario space, leading to difficulties when the actual problem exhibited asymmetry. An asymmetric fitness evaluation mechanism was proposed, where the fitness of scenarios is determined not only by their performance against the current solution population but also by their ability to consistently challenge that population. In 2008, new variants of coevolutionary algorithms aimed at worst-case optimization were introduced [261]. Their research focused on test-based coevolutionary algorithms, where one population evolves solution candidates while the other evolves test cases designed to challenge these solutions. They introduced innovative fitness assignment methods such as *Average Greedy* and *Distance Greedy* to preserve crucial worst-case information during selection, significantly improving performance on benchmark problems. More recently, an efficient co-evolutionary minimax method was developed to optimize both control and noise factors in asymmetric minimax problems [262]. This approach integrates the efficient global optimization framework to efficiently identify the worst-case noise factors, balancing exploration and exploitation.

Despite many developments in coevolutionary methods, a prevalent issue shared with the previous method is the increased computational complexity, since the management and evolution of multiple populations can easily become unmanageable in high-dimensional or real-time applications. Exploration-exploitation trade-offs must be balanced while guaranteeing solution convergence, an open challenge for min-max optimization in different approaches.

Methods Based on Sampling Scenarios: The idea behind this approach requires

[2]The term "Red Queen effect" comes from Lewis Carroll's "Through the Looking-Glass" (1871), where the Red Queen says, "It takes all the running you can do, to keep in the same place."

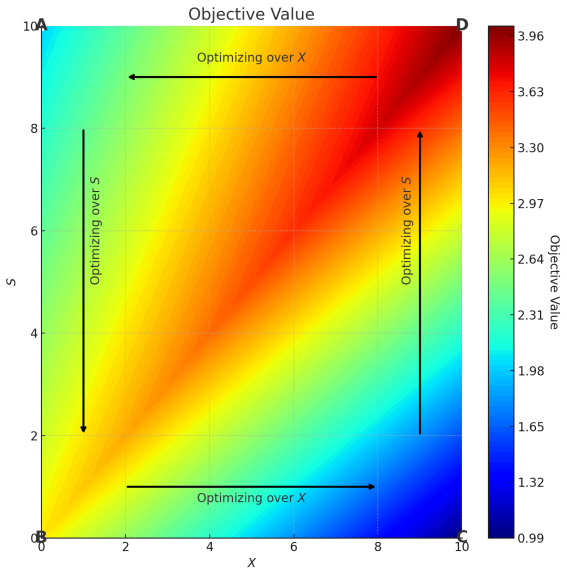


Figure 5.7: Illustration of the Red Queen Effect in min-max optimization.

sampling and evaluating a number of scenarios to guide the optimization process. Unlike coevolutionary methods that evolve both solutions and scenarios simultaneously or nested optimization methods that involve optimizing solutions and scenarios in interdependent loops, sampling scenario methods simplify the problem by generating a representative set of scenarios upfront. These scenarios are typically sampled from the scenario space based on some criteria such as importance or at random. After this sampling, the initial min-max problem is converted into a single-level optimization problem. Then the objective is to find the solution that performs optimally across these pre-sampled scenarios. This method considers the worst-case scenario as part of the sampled set, aiming to optimize the solution relative to this set.

$$\text{Optimize } d \text{ over a set of sampled scenarios } \{s_1, s_2, \ldots, s_n\}$$

The An Aging Sampled Genetic Algorithm (ASGA), which samples scenarios randomly [240] is an example of such an approach. In this case, the fitness of each solution is evaluated based on its performance against these sampled scenarios, recording the worst-case performance as its fitness. ASGA employes an aging mechanism, where the fitness of solutions is continually updated as they remain in the population, allowing older solutions to be tested against additional scenarios. This leads to a more accurate estimate of their worst-case performance. Moreover, a method that uses PSO to search for the design that minimizes the maximum potential inefficiency across a set of pre-sampled scenario which represent potential worst-case conditions was introduced in [263]. Both methods streamline the optimization process by limiting the problem to a single level, eliminating the complications of coevolutionary or nested methods.

Despite this simplification, sampling-based methods like ASGA and PSO may not always capture the true worst-case scenario, as the solution is only optimized over a pre-sampled set of scenarios. The effectiveness of these methods heavily depends on the quality and coverage of the sampled scenarios, and achieving robustness may require a large number of samples, leading to a potential higher computational cost or inaccuracy.

Methods Based on the Surrogate Model: Surrogate model methods use approximations instead of the real objective function in the design and/or in the scenario space to support the optimization process. These models are normally trained from a rather

small dataset and constructed employing methods like radial basis functions, polynomial regression, or Gaussian processes. Once trained, the surrogate model now acts as a proxy for the actual objective function, enabling quicker evaluations in its place for optimization. The optimization algorithm iteratively refines the surrogate model by adding new data points in regions where the model predicts high uncertainty or potential improvement. This iterative process gradually converges toward an optimal solution:

$$\text{Optimize } \hat{f}(x, s), \text{ the surrogate model, as an approximation to } f(x, s)$$

One surrogate-assisted evolutionary algorithm that utilizes a Gaussian process to approximate the relationship between decision variables and the objective function is presented in [264]. This method reduces the number of costly function evaluations by relying primarily on the surrogate model to evaluate most new solutions and only using the actual objective function for top-performing candidates. In [265], another approach applies an expected-improvement algorithm for continuous minimax optimization, also leveraging Gaussian processes as surrogate models. Here, the expected improvement criterion balances exploration and exploitation by selecting new points likely to refine the surrogate model's accuracy and efficiency. A max-min surrogate-assisted evolutionary algorithm designed for robust design problems introduces a Baldwinian trust-region framework, as outlined in [266]. This method builds local surrogate models with radial basis functions to approximate worst-case design performance, significantly lowering the number of high-fidelity evaluations needed for convergence. The trust-region strategy concentrates the search in regions with the greatest potential for improvement, making it particularly effective in high-dimensional design contexts. Besides, a surrogate-assisted evolutionary multitasking algorithm is developed for expensive minimax optimization tasks recently [242]. The proposed algorithm improves the efficiency and convergence of search by multitasking related optimization tasks and transferring knowledge across surrogate models.

Although surrogate model methods can reduce computational costs, their performance largely depends on the quality of the surrogate model itself. If the surrogate model provides only a coarse approximation of the true objective function—particularly in sparsely sampled regions—the optimization process may converge to a suboptimal solution. Additionally, maintaining and updating the surrogate model can introduce additional computational costs, especially in high-dimensional or simulation-based problems.

Other Methods: Beyond commonly used strategies such as scenario space optimization, coevolutionary techniques, sampling approaches, and surrogate models, additional approaches have been developed to address minimax optimization challenges.

One such method is a differential evolution algorithm designed for robust design in minimax optimization [267]. This approach incorporates a "bottom-boosting" scheme to enhance exploration and a "partial-regeneration" strategy to maintain population diversity. These features allow the algorithm to effectively identify promising solutions. Unlike surrogate models, this approach directly optimizes the objective function without relying on approximations, using an enhanced evolutionary framework to explore the search space while preserving diversity. However, it may still require a substantial number of function evaluations, and its effectiveness can vary based on problem-specific characteristics and parameter choices.

Another strategy that can drastically reduce computation time and, in effect, increase efficiency in the optimization process of evolutionary algorithms is parallel processing. This thesis' author has demonstrated in other work that parallel hierarchical differential evolution can speed up min-max optimization, since the scenarios can be evaluated in parallel [236]. Moreover, the following chapter introduces a nested differential evolution method with estimation of distribution [268], [269] for worst-case scenario optimization

[237]. This approach estimates the distribution of high-quality solutions from previous generations to efficiently initialize populations in the scenario space, further improving the convergence.

5.7 Summary

This chapter provided an overview of min-max optimization, covering fundamental concepts and the mathematical formulation. We explored how min-max optimization is essential for handling worst-case scenarios and achieving robust solutions, and reviewed various evolutionary algorithm approaches for tackling these problems.

In the next chapter, we introduce, the nested Differential Evolution with Estimation of Distribution Algorithm (MMNDE-EDA), an approach designed for solving min-max optimization. This approach integrates differential evolution with distribution estimation to enhance the search for optimal solutions, while using apriori knowledge acquired during the optimization process.

Chapter 6

Differential Evolution with Estimation of Distribution for Min-Max Optimization

Having examined the fundamental concepts and existing methods of min-max optimization in the previous chapter, this chapter introduces a novel approach that combines a Differential Evolution (DE) with an estimation of distribution algorithm (MMNDE-EDA). The objective is to minimize the maximum output across all scenarios, which typically requires extensive function evaluations. The proposed MMNDE-EDA algorithm features a nested structure, applying differential evolution to both design and scenario space optimizations. To enhance computational efficiency, the algorithm estimates the distribution of the best worst-case solutions found so far and utilizes this probabilistic model to sample part of the initial population for scenario space optimization. We empirically analyze how the probability of using this probabilistic model influences the algorithm's performance. Additionally, we compare this method with a state-of-the-art algorithm on benchmark problems and an engineering application, showcasing its efficiency. This chapter is based on [237].

Connection to Thesis Objectives

This chapter addresses O3.3, O4.2.

6.1 Problem Definition

Many optimization problems in the real world, especially within engineering design, may involve uncertainties in parameters, environmental conditions, or model inaccuracies. Considering these uncertainties is highly crucial to determine that solutions remain valid for a variety of conditions. Worst-case scenario optimization-that is, a particular approach to robust optimization-seeks solutions that perform reliably even in the most adverse conditions, providing a conservative yet resilient approach. For instance, in structural engineering, uncertainties in material properties or load estimates can impact a structure's safety and reliability. Worst-case optimization ensures that, even under maximum load or minimum material strength within expected bounds, the structure remains stable and secure

The content of this chapter have been published in:
M. Antoniou and G. Papa, "Differential evolution with estimation of distribution for worst-case scenario optimization," *Mathematics*, vol. 9, no. 17, p. 2137, 2021

[270]. In aerospace engineering, uncertainties in aerodynamics or system responses must be managed to maintain reliable aircraft performance. By optimizing for the worst-case scenario, engineers ensure that, even with variations in airflow or turbulence, the aircraft maintains stability and control [271]. Similarly, in finance, portfolio optimization under uncertainty often employs worst-case approaches to guard against extreme market downturns, resulting in a conservative but resilient portfolio that minimizes losses and promotes long-term stability [272].

Worst-case scenario optimization can be modeled as a min-max problem, where the goal is to find a solution that performs optimally in the face of the worst possible scenario within a specified uncertainty range. This formulation is a special case of bilevel optimization (BOP), where one optimization problem is embedded within the constraints of another [15], [273]. Here, the objective function is minimized in the design space while maximized in the uncertainty space, naturally supporting a hierarchical solution approach.

The rest of the chapter is organized as follows: Section 6.2 surveys the related work concerning the methods. Section 6.3 introduces definitions of worst-case scenario and min-max optimization. Section 6.4 briefly introduces general differential evolution (DE) and estimation of distribution (EDA) algorithm and details the proposed method thereafter. Parameter settings used in our experiments, as well as the discussion of results, can be found in Section 6.5. Among others, we investigate how the performance of the algorithm depends on the probabilistic model's usage probability. Moreover, the performance of the proposed method is compared to the performance of a state-of-the-art algorithm on benchmark problems and on an engineering application. This chapter is summarized in Section 6.6.

6.2 Related Work

Min-max optimization has been addressed using classical methods such as mathematical programming [274], branch-and-bound algorithms [275], and approximation methods [276]. These methods, however, possess restricted applicability, as they necessitate simplifying assumptions regarding the fitness function, including linearity and convexity. In recent years, evolutionary algorithms (EAs) have surfaced as effective methods for addressing min-max optimization challenges. EAs mitigate the need for specific assumptions about the underlying problem by employing population-based approaches that directly use the objectives. This allows them to handle mathematically intractable problems that do not follow specific mathematical properties [13], [14].

A popular EA-based approach for solving min-max problems is the co-evolutionary approach, where the populations of design and scenario spaces co-evolve. For instance, a co-evolutionary genetic algorithm was introduced in [13], and particle swarm optimization was used as an evolutionary strategy in [14]. While these methods perform well in symmetrical problems, they often struggle with asymmetrical problems due to the Red Queen effect, where populations continuously adapt without meaningful improvement [277]. These approaches are less suitable for many real-world problems due to this limitation.

Nested algorithms are particularly well-suited for min-max optimization and perform effectively on both symmetrical and asymmetrical problems [263]. Howeever, these methods may incur significant computational costs, since repeated scenario evaluations are needed. To reduce these costs, surrogate models have often been employed to approximate the objective function [264], [278]. While surrogate models help reduce the number of function evaluations, they can be less accurate, especially in high-dimensional or simulation-based problems where it is challenging to capture the objective function's landscape. This limitation may lead to suboptimal solutions.

In contrast, Estimation of Distribution Algorithms (EDAs) [268], [269] take a different approach by building probabilistic models of the decision space, rather than approximating the objective function directly [279]. By guiding the search toward regions with a higher likelihood of containing optimal solutions, EDAs can avoid some of the accuracy issues associated with surrogate models. EDAs have been used effectively in several optimization problems, such as multimodal [280], combinatorial [281], and multiobjective problems [282]. While EDAs avoid some of the limitations associated with surrogate models, their application to min-max optimization problems remains underexplored. This gap presents an opportunity to investigate whether EDAs, particularly in combination with Differential Evolution (DE), can enhance min-max optimization by reducing computational costs and improving search efficiency, especially in scenarios where accurately modeling the decision space is more beneficial than direct objective function approximation.

To address this research gap, we propose a DE-EDA hybrid algorithm for solving min-max optimization problems, called MMNDE-EDA. The algorithm employs a nested structure, with DE used for both the design and scenario space optimizations. A probabilistic model is constructed based on the best worst-case solutions identified at the upper level (UL), which then guides the search process in the lower level (LL). By integrating information from the upper level, the algorithm enhances the efficiency of the search process at the lower level, enabling a focused search within the more promising regions of the decision space. EDAs, by concentrating on the decision space, can improve both the accuracy and efficiency of the overall optimization process.

6.3 Mathematical Definition

In this section, we revisit the definitions of the classical optimization problem and the min-max optimization as a specific instance of robust optimization. Although these definitions have already been presented analytically in the previous Chapter 5, we include them here once more for clarity and ease of reference.

6.3.1 Definition of General Optimization Problem

A typical optimization problem is the problem of minimizing an objective over a set of decision variables subject to a set of constraints. The general mathematical form of an optimization problem is:

$$\begin{aligned} \min_{x \in X} \quad & f(x) \\ \text{subject to} \quad & g(x) \leq 0, \end{aligned} \tag{6.1}$$

where $x \in X$ is a decision vector of n dimension, $f(x)$ is the objective function and $g(x)$ are the inequality constraints. The global optimization techniques solve this problem, giving a deterministic optimal design. Usually, no uncertainties are considered. This approach, though widely used, is not very useful when a designer desires the optimal solution given the uncertainties of the system. Therefore, robust optimization approaches are applied [283].

6.3.2 Definition of Min-Max Optimization Problem

When one seeks the most robust solution under uncertainty, then the worst-case scenario approach is applied. Worst-case scenario optimization deals with minimizing the maximum output in the scenario space of a problem, and it is usually formulated as a min-max problem. The general worst-case scenario optimization problem in its min-max formulation is described as:

$$\min_{x \in X} \max_{y \in Y} f(x, y) \tag{6.2}$$

where $X \subseteq R^n$ represents the set of possible solutions and $Y \subseteq R^m$ the set of possible scenarios. As already discussed, the problem is a special instance of a BOP, where one optimization problem (the upper level) has another optimization problem in its constraints (the lower level). Here, the upper level and lower level share the same objective function $f(x, y)$, where upper level is optimizing with respect to the variables x of the design space and the lower level is optimizing with respect to the uncertain parameters y of the scenario space. If the upper-level problem is a minimization problem, then the worst-case scenario given by the uncertain variables y of a solution x can be found by maximizing $f(x, y)$. From now on, we will refer to the design space as upper level and scenario space as lower level interchangeably. The reader can find more about the BOPs in [15], [273] and in our previous Chapter 2.

An interesting property of min-max problems arises when the following symmetry condition holds:

$$\min_{x \in X} \max_{y \in Y} f(x, y) = \max_{y \in Y} \min_{x \in X} f(x, y).$$

When this condition is satisfied, the problem is symmetrical. Such symmetrical problems are generally simpler to solve, as the feasible regions for the upper and lower levels are independent. This independence reduces the complexity, allowing for more straightforward solution methods.

6.4 Algorithm Method: MMDE-EDA

In this section, we briefly describe the general schema of a estimation of distribution algorithm. For the description of DE, the reader can refer to Section 3.5.1, with a detailed explanation of the algorithm. The respective pseudocode can be found in the same chapter (see Algorithm 3.1) as well. Then, we explain the proposed algorithm.

6.4.1 Estimation of Distribution Algorithms (EDAs)

The basic flowchart of the EDA is shown in Figure 6.1, and the general steps of the EDA algorithm are outlined in Algorithm 6.1 as follows: The basic flowchart of the EDA is shown in Figure 6.1, and the general steps of the EDA algorithm are outlined in Algorithm 6.1 as follows:

1. Initialization: A population is initialized randomly.

2. Selection: The most promising individuals $S(t)$ from the population $P(t)$, where t is the current generation, are selected.

3. Estimation of the probabilistic distribution: A probabilistic model $M(t)$ is built from $S(t)$.

4. Generate new individuals: New candidate solutions are generated by sampling from $M(t)$.

5. Evaluation: The fitness of the new individuals is evaluated based on their objective values.

6. Create new population: The new solutions are incorporated into $P(t)$, and go to the next generation. The procedure ends when the termination criteria are met.

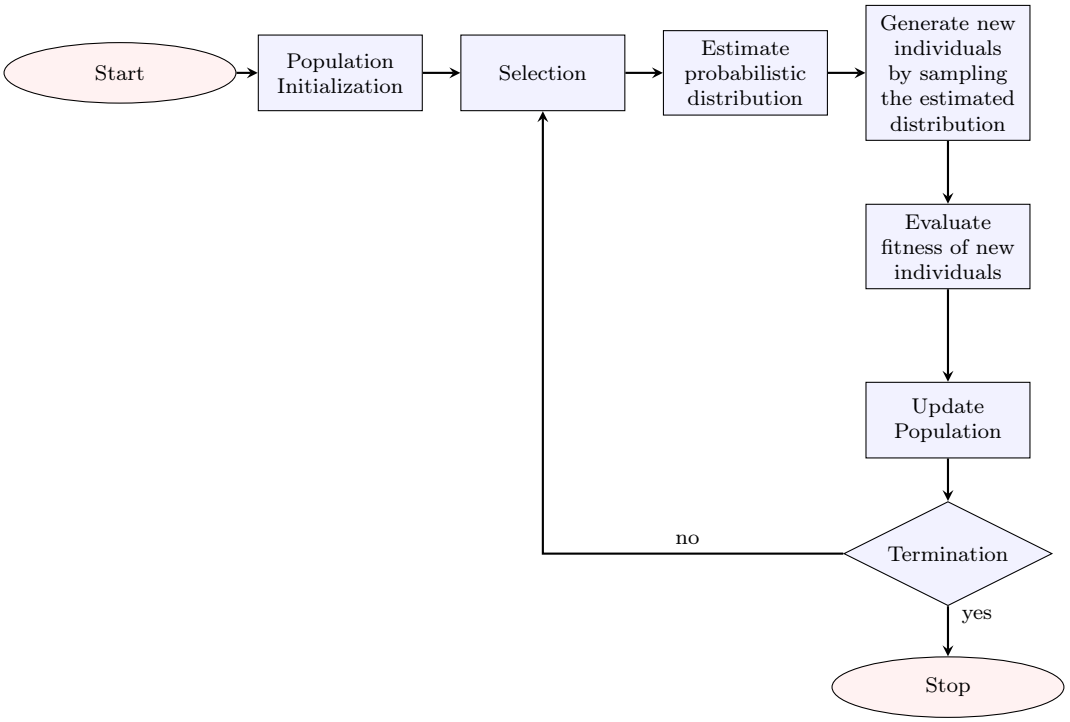


Figure 6.1: Basic flowchart of the estimation of distribution algorithm (EDA).

6.4.2 Proposed Algorithm

In the proposed algorithm, we keep the hierarchically nested formulation of a min-max problem. The design space (upper level) decision variables are evolving with a DE. For the evaluation of each upper level individual, first the scenario space (lower level) problem is solved by the DE. This solution is then transferred to the upper level. To reduce the cost, we apply an estimation of distribution mechanism between the decision space x and the scenario space search y. In that way, we use a priori knowledge obtained during the optimization. To further reduce the FEs, we search only for solutions with good worst-case scenarios. If the objective function of a solution x_1 under any scenario is already worse in terms of worst-case performance of the best solution x_{best} found so far, there is no need for further exploring x_1 over scenario space. Therefore, the mutant individual's performance

Algorithm 6.1: EDA

Input: Initial population size, stopping criteria
Output: Optimized population $P(g)$ and the best solution found
$g \leftarrow 0$
Generate initial population $P(0)$
while (stopping criteria not met) **do**
    Select population of promising solutions $S(g)$ from $P(g)$
    Build probabilistic model $M(g)$ from $S(g)$
    Sample $M(g)$ to generate new candidate solutions $O(g)$
    Evaluate fitness or objective values of new solutions in $O(g)$
    Incorporate $O(g)$ into $P(g)$ based on evaluation
    $g \leftarrow g + 1$
end while

is checked under the parent's worst-case scenario, and further explored only when it is better in terms of the fitness function. The general framework of the proposed approach is illustrated in Figure 6.2. For the upper level, the main steps are described in detail below and summarized in Algorithm 6.2.

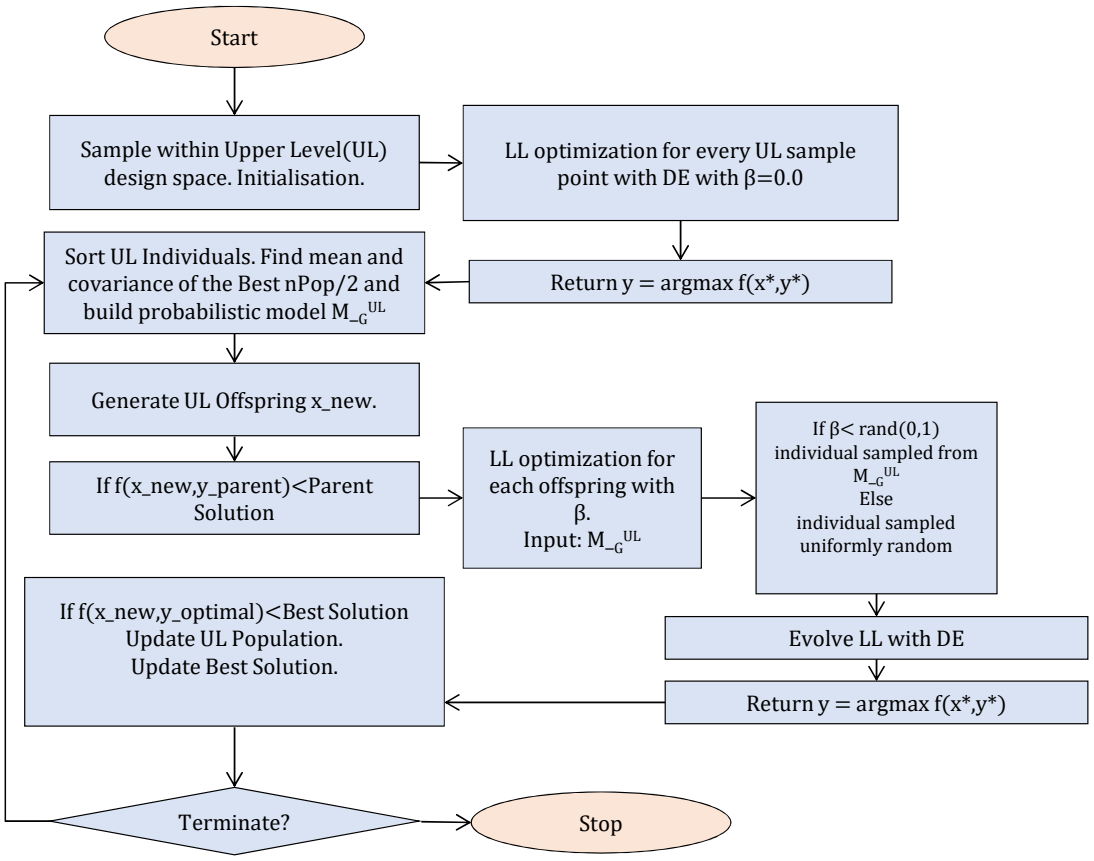


Figure 6.2: General framework of the proposed algorithm.

1. Initialization: A population of size NPop is initialized according to the general DE procedure mentioned in Section (3.5.1), where the individuals are representing candidate solutions in the design space X.

2. Evaluation: To evaluate the fitness function, we need to solve the problem in the scenario space. For a fixed candidate upper level solution X_i, the lower level DE is executed. More detailed steps are given in the next paragraphs. The lower level DE returns the solution corresponding to the worst-case scenario for the specific X_i. For each individual, the corresponding best $Y_{best} = \underset{y \in Y}{\operatorname{argmax}} f(X_i, y)$ solution is stored, meaning the solution y that aims to maximize the objective function $f(X_i, y)$ for a fixed x .

3. Probabilistic Model Building: The individuals in the population $P(i)$ are sorted in ascending order based on their upper-level fitness values. The best $\frac{\text{NPop}}{2}$ individuals are then selected. From these top $\frac{\text{NPop}}{2}$individuals, a distribution is built to construct the probabilistic model M_G for the lower-level solution. Note that $\frac{\text{NPop}}{2}$ is a method parameter, chosen by the user, representing the fraction of the population used for

constructing the probabilistic model and can be modified based on the optimization goals or problem requirements.

The d-dimensional multivariate normal density used to factorize the joint probability density function is given by:

$$F(x, \mu, \Sigma) = \frac{1}{\sqrt{(2\pi)^d |\Sigma|}} e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}, \tag{6.3}$$

where x is the d-dimensional random vector, μ is the d-dimensional mean vector, and Σ is the $d \times d$ covariance matrix. The two parameters, μ and Σ, are estimated from the top$\frac{\text{NPop}}{2}$ individuals in the population, based on their stored lower-level best solutions.

In this way, in each generation, we extract statistical information about the lower-level solutions of the previous upper-level population. The parameters are updated accordingly in each generation, following the general schema of an estimation of distribution algorithm.

4. Evolution: Evolve upper level with the steps of the standard DE of mutation, crossover, producing an offspring $U_{i,G}$.

5. Selection: As mentioned above, the selection operation is a competition between each individual $X_{i,G}$ and its offspring $U_{i,G}$. The offspring will be evaluated in the scenario space and sent in lower level only if $f(U_{i,G}, Y_{i,G}) \leq f(X_{i,G}, Y_{i,G})$, where $Y_{i,G}$ corresponds to the worst case vector of the parent individual $X_{i,G}$. In that way, a lot of unneeded lower level optimization calls can potentially be avoided, reducing FEs. If the offspring is evaluated in the scenario space, the selection procedure in Equation (3.49) is applied. Otherwise, the individual is kept in the population, and continue to evaluate the next individual.

6. Termination criteria:

   - Stop if the maximum number of function evaluations $MaxFEs$ is reached.
   - Stop if the improvement of the best objective value of the last $MaxImpGen$ generations is below a specific number.
   - Stop if the absolute difference of the best and the known true optimal objective value is below a specific number.

7. Output: the best worst case function value $f(x^*, y^*)$, the solution corresponding to the best worst-case scenario (x^*, y^*).

For the lower level, the pseudocode is shown in Algorithm 6.3. The main steps are outlined as follows:

1. Input: Receive the upper-level decision variable x, which serves as the fixed parameter for the lower-level optimization problem.

2. Parameter Setting: Set the parameters for the probability of crossover (CR), the population size ($nPop$), the mutation rate (F), and the sampling probability (β).

3. Population Initialization: Sample NPopindividuals to initialize the population. If $\beta \leq$ random$(0, 1)$, the individual is sampled from the probabilistic model M_G, constructed in the previous upper-level generation using Equation (6.3). The model is sampled using Matlab's built-in function `mvnrnd(mu, Sigma)`, which takes a mean vector

`mu` and covariance matrix `Sigma` as input and returns a random vector drawn from the multivariate normal distribution [284]. Otherwise, the individual is sampled uniformly in the scenario space according to Equation (3.46). Note that in the first upper-level generation no probabilistic model has been constructed yet. For subsequent generations, β can range between 0 and 1, where $\beta = 1$ means the population is sampled exclusively from the probabilistic model. This can potentially lead to the algorithm getting stuck in local optima and converging prematurely.

4. Mutation, crossover, and selection: Perform these operations following the standard DE procedure.

5. Termination criteria:

   - Stop if the maximum number of generations ($MaxGen$) is reached.
   - Stop if the absolute difference between the best solution and the known true optimal objective value is below a predefined threshold:

$$|f(x^*, y^*) - f_{\text{opt}}| < \epsilon,$$

6. Output: Return the maximum function value $f(x^*, y^*)$ and the solution corresponding to the worst-case scenario $y^* = \arg\max(f(x^*, y))$.

An example of an initial population generated with $\beta = 0.5$ is shown in Figure 6.3. Magenta asterisks represent the population generated by the probabilistic model M_G from the previous upper-level generation, while blue points are samples uniformly distributed in the search space. Figure 6.4 illustrates the effect of the probabilistic model on the initial population of the lower level for one of the test-functions we used in our analysis during optimization. As iterations progress, the lower-level population members sampled from the probabilistic distribution converge toward the promising area that maximizes the function. The zoomed subplot in each subfigure shows that these members of the population are closer to the global maximum compared to the randomly distributed members.

6.4.3 Computational Cost Analysis

In many optimization problems, particularly those involving computationally expensive objective functions, the dominant factor in the overall computational effort is the number of exact function evaluations, rather than the time complexity of evolutionary operators such as crossover, mutation, or model building. This is a standard assumption in the study of evolutionary algorithms, where the time required for a single function evaluation can range from seconds to hours, while the time consumed by internal algorithmic operations typically remains negligible in comparison. Therefore, the total computation time is considered directly proportional to the number of exact evaluations, rather than the complexity of the evolutionary algorithm itself [167].

In the nested DE approach, the upper-level DE iterates to minimize the objective, while each upper-level candidate solution triggers a lower-level DE to find the corresponding worst-case (maximized) solution. The number of iterations in both the upper and lower levels is typically defined as the product of the number of generations (gen) and the population size (pop). Thus, the total number of function evaluations can be expressed as:

$$N_{\text{total}} = (gen_{\text{u}} \times pop_{\text{u}}) \times (gen_{\text{l}} \times pop_{\text{l}}) \times C_f,$$

where gen_{u} and pop_{u} denote the number of generations and population size at the upper level, respectively, while gen_{l} and pop_{l} represent the same parameters at the lower level. Here, C_f represents the cost of a single function evaluation.

Algorithm 6.2: MMNDE-EDA Upper-Level Optimization

Input: Population size N_{Pop}, mutation factor F_{mut}, crossover rate C_r, maximum generations G_{max}
Output: X_{best}, Y_{best}
Initialize population $P(0)$ within bounds $[X_{min}, X_{max}]$
Evaluate each individual X_i by running the Lower-Level Optimization
Set X_{best} as the individual with the best worst-case scenario solution
$g \leftarrow 0$
while stopping criteria not met (e.g., $g < G_{max}$) **do**
  Sort individuals based on fitness in ascending order
  Select top $N_{Pop}/2$ individuals for distribution estimation
  Build probabilistic model $M(g) \sim \mathcal{N}(\mu_g, \Sigma_g)$ from the selected individuals
  Generate offspring via DE mutation and crossover
  for each offspring U_i **do**
    if fitness of U_i (with worst-case scenario of parent) is better than parent **then**
      Run lower level to evaluate U_i's worst-case scenario
      if fitness of U_i is better (min) than X_{best} **then**
        Update X_{best} and Y_{best}
      end if
    end if
  end for
  Incorporate offspring into $P(g)$
  $g \leftarrow g + 1$
end while

In the proposed MMDE-EDA variant, a probabilistic model is built using the upper-level best solutions found so far. This model guides the initialization of the lower-level DE, where individuals are sampled either according to the probabilistic model or randomly. Since this initialization occurs once per lower-level generation and does not affect the iterative DE search itself, the additional overhead remains constant. Therefore, the overall computational cost remains: $N_{\text{total}} = (gen_{\text{u}} \times pop_{\text{u}}) \times (gen_{\text{l}} \times pop_{\text{l}}) \times C_f$ indicating that the EDA-guided initialization does not alter the overall evaluation-driven cost compared to the standard nested DE approach. What may vary is the number of iterations required to achieve the same solution accuracy.

To mitigate the computational burden associated with nested evolutionary algorithms, parallelization has been widely adopted as an effective strategy. By distributing lower-level evaluations across multiple cores, parallelization significantly reduces runtime without compromising solution quality. In fact, our study on nested DE for min-max problems demonstrated the efficiency of this approach, showing reductions in computational time while maintaining robust performance, even as problem dimensionality increases [236]. While similar performance can be expected for the MMNDE-EDA framework, further research would be required to confirm its scalability and parallelization.

6.5 Numerical Experiments

6.5.1 Experimental Setup

In this section, we provide the parameter settings for our experiments.The performance of the proposed algorithm was tested on 13 benchmark problems of min-max optimization.

Algorithm 6.3: MMNDE-EDA Lower-Level Optimization

Input: Upper-level solution X_i, probabilistic model M_G (if available), population size, β, DE parameters
Initialize population $P(0)$ within bounds $[Y_{\min}, Y_{\max}]$
for each individual **do**
  if probabilistic model M_G exists and $\beta \leq \text{random}(0,1)$ **then**
    Sample individual from M_G: $y \sim \mathcal{N}(\mu, \Sigma)$
  else
    Generate individual uniformly: $y \sim \text{Uniform}(Y_{\min}, Y_{\max})$
  end if
end for
Evaluate each individual Y_i by computing fitness $f(X_i, Y_i)$
Set Y_{best} as the individual with the worst fitness
$g \leftarrow 0$
while stopping criteria not met **do**
  Perform DE mutation and crossover to generate offspring
  for each offspring U_i **do**
    if fitness $f(X_i, U_i)$ is better than $f(X_i, Y_{\text{best}})$ **then**
      Update Y_{best}
    end if
  end for
  Add offspring into $P(g)$
  $g \leftarrow g + 1$
end while
Output: Y_{best}, $f(X_i, Y_{\text{best}})$

The problems used are found collected in [278] along with their referenced optimal values. The reader can find the problems in Appendix B.

The parameter settings used for all the experiments of this study are shown in Table 6.1. The population size depends on the dimensionality of the problem, where for the upper level $\max(n_x + n_y, 5) * 2$ is used and for the lower level $\max(n_y, 5) * 2$ where n_x, n_y is the dimensionality of the upper level and lower level, respectively. All the simulations were

Table 6.1: Control parameters used in the reported results.

	Upper-Level	**Lower-Level**
Population size	$\max(n_x + n_y, 5) * 2$	$\max(n_y, 5) * 2$
Crossover	0.9	0.9
Mutation	uniformly (0.2, 0.8)	uniformly (0.2, 0.8)
Desired Accuracy	10e-5	10e-5
Maximum Number of Generations	-	10
Maximum Number of Function Evaluations	5000	-
Maximum Number of Improvement Generations	30	-
Least Improvement	10e-5	-

undertaken on an Intel (R) Core (TM) i7-7500 CPU @ 2.70 GHz, 16 GB of RAM, and the Windows 10 operating system. The code and the experiments were implemented and run in Matlab R2018b, without any parallelization.

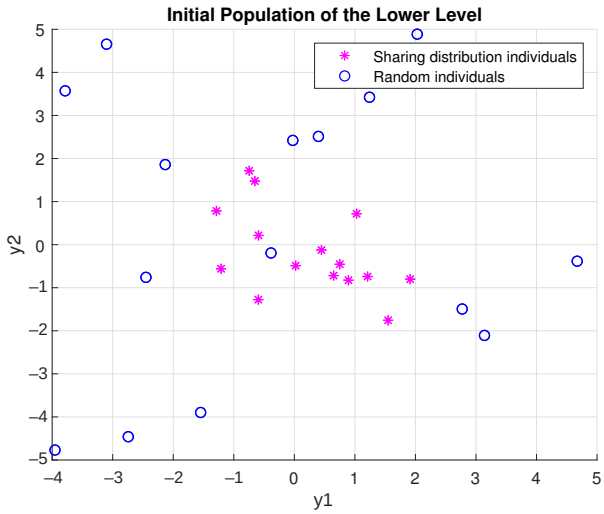


Figure 6.3: Balancing exploration and exploitation with the sharing distribution mechanism of the lower level population. Magenta asterisk points represent the population generated by the distribution of the previous upper level generations. Blue points are samples uniformly distributed in the search space. The idea behind this is to keep the "knowledge" already gained in previous generations while also giving the opportunity to the algorithm to search the whole search space. Here with $\beta = 0.5$.

6.5.2 Results on Effectiveness of the Probabilistic Sharing Mechanism

To evaluate the effectiveness of the probabilistic sharing mechanism of the proposed algorithm, we compare three different instances that correspond to three different β values. The first algorithmic instance has $\beta = 0$, meaning that the estimation of distribution in the optimization procedure is not activated, and the algorithm becomes a traditional nested DE. Therefore, this instance serves as the baseline. The second algorithmic instance corresponds to $\beta = 0.5$, where half of the initial population of the lower level is sampled from the probabilistic model. Last, for the third algorithmic instance, we set a value of $\beta = 0.8$, testing the ability of the algorithm when 80% of the initial population of the lower level is sampled from the probabilistic model.

Due to the inherent randomness of the EAs, repeated experiments are held to assess a statistical analysis of the performance of the algorithm. In Table 6.2, the statistical results of the 30 runs of the different instances of the algorithm are reported. More specifically, we report the mean, median, and standard deviation of the absolute error (AE) of the objective function. We calculate the AE as the absolute differences between the best objective function values provided by the algorithms and the known global optimal objective values of each test function. This is expressed as

$$\text{AE} = |f^{'} - f^{opt}| \tag{6.4}$$

where $f^{'}$ and f^{opt} are the best and the true optimal values, respectively.

Table 6.2: AE comparison of the different instances of the algorithm over the 30 runs.

Problems	**Type**		$\boldsymbol{\beta = 0}$	$\boldsymbol{\beta = 0.5}$	$\boldsymbol{\beta = 0.8}$
		Mean	3.45E-01	2.77E-05	2.29E-05
		Median	9.49E-02	**3.33E-05**	**3.33E-05**
f_1	Symmetrical				

Table 6.2 – continued from previous page

Problems	Type		$\beta = 0$	$\beta = 0.5$	$\beta = 0.8$
		Std	6.55E-01	3.43E-05	1.53E-05
		p-value	≤**0.05**	NA	>0.05
		Median FEs	20,115	**28,300**	46,535
		Mean	1.11E-01	1.25E-03	1.28E-04
		Median	4.96E-02	**5.53E-06**	**5.79E-06**
f_2	Symmetrical	Std	5.38E-01	4.27E-03	5.43E-04
		p-value	≤**0.05**	NA	>0.05
		Median FEs	20,665	**16,180**	17,140
		Mean	1.64E+00	2.27E-03	2.35E-02
		Median	9.51E-01	**1.86E-05**	**2.47E-05**
f_3	Asymmetrical	Std	2.29E+00	1.30E-02	8.35E-02
		p-value	≤**0.05**	NA	>0.05
		Median FEs	27,535	**39,830**	46,785
		Mean	3.49E-01	2.93E-05	1.64E-03
		Median	2.27E-01	**2.03E-05**	**3.39E-05**
f_4	Symmetrical	Std	4.49E-01	4.41E-05	8.88E-03
		p-value	≤**0.05**	NA	>0.05
		Median FEs	19,940	**26,478**	40,516
		Mean	6.23E-02	9.95E-04	8.43E-06
		Median	2.63E-02	**2.99E-04**	**8.55E-07**
f_5	Asymmetrical	Std	1.05E-01	4.18E-03	5.08E-05
		p-value	≤**0.05**	>0.05	NA
		Median FEs	38,694	**78,444**	97,506
		Mean	2.16E-01	1.96E-03	1.19E-02
		Median	1.62E-01	**7.86E-06**	**6.33E-06**
f_6	Symmetrical	Std	2.67E-01	6.74E-03	6.50E-02
		p-value	≤**0.05**	>0.05	NA
		Median FEs	55,740	**69,798**	77,356
		Mean	5.57E-01	7.90E-02	7.90E-02
		Median	4.76E-01	**7.90E-02**	**7.90E-02**
f_7	Asymmetrical	Std	3.75E-01	1.34E-04	9.54E-06
		p-value	≤**0.05**	NA	≤**0.05**
		Median FEs	143,580	**360,460**	541,940
		Mean	9.27E-06	3.03E-06	3.34E-06
		Median	6.16E-06	**1.17E-06**	**2.12E-06**
f_8	Symmetrical	Std	1.99E-05	3.24E-06	3.23E-06
		p-value	≤**0.05**	NA	>0.05
		Median FEs	9,120	8,150	**8,070**
		Mean	0.00E+00	0.00E+00	2.96E-03
		Median	**0.00E+00**	**0.00E+00**	**0.00E+00**
f_9	Asymmetrical	Std	0.00E+00	0.00E+00	1.62E-02
		p-value	NaN	NA	>0.05
		Median FEs	**3,435**	3,935	3,715
		Mean	7.54E-06	8.03E-08	8.74E-04
		Median	**2.86E-07**	2.98E-07	2.95E-07
f_{10}	Asymmetrical	Std	3.60E-05	4.96E-07	4.79E-03
		p-value	NA	>0.05	>0.05
		Median FEs	**4,435**	3,995	3,880
		Mean	5.11E-03	3.62E-03	1.43E-02
		Median	**1.79E-03**	**2.95E-04**	1.14E-02
f_{11}	Asymmetrical	Std	7.44E-03	8.06E-03	1.25E-02
		p-value	>0.05	NA	≤**0.05**
		Median FEs	**21,965**	30,480	33,485
		Mean	3.72E-01	5.21E-01	7.05E-01
		Median	**2.25E-01**	4.77E-01	7.43E-01
f_{12}	Asymmetrical	Std	5.98E-01	5.23E-01	1.42E-01
		p-value	NA	≤**0.05**	≤**0.05**
		Median FEs	**14,945**	15,795	28,210
		Mean	3.99E-02	2.42E-02	1.98E-01
		Median	6.51E-02	**1.82E-04**	**1.07E-05**
f_{13}	Asymmetrical	Std	7.29E-01	1.21E-01	1.04E+00
		p-value	≤**0.05**	>0.05	NA

Table 6.2 – continued from previous page

Problems	Type		$\beta = 0$	$\beta = 0.5$	$\beta = 0.8$
		Median FEs	22,430	**56,880**	61,525

In order to compare the instances, the non-parametric Wilcoxon signed-rank test [285] was conducted at the 5% significance level. For each test function, the instance with the best median AE was used as the control algorithm against the other two. A reported $\leq$0.05 value means that the null hypothesis is rejected, indicating that the two samples are statistically different, while >0.05 indicates no significant difference between them. The instance with the best median AE is shown in bold. Additionally, the median of the total number of function evaluations (FEs) is reported, with the lowest FEs corresponding to the best median AE also in bold.

The proposed method generally outperforms the baseline across most test functions. Specifically, the second and third instances perform significantly better than the baseline in test functions f_1 to f_8 and f_{13}. For these test functions, the results of instances 2 and 3 do not differ significantly (> 0.05), indicating a tie in terms of AE. However, instance 2 consistently requires fewer FEs to achieve similar results, making it more computationally efficient. For test function f_9, all instances perform equally in terms of median AE, with the baseline requiring the fewest FEs. The first instance shows superior performance for test functions f_{10} and f_{12}, and both the first and second instances outperform the third in f_{11}.

In several cases, the baseline instance completes with a low number of FEs, suggesting premature convergence when the "least improvement" termination criterion is triggered, ending the process before reaching the maximum FEs. In 11 out of the 13 test functions, instance 2 either outperforms at least one other instance or performs equally well, making the selection of $\beta = 0.5$ a safe choice.

The nature of symmetry in test functions appears to influence convergence, as the instances generally report fewer function evaluations for symmetrical functions f_1, f_2, f_4, f_6, and f_8 compared to the more challenging asymmetrical functions f_3, f_5, f_7, f_{11}, and f_{13}. This suggests that instance 2's balanced approach (with $\beta = 0.5$) may offer reliable performance across both symmetrical and asymmetrical problems. For f_{12} the FEs needed are lower, but the AE achieved is also quite higher, indicating again the premature convergence for all the instances.

In Figure 6.5, the success rate of each algorithmic instance and each test function is reported. As a success rate, we define the percentage of runs in which the algorithm achieved an absolute error (AE) below a predefined threshold out of the total runs. It is interesting to note that the baseline first instance did not reach the predefined AE threshold in 9 out of 13 test problems. The performance of the algorithm improves with the use of the estimation of distribution. On the other hand, the instance with $\beta = 0.5$ achieved the predefined AE threshold for at least one run in 11 out of 13 problems, while the instance with $\beta = 0.8$ did so in 10 out of 13 problems.

The second instance achieved the predefined AE threshold in 100% of the cases for asymmetrical functions f_9 and f_{10}. For test functions f_7 and f_{12}, none of the algorithms reached the predefined AE threshold within the given number of FEs. f_7 is one of the test problems with higher dimensionality, and a higher number of function evaluations might be needed in order to achieve a lower AE.

Figure 6.6 presents the convergence plots of upper-level AE for each algorithmic instance and test function. Red represents $\beta = 0.0$, magenta $\beta = 0.5$, and blue $\beta = 0.8$. The fluctuations in convergence are likely caused by inaccurate worst-case scenario solutions. In Figure 6.6g, the algorithm initially converges toward an optimal solution but later settles

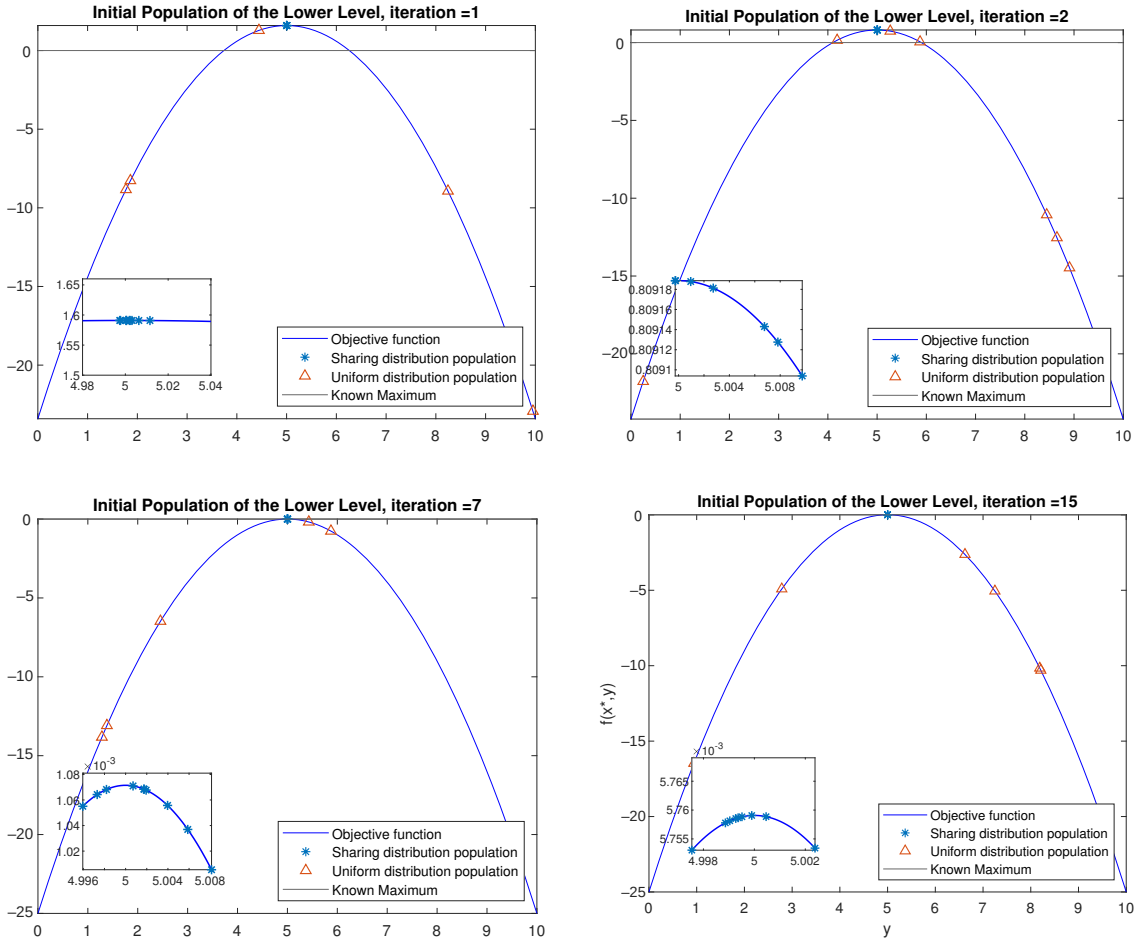


Figure 6.4: Effect of the probabilistic model on the initial population of lower level for f_8. As the iterations increase, the lower level members of the population that were produced from the probabilistic model reach the promising area that maximizes the function. The area they are concentrated in is shown in the zoomed plots of each plot.

on a slightly inaccurate one. In nested min-max optimization, once a suboptimal worst-case scenario is identified, the upper-level optimization minimizes based on this incorrect scenario. Since the worst case is not properly identified, the algorithm becomes trapped, leading to a solution that seems minimized but is actually inaccurate, which is typical in such problems. In Figure 6.6j for f_{10}, algorithmic instances 2 and 3 converge earlier than the baseline instance.

6.5.3 Results on Comparison with State-of-the-Art Method MMDE

In this subsection, we compare the proposed method with a state-of-the-art min-max EA. The MMDE [267] algorithm, which combines DE with a bottom-boosting scheme and a regeneration strategy, is specifically designed to detect robust worst-case solutions without the use of surrogate models. Similarly, our method is also non-surrogate-assisted and utilizes DE, making the comparison fair and direct. Among other min-max EAs, MMDE had shown better performance, making it a suitable choice for a baseline at the time of this study, and its author provided the code for this comparison. For the comparative

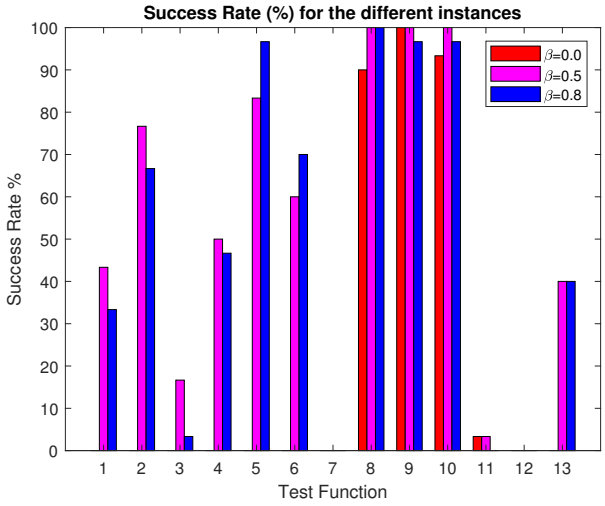


Figure 6.5: Barchart of the success rate (%) of each algorithmic instance and each test function. The red color corresponds to the instance where $\beta = 0.0$, magenta $\beta = 0.5$ and blue $\beta = 0.8$.

experiments, the DE parameters for the upper level in the proposed method are the same as in Table 6.1. For the lower level, the population size is set to $\max(n_y, 5) \times 3$, with $\beta = 0.5$. For MMDE, we used the parameter settings from the reference paper: crossover $CR = 0.5$, mutation $F = 0.7$, and two additional parameters, $K_s = 190$ and $T = 10$, controlling the number of FEs in the bottom-boosting and partial-regeneration strategies. To ensure a fair comparison, both algorithms are run with a total FE limit of 10^4.

Since the number of FEs is limited, an additional check was applied in the proposed method: if a new upper-level solution was already found in the previous population, it is not passed to the lower level, as the worst-case scenario is already known. The algorithms were each run 30 times on all test functions. For comparing the two methods, in addition to AE, we use the mean square error (MSE) of the obtained solutions in the design space (upper level) relative to the true optimum—a metric commonly used in min-max algorithm comparisons. The MSE is calculated as follows:

$$MSE(X_{best}, X_{opt}) = \frac{1}{D_X} \sum_{n=1}^{D_X} (x^n_{best} - x^n_{opt})^2 \tag{6.5}$$

where X_{best} is the best solution found by the algorithm, X_{opt} is the known optimal solution, and D_X is the dimensionality of the solution. In Tables 6.3 and 6.4, we report the mean, median, and standard deviation of the MSE and AE, respectively. The Wilcoxon signed-rank test [285] was conducted at the 5% significance level, and we indicate whether the p-value rejects the null hypothesis.

The proposed method demonstrates competitive performance in terms of MSE, outperforming MMDE on test functions f_1, f_2, f_8, and f_{10}, while achieving comparable performance on f_5 and f_9. However, MMDE performs better on the rest of the test functions. In terms of AE, the proposed method outperforms MMDE on f_1, f_2, f_6, and f_8, while achieving similar performance on f_4, f_9, and f_{10}. MMDE, however, performs better on the rest of the test functions.The differences between MSE and AE arise from the way each metric evaluates deviations from the true optimum. MSE penalizes larger deviations more heavily due to the squaring of errors, making it more sensitive to outliers. In contrast, AE provides a linear measure of deviation, reflecting the average distance without emphasiz-

ing extreme values. As a result, the proposed method's performance varies across the two metrics, depending on the distribution and scale of the errors for each test function.

Table 6.3: MSE Comparison with MMDE over 30 runs and 10^4 FEs.

Problems		**MMNDE-EDA**	**MMDE**
f_1	Mean	2.92E-10	5.79E-05
	Median	**2.92E-10**	5.79E-05
	Std	2.81E-10	6.78E-21
	p-value	NA	$\leq$**0.05**
f_2	Mean	1.80E-06	5.11E-06
	Median	**1.80E-06**	5.12E-06
	Std	8.19E-07	1.18E-07
	p-value	NA	$\leq$**0.05**
f_3	Mean	8.01E-07	7.49E-10
	Median	8.01E-07	**7.63E-10**
	Std	7.91E-07	6.99E-11
	p-value	$\leq$**0.05**	NA
f_4	Mean	7.57E-07	7.97E-10
	Median	7.57E-07	**7.84E-10**
	Std	4.99E-07	2.34E-11
	p-value	$\leq$**0.05**	NA
f_5	Mean	7.39E-10	9.31E-10
	Median	**1.22E-10**	**8.42E-10**
	Std	9.60E-10	3.46E-10
	p-value	NA	>0.05
f_6	Mean	1.59E-06	6.63E-08
	Median	1.53E-06	**3.33E-08**
	Std	1.28E-06	6.68E-08
	p-value	$\leq$**0.05**	NA
f_7	Mean	9.10E-04	7.82E-05
	Median	3.59E-04	**7.24E-05**
	Std	1.22E-03	2.27E-05
	p-value	<**0.05**	NA
f_8	Mean	2.0234E-05	2.6618E-05
	Median	**1.8269E-07**	9.5487E-06
	Std	7.5060E-05	6.1841E-05
	p-value	NA	$\leq$**0.05**
f_9	Mean	2.5849E-01	3.3719E-03
	Median	**0.0000E+00**	**0.0000E+00**
	Std	9.6251E-01	1.1081E-02
	p-value	NA	>0.05
f_{10}	Mean	1.0029E+00	5.1712E-01
	Median	**0.0000E+00**	0.0000E+00
	Std	2.8499E+00	2.4408E+00
	p-value	NA	$\leq$**0.05**
f_{11}	Mean	3.3428E-01	7.9495E-04
	Median	5.5485E-02	**8.8027E-05**
	Std	8.7588E-01	1.4876E-03
	p-value	$\leq$**0.05**	NA
f_{12}	Mean	8.1786E-03	1.1339E-05
	Median	9.6258E-05	**2.1344E-06**
	Std	1.9804E-02	3.2300E-05
	p-value	$\leq$**0.05**	NA
f_{13}	Mean	5.0537E-02	5.5425E-03
	Median	1.9716E-02	**2.7037E-03**
	Std	7.7093E-02	7.7943E-3
	p-value	$\leq$**0.05**	NA

Table 6.4: AE Comparison between MMNDE-EDA and MMDE over all runs.

Problems		**MMNDE-EDA**	**MMDE**
f_1	Mean	2.9293E-05	7.6100E-03
	Median	**3.3299E-05**	7.6100E-03
	Std	3.2970E-05	0
	p-value	NA	**≤0.05**
f_2	Mean	1.2511E-03	2.0000E-05
	Median	**6.0809E-06**	2.0000E-05
	Std	4.2708E-03	0
	p-value	NA	**≤0.05**
f_3	Mean	2.4970E-03	2.0000E-05
	Median	2.4663E-05	**2.0000E-05**
	Std	1.2971E-02	0
	p-value	**<0.05**	NA
f_4	Mean	3.3318E-05	3.4000E-05
	Median	**2.0266E-05**	**3.4000E-05**
	Std	4.1038E-05	0
	p-value	>0.05	NA
f_5	Mean	8.2556E-04	0
	Median	8.5470E-07	**0**
	Std	4.1597E-03	0
	p-value	**<0.05**	NA
f_6	Mean	1.9670E-03	3.0000E-05
	Median	**9.3202E-06**	3.0000E-05
	Std	6.7362E-03	0
	p-value	NA	**<0.05**
f_7	Mean	7.8994E-02	1.5766E-02
	Median	7.9024E-02	**1.5800E-02**
	Std	1.3407E-04	8.9105E-05
	p-value	**<0.05**	NA
f_8	Mean	3.0312E-06	2.2641E-27
	Median	1.1658E-06	**3.4868E-28**
	Std	3.2450E-06	4.7893E-27
	p-value	**<0.05**	NA
f_9	Mean	0	0
	Median	**0**	**0**
	Std	0	0
	p-value	>0.05	NA
f_{10}	Mean	3.7754E-07	3.0000E-07
	Median	**3.0162E-07**	**3.0000E-07**
	Std	3.2372E-07	0
	p-value	>0.05	NA
f_{11}	Mean	4.9814E-03	1.0000E-07
	Median	7.4905E-04	**1.0000E-07**
	Std	1.1319E-02	0
	p-value	**≤0.05**	NA
f_{12}	Mean	5.3333E-01	0
	Median	4.7689E-01	**0**
	Std	5.0920E-01	0
	p-value	**≤0.05**	NA
f_{13}	Mean	2.4249E-02	1.7513E-01
	Median	**1.8154E-04**	1.7515E-01
	Std	1.2134E-01	4.9770E-05
	p-value	NA	**≤0.05**

6.5.4 Results for Engineering Application

To further investigate the performance of the proposed method, we applied it to a simple engineering problem, also used as a benchmark in [265]. This problem involves the optimal design of a vibration absorber (Figure 6.7) for a structure subject to uncertainties in the forcing frequency. A structure with mass m_1 is subjected to a force of unknown frequency, while a smaller structure with mass m_2 is used to counteract oscillations through viscous damping. The design challenge is to make this damper robust to the worst-case force frequency. The objective function is the normalized maximum displacement of the main structure, expressed as [265]:

$$J = \frac{1}{Z}\sqrt{(1-\beta_{freq}^2/T^2 + 4(\zeta_2\beta_{freq}/T)^2} \tag{6.6}$$

where

$$\begin{aligned} Z^2 =& [\beta_{freq}^2(\beta_{freq}^2-1)/T^2 - \beta_{freq}^2(1+\mu) - 4\frac{\zeta_1\zeta_2\beta_{freq}^2}{T} + 1]^2 \\ &+ 4[\zeta_1\beta_{freq}^3/T^2 + \frac{\zeta_2\beta_{freq}^3(1-\mu) - \zeta_2\beta_{freq}}{T} - \zeta_1\beta_{freq}]^2 \end{aligned} \tag{6.7}$$

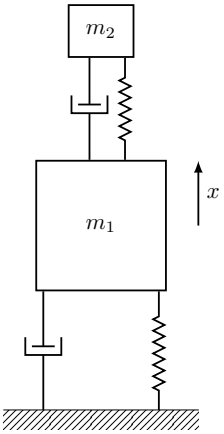


Figure 6.7: Vibration absorber.

The fixed parameters for this problem are $\mu = 0.1$ and $\zeta_1 = 0.1$. The decision variables in the design space are ζ_2 and T, while β_{freq} is the scenario-space variable against which the design should be robust. The problem can be formulated as a min-max problem:

$$\min_{x\in X}\max_{y\in Y} J(x,y) = \min_{\zeta_2,T}\max_{\beta_{freq}} J(\zeta_2, T, \beta_{freq}) \tag{6.8}$$

with $\zeta_2 \in [0,1]$, $T \in [0,1]$, and $\beta_{freq} \in [0, 2.5]$. The known solution of J is $x^* = (\zeta_2^*, T^*) = (0.1986, 0.8619)$ and $y^* = \beta_{freq}^* = 1.043$, with an optimal value approximated at $J(x^*, y^*) = 2.6227$, as reported in [286]. We ran the problem 30 independent times for both the proposed method and the MMDE algorithm, using the same parameter settings as in the previous subsection. In Table 6.5, we report the mean, median, and standard deviation of the obtained AE and MSE for both the proposed method and MMDE. Both algorithms approximate the known global optimum well, achieving an AE on the order of 10^{-2} and MSE on the order of 10^{-4}. The statistical test indicates that the proposed method performs equally well with MMDE on this engineering application.

Table 6.5: Statistical comparison with MMDE over 30 runs and 10^4 FEs for the engineering application.

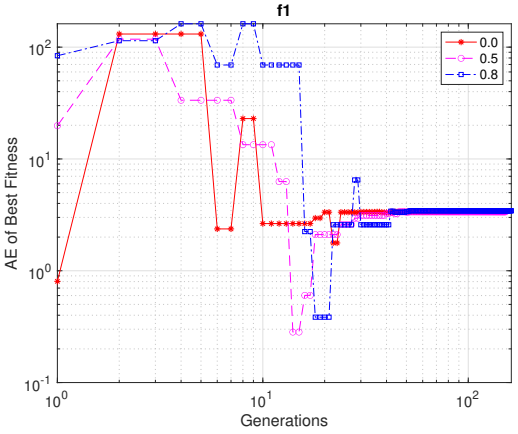
Problems		**MMNDE-EDA**	**MMDE**
$AE(J_{minmax})$	Mean	1.7100E-01	1.0472E-01
	Median	**6.4784E-02**	**9.9247E-02**
	Std	2.6084E-01	6.0458E-02
	p-value	NA	>0.05
$MSE(x)$	Mean	1.4668E-02	7.6533E-04
	Median	**6.4275E-04**	**3.6815E-04**
	Std	5.1909E-02	7.6206E-04
	p-value	>0.05	NA

6.6 Summary

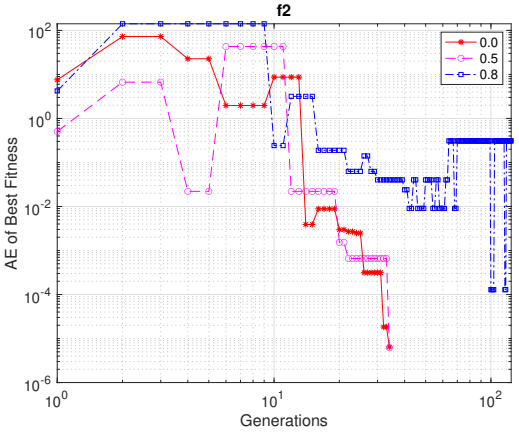
In this chapter, we presented an algorithm designed for solving min-max problems aimed at optimizing under worst-case scenario. The algorithm utilized a nested differential evolution combined with an estimation of distribution to improve efficiency in both accuracy and computational cost. A probabilistic model was constructed from the best worst-case solutions identified so far, which was then used to generate samples for the initial population of the lower-level DE, enhancing convergence speed. We evaluated the efficiency of the algorithm through experiments that compared the nested approach with varying probabilities of using the probabilistic model across 13 test functions of different dimensions and characteristics. Additionally, the proposed method was compared with a state-of-the-art algorithm recognized for its effectiveness on these problems, using both benchmark functions and an engineering application. The results demonstrated that, in most cases the proposed method is comparable to the state-of-the-art algorithm.

Connection to Thesis Hypothesis

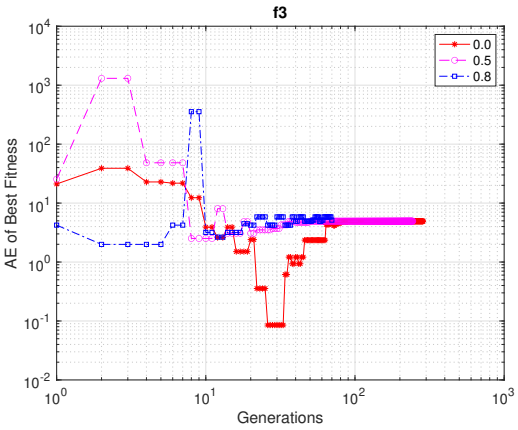
This chapter supports Hypothesis 3 by demonstrating through experiments that integrating a probabilistic estimation of distribution within nested evolutionary algorithms (EAs) generally improves perfomance in solving min-max problems for optimizing under worst-case scenarios in most cases, while maintaining solution quality.



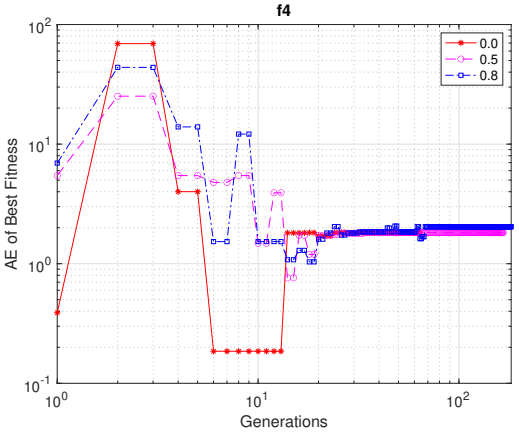
(a) Convergence for f_1



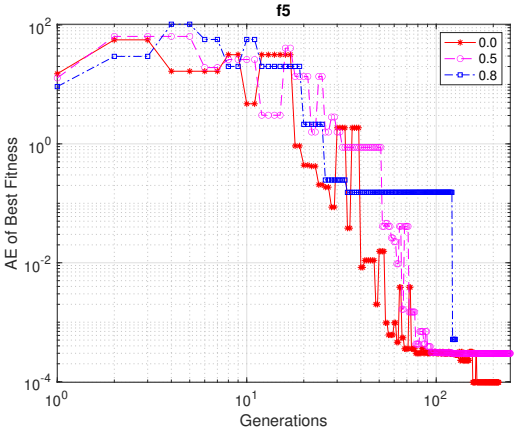
(b) Convergence for f_2



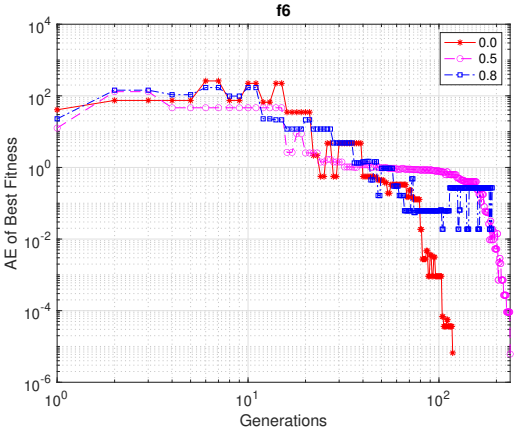
(c) Convergence for f_3



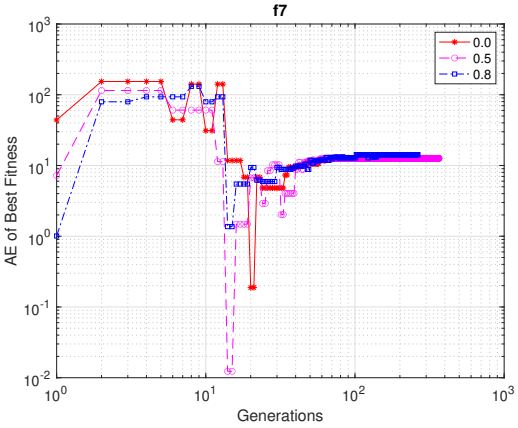
(d) Convergence for f_4



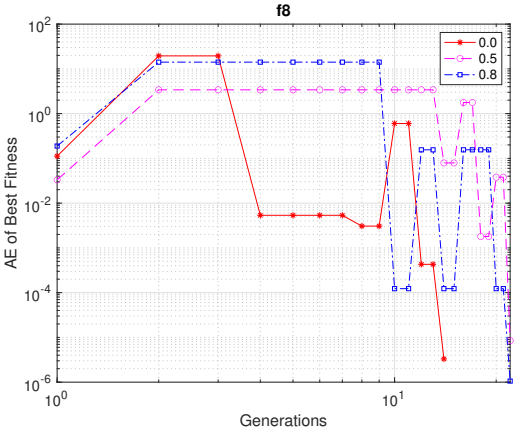
(e) Convergence for f_5



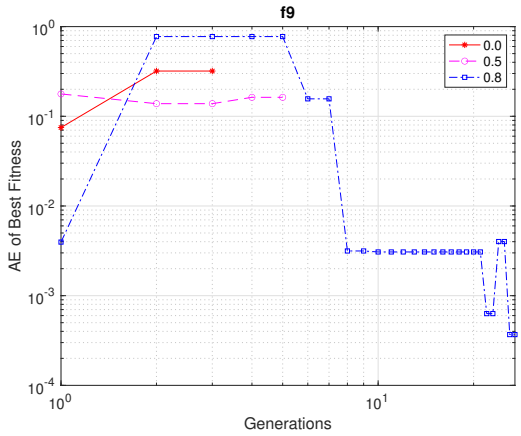
(f) Convergence for f_6



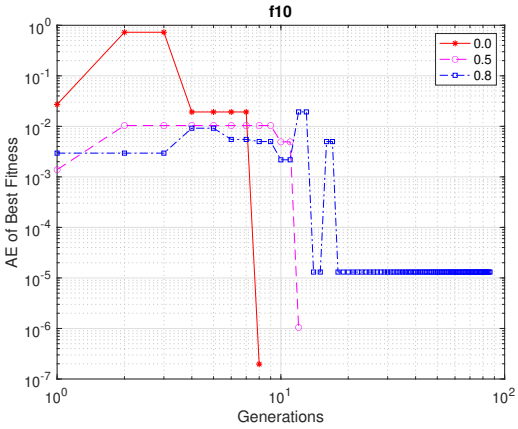
(g) Convergence for f_7



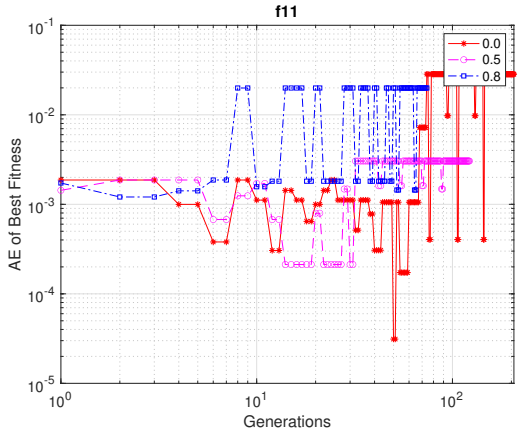
(h) Convergence for f_8



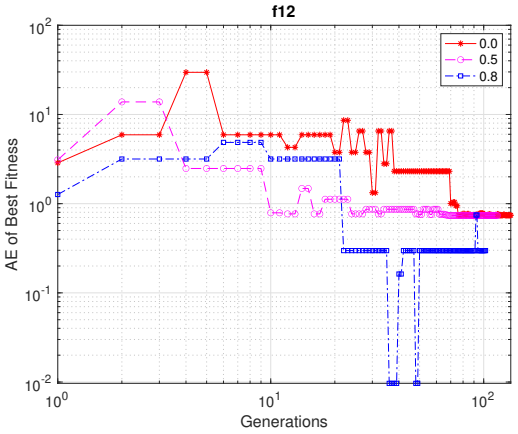
(i) Convergence for f_9



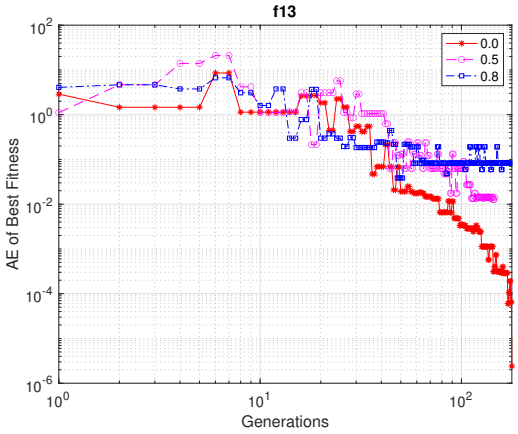
(j) Convergence for f_{10}



(k) Convergence for f_{11}



(l) Convergence for f_{12}



(m) Convergence for f_{13}

Figure 6.6: Fitness AE convergence of the upper level of the median run for all the test functions and algorithm instances. The red color corresponds to the instance where $\beta = 0.0$, magenta $\beta = 0.5$ and blue $\beta = 0.8$. Generations axes is in logarithmic scale.

Chapter 7

Conclusions and Future Work

7.1 Summary of the Thesis

This thesis suggests new methods to solve ill-posed bilevel problems, such as optimistic and pessimistic bilevel problems, as well as min-max problems using evolutionary algorithms. In Chapter 1, we introduced the background and motivation for the study, identified the research gap, and outlined the problem statement. We also provided an overview of the methodology and our hypotheses and our research objectives. Chapter 2 presented an overview of the bilevel optimization problem, detailing its connections to game theory, applications and mathematical definition. We examined optimistic and pessimistic formulations, supported by mathematical examples, and reviewed both classical and evolutionary solution methods. In Chapter 3, we introduced the δ-perturbation method, outlining its application to optimistic and pessimistic bilevel optimization problems. This chapter included mathematical definitions, the explanation of the perturbation method with an error bound analysis, followed by the algorithmic steps of the nested DE and numerical experiments that validated its effectiveness. Chapter 4 extended the δ-perturbation method to multiobjective bilevel optimization, providing a mathematical framework and tested to the existing algorithm m-BLEAQ. This chapter focused on investigating optimistic and pessimistic Pareto fronts, demonstrating the practicality of the proposed method through numerical experiments. Chapter 5 examined min-max optimization as a specific form of bilevel optimization aimed at minimizing the maximum output under worst-case scenarios. The chapter established key concepts and mathematical formulations while reviewing evolutionary algorithms tailored for min-max problems. In Chapter 6, we proposed a novel algorithm that integrates a probabilistic estimation of distribution within nested EAs (MMNDE-EDA) for solving min-max problems. This algorithm improved computational efficiency through a probabilistic model built from previous solutions. The experimental results showed that this strategy decreased computational costs while retaining solution quality in most cases.

This chapter is structured as follows: Section 7.2 covers the scientific contributions and validation of our hypotheses, while Section 7.3 outlines future work.

7.2 Scientific Contributions and Validation of Hypotheses

This thesis presents multiple scientific contributions, and the Bibliography section includes a list of all related publications. Each hypothesis is associated with one or more scientific contributions (SC) that are relevant to the scientific community. To contextualize the contributions, we restate the primary hypotheses addressed in this work:

H1: EAs can be enhanced with perturbation methods to approximate both optimistic and pessimistic solutions for ill-posed single-objective bilevel optimization problems.

H2: The integration of perturbation methods within EAs will enable them to effectively identify both optimistic and pessimistic Pareto Fronts for multiobjective bilevel optimization problems.

H3: Implementing probabilistic Estimation of Distribution within nested EAs can reduce the computational costs associated with solving min-max problems without sacrificing solution quality.

The following scientific contributions validate these hypotheses:

SC1: **Development of a perturbation method for bilevel optimization:** The perturbation method is designed to address ill-posed bilevel optimization problems caused by non-unique lower-level solutions. By employing the δ-perturbed approach, the formulation guarantees the discovery of a unique approximate lower-level optimal solution for both the optimistic and pessimistic cases. This validates **Hypothesis 1** by establishing a theoretical framework and providing proofs of the validity of the perturbation method in bilevel optimization. This research was presented at the *International Bilevel Optimization Conference* [287] and was later published as a peer-reviewed journal paper in *Operations Research Perspectives* [139].

SC2: **Extension of the perturbation method to multiobjective bilevel optimization:** The δ-perturbation method, initially developed for single-objective bilevel problems, was extended to handle multiobjective bilevel optimization problems, where the upper level has multiple objectives, and the lower level has a single objective but multiple optimal solutions. This contribution introduces a mathematical formulation that ensures the approximation of both the optimistic and pessimistic Pareto-optimal frontiers. This work validates **Hypothesis 2**. These findings have resulted in a paper accepted to the *International Conference on Evolutionary Multi-criterion Optimization (EMO)* [206].

SC3: **Validation of theoretical error bounds and the perturbation method:** Through rigorous computational studies, this work validates the theoretical error bounds associated with the perturbation method used in bilevel optimization. The empirical results confirm the validity of this method and demonstrate its effectiveness in handling ill-posed bilevel problems. This validation is discussed in [139] and validates **Hypothesis 1**.

SC4: **Establishing baseline evolutionary algorithms for optimistic and pessimistic bilevel optimization:** To our knowledge, this research represents the first application of EAs to approximate these specific classes of optimistic and pessimistic bilevel problems. It introduces baseline evolutionary algorithms designed for these classes, as detailed in [139] and [206]. This validates **Hypotheses 1 and 2**.

SC5: **Introduction of a new evolutionary approach for min-max optimization:** This research introduces a novel evolutionary algorithm that combines differential evolution with an estimation of distribution algorithm. It uses a nested structure to optimize both design and scenario spaces, with part of the scenario space population sampled based on prior knowledge from earlier iterations. Empirical results demonstrate that this approach outperforms its purely nested variant and performs comparably to state-of-the-art methods in engineering applications and benchmark

problems, as detailed in the journal paper published in *Mathematics* [237]. This work validates **Hypothesis 3**.

7.3 Future Work

This work opens several directions for future research in ill-posed single and multiobjective bilevel, and min-max optimization.

In ill-posed single-objective bilevel optimization, the nested perturbation approach presented here serves as a baseline for developing more advanced methods. For example, calculating the upper-level objective within the lower-level solution process can be computationally demanding. One staightforward direction is incorporating surrogate models, such as approximations based on $y - F(x, y)$, to reduce these costs. Additionally, a unified algorithm capable of tracking both optimistic and pessimistic solutions will provide a valuable tool for decision makers. Techniques such as evolutionary multitasking [288] and evolutionary transfer learning [289] can be a possible way forward. An interesting class where ill-posedness in real bilevel problems appears when the lower level is uncertain. Applying the method in such problems will showcase the real-world impact. Moreover, creating new benchmark functions for measuring and comparing the performance of evolutionary algorithms, as well as scaling existing ones, will give a more solid platform.

For multiobjective bilevel optimization, future work could focus on extending perturbation methods to handle multiobjective problems at both levels. This includes developing benchmark test functions for these cases, particularly for non-convex Pareto fronts, as well as exploring different value function approaches. Establishing error bounds and theoretical proofs for the proposed reformulation would further strengthen these methods.

In min-max optimization, several promising directions exist for extending the proposed approach. Testing the framework with various population-based evolutionary algorithms at either the upper or lower levels would demonstrate its adaptability across evolutionary strategies. Another key direction involves dynamically adapting the parameter β, which governs the probability of utilizing the probabilistic model, to balance exploration and exploitation for improved performance. The method could also be combined with surrogate models to further enhance efficiency, particularly in complex or high-dimensional problem settings. Extending this approach to constrained min-max problems would broaden its applicability, while investigating scalability on high-dimensional and large-scale engineering applications would provide valuable insights into its practical capabilities.

Finally, apart from the theoretical and algorithmic scope of this study, the broader motivation behind is to be applied to real-world applications and to give decision makers a tool to identify and analyze potential solutions to their problems at hand.

Appendix A

KKT Conditions and Example

KKT Conditions: The Karush-Kuhn-Tucker (KKT) conditions are essential for solving optimization problems with constraints, extending the method of Lagrange multipliers to include inequality constraints. They involve necessary conditions such as the gradients of the objective and constraint functions, feasibility conditions, and complementary slackness. These conditions are particularly useful in bilevel optimization, where they can be applied to the lower-level problem, transforming it into a set of constraints for the upper-level problem. This transformation simplifies the bilevel problem, especially when the lower-level problem is convex and smooth [114], [115].

For a bilevel optimization problem, we seek to minimize the upper-level objective function $F(x, y)$ subject to the constraint that y is the optimal solution of the lower-level problem $f(x, y)$. This leads to the following formulation:

Bilevel Optimization Problem (BOP):

$$\min_{x \in X, y \in Y} \quad F(x, y) \quad \text{(Upper-Level Problem)} \quad \text{s.t.} \quad y \in \arg\min_{y} f(x, y) \quad \text{(Lower-Level Problem)}$$

To solve this, we can apply the KKT conditions to the lower-level problem, which provides a set of necessary conditions that must be satisfied for optimality. The KKT conditions for the lower-level problem are:

KKT Conditions for the Lower-Level Problem:

$$\nabla_y f(x, y) + \sum_{i=1}^{p} \lambda_i \nabla_y g_i(x, y) + \sum_{j=1}^{q} \mu_j \nabla_y h_j(x, y) = 0$$

$$g_i(x, y) \leq 0, \quad i = 1, \ldots, p, \quad h_j(x, y) = 0, \quad j = 1, \ldots, q$$

$$\lambda_i \geq 0, \quad i = 1, \ldots, p \quad \text{and} \quad \lambda_i g_i(x, y) = 0, \quad i = 1, \ldots, p$$

Now, let's consider a specific example of bilevel optimization to see how these conditions apply.

Example of KKT Conditions in Bilevel Optimization:

Consider the following bilevel optimization problem:

$$\min_{x \in [0,4]} \quad F(x, y) = (x - 2)^2 + (y - 3)^2 \quad \text{s.t.} \quad y \in \arg\min_{y \in [0,4]} (y - x)^2 + 2y$$

We first solve the lower-level problem. The objective function for the lower-level problem is $f(x, y) = (y - x)^2 + 2y$. Taking the gradient with respect to y and setting it to zero gives:

$$2(y - x) + 2 = 0 \implies y = x - 1$$

Substituting $y = x - 1$ into the upper-level problem, we get:

$$F(x, x-1) = (x-2)^2 + (x-4)^2 = 2x^2 - 12x + 20$$

Now, we minimize the upper-level objective function:

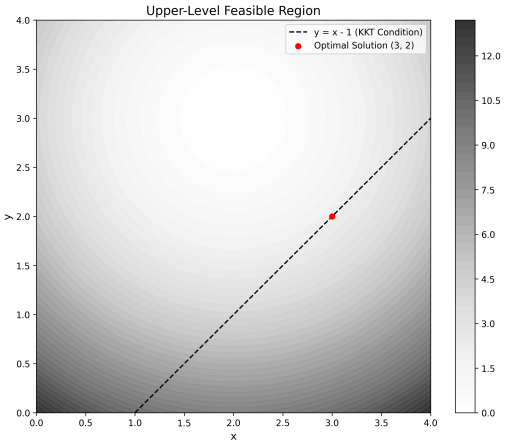
$$\min_{x \in [0,4]} 2x^2 - 12x + 20$$

The derivative of the objective function is $4x - 12 = 0$, which gives $x = 3$. We also check the values at the boundary points:

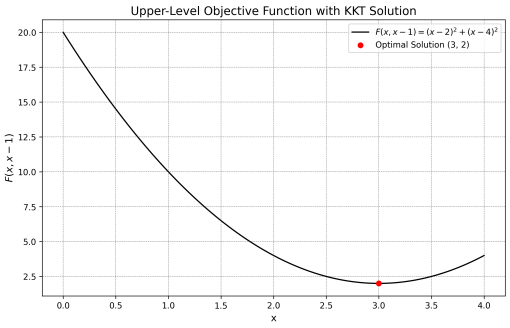
$$F(0, -1) = 20, \quad F(4, 3) = 4, \quad F(3, 2) = 2$$

Thus, the optimal solution is $x = 3$, with the corresponding $y = 2$.

The following figures illustrate the feasible region of the upper-level problem and the objective function:



(a) Upper-level feasible region with KKT condition line $y = x - 1$ and optimal solution.



(b) Upper-level objective function $F(x, x-1)$ with optimal solution.

Figure A.1: Visualizing the upper-level problem and optimal solution.

Appendix B

Test-Functions

B.1 Bilevel Test Functions

The test problems in this work are taken from [202] and [205]. Table B.1 presents these problems along with their optimistic and pessimistic global solutions.

Table B.1: Bilevel Test problems and their optimistic and pessimistic global solutions. [202], [205].

Problem	**Formulation**	**Best Known Solution**	
		Optimistic	**Pessimistic**
Problem 1 ($mb\_1\_1\_4$)	$\min\limits_{x,y} \quad y$ s.t. $y \in \underset{y}{\operatorname{argmin}}\{x(16y^4 + 2y^3 - 8y^2 - 3y/2 + 1/2)$ $-4/5 \leq y \leq 1\}$ $-1 \leq x \leq 1$	$F = -4/5$ $x = 0$ $y = -0.8$	$F = 1/2$ $x = (0, 1]$ $y = 0.5$
Problem 2 ($mb\_1\_1\_6$)	$\min\limits_{x,y} \quad x - y$ s.t. $y \in \underset{y}{\operatorname{argmin}}\{xy^2/2 - yx^3 :$ $-1 \leq y \leq 1\}$ $-1 \leq x \leq 1$	$F = -1$ $x = 0$ $y = 1$	$F = 0$ $(x, y) \in \{(-1, -1), (1, 1)\}$
Problem 3 ($mb\_1\_1\_11$)	$\min\limits_{x,y} \quad x^2 + y^2$ s.t. $y \in \underset{y}{\operatorname{argmin}}\{xy^2 - y^4/2 :$ $-1 \leq y \leq 1\}$ $-1 \leq x \leq 1$	$F = 1/4$ $x = 1/2$ $y = 0$	$F = 1/4 + a^2 + a$ $x = 1/2 + a$ $y = 0$
Problem 4 ($mb\_1\_1\_12$)	$\min\limits_{x,y} \quad xy - y + y^2/2$ s.t. $y \in \underset{y}{\operatorname{argmin}}\{-xy^2 + y^4/2 :$ $-1 \leq y \leq 1\}$ $-1 \leq x \leq 1$	$F = -0.258$ $x = 0.189$ $y = \sqrt{0.189}$	$F = 0$ $x = [-1, 0]$ $y = 0$
Problem 5 ($mb\_1\_1\_17$)	$\min\limits_{x,y} \quad x^2 - y$ s.t. $y \in \underset{y}{\operatorname{argmin}}\{[(y - 1 - x/10)^2 - x/2 - 1/2]^2 :$ $0 \leq y \leq 3\}$ $0 \leq x \leq 1$	$F = -1.755$ $x = 0.2106$ $y = 1.799$	$F = -0.2929$ $x = 0$ $y = 0.2929$
Principal Agent	$\max\limits_{x,y} \quad 5x_1 + 8x_2 + 4y_1 - y_2$ s.t. $y \in \underset{y}{\operatorname{argmax}}\{y_1 + y_2 :$ $y_1 + y_2 \leq 15 - x_1 - 2x_2, y_1, y_2 \geq 0\}$ $x_1 + x_2 \leq 6, x_1, x_2 \geq 0$	$F = 66$ $x = 6, 0$ $y = 9, 0$	$F = 45$ $x = 0, 6$ $y = 0, 3$

* where a is a small positive value

B.2 Min-Max Test Functions

The min-max test problems used are found collected in [278] along with their referenced optimal values. The first 7 problems f_1–f_7 are taken from [290] and they are convex in UL and concave in the LL. The problems are described in Table B.2. A stands for asymmetrical and S for symmetrical test problems.

Table B.2: Summary of Min-Max Test Functions f_1 to f_{13} [278], [290].

Problem	**Formulation**	x^*	y^*	$f(x^*, y^*)$
f_1, S	$\min_{x\in[-5,5]^2} \max_{y\in[-5,5]^2} 5(x_1^2+x_2^2)-(y_1^2+y_2^2)+x_1(-y_1+y_2+5)+x_2(y_1-y_2+3)$	(-0.4833, -0.3167)	(0.0833, -0.0833)	-1.6833
f_2, S	$\min_{x\in[-5,5]^2} \max_{y\in[-5,5]^2} 4(x_1-2)^2-2y_1^2+x_1^2y_1-y_2^2+2x_2^2y_2$	(1.6954, -0.0032)	(0.7186, -0.0001)	1.4039
f_3, A	$\min_{x\in[-5,5]^2} \max_{y\in[-3,3]^2} x_1^4y_2+2x_1^3y_1-x_2^2y_2(y_2-3)-2x_2(y_1-3)^2$	(-1.1807, 0.9128)	(2.0985, 2.666)	-2.4688
f_4, S	$\min_{x\in[-5,5]^2} \max_{y\in[-3,3]^3} -\sum_{i=1}^3(y_i-1)^2+\sum_{i=1}^2(x_i-1)^2+y_3(x_2-1)+y_1(x_1-1)+y_2x_1x_2$	(0.4181, 0.4181)	(0.709, 1.0874, 0.709)	-0.1348
f_5, A	$\min_{x\in[-5,5]^3} \max_{y\in[-1,1]^3} -(x_1-1)y_1-(x_2-2)y_2-(x_3-1)y_3+2x_1^2+3x_2^2+x_3^2-y_1^2-y_2^2-y_3^2$	(0.1111, 0.1538, 0.2)	(0.4444, 0.9231, 0.4)	1.3453
f_6, S	$\min_{x\in[-5,5]^4} \max_{y\in[-2,2]^3} -y_1(x_1^2-x_2+x_3-x_4+2)+y_2(-x_1+2x_2^2-x_3^2+2x_4+1)+y_3(2x_1-x_2+2x_3-x_4^2+5)+5x_1^2+4x_2^2+3x_3^2+2x_4^2-\sum_{i=1}^3 y_i^2$	(-0.2316, 0.2228, -0.6755, -0.0838)	(0.6195, 0.3535, 1.478)	4.543
f_7, A	$\min_{x\in[-5,5]^5} \max_{y\in[-3,3]^5} 2x_1x_5+3x_2x_4+x_3x_5+5x_4^2+5x_5^2-x_4(y_4-y_5-5)+x_5(y_4-y_5+3)+\sum_{i=1}^3 x_i(y_i^2-1)-\sum_{i=1}^5 y_i^2$	(1.4252, 1.6612, 1.2585, -0.9744, -0.7348)	(0.5156, 0.8798, 0.2919, 0.1198,-0.1198)	-6.3509
f_8, S	$\min_{x\in[0,10]} \max_{y\in[0,10]} (x_1-5)^2-(y_1-5)^2$	5	5	0
f_9, A	$\min_{x\in[0,10]} \max_{y\in[0,10]} \min\{3-0.2x_1+0.3y_1, 3+0.2x_1-0.1y_1\}$	0	0	3
f_{10}, A	$\min_{x\in[0,10]} \max_{y\in[0,10]} \frac{\sin(x_1-y_1)}{\sqrt{x_1^2+y_1^2}}$	10	2.1257	0.097794
f_{11}, A	$\min_{x\in[0,10]} \max_{y\in[0,10]} \frac{\cos(\sqrt{x_1^2+y_1^2})}{\sqrt{x_1^2+y_1^2}+10}$	7.0441	10 or 0	0.042488
f_{12}, A	$\min_{x\in[-0.5,0.5]\times[0,1]} \max_{y\in[0,10]^2} 100(x_2-x_1^2)^2+(1-x_1)^2-y_1(x_1+x_2^2)-y_2(x_1^2+x_2)$	(0.5, 0.25)	(0, 0)	0.25
f_{13}, A	$\min_{x\in[-1,3]^2} \max_{y\in[0,10]^2} (x_1-2)^2+(x_2-1)^2+y_1(x_1^2-x_2)+y_2(x_1-x_2-2)$	(1, 1)	any	1

Appendix C

Additional Research Contributions

During my PhD research years, I contributed in several projects and ideas, in the general area of optimization and evolutionary algorithms. Though these publications do not always exactly fit into the story of this thesis, they are worth to be mentioned.

Contributions closely related to the thesis topic

Solving pessimistic bilevel optimisation problems with evolutionary algorithms [291]

Abstract: It is common in practice, an optimal solution of a decision-maker to depend heavily on the response of another decision-maker, formulating a bilevel optimization problem. The optimization of a bilevel problem aims to achieve the optimum solution of the upper-level, taking into consideration the optimal lower-level values too. When the lower-level problem is multi-modal, meaning that it has several global optima, an ambiguity about the optimal upper-level solution appears. The optimistic approach assumes that the follower will respond with an optimal solution, that is favorable by the upper-level as well. In the pessimistic approach, the upper-level is optimising for the worst case. Various evolutionary algorithms have been implemented successfully to solve the optimistic approach of the bilevel problem. To the best of our knowledge, these algorithms have not been extended to the pessimistic approach. In this paper, we use a multi-population nested Differential Evolution to solve the pessimistic bilevel problem when the lower-level has multiple global optima. The performance of the algorithm is examined by solving a test-problem taken from the literature.

Connections to this thesis: One early attempt to solve both optimistic and pessimistic bilevel problem with a unified algorithm using DE.

Solving min-max optimisation problems by means of bilevel evolutionary algorithms: A preliminary study [238]

Abstract: Min-max optimisation is a special instance of a bilevel problem. It deals with the minimisation of the maximum output in all scenarios of a given problem. In this paper, numerical experiments are conducted to assess the accuracy and efficiency of three bilevel algorithms - known to perform well in general bilevel problems - on 13 unconstrained min-max test-functions. This study aims to bring the bilevel and min-max evolutionary community together and create a common ground for both optimisation problems.

Connections to this thesis: Investigating the connections between min-max and bilevel problems in the context of bilevel evolutionary algorithms.

Evaluation of Parallel Hierarchical Differential Evolution for Min-Max Optimization Problems Using SciPy [236]

Abstract: When optimization is applied in real-world applications, optimal solutions that do not take into account uncertainty are of limited value, since changes or disturbances in the input data may reduce the quality of the solution. One way to find a robust solution and consider uncertainty is to formulate the problem as a min-max optimization problem. Min-max optimization aims to identify solutions which remain feasible and of good quality under even the worst possible scenarios, i.e., realizations of the uncertain data, formulating a nested problem. Employing hierarchical evolutionary algorithms to solve the problem requires numerous function evaluations. Nevertheless, Evolutionary Algorithms can be easily parallelized. This work investigates a parallel model for differential evolution using SciPy, to solve general unconstrained min-max problems. A differential evolution is applied for both the design and scenario space optimization. To reduce the computational cost, the design level optimization is parallelized. The performance of the algorithm is evaluated for a different number of cores and different dimensionality of four benchmark test functions. The results show that, when the right parameters of the algorithm are selected, the parallelization can be of high benefit to a nested differential evolution.

Connections to this thesis: Investigating the effect of parallelization of the lower level in nested minmax DE on scalable minmax problems.

Satellite Scheduling Problem

Multi-objective approaches to ground station scheduling for optimization of communication with satellites [292]

Abstract: The ground station scheduling problem is a complex scheduling problem involving multiple objectives. Evolutionary techniques for multi-objective optimization are becoming popular among different fields, due to their effectiveness in obtaining a set of trade-off solutions. In contrast to some conventional methods, that aggregate the objectives into one weighted-sum objective function, multi-objective evolutionary algorithms manage to find a set of solutions in the Pareto-optimal front. Selecting one algorithm, however, for a specific problem adds additional challenge. In this paper the ground station scheduling problem was solved through six different evolutionary multi-objective algorithms, the NSGA-II, NSGA-III, SPEA2, GDE3, IBEA, and MOEA/D. The goal is to test their efficacy and performance to a number of benchmark static instances of the ground scheduling problem. Benchmark instances are of different sizes, allowing further testing of the behavior of the algorithms to different dimensionality of the problem. The solutions are compared to the recent solutions of a weighted-sum approach solved by the GA. The results show that all multi-objective algorithms manage to find as good solution as the weighted-sum, while giving more additional alternatives. The decomposition-based MOEA/D outperforms the rest of the algorithms for the specific problem in almost all aspects.

Connections to this thesis: Multiobjective optimization. Emprical tests and statistical comparisons of EAs.

On formulating the ground scheduling problem as a multi-objective bilevel problem. [293]

Abstract: In this paper, a bilevel multi-objective formulation of the Ground Scheduling Problem is presented. First, the problem is formulated as a bilevel optimisation problem (BOP), wherein the upper level (UL) is a biobjective problem determining the pairs of

Ground Station (GS) to Spacecraft (SC) and the starting time of each event with objectives the maximisation of the access windows and the minimisation of the communication clashes of each GS. These two objectives of the UL can be assumed as a measure of the violation of the feasibility of a schedule. The lower level (LL) consists of a single objective optimisation problem that determines the duration of each event, with objectives the communication time requirement of SCs with GS and the total ground station usage, combined together to a weighted sum function. The approach used to solve this multi-objective BOP is a nested approach, where the Pareto front of the upper level is obtained by a multi-objective optimisation algorithm (NSGA2) and the lower level is solved using a GA. The formulation is tested on one small test case from literature and the relevant results are reported.

Connections to this thesis: Detecting hierarchical structure in a real world problem, solving it as a multiobjective bilevel problem with EAs.

Preferred solutions of the ground station scheduling problem using NSGA-III with weighted reference points selection [294]

Abstract: The ground station scheduling problem is a complex scheduling problem involving multiple objectives. Evolutionary techniques for multi-objective optimization are becoming popular among different fields, due to their effectiveness in obtaining a set of trade-off solutions. In contrast to some conventional methods, that aggregate the objectives into one weighted-sum objective function, multi-objective evolutionary algorithms manage to find a set of solutions in the Pareto-optimal front. Selecting one algorithm, however, for a specific problem adds additional challenge. In this paper the ground station scheduling problem was solved through six different evolutionary multi-objective algorithms, the NSGA-II, NSGA-III, SPEA2, GDE3, IBEA, and MOEA/D. The goal is to test their efficacy and performance to a number of benchmark static instances of the ground scheduling problem. Benchmark instances are of different sizes, allowing further testing of the behavior of the algorithms to different dimensionality of the problem. The solutions are compared to the recent solutions of a weighted-sum approach solved by the GA. The results show that all multi-objective algorithms manage to find as good solution as the weighted-sum, while giving more additional alternatives. The decomposition-based MOEA/D outperforms the rest of the algorithms for the specific problem in almost all aspects.

Connections to this thesis: Multiobjective optimization. Emprical tests and statistical comparisons of EAs.

Dynamic Resource Allocation

Dynamic computational resource allocation for CFD simulations based on Pareto front optimization [295]

Abstract: Computational Fluid Dynamics (CFD) simulations can be extremely computationally demanding and usually rely on the use of High-performance computing (HPC) using both CPU and GPU resources. Modeling the behavior of bubbles size distribution often leads to a symmetric function evaluation problem. This paper proposes a dynamic computational resource allocation, based on Pareto-optimal solutions. The solutions are obtained from the formulation of the resource-constrained symmetric function evaluation problem as a multi-objective problem. After the Pareto-front is obtained, we suggest a dynamic selection method of the solutions that utilize the existing resources. To solve the multi-objective problem ϵ-MOEA, an algorithm known to obtain good diversity of Pareto-front solutions, is applied. As the problem formulated is new, brute-force search and two-specifically designed for this problem-heuristics are implemented and tested to

serve as baselines. The methods are tested and compared to three dimensionalities of the problem. The results showed that ϵ-MOEA can successfully approximate the Pareto-front, allowing to utilize the resources optimally at each simulation time-step.

Connections to this thesis: Multiobjective optimization and EAs.

Reviewer-Proposal Allocation (under submission)

Abstract: A critical task for journal editors and conference program chairs is to assign manuscripts to a set of reviewers. This problem is commonly referred to as a Reviewer Assignment Problem, and successfully finding good solutions can be a challenging task. While solving the problem is often performed manually, the task can be very time-consuming. Furthermore, it is expected that the complexity and time required to solve it increases with the number of reviewers and manuscripts. In this paper, we describe two novel approaches to solving a specific Reviewer Assignment Problem. Both approaches are evaluated on three datasets of different sizes. Constraints regarding reviewers' expertise, professional background, and demographics are considered, along with constraints related to the maximal and minimal amount of assignments per reviewer.

Connections to this thesis: Real world application of optimization.

Differential Evolution for Maximum Cuboid Extraction in Sustainable Stone Cutting: Application to a Granite Rock Example (submitted)

Abstract: Maximizing the extraction of usable volume in the form of a cuboid from polyhedra is a complex challenge in mining and stone-cutting optimization, with significant implications for sustainable resource utilization. This study addresses the problem by employing Differential Evolution (DE), a population-based evolutionary algorithm, to identify the axis-independent maximum cuboid within polyhedra. Using a granite rock as a test case, the algorithm iteratively refines solutions through mutation, crossover, and selection operators. To evaluate the behavior and performance across multiple runs, 20 independent runs were conducted, and the results are presented through convergence plots and snapshots illustrating the evolution of the cuboid solutions. The results demonstrate the potential of DE in solving these complex geometric optimization problems and contributing to sustainable mining practices.

Connections to this thesis: Real world application of optimization, first time using EAs for this kind of problem.

References

[1] P. Hansen, B. Jaumard, and G. Savard, "New branch-and-bound rules for linear bilevel programming," *SIAM Journal on Scientific and Statistical Computing*, vol. 13, no. 5, pp. 1194–1217, 1992.

[2] A. Ben-Tal and A. Nemirovski, "Robust convex optimization," *Mathematics of Operations Research*, vol. 23, no. 4, pp. 769–805, 1998.

[3] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*. Princeton University Press, 2009.

[4] G. Dantzig, *Linear Programming and Extensions*. Princeton University Press, 1963.

[5] J. Nocedal and S. Wright, *Numerical Optimization*, 2nd. Springer, 2006.

[6] J. Holland, *Adaptation in Natural and Artificial Systems*. University of Michigan Press, 1975.

[7] K. Deb, *Multi-Objective Optimization using Evolutionary Algorithms*. Wiley, 2001.

[8] J. S. Angelo, E. Krempser, and H. J. Barbosa, "Differential evolution for bilevel programming," in *2013 IEEE Congress on Evolutionary Computation*, IEEE, 2013, pp. 470–477.

[9] A. Sinha, P. Malo, and K. Deb, "Efficient evolutionary algorithm for single-objective bilevel optimization," *arXiv preprint arXiv:1303.3901*, 2013.

[10] M. M. Islam, H. K. Singh, T. Ray, and A. Sinha, "An enhanced memetic algorithm for single-objective bilevel optimization problems," *Evolutionary computation*, vol. 25, no. 4, pp. 607–642, 2017.

[11] A. Sinha, P. Malo, and K. Deb, "Evolutionary algorithm for bilevel optimization using approximations of the lower level optimal solution mapping," *European Journal of Operational Research*, vol. 257, no. 2, pp. 395–411, 2017.

[12] X. He, Y. Zhou, and Z. Chen, "Evolutionary bilevel optimization based on covariance matrix adaptation," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 2, pp. 258–272, 2018.

[13] H. J. Barbosa, "A coevolutionary genetic algorithm for constrained optimization," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, IEEE, vol. 3, 1999, pp. 1605–1611.

[14] Y. Shi and R. A. Krohling, "Co-evolutionary particle swarm optimization to solve min-max problems," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, IEEE, vol. 2, 2002, pp. 1682–1687.

[15] **M. Antoniou** and P. Korošec, "Multilevel optimisation," in *Optimization Under Uncertainty with Applications to Aerospace Engineering*, Springer, 2021, pp. 307–331.

[16] I. Newton, I. B. Cohen, and A. Whitman, *The Principia: mathematical principles of natural philosophy*. Univ of California Press, 1999.

[17] C. S. Roero, "Gottfried wilhelm leibniz, first three papers on the calculus (1684, 1686, 1693)," in *Landmark Writings in Western Mathematics 1640-1940*, Elsevier, 2005, pp. 46–58.

[18] S. Strickland, *A History of Mathematics*. Elsevier Academic Press, 2005, Discusses Fermat's role in developing optimization concepts.

[19] J. Lagrange, *Mecanique analytique* (Mecanique analytique τ. 1). Ve Courcier, 1811. [Online]. Available: `https://books.google.si/books?id=Q8MKAAAAYAAJ`.

[20] S. M. Stigler, *The History of Statistics: The Measurement of Uncertainty before 1900*. Cambridge, MA: Belknap Press of Harvard University Press, 1986, Discusses Gauss's development of the method of least squares.

[21] G. B. Dantzig, A. Orden, P. Wolfe, *et al.*, "The generalized simplex method for minimizing a linear form under linear inequality restraints," *Pacific Journal of Mathematics*, vol. 5, no. 2, pp. 183–195, 1955.

[22] W. Karush, "Minima of functions of several variables with inequalities as side conditions," Ph.D. dissertation, The University of Chicago, 1939.

[23] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[24] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2013, vol. 87.

[25] V. Gabrel, C. Murat, and A. Thiele, "Recent advances in robust optimization: An overview," *European journal of operational research*, vol. 235, no. 3, pp. 471–483, 2014.

[26] M. v. Stackelberg and E. Schnorrenberg, "Die struktur des aluminiumcarbids al4c3," *Zeitschrift fur Physikalische Chemie*, vol. 27, no. 1, pp. 37–49, 1934.

[27] J. Bracken and J. T. McGill, "Computer program for solving mathematical programs with nonlinear programs in the constraints," Institute for Defense Analyses, Alexandria, VA, Science and Technology Division, Tech. Rep., 1972.

[28] W. Candler and R. Norton, *Multi-level programming and development policy*. 1977.

[29] L. Vicente, G. Savard, and J. Júdice, "Descent approaches for quadratic bilevel programming," *Journal of Optimization theory and applications*, vol. 81, no. 2, pp. 379–399, 1994.

[30] R. Mathieu, L. Pittard, and G. Anandalingam, "Genetic algorithm based approach to bi-level linear programming," *RAIRO-Operations Research*, vol. 28, no. 1, pp. 1–21, 1994.

[31] A. Karoonsoontawong and S. T. Waller, "Dynamic continuous network design problem: Linear bilevel programming and metaheuristic approaches," *Transportation Research Record*, vol. 1964, no. 1, pp. 104–117, 2006.

[32] X. Li, P. Tian, and X. Min, "A hierarchical particle swarm optimization for solving bilevel programming problems," in *International Conference on Artificial Intelligence and Soft Computing*, Springer, 2006, pp. 1169–1178.

[33] M. D. Davis and S. J. Brams, *Game theory*, `https://www.britannica.com/science/game-theory`, Encyclopædia Britannica, Jul. 2024.

[34] H. v. Stackelberg, "Theory of the market economy," English, 1952. [Online]. Available: `http://agris.fao.org/agris-search/search.do?recordID=US201300604530` (visited on 05/04/2018).

[35] J. Lu, C. Shi, and G. Zhang, "On bilevel multi-follower decision making: General framework and solutions," *Information Sciences*, vol. 176, no. 11, pp. 1607–1627, 2006.

[36] J. Lu, J. Han, Y. Hu, and G. Zhang, "Multilevel decision-making: A survey," en, *Information Sciences*, vol. 346-347, pp. 463–487, Jun. 2016, ISSN: 00200255. DOI: 10.1016/j.ins.2016.01.084. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0020025516300202 (visited on 04/23/2018).

[37] G. Zhang, J. Lu, and Y. Gao, "Multi-level decision making," *Models, Methods and Applications*, 2015.

[38] A. Migdalas, "Bilevel programming in traffic planning: Models, methods and challenge," *Journal of global optimization*, vol. 7, pp. 381–405, 1995.

[39] A. Lonardi and C. De Bacco, "Bilevel optimization for traffic mitigation in optimal transport networks," *Physical Review Letters*, vol. 131, no. 26, p. 267 401, 2023.

[40] M. J. Santos, E. Curcio, P. Amorim, M. Carvalho, and A. Marques, "A bilevel approach for the collaborative transportation planning problem," *International Journal of Production Economics*, vol. 233, p. 108 004, 2021.

[41] J. Clegg, M. Smith, Y. Xiang, and R. Yarrow, "Bilevel programming applied to optimising urban transportation," *Transportation Research Part B: Methodological*, vol. 35, no. 1, pp. 41–70, 2001.

[42] M. Patriksson, "Robust bi-level optimization models in transportation science," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 366, no. 1872, pp. 1989–2004, 2008.

[43] E. Dovgan, M. Javorski, T. Tušar, M. Gams, and B. Filipič, "Discovering driving strategies with a multiobjective optimization algorithm," *Applied soft computing*, vol. 16, pp. 50–62, 2014.

[44] N. Adler, A. Brudner, and S. Proost, "A review of transport market modeling using game-theoretic principles," *European Journal of Operational Research*, vol. 291, no. 3, pp. 808–829, 2021.

[45] M. Labbé, P. Marcotte, and G. Savard, "A bilevel model of taxation and its application to optimal highway pricing," *Management science*, vol. 44, no. 12-part-1, pp. 1608–1622, 1998.

[46] R. Askin, F. Camacho, V. Kalashnikov, and N. Kalashnykova, "Comparison of algorithms for solving a bi-level toll setting problem," *Int. J. Innovative Comput. Inf. Control*, vol. 6, no. 8, pp. 3529–3549, 2010.

[47] C. Lombardi, L. Picado-Santos, and A. M. Annaswamy, "Model-based dynamic toll pricing: An overview," *Applied Sciences*, vol. 11, no. 11, p. 4778, 2021.

[48] M. J. Islam, X. Li, and K. Deb, "A speciation-based bilevel niching method for multimodal truss design problems," *Journal of Combinatorial Optimization*, pp. 1–35, 2022.

[49] F. Flager, A. Adya, J. Haymaker, and M. Fischer, "A bi-level hierarchical method for shape and member sizing optimization of steel truss structures," *Computers & Structures*, vol. 131, pp. 1–11, 2014.

[50] D. Liu, V. V. Toroporov, O. M. Querin, and D. C. Barton, "Bilevel optimization of blended composite wing panels," *Journal of aircraft*, vol. 48, no. 1, pp. 107–118, 2011.

[51] J. Herskovits, A. Leontiev, G. Dias, and G. Santos, "Contact shape optimization: A bilevel programming approach," *Structural and multidisciplinary optimization*, vol. 20, pp. 214–221, 2000.

[52] S. Christiansen, M. Patriksson, and L. Wynter, "Stochastic bilevel programming in structural optimization," *Structural and multidisciplinary optimization*, vol. 21, pp. 361–371, 2001.

[53] R. K. Wood, "Bilevel network interdiction models: Formulations and solutions," *Wiley encyclopedia of operations research and management science*, 2010.

[54] T. H. Bhuiyan, H. R. Medal, A. K. Nandi, and M. Halappanavar, "Risk-averse bilevel stochastic network interdiction model for cyber-security risk management," *International Journal of Critical Infrastructure Protection*, vol. 32, p. 100 408, 2021.

[55] G. Whittaker, R. Färe, S. Grosskopf, *et al.*, "Spatial targeting of agri-environmental policy using bilevel evolutionary optimization," *Omega*, vol. 66, pp. 15–27, 2017.

[56] M. Tanaka, Y. Chen, and A. S. Siddiqui, "Regulatory jurisdiction and policy coordination: A bi-level modeling approach for performance-based environmental policy," *Journal of the Operational Research Society*, vol. 73, no. 3, pp. 509–524, 2022.

[57] X.-Y. Bao and L. Zhang, "Green procurement relationships development under carbon emissions regulations: A bi-level programming approach," *International journal of environmental research and public health*, vol. 15, no. 10, p. 2183, 2018.

[58] E. G. Kardakos, C. K. Simoglou, and A. G. Bakirtzis, "Optimal offering strategy of a virtual power plant: A stochastic bi-level approach," *IEEE Transactions on Smart Grid*, vol. 7, no. 2, pp. 794–806, 2015.

[59] E. Fernandez, M. Hossain, K. Mahmud, M. S. H. Nizami, and M. Kashif, "A bi-level optimization-based community energy management system for optimal energy sharing and trading among peers," *Journal of Cleaner Production*, vol. 279, p. 123 254, 2021.

[60] Y. Ding and X. Wei, "Bi-level optimization model for regional energy system planning under demand response scenarios," *Journal of Cleaner Production*, vol. 323, p. 129 009, 2021.

[61] X. Luo, Y. Liu, and X. Liu, "Bi-level multi-objective optimization of design and subsidies for standalone hybrid renewable energy systems: A novel approach based on artificial neural network," *Journal of Building Engineering*, vol. 41, p. 102 744, 2021.

[62] A. Glotić and A. Zamuda, "Short-term combined economic and emission hydrothermal optimization by surrogate differential evolution," *Applied Energy*, vol. 141, pp. 42–56, 2015.

[63] J. Xu, Y. Tu, and Z. Zeng, "Bilevel optimization of regional water resources allocation problem under fuzzy random environment," *Journal of Water Resources Planning and Management*, vol. 139, no. 3, pp. 246–264, 2013.

[64] H. I. Calvete, C. Galé, J. A. Iranzo, and P. M. Mateo, "A decision tool based on bilevel optimization for the allocation of water resources in a hierarchical system," *International Transactions in Operational Research*, vol. 30, no. 4, pp. 1673–1702, 2023.

[65] W. Wang, H. Zhou, and H. Zhang, "Transboundary water resource allocation under drought emergencies: A multistage bilevel programming model considering decision-makers' risk perception," *Applied Soft Computing*, vol. 148, p. 110 888, 2023.

[66] S. Fang, P. Guo, M. Li, and L. Zhang, "Bilevel multiobjective programming applied to water resources allocation," *Mathematical Problems in Engineering*, vol. 2013, no. 1, p. 837 919, 2013.

[67] J. Xu, N. Ma, and C. Lv, "Dynamic equilibrium strategy for drought emergency temporary water transfer and allocation management," *Journal of Hydrology*, vol. 539, pp. 700–722, 2016.

[68] Y.-x. Chen, P. R. Tadikamalla, J. Shang, and Y. Song, "Supply allocation: Bi-level programming and differential evolution algorithm for natural disaster relief," *Cluster Computing*, vol. 23, pp. 203–217, 2020.

[69] S. Liu, C. Zhu, Y. Zheng, and Q. Wang, "Research on bi-level game scheduling for emergency rescue workers under the condition of limited rationality.," *Engineering Letters*, vol. 28, no. 4, 2020.

[70] G. Caselli, M. Iori, and I. Ljubić, *Bilevel optimization with sustainability perspective: A survey on applications*, 2024. arXiv: `2406.07184 [math.OC]`. [Online]. Available: `https://arxiv.org/abs/2406.07184`.

[71] S. Kongsomsaksakul, C. Yang, and A. Chen, "Shelter location-allocation model for flood evacuation planning," *Journal of the eastern Asia society for transportation studies*, vol. 6, pp. 4237–4252, 2005.

[72] W. Xu, Y. Ma, X. Zhao, Y. Li, L. Qin, and J. Du, "A comparison of scenario-based hybrid bilevel and multi-objective location-allocation models for earthquake emergency shelters: A case study in the central area of beijing, china," *International Journal of Geographical Information Science*, vol. 32, no. 2, pp. 236–256, 2018.

[73] M. Cecchini, J. Ecker, M. Kupferschmid, and R. Leitch, "Solving nonlinear principal-agent problems using bilevel programming," *European Journal of Operational Research*, vol. 230, no. 2, pp. 364–373, 2013.

[74] S. Dempe, "Computing optimal incentives via bilevel programming," *Optimization*, vol. 33, no. 1, pp. 29–42, 1995.

[75] Z. Zhu and B. Yu, "Globally convergent homotopy algorithm for solving the kkt systems to the principal-agent bilevel programming," *Optimization Methods and Software*, vol. 32, no. 1, pp. 69–85, 2017.

[76] A. Sinha, T. Khandait, and R. Mohanty, "A gradient-based bilevel optimization approach for tuning regularization hyperparameters," *Optimization Letters*, pp. 1–22, 2023.

[77] Y. Zhang, G. Zhang, P. Khanduri, M. Hong, S. Chang, and S. Liu, "Revisiting and advancing fast adversarial training through the lens of bi-level optimization," in *International Conference on Machine Learning*, PMLR, 2022, pp. 26 693–26 712.

[78] Y. Liu, H. Fan, X. Yuan, and J. Xiang, "Gl-gan: Adaptive global and local bilevel optimization for generative adversarial network," *Pattern Recognition*, vol. 123, p. 108 375, 2022.

[79] A. Sinha, P. Malo, P. Xu, and K. Deb, "A bilevel optimization approach to automated parameter tuning," in *Proceedings of the 2014 Annual Conference on Genetic and Evolutionary Computation*, 2014, pp. 847–854.

[80] **M. Antoniou**, P. Korošec, and G. Papa, "Parameter control in evolutionary bilevel optimisation," *Knjiga povzetkov: science of the future how to stay up-to-date with your research!*, p. 91, 2019, Nasl. z nasl. zaslona Opis vira z dne 22. 5. 2019. [Online]. Available: `http://ipssc.mps.si/Proceedings/Proceedings2019.pdf`.

[81] **M. Antoniou**, P. Korošec, and G. Papa, "An adaptive evolutionary surrogate-based approach for single-objective bilevel optimisation," *UQOP 2019, Uncertainty Quantification and OPtimization: 18-20 Marc, 2019, Paris (France)*, p. 2, 2019, Nasl. z nasl. zaslona Opis vira z dne 30. 8. 2019. [Online]. Available: `https://uqop.sciencesconf.org/242425/document`.

[82] K. P. Bennett, G. Kunapuli, J. Hu, and J.-S. Pang, "Bilevel optimization and machine learning," in *IEEE world congress on computational intelligence*, Springer, 2008, pp. 25–47.

[83] Y. Zhang, P. Khanduri, I. Tsaknakis, Y. Yao, M. Hong, and S. Liu, "An introduction to bilevel optimization: Foundations and applications in signal processing and machine learning," *IEEE Signal Processing Magazine*, vol. 41, no. 1, pp. 38–59, 2024.

[84] R. Liu, J. Gao, J. Zhang, D. Meng, and Z. Lin, "Investigating bi-level optimization for learning and vision from a unified perspective: A survey and beyond," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 10 045–10 067, 2021.

[85] W. Wiesemann, A. Tsoukalas, P.-M. Kleniati, and B. Rustem, "Pessimistic bilevel optimization," *SIAM Journal on Optimization*, vol. 23, no. 1, pp. 353–380, 2013.

[86] A. G. Mersha and S. Dempe, "Linear bilevel programming with upper level constraints depending on the lower level solution," *Applied mathematics and computation*, vol. 180, no. 1, pp. 247–254, 2006.

[87] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. New York, NY, USA: John Wiley & Sons, Inc., 2001, ISBN: 978-0-471-87339-6.

[88] S. Ruuska, K. Miettinen, and M. M. Wiecek, "Connections Between Single-Level and Bilevel Multiobjective Optimization," en, *Journal of Optimization Theory and Applications*, vol. 153, no. 1, pp. 60–74, Apr. 2012, ISSN: 0022-3239, 1573-2878. DOI: `10.1007/s10957-011-9943-y`. [Online]. Available: `http://link.springer.com/10.1007/s10957-011-9943-y` (visited on 08/08/2018).

[89] G. Ünlü, "A linear bilevel programming algorithm based on bicriteria programming," *Computers & operations research*, vol. 14, no. 2, pp. 173–179, 1987.

[90] E.-G. Talbi, Ed., *Metaheuristics for Bi-level Optimization* (Studies in Computational Intelligence), en. Berlin Heidelberg: Springer-Verlag, 2013, ISBN: 978-3-642-37837-9. [Online]. Available: `//www.springer.com/gp/book/9783642378379` (visited on 05/04/2018).

[91] J. Hadamard, *Lectures on Cauchy's Problem in Linear Partial Differential Equations*. Yale University Press, 1923.

[92] A. Wilczyński, A. Jakóbik, and J. Kołodziej, "Stackelberg security games: Models, applications and computational aspects," *Journal of Telecommunications and Information Technology*, no. 3, pp. 70–79, 2016.

[93] S. Dempe, V. Kalashnikov, G. A. Pérez-Valdés, and N. Kalashnykova, "Bilevel programming problems," *Energy Systems. Springer, Berlin*, vol. 10, pp. 978–3, 2015.

[94] B. Liu, Z. Wan, J. Chen, and G. Wang, "Optimality conditions for pessimistic semivectorial bilevel programming problems," *Journal of Inequalities and Applications*, vol. 2014, pp. 1–26, 2014.

[95] O. Ben-Ayed and C. E. Blair, "Computational difficulties of bilevel linear programming," *Operations Research*, vol. 38, no. 3, pp. 556–560, 1990.

[96] J. F. Bard, "Some properties of the bilevel programming problem," *Journal of optimization theory and applications*, vol. 68, no. 2, pp. 371–378, 1991.

[97] X. Deng, "Complexity issues in bilevel linear programming," in *Multilevel optimization: Algorithms and applications*, Springer, 1998, pp. 149–164.

[98] P. Marcotte and G. Savard, "Bilevel programming," 2001.

[99] J. F. Bard, *Practical bilevel optimization: algorithms and applications*. Springer Science & Business Media, 2013, vol. 30.

[100] B. Bhattacharjee, W. H. Green, and P. I. Barton, "Interval methods for semi-infinite programs," *Computational Optimization and applications*, vol. 30, pp. 63–93, 2005.

[101] B. Bhattacharjee, P. Lemonidis, W. H. Green Jr, and P. I. Barton, "Global solution of semi-infinite programs," *Mathematical Programming*, vol. 103, pp. 283–307, 2005.

[102] O. Stein and G. Still, "On generalized semi-infinite optimization and bilevel optimization," *European Journal of Operational Research*, vol. 142, no. 3, pp. 444–462, 2002.

[103] A. A. Khan, C. Tammer, and C. Zalinescu, *Set-valued optimization*. Springer, 2016.

[104] X. Kong, G. Yu, and W. Liu, "Optimality for set-valued optimization in the sense of vector and set criteria," *Journal of Inequalities and Applications*, vol. 2017, pp. 1–11, 2017.

[105] R. Liu, X. Liu, X. Yuan, S. Zeng, and J. Zhang, "A value-function-based interior-point method for non-convex bi-level optimization," in *International conference on machine learning*, PMLR, 2021, pp. 6882–6892.

[106] S. Dempe, "Annotated bibliography on bilevel programming and mathematical programs with equilibrium constraints," 2003.

[107] A. Valejo, M. C. Ferreira de Oliveira, G. P. R. Filho, and A. d. A. Lopes, "Multilevel approach for combinatorial optimization in bipartite network," *Knowledge-Based Systems*, Mar. 2018, ISSN: 0950-7051. DOI: 10.1016/j.knosys.2018.03.021. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0950705118301539 (visited on 04/23/2018).

[108] W. W. Hager, Ed., *Multiscale optimization methods and applications* (Nonconvex optimization and its applications v. 82), en. New York: Springer, 2006, ISBN: 978-0-387-29549-7 978-0-387-29550-3.

[109] P. Korošec, J. Šilc, and B. Robič, "Solving the mesh-partitioning problem with an ant-colony algorithm," en, *Parallel Computing*, vol. 30, no. 5-6, pp. 785–801, May 2004, ISSN: 01678191. DOI: 10.1016/j.parco.2003.12.016. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S0167819104000432 (visited on 05/04/2018).

[110] M. G. Fernández-Godino, C. Park, N.-H. Kim, and R. T. Haftka, "Review of multi-fidelity models," *arXiv preprint arXiv:1609.07196*, 2016.

[111] J. A. Monschke and M. S. Eldred, "Multilevel-multifidelity acceleration of pde-constrained optimization," in *58th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, 2017, p. 0132.

[112] I. C. Kampolis and K. C. Giannakoglou, "A multilevel approach to single- and multiobjective aerodynamic optimization," *Computer Methods in Applied Mechanics and Engineering*, vol. 197, no. 33, pp. 2963–2975, Jun. 2008, ISSN: 0045-7825. DOI: 10.1016/j.cma.2008.01.015. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0045782508000388 (visited on 03/05/2018).

[113] C. Walshaw, "Multilevel Refinement for Combinatorial Optimisation Problems," en, *Annals of Operations Research*, vol. 131, no. 1-4, pp. 325–372, Oct. 2004, ISSN: 0254-5330, 1572-9338. DOI: 10.1023/B:ANOR.0000039525.80601.15. [Online]. Available: https://link.springer.com/article/10.1023/B:ANOR.0000039525.80601.15 (visited on 05/08/2018).

[114] G. B. Allende and G. Still, "Solving bilevel programs with the kkt-approach," *Mathematical programming*, vol. 138, pp. 309–332, 2013.

[115] S. Dempe and A. B. Zemkoho, "The bilevel programming problem: Reformulations, constraint qualifications and optimality conditions," *Mathematical Programming*, vol. 138, pp. 447–473, 2013.

[116] C. Shi, J. Lu, and G. Zhang, "An extended kuhn–tucker approach for linear bilevel programming," *Applied Mathematics and Computation*, vol. 162, no. 1, pp. 51–63, 2005.

[117] E. Roghanian, M.-B. Aryanezhad, and S. J. Sadjadi, "Integrating goal programming, kuhn–tucker conditions, and penalty function approaches to solve linear bi-level programming problems," *Applied Mathematics and Computation*, vol. 195, no. 2, pp. 585–590, 2008.

[118] C. Li and L. Guo, "A single-level reformulation of mixed integer bilevel programming problems," *Operations Research Letters*, vol. 45, no. 1, pp. 1–5, 2017.

[119] M. Solodov, "An explicit descent method for bilevel convex optimization," *Journal of Convex Analysis*, vol. 14, no. 2, p. 227, 2007.

[120] J. Liu, T. Zhang, Y.-X. Fan, B. Han, and Y. Zheng, "An objective penalty method for optimistic bilevel programming problems," *Journal of the Operations Research Society of China*, vol. 8, pp. 177–187, 2020.

[121] P. Marcotte and D. L. Zhu, "Exact and inexact penalty methods for the generalized bilevel programming problem," *Mathematical Programming*, vol. 74, pp. 141–157, 1996.

[122] J. Ye, D. Zhu, and Q. J. Zhu, "Exact penalization and necessary optimality conditions for generalized bilevel programming problems," *SIAM Journal on optimization*, vol. 7, no. 2, pp. 481–507, 1997.

[123] B. Colson, P. Marcotte, and G. Savard, "A trust-region method for nonlinear bilevel programming: Algorithm and computational experience," *Computational Optimization and Applications*, vol. 30, no. 3, pp. 211–227, 2005.

[124] G.-s. Liu, S.-q. Xu, and J.-y. Han, "A trust region algorithm for solving bilevel programming problems," *Acta Mathematicae Applicatae Sinica, English Series*, vol. 29, no. 3, pp. 491–498, 2013.

[125] J. Liu, Y. Fan, Z. Chen, and Y. Zheng, "Methods for pessimistic bilevel optimization," in *Bilevel Optimization*, Springer, 2020, pp. 403–420.

[126] J. Liu, Y. Fan, Z. Chen, and Y. Zheng, "Pessimistic bilevel optimization: A survey," *International Journal of Computational Intelligence Systems*, vol. 11, no. 1, pp. 725–736, 2018.

[127] A. Aboussoror and A. Mansouri, "Weak linear bilevel programming problems: Existence of solutions via a penalty method," *Journal of Mathematical Analysis and Applications*, vol. 304, no. 1, pp. 399–408, 2005.

[128] Y. Zheng, Z. Wan, K. Sun, and T. Zhang, "An exact penalty method for weak linear bilevel programming problem," *Journal of Applied Mathematics and Computing*, vol. 42, no. 1, pp. 41–49, 2013.

[129] J. Liu, Y. Hong, and Y. Zheng, "A new variant of penalty method for weak linear bilevel programming problems," *Wuhan University Journal of Natural Sciences*, vol. 23, no. 4, pp. 328–332, 2018.

[130] Y. Zheng, D. Fang, and Z. Wan, "A solution approach to the weak linear bilevel programming problems," *Optimization*, vol. 65, no. 7, pp. 1437–1449, 2016.

[131] S. Dempe, B. S. Mordukhovich, and A. B. Zemkoho, "Two-level value function approach to non-smooth optimistic and pessimistic bilevel programs," *Optimization*, vol. 68, no. 2-3, pp. 433–455, 2019.

[132] I. Benchouk, K. Nachi, and A. Zemkoho, "Scholtes relaxation method for pessimistic bilevel optimization," *arXiv preprint arXiv:2110.13755*, 2021.

[133] M. B. Lignola and J. Morgan, "Inner regularizations and viscosity solutions for pessimistic bilevel optimization problems," *Journal of Optimization Theory and Applications*, vol. 173, pp. 183–202, 2017.

[134] D. Molodtsov, "The solution of a class of non-antagonistic games," *USSR Computational Mathematics and Mathematical Physics*, vol. 16, no. 6, pp. 67–72, 1976.

[135] W. F. Bialas and M. H. Karwan, "Two-level linear programming," *Management science*, vol. 30, no. 8, pp. 1004–1020, 1984.

[136] U.-P. Wen and S.-T. Hsu, "Linear bi-level programming problems—a review," *Journal of the operational research society*, vol. 42, pp. 125–133, 1991.

[137] D. Cao and L. C. Leung, "A partial cooperation model for non-unique linear two-level decision problems," *European Journal of Operational Research*, vol. 140, no. 1, pp. 134–141, 2002.

[138] P. Loridan and J. Morgan, "Weak via strong stackelberg problem: New results," *Journal of global Optimization*, vol. 8, pp. 263–287, 1996.

[139] **M. Antoniou**, A. Sinha, and G. Papa, "δ-perturbation of bilevel optimization problems: An error bound analysis," *Operations Research Perspectives*, p. 100 315, 2024.

[140] B. Zeng, "Easier than we thought-a practical scheme to compute pessimistic bilevel optimization problem," *Available at SSRN 2658342*, 2015.

[141] A. V. Malyshev and A. S. Strekalovsky, "Global search for pessimistic solution in bilevel problems," in *Proceedings of the Toulouse global optimization workshop*, 2010, pp. 77–80.

[142] A. Sinha, P. Malo, and K. Deb, "A Review on Bilevel Optimization: From Classical to Evolutionary Approaches and Applications," *arXiv:1705.06270 [cs, math]*, May 2017, arXiv: 1705.06270. [Online]. Available: `http://arxiv.org/abs/1705.06270` (visited on 03/06/2018).

[143] M. Sakawa Ichiro Nishizaki, "Computational methods through genetic algorithms for obtaining stackelberg solutions to two-level mixed zero-one programming problems," *Cybernetics & Systems*, vol. 31, no. 2, pp. 203–221, 2000.

[144] S. R. Hejazi, A. Memariani, G. Jahanshahloo, and M. M. Sepehri, "Linear bilevel programming solution by genetic algorithm," *Computers & Operations Research*, vol. 29, no. 13, pp. 1913–1925, 2002.

[145] R. Hayashi, H. Takano, W. M. Nyabuto, H. Asano, and T. Nguyen-Duc, "Bilevel optimization model for sizing of battery energy storage systems in a microgrid considering their economical operation," *Energy Reports*, vol. 9, pp. 728–737, 2023.

[146] Y. Wang, Y.-C. Jiao, and H. Li, "An Evolutionary Algorithm for Solving Nonlinear Bilevel Programming Based on a New Constraint-Handling Scheme," en, *IEEE Transactions on Systems, Man and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 2, pp. 221–232, May 2005, ISSN: 1094-6977. DOI: 10.1109/TSMCC.2004.841908. [Online]. Available: http://ieeexplore.ieee.org/document/1424196/ (visited on 05/31/2018).

[147] Y. Wang, H. Li, and C. Dang, "A New Evolutionary Algorithm for a Class of Nonlinear Bilevel Programming Problems and Its Global Convergence," en, *INFORMS Journal on Computing*, vol. 23, no. 4, pp. 618–629, Nov. 2011, ISSN: 1091-9856, 1526-5528. DOI: 10.1287/ijoc.1100.0430. [Online]. Available: http://pubsonline.informs.org/doi/abs/10.1287/ijoc.1100.0430 (visited on 05/31/2018).

[148] V. Oduguwa and R. Roy, "Bi-level optimisation using genetic algorithm," in *Proceedings 2002 IEEE International Conference on Artificial Intelligence Systems (ICAIS 2002)*, IEEE, 2002, pp. 322–327.

[149] F. Legillon, A. Liefooghe, and E.-G. Talbi, "Cobra: A coevolutionary metaheuristic for bi-level optimization," en, in *Metaheuristics for Bi-level Optimization*, ser. Studies in Computational Intelligence, Springer, Berlin, Heidelberg, 2013, pp. 95–114, ISBN: 978-3-642-37837-9 978-3-642-37838-6. DOI: 10.1007/978-3-642-37838-6_4. (visited on 05/31/2018).

[150] A. Chaabani, S. Bechikh, and L. B. Said, "A new co-evolutionary decomposition-based algorithm for bi-level combinatorial optimization," *Applied Intelligence*, vol. 48, pp. 2847–2872, 2018.

[151] A. Chaabani, S. Bechikh, and L. Ben Said, "A co-evolutionary hybrid decomposition-based algorithm for bi-level combinatorial optimization problems," *Soft Computing*, vol. 24, no. 10, pp. 7211–7229, 2020.

[152] R. Mathieu, L. Pittard, and G. Anandalingam, "Genetic algorithm based approach to bi-level linear programming," en, *RAIRO - Operations Research*, vol. 28, no. 1, pp. 1–21, 1994, ISSN: 0399-0559, 1290-3868. DOI: 10.1051/ro/1994280100011. [Online]. Available: http://www.rairo-ro.org/10.1051/ro/1994280100011 (visited on 05/30/2018).

[153] Yin Yafeng, "Genetic-Algorithms-Based Approach for Bilevel Programming Models," *Journal of Transportation Engineering*, vol. 126, no. 2, pp. 115–120, Mar. 2000. DOI: 10.1061/(ASCE)0733-947X(2000)126:2(115). [Online]. Available: https://ascelibrary.org/doi/abs/10.1061/(ASCE)0733-947X(2000)126:2(115) (visited on 05/30/2018).

[154] M. M. Islam, H. K. Singh, and T. Ray, "A memetic algorithm for solving single objective bilevel optimization problems," in *2015 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2015, pp. 1643–1650.

[155] A. Sinha, P. Malo, and K. Deb, "Test problem construction for single-objective bilevel optimization," *Evolutionary computation*, vol. 22, no. 3, pp. 439–477, 2014.

[156] A. Gupta, J. Mańdziuk, and Y.-S. Ong, "Evolutionary multitasking in bi-level optimization," *Complex & Intelligent Systems*, vol. 1, pp. 83–95, 2015.

[157] L. Chen, H.-L. Liu, K. C. Tan, and K. Li, "Transfer learning-based parallel evolutionary algorithm framework for bilevel optimization," *IEEE Transactions on Evolutionary Computation*, vol. 26, no. 1, pp. 115–129, 2021.

[158] W. Chen, P. Wang, and H. Dong, "Surrogate-based bilevel shape optimization for blended-wing–body underwater gliders," *Engineering Optimization*, vol. 55, no. 6, pp. 998–1019, 2023.

[159] H. Jiang, K. Chou, Y. Tian, X. Zhang, and Y. Jin, "Efficient surrogate modeling method for evolutionary algorithm to solve bilevel optimization problems," *IEEE Transactions on Cybernetics*, 2023.

[160] J. S. Angelo, E. Krempser, and H. J. C. Barbosa, "Differential evolution assisted by a surrogate model for bilevel programming problems," in *Evolutionary Computation (CEC), 2014 IEEE Congress on*, IEEE, 2014, pp. 1784–1791.

[161] Y. Xia, X. Liu, and G. Du, "Solving bi-level optimization problems in engineering design using kriging models," *Engineering Optimization*, vol. 50, no. 5, pp. 856–876, 2018.

[162] J.-A. Mejía-de-Dios and E. Mezura-Montes, "A surrogate-assisted metaheuristic for bilevel optimization," in *Proceedings of the 2020 genetic and evolutionary computation conference*, 2020, pp. 629–635.

[163] M. M. Islam, H. K. Singh, and T. Ray, "A surrogate assisted approach for single-objective bilevel optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 5, pp. 681–696, 2017.

[164] A. Sinha, P. Malo, and K. Deb, "An improved bilevel evolutionary algorithm based on quadratic approximations," in *2014 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2014, pp. 1870–1877.

[165] A. Sinha, Z. Lu, K. Deb, and P. Malo, "Bilevel optimization based on iterative approximation of multiple mappings," *Journal of Heuristics*, vol. 26, pp. 151–185, 2020.

[166] A. Sinha and V. Shaikh, "Solving bilevel optimization problems using kriging approximations," *IEEE Transactions on Cybernetics*, vol. 52, no. 10, pp. 10 639–10 654, 2021.

[167] H. K. Singh, M. M. Islam, T. Ray, and M. Ryan, "Nested evolutionary algorithms for computationally expensive bilevel optimization problems: Variants and their systematic analysis," *Swarm and Evolutionary Computation*, vol. 48, pp. 329–344, 2019.

[168] B. Moradi, M. Kirley, and M. A. Muñoz Acosta, "Sensitivity analysis of surrogate-assisted bilevel optimisation," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, ser. GECCO '24 Companion, Melbourne, VIC, Australia: Association for Computing Machinery, 2024, pp. 411–414, ISBN: 9798400704956. DOI: 10.1145/3638530.3654228. [Online]. Available: https://doi.org/10.1145/3638530.3654228.

[169] K. Deb, Z. Lu, I. Kropp, *et al.*, "Minimizing expected deviation in upper-level outcomes due to lower-level decision-making in hierarchical multi-objective problems," *IEEE Transactions on Evolutionary Computation*, 2022.

[170] M. J. Alves and C. H. Antunes, "A semivectorial bilevel programming approach to optimize electricity dynamic time-of-use retail pricing," *Computers & Operations Research*, vol. 92, pp. 130–144, 2018.

[171] G. Ren, Z. Huang, Y. Cheng, X. Zhao, and Y. Zhang, "An integrated model for evacuation routing and traffic signal optimization with background demand uncertainty," *Journal of Advanced Transportation*, vol. 47, no. 1, pp. 4–27, 2013.

[172] J. Bracken and J. McGill, "Mathematical programs with optimization problems in the constraints," *Operations Research*, vol. 21, pp. 37–44, 1973.

[173] A. Sinha, P. Malo, and K. Deb, "A review on bilevel optimization: From classical to evolutionary approaches and applications," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 276–295, 2018.

[174] A. Migdalas, "Bilevel programming in traffic planning: Models, methods and challenge," *Journal of Global Optimization*, vol. 7, no. 4, pp. 381–405, 1995.

[175] L. Brotcorne, M. Labbe, P. Marcotte, and G. Savard, "A bilevel model for toll optimization on a multicommodity transportation network," *Transportation Science*, vol. 35, no. 4, pp. 345–358, 2001.

[176] A. Sinha, P. Malo, and K. Deb, "Transportation policy formulation as a multi-objective bilevel optimization problem," in *2015 IEEE Congress on Evolutionary Computation (CEC-2015)*, IEEE Press, 2015.

[177] S. Dempe, B. S. Mordukhovich, and A. B. Zemkoho, "Necessary optimality conditions in pessimistic bilevel programming," *Optimization*, vol. 63, no. 4, pp. 505–533, 2014.

[178] S. Dempe and A. B. Zemkoho, "KKT reformulation and necessary conditions for optimality in nonsmooth bilevel optimization," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 1639–1669, 2014.

[179] G. Savard and J. Gauvin, "The steepest descent direction for the nonlinear bilevel programming problem," *Operations Research Letters*, vol. 15, pp. 275–282, 1994.

[180] L. Vicente, G. Savard, and J. Judice, "Descent approaches for quadratic bilevel programming," *Journal of Optimization Theory and Applications*, vol. 81, pp. 379–399, 1994.

[181] G. Liu, J. Han, and S. Wang, "A trust region algorithm for bilevel programing problems," *Chinese science bulletin*, vol. 43, no. 10, pp. 820–824, 1998.

[182] P. Marcotte, G. Savard, and D. L. Zhu, "A trust region algorithm for nonlinear bilevel programming," *Operations research letters*, vol. 29, no. 4, pp. 171–179, 2001.

[183] Y. Ishizuka and E. Aiyoshi, "Double penalty method for bilevel optimization problems," *Annals of Operations Research*, vol. 34, pp. 73–88, 1992.

[184] D. White and G. Anandalingam, "A penalty function approach for solving bi-level linear programs," *Journal of Global Optimization*, vol. 3, pp. 397–419, 1993.

[185] T. Kleinert and M. Schmidt, "Computing feasible points of bilevel problems with a penalty alternating direction method," *INFORMS Journal on Computing*, vol. 33, no. 1, pp. 198–215, 2021.

[186] R. Mathieu, L. Pittard, and G. Anandalingam, "Genetic algorithm based approach to bi-level linear programming," *Operations Research*, vol. 28, no. 1, pp. 1–21, 1994.

[187] Y. Yin, "Genetic algorithm based approach for bilevel programming models," *Journal of Transportation Engineering*, vol. 126, no. 2, pp. 115–120, 2000.

[188] H. Li, "A genetic algorithm using a finite search space for solving nonlinear/linear fractional bilevel programming problems," *Annals of Operations Research*, pp. 1–16, 2015.

[189] J. Angelo, E. Krempser, and H. Barbosa, "Differential evolution for bilevel programming," in *Proceedings of the 2013 Congress on Evolutionary Computation (CEC-2013)*, IEEE Press, 2013.

[190] L. Wu, Z. Liu, H.-L. Wei, and R. Wang, "An efficient bilevel differential evolution algorithm with adaptation of lower level population size and search radius," *Memetic Computing*, vol. 13, no. 2, pp. 227–247, 2021.

[191] E. Kieffer, G. Danoy, P. Bouvry, and A. Nagih, "Bayesian optimization approach of general bi-level problems," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2017, pp. 1614–1621.

[192] X. Zhu, Q. Yu, and X. Wang, "A hybrid differential evolution algorithm for solving nonlinear bilevel programming with linear constraints," in *Cognitive Informatics, 2006. ICCI 2006. 5th IEEE International Conference on*, IEEE, vol. 1, 2006, pp. 126–131.

[193] R. Salinas-Guerra, E. Mezura-Montes, M. Quiroz-Castellanos, J.-A. Mejía-de-Dios, and S. Bechikh, "A hybrid evolutionary approach with group-based solution encoding for solving the constrained bilevel multi-depot vehicle routing problem," in *2023 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2023, pp. 1–8.

[194] Y. Jiang, X. Li, C. Huang, and X. Wu, "Application of particle swarm optimization based on chks smoothing function for solving nonlinear bilevel programming problem," *Applied Mathematics and Computation*, vol. 219, no. 9, pp. 4332–4339, 2013.

[195] A. Sinha, T. Soun, and K. Deb, "Using karush-kuhn-tucker proximity measure for solving bilevel optimization problems," *Swarm and evolutionary computation*, vol. 44, pp. 496–510, 2019.

[196] X. Li, P. Tian, and X. Min, "A hierarchical particle swarm optimization for solving bilevel programming problems," in *Artificial Intelligence and Soft Computing - ICAISC 2006*, ser. Lecture Notes in Computer Science, L. Rutkowski, R. Tadeusiewicz, L. A. Zadeh, and J. M. Zurada, Eds., vol. 4029, Springer Berlin Heidelberg, 2006, pp. 1169–1178, ISBN: 978-3-540-35748-3. DOI: 10.1007/11785231_122. [Online]. Available: http://dx.doi.org/10.1007/11785231_122.

[197] A. Sinha, P. Malo, and K. Deb, "Evolutionary algorithm for bilevel optimization using approximations of the lower level optimal solution mapping," *European Journal of Operational Research*, vol. 257, no. 2, pp. 395–411, 2017.

[198] E. Alekseeva, Y. Kochetov, and E.-G. Talbi, "A matheuristic for the discrete bilevel problem with multiple objectives at the lower level," *International Transactions in Operational Research*, vol. 24, no. 5, pp. 959–981, 2017.

[199] A. Tsoukalas, W. Wiesemann, B. Rustem, *et al.*, "Global optimisation of pessimistic bi-level problems," *Lectures on global optimization*, vol. 55, pp. 215–243, 2009.

[200] R. Lucchetti, F. Mignanego, and G. Pieri, "Existence theorem of equilibrium points in Stackelberg games with constraints," *Optimization*, vol. 18, pp. 857–866, 1987.

[201] P. Loridan and J. Morgan, "Weak via strong Stackelberg problems," *Journal of Global Optimization*, vol. 8, pp. 263–287, 1996.

[202] W. Wiesemann, A. Tsoukalas, P. M. Kleniati, and B. Rustem, "Pessimistic Bilevel Optimization," *SIAM Journal on Optimization*, vol. 23, Feb. 2013. DOI: 10.1137/120864015.

[203] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[204] K. Deb, "An efficient constraint handling method for genetic algorithms," *Computer methods in applied mechanics and engineering*, vol. 186, no. 2-4, pp. 311–338, 2000.

[205] Y. Zheng, G. Zhang, Z. Zhang, and J. Lu, "A reducibility method for the weak linear bilevel programming problems and a case study in principal-agent," *Information Sciences*, vol. 454, pp. 46–58, 2018.

[206] **M. Antoniou**, A. Sinha, and G. Papa, "A study on optimistic & pessimistic pareto-fronts in multiobjective bilevel optimization via δ-perturbation (accepted)," in *Proceedings of the 2025 International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, 2025.

[207] A. Sinha, P. Malo, and K. Deb, "A review on bilevel optimization: From classical to evolutionary approaches and applications," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 2, pp. 276–295, 2017.

[208] J.-A. Mejía-de-Dios, A. Rodríguez-Molina, and E. Mezura-Montes, "Multiobjective bilevel optimization: A survey of the state-of-the-art," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2023.

[209] G. Eichfelder, "Solving nonlinear multiobjective bilevel optimization problems with coupled upper level constraints," 2007. [Online]. Available: `https://api.semanticscholar.org/CorpusID:211008773`.

[210] W. Halter and S. Mostaghim, "Bilevel optimization of multi-component chemical systems using particle swarm optimization," *2006 IEEE International Conference on Evolutionary Computation*, pp. 1240–1247, 2006. [Online]. Available: `https://api.semanticscholar.org/CorpusID:22916372`.

[211] K. Deb and A. Sinha, "An efficient and accurate solution methodology for bilevel multi-objective programming problems using a hybrid evolutionary-local-search algorithm," *Evolutionary computation*, vol. 18, no. 3, pp. 403–449, 2010.

[212] K. Deb and A. Sinha, "Solving bilevel multi-objective optimization problems using evolutionary algorithms," in *International conference on evolutionary multi-criterion optimization*, Springer, 2009, pp. 110–124.

[213] A. Sinha and K. Deb, "Towards understanding evolutionary bilevel multi-objective optimization algorithm," *IFAC Proceedings Volumes*, vol. 42, no. 2, pp. 338–343, 2009.

[214] A. Sinha, P. Malo, and K. Deb, "Approximated set-valued mapping approach for handling multiobjective bilevel problems," *Computers and Operations Research*, vol. 77, pp. 194–209, 2017.

[215] A. Sinha, P. Malo, and K. Deb, "Towards understanding bilevel multi-objective optimization with deterministic lower level decisions," in *Evolutionary Multi-Criterion Optimization: 8th International Conference, EMO 2015, Guimarães, Portugal, March 29–April 1, 2015. Proceedings, Part I 8*, Springer, 2015, pp. 426–443.

[216] A. Sinha, P. Malo, K. Deb, P. Korhonen, and J. Wallenius, "Solving bilevel multi-criterion optimization problems with lower level decision uncertainty," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 199–217, 2015.

[217] A. Sinha, P. Malo, and K. Deb, "Evolutionary algorithm for bilevel optimization using approximations of the lower level optimal solution mapping," *European Journal of Operational Research*, vol. 257, no. 2, pp. 395–411, 2017.

[218] A. Sinha, Z. Lu, K. Deb, and P. Malo, "Bilevel optimization based on iterative approximation of multiple mappings," *Journal of Heuristics*, vol. 26, no. 2, pp. 151–185, 2020.

[219] M. J. Alves, C. H. Antunes, and J. P. Costa, "New concepts and an algorithm for multiobjective bilevel programming: Optimistic, pessimistic and moderate solutions," *Operational Research*, vol. 21, no. 4, pp. 2593–2626, 2021.

[220] L. Lampariello, S. Sagratella, and O. Stein, "The standard pessimistic bilevel problem," *SIAM Journal on Optimization*, vol. 29, no. 2, pp. 1634–1656, 2019.

[221] K. Deb, *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, 2001.

[222] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257–271, 1999.

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

[224] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[225] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: improving the strength pareto evolutionary algorithm," TIK Report, ETH Zurich, Tech. Rep., 2001.

[226] Q. Zhang and H. Li, "Moea/d: A multiobjective evolutionary algorithm based on decomposition," *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712–731, 2007.

[227] E. Zitzler and S. Künzli, "Indicator-based selection in multiobjective search," in *International Conference on Parallel Problem Solving from Nature*, Springer, 2004, pp. 832–842.

[228] D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," *Air Force Institute of Technology*, 2000.

[229] J. D. Knowles and D. W. Corne, "On metrics for comparing nondominated sets," in *Proceedings of the 2002 Congress on Evolutionary Computation*, IEEE, 2002, pp. 711–716.

[230] H. Ishibuchi, R. Shang, C. Zhang, and Y. Nojima, "Modified distance calculation in generational distance and inverted generational distance," in *Evolutionary Multi-Criterion Optimization, EMO 2015*, ser. Lecture Notes in Computer Science, vol. 9019, Springer, 2015, pp. 110–125. DOI: 10.1007/978-3-319-15934-8_8.

[231] S. Gass and T. Saaty, "The computational algorithm for the parametric objective function," *Naval research logistics quarterly*, vol. 2, no. 1-2, pp. 39–45, 1955.

[232] L. Zadeh, "Optimality and non-scalar-valued performance criteria," *IEEE transactions on Automatic Control*, vol. 8, no. 1, pp. 59–60, 1963.

[233] R. Kasimbeyli, "A conic scalarization method in multi-objective optimization," *Journal of Global Optimization*, vol. 56, pp. 279–297, 2013.

[234] H. P. Benson, "Existence of efficient solutions for vector maximization problems," *Journal of Optimization Theory and Applications*, vol. 26, pp. 569–580, 1978.

[235] C. A. Coello Coello and M. Reyes Sierra, "A study of the parallelization of a coevolutionary multi-objective evolutionary algorithm," in *MICAI 2004: Advances in Artificial Intelligence: Third Mexican International Conference on Artificial Intelligence, Mexico City, Mexico, April 26-30, 2004. Proceedings 3*, Springer, 2004, pp. 688–697.

[236] **M. Antoniou** and G. Papa, "Evaluation of parallel hierarchical differential evolution for min-max optimization problems using scipy," in *International Conference on Bioinspired Optimization Methods and Their Applications*, Springer, 2022, pp. 84–98.

[237] **M. Antoniou** and G. Papa, "Differential evolution with estimation of distribution for worst-case scenario optimization," *Mathematics*, vol. 9, no. 17, p. 2137, 2021.

[238] **M. Antoniou** and G. Papa, "Solving min-max optimisation problems by means of bilevel evolutionary algorithms: A preliminary study," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 2020, pp. 187–188.

[239] M. T. Jensen, "A new look at solving minimax problems with coevolution," in *Proceedings of the Fourth Metaheuristics International Conference (MIC-2001)*, Citeseer, 2001, pp. 103–107.

[240] A. M. Cramer, S. D. Sudhoff, and E. L. Zivi, "Evolutionary algorithms for minimax problems in robust design," *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 444–453, 2008.

[241] M. Goerigk, J. Kurtz, M. Schmidt, and J. Thürauf, "Connections between robust and bilevel optimization," 2024.

[242] H. Wang, L. Feng, Y. Jin, and J. Doherty, "Surrogate-assisted evolutionary multitasking for expensive minimax optimization in multiple scenarios," *IEEE Computational Intelligence Magazine*, vol. 16, no. 1, pp. 34–48, 2021.

[243] H.-G. Beyer and B. Sendhoff, "Robust optimization–a comprehensive survey," *Computer methods in applied mechanics and engineering*, vol. 196, no. 33-34, pp. 3190–3218, 2007.

[244] Y. Cai, O. Candogan, C. Daskalakis, and C. Papadimitriou, "Zero-sum polymatrix games: A generalization of minmax," *Mathematics of Operations Research*, vol. 41, no. 2, pp. 648–655, 2016.

[245] G. E. Monahan, "Finding saddle points on polyhedra: Solving certain continuous minimax problems," *Naval Research Logistics (NRL)*, vol. 43, no. 6, pp. 821–837, 1996.

[246] J. Hofbauer and S. Sorin, "Best response dynamics for continuous zero-sum games," *Discrete and Continuous Dynamical Systems Series B*, vol. 6, no. 1, p. 215, 2006.

[247] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*. Princeton University Press, 2009.

[248] J. R. Birge and F. Louveaux, *Introduction to Stochastic Programming*. Springer Science Business Media, 2011.

[249] A. Prékopa, *Stochastic Programming*. Springer, 1995.

[250] P. Hansen and B. Jaumard, "Branch-and-bound methods for global optimization: A review," *Annals of Operations Research*, vol. 30, no. 1, pp. 289–325, 1992.

[251] Y. Chen and M. Hong, "Convergent second-order methods for nonconvex-nonconcave min-max optimization," *Mathematical Programming*, pp. 1–29, 2023.

[252] J. E. Kelly, "A cutting-plane algorithm for solving convex programs," *Journal of the Society for Industrial and Applied Mathematics*, vol. 8, no. 4, pp. 703–712, 1960.

[253] A. Shapiro, "Duality in robust optimization and stochastic programming," *Mathematical Programming*, vol. 120, no. 1, pp. 149–166, 2009.

[254] A. V. Sebald and J. Schlenzig, "Minimax design of neural net controllers for highly uncertain plants," *IEEE Transactions on Neural Networks*, vol. 5, no. 1, pp. 73–82, 1994.

[255] E. C. Laskari, K. E. Parsopoulos, and M. N. Vrahatis, "Particle swarm optimization for minimax problems," in *Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600)*, IEEE, vol. 2, 2002, pp. 1576–1581.

[256] B.-C. Wang, Y. Feng, X.-B. Meng, and S. Wang, "A two-phase differential evolution for minimax optimization," *Applied Soft Computing*, vol. 131, p. 109 797, 2022.

[257] M.-J. Tahk and B.-C. Sun, "Coevolutionary augmented lagrangian methods for constrained optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 2, pp. 114–124, 2000.

[258] R. A. Krohling and L. dos Santos Coelho, "Coevolutionary particle swarm optimization using gaussian distribution for solving constrained optimization problems," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 36, no. 6, pp. 1407–1416, 2006.

[259] F. Fabris and R. A. Krohling, "A co-evolutionary differential evolution algorithm for solving min–max optimization problems implemented on gpu using c-cuda," *Expert Systems with Applications*, vol. 39, no. 12, pp. 10 324–10 333, 2012.

[260] J. Paredis *et al.*, "Coevolving cellular automata: Be aware of the red queen!.," in *ICGA*, Citeseer, 1997, pp. 393–400.

[261] J. Branke and J. Rosenbusch, "New approaches to coevolutionary worst-case optimization," in *International Conference on Parallel Problem Solving from Nature*, Springer, 2008, pp. 144–153.

[262] W. Yang, F. Mei, and G. Zhai, "Efficient global optimization with co-evolutionary strategy for asymmetric minimax problem in robust design," *Quality and Reliability Engineering International*, vol. 39, no. 6, pp. 2496–2514, 2023.

[263] R.-B. Chen, S.-P. Chang, W. Wang, H.-C. Tung, and W. K. Wong, "Minimax optimal designs via particle swarm optimization methods," *Statistics and Computing*, vol. 25, no. 5, pp. 975–988, 2015.

[264] A. Zhou and Q. Zhang, "A surrogate-assisted evolutionary algorithm for minimax optimization," in *IEEE Congress on Evolutionary Computation*, IEEE, 2010, pp. 1–7.

[265] J. Marzat, E. Walter, and H. Piet-Lahanier, "A new expected-improvement algorithm for continuous minimax optimization," *Journal of Global Optimization*, vol. 64, no. 4, pp. 785–802, 2016.

[266] Y.-S. Ong, P. B. Nair, and K. Y. Lum, "Max-min surrogate-assisted evolutionary algorithm for robust design," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 392–404, 2006.

[267] X. Qiu, J.-X. Xu, Y. Xu, and K. C. Tan, "A new differential evolution algorithm for minimax optimization in robust design," *IEEE transactions on cybernetics*, vol. 48, no. 5, pp. 1355–1368, 2017.

[268] M. Hauschild and M. Pelikan, "An introduction and survey of estimation of distribution algorithms," *Swarm and evolutionary computation*, vol. 1, no. 3, pp. 111–128, 2011.

[269] P. Larrañaga and J. A. Lozano, *Estimation of distribution algorithms: A new tool for evolutionary computation*. Springer Science & Business Media, 2012, vol. 2.

[270] K. Brittain, M. Silva, and D. A. Tortorelli, "Minmax topology optimization," *Structural and Multidisciplinary Optimization*, vol. 45, pp. 657–668, 2012.

[271] R. Chai, A. Tsourdos, A. Savvaris, S. Chai, Y. Xia, and C. P. Chen, "Review of advanced guidance and control algorithms for space/aerospace vehicles," *Progress in Aerospace Sciences*, vol. 122, p. 100 696, 2021.

[272] L. El Ghaoui, M. Oks, and F. Oustry, "Worst-case value-at-risk and robust portfolio optimization: A conic programming approach," *Operations Research*, vol. 51, no. 4, pp. 543–556, 2003.

[273] A. Sinha, Z. Lu, K. Deb, and P. Malo, "Bilevel optimization based on iterative approximation of multiple mappings," *Journal of Heuristics*, pp. 1–35, 2017.

[274] Y. Feng, L. Hongwei, Z. Shuisheng, and L. Sanyang, "A smoothing trust-region newton-cg method for minimax problem," *Applied Mathematics and Computation*, vol. 199, no. 2, pp. 581–589, 2008.

[275] R. Montemanni, L. M. Gambardella, and A. V. Donati, "A branch and bound algorithm for the robust shortest path problem with interval data," *Operations Research Letters*, vol. 32, no. 3, pp. 225–232, 2004.

[276] H. Aissi, C. Bazgan, and D. Vanderpooten, "Min–max and min–max regret versions of combinatorial optimization problems: A survey," *European journal of operational research*, vol. 197, no. 2, pp. 427–438, 2009.

[277] M. Vasile, "On the solution of min-max problems in robust optimization," in *The EVOLVE 2014 International Conference, A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computing*, 2014.

[278] J. Marzat, E. Walter, and H. Piet-Lahanier, "Worst-case global optimization of black-box functions through kriging and relaxation," *Journal of Global Optimization*, vol. 55, no. 4, pp. 707–727, 2013.

[279] P. Larranaga, "A review on estimation of distribution algorithms," in *Estimation of distribution algorithms*, Springer, 2002, pp. 57–100.

[280] Q. Yang, W.-N. Chen, Y. Li, C. P. Chen, X.-M. Xu, and J. Zhang, "Multimodal estimation of distribution algorithms," *IEEE transactions on cybernetics*, vol. 47, no. 3, pp. 636–650, 2016.

[281] J. Ceberio, E. Irurozki, A. Mendiburu, and J. A. Lozano, "A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems," *Progress in Artificial Intelligence*, vol. 1, pp. 103–117, 2012.

[282] M. Pelikan, K. Sastry, and D. E. Goldberg, "Multiobjective estimation of distribution algorithms," in *Scalable optimization via probabilistic modeling*, Springer, 2006, pp. 223–248.

[283] B. L. Gorissen, İ. Yanıkoğlu, and D. den Hertog, "A practical guide to robust optimization," *Omega*, vol. 53, pp. 124–137, 2015.

[284] `https://www.mathworks.com/help/stats/mvnrnd.html`, Accessed: 2021-08-03.

[285] F. Wilcoxon, *Individual comparisons by ranking methods. biom. bull., 1, 80–83*, 1945.

[286] B. Brown and T. Singh, "Minimax design of vibration absorbers for linear damped systems," *Journal of Sound and Vibration*, vol. 330, no. 11, pp. 2437–2448, 2011.

[287] **M. Antoniou**, A. Sinha, and G. Papa, "An evolutionary approach to pessimistic bilevel optimization problems," *International Conference on Bilevel Optimization: conference book*, p. 58, 2023, Winner of 3rd Prize for the Poster Competition. [Online]. Available: `https://dirros.openscience.si/IzpisGradiva.php?lang=slv&id=20431`.

[288] A. Gupta, Y.-S. Ong, and L. Feng, "Multifactorial evolution: Toward evolutionary multitasking," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 3, pp. 343–357, 2015.

[289] K. C. Tan, L. Feng, and M. Jiang, "Evolutionary transfer optimization-a new frontier in evolutionary computation research," *IEEE Computational Intelligence Magazine*, vol. 16, no. 1, pp. 22–33, 2021.

[290] B. Rustem and M. Howe, *Algorithms for worst-case design and applications to risk management*. Princeton University Press, 2009.

[291] **M. Antoniou** and G. Papa, "Solving pessimistic bilevel optimisation problems with evolutionary algorithms," ECCOMAS Proceedia EUROGEN, 2021, pp. 224–233. DOI: `https://doi.org/10.7712/140121.7962.18327`.

[292] G. Petelin, **M. Antoniou**, and G. Papa, "Multi-objective approaches to ground station scheduling for optimization of communication with satellites," *Optimization and Engineering*, pp. 1–38, 2021.

[293] **M. Antoniou**, G. Petelin, and G. Papa, "On formulating the ground scheduling problem as a multi-objective bilevel problem," in *International Conference on Bioinspired Methods and Their Applications*, Springer, 2020, pp. 177–188.

[294] **M. Antoniou**, G. Petelin, and G. Papa, "Preferred solutions of the ground station scheduling problem using NSGA-III with weighted reference points selection," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2021, pp. 1840–1847.

[295] G. Petelin, **M. Antoniou**, and G. Papa, "Dynamic computational resource allocation for CFD simulations based on Pareto front optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2022, pp. 1666–1673.

Bibliography

Publications Related to the Thesis

The papers listed below have contributed to this thesis, either directly or indirectly. The author appreciates the opportunity to present and discuss unpublished work at international workshops, conferences and seminars.

Journal Articles

M. Antoniou, A. Sinha, and G. Papa, "δ-perturbation of bilevel optimization problems: An error bound analysis," *Operations Research Perspectives*, p. 100 315, 2024.

M. Antoniou and G. Papa, "Differential evolution with estimation of distribution for worst-case scenario optimization," *Mathematics*, vol. 9, no. 17, p. 2137, 2021.

G. Petelin, **M. Antoniou**, and G. Papa, "Multi-objective approaches to ground station scheduling for optimization of communication with satellites," *Optimization and Engineering*, pp. 1–38, 2021.

Conference Proceedings

M. Antoniou, A. Sinha, and G. Papa, "A study on optimistic & pessimistic pareto-fronts in multiobjective bilevel optimization via δ-perturbation (accepted)," in *Proceedings of the 2025 International Conference on Evolutionary Multi-Criterion Optimization (EMO)*, 2025.

M. Antoniou, G. Petelin, and G. Papa, "On formulating the ground scheduling problem as a multi-objective bilevel problem," in *International Conference on Bioinspired Methods and Their Applications*, Springer, 2020, pp. 177–188.

M. Antoniou and G. Papa, "Solving min-max optimisation problems by means of bilevel evolutionary algorithms: A preliminary study," in *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, 2020, pp. 187–188.

M. Antoniou, G. Petelin, and G. Papa, "Preferred solutions of the ground station scheduling problem using NSGA-III with weighted reference points selection," in *2021 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, 2021, pp. 1840–1847.

G. Petelin, **M. Antoniou**, and G. Papa, "Dynamic computational resource allocation for CFD simulations based on Pareto front optimization," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2022, pp. 1666–1673.

M. Antoniou and G. Papa, "Evaluation of parallel hierarchical differential evolution for min-max optimization problems using scipy," in *International Conference on Bioinspired Optimization Methods and Their Applications*, Springer, 2022, pp. 84–98.

D. Irawan, **M. Antoniou**, B. Naujoks, and G. Papa, "Refining the CC-RDG3 algorithm with increasing population scheme and persistent covariance matrix," in *International Conference on Bioinspired Methods and Their Applications*, Springer, 2020, pp. 69–83.

M. Antoniou and G. Papa, "Solving pessimistic bilevel optimisation problems with evolutionary algorithms," ECCOMAS Proceedia EUROGEN, 2021, pp. 224–233. DOI: https://doi.org/10.7712/140121.7962.18327.

R. Hribar, G. Petelin, **M. Antoniou**, A. Biasizzo, S. Ciglarič, and G. Papa, "On suitability of the customized measuring device for electric motor," in *Annual Conference of Industrial Electronics Society*. 2022. [Online]. Available: https://dirros.openscience.si/IzpisGradiva.php?lang=slv&id=15910.

Book Chapters

M. Antoniou, R. Hribar, and G. Papa, "Parameter control in evolutionary optimisation," in *Optimization Under Uncertainty with Applications to Aerospace Engineering*, Springer, 2021, pp. 357–385.

M. Antoniou and P. Korošec, "Multilevel optimisation," in *Optimization Under Uncertainty with Applications to Aerospace Engineering*, Springer, 2021, pp. 307–331.

R. Hribar, **M. Antoniou**, and G. Papa, "A framework for applying data-driven AI/ML models in reliability," in *Recent Advances in Microelectronics Reliability: Contributions from the European ECSEL JU Project IRel40*. Springer Nature, 2024, 1 spletni vir (1 PDF dokument (323–337)), Nasl. z nasl. zaslona. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-59361-1_12.

International Presentations, Workshops and Seminars

M. Antoniou, P. Korošec, and G. Papa, "Parameter control in evolutionary bilevel optimisation," *Knjiga povzetkov: science of the future how to stay up-to-date with your research!*, p. 91, 2019, Nasl. z nasl. zaslona Opis vira z dne 22. 5. 2019. [Online]. Available: http://ipssc.mps.si/Proceedings/Proceedings2019.pdf.

M. Antoniou, T. Krak, A. Erreygers, J. De Bock, and G. De Cooman, "Bounding limit expectations of markov chains using evolutionary algorithms," in *Book of Abstracts: OSE5, Optimisation in Space Engineering Workshop*. Ljubljana, Slovenia, 2019, Presented at OSE5, 21–22 November, 2019. [Online]. Available: https://dirros.openscience.si/IzpisGradiva.php?lang=slv&id=16529.

M. Antoniou, P. Korošec, and G. Papa, "An adaptive evolutionary surrogate-based approach for single-objective bilevel optimisation," *UQOP 2019, Uncertainty Quantification and OPtimization: 18-20 Marc, 2019, Paris (France)*, p. 2, 2019, Nasl. z nasl. zaslona Opis vira z dne 30. 8. 2019. [Online]. Available: https://uqop.sciencesconf.org/242425/document.

M. Antoniou, "Evolutionary approaches for bilevel optimisation," *Zbornik povzetkov slovenskih delavnic "Algoritmi po vzorih iz narave" v študijskem letu 2018/2019*, B. Filipič, M. Mernik, and G. Papa, Eds., p. 8, 2019.

M. Antoniou, A. Sinha, and G. Papa, "An evolutionary approach to pessimistic bilevel optimization problems," *International Conference on Bilevel Optimization: conference book*, p. 58, 2023, Winner of 3rd Prize for the Poster Competition. [Online]. Available: https://dirros.openscience.si/IzpisGradiva.php?lang=slv&id=20431.

GitHub Repository

M. Antoniou, `https://github.com/MargAnt0/ParallelMinMax_Scipy`, 2022.

Biography

Margarita Antoniou was born in Giannitsa, Greece. She earned her diploma in Civil Engineering from the Aristotle University of Thessaloniki in 2014, followed by a Master's degree in Environmental Protection and Sustainable Development in 2015. She then pursued a second Master's program in Computational Physics, which she completed in 2020. During her undergraduate studies, she spent one semester at RWTH Aachen University through the Erasmus exchange program.

In 2017, Margarita began her PhD studies in Information and Communication Technologies at the Jožef Stefan International Postgraduate School, Slovenia, under the supervision of Prof. Dr. Gregor Papa. Throughout her doctoral studies, she has worked as a research assistant at the Computer Systems Department of the Jožef Stefan Institute, where she was a Marie Skłodowska-Curie Early Stage Researcher within the UTOPIAE project and contributed to several other EU-funded projects, such as iRel40 and CONDUCTOR. As part of the UTOPIAE project, she completed secondments at ESTECO, an optimization software company in Trieste, Italy, and the University of Ghent in Belgium. Prior to her PhD studies, she gained practical experience in infrastructure projects while working as a civil engineer at the Municipal Water Utility of Pella and in a technical office.

Margarita's research focuses on evolutionary algorithms and optimization, particularly in hierarchical decision-making problems, including bilevel and min-max optimization. In addition, she has explored optimization techniques applied to real-world applications in domains such as water management, transport, satellite scheduling, and sustainable mining. Her work has been published in peer-reviewed journals and presented at several international conferences and workshops.