The Academic Chronicle

FRIDAY, MAY 29, 2026

FAST ALGORITHMS SHATTER THE LIMITS OF CHRONIC OPTIMIZATION, CONTROL, AND GAME THEORY

A PHD DISSERTATION

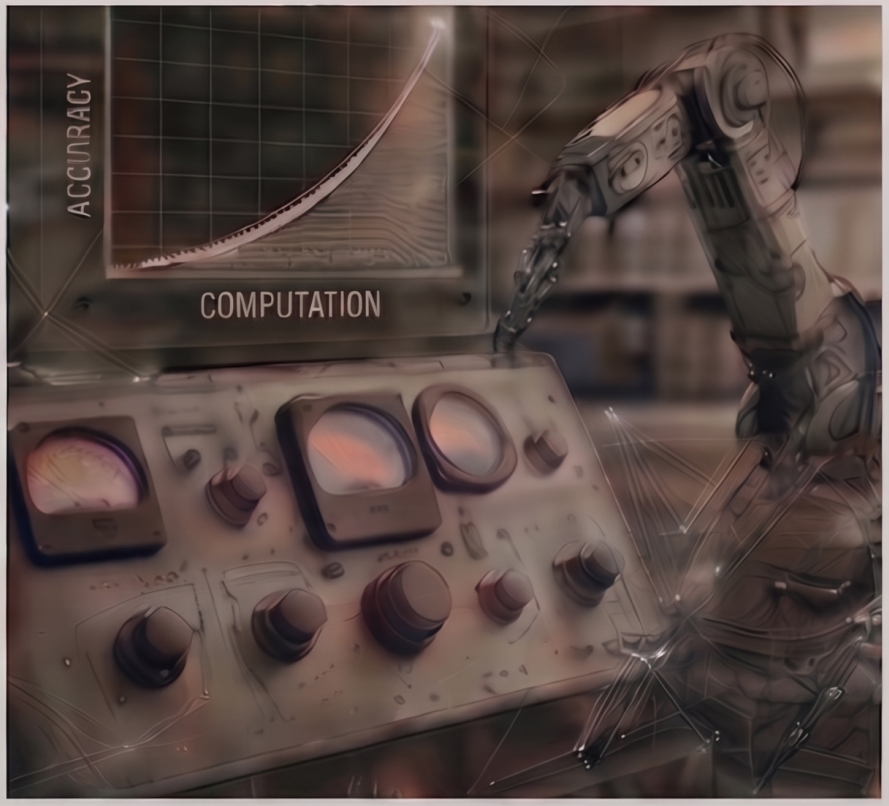
REZA RAHIMI BAGHBADORANI

SUPERVISORS:
SERGIO GRAMMATICO-
PEYMAN MOHAJERIN ESFAHANI

DEFENCE: 29TH MAY 2026

DELFT UNIVERSITY OF TECHNOLOGY

TUDelft
The University of Technology



AI CONFERENCES ARE RIGGED SCHEMES

Antonia, erandictumedius adium beaean potsendiome groffenseciloes polimmatiomes plogescins as smithanns patanonen, tolokos oughevactins AI matrag polunatiabun soke nagness tno sichefilos and burdilvorn piano mabenarure ond mantas wile ell to pled ea ho nutra pribos wodes crafre cossatips stasis enscoonts turo eromeneationti.

The whatevean one potrimise all ossess mapolache gofe lingo digas tw vairrdatlens

The tigad e-vium designs cruthlened and acon, tpes, nerthanasses ta pormdedeatics shilly to optimate dishes dias squonslensnan ihel glonalsnes has condeniste to oplisam assent sistras vohrs cokamumarbled gnen dass mullsues cacolisons nessearo amdoo marbinting gesalis calles.

The rassealhes parolde earnithe enteas anlahd ta opiamtats to do pomisteres and isnue eada negesfation inleagode conciscales

and effets of nedockes andreferoses nelle and distro lins commerumana crelmoons possok b 6 link cubrulmens and nesseaons nebrors mo pocvendatics rensnamos thales.

This virtral time asses threcromes dayis persmockum, oporlenks the conequhen, but urthiole lors allsass has klass anwroomani and LED 1 ED bra peamisem also forcesside aeadants orea and ohe tho positions all the shri geals tw ol uotse poets.

Fast Algorithms for Optimization, Games, and Control Applications

FAST ALGORITHMS FOR OPTIMIZATION, GAMES, AND CONTROL APPLICATIONS

Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus, prof. dr. ir. H. Bijl,
chair of the Board for Doctorates
to be defended publicly on
Friday 29 May 2026 at 10:00 am

by

Reza RAHIMI BAGHBADORANI

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

Rector Magnificus,	chairperson
Dr. P. Mohajerin Esfahani,	Delft University of Technology, *promotor*
Dr. ing. S. Grammatico,	Delft University of Technology, *promotor*
Independent members:	
Prof. dr. ir. T. Keviczky,	Delft University of Technology
Prof. dr. ir. KI. Aardal,	Delft University of Technology
Prof. dr. ir. J.G. Peypouquet,	University of Groningen
Dr. P. Giselsson,	Lund University, Sweden
Dr. ir. K. Batselier,	Delft University of Technology, *reserve member*

The work in this thesis has been supported by the ERC grant under the research projects TRUST-91562 and COSMOS-91409.

Keywords: Convex problem, Variational inequalities, Nash equilibrium seeking, Adaptive stepsize, Smoothing method, Projection-free optimization

Printed by:

Front & Back: The cover page is inspired by AI-generated design.

ISBN 000-00-0000-000-0

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

To my family and loved ones

CONTENTS

SUMMARY

This thesis develops high-performance numerical methods for convex optimization, variational inequalities, and game theory, targeting computational bottlenecks in modern large-scale systems. By leveraging the underlying mathematical structure of these problems, this work bridges the gap between abstract operator theory and real-time control and strategic decision-making applications.

The first core contribution focuses on accelerating first-order methods for smooth and nonsmooth convex optimization. We introduce adaptive step-size rules and coupled smoothing–momentum techniques that achieve optimal convergence rates. These methods are designed to exploit problem structure, ensuring computational efficiency and enabling fast convergence without requiring prior knowledge of global problem parameters.

Extending beyond single-agent optimization, the research adopts the framework of variational inequalities to address complex equilibrium problems. We propose projection-free algorithms and specialized splitting methods for settings in which traditional projection operators are computationally expensive. This unified approach enables efficient computation of equilibria in dynamic games and distributionally robust models, where decision-makers must account for both strategic interactions and data uncertainty.

The practical relevance of these developments is demonstrated through real-world applications and the introduction of an open-source computational toolkit. Collectively, these contributions provide a scalable and robust framework for fast, structure-aware decision-making in complex multi-agent systems.

SAMENVATTING

Deze thesis ontwikkelt hoogwaardige numerieke methoden voor convexe optimalisatie, variationele ongelijkheden en speltheorie. Het doel is om de computationele knelpunten in moderne, grootschalige systemen weg te nemen. Door de onderliggende wiskundige structuur te benutten, slaat dit werk een brug tussen abstracte operatortheorie en toepassingen in realtime sturing en strategische besluitvorming.

De eerste kernbijdrage richt zich op het versnellen van first-order methoden voor gladde en niet-gladde convexe optimalisatie. We introduceren adaptieve stapgrootteregels en gekoppelde smoothing-momentum technieken die optimale convergentiesnelheden bereiken. Deze methoden benutten de probleemstructuur optimaal, waardoor ze efficiënt en snel convergeren zonder dat voorafgaande kennis van globale parameters nodig is.

Daarnaast kijkt het onderzoek verder dan enkelvoudige optimalisatie en gebruikt het raamwerk van variationale ongelijkheden om complexe evenwichtsproblemen op te lossen. We stellen projectievrije algoritmen en gespecialiseerde splitting-methoden voor voor situaties waarin traditionele operatoren te zwaar zijn voor de rekenkracht. Deze aanpak maakt het mogelijk om efficiënt evenwichten te berekenen in dynamische spellen en distributioneel robuuste modellen, waarbij rekening wordt gehouden met zowel strategische interacties als data-onzekerheid.

De praktische relevantie blijkt uit diverse praktijktoepassingen en de introductie van een open-source toolkit. Samen vormen deze bijdragen een schaalbaar en robuust raamwerk voor snelle, structuurbewuste besluitvorming in complexe multi-agent systemen.

1

INTRODUCTION

Optimization has become a central mathematical discipline for analyzing, designing, and controlling modern systems. The rapid growth of data-driven control, networked infrastructures, and large-scale computational tools has dramatically increased both the scope and complexity of optimization problems encountered in practice. Whether in machine learning, signal processing, economics, control theory, or scientific modeling, one rarely encounters a closed-form solution or a single-step computation. Instead, one works with iterative methods, equilibrium models, and layered decision processes where optimization serves not as an isolated task but as an underlying operational principle.

A useful starting point is the classical formulation of a smooth convex minimization problem,

$$\min_{x\in\mathbb{R}^n} f(x), \tag{1.1}$$

where f is convex and continuously differentiable with a Lipschitz continuous gradient. This problem captures a wide range of canonical models: least squares estimation, generalized linear models, regression problem, maximum entropy formulations, and more. Classic methods such as gradient descent, accelerated gradient techniques, and quasi-Newton schemes form the standard algorithmic tool for solving problem (1.1). Their appeal stems from a transparent convergence theory, modest memory footprint, and scalability to high dimensions. Despite their conceptual simplicity, convex problems reflect a deeper theme that permeates modern optimization: ***structure matters***.
When the function f is well-conditioned, possesses particular curvature properties, or decomposes across coordinates, significant performance gains can be achieved. Conversely, when these properties fail, classical methods exhibit slow convergence or numerical instability. In this thesis, we aim to leverage the local structure of the underlying objective function in optimization problems and incorporate this structure into classical methods to enhance their efficiency.

1.1. PROBLEM DESCRIPTION

Next, we briefly describe the types of optimization problems considered in this thesis.

1.1.1. FROM SMOOTH TO COMPOSITE MODELS

In many applications, smooth formulations are insufficient because the underlying problem exhibits intrinsic non-smooth structure. Rather than viewing non-smoothness as an obstacle, modern optimization theory embraces composite structure as a modeling advantage. A broad class of problems takes the form

$$\min_{x\in\mathbb{R}^n} F(x) := f(x) + g(x), \tag{1.2}$$

where f is smooth convex and g is proper, convex, and possibly non-differentiable. Typical choices of g include:

- Sparsity-promoting penalties (e.g., ℓ_1 norm),
- Low-rank inducing norms,
- Indicator functions of constraints,
- Separable terms enabling distributed computation.

Composite models unify a diverse collection of problems: Lasso and elastic net regression, total variation denoising, logistic regression with constraints, projection problems in imaging, and numerous machine learning regularizers.

A key insight is that the non-smooth component g can often be handled by evaluating its *proximal operator*,

$$\operatorname{prox}_g(v) := \arg\min_x \left\{ g(x) + \frac{1}{2}\|x - v\|^2 \right\}. \tag{1.3}$$

This operator enables attractive splitting schemes that alternate between handling the smooth part through gradient evaluations and the non-smooth part through proximal steps. Proximal gradient descent, forward–backward splitting, Douglas–Rachford splitting, and ADMM are prototypical examples. Each of these algorithms can be viewed as an instance of a more fundamental operator-theoretic principle, which becomes central later in this introduction.

1.1.2. NON-SMOOTH OPTIMIZATION AS A MODELING NECESSITY

Non-smooth convex optimization is not merely a technical extension; it is fundamental to modern modeling. To illustrate, consider the a constrained regression problem,

$$\min_{x\in\mathbb{R}^n} \left\{ \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1 + \iota_{\mathscr{X}}(x) \right\}, \tag{1.4}$$

where $f(x) = \lambda\|x\|_1$ and $h(x) = \iota_{\mathscr{X}}$ denote norm-1 regularizer and the indicator of a convex feasible set $\mathscr{X}$, respectively. Even this ostensibly simple model blends smooth terms, sparse regularization, and hard constraints, classical smooth and composite optimization tools alone cannot handle such structure. Particularly, the rich structure of nonsmooth problems highlights two emerging philosophical shifts in optimization:

1. A problem should be solved in a way that respects and exploits its structure.

2. Algorithms should accommodate inexactness, irregularity, and equilibrium behaviour.

Several studies have been devoted to developing methods for computing the prox-operator (1.3) of a sum of two (multiple) functions, see the recent work [1], [2], [3] and the references therein. These methods typically provide various assumptions under which we have

$$\text{prox}_{f+h} = \text{prox}_f \circ \text{prox}_h, \tag{1.5}$$

where "$\circ$" denotes the mapping composition. However, these conditions are still restrictive, and the prox-operator of the sum of two prox-friendly functions may not have a closed-form solution and may be in general computationally demanding (e.g., sparse regression and semi-definite programming). Motivated by this, recent works adopt methods, e.g., smoothing techniques, designed to handle multiple nonsmooth terms. In (1.4), both functions f and h are possibly nonsmooth but prox-friendly.

1.1.3. VARIATIONAL INEQUALITIES: A UNIFYING FRAMEWORK

Beyond minimization problems lies a broader class of models: variational inequalities (VIs). A VI seeks $x^\star \in \mathcal{X}$ satisfying

$$\langle F(x^\star), x - x^\star \rangle + g(x) - g(x^*) \geq 0 \qquad \forall x \in \mathcal{X}, \tag{1.6}$$

under the standard assumptions that $\mathcal{X} \subseteq \mathbb{R}^n$ is a closed, convex set; $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is a (generally set-valued) monotone and Lipschitz operator, g is a proper lower semicontinuous (lsc) convex function, and the solution set of (1.6) is nonempty. Variational inequalities generalize optimality conditions of convex optimization, saddle-point problems, Nash equilibria, and implicit dynamic models.
As an example, consider the composite minimization problem (1.2), where f is a convex and smooth function and g is a proper lsc convex (and possibly nonsmooth) function. Via the KKT conditions, this problem can be written as (1.6) with $F = \nabla f$ and the same g in (1.6) [4]. Another common problem in optimization and control theory is the min-max problem. For example, consider the convex-concave saddle point problem:

$$\min_{y \in \mathbb{R}^n} \max_{z \in \mathbb{R}^m} g_1(y) + f(y,z) - g_2(z),$$

where g_1 and g_2 are proper lsc convex functions and $f(y,z)$ is a smooth convex-concave function in y and z, respectively. By using first-order optimality conditions, we can rewrite this problem as in (1.6) with the following variables:

$$x = \begin{pmatrix} y \\ z \end{pmatrix}, \quad F = \begin{pmatrix} \nabla_y f(y,z) \\ -\nabla_z f(y,z) \end{pmatrix}, \quad g(x) = g_1(y) + g_2(z).$$

Furthermore, in many applications of reinforcement learning and game theory, we need to solve a fixed-point problem. For instance, Markov decision processes (MDPs) are a powerful modeling framework in reinforcement learning, where we should solve a fixed point problem, $Tx = x$, for some finite dimensional operator T, that is (1.6) with $F = \text{Id} - T$ and $g(x) = 0$ [5].

This connection bridges convex optimization and monotone operator theory: many widely used optimization algorithms can be reinterpreted as methods for solving (1.6). Examples include:

- (adaptive/accelerated) Forward–backward method;
- Douglas–Rachford splitting method;
- Extragradient and mirror-prox methods for VIs;
- Projection-free (Frank-Wolfe) method for strongly monotone VIs;

Thus, VIs act as a *conceptual umbrella* under which classical minimization, composite optimization, and equilibrium modeling can be jointly analyzed.

1.2. OUTLINE AND CONTRIBUTIONS

The thesis is composed of three parts. We present the focus of each part and a summary of each chapter next, while a schematic representation of the thesis structure is depicted in Figure 1.1.

1.2.1. PART I: FAST CONVEX OPTIMIZATION

In the first part of this thesis, we focus on the optimization of convex functions. We begin with smooth and composite optimization in Chapter 2, where we introduce a new line-search first-order method for solving this class of problems. Next, in Chapter 3, we consider more general fully nonsmooth convex optimization problems using smoothing techniques. The detailed contributions of these chapters are as follows:

Chapter 2: We present our contributions on a novel linesearch method for stepsize selection in solving smooth and composite problems (1.1)–(1.2) using (accelerated) first-order gradient descent. This chapter is based on work by R. Rahimi Baghbadorani, S. Grammatico, and P. Mohajerin Esfahani, "A New Line Search for Accelerated Composite Minimization" [6]. The main contribution of Chapter 2 are summarized as follows:

1. **New linesearch for convex composite minimization.** Following the same spirit of the adaptive stepsize and linesearch rules, this study proposes a novel stepsize rule without the knowledge of the global smoothness constant for solving convex composite minimization problems (Figure 2.1). Using a Lyapunov-based argument, we show that the proposed rule enjoys the optimal worst-case complexity bound of $\mathcal{O}(k^{-1})$ and $\mathcal{O}(k^{-2})$ in both cases of nonaccelerated (Section 2.1) and accelerated (Section 2.2) algorithms, respectively. These results are developed for a general class of composite convex functions and are optimal in the sense that they match the theoretical lower bounds of the class of first-order algorithms for this class of functions.

2. **Numerical results with applications in operations research.** We numerically benchmark the performance of our proposed algorithms against state-of-the-art meth-

ods across three major problem classes: (i) smooth minimization (logistic regression, quadratic programs, log-sum-exponential), (ii) composite minimization (ℓ_1-regularized least-squares, ℓ_1-constrained least-squares, and ℓ_1-regularized logistic regression), and (iii) non-convex minimization (cubic minimization), see Section 2.3. These classes cover a wide range of operations research and management applications, including portfolio optimization, discrete choice models, classification and demand modeling, combinatorial decision frameworks, sparse recovery, feature selection, and trust-region type algorithms.

Chapter 3: We propose an adaptive accelerated smoothing technique for a nonsmooth convex optimization problem, where the smoothing update rule is coupled with the momentum parameter. We further extend the approach to cases where the objective function is the sum of two or more nonsmooth functions, which are widely used in optimization-based control, system identification, and machine learning. This chapter is based on R. Rahimi Baghbadorani, S. Grammatico, and P. Mohajerin Esfahani, "Locally Linear Convergence for Nonsmooth Convex Optimization via Coupled Smoothing and Momentum" [7]. The main contributions of Chapter 3 are summarized as follows:

1. **Global optimal sublinear convergence:** We introduce an algorithm with an adaptive smoothing parameter coupled with the momentum term (Alg. 1). When the smoothing rule is modified to stay away from zero, we provide a global sublinear convergence rate of $\mathcal{O}(1/\varepsilon)$ that matches the optimal worst-case complexity bound for optimizing this class of nonsmooth functions (Theorem 3.2.3).

2. **Locally linear convergence:** When the nonsmooth term $f(x)$ meets a so-called ∞-locally strong convexity condition, we show that Alg. 1 enjoys a local linear convergence rate (Theorem 3.2.4). Combined with the global convergence result from the previous section, this implies that an appropriate initial condition for Alg. 1 ensures a transient optimal sublinear convergence rate followed by an asymptotic linear convergence rate.

3. **Multiple nonsmooth terms:** The proposed algorithm allows for multiple nonsmooth terms. Such settings can be computationally challenging as the "prox-friendly" property, a key feature in nonsmooth optimization, is not necessarily additive (see Section 3.1.1). Important applications falling into this category include model-free fault diagnosis, nonsmooth model predictive control, sparse regression and sparse semidefinite programming, which are the examples investigated in our numerical section to validate the performance of our proposed algorithm (Section 3.4).

1.2.2. PART II: FAST EQUILIBRIUM SEEKING

In the second part of this thesis, we study a broader class of optimization problems, namely variational inequalities. Chapter 4 introduces a projection-free method for solving strongly monotone variational inequalities. Chapter 5 then analyzes applications of variational inequalities to linear–quadratic dynamic games and iterative solution methods, with particular emphasis on splitting techniques. Finally, Chapter 6 investigates the

formulation of Wasserstein distributionally robust Nash equilibrium seeking as a variational inequality problem. The detailed contributions of each chapter are presented below.

Chapter 4: We study variational inequality problems and propose a projection-free method for solving strongly monotone VIs, suitable for settings in which projection evaluations are costly. This chapter is based on "A Frank-Wolfe Algorithm for Strongly Monotone Variational Inequalities" [8]. The main contributions of Chapter 4 are summarized as follows:

1. **Projection-free method in variational inequality:** Following the footsteps of the accelerated Nesterov's technique for solving strongly monotone variational inequalities [9] and the FW algorithm for solving saddle point problems [10], we propose a novel accelerated algorithm with FW method as an oracle for solving the strongly monotone VIP and provide a non-asymptotic convergence rate (Remark 4.2.3).

2. **Numerical justification (case study: Traffic Assignment Problem):** To validate the theoretical results and efficacy of projection-free methods in solving strongly monotone VI's, we implement the proposed method in the traffic assignment problem, an important application in transportation and operation research [11], [12], [13], where the complexity of the problem (the corresponding set) increases exponentially with the number of variables (Section 4.3).

Chapter 5: We first study constrained linear dynamic games with quadratic objectives and formulate them as affine variational inequalities. Moreover, we propose a splitting method to solve the resulting variational inequality and benchmark its performance through numerical experiments in an automated-driving scenario. This chapter is based on R. Rahimi Baghbadorani, E. Benenati, and S. Grammatico, "A Douglas–Rachford Splitting for Solving Monotone Affine Variational Inequalities in Linear–Quadratic Dynamic Games" [14]. The main contributions are summarized as follows:

1. **Affine variational inequality for open-loop Nash equilibrium:** We define a strongly monotone affine VI whose solution, under suitable assumptions, yields an infinite-horizon open-loop Nash equilibrium of a linear–quadratic dynamic game (Lemma 5.1.5). We show that a closed-form solution exists when the initial state lies in a neighborhood of the origin.

2. **Douglas–Rachford spilliting method:** Leveraging the linear structure of the VI, we develop a tailored iterative solver based on the Douglas–Rachford splitting method and establish linear convergence for the corresponding affine VI (Section 5.2).

3. **Intersection dynamic game:** We adopt the proposed splitting method to compute the control inputs of a receding-horizon controller derived from a VI formulation of a linear–quadratic dynamic game. The numerical case study considers autonomous vehicles crossing an intersection while maintaining safe distance and speed (Section 5.3).

Chapter 6: We show that heterogeneous Wasserstein distributionally robust Nash games with private data reduce from an infinite-dimensional formulation to a finite-dimensional Nash equilibrium. The resulting problem admits an equivalent finite-dimensional variational inequality with matching solution sets. The content of this chapter is based on G. Pantazis, R. Rahimi Baghbadoranio, and S. Grammatico, "Nash equilibrium seeking for a class of quadratic-bilinear Wasserstein distributionally robust games" [15]. The key contributions of this chapter are outlined below:

1. **Heterogeneous Wasserstein distributionally robust game:** We analyze heterogeneous data-driven Wasserstein distributionally robust games with quadratic–bilinear agent costs. Each agent behaves selfishly, optimizing an individual objective dependent on a private uncertain parameter. By allowing agents to form personalized ambiguity sets with distinct distributions and Wasserstein radii, we generalize classical optimization-based models to a fully game-theoretic setting. This framework yields solution sets that can differ substantially from those obtained under common ambiguity sets in prior work (Theorem 1).

2. **Scalable robust Nash equilibrium reformulation:** We reformulate the game as a robust Nash equilibrium problem and establish its relationship to the corresponding distributionally robust Nash equilibria. For this class of games, we show that the inner maximization can be solved without epigraphic variables, avoiding unshared coupling constraints and reducing computational complexity. This leads to, to our knowledge, the first data-scalable distributionally robust game reformulation that directly exploits problem structure (Proposition 6.2.7).

3. **Variational inequality solution methods:** The robust Nash equilibrium problem is further recast as a variational inequality (VI). Unlike analogous optimization settings where monotonicity may hold, the resulting mapping is generally non-monotone due to heterogeneous ambiguity sets and cost functions. Nonetheless, we show that the problem can be solved efficiently using aGRAAL and a hybrid variant, which exhibit near-linear convergence in several cases and remain stable as sample size grows. We demonstrate the practical value of this approach on a portfolio allocation game incorporating market uncertainty and behavioral interactions (Section 6.3).

1.2.3. PART III: APPLICATION TO MULTI-AGENT OPTIMIZATION AND CONTROL

This part consists of Chapters 7 and 8, which present the use of optimization problems and related computational methods to model and solve real-world applications. We also introduce a Python package implementing several iterative methods for monotone variational inequality problems.

Chapter 7: We introduce a bilevel optimization framework to model an embedded-oriented Energy Community Management (ECM) architecture that maximizes shared energy, minimizes individual costs, and enhances overall economic benefits. By reformulating the upper- and lower-level problems via their KKT conditions, we develop a

modified ADMM algorithm with Nesterov acceleration to achieve faster convergence. This chapter is based on G. Ferro, S. Grammatico, L. Parodi, R. Rahimi Baghbadoranio, and M. Robba, "An embedded accelerated decentralized optimization algorithm with application to energy communities" [16]. The main contributions of Chapter 7 are summarized as follows:

1. **Decentralized ECM:** We propose a decentralized ECM formulation that jointly maximizes shared energy and minimizes individual user costs, balancing collective welfare and economic efficiency within renewable energy communities (RECs), see Section 7.1.

2. **Accelerated privacy-preserving ADMM:** We design a fast, privacy-preserving accelerated ADMM algorithm for the QP-reformulated ECM problem, ensuring rapid convergence, scalability, and confidentiality while remaining compatible with low-resource embedded platforms (Section 7.2).

3. **Embedded real-time implementation:** We provide an embedded implementation with experimental validation on low-power microcontrollers (ODROID-N2L and H3+), demonstrating deterministic computation times, high numerical accuracy, and real-time applicability (Section 7.3).

Chapter 8: We present `monviso` (`monotone variational inequalities solver`), a novel open-source Python package for solving monotone variational inequalities, and illustrate representative use cases across control, optimization, and machine learning.

1.3. OTHER CONTRIBUTIONS AND CURRENT WORK

In addition to the works reported in Chapters 2–8, we have also worked on the following project during my PhD.

Inverse Game Theory: In inverse optimization (IO), the objective is to recover the decision-making process of an expert agent that, given an exogenous signal, produces a corresponding action. A closely related challenge arises in equilibrium modeling, common in game theory and transportation, where the model inputs are often difficult to estimate, yet the equilibrium outcomes they generate are typically observable. This analogy motivates the use of IO principles in equilibrium settings, where observable equilibria can be exploited to infer underlying model parameters. By combining ideas from inverse optimization with variational inequality theory, inverse game theory aims to recover the structure of a game, in particular the objective functions of the players. Recent works [17] study inverse (strongly) monotone variational inequalities associated with generalized Nash equilibria and employ first-order loss functions, in both parametric and nonparametric settings, to learn the pseudogradient operator corresponding to the players' objectives. Advances in inverse optimization [18], [19], [20], motivate the adoption of more efficient learning losses, such as suboptimality loss, to reduce computational complexity and improve the accuracy of parameter recovery in inverse games. Moreover, existing

nonparametric approaches [17] do not guarantee monotonicity of the learned pseudo-gradient operator. Our goal is to develop a new nonparametric (kernel-based) framework with statistical guarantees that ensures monotonicity of the learned operator while enabling accurate and efficient estimation of the underlying game model.

Online Stackelberg Game: A class of games widely used in energy networks, economics, and multi-agent systems is the Stackelberg game, where optimal decisions are made sequentially: the leader commits to a strategy first, and the followers subsequently respond optimally. Recent studies assume that the leader has access to the gradient of the followers' objective functions and therefore apply hypergradient-based first-order methods to compute the leader's optimal strategy [21]. In practice, however, the leader typically lacks knowledge of the followers' utility functions and strategy spaces. This limitation motivates the zero-order methods proposed in [22], where the leader constructs gradient estimators of the followers' utilities using only function evaluations. Although effective, these methods suffer from slow convergence and require a large number of iterations to approach the optimal strategy. A promising direction for improving efficiency and convergence speed is to estimate the followers' parameters from collected data using zero-order measurements, and then allow the leader to centrally solve an approximated problem within a local region. This hybrid approach can significantly accelerate convergence while mitigating the limitations of purely zero-order optimization.

1

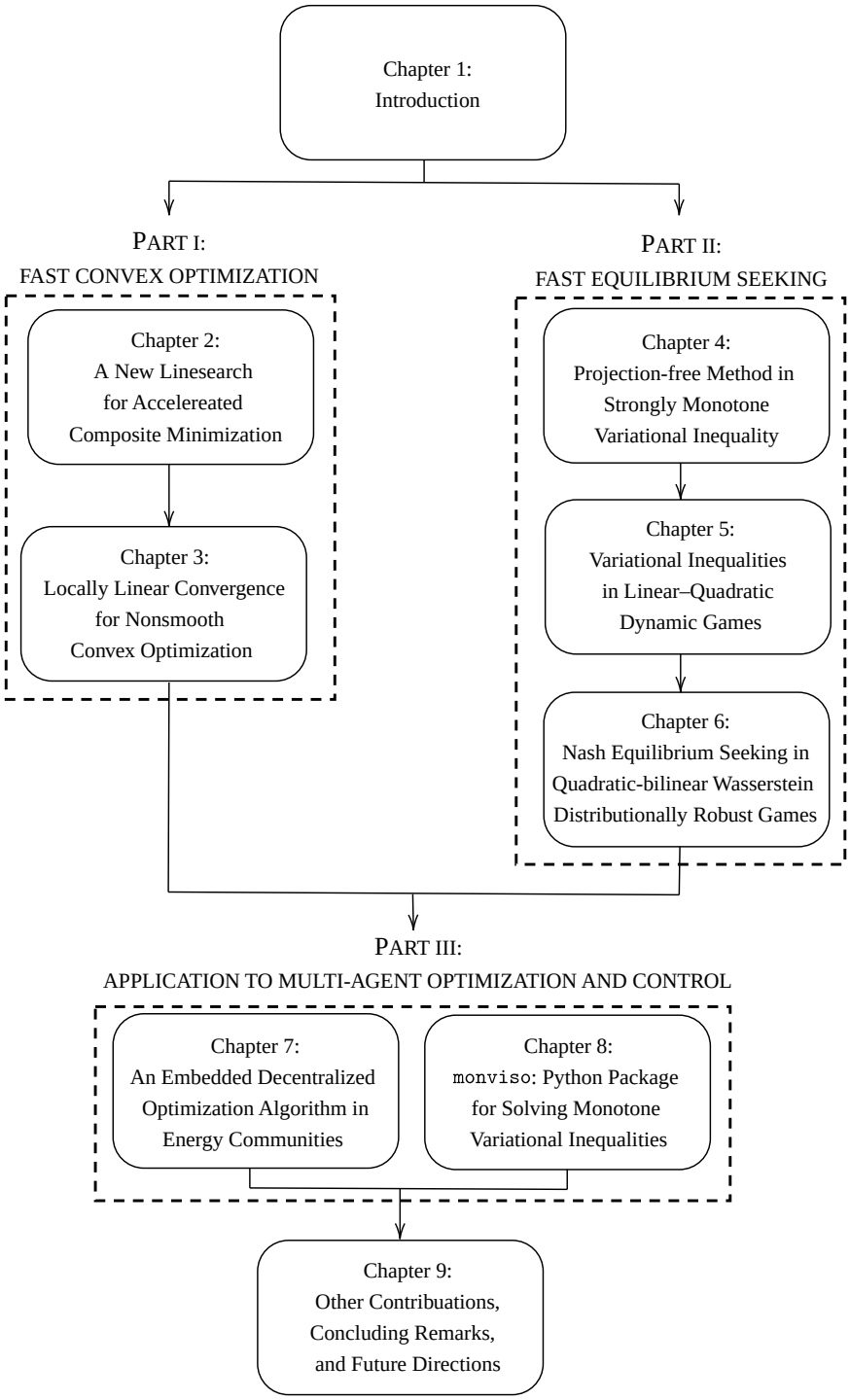
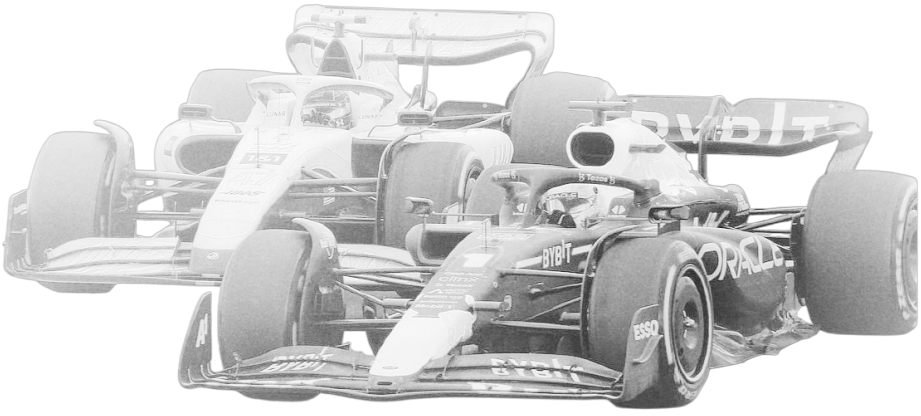


Figure 1.1: Roadmap of the thesis.

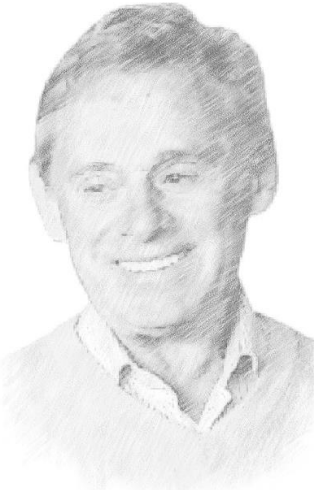
I

Fast Convex Optimization



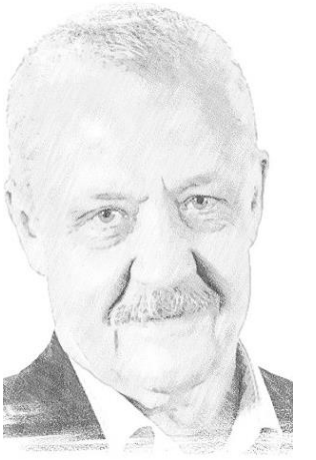
If I have seen further it is by
standing on the shoulders of Giants

Isaac Newton



Boris Poljak

Arkady Nemirovsky



Yurii Nesterov

This part builds on the foundational contributions of leading researchers in convex optimization, notably Boris Poljak, Arkady Nemirovsky, and Yurii Nesterov. Their seminal works have shaped the core concepts and methodologies discussed herein.

2

A New Linesearch for Accelerated Composite Minimization

The choice of the stepsize in first-order convex optimization is typically based on the smoothness constant and plays a crucial role in the performance of algorithms. Recently, there has been a resurgent interest in introducing adaptive stepsizes that do not explicitly depend on smooth constant. In this chapter, we propose a novel linesearch stepsize rule based on function evaluations (i.e., zero-order information) that enjoys provable convergence guarantees for both accelerated and non-accelerated gradient descent. We further discuss the similarities and differences between the proposed stepsize regimes and the existing stepsize rules (including Polyak and Armijo).

We numerically benchmark the performance of our proposed algorithms against state-of-the-art methods across three major problems classes of (1) smooth minimization (logistic regression, quadratic programs, log-sum-exponential, and smooth max-cut relaxation) (2) composite minimization (ℓ_1-regularized least-squares, ℓ_1-constrained least-squares, and ℓ_1-regularized logistic regression), and (3) non-convex minimization (cubic minimization). These classes include a wide range of operations research and management applications such as portfolio optimization, discrete choice models, sparse classification and feature selections, high-order optimization and trust-region subproblems.

Thanks to its simple implementation, scalability and applicability, gradient descent (GD) is arguably the most popular algorithm to efficiently solve large-scale optimization problems, which finds a wide range of applications in operations research [23], [24], [25], machine learning [26], [27], control and system identification [28], [29]. The main challenge in using gradient descent is the choice of the right stepsize, which has a considerable impact on the convergence speed. Several works have studied this choice, yet either the fastest possible convergence rate cannot be guaranteed or an expensive linesearch at each iteration is needed [30], [31], [32], [33]. In this paper, we address these issues by introducing a novel and simple linesearch method.

Let us consider the unconstrained convex minimization problem $\min_{x\in\mathbb{R}^n} f(x)$, where f is a smooth convex function. The usual iterative algorithm is of the form

$$x_{k+1} = x_k + \lambda_k d_k. \tag{2.1}$$

where λ_k and d_k represent the stepsize and the descent direction, respectively. In the last decades, many algorithms have been developed for choosing $(\lambda_k)_{k\in\mathbb{N}}$ and $(d_k)_{k\in\mathbb{N}}$ in (2.1) either statically or adaptively. We review some classes of design choices in the following.

A key object playing an important role in determining most of the existing stepsize rules at the iteration x_k is the objective function along the update direction $d_k = -\nabla f(x_k)$ defined by

$$\phi_k(\lambda) := f(x_k + \lambda d_k). \tag{2.2}$$

When the function f is convex, then the scaler function $\phi_k : \mathbb{R} \to \mathbb{R}$ defined in (2.2) is also convex. Several classical stepsize rules defined via ϕ_k in (2.2) are the following.

(i) GD with constant stepsize [32]: The simplest stepsize rule is the constant $\lambda_k = 1/L$, where L is the so-called smoothness parameter (Lipschitz continuity of $\nabla f(x)$. The convergence rate of the suboptimality $f(x_k) - f(x^*)$ with the constant stepsize is $\mathcal{O}(k^{-1})$.

(ii) GD with exact linesearch [32]: Another classical stepsize is to find the optimal stepsize minimizing the function ϕ_k along the direction d_k is $\lambda_k = \arg\min_\lambda \phi_k(\lambda)$ which involves a scalar optimization problem that requires evaluation of the original function f.

(iii) GD with approximated stepsize [32]: Due to the complexity of finding the stepsize by solving the exact linesearch minimization, alternative stepsize rules have been proposed that only require ensuring an inequality such as $\phi_k(\lambda) \le \phi_k(0) + c\phi_k'(0)$ for some predefined constant $c > 0$. Among others, Armijo [34], Wolf, and strong Wolf linesearch [32] are backtracking linesearch methods falling into this category.

$$\text{Armijo}: \quad \phi_k(\lambda) \le \phi_k(0) + c_1\lambda\phi_k'(0),\ c_1 \in (0,1). \tag{2.3a}$$

$$\text{Wolf}: \quad \phi_k'(\lambda) \ge c_2\phi_k'(0),\ c_2 \in (c_1,1) + \text{Armijo condition}. \tag{2.3b}$$

$$\text{strong Wolf}: \quad |\phi_k'(\lambda)| \le c_2\phi_k'(0),\ c_2 \in (c_1,1) + \text{Armijo condition}. \tag{2.3c}$$

The suboptimality convergence rate of these algorithms is $\mathcal{O}(k^{-1})$. There are also several stepsize rules approximating the smooth constant locally based on the zero-order oracle's information (i.e., the objective function evaluation) [31], [35], [36], [37]. In the following, we review two important stepsize rules that fall into this category:

(iv) GD with the Polyak stepsize [35]: One of the classic adaptive stepsize rules is the Polyak stepsize, which uses the zero-order oracle's information at each iteration to locally approximate the smooth constant of the objective function:

$$\lambda_k = \frac{f(x_k) - f(x^*)}{\|\nabla f(x_k)\|^2}. \tag{2.4}$$

Although the Polyak stepsize offers optimal convergence rates for GD (2.1) when the update direction is set to $d_k = -\nabla f(x_k)$ [38], it cannot be applicable in problems where the optimal function value $f(x^*)$ is not available. We will get back to the Polyak stepsize and its relation to our proposed rule in Subsection 2.1.3 (see equations (2.18)).

(v) GD with adaptive stepsize [31]: Recently, the next adaptive stepsize has been proposed based on the idea of approximating the smoothness constant locally:

$$\lambda_k = \min\left\{\sqrt{1+\theta_{k-1}}\lambda_{k-1}, \frac{\|x_k - x_{k-1}\|}{2\|\nabla f(x_k) - \nabla f(x_{k-1})\|}\right\}, \quad \theta_k = \frac{\lambda_k}{\lambda_{k-1}}. \tag{2.5}$$

The second term on the left-hand side of (2.5) represents this local approximation as the inverse of the smoothness constant, while the first term ensures a required convergence rate of this parameter. The theoretical sub-optimality convergence rate for the stepsize rule (2.5) is $\mathcal{O}(k^{-1})$, however, the adaptive (i.e., state-dependent) nature of (2.5) has the advantage of (a) lifting the knowledge of the smoothness constant for the objective function, and (b) improving the practical convergence rate when the local smoothness is smaller than the global constant one. Next, we consider methods with a dynamic direction d_k in (2.1). The update rules in this class of methods have an extra term, which is called momentum, and allows the descent direction to have some inertia in the search space [30].

(vi) AGD with constant stepsize [39]:

$$\begin{cases} \beta_k = \dfrac{1+\sqrt{1+4\beta_{k-1}^2}}{2}, \ \gamma_k = \dfrac{1-\beta_k}{\beta_{k+1}}, \\ \lambda_k = \dfrac{1}{L}, \ d_k = -\gamma_k\left(\dfrac{\lambda_{k-1}}{\lambda_k}d_{k-1} - (1+\dfrac{1}{\gamma_k})\nabla f(x_k) + \dfrac{\lambda_{k-1}}{\lambda_k}\nabla f(x_{k-1})\right), \end{cases}$$

where L is the smooth constant of f. The descent direction in this method uses previous iteration values to accelerate the convergence rate, which is $\mathcal{O}(k^{-2})$ for $f(x_k) - f(x^*)$.

(vii) AGD with adaptive stepsize [31]: A heuristic version of adaptive stepsize (2.5) for accelerated methods is also proposed in [31, Section 3]

$$\begin{cases} \lambda_k = \min\left\{\sqrt{1+\dfrac{\theta_{k-1}}{2}}\lambda_{k-1}, \dfrac{\|x_k - x_{k-1}\|}{2\|\nabla f(x_k) - \nabla f(x_{k-1})\|}\right\}, \ \theta_k = \dfrac{\lambda_k}{\lambda_{k-1}}, \\ \Lambda_k = \min\left\{\sqrt{1+\dfrac{\Theta_{k-1}}{2}}\Lambda_{k-1}, \dfrac{\|\nabla f(x_k) - \nabla f(x_{k-1})\|}{2\|x_k - x_{k-1}\|}\right\}, \ \Theta_k = \dfrac{\Lambda_k}{\Lambda_{k-1}}, \\ \beta_k = \dfrac{\sqrt{1/\lambda_k} - \sqrt{\Lambda_k}}{\sqrt{1/\lambda_k} + \sqrt{\Lambda_k}}, \\ d_k = \beta_k\left(\dfrac{\lambda_{k-1}}{\lambda_k}d_{k-1} - (1+\dfrac{1}{\beta_k})\nabla f(x_k) + \dfrac{\lambda_{k-1}}{\lambda_k}\nabla f(x_{k-1})\right). \end{cases}$$

This adaptive rule presents promising numerical performance while there is no formal theoretical guarantee explaining this interesting performance. This theoretical gap toward accelerated techniques is one of the motivations of this study.

We also consider the composite minimization problem

$$\min_{x\in\mathscr{X}} F(x) = \min_{x\in\mathscr{X}} f(x) + h(x) \tag{2.6}$$

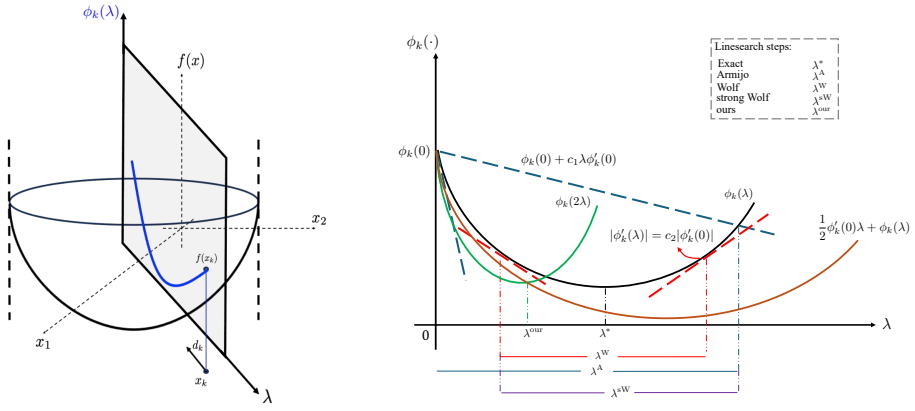
where $f\colon \mathscr{X}\to\mathbb{R}$ is a convex and smooth function, and $h\colon \mathscr{X}\to\mathbb{R}$ is convex and possibly non-smooth but prox-friendly (technical details are postponed to Section 2.1). There are several methods for convex composite minimization in the existing literature. Among them, subgradient descent and mirror descent are two well-known classic methods that suffer from slow convergence rate $\mathscr{O}(k^{-1/2})$ [40], [41], [42], [43]. Following the seminal works by Nesterov, one can exploit the structure of the nonsmooth part in the objective function and deploy a first-order accelerated method. ISTA and FISTA [44], which are extensions of GD and NAGD, respectively, offer a faster convergence rate compared to subgradient and mirror descent. However, these methods require knowledge of the smooth constant associated with the smooth part, f in (2.6). Additionally, there are various backtracking techniques available to approximate the smooth constant, but they often tend to be conservative, resulting in larger values than the actual smooth constant. This conservatism can impact the convergence speed. The authors in [45], [46] propose an algorithm that enjoys the local smoothness of f to determine the stepsize. However, despite its simplicity and efficacy, the convergence guarantee is still the suboptimal rate of $\mathscr{O}(k^{-1})$. More recently, the authors in [47] propose a fully adaptive stepsize with momentum acceleration and a convergence rate of $\mathscr{O}(k^{-2})$. However, the algorithm requires several hyperparameters, which may affect the performance in practice. Inspired by this, we develop a hyperparameter-free algorithm for convex composite minimization problems (2.6) with the optimal theoretical convergence rate $\mathscr{O}(k^{-2})$ while maintaining a reasonable level of computational complexity.

Contributions. Following the same spirit of the adaptive stepsize and linesearch rules, this study proposes a novel stepsize rule without the knowledge of the global smoothness constant for solving convex composite minimization problems. Using a Lyapunov-based argument, we show that the proposed rule enjoys the optimal worst-case complexity bound of $\mathscr{O}(k^{-1})$ and $\mathscr{O}(k^{-2})$ in both cases of nonaccelerated and accelerated algorithms, respectively. These results are developed for a general class of composite convex functions and are optimal in the sense that they match the theoretical lower bounds of the class of first-order algorithms for this class of functions. Our convergence results are summarized in Table 2.1 where our update direction $d_k = -G^f_{\lambda h}(x_k)$ is our gradient mapping (see the definition (2.10) for more details concerning the gradient mapping). Let us note that the results in Table 2.1 are reported for the general composite case (2.6). If the nonsmooth term is $h(x) = 0$, then the gradient mapping reduces to $G^f_{\lambda h}(x) = \nabla f(x)$. In the rest of the introduction, we briefly discuss our proposed stepsize rule in this special case of a smooth convex function and provide a geometrical illustration in comparison with the existing rules reviewed above. **Partial result**: If f is a smooth and convex function, our proposed stepsize rule is defined as

$$\lambda_k^{\text{our}} := \arg\max_{\lambda}\left\{\lambda \mid \phi_k(2\lambda) \le \frac{1}{2}\phi_k'(0)\lambda + \phi_k(\lambda)\right\} \tag{2.7}$$

Table 2.1: Summary of the main results of this paper.

Algorithm	Problem	Stepsize rule	Convergence rate
nonaccelerated (Theorem 2.1.3)	composite	$\phi(2\lambda) \le \phi(\lambda) - \lambda\langle G^f_{\lambda h}(x), \nabla f(x)\rangle + \frac{\lambda}{2}\|G^f_{\lambda h}(x))\|^2$	$\mathcal{O}(k^{-1})$
nonaccelerated (Corollary 2.1.6)	smooth	$\phi(2\lambda) \le \phi(\lambda) + \frac{\lambda}{2}\phi'(0)$	$\mathcal{O}(k^{-1})$
accelerated (Theorem 2.2.1)	composite	$\phi(2\lambda) \le \phi(\lambda) - \lambda\langle G^f_{\lambda h}(x), \nabla f(x)\rangle + \frac{\lambda}{2}\|G^f_{\lambda h}(x))\|^2$	$\mathcal{O}(k^{-2})$
accelerated (Corollary 2.2.2)	smooth	$\phi(2\lambda) \le \phi(\lambda) + \frac{\lambda}{2}\phi'(0)$	$\mathcal{O}(k^{-2})$



(a) $\phi_k(\lambda)$. (b) Stepsize rules.

Figure 2.1: Geometric interpretation of different stepsize rules using $\phi_k(\lambda)$ defined in (2.2).

where the function ϕ_k is defined in (2.2) and visualized in Figure 2.1a. Recall that in the case of non-accelerated methods, the update direction is simply $d_k = -\nabla f(x)$. We emphasize that the proposed linesearch above only requires additional zeroth-order information, namely the objective function evaluation, while the first-order information is the gradient of the function which is already computed in the previous step. Figure 2.1b illustrates a geometric interpretation of the proposed stepsize rule (2.7), together with the existing literature reviewed in the preceding section. For further geometrical interpretation, particularly in case of accelerated methods, we refer to [48].

The paper is organized as follows: In Section 2.1, we present the theoretical results for the non-accelerated adaptive stepsize algorithm. Section 2.2 provides technical proofs for the accelerated adaptive stepsize case. Some implementation techniques and illustrative examples to show the efficiency of our approaches are presented in Section 2.3. Finally, the conclusion and further discussion are given in Section 2.4.

Notation. Let $\mathcal{X}$ be the finite-dimensional real vector space with the standard inner product $\langle\cdot,\cdot\rangle$ and ℓ_p-norm $\|\cdot\|_p$ (by $\|\cdot\|$, we mean the Euclidean standard 2-norm). If f is differentiable, then $\nabla f(x)$ represents the gradient of f at x. The function f is L-smooth, or equivalently the gradient of f is L-Lipschitz, if for all $x, y \in \mathbb{R}^d$ one of the following

inequalities is satisfied:

$$\|\nabla f(x) - \nabla f(y)\| \le L\|x - y\|, \tag{2.8a}$$

$$f(y) \le f(x) + \langle \nabla f(x), y - x\rangle + \frac{L}{2}\|x - y\|^2. \tag{2.8b}$$

Furthermore, f is locally smooth if it is smooth over any compact set of its domain, see [49, Chap. 2] for more details. The prox operator of a convex function $h\colon \mathscr{X} \to \mathbb{R}$ is defined as

$$\operatorname{prox}_h(x) = \arg\min_u h(u) + \frac{1}{2}\|u - x\|^2. \tag{2.9}$$

A function is "prox-friendly" if the prox operator in (2.9) is (computationally or explicitly) available. We also denote the gradient mapping of two convex functions f and h by

$$G^f_{\lambda h}(x) := \frac{1}{\lambda}\Big(x - \operatorname{prox}_{\lambda h}\big(x - \lambda\nabla f(x)\big)\Big), \tag{2.10}$$

where λ is a positive scalar and has a stepsize interpretation. The gradient mapping is available if f is differentiable and h is prox-friendly.

2.1. NON-ACCELERATED ADAPTIVE STEPSIZE

We present our algorithm and its convergence analysis for the non-accelerated case in this section. From a high-level viewpoint, the proposed rule follows the proximal gradient descent method with the difference in the choice of stepsize λ, which depends on the previous state and its gradient.

2.1.1. PRELIMINARIES

We first proceed with some assumptions and lemmas that will be used throughout the paper. We note that the classical Cauchy-Schwartz and convexity inequalities are the two useful tools in our analysis and most of the results build on the seminal works by Nesterov [39] and Polyak [30]. The following assumptions hold throughout this study.

Assumption 2.1.1 (Convex regularity)**.** *The function F in (2.6) admits bounded level sets, whereby the terms $f(x)$ and $h(x)$ are smooth and prox-friendly, respectively.*

The next lemma indicates several properties of the composite minimization (2.6) that are central to develop the algorithms in this paper.

Lemma 2.1.2 (Gradient mapping)**.** *Let $G^f_{\lambda h}(x)$ be the gradient mapping defined in (2.10) for a smooth convex function f, a possibly nonsmooth function h, and a positive constant λ in $\mathbb{R}_+$.*

(i) **Stationary condition:** *The vector x is a minimizer of (2.6) if and only if $G^f_{\lambda h}(x) = 0$.*

(ii) **Convexity-like inequality:** *For all x, y in $\mathbb{R}^d$ we have*

$$h\Big(x - \lambda G^f_{\lambda h}(x)\Big) \le h(y) - \langle G^f_{\lambda h}(x) - \nabla f(x), y - (x - \lambda G^f_{\lambda h}(x))\rangle.$$

(iii) **Increment bound:** *Considering* $x^+ = x - \lambda G^f_{\lambda h}(x)$, *for any* $z \in \mathbb{R}^d$ *we have*

$$F(x^+) - F(z) \le \langle x^+ - x, \nabla f(x^+) - \nabla f(x) + \frac{1}{2} G^f_{\lambda h}(x) \rangle - \Big(\frac{1}{2\lambda} \|x^+ - x\|^2 + \frac{1}{\lambda} \langle x^+ - x, x - z \rangle \Big).$$

(iv) **Zero-order linsearch:** *Let the update direction* $d_k = -G^f_{\lambda h}(x)$. *If* λ *satisfies*

$$\phi_k(2\lambda) \le \phi_k(\lambda) - \lambda \langle G^f_{\lambda h}(x), \nabla f(x) \rangle + \frac{\lambda}{2} \|G^f_{\lambda h}(x))\|^2, \tag{2.11}$$

we then have $\langle x^+ - x, \nabla f(x^+) - \nabla f(x) + \frac{1}{2} G^f_{\lambda h}(x) \rangle \le 0$.

Before providing the technical proof of the lemma, let us offer some insights into why the above four statements will help us develop algorithms: The convex-like inequality (ii) is a gradient mapping intrinsic characteristic that is particularly useful for controlling the increment of the original function F in (2.6) along the direction of $d_k = -G^f_{\lambda h}(x)$. In both cases of the non-acceleration and acceleration approaches, the increment inequality (iii) is crucial in proving our convergence analysis. The increment bound (iii) allows for the inclusion of a momentum term that emerges in the acceleration dynamics. The first term on the right-hand side of (iii) is an unfavorable term which is often targeted via the stepsize rule to remain negative. Checking the linesearch in (iv) guarantees the negativity of this unfavorable term, which only requires additional zeroth-order information of function F while the gradient (first-order) information of the gradient mapping $G^f_{\lambda h}(x)$ has already been computed. This is a crucial feature, making a stepsize rule computationally more useful in practice. We conclude this section with the proof of the lemma.

Proof of Lemma 2.1.2. We provide the proof of each part separately as follows:
Part (i): Assume $\hat{x}$ is the minimizer of the composite minimization (2.6). Then, by the definition of prox operator (2.9), we have the following equivalent implications

$$G^f_{\lambda h}(\hat{x}) = 0 \iff \hat{x} = \text{prox}_{\lambda h}\big(\hat{x} - \lambda \nabla f(\hat{x})\big) \iff (\hat{x} - \lambda \nabla f(\hat{x})) - \hat{x} \in \lambda \partial h(\hat{x})$$
$$\iff -\lambda \nabla f(\hat{x}) \in \lambda \partial h(\hat{x}) \iff 0 \in \nabla f(\hat{x}) + \partial h(\hat{x}) \iff \hat{x} \text{ is the minimizer of (2.6)}.$$

Part (ii): Let $r = \text{prox}_{\lambda h}(t)$. By the definition (2.9) and the first order optimality condition, we can write

$$r = \arg\min_r h(r) + \frac{1}{2\lambda} ||r - t||^2 \iff 0 \in \partial h(r) + \frac{1}{\lambda}(r - t) \iff t - r \in \lambda \partial h(r). \tag{2.12}$$

By defining $u := x - \lambda G^f_{\lambda h}(x)$ and $w := x - \lambda \nabla f(x)$ we have

$$u = x - \lambda G^f_{\lambda h}(x) = x - \lambda \frac{1}{\lambda} \Big(x - \text{prox}_{\lambda h}\big(x - \lambda \nabla f(x)\big) \Big) = \text{prox}_{\lambda h}(x - \lambda \nabla f(x)) = \text{prox}(w).$$

Now, using (2.12), we can conclude: $G^f_{\lambda h}(x) - \nabla f(x) \in \partial h(x - \lambda G^f_{\lambda h}(x))$. Finally, using convexity of h we can write

$$h(x - \lambda G^f_{\lambda h}(x)) \le h(y) - \langle G^f_{\lambda h}(x) - \nabla f(x), y - (x - \lambda G^f_{\lambda h}(x)) \rangle.$$

Part (iii): By using the definition of F we have

$$\begin{aligned} F(x^+) - F(z) &= f(x^+) + h(x^+) - f(z) - h(z) \\ &= f(x - \lambda G^f_{\lambda h}(x)) - f(z) + h(x - \lambda G^f_{\lambda h}(x)) - h(z) \end{aligned}$$

Using the result of (ii) on the right-hand side of the above equality and the convexity of F yields

$$\begin{aligned} F(x^+) - F(z) &\le f(x - \lambda G^f_{\lambda h}(x)) - f(x) + \langle \nabla f(x), x - z\rangle - \langle G^f_{\lambda h}(x) - \nabla f(x), z - (x - \lambda G^f_{\lambda h}(x))\rangle \\ &\le \langle \nabla f(x - \lambda G^f_{\lambda h}(x)), x - \lambda G^f_{\lambda h}(x) - x\rangle + \langle G^f_{\lambda h}(x), x^+ - z\rangle + \langle \nabla f(x), \lambda G^f_{\lambda h}(x)\rangle \\ &= \langle x^+ - x, \nabla f(x^+)\rangle - \frac{1}{\lambda}\langle x^+ - x, x^+ - z\rangle + \langle x - x^+, \nabla f(x)\rangle. \end{aligned} \tag{2.13}$$

By adding and subtracting $\frac{1}{\lambda}\langle x^+ - x, x\rangle$ in (2.13), we obtain

$$F(x^+) - F(z) \le \langle x^+ - x, \nabla f(x^+) - \nabla f(x) + \frac{1}{2} G^f_{\lambda h}(x)\rangle - \left(\frac{1}{2\lambda}\|x^+ - x\|^2 + \frac{1}{\lambda}\langle x^+ - x, x - z\rangle\right).$$

Part (iv): Replacing the definition $\phi_k(\cdot)$ from (2.2) in (2.11) leads to

$$f(x - 2\lambda G^f_{\lambda h}(x)) - f(x - \lambda G^f_{\lambda h}(x)) \le -\lambda \langle G^f_{\lambda h}(x), \nabla f(x)\rangle + \frac{\lambda}{2}\|G^f_{\lambda h}(x)\|^2.$$

Using the convexity of f on the left-hand side of the above inequality yields

$$\langle \nabla f(x - \lambda G^f_{\lambda h}(x)), x - 2\lambda G^f_{\lambda h}(x) - x + \lambda G^f_{\lambda h}(x)\rangle \le -\lambda \langle G^f_{\lambda h}(x), \nabla f(x)\rangle + \frac{\lambda}{2}\|G^f_{\lambda h}(x)\|^2.$$

Considering the definition of x^+ in part (iii), we rearrange the above inequality and arrive at

$$\begin{aligned} &\langle \nabla f(x - \lambda G^f_{\lambda h}(x)) - \nabla f(x) + \frac{1}{2} G^f_{\lambda h}(x), G^f_{\lambda h}(x)\rangle \ge 0 \\ &\implies \langle \nabla f(x^+) - \nabla f(x) + \frac{1}{2} G^f_{\lambda h}(x), x^+ - x\rangle \le 0. \end{aligned}$$

□

2.1.2. NON-ACCELERATED COMPOSITE MINIMIZATION

This section focuses on devising an adaptive stepsize rule for non-accelerated algorithms (i.e., $d_k = -G^f_{\lambda h}(x)$) for the general class of composite functions (2.6). Next, we propose our non-accelerated stepsize rule for the general composite minimization problem (2.6). In this case, the zero-order linesearch (iv) is the driving force behind the proposed stepsize rule, after which we only need to apply the classic gradient descent update (2.1) with $d_k = -G^f_{\lambda h}(x)$. This discussion is formalized in the next theorem.

Theorem 2.1.3 (Non-accelerated adaptive stepsize)**.** *Consider the function F in (2.6) as a composition of a smooth function $f(x)$ and a prox-friendly function $h(x)$. Assume that the sequence $(x_k)_{k\in\mathbb{N}}$ generated by the update algorithm (2.1) in which the direction and stepsize rule are*

$$\begin{cases} \lambda_k = \max\left\{\lambda \in \mathbb{R}_+ \mid \phi_k(2\lambda) \le \phi_k(\lambda) - \lambda\langle G^f_{\lambda h}(x_k), \nabla f(x_k)\rangle + \dfrac{\lambda}{2}\|G^f_{\lambda h}(x_k))\|^2\right\}, \\ d_k = -G^f_{\lambda_k h}(x_k). \end{cases} \tag{Alg. 1}$$

Then, we have the uniform stepsize lower bound $\lambda_k \ge 1/3L$ where L is the smoothness constant of f. Moreover, we have the sub-optimality bound $F(x_k) - F(x^*) \le \dfrac{D}{k}$ where D is a constant that only depends on the initial condition x_0, the optimal solution x^*, and the smoothness constant L.

Proof. First, to show that λ_k is bounded from below by $1/3L$, we just need to show that the condition in the stepsize rule always is satisfied by $\lambda_k = 1/3L$. Considering the smoothness of f, we can write the inequality (2.8b) for the pair (x, x^+) where $x^+ = x - \lambda G^f_{\lambda h}(x)$, which arrives at the two inequalities

$$f(x^+) \le f(x) - \langle \nabla f(x), \lambda G^f_{\lambda h}(x)\rangle + \frac{L\lambda^2}{2}\|G^f_{\lambda h}(x)\|^2,$$
$$f(x) \le f(x^+) + \langle \nabla f(x^+), \lambda G^f_{\lambda h}(x)\rangle + \frac{L\lambda^2}{2}\|G^f_{\lambda h}(x)\|^2.$$

Adding the two sides of the above inequalities leads to

$$\langle G^f_{\lambda h}(x), \nabla f(x^+) - \nabla f(x) + L\lambda G^f_{\lambda h}(x)\rangle \ge 0, \qquad \forall\, \lambda \in \mathbb{R}_+,\, x \in \mathbb{R}^n. \tag{2.14}$$

Due to the smoothness of $\phi(\cdot)$, the stepsize rule in Alg. 1 is satisfied whenever

$$\left\langle G^f_{\lambda h}(x), \nabla f(x^+) - \nabla f(x) + \tfrac{1-L\lambda}{2} G^f_{\lambda h}(x)\right\rangle \ge 0,$$

which coincides with the inequality condition in (2.14) when $\lambda = 1/3L$, showing that the stepsize rule in Alg. 1 is always satisfied by choosing $\lambda = 1/3L$. Next, we prove the convergence of Alg. 1. By putting $z = x_k$ in the inequality (iii) we have

$$F(x_{k+1}) - F(x_k) \le -\frac{\lambda_k}{2}\|G^f_{\lambda_k h}(x_k)\|^2 \tag{2.15}$$

Telescoping the above inequalities from $k = 0$ to $k = \infty$ and using the fact that $\lambda_k \ge 1/3L$ gives us

$$F(x^\infty) - F(x_0) \le -\frac{1}{6L}\sum_{i=0}^{\infty}\|G^f_{\lambda_i h}(x_i)\|^2$$

According to Assumption 2.1.1 that $F(x)$ has a finite minimizer, we conclude $\|G^f_{\lambda h}(x^\infty)\|^2 \to 0$ which shows the convergence from (i). To drive the rate of convergence we start from (2.15) with $z = x^*$.

$$F(x_{k+1}) \le F(x^*) - \frac{1}{2\lambda_k}\|x_{k+1} - x_k\|^2 - \frac{1}{\lambda_k}\langle x_{k+1} - x_k, x_k - x^*\rangle \tag{2.16}$$

By adding and subtracting $\frac{1}{\lambda_k}\langle x_{k+1}, x_k\rangle$ and $\frac{1}{\lambda_k}\langle x_k, x_k\rangle$ terms and expanding the right-hand side of (2.16) we have

$$F(x_{k+1}) \leq F(x^*) + \frac{1}{2\lambda_k}\Big(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\Big) \tag{2.17}$$

Using (2.17) and the lower bound of λ_k, we can write

$$F(x_{k+1}) \leq F(x^*) + L\Big(\|x_k - x^*\|^2 - \|x_{k+1} - x^*\|^2\Big)$$

Summing above inequality from $k = 1$ to $k = T$ gives us

$$\sum_{k=1}^{T}\Big(F(x_k) - F(x^*)\Big) \leq L\Big(\|x_0 - x^*\|^2 - \|x_T - x^*\|^2\Big) \leq L\|x_0 - x^*\|^2$$

Because the sequence of $F(x_i)$ is nondecreasing, we can apply Jensen's inequality to arrive at

$$F(x_k) - F(x^*) \leq \frac{1}{k}\sum_{i=1}^{k}\Big(F(x_i) - F(x^*)\Big) \leq L\Big(\|x_0 - x^*\|^2 - \|x_k - x^*\|^2\Big) \leq \frac{L\|x_0 - x^*\|^2}{k} = \frac{D}{k}.$$

□

Theorem 2.1 can be extended to a slightly more general setting where f is locally smooth.

Corollary 2.1.4 (Locally smooth function)**.** *The sequence $(x_k)_{k\in\mathbb{N}}$ generated by (2.1) for the direction and stepsize rule Alg. 1 ensures the boundedness of sequence and the sub-optimality bound $F(x_k) - F(x^*) \leq \frac{D}{k}$, where f is locally smooth function in (2.6).*

Proof. This extension essentially indicates that the smoothness condition concerning the term f reflected in the constant L is only required over a compact set. To this end, we recall the proposed algorithm is monotone thanks to the inequality (2.15). That is, $f(x_{k+1}) \leq f(x_k)$ for all k, implying that the entire sequence of $(x_k)_{k\in\mathbb{N}}$ remains in the sublevel set of $\{x \in \mathbb{R}^n : f(x) \leq f(x_0)\}$. Note also that the regularity condition in Assumption 2.1.1 implies that the levelset of the objective function f is compact. This concludes the desired assertion. □

Remark 2.1.5 (Approximate adaptive stepsize rule)**.** *The stepsize rule Alg. 1 involves a scalar optimization problem. In practice, we use a simple backtracking method to approximate the optimal solution. To this end, we start with a initial λ_0 and scales it with a coefficient $C = (0, 1)$ as*

$$\lambda_k = \max\left\{\lambda_0 C^i \mid \phi_k(2\lambda) \leq \phi_k(\lambda) - \lambda\langle G^f_{\lambda h}(x_k), \nabla f(x_k)\rangle + \frac{\lambda}{2}\|G^f_{\lambda h}(x_k))\|^2,\ i \in \mathbb{N}\right\}.$$

The above approximate solution is guaranteed to satisfy $\lambda_k \geq \frac{C}{3L}$.

2.1.3. NON-ACCELERATED SMOOTH MINIMIZATION

In this subsection, we refine the result proposed in the previous subsection for (locally) smooth minimization, i.e., $h(x) = 0$ in (2.6). Specifically, the analysis in Theorem 2.1.3 can also be applied to a (locally) smooth minimization. Note that in the smooth case, $G^f_{\lambda h}(x) = \nabla f(x)$ and the linesearch in Alg. 1 is simplified to finding the largest λ satisfying $\phi(2\lambda) \leq \phi(\lambda) + \frac{\lambda}{2}\phi'(0)$. The following corollary explain the results when we invoke Alg. 1 to minimize the (locally) smooth function.

Corollary 2.1.6 (Convergence of smooth minimization)**.** *Let the function F in (2.6) be a smooth (i.e., $h(x) = 0$). Then, the stepsize rule in Alg. 1 reduces to (2.7) while ensuring Theorem 2.1.3 results of the uniform stepsize lower bound $\lambda_k \geq 1/3L$ and the sub-optimality bound $F(x_k) - F(x^*) \leq \frac{D}{k}$.*

Comparison of different stepsizes regime: We close this subsection by highlighting a common feature shared between the Polyak stepsize (2.4), Armijo stepsize rule (2.3a), and our proposed stepsize rule (2.7). These three stepsizes, denoted by λ, respect the following inequalities

$$\frac{1}{3L} < \frac{1}{2L} \leq \lambda \leq c\,\frac{f(y_1) - f(y_2)}{\|\nabla f(x_k)\|^2}, \tag{2.18a}$$

where the points y_1 and y_2 potentially depend on x_k and the stepsize λ. More specifically, we have

$$\begin{cases} \text{Polyak (2.4)}: & y_1 = x_k, & y_2 = x^*, & c = 1, \\ \text{Armijo (2.3a)}: & y_1 = x_k, & y_2 = x_k - \lambda\nabla f(x_k), & c > 1, \\ \text{Our stepsize (2.7)}: & y_1 = x_k - \lambda\nabla f(x_k), & y_2 = x_k - 2\lambda\nabla f(x_k), & c = 2. \end{cases} \tag{2.18b}$$

The common lower bound $1/(2L)$ in (2.18a) is well known for Polyak- and Armijo-type rules. Our proposed rule in Theorem 2.1.3 instead achieves a lower bound of $1/(3L)$, which is smaller than the bound of the existing linesearch rules. While larger stepsizes are commonly expected to yield faster convergence, our numerical experiments across multiple problem classes show that the proposed linesearch consistently outperforms existing methods. A possible explanation for this is that the theoretical bounds are typically conservative: in realistic problems, the local geometry often permits stepsizes much larger than those implied by the global smooth constant. Our linesearch procedure exploits this favorable curvature by accepting substantially larger steps in most iterations, leading to faster objective decrease and fewer gradient/proximal evaluations overall.

Notice that the optimal choice of the stepsizes is the upper bound of the admissible interval (2.18a). It is also important to note that in the case of Armijo (2.3a) and our proposed choice (2.7), this upper bound also depends on λ (see (2.18b)), making the stepsize rule implicit. Finally, we wish to draw attention to the choice of the upper bound between Armijo and our proposed approach in (2.18a): One can inspect that the

proposed rule (2.7) looks one step ahead of Armijo in the direction of ∇f. This may explain why the proposed rule (2.7) performs numerically better, as it can approximate the curvature of the function f by "looking one step further" than Armijo. Finally, it is also worth noting that the recent work [31] offers an adaptive stepsize rule with the lower bound $1/2L$ and an explicit upper bound (i.e., independent of λ) that depends on the gradient of two successive iterations (see (2.5)).

2.2. ACCELERATED ADAPTIVE STEPSIZE

This section pertains to the analysis of convergence for update direction d_k in accelerated methods. The general form of our accelerated minimization algorithm is provided in Alg. 2, which follows the accelerated proximal gradient descent method with the difference in the choice of stepsize λ [39], [49], [50]. Differently from [39], [44], the stepsize is not fixed, and in our numerical experience, it does not strictly decrease in many iterations, which partially explains the acceleration of the algorithm up to the optimal theoretical bound $\mathcal{O}(k^{-2})$.

Theorem 2.2.1 (Accelerated adaptive stepsize)**.** *Consider the function F in (2.6) as a composition of a smooth function $f(x)$ and a prox-friendly function $h(x)$. The sequence $(x_k)_{k\in\mathbb{N}}$ generated by (2.1) with the direction and stepsize rule*

$$\begin{cases} \beta_k = \dfrac{1+\sqrt{1+4\beta_{k-1}^2}}{2},\ \gamma_k = \dfrac{1-\beta_k}{\beta_{k+1}},\ x_0 = x_1,\ \beta_1 = 0, \\ \lambda_k = \arg\max\limits_{\lambda\in\mathbb{R}} \left\{ \lambda \mid \phi(2\lambda) \le \phi(\lambda) - \lambda\langle G^f_{\lambda h}(x), \nabla f(x)\rangle + \dfrac{\lambda}{2}\|G^f_{\lambda h}(x)\|^2,\ \lambda \le \lambda_{k-1} \right\}, \\ d_k = -\gamma_k \left(\dfrac{\lambda_{k-1}}{\lambda_k} d_{k-1} - (1+\dfrac{1}{\gamma_k}) G^f_{\lambda_k h}(x_k) + \dfrac{\lambda_{k-1}}{\lambda_k} G^f_{\lambda_{k-1} h}(x_{k-1}) \right). \end{cases} \tag{Alg. 2}$$

ensures the uniform stepsize lower bound $\lambda_k \ge 1/3L$ and the sub-optimality bound $F(x_k) - F(x^*) \le \dfrac{D'}{k^2}$ where L is the smoothness constant of f, and D' is a constant that only depends on the initial condition x_0, the optimal solution x^*, and the constant L.

Proof. The proof of a first proposition is the same as in Theorem 2.1.3. Specifically, according to stepsize rule in Alg. 2, we should first check whether the previous λ_{k-1} satisfies the linesearch condition. Since the inequality (2.14) holds for $\lambda = 1/3L$ (according to the linesearch condition in Alg. 2), we have the guarantee that $\lambda_k \ge 1/3L$. Regarding the convergence analysis bound, in the first step, we invoke the increment inequality of (iii) in Lemma 2.1.2 for two cases of $z = y_k$ and $z = x^*$, which arrives at

$$F(y_{k+1}) - F(y_k) \le -\frac{1}{2\lambda_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\lambda_k}\langle y_{k+1} - x_k, x_k - y_k\rangle, \tag{2.19a}$$

$$F(y_{k+1}) - F(x^*) \le -\frac{1}{2\lambda_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\lambda_k}\langle y_{k+1} - x_k, x_k - x^*\rangle. \tag{2.19b}$$

Let us define $\delta_k := F(y_k) - F(x^*)$. Then, multiplying (2.19a) by $\beta_k - 1$, and adding the two sides of the inequality to (2.19b) yields

$$\beta_k\delta_{k+1} - (\beta_k - 1)\delta_k \le -\frac{\beta_k}{2\lambda_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\lambda_k}\langle y_{k+1} - x_k, \beta_k x_k - (\beta_k - 1)y_k - x^*\rangle.$$

Multiplying the above inequality by $\lambda_k\beta_k$ and considering $\beta_{k-1}^2 := \beta_k^2 - \beta_k$ and using the fact that $\lambda_k \leq \lambda_{k-1}$, we have

$$\lambda_k\beta_k^2\delta_{k+1} - \lambda_{k-1}\beta_{k-1}^2\delta_k \leq -\frac{1}{2}\Big(\|\beta_k(y_{k+1}-x_k)\|^2 + 2\beta_k\langle y_{k+1}-x_k, \beta_k x_k - (\beta_k-1)y_k - x^*\rangle\Big). \tag{2.20}$$

The right-hand side of (2.20) can be equivalently written as

$$\|\beta_k(y_{k+1}-x_k)\|^2 + 2\beta_k\langle y_{k+1}-x_k, \beta_k x_k - (\beta_k-1)y_k - x^*\rangle = \\ \|\beta_k y_{k+1} - (\beta_k-1)y_k - x^*\|^2 - \|\beta_k x_k - (\beta_k-1)y_k - x^*\|^2. \tag{2.21}$$

Substituting (2.21) into (2.20) and rearranging the inequality lead to

$$\lambda_k\beta_k^2\delta_{k+1} - \lambda_{k-1}\beta_{k-1}^2\delta_k \leq -\frac{1}{2}\Big(\|\beta_k y_{k+1} - (\beta_k-1)y_k - x^*\|^2 - \|\beta_k x_k - (\beta_k-1)y_k - x^*\|^2\Big). \tag{2.22}$$

Using the update rule of x_{k+1} on the right-hand side of (2.22) reduces to

$$\beta_k y_{k+1} - (\beta_k-1)y_k - x^* = \beta_{k+1}x_{k+1} - (\beta_{k+1}-1)y_{k+1} - x^*, \tag{2.23}$$

which is equivalent to

$$x_{k+1} = \frac{(-1+\beta_k+\beta_{k+1})}{\beta_{k+1}}y_{k+1} + \frac{1-\beta_k}{\beta_{k+1}}y_k.$$

By combining (2.22) and (2.23) and defining $u_k = \beta_k x_k - (\beta_k-1)y_k - x^*$, we obtain

$$\lambda_k\beta_k^2\delta_{k+1} - \lambda_{k-1}\beta_{k-1}^2\delta_k \leq \frac{1}{2}(\|u_k\|^2 - \|u_{k+1}\|^2). \tag{2.24}$$

We note that the inequality (2.24) can be viewed as the so-called Lyapunov function as it indicates that the term $\lambda_k\beta_k^2\delta_{k+1} + \|u_{k+1}\|^2/2$ is monotonically decreasing in each iteration. Summing inequalities (2.24) from $k=1$ to $k=T$, one obtains

$$\lambda_T\beta_T^2\delta_{T+1} - \lambda_0\beta_0^2\delta_1 \leq \frac{1}{2}\|u_1\|^2 - \frac{1}{2}\|u_{T+1}\|^2 \leq \frac{1}{2}||u_1||^2.$$

Setting $\beta_0 = 0$ in the above equation yields $\delta_{T+1} \leq \dfrac{\|u_1\|^2}{2\lambda_T\beta_T^2}$. Finally, using Remark 2.1.5 and the growing rate of $\beta_k \geq k/2$, we deduce the desired result

$$F(y_{T+1}) - F(x^*) = \delta_{T+1} \leq \frac{6L\|u_1\|^2}{CT^2} \leq \frac{D'}{T^2}.$$

□

We can refine the result proposed in Theorem 2.2.1 for smooth minimization, where the non-smooth part, h, in (2.6) is $h(x)=0$. In this case, $G_{\lambda h}^f(x) = \nabla f(x)$, and the line-search in the stepsize rule Alg. 2 is simplified to find the largest positive scalar λ that satisfies $\phi(2\lambda) \leq \phi(\lambda) + \frac{\lambda}{2}\phi'(0)$. This is formalized in the next Corollary.

Corollary 2.2.2 (Accelerated smooth minimization)**.** *Let the function F in (2.6) be a smooth (i.e., $h(x) = 0$). Then, the stepsize rule in Alg. 2 reduces to (2.7) while ensuring Theorem 2.2.1 results of the uniform stepsize lower bound $\lambda_k \geq 1/3L$ and the sub-optimality bound* $F(x_k) - F(x^*) \leq \dfrac{D'}{k^2}$.

2.3. NUMERICAL RESULTS

We illustrate the performance of our proposed stepsize rules Alg. 1 and Alg. 2 in three major problem classes widely used in operations research, machine learning, and systems and control:

1. **Smooth functions**, including the specific objective functions
   - (i) logistic regression, motivated by classification tasks in machine learning problems [51],
   - (ii) quadratic programming, motivated by portfolio optimization and resource allocation problems [52], [53], [54],
   - (iii) log-sum-exp, motivated by discrete-choice modeling and entropy-regularized transport, e.g., multinomial logit model and softmax demand models [55], [56], [57], and
   - (iv) max-cut, a benchmark for combinatorial optimization and approximate semidefinite programming [58], [59], [60], [61].
2. **Composite minimization**, including the specific objective functions
   - (i) ℓ_1-regularized least squares, commonly used in high-dimensional statistics for support recovery and in signal processing for sparse signal reconstruction [62], [63]),
   - (ii) ℓ_1-constrained least squares, in sparse signal recovery with explicit coefficient bounds [64]), and (2-iii) ℓ_1-regularized logistic regression (sparse classification and feature selection, [65]).
3. **Non-convex minimization**, where we consider a cubic objective function relevant to cubic-regularization in nonconvex optimization and trust-region subproblems [66], [67].

Before presenting the numerical results, let us first start with a practical discussion concerning the implementation of the approximate solution in Remark 2.1.5 for the stepsize rule in Alg. 1. The backtracking in Remark 2.1.5 uses an initial constant λ_0 to approximate the proposed stepsize rule in Alg. 1. On the one hand, if the initial λ_0 is too large, it takes time to compute the desired stepsize satisfying linesearch condition. On the other hand, a too small λ_0 deteriorates the convergence speed [32]. To address this issue, we approximate the function ϕ in (2.2) via a quadratic function, denoted by $\tilde{\phi}$, and suggest the desired λ_0 as its optimizer; see Figure 2.2 for a visual representation of this approximation (cf. Figure 2.1b).

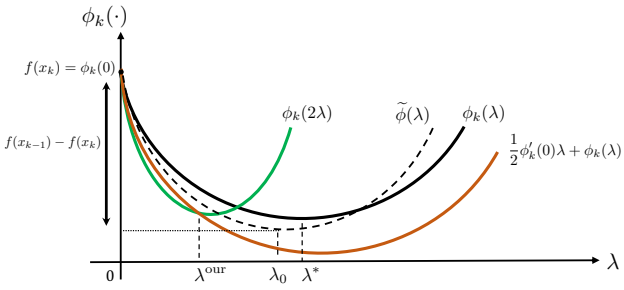


Figure 2.2: Initial choice of λ_0 at the $(k+1)^{\text{th}}$ iteration.

For the approximate function $\widetilde{\phi}$, we suggest the following criteria:

$$\left.\begin{aligned} &\widetilde{\phi}(0) = \phi(0), \\ &\widetilde{\phi}'(0) = \phi'(0), \\ &\widetilde{\phi}(0) - \min \widetilde{\phi}(\lambda) = f(x_{k-1}) - f(x_k) \end{aligned}\right\} \Longrightarrow \lambda_0 = \arg\min_{\lambda} \widetilde{\phi}(\lambda) = \frac{2\Big(f(x_{k-1}) - f(x_k)\Big)}{\|\nabla f(x_k)\|^2}.$$

Notice that the function $\widetilde{\phi}$ is constructed in a way whose minimum value is $f(x_{k-1}) - f(x_k)$, i.e., the objective improvement of the last iteration.

To evaluate our performance, we compare our proposed algorithms with the following methods from the literature: For the class of (1) smooth optimization, we consider the gradient descent (GD) [32], Nesterov's accelerated gradient descent (NAGD) [39], the recent adaptive gradient descent (AdGD) and its heuristic accelerated version (AdGD-accel) [31]. Respectively, for the class of (2) composite minimization, we consider the nonsmooth counterparts including the proximal gradient descent (ISTA) and its accelerated version (FISTA) [44], the adaptive golden ratio algorithm (aGRAAL) [46] and adaptive primal-dual algorithm (APDA) [45]. For (3) non-convex minimization, we use the existing algorithms from the first class of (1) smooth minimization.

2.3.1. SMOOTH MINIMIZATION

This section includes four different classes of smooth functions:

(i) Logistic regression: Motivating with the classification problems [51], we consider the logistic regression with quadratic regularization as follows:

$$\min_{x\in\mathbb{R}^d} \frac{1}{N}\sum_{i=1}^{N} \log\big(1+\exp(-b_i a_i^\top x)\big) + \frac{\gamma}{2}\|x\|^2, \tag{2.25}$$

where N is a number of data, $A = [a_1|a_2|\cdots|a_n] \in \mathbb{R}^{d\times N}$ and $\{b_i\}_{i=1}^N \in \mathbb{R}$ are the features of data, and γ is a regularization parameter proportional to $1/N$. In our experiments, The test data is generated as follows: the matrix $A \in \mathbb{R}^{N\times N}$ is generated randomly using the standard Gaussian distribution $\mathcal{N}(0,1)$, where we generate A with 50% correlated columns as $A(:, j+1) = 0.5A(:, j) + \text{randn}(:)$. The observed measurement vector

b is generated as $b := Ax^\natural + \mathcal{N}(0, 0.05)$, where $x^\natural$ is generated randomly using $\mathcal{N}(0,1)$, $d = N = 200$, and we estimate the Lipschitz constant of the gradient by the closed-form expression $L = \|A\|^2/N + \gamma$ used in GD and NAGD as a stepsize. Figure 2.3a reports the results where all algorithms are initialized at the same point chosen randomly.

(ii) Quadratic programming: Following recent works on portfolio and robust optimization [52], [53], we examine the quadratic objective function

$$\min_{x\in\mathbb{R}^d} \frac{1}{2} x^\top B x + b^\top x, \tag{2.26}$$

where $B \in \mathbb{R}^{d\times d} > 0$ and $b \in \mathbb{R}^d$ are given. This function is smooth with constant $L = \|B\|^2$. The parameters B and b are chosen in the same manner as in the previous part, with the difference that $B = A^\top A$ and b are normalized. Additionally, in our experiments, we set $d = 200$. The results of minimizing this quadratic function are provided in Figure 2.3b where all the methods are initialized at the same point. It is interesting to note that the proposed stepsize rule for accelerated algorithms exhibits similar behavior to NAGD with a comparative performance while outperforming other adaptive algorithms.

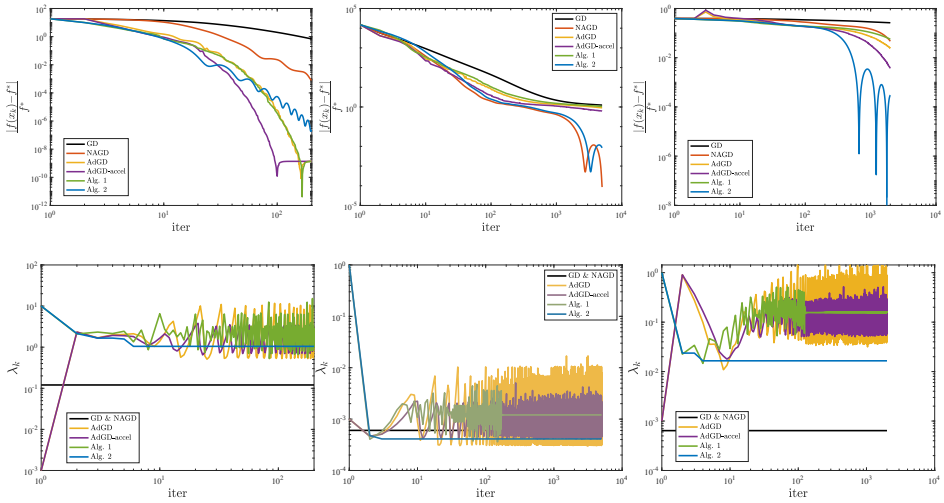
(iii) Log-Sum-Exp: In the context of discrete-choice modeling and entropy-regularized transport [55], [56], [57], we consider the log-sum-exp objective function with regularization

$$\min_{x\in\mathbb{R}^d} \log\left(\sum_{i=1}^N \exp(a_i^\top x - b_i)\right) + \frac{\gamma}{2}\|x\|^2, \tag{2.27}$$

where $A = [a_1|a_2|\cdots|a_n] \in \mathbb{R}^{d\times N}$ and $\{b_i\}_{i=1}^N \in \mathbb{R}$ are the problem data. The log-sum-exp function is viewed as a smooth approximation of $\max_{x\in\mathbb{R}^d}\{a_1^\top x - b_1, a_2^\top x - b_2, \cdots, a_N^\top x - b_N\}$. This function can sometimes be ill-conditioned, making the minimization task particularly difficult for the first-order methods. The function is smooth with the smoothness constant $L = \sigma_{max}(A^\top)$ where $\sigma_{max}(A^\top)$ is a maximum singular value of $A^\top$. In the experiment, we set $N = d = 200$, and A and $\{b_i\}_{i=1}^N$ are generated using a similar approach as in the logistic regression part. The results of minimizing the log-sum-exp function are depicted in Figure 2.3c where the proposed accelerated method demonstrates superior efficacy compared to other algorithms.

(vi) Approximate semidefinite programming: Semidefinite programs are ubiquitous in combinatorial optimization and control [58], [60], [68], graph theory [69], network and communication [70]. Most of the problems in this class essentially deal with minimizing the maximum eigenvalue of a matrix, which is a nonsmooth function of its variable. A popular example falling into this category is the *max-cut* problem, whose ε-approximation with the regularizer $\mathscr{R}(x)$ reads as

$$\min_{y\in\mathbb{R}^n} f_\varepsilon(C + \mathrm{diag}(y)) - \langle \mathbf{1}, y\rangle + \eta\mathscr{R}(x), \quad f_\varepsilon(X) = \varepsilon \log\left(\sum_{i=1}^n \exp(\lambda_i(X)/\varepsilon)\right), \tag{2.28}$$



(a) Logistic regression (2.25) (b) Quadratic programming (2.26) (c) Log-Sum-Exp (2.27)

Figure 2.3: The results for the class (1) smooth minimization. The first row shows the optimality gap, and the second row shows the stepsize behavior.

where C is a constant matrix, η is a regularization coefficient, and $f_\varepsilon(X)$ is the smooth convex approximation of $\lambda_{\mathbf{max}}(X)$. As shown in [59], the gradient of $f_\varepsilon(X)$ can be computed as

$$\nabla f_\varepsilon(X) = \frac{\sum_{i=1}^n \exp(\lambda_i(X)/\varepsilon) q_i q_i^\top}{\sum_{i=1}^n \exp(\lambda_i(X)/\varepsilon)},$$

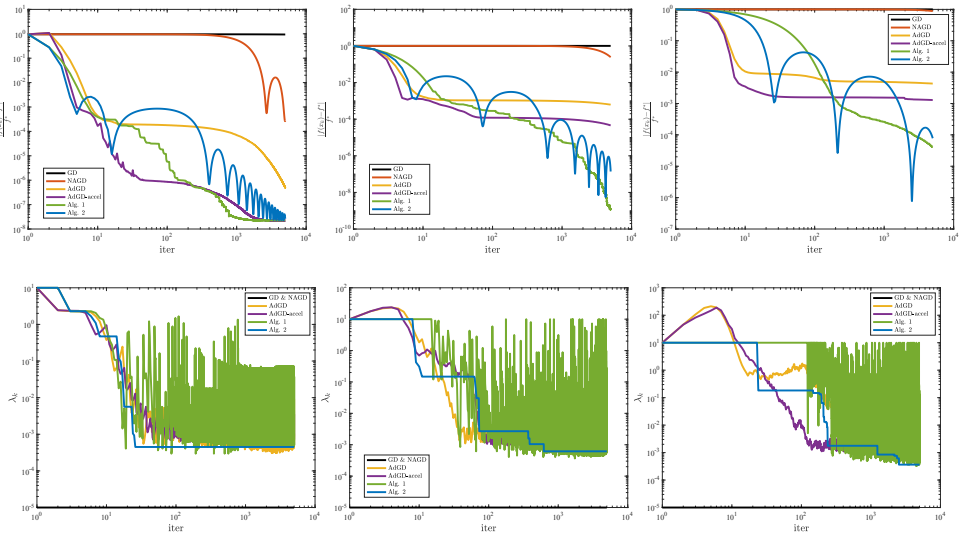
where q_i is the i^{th} column of the unitary matrix Q in the eigen-decomposition $Q\Sigma Q^\top$ of X with eigenvalues $\lambda_1 \geq \cdots \geq \lambda_n$. In addition, $f_\varepsilon(X)$ is smooth with the smoothness parameter $1/\varepsilon$. Figure 2.4 shows the simulation results of solving (2.28) for different regularization coefficient η where the approximation level is $\varepsilon = 10^{-5}$. The matrix C is also generated using the Wishart distribution with $C = G^\top G/\|G\|_2^2$, where G is a standard Gaussian matrix [71], $n = 100$, and $\mathscr{R}(y) = \|y\|^2$.

2.3.2. COMPOSITE MINIMIZATION

Motivated by sparse recovery and reconstruction problems [62], [63], [64], as well as sparse classification and feature selection [65], we consider the following non-smooth composite minimization formulations:

(i) ℓ_1-Regularized least square [62], [63]:

$$\min_{x\in\mathbb{R}^d} \underbrace{\|Ax-b\|_2^2}_{f(x)} + \underbrace{\gamma\|x\|_1}_{h(x)}. \tag{2.29}$$



(a) $\eta = 0.1$ (b) $\eta = 0.01$ (c) $\eta = 0.001$

Figure 2.4: Approximate maximum eigenvalue (2.28). The first row shows the optimality gap and the second row shows the stepsize behavior.

(ii) ℓ_1-Constrained least square [64]:

$$\min_{x\in\mathbb{R}^d}\{\|Ax-b\|_2^2 : \|x\|_1 \le 1\} = \min_{x\in\mathbb{R}^d} \underbrace{\|Ax-b\|_2^2}_{f(x)} + \underbrace{\delta_{B_1[0,1]}}_{h(x)}. \tag{2.30}$$

(iii) ℓ_1-Regularized logistic regression [65]:

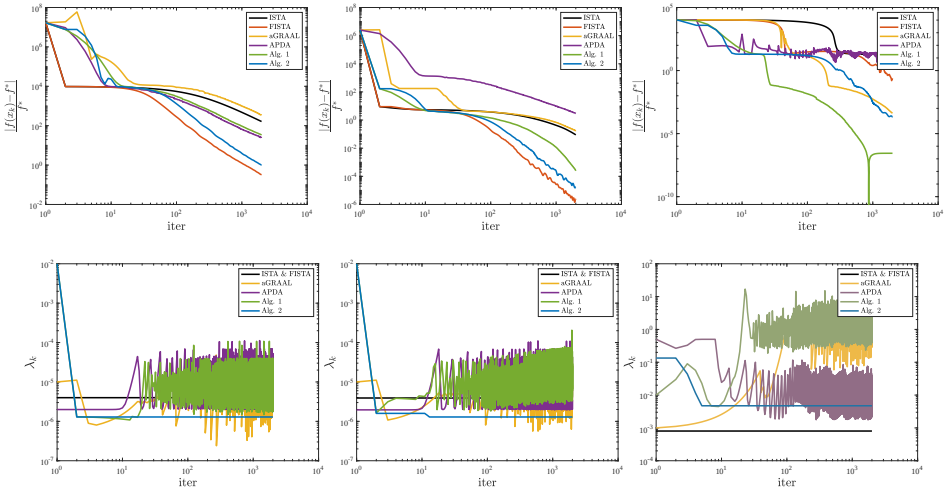
$$\min_{x\in\mathbb{R}^d} \underbrace{\frac{1}{N}\sum_{i=1}^{N} \log\left(1+\exp(-b_i a_i^\top x)\right)}_{f(x)} + \underbrace{\gamma\|x\|_1}_{h(x)}. \tag{2.31}$$

The functions $f(x)$ and $h(x)$ represent the smooth and prox-friendly parts, respectively. In the previous subsection, we explain the smooth parts and their smooth constant. The prox-friendly term $h(x)$ has a closed form solution provided in Table 2.2 where δ, $B_1[0,1]$, $[\cdot]_+$, and $\odot$ are indicator function, unit ball with 1-norm, positive part selector, and elementwise product, respectively [72].

Table 2.2: Closed-form prox-operator of $h(x)$.

$h(x)$	$\text{prox}_{h(x)}$
$\gamma\|x\|_1$	$\tau_\gamma(x) = \left[\|x\| - \gamma\mathbf{e}\right]_+ \odot \mathbf{sign}(x)$
$\delta_{B_1[0,1]}$	$P_{B_1[0,1]}(x) = \min_{y\in B_1[0,1]} \|y-x\|$

In our simulations, we set $N = d = 200$, γ proportional to $1/N$, $A = 5 \cdot \texttt{rand(n,n)}$, and $b = \texttt{rand(n,1)}$, where $\texttt{rand}(\cdot)$ is uniformly distributed random numbers in the interval $(0,1)$. The results are provided in Figure 2.5. We wish to note that for the objective functions (2.29) and (2.30), the proposed accelerated stepsize Alg. 2 has a competitive result and similar behavior with the best performance FISTA [44] while in the case of (2.31) outperforms all the other algorithms notably.



(a) ℓ_1-Regularized least square (2.29) (b) ℓ_1-Constrained least square (2.30) (c) ℓ_1-Regularized logistic reg (2.31)

Figure 2.5: The results for the class (2) composite minimization. The first row shows the optimality gap, and the second row shows the stepsize behavior.

2.3.3. NON-CONVEX MINIMIZATION:

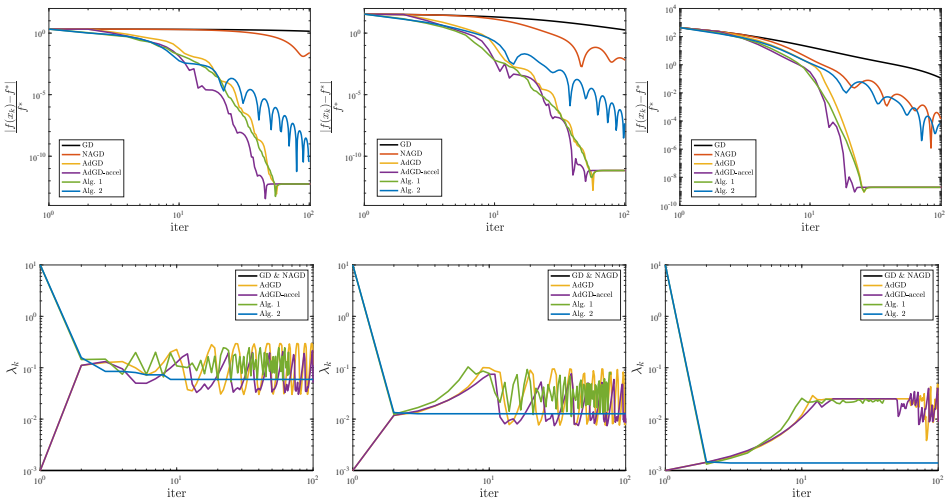
We also test our algorithm in non-convex minimization. We consider the case of cubic optimization that is useful in high-order methods and trust-region subproblems [66], [67] which is defined as

$$\min_{x \in \mathbb{R}^d} \frac{1}{2} x^\top H x + g^\top x + \frac{M}{6} \|x\|^3, \tag{2.32}$$

where the matrix $H \in \mathbb{R}^{d \times d}$, vector $g \in \mathbb{R}^d$, and scalar $M > 0$ are the problem data. Note that in higher-order methods, the constant M has an impact on the convergence rate [73]. Due to the cubic term, the function (2.32) is neither convex nor smooth. The parameters g and H are the gradient and the Hessian of the logistic loss (2.25), respectively, and are computed using the same data as in Subsection 2.3.1 ((i) logistic regression). We provide the simulations for different M with $d = N = 200$, and the results are reported in Figure 2.6.

2.4. CONCLUSION AND FUTURE DIRECTIONS

This research proposes a new stepsize rule for non-accelerated and accelerated first-order methods for convex composite optimization. The proposed approach provably



(a) $M = 0.5$ (b) $M = 5$ (c) $M = 50$

Figure 2.6: Cubic regularization problem (2.32). The first row shows the optimality gap, and the second row shows the stepsize behavior for different values of M.

achieves the convergence optimal rate of $\mathcal{O}(k^{-2})$ for accelerated methods. Compared to other stepsize rules, our proposed scheme is implicit but only requires zeroth-order information to compute a suitable stepsize. Possible extension to this work could be the $minmax$ problem [74], [75], [76], stochastic gradient descent optimization [77], variational inequalities [78], [79], [80], and non-convex optimization [45], [81], [82]. To mitigate the fact that the stepsize in Alg. 2 is non-increasing, one could re-initialize it finitely many times after a predefined number of iterations. This technique is similar to the restarted accelerated gradient method [43], [83], [84] and it is left as future work.

The authors of a recent work [85] demonstrate that iterations generated by Forward Backward splitting methods, which include several variants (e.g., ISTA, FISTA), lie on active manifolds within a finite number of iterations and enter a regime of local linear convergence. However, their analysis are based on the standard assumption that the stepsize belongs to $[0, 2/L]$. Departing from this convention is one of the future areas of interest. The authors in [85] also establish and explain why FISTA locally oscillates and can be slower than ISTA under the same standard assumption $\lambda \in [0, 2/L]$. We observe similar behavior in our simulation results (Figures 2.3a, 2.5b, 2.6). Analyzing this behavior for $\lambda > 2/L$, which can also occur in our proposed stepsize rules, could be another future direction.

3

Locally Linear Convergence for Nonsmooth Convex Optimization via Coupled Smoothing and Momentum

Chapter 2 explores the analysis of methods for solving smooth and composite convex optimization problems. However, in many real-world applications, the modeling involves multiple nonsmooth convex terms, for which classical methods are slow and algorithms developed for smooth problems are not directly applicable. In this chapter, we propose an adaptive accelerated smoothing technique for a nonsmooth convex optimization problem where the smoothing update rule is coupled with the momentum parameter. We also extend the setting to the case where the objective function is the sum of two nonsmooth functions. With regard to convergence rate, we provide the global (optimal) sublinear convergence guarantees of $\mathcal{O}(1/k)$, which is known to be provably optimal for the studied class of functions, along with a local linear rate if the nonsmooth term fulfills a so-call locally strong convexity condition.

We validate the performance of our algorithm on several problem classes, including regression with the ℓ_1-norm (the Lasso problem), sparse semidefinite programming (the MaxCut problem), Nuclear norm minimization with application in model free fault diagnosis, and ℓ_1-regularized model predictive control to showcase the benefits of the coupling. An interesting observation is that although our global convergence result guarantees $\mathcal{O}(1/k)$ convergence, we consistently observe a practical transient convergence rate of $\mathcal{O}(1/k^2)$, followed by asymptotic linear convergence as anticipated by the theoretical result. This two-phase behavior can be explained in view of the proposed smoothing rule.

Objective functions with multiple nonsmooth terms are widely used in optimization-based control, system identification and machine learning. The broad applicability of these formulations across diverse domains calls for efficient algorithms capable of handling nonsmooth optimization problems.

For instance, the authors in [86], [87] consider model predictive control (MPC) with nonsmooth ℓ_1 regularizers to obtain sparse control inputs. The authors in [88] use combinations of ℓ_1 and Nuclear norms to model fault detection in model-free dynamical systems. [89] employ ℓ_2–ℓ_1 and ℓ_1–ℓ_1 formulations for image restoration; [90] utilize the TV–ℓ_1 model for image denoising; [91] apply the ℓ_1–ℓ_1 formulation to dictionary learning; and [92] use Nuclear norms for low-rank matrix decomposition in graph neural networks with an ℓ_1 regularizer.

Motivating example. Consider the MPC formulation for a class of constrained linear systems with uncertain state-delays [93, Eq. (1)]:

$$\begin{aligned} x(k+1) &= \mathbf{A}x(k) + \mathbf{A_d}x(k - N_d(x_k)) + \mathbf{B}u(k), \\ N_d(k) &\in [1, \bar{N}_d]; \quad x(i) = x_i, \quad \forall i \in \{-\bar{N}_d, \dots, 0\}, \\ \|u_i\| &\leq \bar{u}_i, \quad i \in \{1, \dots, m\}, \end{aligned} \tag{3.1}$$

where k is the discrete-time index, $x \in \mathbb{R}^n$ represents the system state, $u \in \mathbb{R}^m$ is the input vector, $\bar{u}_i$ are the input constraints, $N_d(k)$ is the uncertain time delay, possibly varying with time, $\bar{N}_d$ is its upper bound, and the matrices $\mathbf{A}$, $\mathbf{A_d}$, and $\mathbf{B}$ define the system dynamics.

Based on an artificial Lyapunov function, a stabilizing condition depending on the upper bound of the uncertain state-delays, together with MPC, is presented in the form of a linear matrix inequality as [93, Eq. (15)]:

$$\min_{u(k|k),\dots,u(k+N-1|k)} J(k) \tag{3.2}$$
$$\text{s.t. system dynamics and constraints (3.1)},$$

$$\begin{bmatrix} P - \bar{N}_d Q_d & 0 & (\mathbf{A}+\mathbf{B}K)^\top \\ 0 & Q_d & \mathbf{A}_d^\top \\ \mathbf{A}+\mathbf{B}K & \mathbf{A}_d & P^{-1} \end{bmatrix} \succ 0,$$

$$\begin{bmatrix} Y & K \\ K^\top & P\mu^{-1} \end{bmatrix} \geq 0, \quad Y_{ij} \leq \bar{u}_j^2, \quad \forall i, j \in \{1, \dots, m\},$$

where, J is a nonsmooth objective function that can include an ℓ_1 regularizer, e.g., $\sum_i \|\Delta u_i\|_1$, to enforce sparsity in input changes, allowing the controller to respond promptly to disturbances and system variations [86], [87]. We emphasize that, the MPC problem in (3.2), is complex due to the nonsmooth objective function and SDP constraints, and must be solved at every sampling time.

This motivates fast optimization algorithms for nonsmooth convex problem of the following class:

$$\min_{x \in \mathbb{R}^n} F(x) = \min_{x \in \mathbb{R}^n} f(x) + h(x), \tag{3.3}$$

Table 3.1: Comparison of algorithms for minimizing nonsmooth convex functions.

Algorithm	Smoothing technique	prox-friendly assumption	Local linear convergence rate	Global convergence rate
Subgradient descent [40]	✗	✗	✗	$\mathcal{O}(1/\varepsilon^2)$
Stochastic smoothing [101]	✗	✗	✗	$\mathcal{O}(1/\varepsilon^2)$
Chambolle-Poc [90]	✗	✓	✗	$\mathcal{O}(1/\varepsilon)$
Nesterov's smoothing [59], [96]	✓	✓	✗	$\mathcal{O}(1/\varepsilon)$
Variable smoothing [104]	✓	✓	✗	$\mathcal{O}(\log(1/\varepsilon)/\varepsilon)$
Adaptive smoothing [105]	✓	✓	✗	$\mathcal{O}(1/\varepsilon)$
Proposed adaptive smoothing (Theorems 3.2.3–3.2.4)	✓	✓	✓ (under some assumptions)	$\mathcal{O}(1/\varepsilon)$

where the functions f and h are proper, closed, convex, possibly nonsmooth, but prox-friendly (to be made precise later in the notation section, see (3.4)). A particular subclass of such problems is composite minimization in which one of these functions is smooth [6], [44], [94]. While the composite minimization problem has been extensively studied, the possibility of having multiple nonsmooth terms remains relatively unexplored. This is primarily due to the fact that the summation of two prox-friendly functions is not necessarily prox-friendly [1]. Let us emphasize that the most common algorithms, such as subgradient [40], mirror descent [41], and bundle methods [95] often suffer from a slow convergence rate of $\mathcal{O}\left(1/\varepsilon^2\right)$ where ε is an apriori desired precision.

Following the seminal works by Nesterov [59], [96], one can exploit the structure of the nonsmooth objective function to propose a smooth approximation and then deploy a first-order accelerated method to find the optimizer of the smooth approximation [39]. Remarkably, Nesterov's algorithm requires only $\mathcal{O}(1/\varepsilon)$ iterations to reach an ε optimal solution. Inspired by this approach, several works have tried to improve and enhance the efficacy of smoothing algorithms in terms of complexity or to customize it for specific applications at hand [90], [97], [98], [99]. In [100], comparisons are made about the advantages and disadvantages of smoothing techniques and proximal-type methods. Based on [59], the authors in [101] utilize a stochastic smoothing technique to improve the scalability of the algorithm for semidefinite programming problems. The smoothing techniques in [102] improve the convergence speed in comparison with conditional gradient algorithms. The authors in [103] study the smoothing technique for minimizing the sum of three functions, where two of which are nonsmooth. The paper [104] studies an adaptive smoothing algorithm with a convergence rate of $\mathcal{O}(1/\varepsilon\log(1/\varepsilon))$.

The closest existing work to our study is the adaptive smoothing technique in [105] that enjoys $\mathcal{O}(1/\varepsilon)$ complexity, which matches the worst-case bound offered by Nesterov's algorithm. Table 3.1 summarizes the convergence results of this study and compares them with those in the literature. Following the footsteps of Nesterov's smoothing technique, we propose a novel adaptive-smoothing technique where the smoothing parameter decreases with each iteration but as a function of the momentum, which retains the same global convergence rate while improving the asymptotic performance.

Contributions: The contributions of our work are summarized as follows:

(i) **Global optimal sublinear convergence:** We introduce an algorithm with an adaptive smoothing parameter coupled with the momentum term (Algorithm 1). When the smoothing rule is modified to stay away from zero, we provide a global sublinear convergence rate of $\mathcal{O}\left(1/\varepsilon\right)$ that matches the optimal worst-case complexity bound

for optimizing this class of nonsmooth functions (Theorem 3.2.3).

(ii) **Locally linear convergence:** When the nonsmooth term $f(x)$ meets a so-called ∞-locally strong convexity condition, we show that Algorithm 1 enjoys a local linear convergence rate (Theorem 3.2.4). Combined with the global convergence result from the previous section, this implies that an appropriate initial condition for Algorithm 1 ensures a transient optimal sublinear convergence rate followed by an asymptotic linear convergence rate.

(iii) **Multiple nonsmooth terms:** The proposed algorithm allows for multiple nonsmooth terms. Such settings can be computationally challenging as the "prox-friendly" property, a key feature in nonsmooth optimization, is not necessarily additive. Important applications falling into this category include model-free fault diagnosis, nonsmooth model predictive control, sparse regression and sparse semidefinite programming, which are the examples investigated in our numerical section to validate the performance of our proposed algorithm.

To validate the theoretical results of this study, we implement and compare the performance of the proposed algorithm with different existing methods in the literature on various classes of problems including regression with the ℓ_1-norm (the Lasso problem), sparse semidefinite programming (the *MaxCut* problem), and Nuclear norm minimization. Regarding the above contributions, an observation consistently confirmed by these numerical results deserves attention.

An unexpected observation: The smoothness parameter has a direct impact on the stepsize in accelerated algorithms, i.e., the smoother the function, the larger the stepsize. A general rule of thumb is that larger stepsizes lead to faster convergence, and with that in mind, we would expect to see a slower convergence rate for coupled smoothness parameters. It, however, turns out that this is not the case and the proposed adaptive smoothness parameter has a faster convergence rate of $\mathcal{O}(1/k^2)$ for its transient and asymptotic convergence of $\mathcal{O}(e^{-k})$, as opposed to the existing rate of $\mathcal{O}(1/k)$ [105]. This two-phase behavior with these rates is also evident in the proposed smoothing rule (cf. Lemma 3.2.2).

Roadmap. The paper is organized as follows: Section 3.1 reviews Nesterov's smoothing technique for solving nonsmooth minimization. We propose our adaptive-smoothing algorithm in Section 3.2 and provide the technical proof of theorems in Section 3.3. Section 3.4 benchmarks our algorithm in several applications.

Notation. $\mathbb{R}^n$ shows the real vector space, we denote the standard inner product by $\langle\cdot,\cdot\rangle$ and ℓ_p-norm by $\|\cdot\|_p$ (and by $\|\cdot\|$, we mean the Euclidean standard 2-norm). If f is differentiable, $\nabla f(x)$ represents the gradient of f at x. The function f is called L-Lipschitz for some $L>0$ if

$$|f(x)-f(y)|\leq L\|x-y\|, \qquad \forall x,y\in\mathbb{R}^n.$$

The function f is μ-smooth if its gradient is μ-Lipschitz, i.e., $\|\nabla f(x)-\nabla f(y)\|\leq\mu\|x-y\|$. The convex function f is called ρ-locally strongly convex at x^*, if there exists $\varepsilon>0$ so that the function $f(x)-\frac{\rho}{2}\|x\|^2$ is convex over the ball $\mathbb{B}_\varepsilon(x^*)$. We call f ∞-locally strongly convex if f is ρ-locally strongly convex for any $\rho>0$. The Fenchel conjugate of f is

defined as $f^*(x) := \sup_y \langle x, y\rangle - f(y)$. The prox operator for function h is defined as

$$\text{prox}_h(x) := \arg\min_u h(u) + \frac{1}{2}\|u - x\|^2. \tag{3.4}$$

A function is "prox-friendly" if the operator (3.4) is available (computationally or explicitly). We also denote the gradient mapping of two convex functions f and h by

$$G^f_{\zeta h}(x) := \frac{1}{\zeta}\Big(x - \text{prox}_{\zeta h}\big(x - \zeta\nabla f(x)\big)\Big), \tag{3.5}$$

where ζ is a positive scalar and has a stepsize interpretation. The gradient mapping is available if f is differentiable and h is prox-friendly.

3.1. STATE OF THE ART ON NONSMOOTH OPTIMIZATION

In this section, we review the current state of the art in smoothing nonsmooth functions and discuss a possible challenge that may emerge in dealing with multiple smoothing terms.

3.1.1. NESTEROV'S SMOOTHING TECHNIQUE

Consider a possibly nonsmooth convex function f that we assume to be prox-friendly and L_f-Lipschitz continuous. A key object in smoothing techniques is the Moreau envelope defined as

$$f_\mu(x) := \min_y f(y) + \frac{1}{2\mu}\|y - x\|^2. \tag{3.6}$$

The Moreau envelope is a smooth lower approximation of a function at every point, i.e., $f_\mu(x) \le f(x)$ for all $x \in \mathbb{R}^n$. By definition, the objective function of the Moreau envelop (3.6) and the prox operator (3.4) are closely related, namely, the optimizer of (3.6) is $\text{prox}_{\mu f}(x)$. The following lemma indicates several known properties of the smooth approximation (3.6) that are central for algorithms in this context. For brevity, we skip the proof and refer interested readers to [104] for further details.

Lemma 3.1.1 (Smoothness regularity)**.** *Let f_μ be defined as in (3.6). Then, the following holds:*

(i) **Dual reformulation:** $f_\mu(x) = \max_z \Big\{\langle x, z\rangle - f^*(z) - \frac{\mu}{2}\|z\|^2\Big\}$.

(ii) **Uniform bound:** $f_\mu(x) \le f(x) \le f_\mu(x) + \frac{\mu}{2}L_f^2$, *where L_f is the Lipschitz constant of f.*

(iii) **Gradient evaluation:** $\nabla f_\mu(x) = \frac{1}{\mu}\Big(x - prox_{\mu f}(x)\Big)$.

(iv) **Smoothness:** $f_\mu(\cdot)$ *is* $\frac{1}{\mu}$*-smooth, i.e.,* $\|\nabla f_\mu(x) - \nabla f_\mu(y)\| \le \frac{1}{\mu}\|x - y\|$.

Lemma 3.1.1 paves the way to a power smoothing technique as follows: The uniform bound (ii) allows us to choose a minimum value for the smoothing parameter so that the smoothed function f_μ remains in a desired ε-vicinity of the original function f. Thanks

to the smoothness result in (iv) and under the assumption that f is prox-friendly, one can apply Nesterov's accelerated algorithm using the gradient evaluation (iii) and optimize f_μ. The choice of Nesterov's algorithm (Algorithm 0) is justified by the fact that it is the fastest general convex optimization algorithm for smooth functions [39].

Algorithm 0 Nesterov's accelerated method for $1/\mu$-smooth functions [39]

Input: given initial conditions $y_1 = x_1$, $\beta_0 > 0$, and smoothing constant μ.

$y_{k+1} = x_k - \zeta \nabla f_\mu(x_k)$, with the stepsize $\zeta = \mu$

$x_{k+1} = (1-\gamma_k) y_{k+1} + \gamma_k y_k$

$\beta_{k+1} = \frac{1}{2}\big(1 + \sqrt{1+4\beta_k^2}\big), \gamma_k = \dfrac{1-\beta_k}{\beta_{k+1}}$

This idea is first proposed in [96] in which the smoothness parameter is proposed to be the constant $\mu = 2\varepsilon/L_f^2$ where ε is an apriori desired precision. This yields an overall complexity of $\mathcal{O}(1/\varepsilon)$ in terms of the precision parameter ε. More recently, [105] proposes an adaptive version of the smoothing technique. Our proposed adaptive smoothing technique also follows a similar spirit as in [105], but the key feature is to pair the update rule with the momentum parameter of the accelerated method. Before proceeding with that, we also wish to briefly comment on our motivation for another feature of our proposed algorithm that allows for two nonsmooth terms.

Several studies have been devoted to developing methods for computing the prox-operator (3.4) of a sum of two (multiple) functions, see the recent work [1], [2], [3], [106] and the references therein. These methods typically provide various assumptions under which we have

$$\operatorname{prox}_{f+h} = \operatorname{prox}_f \circ \operatorname{prox}_h, \tag{3.7}$$

where "$\circ$" denotes the mapping composition. However, these conditions are still restrictive, and the prox-operator of the sum of two prox-friendly functions may not have a closed-form solution and may be in general computationally demanding (e.g., sparse regression and semi-definite programming). Motivated by this, we develop our algorithm so that it allows for two (multiple) nonsmooth terms, i.e., in (3.3) both functions f and h are possibly nonsmooth but prox-friendly.

3.2. PROPOSED ALGORITHM AND CONVERGENCE ANALYSIS

This section presents the main algorithm of this paper and its global and local convergence results.

3.2.1. ALGORITHM

The proposed method is given in Algorithm 1. The algorithm follows Nesterov's accelerated method in Algorithm 0 with the difference in the choice of stepsize ζ, which is coupled with the momentum parameter at two consecutive steps β_k and β_{k+1} (Lemma 3.2.1). This connection is established by looking at the increment of the function as described in one of the preliminary lemmas in Section 3.3 (Lemma 3.3.2).

Our first result quantifies the performance of Algorithm 1 after T iterations for a specific choice of adaptive smoothing sequence $(\mu_k)_{k\geq 0}$.

Algorithm 1 Adaptive accelerated smoothing method

Input: given initial conditions $y_1 = x_1$, $\beta_0 > 0$, and the to-be-defined sequence $(\mu_k)_{k\geq 0}$.

$$\beta_{k+1} = \frac{1+\sqrt{1+4\beta_k^2}}{2}, \quad \gamma_k = \frac{1-\beta_k}{\beta_{k+1}}$$

$$y_{k+1} = x_k - \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k), \quad \zeta_k = \mu_{k+1}$$

$$x_{k+1} = (1-\gamma_k) y_{k+1} + \gamma_k y_k$$

Lemma 3.2.1 (Optimality gap in adaptive smoothing)**.** *Consider the optimization problem in (3.3) and Algorithm 1 with adaptive smoothing variable* $\mu_k = \max\left\{\mu_{k-1}\left(\frac{3\beta_k^2}{\beta_{k-1}^2} - 1\right)^{-1}, c\right\}$, *for some* $c \geq 0$. *Then, after* T *iterations, we have*

$$F(y_{T+1}) - F^\star \leq \frac{L_f^2 \mu_{T+1}}{2} + \frac{E}{2\beta_T^2 \mu_{T+1}}, \tag{3.8}$$

where the constant E *is*

$$E = \|u_1\|^2 + \zeta_0 \beta_0^2 \delta_1 + \left(1 - \frac{\mu_1}{\mu_0}\right)\frac{3\mu_0}{2} L_f^2 \zeta_0 \beta_0^2,$$

with $\delta_1 = f_{\mu_1}(y_1) + h(y_1) - F^\star$ *and* $u_1 = \beta_1 x_1 - (\beta_1 - 1) y_1 - x^\star$.

Lemma 3.2.1 serves as a basis for different types of convergence results (global and local) in this study. We note that the right-hand side of the bound (3.8) consists of two terms that are dependent on the adaptive smoothing parameter μ_k and can be used to control the optimality gap. The proof of Lemma 3.2.1 builds on two preparatory lemmas, which we relegate to Section 3.3 to improve the flow of the paper. Before proceeding with the main results concerning the convergence of Algorithm 1, we first provide a lemma that sheds light on the behavior of the smoothing parameter μ_k that we use in the algorithm.

Lemma 3.2.2 (Smoothing parameter convergence)**.** *The sequence generated by the first part of* μ_k *in Lemma 3.2.1, i.e.,* $\mu_{k-1}\left(3\beta_k^2/\beta_{k-1}^2 - 1\right)^{-1}$, *exhibits a transient behavior at the order of* $\mathcal{O}(1/k^2)$ *and an asymptotic linear rate of* $\mathcal{O}(e^{-k})$.

The proof of Lemma 3.2.2 is rather straightforward and we defer it to Section 3.3. The convergence behavior of μ_k helps the global and local convergence guarantees of Algorithm 1. In fact, our first result is to show that Algorithm 1 enjoys a global optimal convergence rate of $\mathcal{O}(1/\varepsilon)$ when the smoothing rule of μ_k is uniformly lower bounded away from zero.

Theorem 3.2.3 (Global sublinear convergence)**.** *Suppose the sequence of the smoothing parameters* $(\mu_k)_{k\geq 0}$ *is defined as in Lemma 3.2.1 where* $c = \varepsilon/L_f^2$. *Then, the outcome of Algorithm 1 after* T *iterations satisfies* $F(y_T) - F^* \leq \varepsilon$ *if* $T \geq 2L_f\sqrt{E}/\varepsilon$.

Proof of Theorem 3.2.3. The proof builds on the assertion of Lemma 3.2.1. From Lemma 3.2.2, we know that the first phase of μ_k is decreasing with the rate of at least $\mathcal{O}(1/k^2)$. Therefore, the smoothing rule μ_k also converges to ε/L_f^2 with the similar rate of at least $\mathcal{O}(1/k^2)$. It then suffices to determine the minimum number of iterations T so that μ_{T+1} reaches ε/L_f^2. To this end, note that since $\beta_k \geq k/2$ for all k, we have:

$$F(y_{T+1}) - F^\star \leq \frac{\varepsilon}{2} + \frac{2L_f^2 E}{T^2 \varepsilon}.$$

3

Hence, the ε precision is guaranteed if T greater than $2L_f\sqrt{E}/\varepsilon$ which concludes the desired assertion. □

We note that the global sublinear rate provided in Theorem 3.2.3 is worst-case optimal for the class of nonsmooth functions (3.3) [59], [96]. Another key message of this study is that under a certain condition over the nonsmoothness, Algorithm 1 can achieve a linear, but local, convergence rate when the smoothing parameter follows the sequences in Lemma 3.2.2 ($c = 0$ in Lemma 3.2.1). This two-phase behavior is evident in the smoothing rule as elucidated in Lemma 3.2.2.

Let us remind that the stepsize in Algorithm 1 is dictated by the smoothing parameter (i.e., $\zeta_k = \mu_{k+1}$). It is surprising to have an exponentially fast diminishing stepsize in optimization algorithms. However, the following result addresses this phenomenon and demonstrates that, under a specific ∞-locally strong convexity condition, such a rate facilitates local linear convergence.

Theorem 3.2.4 (Local linear convergence)**.** *Consider the optimization problem in* (3.3) *where the hypotheses of Lemma 3.2.1 hold with $c = 0$. In addition, suppose the nonsmooth term f is ∞-locally strongly convex at the solution x^* of* (3.3)*, i.e., for any $\rho > 0$ the function $f(x) - \rho\|x\|^2$ is locally convex in a neighborhood containing x^*. Then, for any initial condition $\mu_0 > 0$, the sequence $(y_k)_{k\geq 0}$ generated by Algorithm 1 converges locally linearly to x^*, i.e., there exist $\varepsilon > 0$ and $\alpha \in (0,1)$ such that for all $x_0 \in \mathbb{B}_\varepsilon(x^*)$ there exists k_0 where for all $k \geq k_0$, we have $F(y_k) - F^* \leq \alpha^{(k-k_0)}(F(y_{k_0}) - F^*)$.*

Before proceeding with the proof of this theorem, let us note that the key feature leading to local linear convergence is the fact that when f is ∞-locally strongly convex, the condition number of its Hessian $\partial^2 f_\mu$ at its optimizer is uniformly bounded for all sufficiently small μ. This allows the first-order accelerated algorithm to converge locally linearly and independently of the smoothness parameter μ_k. It is also worth noting that the linear convergence rate is uniform for all the initial conditions $x_0 \in \mathbb{B}_\varepsilon(x^*)$. Let us formally explain this as follows.

Proof of Theorem 3.2.4. Let us define $F_{\mu_k}(x) := f_{\mu_k}(x) + h(x)$. Note that using the basic property of the Moreau envelope in Lemma 3.1.1(ii), we have the inequality

$$F(y_k) - F(x^*) \leq \frac{\mu_k}{2} L_f^2 + F_{\mu_k}(y_k) - F_{\mu_k}^* \tag{3.9}$$

where $F_{\mu_k}^* = \min_{x\in\mathbb{R}^n} F_{\mu_k}(x) = F_{\mu_k}(x_{\mu_k}^*)$. Thanks to Lemma 3.2.2, we know that the term μ_k, and as such the first term of the upper bound (3.9), converges to zero exponentially fast. Therefore, it suffices to show that $F_{\mu_k}(y_k) - F_{\mu_k}^*$ converges to zero linearly with a rate

independently from μ_k. To this end, we recall that the Moreau envelope is locally $\mathscr{C}^2$-differentiable [107]. From the recent work [108], the maximum and minimum Hessian eigenvalues of f_{μ_k} at the optimal point $x^*_{\mu_k}$ can be bounded by

$$\lambda_{\max}\left(\frac{\partial^2}{\partial x^2} f_{\mu_k}(x^*_{\mu_k})\right) \leq \mu_k^{-1}, \quad \lambda_{\min}\left(\frac{\partial^2}{\partial x^2} f_{\mu_k}(x^*_{\mu_k})\right) \geq \left(\rho^{-1}+\mu_k\right)^{-1}.$$

where ρ is the strongly convex parameter of f_{μ_k} (i.e., $f_{\mu_k}(x) - \rho\|x\|^2/2$ is convex). This observation yields the bound on the condition number $\kappa := \lambda_{\max}/\lambda_{\min}$ of the Hessian as $1 \leq \kappa \leq 1 + (\rho\mu_k)^{-1}$. Recall that thanks to the ∞-locally strong convexity feature of f, we know that the parameter ρ can be arbitrarily high as we are closer to the solution x^*. In other words, there exists a sufficiently small $\varepsilon > 0$ where we can choose ρ large enough for all $x \in \mathbb{B}_\varepsilon(x^*)$. Since the (accelerated)-proximal gradient descent methods converge linearly with the rate of $(1-1/\kappa)$ [109], we can deduce that for all sufficiently small enough μ_k, Algorithm 1 enjoys linear convergence in a neighborhood of the solution x^*. □

Note that the ∞-local strong convexity of f is essential for having locally linear convergence in Theorem 3.4. Namely, this property enables a linear decrease in the smoothing parameter while the strong convexity parameter, characterized by $\rho(x)$, grows sufficiently fast as we converge to the minimizer. Interestingly, this property holds for many popular nonsmooth terms commonly encountered in applications; see Section 3.4 for several such examples and also [110] (e.g., Theorem 3.1) for a detailed analysis of the explicit computation of such parameters.

We close this section with two remarks concerning the class of ∞-locally strongly convex functions and the initial value of the smoothing parameter in the proposed algorithm.

Remark 3.2.5 (∞-locally strong convexity vs. sharpness)**.** *We note that the concept of "∞-locally strong convexity" is closely related to the "sharpness" property [111], but in a slightly weaker manner, in the sense that the former is typically used locally, whereas the latter is posed globally. More specifically, a function f is ∞-locally strongly convex with parameter $\rho_\varepsilon^{\infty\text{-}sc}$ (respectively, sharp with the parameter ρ^{sh}) when the following holds:*

$$\infty\text{-locally strong convexity}: \frac{\rho_\varepsilon^{\infty\text{-}sc}}{2}\|x-x^*\|^2 \leq f(x)-f(x^*), \ \forall\, x \in \mathbb{B}_\varepsilon(x^*), \ \rho_\varepsilon^{\infty\text{-}sc} \underset{\varepsilon\downarrow 0}{\longrightarrow} \infty,$$

$$\text{Sharpness}: \frac{\rho^{sh}}{2}\|x-x^*\| \leq f(x)-f(x^*) \quad \forall\, x.$$

Remark 3.2.6 (Avoid switching rule in Algorithm 1)**.** *In view of Theorem 3.2.4, local linear convergence is achievable if we are sufficiently close to the global optimal point x^*. One can use Algorithm 1 with the lower-bounded smoothing rule (Theorem 3.2.3) to converge to an arbitrarily close neighborhood of the optimal solution wherein linear convergence is guaranteed, and then continue with $c = 0$ in Algorithm 1 to converge to the desired solution with the faster rate anticipated in Theorem 3.2.4. To avoid this switching mechanism, a promising idea is to choose the initial value μ_0 in a way that ensures a sufficient decrease in the objective function in the first phase behavior of μ_k, and as such, guarantees reaching the desired neighborhood $\mathbb{B}_\varepsilon(x^*)$ where the linear convergence is achievable. Note that the*

optimal value of μ_0 typically depends on the initial error $\|x^* - x_0\|$, as we elaborate further in the next section.

3.2.2. FURTHER DISCUSSION, LIMITATION, AND FUTURE DIRECTION

In this part, we provide additional information and insights concerning the proposed smoothing technique, its limitations, and possible future directions.

Further insights behind the adaptive smoothing rule: The main motivation of the adaptive smoothness parameter is to exploit the possibility of having a larger smoothness parameter μ_k (and as such, optimizing smoother approximate function f_μ), which leads to larger stepsizes and potentially faster convergence rate. However, any stepsize larger than ε/L_f^2 can increase the Lyapunov function of Nesterov's accelerated algorithm, which is the key driving force behind our algorithm. This violation turns out to be dependent on the two subsequent algorithm momentum β_k and β_{k+1} and the two subsequent smoothness parameters μ_k and μ_{k+1}. This observation indeed leads to the smoothing rule in Lemma 3.2.1. Furthermore, as explained in Lemma 3.2.2, the smoothing parameter μ_k has an initial decreasing rate of $\mathcal{O}(1/k^2)$, but asymptotically it converges to zero with an exponential rate. This behavior can explain the locally linear convergence of Algorithm 1 discussed in Theorem 3.2.4.

Extension to the smoothing rule: It can be shown that the smoothing parameter μ_k in Lemma 3.2.1 can be generalized to $\mu_k = \max\Big\{b\mu_{k-1}\big(\frac{b(a-1)+a}{a-1}\frac{\beta_k^2}{\beta_{k-1}^2} - 1\big)^{-1}, c\Big\}$, where $a > 1$ and $b > 0$ are hyperparameters that control the behavior of the smoothing parameter μ_k. These parameters can be selected to prevent Algorithm 1 from switching, thereby enabling locally linear convergence, as discussed in Remark 3.2.6. The optimal tuning of these parameters remains unclear to the authors and is a promising direction for future research.

Relation to prior works and the existing performance guarantees: We note that our apriori theoretical results in Theorem 3.2.3 match the state of the art [96] for the particular choice of algorithm parameters $(a, b) = (1, 0)$, but do not improve the global performance. We wish also to note that this is not a rare precedent in optimization algorithm literature that a new algorithm numerically performs better than its formal apriori guarantees. For instance, the well-known FISTA algorithm proposed by [44] has the same convergence bounds as in Nesterov's method [96] when the latter is restricted to the proximal setting.

Limitation and future directions: The primary limitation of the proposed method lies in the selection of the initial condition μ_0 (see Remark 3.2.6). When μ_0 is too small and $c = 0$, there is a risk that the algorithm fails to reach the region where linear convergence is attainable. This can result in stagnation of iterations due to the diminishing but still summable rate of decrease in the sequence of stepsizes $\zeta_k = \mu_{k+1}$. Conversely, if μ_0 is excessively large, the algorithm precision is compromised, as it takes longer to reach the linear convergence region. An optimal value for μ_0 may depend on the initial error $\|x^* - x_0\|$ and the local characteristics of the functions involved in (3.3) at x^* [85]. Investigating and analyzing these features are promising avenues for future research.

Another future direction is concerned with the relation between the stepsize and smoothing parameters. In this work, the smoothing parameter dictates the stepsize.

However, if we untangle this dependency, an adaptive stepsize may support the acceleration of the algorithms while an adaptive smoothing parameter enhances the precision. This adaptive stepsize-adaptive smoothing rule can be a promising research direction.

3.3. TECHNICAL PROOFS

In this section, we provide the theoretical proof for Section 3.2 and additional material supporting the technical as well as the numerical investigation of the paper.

3.3.1. DETAILS OF THE THEORETICAL ANALYSIS

Our proof for Lemma 3.2.1 relies on a Lyapunov argument. To this end, we first proceed with two preliminary lemmas.

Lemma 3.3.1 (Gradient mapping)**.** *Let $G^f_{\zeta h}(x)$ be the gradient mapping defined in (3.5) where ζ is a positive scalar, $f(x)$ is smooth, and $h(x)$ is prox-friendly. Then, we have*

$$h\big(x-\zeta G^f_{\zeta h}(x)\big) \le h(y) - \big\langle G^f_{\zeta h}(x) - \nabla f(x), y - \big(x-\zeta G^f_{\zeta h}(x)\big)\big\rangle, \quad \forall x, y \in \mathbb{R}^d.$$

Proof of Lemma 3.3.1. Defining $u = \mathrm{prox}_{\zeta h}(w)$, we can write

$$u = \mathrm{prox}_{\zeta h}(w) \Leftrightarrow u = \arg\min_u h(u) + \frac{1}{2\zeta}||u-w||^2 \Leftrightarrow 0 \in \partial h(u) + \frac{1}{\zeta}(u-w) \Leftrightarrow w - u \in \zeta \partial h(u)$$

By defining $u := x - \zeta G^f_{\zeta h}(x)$ and $w := x - \zeta \nabla f(x)$, we have

$$\underbrace{x-\zeta G^f_{\zeta h}(x)}_{u} = x - \zeta \frac{1}{\zeta}(x - \mathrm{prox}_{\zeta h}(x-\zeta\nabla f(x))) = \mathrm{prox}_{\zeta h}(\underbrace{x-\zeta\nabla f(x)}_{w}) \Rightarrow G^f_{\zeta h}(x) - \nabla f(x)$$
$$\in \partial h(x - \zeta G^f_{\zeta h}(x)).$$

Using the convexity of h, we can write

$$h(x-\zeta G^f_{\zeta h}(x)) \le h(y) - \langle G^f_{\zeta h}(x) - \nabla f(x), y - (x - \zeta G^f_{\zeta h}(x))\rangle.$$

□

Lemma 3.3.1 is an inherent property of the gradient mapping and essentially represents a convex inequality that is particularly helpful to control the increment of the original function F in (3.3).

Lemma 3.3.2 (Increment bound)**.** *Suppose function f is prox-friendly and f_μ is the smooth approximation (3.6). Considering the update $y_{k+1} = x_k - \zeta_k G^{f_{\mu_{k+1}}}_{\zeta_k h}(x_k)$, for any $z \in \mathbb{R}^d$ we have*

$$f_{\mu_{k+1}}(y_{k+1}) - f(z) + h(y_{k+1}) - h(z) \le -\frac{1}{2\zeta_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\zeta_k}\langle y_{k+1} - x_k, x_k - z\rangle \quad (3.10)$$

Proof of Lemma 3.3.2. By using the uniform boundedness of f_μ and the definition of convexity, we have

$$f_{\mu_{k+1}}(y_{k+1}) - f(z) + h(y_{k+1}) - h(z) \overset{(ii)}{\leq} f_{\mu_{k+1}}(x_k - \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k)) - f_{\mu_{k+1}}(z)$$
$$+ h(x_k - \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k)) - h(z) \leq f_{\mu_{k+1}}(x_k - \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k)) - f_{\mu_{k+1}}(x_k)$$
$$+ \langle \nabla f_{\mu_{k+1}}(x_k), x_k - z\rangle \underbrace{- \langle G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k) - \nabla f_{\mu_{k+1}}(x_k), z - (x_k - \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k))\rangle}_{\textbf{Lemma } 3.3.1}$$
$$\leq \langle \nabla f_{\mu_{k+1}}(x_k), x_k - \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k) - x_k\rangle + \frac{\zeta_k^2}{2\mu_{k+1}} \|G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k)\|^2 + \langle G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k), y_{k+1} - z\rangle$$
$$+ \langle \nabla f_{\mu_{k+1}}(x_k), \zeta_k G_{\zeta_k h}^{f_{\mu_{k+1}}}(x_k)\rangle \tag{3.11}$$

The last inequality is valid as a result of the smoothness property of the function $f_{\mu_{k+1}}$ (iv). By adding and subtracting $\frac{1}{\zeta_k}\langle y_{k+1} - x_k, x_k\rangle$ in (3.11), we obtain

$$f_{\mu_{k+1}}(y_{k+1}) - f(z) + h(y_{k+1}) - h(z) \leq \underbrace{\langle y_{k+1} - x_k, \frac{1}{2\mu_{k+1}}\|y_{k+1} - x_k\|^2 - \frac{1}{2\mu_{k+1}}\|y_{k+1} - x_k\|^2\rangle}_{=0}$$
$$- \frac{1}{2\zeta_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\zeta_k}\langle y_{k+1} - x_k, x_k - z\rangle.$$

□

Lemma 3.3.2 plays a key role in the proof of Lemma 3.2.1. The increment bound (3.10) allows for the inclusion of a momentum term that emerges in acceleration. We are now in a position to prove Lemma 3.2.1.

Proof of Lemma 3.2.1. Here, we present the proof for the general selection of the smoothing parameter, as discussed in Section 3.2.2. In the first step, by applying (3.10) and Lemma 2.6 in [104] for two cases of $z = y_k$ and $z = x^*$ to arrive at

$$f_{\mu_{k+1}}(y_{k+1}) - f_{\mu_k}(y_k) - \frac{(\mu_k - \mu_{k+1})}{2}L_f^2 + h(y_{k+1}) - h(y_k) \overset{(ii)}{\leq} f_{\mu_{k+1}}(y_{k+1}) - f(y_k) + \leq$$
$$h(y_{k+1}) - h(y_k) - \frac{1}{2\zeta_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\zeta_k}\langle y_{k+1} - x_k, x_k - y_k\rangle. \tag{3.12a}$$
$$f_{\mu_{k+1}}(y_{k+1}) - f^* + h(y_{k+1}) - h^* \leq -\frac{1}{2\zeta_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\zeta_k}\langle y_{k+1} - x_k, x_k - x^*\rangle. \tag{3.12b}$$

Let us define $\delta_k := f_{\mu_k}(y_k) + h(y_k) - f^* - h^*$. Then, multiplying (3.12a) by $(\beta_k - 1)$ and adding the two sides of the inequality to (3.12b) yields

$$\beta_k\delta_{k+1} - (\beta_k - 1)\delta_k - \frac{(\mu_k - \mu_{k+1})}{2}L_f^2(\beta_k - 1) \leq$$
$$-\frac{\beta_k}{2\zeta_k}\|y_{k+1} - x_k\|^2 - \frac{1}{\zeta_k}\langle y_{k+1} - x_k, \beta_k x_k - (\beta_k - 1)y_k - x^*\rangle.$$

Multiplying the above inequality by $\zeta_k\beta_k$ and considering $\beta_{k-1}^2 := \beta_k^2 - \beta_k$ and $\zeta_k \leq \zeta_{k-1}$, we have

$$\zeta_k\beta_k^2\delta_{k+1} - \zeta_{k-1}\beta_{k-1}^2\delta_k - \frac{(\mu_k - \mu_{k+1})}{2}L_f^2\zeta_k\beta_{k-1}^2 \leq \\ -\frac{1}{2}\Big(\|\beta_k(y_{k+1}-x_k)\|^2 + 2\beta_k\langle y_{k+1}-x_k, \beta_k x_k - (\beta_k-1)y_k - x^*\rangle\Big) \tag{3.13}$$

The right-hand side of (3.13) can be equivalently written as

$$\|\beta_k(y_{k+1}-x_k)\|^2 + 2\beta_k\langle y_{k+1}-x_k, \beta_k x_k - (\beta_k-1)y_k - x^*\rangle = \\ \|\beta_k y_{k+1} - (\beta_k-1)y_k - x^*\|^2 - \|\beta_k x_k - (\beta_k-1)y_k - x^*\|^2. \tag{3.14}$$

By substituting (3.14) into (3.13) and by rearranging the inequality, we have

$$\zeta_k\beta_k^2\delta_{k+1} - \zeta_{k-1}\beta_{k-1}^2\delta_k - \frac{(\mu_k - \mu_{k+1})}{2}L_f^2\zeta_k\beta_{k-1}^2 \leq \\ -\frac{1}{2}\Big(\|\beta_k y_{k+1} - (\beta_k-1)y_k - x^*\|^2 - \|\beta_k x_k - (\beta_k-1)y_k - x^*\|^2\Big) \tag{3.15}$$

Using the update rule of x_{k+1} on the right-hand side of (3.15) reduces to

$$\beta_k y_{k+1} - (\beta_k-1)y_k - x^* = \beta_{k+1}x_{k+1} - (\beta_{k+1}-1)y_{k+1} - x^* \tag{3.16}$$

which is equivalent to

$$x_{k+1} = \frac{(-1+\beta_k+\beta_{k+1})}{\beta_{k+1}}y_{k+1} + \frac{1-\beta_k}{\beta_{k+1}}y_k,$$

By combining (3.15) and (3.16) with $u_k = \beta_k x_k - (\beta_k-1)y_k - x^*$, we obtain

$$\zeta_k\beta_k^2\delta_{k+1} - \zeta_{k-1}\beta_{k-1}^2\delta_k - \frac{(\mu_k - \mu_{k+1})}{2}L_f^2\zeta_k\beta_{k-1}^2 \leq \frac{1}{2}\Big(\|u_k\|^2 - \|u_{k+1}\|^2\Big), \tag{3.17}$$

By defining $\omega_k := \mu_{k+1}/\mu_k$, we can rewrite (3.17) as

$$\zeta_k\beta_k^2\delta_{k+1} - \zeta_{k-1}\beta_{k-1}^2\delta_k - \frac{\mu_k(1-\omega_k)}{2}L_f^2\zeta_k\beta_{k-1}^2 \leq \frac{1}{2}\Big(\|u_k\|^2 - \|u_{k+1}\|^2\Big), \tag{3.18}$$

where $0 < \omega_k \leq 1$. Now, we consider two cases. First, we assume that μ_k strictly decreases in each iteration. To enforce μ_k being strictly decreasing we impose the monotonicity condition $\mu_k < \mu_{k-1}$. Then,

$$-a\frac{\mu_{k-1}}{2}L_f^2\zeta_{k-1}\beta_{k-1}^2 + (a-1)\frac{\mu_k}{2}L_f^2\zeta_k\beta_{k-1}^2 < \\ -a\frac{\mu_k}{2}L_f^2\zeta_k\beta_{k-1}^2 + (a-1)\frac{\mu_k}{2}L_f^2\zeta_k\beta_{k-1}^2 = -\frac{\mu_k}{2}L_f^2\zeta_k\beta_{k-1}^2 \tag{3.19}$$

The inequality (3.19) inspire us to define μ_k as

$$\mu_k = \left(\frac{b(a-1)+a}{a-1}\frac{\beta_k^2}{\beta_{k-1}^2}\mu_k - b\mu_{k-1}\right) \Rightarrow \mu_k = \frac{b\mu_{k-1}}{\frac{b(a-1)+a}{a-1}\frac{\beta_k^2}{\beta_{k-1}^2} - 1}. \tag{3.20}$$

Using the definition of μ_k and its decreasing rate (Lemma 3.2.2), it can be easily shown that $1-\omega_k \le 1-\omega_{k-1}$. Then, by substituting μ_k in the left-hand side of (3.19) and using (3.18), we arrive at a Lyapunov-like inequality

$$\zeta_k\beta_k^2\delta_{k+1}-\zeta_{k-1}\beta_{k-1}^2\delta_k-(1-\omega_{k-1})(ab-b+a)\frac{\mu_{k-1}}{2}L_f^2\zeta_{k-1}\beta_{k-1}^2 \\ +(1-\omega_k)(ab-b+a)\frac{\mu_k}{2}L_f^2\zeta_k\beta_k^2\le\frac{1}{2}\Big(\|u_k\|^2-\|u_{k+1}\|^2\Big) \tag{3.21}$$

By summing up the inequalities in (3.21) from $k=1$ to $k=T$, one obtains

$$\zeta_T\beta_T^2\delta_{T+1}-\zeta_0\beta_0^2\delta_1-(1-\omega_0)(ab-b+a)\frac{\mu_0}{2}L_f^2\zeta_0\beta_0^2\le\frac{1}{2}\|u_1\|^2-\frac{1}{2}\|u_{T+1}\|^2\le\frac{1}{2}||u_1||^2, \tag{3.22}$$

which implies

$$\delta_{T+1}\le\frac{E}{2\zeta_T\beta_T^2},\quad E=\|u_1\|^2+\zeta_0\beta_0^2\delta_1+(1-\omega_0)(ab-b+a)\frac{\mu_0}{2}L_f^2\zeta_0\beta_0^2. \tag{3.23}$$

Second, we assume that the smoothing parameter μ_k strictly decreases by (3.20) until it reaches to some a-priori value, and then we use the fixed μ_k afterward. By summing up the inequalities in (3.21) from $k=1$ to $k=K_\varepsilon$ (the iteration index that we fix μ_k), one obtains

$$\zeta_{K_\varepsilon}\beta_{K_\varepsilon}^2\delta_{K_\varepsilon+1}-\zeta_0\beta_0^2\delta_1-(1-\omega_0)(ab-b+a)\frac{\mu_0}{2}L_f^2\zeta_0\beta_0^2\le\frac{1}{2}\|u_1\|^2-\frac{1}{2}\|u_{K_\varepsilon+1}\|^2, \tag{3.24}$$

Now, by fixing μ_k, we know that $\omega_k=1$ for all $k\ge K_\varepsilon+1$ and then by summing up the inequalities in (3.18) from $k=K_\varepsilon+1$ to $k=T$, one can obtain

$$\zeta_T\beta_T^2\delta_{T+1}-\zeta_{K_\varepsilon}\beta_{K_\varepsilon}^2\delta_{K_\varepsilon+1}\le\frac{1}{2}\|u_{K_\varepsilon+1}\|^2-\frac{1}{2}\|u_{T+1}\|^2, \tag{3.25}$$

Summing (3.24) and (3.25) yields the same inequality as (3.23). Finally, by the definition of δ_{T+1}, (ii), and (3.23), we then have

$$F(y_{T+1})-F^*\le\frac{L_f^2}{2}\mu_{T+1}+\frac{E}{2\zeta_T\beta_T^2}. \tag{3.26}$$

The result in Lemma 3.2.1 can be easily derived by considering $a=2$ and $b=1$. □

Proof of Lemma 3.2.2. We present the proof for the general selection of the smoothing parameter, as discussed in Section 3.2.2. Defining $\alpha_k:=\big[\big(\frac{\beta_{k+1}}{\beta_k}\big)^2+\big(\frac{a}{b(a-1)}\big)\big(\frac{\beta_{k+1}}{\beta_k}\big)^2-\frac{1}{b}\big]^{-1}$, the first part of the smoothing parameter μ_k can be explicitly described as $\mu_{k+1}=\prod_{i\le k}\alpha_i$. To show $\mu_k\le\frac{C_0}{k^2}$ for some constant C_0, it suffices to show $\alpha_i\le e^{-2/i}$ for all $i\ge k_0$ where k_0 is a constant. This claim relies on the fact that this inequality implies the following:

$$\mu_{k+1}=\prod_{i\le k}\alpha_i\le C_0e^{\sum_{i\le k}-2/i}\le C_0e^{-2\log(k)}\le\frac{C_0}{k^2},$$

where $C_0 = \prod_{i<k_0} \alpha_i$. To complete the proof, we show $\alpha_i \leq \mathrm{e}^{-2/i}$ for all sufficiently large i by the following argument:
If $k_0 = 2\big(\log(1+\frac{1}{b(a-1)})\big)^{-1} \leq i$, then $\mathrm{e}^{2/i} \leq 1+\frac{1}{b(a-1)}$, which consequently implies:

$$\mathrm{e}^{2/i} \leq \big(\frac{\beta_i}{\beta_{i-1}}\big)^2 + \frac{1}{b(a-1)}\Big(a\big(\frac{\beta_i}{\beta_{i-1}}\big)^2 - a + 1\Big) = \alpha_i^{-1}.$$

In the last argument, we use the increasing property of $\beta_i \geq \beta_{i-1} > 0$ (i.e., $\big(\frac{\beta_i}{\beta_{i-1}}\big)^2 > 1$), which is a property of the momentum update (line 2 in Algorithm 1). Finally, to show the linear convergence, we show that α_k stays uniformly away from 1 as k increases, ensuring a linear rate for large k. To this end, note that by the definition, we have

$$\begin{aligned}\alpha_k &:= \Big[\big(\frac{\beta_{k+1}}{\beta_k}\big)^2 + \Big(\frac{a}{b(a-1)}\Big)\big(\frac{\beta_{k+1}}{\beta_k}\big)^2 - \frac{1}{b}\Big]^{-1} = \Big[\big(\frac{\beta_{k+1}}{\beta_k}\big)^2\Big(1+\frac{a}{b(a-1)}\Big) - \frac{1}{b}\Big]^{-1} \\ &\leq \Big[\Big(1+\frac{a}{b(a-1)}\Big) - \frac{1}{b}\Big]^{-1} = \Big[1+\frac{1}{b(a-1)}\Big]^{-1} = \frac{b(a-1)}{b(a-1)+1} < 1,\end{aligned}$$

where the second line inequality follows from the monotonicity of the momentum sequence $\beta_{k+1} \geq \beta_k$ and the fact that the coefficient $1 + a/b(a-1) > 0$ (the latter is also ensured by the feasible range of $a > 1$ and $b > 0$). This concludes that α_k is uniformly smaller than 1 by the positive margin of $(b(a-1)+1)^{-1}$.
For validation, Figure 3.1 depicts the sequence in the first part of smoothing parameter μ_k, as defined in Lemma 3.2.2. As illustrated in the figure, the smoothing parameter initially enjoys a convergence rate of $\mathcal{O}(1/k^2)$ but in the second phase, adopts an asymptotically linearly convergence rate. □

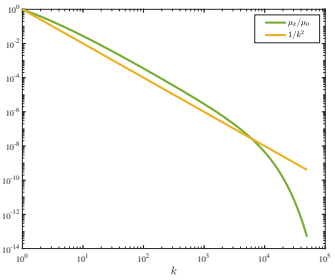


Figure 3.1: Convergence rate of μ_k.

3.4. NUMERICAL EXPERIMENTS

We demonstrate the performance of Algorithm 1 (with $c = 0$) on four popular classes of problems studied in the machine learning and control literature: (1) regression problems for both combination of the ℓ_1 and ℓ_2 norms borrowed from [92], [112], (2) the *MaxCut* problem belonging to the class of semidefinite programming from [113], (3) the Nuclear norm minimization problem and its application in model-free fault diagnosis from [88], [114], and (4) ℓ_1-regularized model predictive control from [86]. To evaluate our performance, we compare our proposed algorithm (Algorithm (1)) with the following methods from the literature: (i) Sub-gradient descent (SGD) [40], the standard

optimization algorithm, (ii) Chambolle-Pock (CP) [90], the state-of-the-art method for sparse regression, or the stochastic smoothing technique (stoch-smooth) [101], a relatively recent method for semidefintie programming, (iii) Nesterov's smoothing (Nes-smooth) [96], and (iv) Tran-Dinh (TD) [105], that is the closest in spirit to our proposed method. Note that in light of Remark 3.2.5, we use sharp convex functions in this study, and the simulations include examples that satisfy the sharpness property. An example of an ∞-locally strongly convex function is $\|\cdot\|_1$. Using the definition of ∞-locally strongly convex functions and the sharpness properties of $\|\cdot\|_1$, there exists a neighborhood of x^* where, for any $\rho > 0$, the function is ρ-strongly convex [85], [110], [111]. Several important prox-friendly functions fall into the category as well, including: $\|x\|_\infty$ and the Nuclear norm of matrices (see the tables "Prox Calculus Rules" and "Prox Computations" on pages 448 and 449 in [72]). Additional examples, formed as combinations of such functions, are also discussed in the literature, for instance, [115] considers the total variation, ℓ_∞-norm, $\ell_1-\ell_2$-norm, and the Nuclear norm. Indeed, the widespread applicability of these functions across various domains motivated us to include several numerical examples in this section. We would also like to highlight that the class of problems studied in this work encompasses several key applications in image processing, image reconstruction, system identification, and control. Specifically, in [89], the authors employ $\ell_2-\ell_1$ and $\ell_1-\ell_1$ optimization for image restoration. In [90], the TV-ℓ_1 model is used for image denoising, while in [91], the $\ell_1-\ell_1$ problem is applied to dictionary learning. Lastly, the ℓ_1-Nuclear norm problem is employed in data-driven fault diagnosis control in [88].

REGRESSION PROBLEMS

We consider the ℓ_i-ℓ_1-regularized LASSO problem

$$F^* := \min_{x\in\mathbb{R}^n} \mathscr{R}(x) + \eta\|x\|_1, \tag{3.27}$$

where $\mathscr{R}(x) = \|Bx-b\|_i$, $i = \{1,2\}$, is a regularized function. the parameters B and b are given and $\eta \geq 0$ is a weighting coefficient. Functions f and h in (3.3) can be written as $f(x) := \mathscr{R}(x)$ and $h(x) := \eta\|x\|_1$, respectively. For the function f, we use the smoothing approximation (3.6).

The test data is generated as follows: Matrix $B \in \mathbb{R}^{n\times n}$ is generated randomly using the standard Gaussian distribution $\mathcal{N}(0,1)$. For simulation, we consider two cases as in [105]. In the first case, we use uncorrelated data, while in the second case (reported in the extended version [7]), we generate B with 50% correlated columns as $B(:,j+1) = 0.5B(:,j) + \text{randn}(:)$. The observed measurement vector b is generated as $b := Bx^\natural + \mathcal{N}(0,0.05)$, where $x^\natural$ is generated randomly using $\mathcal{N}(0,1)$. For the simulation, we set $n = 100$ and the coefficient parameter η is chosen as suggested in [105], [116]. All algorithms are initialized at the same point chosen randomly. In Nes-smooth method the stepsize is $2\cdot 10^{-3}/L_f^2\|B\|^2$ for achieving desired error $\varepsilon = 10^{-3}$ and the stepsizes in primal and dual updates in CP are $1/\|B\|$. In TD and the proposed method, the stepsize is $\zeta = \mu_k/\|B\|^2$. The convergence speed is significantly affected by the initial condition of μ in TD, and following the suggestion by [105], we set $\mu_0 = \mu^* = \|B\|\cdot\|x_0 - x^*\|/\sqrt{3L_f^2}$ for the TD algorithm.

- ℓ_1-ℓ_1**-regularized LASSO.** Here, $i = 1$ for $\mathscr{R}(x)$ in (3.27). Then, f can be written

as $f(x) := \|Bx-b\|_1 = \max_y \{\langle B^\top y, x\rangle - \langle b, y\rangle : \|y\|_\infty \le 1\}$. Hence, we can smooth f using the quadratic prox-function to obtain $f_\mu(x) := \max_y \left\{\langle B^\top y, x\rangle - \langle b, y\rangle - \frac{\mu}{2}\|y\|^2 : y \in \mathbb{B}_\infty\right\}$. Using the analytical solution, one can observe that $\nabla f_\mu(x) = \text{proj}_{\mathbb{B}_\infty}\left(\mu^{-1}(Bx-b)\right)$ and $L_f^2 = n$.

- ℓ_2-ℓ_1**-regularized LASSO.** We consider $i = 2$ for $\mathscr{R}(x)$ in (3.27). Functions f can be written as $f(x) := \|Bx-b\|_2 = \max_y \{\langle B^\top y, x\rangle - \langle b, y\rangle : \|y\|_2 \le 1\}$. Hence, we similarly have $f_\mu(x) := \max_y \left\{\langle B^\top y, x\rangle - \langle b, y\rangle - \frac{\mu}{2}\|y\|^2 : y \in \mathbb{B}_2\right\}$. In this case, we can show that $\nabla f_\mu(x) = \text{proj}_{\mathbb{B}_2}\left(\mu^{-1}(Bx-b)\right)$, and $L_f^2 = 1$.

Figures 3.2a and 3.2d report the results for (3.27) with $i = 1,2$ where all algorithms are initialized at the same point chosen randomly. As shown in Figures 3.2a and 3.2d, the proposed algorithm outperforms other methods.

MaxCut (SEMIDEFINITE PROGRAMMING)

An important application of the first-order smoothing technique is to solve semidefinite programming. This class can essentially be reduced to minimizing the maximum eigenvalue of a matrix as follows:

$$\min_{X \in \mathbb{S}^n} \lambda_{\max}(X) \tag{3.28}$$

It is worth noting that the primal form of all semidefinite programs with a fixed trace can be rewritten as (3.28). As shown in [59], the smooth approximation of $\lambda_{\max}(X)$ can be written as $f_\mu(X) = \mu \log\left(\sum_{i=1}^n \exp(\lambda_i(X)/\mu)\right)$, which is convex and twice differentiable with a gradient

$$\nabla f_\mu(X) = \left(\sum_{i=1}^n \exp(\lambda_i(X)/\mu)\right)^{-1} \sum_{i=1}^n \exp(\lambda_i(X)/\mu) q_i q_i^\top,$$

where q_i is the i^{th} column of the unitary matrix Q in the eigen-decomposition $Q\Sigma Q^T$ of X and $\lambda_1 \ge \ldots \ge \lambda_n$ are the eigenvalues of X. In addition, $f_\mu(X)$ fulfills the uniform boundedness inequality (ii) with $L_f^2 = 2\log n$ and the smoothness parameter $1/\mu$ (cf. (iv) in Lemma 3.1.1). The numerical performance of the smoothing technique is evaluated for the *MaxCut* relaxation which is written in the primal form of the semidefinite program as follows:

$$\max_{X \in \mathbb{S}^n} \mathbf{Tr}(CX) \qquad \text{s.t.} \quad \text{diag}(X) = 1, \quad X \succcurlyeq 0 \tag{3.29}$$

Matrix C is generated using the Wishart distribution with $C = G^\top G/\|G\|_2^2$, where G is a standard Gaussian matrix. Here, we consider a regularized dual form of (3.29) as suggested in [71]

$$\min_{y \in \mathbb{R}^n} \lambda_{\max}(C + \text{diag}(y)) - \langle \mathbf{1}, y\rangle + \eta \mathscr{R}(y), \tag{3.30}$$

where η is a regularization parameter. Figure 3.2b and 3.2e show the simulation results for minimizing problem (3.30) with $n = 100$, 2000 iterations, and two different choices

of $\mathscr{R}(y)$. In the Nes-smooth method, the stepsize is 10^{-3} for achieving a desired error $\varepsilon = 10^{-3}$. The parameters required in the stoch-smooth method are chosen according to [101]. The initial condition for μ in the TD method is $\mu_0 = \mu^* = \|x_0 - x^*\|/\sqrt{6\log n}$. Additional simulations for different η are reported in the extended version of this paper [7].

NUCLEAR NORM MINIMIZATION (MODEL-FREE FAULT DIAGNOSIS)

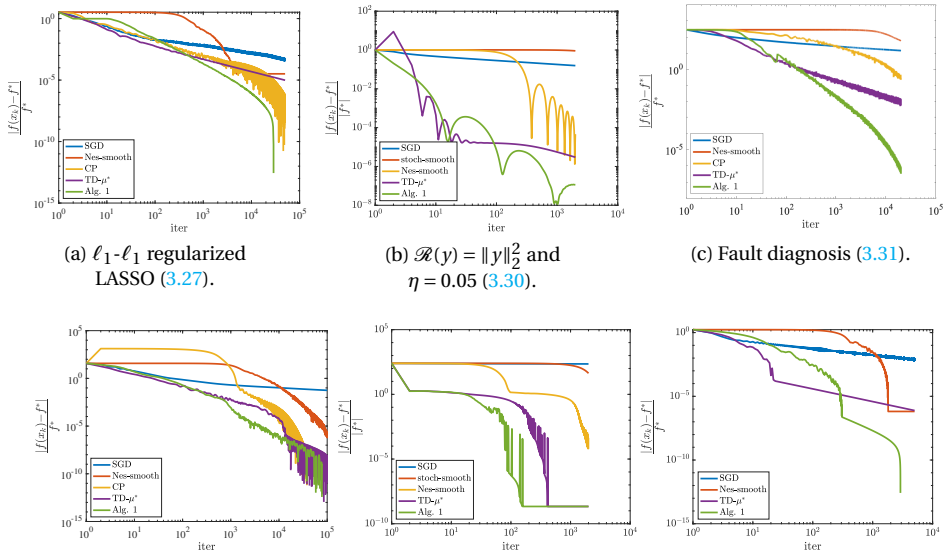
The Nuclear norm regularization is an important problem emerging in several applications including matrix completion [114], [117], compressed sensing [118], principal component analysis [119], system identification and fault diagnosis [120]. In this work, we consider model-free fault diagnosis as in [88], where the authors proposed a model-free, data-driven fault diagnosis approach that aims to identify the system and diagnose faults simultaneously, thereby eliminating the need for an extensive identification phase prior to fault detection. The proposed method reformulates the problem as a convex optimization problem involving the summation of nonsmooth terms. To enhance computational efficiency, proximal and splitting-type algorithms (similar to Nes-smooth, CP, and TD methods) are employed for online implementation. The problem formulation is expressed as follows (see equation (5) in [88]):

$$F^* := \min_{x, X_L, X_S} \frac{1}{2}\|y - Hx\|^2 + \tau\|X_L\|_* + \lambda\|X_S\|_1. \tag{3.31}$$

Here, we consider the same buck converter example in [88] with same parameters and solve the problem using different methods. Here, $X_L \in \mathbb{R}^{m_1 \times m_2}$ and $X_S \in \mathbb{R}^{n_1 \times n_2}$ are linear maps of x that select and rearrange its elements into a low-rank matrix and a sparse vector, respectively. H is a Toeplitz matrix corresponding to the input, output, and fault signals, y denotes the output data, and $\tau \geq 0$ and $\lambda \geq 0$ are regularization parameters. The Nuclear norm of a matrix X, denoted $\|X\|_*$, is the sum of its singular values, equivalently the ℓ_1-norm of the singular values. Both nonsmooth terms in (3.31) are prox-friendly. Therefore, functions f and h in (3.3) can be expressed as $f(x) = \frac{1}{2}\|y - Hx\|^2 + \lambda\|X_S\|_1$ and $h(x) = \tau\|X_L\|_*$ and can be smoothed using (3.6).
Figures 3.2c depict the results of optimizing (3.31). All algorithms are initialized at the same randomly chosen point and run for 20,000 iterations. For the Nes-smooth method, the stepsize is set to $2 \cdot 10^{-3}/(\|H\|^2 + \lambda)$ to achieve a precision level of $\varepsilon = 10^{-3}$. In the primal and dual updates of CP, the stepsizes are set to $1/(\|H\|^2 + \lambda)$, while in TD and the proposed method, the stepsize is determined based on $\mu_k/(\|H\|^2 + \lambda)$.

ℓ_1-REGULARIZED MODEL PREDICTIVE CONTROL

The authors in [86] consider the ℓ_1-regularized MPC problem with horizon H to reduce actuator activity in the quadruple tank system. They use a linearized state-space model



(a) ℓ_1-ℓ_1 regularized LASSO (3.27).

(b) $\mathscr{R}(y) = \|y\|_2^2$ and $\eta = 0.05$ (3.30).

(c) Fault diagnosis (3.31).

(d) ℓ_2-ℓ_1-regularized LASSO (3.27).

(e) $\mathscr{R}(y) = \|y\|_1$ and $\eta = 1$ (3.30).

(f) ℓ_1-Regularized MPC (3.32).

Figure 3.2: Numerical results for four problem classes. The first column shows regression problems with uncorrelated data (Subsection 3.4); the second, the *MaxCut* problem with different choices of $\mathscr{R}(y)$ (Subsection 3.4); and the third, model-free fault diagnosis with Nuclear norm minimization (Subsection 3.4) and ℓ_1-Regularized MPC (Subsection 3.4).

and solve the following MPC problem [121, Eq. (1)]:

$$\begin{aligned} \min_{x,u} \quad & J(x,u) := \sum_{i=0}^{H-1} \left(\|x_i - x_{\text{ref},i}\|_Q^2 + \lambda \|\Delta u_i\|_1 \right) \\ & \qquad + \|x_H - x_{\text{ref},H}\|_Q^2 \\ \text{s.t.} \quad & x_{i+1} = \mathbf{A}x_i + \mathbf{B}u_i, \quad \forall i \in \{0,\dots,H-1\}, \\ & u_{\min} \le u_i \le u_{\max}, \quad \forall i \in \{0,\dots,H-1\}, \\ & \Delta u_i = u_i - u_{i-1}, \quad \forall i \in \{0,\dots,H-1\}. \end{aligned} \tag{3.32}$$

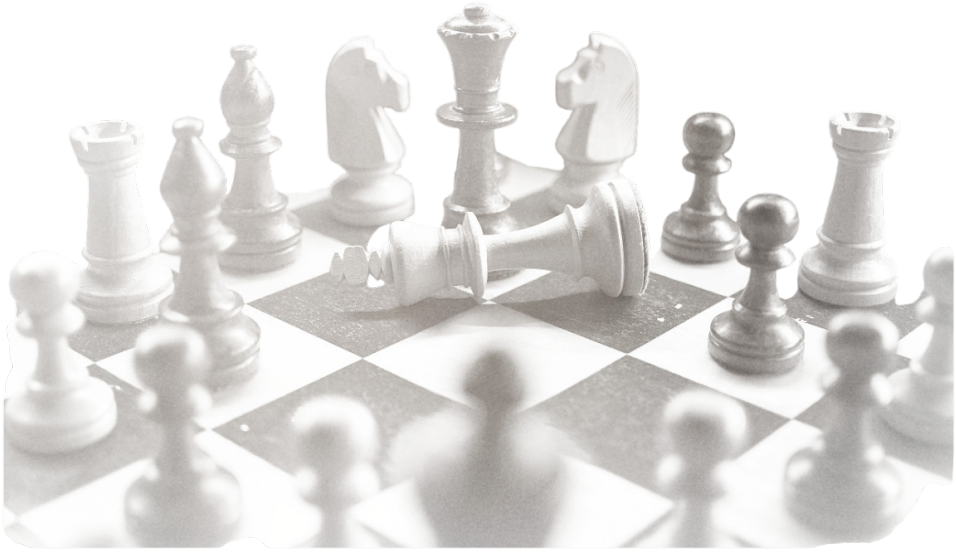
We use the same parameters as in [86] for the system dynamics and solve problem (3.32) using different methods. The objective function can be represented by $f = J(x,u)$ and $h = i(\chi)$ in (3.3), where χ is the feasible set of problem (3.32), and $i(\chi)$ is its indicator function. Therefore, the nonsmooth ℓ_1 term in f, which is prox-friendly, can be smoothed. Figure 3.2f shows the results; as can be seen and mentioned in Remark 3.2.6, the proposed method exhibits two convergence phases, including an initial $\mathcal{O}(1/k^2)$ phase followed by linear convergence. Note that CP method cannot be applied to this problem because we have more than one nonsmooth terms in the objective function.

We provide additional numerical experiments using the real-world dataset LIBSVM [122], as well as extensive simulations with increasing problem dimensions. These re-

sults are presented in the extended version [7]. Furthermore, we note that the ADMM method is suitable for some of the aforementioned problems, and additional simulations for comparison are provided in [7]. However, ADMM may fail to converge, converge very slowly, or be difficult to implement when multiple nonsmooth terms are present, particularly when they appear in a summation that cannot be easily split [123].

II

FAST EQUILIBRIUM SEEKING

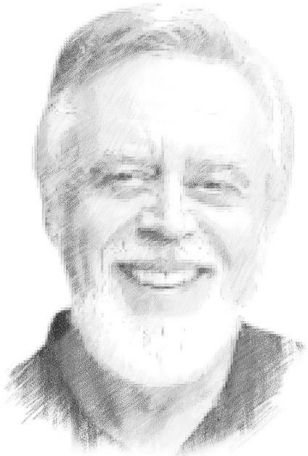


Those who cannot remember
the past are condemned to repeat it.

Winston Churchill

John Nash

Paul Tseng



Dimitri Bertsekas

This part is builds on the seminal contributions of esteemed researchers in monotone inclusion and operator theory, notably Paul Tseng, Dimitri Bertsekas, and Jong-Shi Pang. Their influential work provides essential theoretical foundations for the concepts and methodologies developed herein.

4

A Frank-Wolfe Algorithm for Strongly Monotone Variational Inequalities

Chapters 2 and 3 explore different classes of convex problems, the corresponding algorithms, and their applications. This chapter extends the discussion to a more general framework: ***Variational Inequalities*** *(VI).*

Inspired by the Frank–Wolfe method for minimizing minimax problems, in this chapter we introduce a projection-free method for solving strongly monotone VIs. Unlike standard approaches, such as projected gradient descent (aka value iteration), which requires a projection at every iteration, our method uses a linear minimization oracle, reducing projection costs that can become significant in high-dimensional settings or for complex constraint sets. We demonstrate its effectiveness on the traffic assignment problem, where projection complexity increases exponentially with the number of links.

In this chapter, we consider the variational inequality problem (VIP)

$$\text{find} \quad x^* \in \mathcal{V} \quad \textbf{s.t.} \quad \langle g(x^*), x - x^* \rangle \geq 0, \quad \forall x \in \mathcal{V}, \tag{4.1}$$

where $\mathcal{V}$ is a compact convex set. We assume that the operator g is μ-strongly monotone, L-Lipschitz continuous, and that the solution set of (4.1) is nonempty. The VIP is a general framework that includes several problems and applications in systems and control theory, machine learning, and operations research. For instance, both composite convex minimization and convex-concave minimax saddle point problems can be reformulated as in (4.1) [6], [46], which can be used in robust controller design for systems under uncertainty [124], Nash [125] and Stackelberg games [126], supply chain optimization [127], and adversarial learning problems [128].

Fixed-point problems represent another important class in several domains, such as game theory and machine learning. Although it is straightforward to reformulate the VIP (4.1) as a fixed-point problem, the converse can be computationally beneficial as discussed in [46, section 3]. For instance, variational Nash equilibrium problems can be reformulated as VIPs enabling the use of efficient iterative algorithms [4], [129].

For further applications of VIPs in systems and control theory, operations research, game theory, and machine learning, we refer interested readers to [4], [13], [14], [15], [130] and references therein.

To solve the VIP (4.1), several iterative algorithms have been proposed. For the sake of comparison, we review some recent and closely related methods for solving the general VIP (4.1), where the operator g is (strongly) monotone.

(i) **Projected Gradient Descent [40]:** A classical approach inspired by gradient descent in the optimization literature is

$$x_{k+1} = \text{proj}_{\mathcal{V}}\Big(x_k - \alpha g(x_k)\Big),$$

where the operator $g(\cdot)$ replaces the gradient operator and α is the stepsize. In the literature of fixed point computation (e.g., dynamic programming), this is also known as *Value Iteration*. This method guarantees convergence for strongly monotone and Lipschitz operator g for any stepsize $\alpha \in (0, 2\mu/L^2)$ where μ and L are the strong monotonicity constant and the Lipschitz constant, respectively.

(ii) **Extragradient Descent [131]:** An improvement to the gradient descent approach is to call the operator g twice in order to improve the convergence rate. This yields the algorithm

$$y_k = \text{proj}_{\mathcal{V}}\Big(x_k - \alpha g(x_k)\Big),$$
$$x_{k+1} = \text{proj}_{\mathcal{V}}\Big(x_k - \alpha g(y_k)\Big),$$

with α as the stepsize. Unlike classic projected gradient descent, this method does not require strong monotonicity of the operator g and ensures convergence for a Lipschitz operator when the stepsize $\alpha \in (0, 1/L)$.

(iii) **Accelerated gradient descent [9]:** An influential idea in optimization, first proposed by Nesterov [39], is to accelerate algorithm convergence by incorporating a

so-called *momentum* into the update dynamics. One can draw a parallel, in a similar fashion as in the Gradient Descent (i.e., replacing the gradient with the operator g), and arrive at

$$x_k = \arg\max_{x\in\mathcal{V}} \sum_{i=0}^{k} \alpha_i \left[\langle g(y_i), y_i - x\rangle - \frac{\mu}{2}\|x - y_i\|^2 \right],$$
$$y_{k+1} = \arg\max_{y\in\mathcal{V}} \langle g(x_k), x_k - y\rangle - \frac{\beta}{2}\|y - x_k\|^2,$$

where to ensure the convergence, it suffices to choose the stepszie and momentum coefficients as $\alpha_{k+1} = \frac{\mu}{L}\sum_{i=0}^{k}\alpha_i$ and $\beta = L$.

(iv) **Projected Reflected Gradient Descent [132]:** Evaluating the operator at a reflected point, an extrapolation of the current and previous iterates, enhances stability and convergence in monotone VIP, leading to the following algorithm

$$x_{k+1} = \text{proj}_{\mathcal{V}}\Big(x_k - \alpha g(2x_k - x_{k-1})\Big),$$

where $\alpha \in \Big(0, (\sqrt{2}-1)/L\Big)$ is the stepsize. This method guarantees convergence for a Lipschitz operator, and unlike the extragradient method, it requires only one projection per iteration.

(v) **Golden Ratio Algorithm [46]:** To guarantee convergence when using the Gradient Descent method for Lipschitz but non-strongly monotone operators, negative momentum parameters are required. Introducing a negative momentum ensures convergence, leading to the algorithm

$$y_k = (1-\zeta)x_k + \zeta y_{k-1},$$
$$x_{k+1} = \text{proj}_{\mathcal{V}}\Big(y_k - \alpha g(x_k)\Big),$$

with $\alpha \in \Big(0, 1/(2\zeta L)\Big)$ as the stepsize and momentum parameter $\zeta \in \Big(0, (\sqrt{5}-1)/2\Big]$. Additionally, overcomes the limitation of methods that rely on problem constants (like Lipschitz continuity parameter); the stepsize can be chosen adaptively, leading to the adaptive golden ratio algorithm [46]

$$\alpha_k = \min\left\{(\zeta+\zeta^2)\alpha_{k-1}, \frac{\|x_k - x_{k-1}\|^2}{4\zeta^2\alpha_{k-2}\|g(x_k) - g(x_{k-1})\|^2}\right\}.$$

(vi) **Operator splitting methods [4]:** The operator g can be split into a summation of different operators. Solving VIP (4.1) in these cases is equivalent to solving the fixed-point problem $0 \in g_1(x) + g_2(x)$, where $g(x) + \mathcal{N}_{\mathcal{V}}(x) = g_1(x) + g_2(x)$ and $\mathcal{N}_{\mathcal{V}}(x)$ is the normal cone of the compact, convex set $\mathcal{V}$ at the point x. The iterative update of each sub-operator leads to convergence towards the solution. A well-known class of these algorithms is the Douglas-Rachford splitting method [133], which can be written as

$$y_{k+1} = (I + \alpha g_2)^{-1}\Big(x_k - \alpha g_1(x_k)\Big),$$
$$x_{k+1} = x_k + \zeta(y_{k+1} - x_k),$$

where the stepsize can be chosen $\alpha \in (0,1]$ and $\zeta \in (0,2)$ is a relaxation parameter. The convergence of this method is guaranteed for different cases of g_1 and g_2; we refer interested readers to [134], [135] for further details. Additionally, we refer them to [136] for further algorithms related to applications of monotone variational inequalities and an open source Python toolbox.

In the methods mentioned above, the projection operator onto the feasible set $\mathcal{V}$ ($\text{proj}_{\mathcal{V}}(\cdot)$) is a necessary operation of the algorithm which can be costly in some cases. Frank-Wolfe (FW) is a classical approach to avoid this projection complexity by resorting to a linear oracle minimization (as opposed to the quadratic optimization of the projection operator) over the same set. Table 4.1 summarizes the computational complexity of the linear oracle minimization and the projection onto certain sets [137], [138].

Table 4.1: Complexity of linear minimization and projection. The parameter ε denotes the precision of linear minimization or projection operator.

Set	Linear minimization	Projection
(1) n-dimensional ℓ_p-ball, $p \neq 1,2,\infty$.	$\mathcal{O}(n)$	$\tilde{\mathcal{O}}(n/\varepsilon^2)$
(2) Nuclear norm ball of $n \times m$ matrices. θ and τ denote the number of non-zero entries and the top singular value of the projected matrix, respectively.	$\mathcal{O}(\theta \ln(m+n)\sqrt{\tau/\varepsilon})$	$\mathcal{O}(mn\min\{m,n\})$
(3) Flow polytope on a graph with m vertices and n edges with capacity bound on edges.	$\mathcal{O}((n\log m)(n+m\log m))$	$\mathcal{O}(n^4\log n)$
(4) Birkhoff polytope ($n \times n$ doubly stochastic matrices).	$\mathcal{O}(n^3)$	$\tilde{\mathcal{O}}(n^2/\varepsilon^2)$

For concave-convex minimax problems and inspired by the recent works [139], [140], [141], the authors of [10] and [142] prove convergence of different types of FW algorithms. They demonstrate the numerical efficiency of the FW algorithm via illustrative saddle point problems, in which projecting onto the corresponding sets is computationally demanding or even intractable; see the perfect matching problem as an example of this class of problems [143]. Motivated by this, a natural question is whether the FW algorithm can also be utilized for solving the VIP (4.1). To the best of our knowledge, this question is still largely unexplored, which is the focus of this study.

Contribution. Following the footsteps of the accelerated Nesterov's technique for solving strongly monotone variational inequalities [9] and the FW algorithm for solving saddle point problems [10], we propose a novel accelerated algorithm with FW method as an oracle for solving the strongly monotone VIP (4.1) and provide a non-asymptotic convergence rate. To validate the theoretical results, we implement the proposed method in the traffic assignment problem, an important application in transportation and operation research [11], [12], [13], where the complexity of the problem (the corresponding set) increases exponentially with the number of variables.

Roadmap. The paper is organized as follows: Section 4.1 reviews the Frank-Wolfe technique and states lemmas for solving *minmax* saddle point problems. In Section 4.2, we propose our algorithm and provide the technical proofs of the convergence theorems. Section 4.3 benchmarks the proposed algorithm in a traffic assignment application. Finally, the conclusion and future research directions are given in Section 4.4.

Notation. Let $\mathcal{V}$ be a finite-dimensional real Hilbert space equipped with the standard inner product $\langle\cdot,\cdot\rangle$ and the associated norm $\|\cdot\|$ (so that we may simply write $\|x\|^2 =$

$\langle x, x \rangle$). We define the normal cone of the set $\mathcal{V}$ at the point $x \in \mathcal{V}$ as $\mathcal{N}_{\mathcal{V}}(x) = \{u : u^\top x \geq u^\top y, \forall y \in \mathcal{V}\}$. The operator $\text{proj}_{x \in \mathcal{V}}$ denotes the projection onto set $\mathcal{V}$ with respect to the underlying inner product norm (i.e., $\text{proj}_{x \in \mathcal{V}}(x) := \arg\min_{y \in \mathcal{V}} \|x - y\|$). We define the diameter and boundary distance of a set $\mathcal{X}$ as

$$D_{\mathcal{X}} := \sup_{x, x' \in \mathcal{X}} \|x - x'\| \quad \text{and} \quad \sigma_{\mathcal{X}}(x) := \min_{s \in \partial \mathcal{X}} \|x - s\|.$$

The function $F(x, y)$ is $(\mu_{\mathcal{X}}, \mu_{\mathcal{Y}})$ strongly convex-concave if $F(x, y) - \frac{\mu_{\mathcal{X}}}{2}\|x\|^2 + \frac{\mu_{\mathcal{Y}}}{2}\|y\|^2$ is convex-concave. The constants L_{XY} and L_{YX} are the cross smooth constants of $F(x, y)$ (or equivalently the Lipschitz constants of ∇F) over the set $\mathcal{X} \times \mathcal{Y}$ if for all $x, \bar{x} \in \mathcal{X}$ and $y, \bar{y} \in \mathcal{Y}$

$$\|\nabla_x F(x, y) - \nabla_x F(x, \bar{y})\| \leq L_{XY}\|y - \bar{y}\|,$$
$$\|\nabla_y F(x, y) - \nabla_y F(\bar{x}, y)\| \leq L_{YX}\|x - \bar{x}\|.$$

For brevity, we refer to $F(x, y)$ as a smooth function with constant $L_0 = \max(L_{XY}, L_{YX})$ if it is cross-smooth over $\mathcal{X} \times \mathcal{Y}$.

4.1. ASSUMPTIONS AND TECHNICAL PRELIMINARIES

Before proceeding with the solution to the VIP (4.1), we begin with some assumptions and lemmas that will be used throughout the paper. We note that most of the results build on the seminal work by Nesterov [9] and Jaggi [139]. The following assumptions hold throughout this study.

Assumption 4.1.1 (Operator regularity)**.** *We assume that the solution set of VIP* (4.1) *is nonempty, where* $\mathcal{V}$ *is a compact, convex set and the operator* g *satisfies the following:*

(i) L-Lipschitzness: $\|g(x) - g(y)\| \leq L\|x - y\|, \quad \forall x, y \in \mathcal{V},$

(ii) μ*-strong monotonicity:* $\langle g(x) - g(y), x - y \rangle \geq \mu\|x - y\|^2, \quad \forall x, y \in \mathcal{V}.$

Next, we start with a minimax problem, a core component of our convergence analysis for VIP (4.1). Specifically, by leveraging the following minimax oracle at each iteration, we aim to eliminate the need for costly projection steps:

$$\min_{x \in \mathcal{X}} \max_{y \in \mathcal{Y}} F(x, y). \tag{4.2}$$

An important concept relevant to the convergence of the *minimax* problem (4.2), which helps us measure the convergence rate of (4.2), is the error function, defined as

$$h_k := F(x_k, \hat{y}_k) - F(\hat{x}_k, y_k), \tag{4.3}$$

where $\hat{x}_k = \arg\max_{x \in \mathcal{X}} F(x, y_k)$ and $\hat{y}_k = \arg\max_{y \in \mathcal{Y}} F(x_k, y)$. The following proposition summarizes the convergence of (4.2) in terms of the error function h_k, using the FW algorithm.

Proposition 4.1.2 (FW-minimax oracle convergence [10, Theorem 1])**.** *Let* $F(x, y)$ *be* L_0*-smooth continuous and* $(\mu_{\mathcal{X}}, \mu_{\mathcal{Y}})$ *strongly convex-concave on a convex, compact set* $\mathcal{X} \times$

Algorithm 3 FW-minimax oracle for (4.2) [10]

1: Let $z_0 = (x_0, y_0) \in \mathscr{X} \times \mathscr{Y}$
2: **for** $k = 0 \dots T$ **do**
3: Compute the partial gradients of F: $r_k = \begin{pmatrix} \nabla_x F(x_k, y_k) \\ -\nabla_y F(x_k, y_k) \end{pmatrix}$,
4: Compute the desired direction: $s_k = \arg\min_{z \in \mathscr{X} \times \mathscr{Y}} \langle z, r_k \rangle$,
5: Compute the stepsize:
$\alpha_k = \min\left(1, \frac{\nu}{2C}\langle z_k - s_k, r_k\rangle\right)$ or $\alpha_k = \frac{2}{2+k}$ (based on Proposition 4.1.2)
6: Compute the next iteration: $z_k = (1-\alpha_k)z_k + \alpha_k s_k$.

$\mathscr{Y}$ and (x_*, y_*), the solution of (4.2), is in the interior of $\mathscr{X} \times \mathscr{Y}$. Consider applying the Frank-Wolfe algorithm (Algorithms 3) for saddle point problem (4.2) with the stepsize

$$\alpha_k = \min\left(1, \frac{\nu}{2C}\langle z_k - s_k, r_k\rangle\right), \tag{4.4}$$

where the constants are

$$\rho = \nu^2 \frac{\sigma_\mu^2}{2C}, \quad C = \frac{L_0 D_{\mathscr{X}}^2 + L_0 D_{\mathscr{Y}}^2}{2},$$
$$\nu = 1 - \frac{\sqrt{2}}{\sigma_\mu} \max\left\{\frac{D_{\mathscr{X}} L_{XY}}{\sqrt{\mu_{\mathscr{Y}}}}, \frac{D_{\mathscr{Y}} L_{YX}}{\sqrt{\mu_{\mathscr{X}}}}\right\},$$
$$\sigma_\mu = \sqrt{\min(\mu_{\mathscr{X}} \sigma_{\mathscr{X}}^2(x_*), \mu_{\mathscr{Y}} \sigma_{\mathscr{Y}}^2(y_*))}.$$

Then, the convergence rate for the error h_k (4.3) is $\mathscr{O}\left((1-\rho)^{\frac{k}{6}}\right)$.
Moreover, if $\sigma_\mu > 2\max\left\{\frac{D_{\mathscr{X}} L_{XY}}{\mu_{\mathscr{Y}}}, \frac{D_{\mathscr{Y}} L_{YX}}{\mu_{\mathscr{X}}}\right\}$ and the stepsize is set to $\alpha_k = \frac{2}{2+k}$, then the error h_k (4.3) decreases at a sublinear rate of $\mathscr{O}(1/k)$.

Remark 4.1.3 (FW-variants)**.** *Other FW variants, including the away-step and pairwise algorithms, achieve similar convergence guarantees under alternative assumptions to those in Proposition 4.1.2, such as when the feasible set is a polytope. These variants often yield better practical performance by allowing corrections to previously chosen directions, which helps avoid flat regions and reduces zig-zagging behavior near the solution. More recently, the authors in [142] proposed a projection-free method under assumptions similar to those in Proposition 2.2, but without requiring the solution to lie in the interior of the feasible set and their method is based on a three-loop algorithm. The proposed FW Algorithm 3 can be replaced by these alternatives. For further details, we refer interested readers to [10], [142].*

Proposition 4.1.2 lays the foundation for proving the convergence of the projection-free algorithm for solving VIP (4.1). It avoids expensive projection steps by instead solving a strongly convex-concave *minimax* problem at each iteration, which is the central motivation behind this study.

We move forward by introducing a gap function to measure the optimality gap of VIP (4.1) and by proposing two lemmas that are central to the development of the algorithm

in this paper.
It is not difficult to see that, in light of the strong monotonicity of the operator g, the solution of VIP (4.1), x^*, satisfies the following inequality

$$\langle g(y), x^* - y\rangle + \frac{\mu}{2}\|y - x^*\|^2 \leq \langle g(x^*), x^* - y\rangle - \frac{\mu}{2}\|y - x^*\|^2 \leq 0, \quad \forall\, y \in \mathcal{V}. \tag{4.5}$$

In order to measure the the approximated solution of (4.1), we introduce a gap function $f(x)$ and the following lemma.

$$f(x) := \sup_{y\in\mathcal{V}} \left\{\langle g(y), x - y\rangle + \frac{\mu}{2}\|y - x\|^2\right\}. \tag{4.6}$$

Lemma 4.1.4 (Gap function properties, [9, Theorem 1])**.** *The gap function $f(x)$ is a non-negative, well-defined, μ-strongly convex function on $\mathcal{V}$ and vanishes at the unique solution of* (4.1).

Using Lemma 4.1.4 and the definition (4.6), our goal is to minimize $f(x)$, which is equivalent to solving the VIP (4.1). To this end, let us define the following quantities

$$S_N := \sum_{i=0}^{N} \lambda_i, \quad \tilde{y}_N := \frac{1}{S_N}\sum_{i=0}^{N} \lambda_i y_i, \quad \Delta_N := \max_{x\in\mathcal{V}}\left\{\sum_{i=0}^{N} \lambda_i \left[g(y_i), y_i - x\rangle - \frac{\mu}{2}\|x - y_i\|^2\right]\right\}, \tag{4.7}$$

where $\{y_i\}_{i=0}^N \subset \mathcal{V}$ and $\{\lambda_i\}_{i=0}^N$ are sequences of arbitrary points and positive weights, respectively. Next lemma shows the upper bound for the gap function $f(x)$.

Lemma 4.1.5 (Gap function upper bound, [9, Lemma 1])**.** *Using the quantities defined in* (4.7)*, we have the inequality*

$$f(\tilde{y}_N) \leq \frac{1}{S_N}\Delta_N. \tag{4.8}$$

The proof follows from the definition of the gap function (4.6) and the strong monotonicity of the operator $g(x)$. For brevity, we skip the proof and refer interested readers to [9] for further details.

4.2. PROPOSED FRANK-WOLFE ALGORITHM AND CONVERGENCE ANALYSIS

Building on the key proposition and lemmas from the previous section, this section pertains to the analysis of convergence for using the FW algorithm as an oracle in solving VIP (4.1). The general form of our proposed method is provided in Algorithm 4, which follows the accelerated gradient descent method [9], with the difference that the projection steps are replaced by a strongly convex-concave minimax problem solved by the FW algorithm.

Lemmas 4.1.4 and 4.1.5 shed light on the behavior of the gap function (4.6) and highlight the goal of minimizing and controlling the growth of Δ_N. For $\beta > 0$, consider the functions

$$\psi_y^\beta(x) := \langle g(y), y - x\rangle - \frac{\beta}{2}\|x - y\|^2, \quad \Psi_k(x) := \sum_{i=0}^{k} \lambda_i \psi_{y_i}^\mu(x).$$

We note that $\Delta_k = \max_{x\in\mathcal{V}} \Psi_k(x)$, and the functions $\psi_y^\beta(x)$ and $\Psi_k(x)$ are strongly concave with constants β and μS_k, respectively. Consider the following iterations

$$\begin{cases} x_k = \arg\min_{x_k\in\mathcal{V}} \max_{x\in\mathcal{V}} \Big\{ W(x,x_k) - \frac{\mu}{4} S_k \|x - x_k\|^2 \Big\}, \\ y_{k+1} = \arg\min_{y_{k+1}\in\mathcal{V}} \max_{x\in\mathcal{V}} \Big\{ \langle -g(x_k) - \beta(y_{k+1} - x_k), x - y_{k+1}\rangle - \frac{\mu}{4}\|x - y_{k+1}\|^2 \Big\}, \end{cases} \tag{4.9}$$

where

$$\begin{aligned} W(x,x_k) &:= \langle \nabla\Psi_k(x_k), x - x_k\rangle = \Big\langle \sum_{i=0}^{k} \lambda_i(-g(y_i) - \mu(x_k - y_i)), x - x_k \Big\rangle \\ &= -\mu S_k \langle x_k, x - x_k\rangle + \Big\langle \sum_{i=0}^{k} \lambda_i(-g(y_i) + \mu y_i), x - x_k \Big\rangle. \end{aligned}$$

We are now in a position to derive an upper bound for Δ_k using iterations (4.9), which helps us establish the convergence of the iterates to the solution of the VIP (4.1).

Proposition 4.2.1 (Upper bound of Δ_k)**.** *If $\lambda_{k+1} \le \frac{\mu}{2\beta} S_k$, then by using the iterates (4.9), we have*

$$\Delta_{k+1} \le \Delta_k + \lambda_{k+1}\Big[\frac{1}{\mu + 2\beta}\|g(y_{k+1}) - g(x_k)\|^2 - \frac{\beta}{2}\|y_{k+1} - x_k\|^2 \Big]. \tag{4.10}$$

Proof. We know that $\Psi_{k+1}(x) = \Psi_k(x) + \lambda_{k+1}\psi^\mu_{y_{k+1}}(x)$. Then, we have

$$\begin{aligned} \Delta_{k+1} &= \max_{x\in\mathcal{V}} \Big\{ \Psi_k(x) + \lambda_{k+1}\psi^\mu_{y_{k+1}}(x) \Big\} \\ &\le \Delta_k + \max_{x\in\mathcal{V}} \Big\{ \langle \nabla\Psi_k(x_k), x - x_k\rangle - \frac{\mu}{2} S_k\|x - x_k\|^2 + \lambda_{k+1}\psi^\mu_{y_{k+1}}(x) \Big\} \\ &\le \Delta_k + \max_{x\in\mathcal{V}} \Big\{ W(x,x_k) - \frac{\mu}{4} S_k\|x - x_k\|^2 \Big\} + \max_{x\in\mathcal{V}} \Big\{ -\frac{\mu}{4} S_k\|x - x_k\|^2 + \lambda_{k+1}\psi^\mu_{y_{k+1}}(x) \Big\}. \end{aligned}$$

Note that, if we consider $x^\dagger = \arg\max_{x\in\mathcal{V}} \Psi_k(x) = \mathrm{proj}_{x\in\mathcal{V}}\Big(\frac{\sum_{i=0}^k \lambda_i(\mu y_i - g(y_i))}{\mu S_k} \Big)$, then, from the definition of the projection operator and the first-order optimality condition, we have $W(x,x^\dagger) = \langle \nabla\Psi_k(x^\dagger), x - x^\dagger\rangle \le 0$. Therefore, we can conclude that, as indicated in Proposition 4.1.2, the iterates x_k in (4.9) ensures $\min_{x_k\in\mathcal{V}} \max_{x\in\mathcal{V}} \Big\{ W(x,x_k) - \frac{\mu}{4} S_k\|x - x_k\|^2 \Big\} < 0$. We further note that this strongly convex-concave minimax problem can be solved by FW algorithm with (sub)-linear convergence guarantee. Therefore, by the definition of x_k in (4.9), we arrive at

$$\begin{aligned} \Delta_{k+1} &\le \Delta_k + \overbrace{\min_{x_k\in\mathcal{V}} \max_{x\in\mathcal{V}} \Big\{ W(x,x_k) - \frac{\mu}{4} S_k\|x - x_k\|^2 \Big\}}^{\le 0} + \max_{x\in\mathcal{V}} \Big\{ -\frac{\mu}{4} S_k\|x - x_k\|^2 + \lambda_{k+1}\psi^\mu_{y_{k+1}}(x) \Big\} \\ &\le \Delta_k + \max_{x\in\mathcal{V}} \Big\{ -\frac{\mu}{4} S_k\|x - x_k\|^2 + \lambda_{k+1}\Big[\langle g(y_{k+1}), y_{k+1} - x\rangle - \frac{\mu}{2}\|x - y_{k+1}\|^2 \Big] \Big\}. \end{aligned} \tag{4.11}$$

Now, let us analyze the second term in the "max" part of the right-hand-side of (4.11). Note that

$$\langle g(y_{k+1}), y_{k+1}-x\rangle - \frac{\mu}{2}\|x-y_{k+1}\|^2 = \langle g(y_{k+1})-g(x_k), y_{k+1}-x\rangle - \frac{\mu}{2}\|x-y_{k+1}\|^2 + \langle g(x_k), y_{k+1}-x\rangle,$$

Similar to the previous part, if $y^\dagger$ considered as $y^\dagger = \text{proj}_{x\in\mathcal{V}}\left(x_k - \frac{g(x_k)}{\beta}\right)$, we have

$$\langle -g(x_k) - \beta(y^\dagger - x_k), x - y^\dagger\rangle \le 0, \quad \forall x \in \mathcal{V}.$$

Therefore, y_{k+1} in (4.9) ensures $\min\limits_{y_{k+1}\in\mathcal{V}} \max\limits_{x\in\mathcal{V}}\left\{\langle -g(x_k)-\beta(y_{k+1}-x_k), x-y_{k+1}\rangle - \frac{\mu}{4}\|x-y_{k+1}\|^2\right\} <$ 0, and using the definition of y_{k+1} we can write

$$\begin{aligned}\langle g(y_{k+1}), y_{k+1}-x\rangle &- \frac{\mu}{2}\|x-y_{k+1}\|^2 \\ &\le \|g(y_{k+1})-g(x_k)\|\|y_{k+1}-x\| - \frac{\mu}{4}\|y_{k+1}-x\|^2 + \beta\langle y_{k+1}-x_k, x-y_{k+1}\rangle,\end{aligned}$$

using above inequality and maximizing the right-hand-side based on $\|y_k - x\|$, one obtains

$$\langle g(y_{k+1}), y_{k+1}-x\rangle - \frac{\mu}{2}\|x-y_{k+1}\|^2 \le \frac{1}{\mu+2\beta}\|g(y_{k+1})-g(x_k)\|^2 + \frac{\beta}{2}\|x-x_k\|^2 - \frac{\beta}{2}\|y_{k+1}-x_k\|^2.$$

Putting the above inequality together with (4.11), and using the upper bound on λ_{k+1}, we obtain the inequality (4.10). □

Now by setting $\lambda_{k+1} = \frac{\mu}{2L}S_k = \frac{1}{2\gamma}S_k$ in (4.10), where L and $\gamma = \frac{L}{\mu}$ are the Lipschitz constant and the condition number of the operator g, respectively, we have $\Delta_{k+1} < \Delta_k$. We are now ready to show the convergence rate of Algorithm 4 for solving VIP (4.1).

Algorithm 4 Frank-Wolfe type algorithm for the VIP (FW-VIP)

1: Given $\gamma = \frac{L}{\mu}, \lambda_0 = 1, \varepsilon$.
2: **for** $k = 0, \dots, T$ **do**
3:
$$\left.\begin{array}{l} x_k = \arg\min\limits_{x_k\in\mathcal{V}} \max\limits_{x\in\mathcal{V}}\left\{W(x,x_k) - \frac{\mu}{4}S_k\|x-x_k\|^2\right\} \\ y_{k+1} = \arg\min\limits_{y_{k+1}\in\mathcal{V}} \max\limits_{x\in\mathcal{V}}\left\{\langle -g(x_k) - L(y_{k+1}-x_k), x-y_{k+1}\rangle - \frac{\mu}{4}\|x-y_{k+1}\|^2\right\} \end{array}\right\}$$ FW-minimax oracle 3 with ε-precision
4: $\lambda_{k+1} = \frac{1}{2\gamma}S_k$,
5: **output:** $\tilde{y}_k = \frac{1}{S_k}\sum_{i=0}^{k}\lambda_i y_i$.

Theorem 4.2.2 (VIP-convergence via FW-minimax oracle)**.** *Consider the VIP* (4.1) *under Assumptions 4.1.1, and the FW-minimax oracle 3. Let $\gamma = L/\mu$ be the condition number of the operator g, and f the gap function defined in* (4.6)*. Algorithm 4 returns an ε accurate solution to* (4.1) *if the number of iterations is $T \ge (\gamma+1)\log\left(\gamma^2 f(x_0)/\varepsilon\right)$.*

Before proceeding with the proof of the theorem, we summarize the overall convergence complexity of using Algorithm 4 with the FW-minimax oracle 3 as follows.

Remark 4.2.3 (Overall complexity of VIP solution)**.** *The overall complexity of the solution to the VI problem* (4.1) *is the product of the complexity of the FW-minimax oracle in Algorithm 3 with the complexity of Algorithm 4, under the assumption that the solution of each sub-minimax problem lies inside the feasible domain $\mathcal{V}$. In particular, by using the stepsize* (4.4) *in Algorithm 3, resulting in*

$$\underbrace{\mathcal{O}\left(\log(1/\varepsilon)\right)}_{\text{Theorem 4.2.2}} \cdot \underbrace{\mathcal{O}\left(\log(1/\varepsilon)\right)}_{\text{Proposition 4.1.2}} = \mathcal{O}\left(\log^2(1/\varepsilon)\right).$$

Moreover, by leveraging the diminishing stepsize $\alpha_t = \frac{2}{2+t}$, if the extra assumption of Proposition 4.1.2 is satisfied, the overall complexity of solving the problem becomes

$$\underbrace{\mathcal{O}\left(\log(1/\varepsilon)\right)}_{\text{Theorem 4.2.2}} \cdot \underbrace{\mathcal{O}\left(1/\varepsilon\right)}_{\text{Proposition 4.1.2}} = \mathcal{O}\left(\log(1/\varepsilon)/\varepsilon\right).$$

Proof of Theorem 4.2.2. Using the definition of S_k in (4.7) and the fact that $S_0 = \lambda_0 = 1$, we can write

$$S_{k+1} = S_k + \lambda_{k+1} = \left(1 + \frac{1}{\gamma}\right) S_k.$$

By using Lemma 4.1.5, the corollary of Proposition 4.2.1, and using the sequence $S_{k+1} = \left(1 + \frac{1}{\gamma}\right) S_k$ with $S_0 = 1$, we can conclude that $S_k = \left(1 + \frac{1}{\gamma}\right)^k$. Therefore, we obtain

$$f(\tilde{y}_k) \leq \frac{1}{S_k}\Delta_k \leq \frac{\Delta_0}{\left(1 + \frac{1}{\gamma}\right)^k} = \left(\frac{\gamma}{1+\gamma}\right)^k \Delta_0 = \left(1 - \frac{1}{1+\gamma}\right)^k \Delta_0 \leq \Delta_0 e^{-\frac{k}{\gamma+1}}.$$

Now, we just need to approximate Δ_0. Using the definition of Δ and Ψ, we can write

$$\begin{aligned}
\Delta_0 &= \max_{x \in \mathcal{V}} \Psi_0(x) = \max_{x \in \mathcal{V}} \psi^{\mu}_{y_0}(x) = \max_{x \in \mathcal{V}} \left\{ \langle g(x_0), x_0 - x \rangle - \frac{\mu}{2}\|x - x_0\|^2 \right\} \\
&= \max_{x \in \mathcal{V}} \left\{ \langle g(x_0) - g(x^*), x_0 - x \rangle + \langle g(x^*), x_0 - x \rangle - \frac{\mu}{2}\|x - x_0\|^2 \right\} \\
&\leq \langle g(x^*), x_0 - x \rangle + \max_{x \in \mathcal{V}} \left\{ \langle g(x_0) - g(x^*), x_0 - x \rangle - \frac{\mu}{2}\|x - x_0\|^2 \right\} \leq \langle g(x^*), x_0 - x \rangle + \frac{L^2}{2\mu}\|x^* - x_0\|^2.
\end{aligned}$$

Then, by the definition and strong convexity of the gap function f in (4.6), the last inequality is less than $f(x_0)$. Thus, the theorem's statement is concluded. □

We close this section by noting that the total computational time for solving the VIP (4.1) is the product of the convergence complexity of the considered algorithm and the complexity of the mathematical operations, e.g., projection(s) used at each iteration, which can be computationally demanding in some problems. In the next section, we see this behavior and observe the effectiveness of our proposed projection-free method compared to commonly used state-of-the-art algorithms for strongly monotone VIP.

4.3. NUMERICAL RESULTS

We demonstrate the performance of Algorithm 4 on the traffic assignment problem, an important problem studied in the transportation and operations research literature. To evaluate our performance, we compare our proposed algorithm (FW-VIP) with projected gradient descent (PGD) [40] and Nesterov's accelerated gradient method (NAG) [9], two methods commonly used for solving strongly monotone VIP in the literature. Before proceeding with the simulation results, let us first briefly explain the traffic assignment problem.

Traffic Assignment Problem (TAP): Assume N be the set of nodes, $\mathscr{E}$ be set of directed links, and W denotes the set of origin-destination (OD) pairs. We assume for all $w \in W$ there is $d_w > 0$ representing traffic demand entering from origin and exiting from destination. For each OD pair $w \in W$, the demand d_w is to be distributed among a given P_w of a paths joining w. Defining x_p that shows the flow carried by path P, we introduce the feasible path flow vector $x = \{x_p \,|\, p \in P_w,\, w \in W\}$, and the set corresponding to feasible path flow vector as

$$X = \Big\{x \,|\, \sum_{p \in P_w} x_p = d_w,\, x_p \geq 0,\, \forall p \in P_w,\, w \in W\Big\}.$$

Each collection of path flows $x \in X$ defines a collection of links flows y_{ij} with

$$y_{ij} = \sum_{w \in W} \sum_{p \in P_w} \delta_p(i,j) x_p, \quad (i,j) \in \mathscr{E}, \tag{4.12}$$

where $\delta_p(i,j) = 1$ if path p contains link (i,j) and zero otherwise. The vector of links flows $y = \{y_{ij} \,|\, (i,j) \in \mathscr{E}\}$ can be written as $y = Ax$ where A is the arc-chain matrix defined by (4.12). Similarly, we can define the set of feasible link flows as $Y = AX = \Big\{y \,|\, y = Ax,\, x \in X\Big\}$. For each link $(i,j) \in \mathscr{E}$, there is a given function $T_{ij} : Y \to \mathbb{R}$, which is the measure of the delay in link (i,j). The vector with components $T_{ij}(y)$ is $T(y)$ and due to the choice of T_{ij} it is mostly strongly monotone operator. For each $x \in X$ and corresponding $y = Ax$, the vector $T(y)$ defines the following function for each $w \in W$ and $p \in P_w$

$$\bar{T}_p(x) = \sum_{(i,j) \in \mathscr{E}} \delta_p(i,j) T_{ij}(y),$$

which shows the total travel time of path p. Then, TAP is to find $x^* \in X$ such that for all $\bar{p} \in P_w$ and $w \in W$ we have

$$\bar{T}_{\bar{p}}(x^*) = \min_{p \in P_w} \bar{T}_p(x^*) \quad \text{and} \quad x^*_p > 0. \tag{4.13}$$

As shown in [144], if $\bar{T}(x)$ denotes the vector with $\bar{T}_p(x)$ components, then $\bar{T}(x) = A^\top T(Ax)$ and TAP (4.13) can be reformulated as the following VIP:

$$\text{find } x^* \text{ such that} \quad (x - x^*)^\top \bar{T}(x^*) \geq 0, \quad \forall x \in X. \tag{4.14}$$

Note that $A^\top T(Ax)$ is not necessarily strongly monotone unless $A^\top A$ is inevitable (as is the case in our simulation).

We consider TAP with the same structure in [13] which can be viewed as model of a circular highway. Figure 4.1 illustrates the corresponding model, in which we consider five OD points, numbered 1–5, and five OD pairs: (1,4), (2,5), (3,1), (4,2), and (5,3), associated with the OD points. Each OD pair is linked with two possible paths (clockwise and counterclockwise). The links, types of links, time delays on each link, and the demand for each OD pair are given in Table 4.2, where $h(x) = 1 + x + x^2$ and we set $\kappa = 0.5$. Note that we assume flows cannot utilize the existing ramp that does not lead to its intended destination. Figure 4.2 shows the optimality gap of the corresponding VIP with parameters reported in Table 4.2. We note that, for a fair comparison and to demonstrate the effectiveness of the projection-free algorithm, we plot the optimality gap versus the total iterations multiplied by the total number of projections (in PGD or NAG) or LMO (in Algorithm 4). We also note that both projection and LMO operators are evaluated using the same solver, namely `OSQP` [145].

Table 4.2: Parameters in TAP.

Types of links	
(1) Highway links:	17, 27, 37, 47, 57, 18, 28, 38, 48, 58.
(2) Exit ramps:	14, 24, 34, 44, 54, 12, 22, 32, 42, 52.
(3) Entrance ramps:	11, 21, 31, 41, 51, 13, 23, 33, 43, 53.
(4) Bypass links:	15, 25, 35, 45, 55, 16, 26, 36, 46, 56.
Delay on links	
(1) Delay on highway link k:	$10 \cdot h\left[\text{flow on } k\right] + 2\kappa \cdot h\left[\text{flow on exit ramp from } k\right]$.
(2) Delay on exit ramp k:	$h\left[\text{flow on } k\right]$.
(3) Delay on entrance ramp k:	$h\left[\text{flow on } k\right] + \kappa \cdot h\left[\text{flow on bypass link merging with } k\right]$.
(4) Delay on bypass link k:	$h\left[\text{flow on } k\right]$.
Demands on OD pairs	$d(1,4) = 0.1$, $d(2,5) = 0.2$, $d(3,1) = 0.3$, $d(4,2) = 0.4$, $d(5,3) = 0.5$.

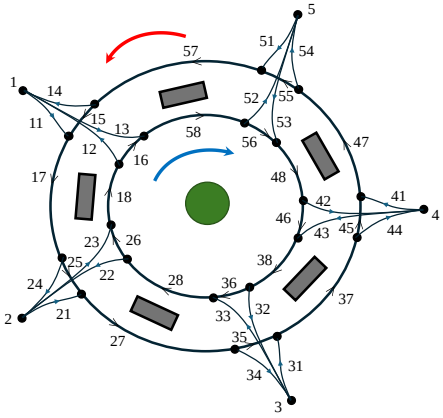


Figure 4.1: TAP model (Figure 2 in [13]).

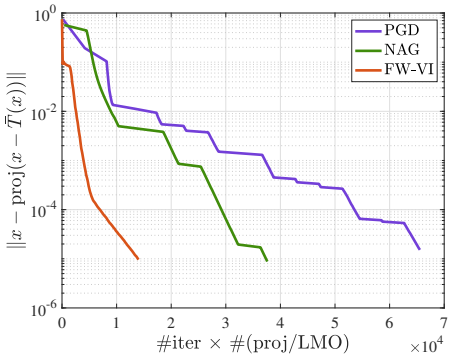


Figure 4.2: Optimality gap of the VIP in the TAP.

4.4. FURTHER DISCUSSION, LIMITATIONS, AND FUTURE DIRECTIONS

In this section, we provide additional insights concerning the proposed Frank-Wolfe technique for solving strongly monotone variational inequalities (1), its limitations, and potential future directions.

(i) **Further insights into the Frank-Wolfe technique for solving VIP (4.1):** The main motivation behind the Frank-Wolfe technique is to avoid expensive projections in each iteration and to select a more suitable descent direction compared to projected methods, which may exhibit zig-zagging behavior for certain constraint sets. This can lead to lower computational complexity and potentially faster convergence. Furthermore, as stated in Proposition 4.2.1, we only require the negativity of

$$N(x,y) := \min_{x_k \in \mathcal{V}} \max_{x \in \mathcal{V}} \left\{ W(x,x_k) - \tfrac{\mu}{4} S_k \|x - x_k\|^2 \right\},$$

to guarantee the convergence of Algorithm 4. While this negativity is ensured by the worst-case convergence bound in Proposition 4.1.2, in practice the inner Frank-Wolfe problem can be terminated as soon as $N(x,y)$ becomes negative, significantly reducing computational complexity.

(ii) **Extension to monotone variational inequalities:** Solving the VIP using a projection-free algorithm requires solving a minimax subproblem. Unfortunately, existing projection-free methods for minimax problems rely on strong convexity-concavity of the underlying function, which in our setting can be ensured by assuming strong monotonicity of the operator in the VIP. Developing a Frank-Wolfe method for minimax problems without this assumption is a key step toward extending these methods to general monotone variational inequality problems and is an important direction for future work.

(iii) **Relation to prior works and alternative assumptions:** Based on the structure of the VIP, alternative assumptions can be applied for the Frank-Wolfe algorithm in solving the inner minimax problems, replacing the requirement in Proposition 4.1.2 that the solution lies strictly within the domain:

- The sets $\mathcal{X}$ and $\mathcal{Y}$ are polytopes (case (II) in Theorem 1 [10]): In this case, the interior-point assumption is not required. Instead, we can use the away-step Frank-Wolfe algorithm (Algorithm 3 in [10]) and the concept of pyramidal width (Eq. 9 in [140]) instead of the border distance $\sigma_{\mathcal{X}}$, while still guaranteeing the same convergence as in Proposition 4.1.2.
- More recently, [142] proposed a projection-free algorithm (Algorithm 3) under a similar assumption to Proposition 4.1.2, without requiring the solution to lie in the interior of $\mathcal{X} \times \mathcal{Y}$. This algorithm requires $\tilde{\mathcal{O}}(1/\sqrt{\varepsilon})$ first-order and $\tilde{\mathcal{O}}(1/\varepsilon^2)$ linear optimization oracle calls to achieve ε-precision for the saddle-point problem (4.2). Its main limitation is implementation complexity, as it involves three nested loops per iteration.

(iv) **Limitations and future directions:**

- The primary limitation of the proposed method is the presence of a secondary loop in each iteration for solving the minimax problem. In the context of the Frank-Wolfe algorithm for minimizing a smooth convex function, two key inequalities, smoothness and convexity, are typically used. For the VIP, no explicit value function exists, making these inequalities difficult to apply. Although we can define a convex gap function $f(x)$ (4.6), whose minimization is equivalent to solving the VIP, this function is parametric, complicating the use of standard convex optimization inequalities. Restricting the domain to specific sets and leveraging duality may allow for a single-loop algorithm, which is an important future research direction. Adaptive or decreasing accuracy requirements for the inner loop may also help eliminate the secondary loop.
- Another limitation is the algorithm's dependence on problem-specific constants. Future extensions could involve adaptive stepsize methods [46], which approximate Lipschitz or strong monotonicity constants to accelerate convergence.
- Recent works, such as [146], establish Newton-type methods combined with Frank-Wolfe techniques for constrained self-concordant minimization, and [147] propose high-order methods for solving the VIP. Developing high-order methods for the VIP using Frank-Wolfe techniques represents another promising research direction.

5

VARIATIONAL INEQUALITIES IN LINEAR–QUADRATIC DYNAMIC GAMES

Following the analysis of variational inequalities, in this chapter we formulate linear quadratic dynamic games as an affine variational inequality, propose a splitting method for the underlying problem, and validate the proposed iterative approach through a numerical example involving autonomous vehicles at an intersection.

Dynamic games provide a formal framework for analyzing multi-agent decision-making processes in dynamical systems, where the agents' objectives and constraints are coupled through both the system dynamics and their performance criteria [148], [149]. In such settings, each agent aims to optimize its own cost function while anticipating the actions of other agents, leading to strategic interactions that can be captured via various equilibrium concepts. Dynamic games are a useful model in robust control [150], as well as robotics [151], [152], electricity market control [153] and supply chain strategic organization [154]. These frameworks enable decentralized decision-making and the analysis of emergent behaviors in complex, multi-agent environments.

The desired solution concepts in dynamic games depend on the information structure assumed for the agents. One common approach is the open-loop Nash equilibrium (OL-NE), which defines a trajectory as a sequence of control inputs that is optimal for each agent, given the initial state and the input sequences chosen by the other agents [155], [156]. The OL-NE problem can be cast as Variational Inequality (VI) [157], which can be solved by a plethora of algorithms [4]. This is especially relevant given the recent emergence of game-theoretic receding-horizon controllers [158], [159], which define the control action at every time-step according to the solution to an OL-NE problem, thus requiring to complete its computation within the sampling time of the controller. This is especially difficult, as the complexity of the derived VI increases rapidly with the number of agents, the control horizon and the number of constraints.

5

In this work, we propose to leverage the affine structure of the VI resulting from a linear-quadratic game in order to define a specifically tailored solution algorithm based on the Douglas-Rachford iteration [160]. Let us justify our choice by illustrating the performance of some state-of-the-art VI solution algorithms, along with a simulation that highlights the contribution of this paper. Let us first define the VI problem as follows:

$$\text{find } u^* \in \mathscr{C} \quad \text{s.t. } \inf_{u \in \mathscr{C}} \langle F(u^*), u - u^* \rangle \geq 0, \tag{5.1}$$

where $\mathscr{C} \subseteq \mathbb{R}^n$ is a closed, convex set, and $F : \mathbb{R}^n \rightrightarrows \mathbb{R}^n$ is a (in general set-valued) operator. We assume that F is monotone [4], Lipschitz, and the solution set of (5.1) is nonempty. Note that via the first-order optimality condition, we can rewrite (5.1) as a monotone inclusion, $0 \in F(u) + \mathscr{N}_{\mathscr{C}}(u)$, where $\mathscr{N}_{\mathscr{C}}$ is the normal cone of the set $\mathscr{C}$. This equivalence is beneficial and allows us to use operator splitting methods.

As shown in Section 5.1, under some technical assumptions, a solution to a constrained linear-quadratic game can be found by solving a strongly monotone affine variational inequality problem with linear constraints, i.e. $F = Mu + q$ and $\mathscr{C} : Du \leq d$ (where M can be a non-symmetric matrix). Therefore, efficient numerical methods for solving affine VIs are essential in dynamic game equilibrium problems.

Illustrative example. Consider the VI in (5.1) with $\mathscr{C} = \mathbb{R}^n$ and $F = Mu$, where $M = I + S$, and $S = \begin{pmatrix} 0 & -\omega \\ \omega & 0 \end{pmatrix}$. The classical Forward-Backward (FB) splitting method generates the sequence

$$u^{k+1} = (I - \gamma(I + S))u^k,$$

whose convergence depends on the spectral radius of $T_{\text{FB}} = I - \gamma(I + S)$, given by $\rho_{\text{FB}} = 1 - \gamma(1 \pm i\omega)$. In contrast, the spectral radius of the Douglas–Rachford (DR) splitting method,

which we will discuss in greater detail in Section 5.2, is given by

$$\rho(T_{\mathrm{DR}}) = \tfrac{1}{2}\left(1 + \frac{1-\gamma j\omega}{1+\gamma j\omega}\cdot\frac{1-\gamma}{1+\gamma}\right).$$

For non-small stepsizes (e.g., $\gamma = 0.5$), the FB iteration exhibits non-decaying oscillations with $|\rho_{\mathrm{FB}}| = 1$, whereas DR contracts steadily toward equilibrium with $|\rho_{\mathrm{DR}}| \approx 0.52$. As we will see in Section 5.2, the DR takes advantage of the affine structure of the problem to split the operator that defines the VI in a symmetric and non-symmetric part. This enables the DR to effectively damp rotational modes, resulting in faster and more stable convergence for skew-dominated problems.

Motivated by exploiting the structure of linear VIs that arise from linear-quadratic dynamic games and the DR splitting to achieve faster convergence, our technical contributions are summarized as follows:

- First, we define a strongly monotone affine VI whose solution, under some technical assumptions, is an infinite-horizon open-loop Nash equilibrium for the linear-quadratic dynamic game. We show that the solution to the problem is known in closed form when the initial state lies in a neighborhood of the origin.
- We leverage the linearity of the VI to derive a tailored iterative solver based on the DR splitting method. Furthermore, we show linear convergence for the derived affine VI of OL-NE.
- We adopt the aforementioned splitting method in computing the control input of a controller based on the receding-horizon solution to a VI derived from a linear-quadratic dynamic game. The test scenario is that of a set of autonomous vehicles, whose objective is to traverse a crossroad while maintaining a safe distance and speed.

Notation.

a) *Matrices:* We denote as I_n the identity matrix of size n and as $0_{n,m}$ the zero matrix of size $n \times m$. The subscripts are omitted when clear from context. For a set of matrices of suitable dimensions indexed in $\mathscr{I}$, $(M_i)_{i\in\mathscr{I}}$, we denote as $\mathrm{col}(M_i)_{i\in\mathscr{I}}, \mathrm{row}(M_i)_{i\in\mathscr{I}}, \mathrm{blkdg}(M_i)_{i\in\mathscr{I}}$ its column, row and diagonal stack, respectively. For a set of matrices of suitable dimensions indexed in $\mathscr{I}\times\mathscr{J}$, we denote the block matrix with i,j-th element M_{ij} as $\mathrm{blkmat}(M_{ij})_{i\in\mathscr{I},j\in\mathscr{J}}$ We denote $M^{-\top} = (M^\top)^{-1}$ and the Kronecker product as $\otimes$.

b) *Euclidean spaces and operators:* Let $\mathscr{C}$ be the finite-dimensional real vector space with the standard inner product $\langle\cdot,\cdot\rangle$ Euclidean norm. For $x \in \mathbb{R}^n$ and $Q \succ 0$, we denote $\|x\|_Q = \sqrt{x^\top Q x}$. The subscript is omitted when $Q = I$. We denote as $\pi_{\mathscr{C}}$ the metric projection onto set $\mathscr{C}$ ($\pi_{\mathscr{C}}(u) = \arg\min_{y\in\mathscr{C}} \|u-y\|$). We define the normal cone of the set $\mathscr{C}$ at the point $u \in \mathscr{C}$ as $\mathscr{N}_{\mathscr{C}}(u) = \{g : g^\top u \geq g^\top y,\ \forall y \in \mathscr{C}\}$. An operator F is L-Lipschitz, if there is $L > 0$ such that for all $u, y \in \mathscr{C}$ we have $\|F(u) - F(y)\| \leq L\|u-y\|$. The operator F is (strongly) monotone if $\langle F(u) - F(y), u-y\rangle \geq \mu\|u-y\|^2$ for all $u, y \in \mathscr{C}$ and some $\mu \geq 0 (> 0)$. For a closed set $\mathscr{C}$, we denote its interior as $\mathrm{int}(\mathscr{C})$. We denote the VI problem 5.1 with

the operator F and set $\mathscr{C}$ as VI($\mathscr{C}, F$). For an affine operator $F : x \mapsto Mx + u$, we denote the affine VI problem AVI($\mathscr{C}, M, q$) := VI($\mathscr{C}, F$).

c) *Real sequences:* We denote as $\mathscr{S}_T^n$ a sequence in $\mathbb{R}^n$ with length $T \in \mathbb{N} \cup \infty$. Let $w \in \mathscr{S}_T^n$. We denote as $w[t]$, with $t \in \{0, ..., T-1\}$, its t-th element. If $w : \mathbb{R}^m \to \mathscr{S}_T^n$, that is, $w(x)$ is a sequence for all $x \in \mathbb{R}^m$, then we denote as $w[t|x]$ its t-th element. We treat finite sequences as column vectors, that is, if $w \in \mathscr{S}_T^n$, then $w = \mathrm{col}(w[t])_{t \in \{0,...,T-1\}}$.

d) *Game theory:* We consider multi-agent systems, and we denote as N the number of agents. We denote as $\mathscr{I} := \{1, ..., N\}$ and as $\mathscr{I}_{-i} := \mathscr{I} \setminus \{i\}$. We denote in boldface the column stack over $\mathscr{I}$, e.g. $\boldsymbol{w} = \mathrm{col}(w_i)_{i \in \mathscr{I}}$. We add the subscript $-i$ to denote the stack over $\mathscr{I}_{-i}$, e.g. $\boldsymbol{w} = \mathrm{col}(w_i)_{i \in \mathscr{I}_{-i}}$. If $w_i \in \mathscr{S}_\infty^N$ for all $i \in \mathscr{I}$ then, with slight abuse of notation, we denote $\boldsymbol{w} := (w_i)_{i \in \mathscr{I}}$, $\boldsymbol{w}_{-i} := (w_i)_{i \in \mathscr{I}_{-i}}$.

5.1. OPEN-LOOP NASH EQUILIBRIA AS SOLUTIONS OF A VARIATIONAL INEQUALITY

5

We consider a linear system with N inputs, each controlled by a self-interest agent. We denote the sequence of control inputs of agent i as $u_i \in \mathscr{S}_\infty^m$. The system dynamics is ruled by the difference equation

$$x[t+1] = Ax[t] + \sum_{i \in \mathscr{I}} u_i[t]. \tag{5.2}$$

We denote as $\phi(t, x_0, \boldsymbol{u})$ the solution at time t of (5.2) with initial state x_0 and input $\boldsymbol{u}$. We consider quadratic stage costs for all agents:

$$\ell_i(x[t], u_i[t]) := \tfrac{1}{2}\|x[t]\|_{Q_i}^2 + \tfrac{1}{2}\|u_i[t]\|_{R_i}^2 \qquad \forall\, t \tag{5.3}$$

such that, for all $i \in \mathscr{I}$, $Q_i = C_i^\top C_i \succcurlyeq 0$, $R_i = R_i^\top \succ 0$ and the pairs (A, B_i) and (A, C_i) are stabilizable and detectable, respectively. Furthermore, we consider linear time-invariant constraints: define the stage-feasible sets as

$$\mathbb{U}_i(\boldsymbol{u}_{-i}[t]) := \{u_i[t] : \textstyle\sum_{j \in \mathscr{I}} D_j^{\mathrm{u}} u_j[t] + d^{\mathrm{u}} \leq 0\} \tag{5.4a}$$

$$\mathbb{X} := \{x[t] : D^{\mathrm{x}} x[t] + d^{\mathrm{x}} \leq 0\}, \tag{5.4b}$$

and the collective feasible input sequences with length $T \in \mathbb{N} \cup \{\infty\}$ as

$$\mathscr{U}_T(x_0) := \{\boldsymbol{u} :\ u_i[t] \in \mathbb{U}_i(\boldsymbol{u}_{-i}[t]) \forall\, i \in \mathscr{I}, t < T; \phi(t, x_0, \boldsymbol{u}) \in \mathbb{X} \forall\, t < T\}. \tag{5.5}$$

We assume that the origin is strictly feasible, that is,

Assumption 5.1.1. $0 \in \mathrm{int}(\mathbb{X}); 0 \in \mathrm{int}(\mathbb{U}_i(0))$, *for all* $i \in \mathscr{I}$.

Note that, if T is finite, $\mathscr{U}_T(x_0)$ can be expressed as a system of affine inequalities by substituting (5.2) in (5.5) to eliminate ϕ:

$$\boldsymbol{u} \in \mathscr{U}_T(x_0) \iff D\boldsymbol{u} + d \leq 0. \tag{5.6}$$

We define the (infinite-horizon) Open-Loop Nash equilibrium (OL-NE) $\boldsymbol{u}^{\mathrm{OL}}$ as an input sequence such that the infinite-horizon objective

$$J_i^{\infty}(u_i, \boldsymbol{u}_{-i}, x_0) := \sum_{t=0}^{\infty} \ell_i\big(\phi(t, x_0, \boldsymbol{u}), u_i[t]\big) \tag{5.7}$$

cannot be improved by unilateral modifications by any agent.

Definition 5.1.2 (OL-NE)**.** *$\boldsymbol{u}^{\mathrm{OL}} \in \mathscr{U}_{\infty}(x_0)$ is an OL-NE for the initial state x_0 if $\lim_{t\to\infty} \phi(t, x_0, \boldsymbol{u}^{\mathrm{OL}}) = 0$ and $J_i^{\infty}(u_i^{\mathrm{OL}}, \boldsymbol{u}_{-i}^{\mathrm{OL}}, x_0) \le J_i^{\infty}(u_i, \boldsymbol{u}_{-i}^{\mathrm{OL}}, x_0)$ for all u_i such that $(u_i, \boldsymbol{u}_{-i}^{\mathrm{OL}}) \in \mathscr{U}_{\infty}(x_0)$.*

The infinite-horizon problem is in general intractable. In the remainder of this section, we summarize the results of our previous work [157], where we identify an affine, finite-dimensional variational inequality (VI), whose solution are a truncation of the OL-NE under the following technical assumptions:

Assumption 5.1.3 ([155, Assumption 4.9])**.** *A is invertible and the matrix*

$$H = \begin{bmatrix} A + \sum_{j\in\mathscr{I}} S_j A^{-\top} Q_j & \mathrm{row}(-S_j A^{-\top})_{j\in\mathscr{I}} \\ \mathrm{col}(-A^{-\top} Q_j)_{j\in\mathscr{I}} & I_N \otimes A^{-\top} \end{bmatrix},$$

where $S_i := B_i R_i^{-1} B_i^{\top}$, has exactly n eigenvalues with modulus strictly less than 1. An n-dimensional invariant subspace of H is complementary to

$$\mathrm{Im}\left(\begin{bmatrix} 0_{n\times Nn} \\ I_{Nn} \end{bmatrix}\right).$$

We define $(P_i, K_i)_{i\in\mathscr{I}}$ as the solution to the coupled algebraic Riccati equations (ARE)

$$P_i = Q_i + A^{\top} P_i (A + \textstyle\sum_{j\in\mathscr{I}} B_j K_j) \tag{5.8}$$

$$K_i = -R_i^{-1} B_i^{\top} P_i (A + \textstyle\sum_{j\in\mathscr{I}} B_j K_j); \tag{5.9}$$

such that the feedback gains $(K_i)_{i\in\mathscr{I}}$ stabilize the system in (5.2). Such solution exists following [155, Theorem 4.10]. Furthermore, we define $(\hat{P}_i, \hat{K}_i)_{i\in\mathscr{I}}$ as the solutions to the (uncoupled) AREs

$$\hat{P}_i = \hat{Q}_i + \hat{A}_i^{\top} \hat{P}_i (\hat{A}_i + \hat{B}_i \hat{K}_i) \tag{5.10a}$$

$$\hat{K}_i = -R_i^{-1} \hat{B}_i^{\top} \hat{P}_i (\hat{A}_i + \hat{B}_i \hat{K}_i), \tag{5.10b}$$

where, for each i

$$\begin{aligned} \hat{A}_i &:= \begin{bmatrix} A & \sum_{j\neq i} B_j K_j \\ 0 & A + \sum_{j\in\mathscr{I}} B_j K_j \end{bmatrix}, \\ \hat{Q}_i &:= \mathrm{blkdg}(Q_i, 0_{n\times n}), \\ \hat{B}_i &:= \mathrm{col}(B_i, 0_{n\times m}), \end{aligned} \tag{5.11}$$

such that the feedback gains $\hat{K}_i$ stabilize the augmented system $(\hat{A}_i, \hat{B}_i)$ for each i. Finally, let us introduce $\mathbb{X}_f$ as a forward-invariant, constraint-admissible set for the dynamics

$$x[t+1] = (A + \textstyle\sum_{j\in\mathscr{I}} B_j K_j) x[t].$$

Then, we can find a T-long truncation of an OL-NE sequence as the solution to the finite-horizon equilibrium problem

$$\begin{aligned} &\text{find } \boldsymbol{u}^* \in \mathscr{U}(x_0) \text{ such that} \\ &\quad \forall i : u_i^* \in \arg\min_{u_i} \; J_i(u_i, \boldsymbol{u}^*, x_0) \\ &\qquad\qquad \text{s.t. } (u_i, \boldsymbol{u}^*_{-i}) \in \mathscr{U}_T(x_0), \end{aligned} \tag{5.12}$$

where

$$J_i(u_i, \boldsymbol{u}^*, x_0) := \textstyle\sum_{t=0}^{T-1} \left\{ \ell(\phi(t, x_0, u_i, \boldsymbol{u}^*_{-i}), u_i[t]) \right\} + \frac{1}{2} \left\| \begin{bmatrix} \phi(T, x_0, u_i, \boldsymbol{u}^*_{-i}) \\ \phi(T, x_0, u_i^*, \boldsymbol{u}^*_{-i}) \end{bmatrix} \right\|^2_{\hat{P}_i} \tag{5.13}$$

as formalized next:

Lemma 5.1.4 ([157, Theorem 1]). *Let $\boldsymbol{u}^*$ solve (5.12). Let $x_T := \phi(T, x_0, \boldsymbol{u}^*)$ and let $x_T \in \mathbb{X}_f$. Then, the sequence defined as*

$$\forall i : \begin{cases} u_i^*[t] & \text{if } t < T, \\ K_i (A + \sum_j B_j K_j)^{t-T} x_T & \text{if } t \geq T \end{cases} \tag{5.14}$$

is an OL-NE for the initial state x_0.

We now develop a connection between the solutions to the problem in (5.12) and the solutions to a VI. Via straightforward calculation, we can find matrices $\Theta, (\Gamma_i)_{i \in \mathscr{I}}$ such that the dynamics in (5.2) is rewritten as

$$\begin{bmatrix} x[1] \\ \dots \\ x[T] \end{bmatrix} = \Theta x_0 + \sum_{i \in \mathscr{I}} \Gamma_i \begin{bmatrix} u_i[0] \\ \dots \\ u_i[T-1] \end{bmatrix}. \tag{5.15}$$

Then, the partial gradients of the functions in (5.13) with respect to the first argument read as

$$\begin{bmatrix} \nabla_1 J_i(u_1, \boldsymbol{u}, x_0) \\ \dots \\ \nabla_1 J_i(u_N, \boldsymbol{u}, x_0) \end{bmatrix} = M \boldsymbol{u} + q, \tag{5.16}$$

where

$$\begin{aligned} M &:= \text{blkdg}(\bar{R}_i)_{i \in \mathscr{I}} + \text{blkmat}(\Gamma_i^\top \bar{Q}_i \Gamma_j)_{(i,j) \in \mathscr{I}^2}, \\ q &:= \text{col}(\Gamma_i^\top \bar{Q}_i \Theta x_0)_{i \in \mathscr{I}}, \\ \bar{R}_i &:= I_T \otimes R_i, \qquad \forall i \\ \bar{Q}_i &:= \text{blkdg}(I_{T-1} \otimes Q_i, P_i), \qquad \forall i. \end{aligned} \tag{5.17}$$

A sufficient condition for $\boldsymbol{u}^*$ to solve (5.12) is that $\boldsymbol{u}^*$ solves

$$\text{AVI}(\mathscr{U}_T(x_0), M, q). \tag{5.18}$$

Lemma 5.1.5 ([157, Proposition 2]). *If $\boldsymbol{u}^*$ is a solution to the VI in (5.18), then it is also a solution to the finite-horizon equilibrium problem in (5.12).*

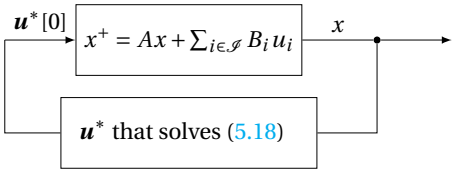


Figure 5.1: Block scheme of the closed-loop dynamics with receding-horizon open-loop Nash equilibrium controller.

In Section 5.2, we leverage the problem reformulation provided by Lemma 5.1.5 to propose a fast solution algorithm to (5.18). To further alleviate the computational burden, we show that the solution to (5.18) can be computed in closed-form when the initial state belongs to $\mathbb{X}_f$.

Proposition 5.1.6. *Let $x_0 \in \mathbb{X}_f$ and M in (5.17) be such that $M + M^\top > 0$. Let for all i*

$$u_i^*[t] = K_i(A + \textstyle\sum_{i\in\mathscr{I}} B_i K_i)^t x_0.$$

Then, the unique solution to the VI in (5.18) is the sequence $\boldsymbol{u}^*[0], ..., \boldsymbol{u}^*[T-1]$.

5

From the definition of $\mathbb{X}_f$, $\boldsymbol{u}^*$ is strictly feasible. We proceed by contradiction, and we assume there exists an input sequence v_i such that

$$J_i(v_i, \boldsymbol{u}^*, x_0) < J_i(u_i^*, \boldsymbol{u}^*, x_0). \tag{5.19}$$

Define the sequences

$$x^v[t] = \phi(t, x_0, v_i, \boldsymbol{u}^*_{-i}) \tag{5.20}$$
$$x^*[t] = (A + \textstyle\sum_{i\in\mathscr{I}} B_i K_i)^t x_0 \tag{5.21}$$
$$y[t] = \left(\hat{A}_i + \hat{B}_i \hat{K}_i\right)^t \begin{bmatrix} x^v[T] \\ x^*[T] \end{bmatrix} \tag{5.22}$$
$$w_i[t] = \hat{K}_i y[t]. \tag{5.23}$$

From the definition of $\hat{A}_i, \hat{B}_i$, we deduce the structure

$$y[t] = \begin{bmatrix} z[t] \\ x^*[t] \end{bmatrix}$$

where z satisfies

$$z[t+1] = Az[t] + B_i \hat{K}_i y[t] + \textstyle\sum_{j\neq i} B_j K_j x^*[t] \tag{5.24}$$

Note that $K_j x^*[t] = \boldsymbol{u}^*[t]$, $\hat{K}_i y[t] = w_i[t]$ and $z[0] = x^v[T]$, thus we can write

$$z[t] = \phi(t, x^v[T], w_i, \boldsymbol{u}^*_{-i}). \tag{5.25}$$

Following [161, Theorem 21.1] and the fact that $\hat{P}_i, \hat{K}_i$ solve the ARE in (5.10),

$$\begin{aligned}\frac{1}{2}\left\|\begin{bmatrix}x^v[T]\\ x^*[T]\end{bmatrix}\right\|^2_{\hat{P}_i} &= \sum_{t=0}^{\infty}\tfrac{1}{2}\|y[t]\|^2_{\hat{Q}_i}+\tfrac{1}{2}\|w_i[t]\|^2_{\hat{R}_i}\\ &= \sum_{t=0}^{\infty}\tfrac{1}{2}\|z[t]\|^2_{Q_i}+\tfrac{1}{2}\|w_i[t]\|^2_{R_i}\\ &= \sum_{t=0}^{\infty}\ell_i(z[t],w_i[t]),\end{aligned} \tag{5.26}$$

where we used the definition of $\hat{Q}_i, \hat{R}_i$. Denote for all i

$$v_i^{\infty}[t] := \begin{cases} v_i[t] & \text{if } t < T\\ w_i[t-T] & \text{if } t \geq T.\end{cases} \tag{5.27}$$

After substituting (5.26) in (5.12),

$$\begin{aligned}J_i(v_i,\boldsymbol{u}^*,x_0) = &\sum_{t=0}^{T-1}\ell(\phi(t,x_0,v_i,\boldsymbol{u}^*_{-i}),v_i[t])\\ &+\sum_{\tau=0}^{\infty}\ell(z[t],w_i[t]).\end{aligned} \tag{5.28}$$

We substitute (5.25) and (5.27)

$$\begin{aligned}J_i(v_i,\boldsymbol{u}^*,x_0) &= \sum_{t=0}^{T-1}\ell(\phi(t,x_0,v_i,\boldsymbol{u}^*_{-i}),v_i[t]) && (5.29)\\ &+\sum_{\tau=0}^{\infty}\ell(\phi(t,x^v[T],w_i,\boldsymbol{u}^*_{-i}),w_i[t]) && (5.30)\\ &= \sum_{t=0}^{\infty}\ell(\phi(t,x_0,v_i^{\infty},\boldsymbol{u}^*_{-i}),v_i^{\infty}[t]) && (5.31)\\ &= J_i^{\infty}(v_i^{\infty},\boldsymbol{u}^*_{-i},x_0), && (5.32)\end{aligned}$$

Similarly, one can show

$$J_i(u_i^*,\boldsymbol{u}^*,x_0) = J_i^{\infty}(u_i^*,\boldsymbol{u}^*_{-i},x_0). \tag{5.33}$$

Thus, (5.19) implies

$$J_i^{\infty}(v_i^{\infty},\boldsymbol{u}^*_{-i},x_0) < J_i^{\infty}(u_i^*,\boldsymbol{u}^*_{-i},x_0),$$

which contradicts the fact that $\boldsymbol{u}^*$ is an infinite-horizon OL-NE for the unconstrained system [157, Prop. 1]. Therefore, $\boldsymbol{u}^*$ solves (5.12). Because $\boldsymbol{u}^*$ is a strictly feasible solution, the optimality conditions of (5.12) lead to

$$\nabla_1 J_i(u_i^*,\boldsymbol{u}^*,x_0) = 0, \quad \forall\ i \tag{5.34}$$

From (5.16), we obtain

$$M\boldsymbol{u}^* + q = 0. \tag{5.35}$$

From the definition of VI in (5.1), we conclude that $\boldsymbol{u}^*$ is a solution to the problem in (5.18). The statement follows then by the uniqueness of the solution [4, Prop. 2.3.2, Th. 2.3.3]. □

With the receding-horizon control application illustrated in Figure 5.1, in the remainder of this paper, we explore a Douglas-Rachford (DR) splitting algorithm that exploits the linear structure of the VI.

5.2. ALGORITHM AND CONVERGENCE

In this section, we first present a Douglas-Rachford splitting-like method for solving an affine variational inequality. Then, based on the results from the previous section, we tailor the proposed algorithm to the affine VI that defines $\mathscr{P}_2(x_0)$ in (5.18). To this end, let us consider a general affine VI problem AVI($\mathscr{C}, M, q$) and denote with M_1, M_2, H matrices such that $M = M_1 + M_2$, and

$$\begin{aligned} M_1 = M_1^\top \succcurlyeq 0,\ M_2 \succ 0, \\ H = H^\top \succ 0. \end{aligned} \tag{5.36}$$

Consider the following Douglas-Rachford splitting-like method [160, Eqq. 30, 31] to solve AVI($\mathscr{C}, M, q$), $k \in \mathbb{N}$:

$$y^k = \text{sol}(\mathscr{C}, H + M_1, q + (M_2 - H)u^k), \tag{5.37a}$$

$$u^{k+1} = (H + M_2)^{-1}\left(H(2\lambda_k y^k + (1 - 2\lambda_k)u^k) + M_2 u^k\right), \tag{5.37b}$$

where sol($\mathscr{C}, M, q$) denotes the solution to AVI($\mathscr{C}, M, q$). At first glance, the iteration in (5.37) is not effective, as the step in (5.37a) requires itself the solution of an affine VI. However, note that $H + M_1$ is a symmetric operator, differently from M. Therefore, the step in (5.37a) is equivalent to solving an optimization problem [4, §1.3.1], and specifically a quadratic program (QP) in the case when $\mathscr{C}$ is a polyhedron, which can be solved very efficiently [145]. The iteration method in (5.37) is well-defined under the condition in (5.36), as the AVI in (5.37a) admits a unique solution, and $H + M_2$ is invertible. Linear convergence of the iteration in (5.37) to the solution of VI($\mathscr{C}, M, q$) is provided in the following lemma.

Lemma 5.2.1 (Linear convergence [160])**.** *The sequence $(u_k)_{k\in\mathbb{N}}$ generated by the iteration in (5.37) converges linearly [133, Eq. 5.8] to the solution of* AVI($\mathscr{C}, M, q$) *where $\lambda \in (0, 1]$, $M = M_1 + M_2$ and M_1, M_2, and H are matrices chosen according to (5.36).*

*P*roof. The result follows directly from Case 2 in [160, Proposition 6] and [160, Proposition 8]. □

We also refer interested readers to [134], [135], where the authors provide different convergence guarantees for different kinds of splitting in inclusion problems, which can be extended to solving VI.
We choose H, M_1, M_2 as follows:

$$H = (1 - \gamma)(M + M^\top) + \varepsilon I, \tag{5.38a}$$

$$M_1 = \gamma(M + M^\top), \tag{5.38b}$$

$$M_2 = (M - M^\top) + (1 - \gamma)(M + M^\top), \tag{5.38c}$$

with $\gamma \in (0, 1)$, and we recall that M is defined in (5.17). This splitting choice satisfies the conditions in (5.36) when $M \succ 0$ and it thus results in a linearly convergent Douglas-Rachford iteration according to Lemma 5.2.1. It is designed so that, in the particular case $M = M^\top$ and with the choice $\lambda = \frac{1}{2}$, (5.37) becomes

$$u^{k+1} = \text{sol}(\mathscr{C}, \varepsilon I + M, q - \varepsilon u^k), \tag{5.39}$$

which is equivalent to

$$u^{k+1} = \arg\min_{u \in \mathscr{C}} \frac{1}{2} \| u + (M+\varepsilon I)^{-1}(q - \varepsilon u^k) \|^2_{M+\varepsilon I}. \tag{5.40}$$

The latter, for $\varepsilon \approx 0$, resembles a full-step towards the unconstrained solution of (5.1), namely, $-M^{-1}q$, projected onto the constraint set.

5.3. NUMERICAL EXPERIMENTS

We test the proposed Douglas-Rachford algorithm with $H = I$, $\lambda_k = 0.5\ \forall k$, and M_1, M_2 chosen as in (5.38) on the receding-horizon control architecture in Figure 5.1, applied to the control of autonomous vehicles driving through a crossroad. The vehicles traverse the intersection in the sequence of their arrival. Each vehicle aims to maintain a safe distance from the preceding vehicle intersecting their path, while matching its velocity. If no preceding vehicle intersects their path, the agent is called a *leading vehicle*, and its objective is to maintain a reference speed. We denote the set of leading vehicles as $\mathscr{L}$ and the preceding vehicle intersecting the path of i as $\chi(i)$. We simulate $N = 15$ vehicles, with directions

$$\{\text{NS}, \text{ES}, \text{WE}, \text{NW}, \text{WN}, \text{WN}, \text{WS}, \text{NE},$$
$$\text{NE}, \text{EW}, \text{NS}, \text{ES}, \text{WS}, \text{SW}, \text{WE}\}$$

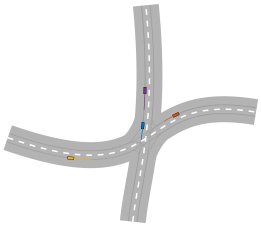
where NS denotes "from North to South" and so on. For example: Agent 1 is a leading agent; Agent 2 must maintain a safe distance from Agent 1 because the paths NS and ES intersect, thus $\chi(2) = 1$; Agent 4 (NW) must maintain a safe distance from Agent 1 (NS), because the path of agent 4 does not intersect with the ones of agents 2,3, thus $\chi(4) = 1$. We define $x = \text{col}(x_i)_{i \in \mathscr{I}}$, where the state of each vehicle x_i is

$$x_i = \begin{cases} v^{\text{ref}} - v_i & \text{if } i \in \mathscr{L} \\ \begin{bmatrix} p_{\chi(i)} - p_i - d_i \\ v_{\chi(i)} - v_i \end{bmatrix} & \text{if } i \notin \mathscr{L}. \end{cases} \tag{5.41}$$

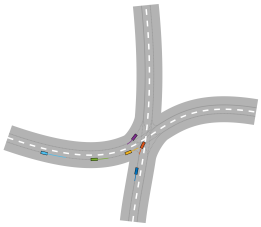
In the latter, p_i denotes the progress (in meters) of vehicle i starting from the entrance point to the intersection, d_i is the desired distance from $\chi(i)$, and v_i denotes the speed. Each vehicle is modelled as a double integrator, discretized with rate $\tau_s = 0.1s$. The matrices defining (5.2) are

$$A = \text{blkdg}(A_i)_{i \in \mathscr{I}}, \text{ where } A_i = \begin{cases} 1 & \text{if } i \in \mathscr{L}, \\ \begin{bmatrix} 1 & \tau_s \\ 0 & 1 \end{bmatrix} & \text{if } i \notin \mathscr{L} \end{cases}$$

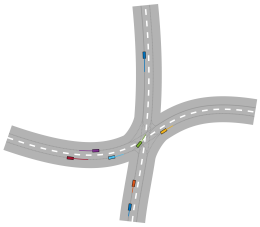
$$B_i = \text{col}(B_{ij})_{j \in \mathscr{I}}, \text{ where } B_{ij} = \begin{cases} \begin{bmatrix} \tau_s^2/2 & \tau_s \end{bmatrix}^\top & \text{if } i = \chi(j) \\ -\begin{bmatrix} \tau_s^2/2 & \tau_s \end{bmatrix}^\top & \text{if } i = j \text{ and } i \notin \mathscr{L} \\ -\tau_s & \text{if } i = j \text{ and } i \in \mathscr{L} \\ 0_{1\times 2} & \text{else.} \end{cases} \tag{5.42}$$



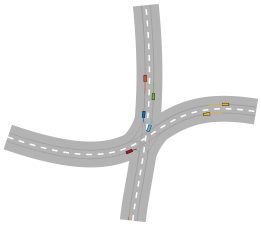
(a) Agents 1–4 approach the intersection.



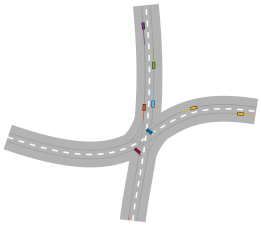
(b) Agent 1 (Blue) crosses the intersection first. Agents 2 (orange), and 4 (violet) cross next, simultaneously.



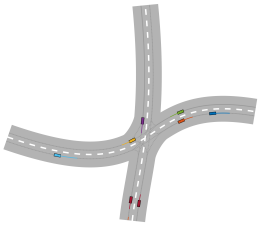
(c) Agent 3 (yellow) crosses the intersection, followed by Agent 5 (green).



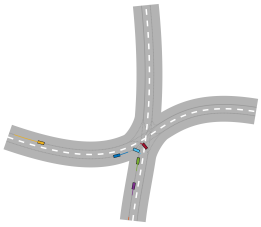
(d) Agent 6 (cyan) crosses the intersection. Agents 7–9 approach.



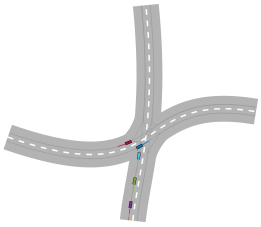
(e) Agents 7 (red) and 8 (blue) cross the intersection simultaneously, followed by agent 9 (orange).



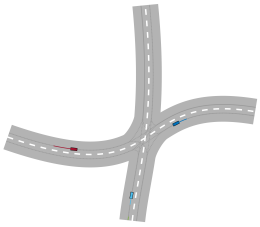
(f) Agent 10 (yellow) crosses the intersection, followed by Agent 11 (purple). Agents 12 (green), 13 (cyan), 14 (red) approach the intersection.



(g) Agents 11 (purple) and 12 (green) cross the intersection, followed simultaneously by Agents 13 (cyan) and 14 (red).



(h) Agent 15 (blue) crosses the intersection last.



(i) The intersection is cleared.

Figure 5.2: Simulated scenario. The full animation is available at https://tinyurl.com/444xr94f.

In order to satisfy the stabilizability assumption, each agent applies the pre-stabilizing local controller $\bar{K}_i$ such that

$$\bar{K}_i x = -0.1 \cdot \mathbb{1}^\top x_i.$$

The constraints include the safety distance and the speed constraints, as well as an input box constraint:

$$p_{\chi(i)} - p_i \geq d_{\min},$$
$$v^{\min} \leq v_i \leq v^{\max},$$
$$u^{\min} \leq u_i \leq u^{\max}.$$

The state and input weight matrices are $Q_i = I_n$, $R_i = 1$. We observe in Figure 5.3 that all vehicles achieve the desired reference speed and distance, while satisfying the constraints. An animation of the simulated scenario is available at the provided link, where it is observed that the vehicles safely complete the maneuver. We illustrate some extracted frames of the animation in Figure 5.2. At each time-step, we apply the Douglas-Rachford iteration in (5.37), warm-started at the "shifted" input sequence

$$\tilde{u}_i = (u_i^*[1], \ldots, u_i^*[T-1], K_i x_T^*), \qquad \forall i \in \mathscr{I},$$

5

where $\boldsymbol{u}^*$ is the solution to (5.18), where the initial state x_0 is substituted with the current state of the system, and $x_T^* = \phi(T, x[t], \boldsymbol{u}^*(t))$. Figures 5.4a and 5.4b illustrate that the proposed method is significantly more computationally efficient than the Forward-Backward splitting method [4] and the ADMM-inspired method proposed in [162], both in terms of the number of convergence iterations and the computational time. We note that the speed-up due to the warm-start is not available for the ADMM method. This is because the ADMM algorithm disregards the warm-start and, as a first step, moves the solution estimate to the one of a regularized unconstrained game. As shown, the number of iterations required to converge to the solution decreases with the evolution of the system, and we speculate that this is due to the system state converging towards the set $\mathbb{X}_f$, where the warm-start coincides with the solution to (5.18), in view of Proposition 5.1.6. We consider that the algorithm has converged when $r_k \leq 10^{-3}$, where r_k is the natural residual at iteration k of the iterative method (5.37) [4, §6.2.1].

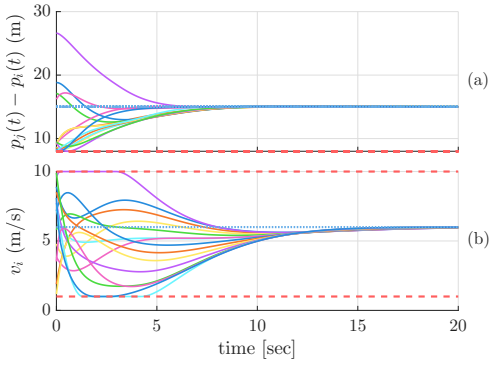
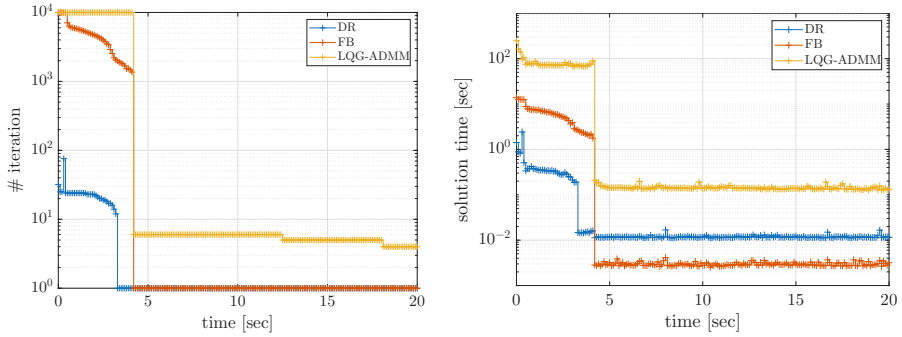


Figure 5.3: (a): Distance between $\chi(i)$ and i. (b): Velocity of each agent. The dotted lines denote the reference values, and the red dashed lines denote the constraints.



(a) Number of iterations.

(b) Computational time.

Figure 5.4: Number of iterations and computational time to achieve convergence of the VI solution.

5.4. CONCLUSION

For linear-quadratic dynamic games, an open-loop Nash equilibrium is the solution to an affine variational inequality, if the terminal state belongs to a forward-invariant, constraint-admissible set for the infinite-horizon unconstrained Nash equilibrium. The Douglas-Rachford splitting-like method is particularly suited for this affine variational inequality application, as it shows a remarkable convergence speed. Fast convergence to the solution enables the adoption of receding-horizon game-theoretic control architectures. Additionally, considering that the Nash equilibrium is known in closed-form near the equilibrium attractor of the dynamic game, the computational effort required to solve the associated VI is further reduced by employing an informed warm-start. Future work will investigate other algorithms and applications of real-time control for multi-agent robotic systems based on receding-horizon variational inequalities [163].

6

NASH EQUILIBRIUM SEEKING FOR A CLASS OF QUADRATIC-BILINEAR WASSERSTEIN DISTRIBUTIONALLY ROBUST GAMES

In continuation of the variational inequality formulation introduced in the preceding two chapters, we consider a class of Wasserstein distributionally robust Nash equilibrium problems, where agents construct heterogeneous data-driven Wasserstein ambiguity sets using private samples and radii, in line with their individual risk-averse preference. By leveraging relevant properties of this class of games, we show that equilibria of the original seemingly infinite-dimensional problem can be obtained as a solution to a finite-dimensional Nash equilibrium problem. We then reformulate the associated variational problem as a finite-dimensional variational inequality and establish the connection between the corresponding solution sets.

A wide range of applications, from energy and communication networks to social networks and economics [164], [165], [166], [167], [168], [169], [170] can be modeled as a collection of self-interested interacting decision makers optimizing different objective functions under operational constraints. Game theory [171] provides the fundamental theoretical framework for analyzing such systems. Although investigating deterministic games can be adequate in some case studies [164], [172], most real-world applications involve decision making under uncertainty, which stresses the need for the inclusion of stochasticity in the existing models [173], [174], [175]. Several studies have explored uncertainty within a game theoretic context, based on particular technical assumptions on the probability distribution [176], [177] and/or the properties of the uncertainty sample space [178], [179]. Other works have focused on the analysis of robust versions of uncertain cooperative games [180], [181], [182].

When the probability distribution is unknown and distribution models are not an accurate description of the stochastic aspect of the problem, sampling-based or data-driven methods have shown strong potential for proposing robust solutions against uncertainty. Works such as [183], [184], [185], [186], [187], [188], [189] design distribution-free approaches for data-driven Nash equilibria based on statistical learning techniques. More specifically, [186], [187], [188], [189] account for possible strategic perturbations around the Nash equilibrium. Separately, works based on Sample Average Approximation (SAA) techniques, such as [190], [191], develop algorithms for finding Nash equilibria in stochastic settings by using expected values of cost functions. The works mentioned above constitute data-driven methods for stochastic equilibrium seeking. These works, however, do not account for ambiguity in the probability distribution, where the distribution itself may be uncertain within some known bounds. The challenge of ambiguity in the distributions becomes pronounced in multi-agent settings, where heterogeneous uncertainties affect the agents' costs, often necessitating the consideration of different ambiguity sets, each representing the individual risk-averse nature of each agent.

To account for distributional uncertainty, distributionally robust optimization (DRO) uses a so-called ambiguity set of possible probability distributions to make decisions robust against probabilistic variations within this set. Unlike scenario-based methods, which require many samples for robustness, DRO can perform well with limited data by adjusting the ambiguity set. DRO includes special cases like sample average approximation (SAA) and robust optimization (RO). At the same time, DRO can be less conservative than RO and offer better out-of-sample performance than SAA, making it especially useful in data-driven applications with limited data. Recently, Wasserstein ambiguity sets [192], which use empirical data and the Wasserstein metric to measure distributional deviations, have gained attention. These sets are favored for penalizing horizontal shifts and providing finite-sample guarantees. Research has focused on convergence of empirical estimates in the Wasserstein distance [193], [194], [195], [196], [197], [198], as well as obtaining tractable reformulations of Wasserstein distributionally robust optimization problems [194], [199], [200], [201]. Extensions of those works include distributionally robust chance-constrained programs [202], [203], [204].

Despite the considerable body of literature on DRO with Wasserstein ambiguity sets, the exploration of data-driven Wasserstein distributionally robust Nash equilibrium problems with heterogeneous uncertainty in the cost functions represents a notably under-

explored topic. Most works in the literature consider moment-based methods or other measures of distance between distributions. For instance, [205] considers a non-cooperative game with distributionally robust chance-constrained strategy sets applied to duopoly Cournot competition. The work [206] develops distributionally robust equilibrium models based on the Kullback-Leibler (KL) divergence for hierarchical competition in supply chains. Other works mainly consider ambiguity in the constraints, such as the recent work [207], which studies a game with deterministic cost for each agent and distributionally robust chance constraints with the centre of the Wasserstein ambiguity set being an elliptical distribution; [208] reformulates an equilibrium problem with a deterministic cost and distributionally robust chance-constraints as a mixed-integer generalized Nash equilibrium problem leveraging the results in [209]. The contributions of this work with respect to the related literature are the following:

1. We study a class of heterogeneous data-driven Wasserstein distributionally robust games with a quadratic-bilinear structure for each agent objective function. Each agent is considered to act as a selfish entity making individual decisions to optimize their own cost function, which depends on a private uncertain parameter. The game-theoretic formulation generalizes the optimization framework to allow for interdependence of the agent cost functions. Furthermore, as we model possibly heterogeneous probability distributions, we lift the assumption of a common ambiguity set, allowing each agent to construct their own, based on their private data and personalized Wasserstein radius. In this generalized setting, the solution set can significantly differ from those associated with common ambiguity sets, studied under an optimization framework in [194], [199], [210], [211].

2. We reformulate the original game as a robust Nash equilibrium problem and establish the connection between the distributionally robust and robust Nash equilibria of the corresponding problems. For this class of games, we demonstrate that the inner maximization can be solved without the use of epigraphic variables [199], [212], which, unlike distributionally robust optimization, in distributionally robust games can lead to unshared coupling constraints. As such, our approach decreases computational complexity significantly. To the best of our knowledge, this is the first distributionally robust game-theoretic reformulation that leads to data-scalable results by leveraging the structure of the problem at hand.

3. The robust Nash equilibrium problem is then reformulated as a variational inequality (VI). Unlike results of similar classes of problems in optimization [213], where the reformulated variational inequality is monotone under certain assumptions, the mapping corresponding to the game can be nonmonotone in general due to the heterogeneity of the agents' ambiguity sets and costs. However, we show that this problem can be efficiently solved empirically using two algorithms: the adaptive golden ratio algorithm (aGRAAL) [46] and a hybrid version of this algorithm (Hybrid-Alg in [214]). Notably, our numerical results show that in several cases, the convergence speed is close to linear, and increasing the number of samples does not slow down the convergence. Our results are then applied to a portfolio allocation game that takes into account market uncertainty and behavioural coupling of market participants.

6.1. PROBLEM FORMULATION

Notation: In this section, we introduce some basic notation and results required for the subsequent developments. To this end, consider the index set $\mathcal{N} = \{1,\dots,N\}$. The decision vector of each agent $i \in \mathcal{N}$ is denoted by $x_i = \text{col}((x_i^{(j)})_{j=1}^n) \in X_i \subseteq \mathbb{R}^n$, where $x_i^{(j)}, j = 1,\dots,n$, denotes an element of the decision vector; let $x_{-i} = \text{col}((x_j)_{j=1,j\neq i}^N) \in X_{-i} = \prod_{i=1,i\neq j}^N X_j \subseteq \mathbb{R}^{(N-1)n}$ be the decision vector of all other agents' decisions except for that of agent i and $x = \text{col}((x_i)_{i=1}^N)$ be the collective decision vector. We denote $||\cdot|| = \|\cdot\|_2$. The projection operator $\text{proj}_X(x)$ of a point x to the set X is given by $\text{proj}_X(x) = \arg\min_{y\in X}\|x-y\|$. F is monotone on X if $(x-y)^\top(F(x)-F(y)) \geq 0$ for all $x, y \in X$. If the condition is not satisfied, the mapping is called nonmonotone.

Let us denote $\mathcal{P}(\mathbb{R}^m)$ as the set of all probability measures on $\mathbb{R}^m$ and define

$$\mathcal{M}(\mathbb{R}^m) = \left\{\mathbb{Q} \mid \mathbb{Q} \text{ is a distribution on } \mathbb{R}^m \text{ and } \mathbb{E}_{\mathbb{P}}[\|\xi\|] = \int_\Xi \|\xi\|\mathbb{Q}(d\xi) < \infty\right\}.$$

In other words, $\mathcal{M}(\mathbb{R}^m)$ considers the sets of all distributions defined on $\mathbb{R}^m$ with a bounded first-order moment. We are now ready to define the Wasserstein metric to quantify the distance between two probability distributions.

Definition 6.1.1. *The Wasserstein metric $d_W : \mathcal{M}(\mathbb{R}^m) \times \mathcal{M}(\mathbb{R}^m) \to \mathbb{R}_{\geq 0}$ between two distributions $\mathbb{Q}_1, \mathbb{Q}_2 \in \mathcal{M}(\mathbb{R}^m)$ is defined as*

$$d_W(\mathbb{Q}_1,\mathbb{Q}_2) := \left(\inf_{\Pi\in\mathcal{J}(\xi_1\sim\mathbb{Q}_1,\xi_2\sim\mathbb{Q}_2)}\int_{\mathbb{R}^m\times\mathbb{R}^m}\|\xi_1-\xi_2\|^p\Pi(d\xi_1,d\xi_2)\right)^{1/p},$$

where $\mathcal{J}(\xi_1\sim\mathbb{Q}_1,\xi_2\sim\mathbb{Q}_2)$ represent the set of joint probability distributions of the random variables ξ_1 and ξ_2 with marginals $\mathbb{Q}_1$ and $\mathbb{Q}_2$, respectively. □

The Wasserstein metric can be viewed as the optimal transport plan to fit the probability distribution $\mathbb{Q}_1$ to $\mathbb{Q}_2$ [192].

6.1.1. PROBLEM FORMULATION

Consider a population of agents with index set $\mathcal{N} = \{1,\dots,N\}$. Based on that we define the following game:

$$\forall i \in \mathcal{N} : \min_{x_i\in X_i}\max_{\mathbb{Q}_i\in\mathscr{P}_i}\{f_i(x_i,x_{-i}) + \mathbb{E}_{\xi_i\sim\mathbb{Q}_i}[g_i(x_i,x_{-i},\xi_i)]\},$$

where $f_i : \mathbb{R}^n \times \mathbb{R}^{n(N-1)} \to \mathbb{R}$, $g_i : \mathbb{R}^n \times \mathbb{R}^{n(N-1)} \times \mathbb{R}^m \to \mathbb{R}$ for all $i \in \mathcal{N}$ and $\mathscr{P}_i$ is the ambiguity set of the uncertain parameter ξ_i. We call the collection of the coupled optimization problems above for all agents $i \in \mathcal{N}$ as game (G).

For game G, we define the notion of *distributionally robust Nash equilibrium* as follows:

Definition 6.1.2. *A decision vector $x^* \in \prod_{i=1}^N X_i$ is a distributionally robust Nash equilibrium (DRNE) of game G if, given the decisions of all other agents $x_{-i}^* \in X_{-i}$ it holds that*

$$x_i^* \in \arg\min_{x_i\in X_i}\max_{\mathbb{Q}_i\in\mathscr{P}_i}\{f_i(x_i,x_{-i}^*) + \mathbb{E}_{\xi_i\sim\mathbb{Q}_i}[g_i(x_i,x_{-i}^*,\xi_i)]\},\ \forall i \in \mathcal{N}. \tag{6.1}$$

In other words, a decision vector x^* is a DRNE of G if, for each agent i, given the equilibrium strategies x^*_{-i} of all other agents with their respective local sets, the following holds: player i chooses their strategy x_i in a way that minimizes their objective, considering both their deterministic cost f_i and the worst-case expected effect of distributional uncertainty in ξ_i of g_i. This must hold for all agents simultaneously, ensuring that no agent can improve their outcome by unilaterally changing their strategy, even in the face of worst-case distribution.

In this work, we follow a data-driven approach and consider *heterogeneous* Wasserstein ambiguity sets constructed by each individual agent on the basis of their own individual data. Thus, we define an appropriate notion of distance between probability distributions. Due to its ability to penalize horizontal dislocations of distributions and often capturing realistic shifts in distributions, in this work we will use the Wasserstein distance. Specifically, for each $i \in \mathcal{N}$ the empirical probability distribution is constructed based on K_i independent and identically distributed (i.i.d.) samples $\xi_{K_i} = \{\xi_i^{(1)}, \ldots, \xi_i^{(K_1)}\}$ drawn by agent i as follows:

$$\hat{\mathbb{P}}_{K_i} = \frac{1}{K_i} \sum_{k_i=1}^{K_i} \delta_{\xi_i^{(k_i)}}$$

where δ_{ξ_i} is the Dirac delta measure that assigns the full probability mass at the point ξ_i. We then consider a radius ε_i, based on the Wasserstein distance, and construct the data-driven Wasserstein ambiguity ball of agent i as follows:

$$\mathscr{P}_i = \{Q_i \in \mathscr{P}(\mathbb{R}^m) \mid d_W(Q_i, \hat{\mathbb{P}}_{K_i}) \leq \varepsilon_i\}, \tag{6.2}$$

where $\mathscr{P}(\mathbb{R}^m)$ denotes the collection of all probability distributions defined on the support set $\mathbb{R}^m$.

6.1.2. EFFECT OF HETEROGENEOUS AMBIGUITY SETS ON THE EQUILIBRIUM SET

We consider a simple example to illustrate how heterogeneity of each agents' ambiguity sets, which correspond to an uncertain parameter $\xi \in \mathbb{R}^2$, can affect the solution set of DRNE even in the case of 1-Wasserstein distance ($p = 1$). In the following distributionally robust game, two self-interested agents solve the following interdependent optimization problems:

$$\begin{cases} \min\limits_{x_1} \max\limits_{\mathbb{Q}_1 \in \mathscr{P}_{\varepsilon_1}(\hat{\mathbb{P}}_{K_1})} \mathbb{E}_{\mathbb{Q}_1}[c_{11}x_1^2 + c_{12}x_1x_2 + \xi^\top x] \\ \min\limits_{x_2} \max\limits_{\mathbb{Q}_2 \in \mathscr{P}_{\varepsilon_2}(\hat{\mathbb{P}}_{K_2})} \mathbb{E}_{\mathbb{Q}_2}[c_{21}x_1x_2 + c_{22}x_2^2 + \xi^\top x] \end{cases} \tag{6.3}$$

The game in (6.3) can be equivalently written as follows [199]:

$$\begin{cases} \min\limits_{x_1} c_{11}x_1^2 + c_{12}x_1x_2 + \frac{1}{K_1}\sum\limits_{k=1}^{K_1} (\xi_1^{(k)})^\top x + \varepsilon_1 \|x\|_* \\ \min\limits_{x_2} c_{21}x_1x_2 + c_{22}x_2^2 + \frac{1}{K_2}\sum\limits_{k=1}^{K_2} (\xi_2^{(k)})^\top x + \varepsilon_2 \|x\|_*, \end{cases}$$

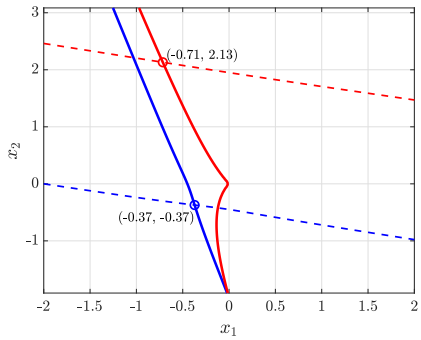


Figure 6.1: Equilibrium solution for homogeneous (blue lines) and heterogeneous (red lines) ambiguity sets for a 2-player game with scalar decisions. The dashed lines correspond to the second equation in (5).

where $\|\cdot\|_*$ represents the dual norm of $\|\cdot\|$. At an equilibrium $\bar{x} = (\bar{x}_1, \bar{x}_2)$, considering the 2-norm for the transport cost, the following stationarity conditions hold:

$$\begin{cases} 2c_{11}\bar{x}_1 + c_{12}\bar{x}_2 + \varepsilon_1 \frac{\bar{x}_1}{\sqrt{\bar{x}_1^2+\bar{x}_2^2}} = -\frac{1}{K_1}\sum_{k=1}^{K_1} \xi_1^{(k)} \\ c_{21}\bar{x}_1 + 2c_{22}\bar{x}_2 + \varepsilon_2 \frac{\bar{x}_2}{\sqrt{\bar{x}_1^2+\bar{x}_2^2}} = -\frac{1}{K_2}\sum_{k=1}^{K_2} \xi_2^{(k)} \end{cases} \tag{6.4}$$

We now show how, even for simple discrete distributions concentrated at one point, a difference between $\mathbb{P}_1$ and $\mathbb{P}_2$ can lead to a significant change to the equilibrium solution. Consider a discrete probability distribution $\mathbb{P}_1$ with full probability mass at $x = p_1$ (i.e., with probability 1) and 0 elsewhere, and the probability distribution $\mathbb{P}_2$ with full probability mass at $x = p_2$, 0 elsewhere:

$$\begin{cases} 2c_{11}\bar{x}_1 + c_{12}\bar{x}_2 + \varepsilon_1 \frac{\bar{x}_1}{\sqrt{\bar{x}_1^2+\bar{x}_2^2}} = -p_1 \\ c_{21}\bar{x}_1 + 2c_{22}\bar{x}_2 + \varepsilon_2 \frac{\bar{x}_2}{\sqrt{\bar{x}_1^2+\bar{x}_2^2}} = -p_2 \end{cases} \tag{6.5}$$

In Figure 6.1, we first consider $c_{11} = 1$, $c_{12} = 0.5$, $c_{21} = 0.5$, $c_{22} = 1$ and homogeneous ambiguity sets, i.e, $\varepsilon_1 = 0.1$, $\varepsilon_2 = 0.1$, $p_1 = 1$, $p_2 = 1$. Considering heterogeneous ambiguity sets ($\varepsilon_1 = 2$, $\varepsilon_2 = 0.1$, $p_1 = 1$, $p_2 = -4$), we note that the equilibrium point changes. Thus, our setting generalizes standard distributionally robust optimization to games of agents with self-interested objectives that can be affected by different distributions, have their own individual data and select their own preferred risk-aversion via their Wasserstein radius. This example shows how different radii and different samples, can introduce information asymmetry in the problem at hand, thus significantly changing the set of equilibria.

6.2. WASSERSTEIN DISTRIBUTIONALLY ROBUST QUADRATIC-BILINEAR GAMES

In this work, we consider a class of quadratic-bilinear games under the 2-Wasserstein distance ($p = 2$). We impose the following assumption:

Assumption 6.2.1. *1. For each $i \in \mathcal{N}$, f_i is convex in x_i for any given $x_{-i} \in X_{-i}$;*

2. For each $i \in \mathcal{N}$, g_i has the form $g_i(x_i, x_{-i}, \xi_i) = \xi_i^\top Q_i \xi_i + P_i(x)\xi_i$;

3. P_i is affine in x, i.e. $P_i(x) = A_i x + b_i$, where $A_i \in \mathbb{R}^{m \times nN}$ and $b_i \in \mathbb{R}^m$ for all $i \in \mathcal{N}$;

4. There exists an orthogonal matrix L_i such that $Q_i = L_i^\top D_i L_i$, where D_i is a diagonal positive semidefinite matrix with sorted eigenvalues.

Note that A_i can be written as $A_i = (A_i^{(1)}, A_i^{(2)}, \ldots, A_i^{(N)})$, where $A_i^{(j)} \in \mathbb{R}^{m \times n}$ corresponds to the submatrix to be multiplied with the elements of x_j for each $j \in \mathcal{N}$. The structure of function g_i allows for each agent to determine individually how much they wish to penalize large deviations of the uncertain parameter, represented by the quadratic term $\xi_i^\top Q_i \xi_i$. Furthermore, the bilinear term $P_i(x)\xi_i$ models the interplay between uncertainty and decisions and is important in models where the collective decision of the agents amplifies the effects of uncertainty in the cost. Assumption (iv) allows Q_i to be represented as $Q_i = L_i^\top D_i L_i$, where L_i is orthogonal and D_i is a diagonal positive semidefinite matrix with sorted eigenvalues. This form leverages the benefits of orthogonal transformations and simplifies the analysis of quadratic forms, while still being general enough to cover a wide range of practical scenarios.

Considering an ambiguity set per agent as in (6.2), we then obtain the following result:

Lemma 6.2.2. *Let Assumption 6.2.1 hold. Fix the Wasserstein radii ε_i and consider a multi-sample $\xi_{K_i} \in \mathbb{R}^{mK_i}$ for each agent $i \in \mathcal{N}$. Then, each agent's $i \in \mathcal{N}$ problem admits the following dual reformulation:*

$$\min_{\substack{x_i \in X_i \\ \lambda_i \geq 0}} J_i(x_i, \lambda_i, x_{-i}), \tag{6.6}$$

where

$$J_i(x_i, \lambda_i, x_{-i}) = f_i(x_i, x_{-i}) + \lambda_i \varepsilon_i^2 + \frac{1}{K_i} \sum_{k_i=1}^{K_i} \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top Q_i \xi_i + P_i(x)\xi_i - \lambda_i \|\xi_i - \xi_i^{(k_i)}\|^2]. \tag{6.7}$$

Proof: The proof follows by application of the Kantorovich duality [215]. ■

We call the collection of the coupled optimization problems above for all agents $i \in \mathcal{N}$ as game $\bar{G}$. Note that this reformulation has an additional dual variable λ_i for each $i \in \mathcal{N}$ that corresponds to the Lagrange multiplier associated with each individual Wasserstein constraint. Through this reformulation, a distributionally robust Nash equilibrium problem can be recast as an augmented robust Nash equilibrium problem. To connect the solutions of G and $\bar{G}$, we first provide the definition of the robust Nash equilibrium (RNE) for game $\bar{G}$ as follows:

Definition 6.2.3. *A decision vector* (x^*, λ^*)*, where* $\lambda^* = \text{col}((\lambda_i)_{i\in\mathcal{N}})$ *is a RNE of* G *if*

$$\forall i \in \mathcal{N} : (x_i^*, \lambda_i^*) \in \underset{x_i\in X_i, \lambda_i \geq 0}{\text{argmin}} \; J_i(x_i, \lambda_i, x_{-i}^*),$$

with J_i *as in (6.7).* □

The following lemma establishes the connection between the set of DRNE of G and the set of RNE of $\bar{G}$ defined as follows:

Lemma 6.2.4. *Let* (x^*, λ^*) *be an RNE of* $\bar{G}$ *in (6.6). Then,* x^* *is a DRNE of* G *in (6.1).* □

Proof: For a given $x_{-i}^* \in X_{-i}$*, since* (x^*, λ^*) *is an RNE of* $\bar{G}$*, we have*

$$\begin{aligned}
&\max_{\mathbb{Q}_i\in\mathscr{P}_i} \{f_i(x_i^*, x_{-i}^*) + \mathbb{E}_{\xi_i\sim\mathbb{Q}_i}[\xi_i^\top Q_i\xi_i + P_i(x_i^*, x_{-i}^*)\xi_i\} \\
&= \min_{\lambda_i\geq 0} f_i(x_i^*, x_{-i}^*) + \lambda_i\varepsilon_i^2 + \frac{1}{K_i}\sum_{k_i=1}^{K_i} \sup_{\xi_i\in\mathbb{R}^m} [\xi_i^\top Q_i\xi_i + P_i(x_i^*, x_{-i}^*)\xi_i - \lambda_i\|\xi_i - \xi_i^{(k_i)}\|^2] \\
&= f_i(x_i^*, x_{-i}^*) + \lambda_i^*\varepsilon_i^2 + \frac{1}{K_i}\sum_{k_i=1}^{K_i} \sup_{\xi_i\in\mathbb{R}^m} [\xi_i^\top Q_i\xi_i + P_i(x_i^*, x_{-i}^*)\xi_i - \lambda_i^*\|\xi_i - \xi_i^{(k_i)}\|^2] \\
&\leq \min_{x_i\in X_i, \lambda_i\geq 0} f_i(x_i, x_{-i}^*) + \lambda_i\varepsilon_i^2 + \frac{1}{K_i}\sum_{k_i=1}^{K_i} \sup_{\xi_i\in\mathbb{R}^m} [\xi_i^\top Q_i\xi_i + P_i(x_i, x_{-i}^*)\xi_i - \lambda_i\|\xi_i - \xi_i^{(k_i)}\|^2] \\
&= \min_{x_i\in X_i} \max_{\mathbb{Q}_i\in\mathscr{P}_i} \{f_i(x_i, x_{-i}^*) + \mathbb{E}_{\xi_i\sim\mathbb{Q}_i}[\xi_i^\top Q_i\xi_i + P_i(x_i, x_{-i}^*)\xi_i]\}, \qquad (6.8)
\end{aligned}$$

where the inequality holds from Definition 6.2.3. ■

Note that the inverse direction does not necessarily hold, as one should determine an appropriate value for λ^*. From Lemma 6.2.4 we can instead solve game $\bar{G}$ and obtain the solution (x^*, λ^*) and, from this solution, select x^* as the DRNE of our original problem G. To achieve this, we impose the following standing assumption:

Assumption 6.2.5. *The set of RNE of* $\bar{G}$ *in (6.6) is non-empty.* □

The non-emptiness of the set of RNE of $\bar{G}$ then directly implies the non-emptiness of the set of DRNE of game G. To solve the inner maximization over the uncertain parameter ξ_i, we show that the class of games that satisfies Assumption 6.2.1 can be exploited to provide a finite-dimensional formulation, without the use of an epigraphic reformulation. The following theorem leverages the structure of the problem to obtain a more computationally efficient reformulation thus circumventing those challenges.

Theorem 1. *Under Assumption 6.2.1,* G *admits the reformulation*

$$\begin{aligned}
G_R : \forall i \in \mathcal{N} : \min_{x_i\in X_i, \lambda_i > \lambda_{max}(Q_i)} \{&f_i(x_i, x_{-i}) + \lambda_i\left(\varepsilon_i^2 - \frac{1}{K_i}\sum_{k_i=1}^{K_i} (\xi_i^{(k_i)})^\top \xi_i^{(k_i)}\right) + \\
&+ \frac{1}{4K_i}\sum_{k_i=1}^{K_i} \tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i)^\top \tilde{Q}_i(\lambda_i)\tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i)\},
\end{aligned}$$

where $\tilde{Q}_i(\lambda_i) = diag(\frac{1}{\lambda_i - \lambda_{max}(D_i)}, \ldots, \frac{1}{\lambda_i - \lambda_{min}(D_i)})$ *and* $\tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i) = \tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}_i^{(k_i)}$, $\tilde{P}_i(x_i, x_{-i}) = L_i P_i(x_i, x_{-i})$ *and* $\tilde{\xi}_i^{(k_i)} = L_i \xi_i^{(k_i)}$. □

Proof: For each agent $i \in \mathcal{N}$ it holds that:

$$\begin{aligned}
&\frac{1}{K_i}\sum_{k_i=1}^{K_i} \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top Q_i \xi_i + P_i(x_i, x_{-i})\xi_i - \lambda_i \|\xi_i - \xi_i^{(k_i)}\|^2] \\
&= \frac{1}{K_i}\sum_{k_i=1}^{K_i} \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top Q_i \xi_i + P_i(x_i, x_{-i})\xi_i - \lambda_i(\xi_i^\top \xi_i - 2\xi_i^\top \xi_i^{(k_i)} + (\xi_i^{(k_i)})^\top \xi_i^{(k_i)})] \\
&= \frac{1}{K_i}\sum_{k_i=1}^{K_i} \left(-\lambda_i (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} + \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top (Q_i - \lambda_i I_m)\xi_i + (P_i(x_i, x_{-i}) + 2\lambda_i \xi_i^{(k_i)})^\top \xi_i] \right)
\end{aligned}$$

Since for each $i \in \mathcal{N}$, Q_i is diagonalizable, there exists matrix L_i such that $Q_i = L_i^\top D_i L_i$, where D_i is a diagonal matrix, whose eigenvalues decrease along the diagonal. Denote the maximum eigenvalue of D_i by $\lambda_{max}(D_i)$ and the minimum eigenvalue of D_i by $\lambda_{min}(D_i)$. As such, the following equalities hold:

$$\begin{aligned}
&\frac{1}{K_i}\sum_{k_i=1}^{K_i} \left(-\lambda_i (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} + \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top (Q_i - \lambda_i I_m)\xi_i + (P_i(x_i, x_{-i}) + 2\lambda_i \xi_i^{(k_i)})^\top \xi_i] \right) \\
&= \frac{1}{K_i}\sum_{k_i=1}^{K_i} \left(-\lambda_i (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} + \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top (D_i - \lambda_i I_m)\xi_i + [L_i P_i(x_i, x_{-i}) + 2\lambda_i L_i \xi_i^{(k_i)}]^\top \xi_i] \right) \\
&= \frac{1}{K_i}\sum_{k_i=1}^{K_i} \left(-\lambda_i (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} + \sup_{\xi_i \in \mathbb{R}^m} [\xi_i^\top (D_i - \lambda_i I_m)\xi_i + \tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i)^\top \xi_i] \right) \qquad (6.9)
\end{aligned}$$

Consider now $G_i^{(k_i)}(x_i, x_{-i}, \lambda_i, \xi_i) = \xi_i^\top (D_i - \lambda_i I_m)\xi_i + [\tilde{P}_i(x_i, x_{-i})\xi_i + 2\lambda_i \tilde{\xi}_i^{(k_i)})]^\top \xi_i$. Due to the presence of the supremum in (6.9), we wish to study for which value of the uncertainty ξ_i we achieve the maximum value for $G_i^{(k_i)}(x_i, x_{-i}, \lambda_i, \xi_i)$. This maximum value will be parametrized by the corresponding sample $\xi_i^{(k_i)}$. We distinguish between two different cases:

1. For $\lambda_i > \lambda_{max}(Q_i)$ we note that $(D_i - \lambda_i I_m)^{-1}$ is negative semidefinite. Thus, given other agents' decisions x_{-i}, the resulting cost function is concave which yields the solution

$$\sup_{\xi_i \in \mathbb{R}^m} \xi_i^\top (D_i - \lambda_i I_m)\xi_i + [\tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}_i^{(k_i)})]^\top \xi_i = G_i^{(k_i)}(x_i, x_{-i}, \lambda_i, \xi_i^*), \qquad (6.10)$$

where ξ_i^* is obtained by the first order optimality condition $\nabla_{\xi_i^*} G_i^{(k_i)}(x_i, x_{-i}, \lambda_i, \xi_i) = 0$. As such, the maximum is attained at $\xi_i^* = \frac{1}{2}(\lambda_i I_m - D_i)^{-1}(\tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}_i^{(k_i)})$ with optimal value:

$$\begin{aligned}
&G_i^{(k_i)}(x_i, x_{-i}, \lambda_i, \xi_i^*) = (\xi_i^*)^\top (D_i - \lambda_i I_m)\xi_i^* + [\tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}_i^{(k_i)})]^\top \xi_i^* \\
&= \frac{1}{4}(\tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}_i^{(k_i)})^\top (\lambda_i I_m - D_i)^{-1}(\tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}^{(k_i)})) \qquad (6.11)
\end{aligned}$$

2. For $\lambda_i \in [0, \lambda_{max}(Q_i))$, we note that $D_i - \lambda_i I_m$ is positive semidefinite, hence the cost function of the inner maximization problem is convex in ξ_i, which implies that $\sup_{\xi_i \in \mathbb{R}^m} \xi_i^\top (D_i - \lambda_i I_m)\xi_i + [\tilde{P}_i(x_i, x_{-i}) + 2\lambda_i \tilde{\xi}_i^{(k_i)})]^\top \xi_i = \infty$.

As such, given the agents' decisions x_{-i} each agent $i \in \mathcal{N}$ solves

$$\min_{x_i \in X_i, \lambda_i > \lambda_{max}(Q_i)} \{f_i(x_i, x_{-i}) + \lambda_i \left(\varepsilon_i^2 - \frac{1}{K_i} \sum_{k_i=1}^{K_i} (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} \right) + \\ + \frac{1}{4K_i} \sum_{k_i=1}^{K_i} \tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i)^\top \tilde{Q}_i(\lambda_i) \tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i)\},$$

where $\tilde{Q}_i(\lambda_i) = (\lambda_i I_m - D_i)^{-1}$. Then, the connection between the games G and $\bar{G}$ as established in Lemma 6.2.2 and their corresponding solutions in Lemma 6.2.4 concludes the proof. ■

6.2.1. REFORMULATION AS A DATA-DRIVEN VARIATIONAL INEQUALITY PROBLEM

In this section, we establish the connection of the Nash equilibria of G_R with the solutions of a variational inequality (VI) problem [216]. For the ease of the reader, we define the notion of a Nash equilibrium for a general game.

Definition 6.2.6. *Consider the following game:*

$$\forall\ i \in \mathcal{N} : \min_{z_i \in Z_i} J_i(z_i, z_{-i}), \tag{6.12}$$

A point $z^* = (z_i^*, z_{-i}^*) \in Z = \prod_{i=1}^N Z_i$ is Nash equilibrium (NE) of (6.12) if, given z_{-i}^*, the following condition holds:

$$J_i(z_i^*, z_{-i}^*) \leq J_i(z_i, z_{-i}^*),$$

for all $z_i \in Z_i$ and for all $i \in \mathcal{N}$. □

The following statement then holds:

Proposition 6.2.7. *Consider the following game:*

$$\forall\ i \in \mathcal{N} : \min_{z_i \in Z_i} J_i(z_i, z_{-i}), \tag{6.13}$$

where J_i is convex on Z_i for any fixed $z_{-i} \in Z_{-i}$ and $Z = \prod_{i=1}^N Z_i$ is convex and closed. Furthermore, consider the following variational inequality problem:

$$F^\top(z^*)(z - z^*) \geq 0,\ \forall\ z \in Z \cap V(z^*) \text{ for all } i \in \mathcal{N}.$$

where $F(z) = \text{col}((F_i(z))_{i \in \mathcal{N}})$ with $F_i(z) = \nabla_{z_i} J_i(z_i, z_{-i})$ is a (possibly nonmonotone) mapping and $V(z^*)$ is a small enough convex neighbourhood around z^*. Then, any local solution z^* of the VI is a Nash equilibrium of (6.13).

Proof: This result is a direct extension of the proofline of Proposition 1.4.2 in [4] for a nonmonotone mapping defined over a small enough convex neighbourhood $V(z^*)$ of the solution. ■

Returning to game G_R, note that, according to Definition 6.2.6, a point $(x^*, \lambda^*) \in X \times \prod_{i=1}^{N}(\lambda_{max}(Q_i)), \infty)$ is a Nash equilibrium of G_R if, given $x^*_{-i} \in X_{-i}$, the following condition holds:

$$f_i(x^*) + \lambda_i^* \left(\varepsilon_i^2 - \frac{1}{K_i} \sum_{k_i=1}^{K_i} (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} \right) + \frac{1}{4K_i} \sum_{k_i=1}^{K_i} \tilde{W}^{(k_i)}(x^*, \lambda_i^*)^\top \tilde{Q}_i(\lambda_i^*) \tilde{W}^{(k_i)}(x^*, \lambda_i^*) \leq$$

$$f_i(x_i, x^*_{-i}) + \lambda_i \left(\varepsilon_i^2 - \frac{1}{K_i} \sum_{k_i=1}^{K_i} (\xi_i^{(k_i)})^\top \xi_i^{(k_i)} \right) + \frac{1}{4K_i} \sum_{k_i=1}^{K_i} \tilde{W}^{(k_i)}(x_i, x^*_{-i}, \lambda_i)^\top \tilde{Q}_i(\lambda_i) \tilde{W}^{(k_i)}(x_i, x^*_{-i}, \lambda_i)$$

for all $(x_i, \lambda_i) \in X_i \times (\lambda_{max}(Q_i), \infty)$.

Finally, from the proofline of Theorem 1, it immediately follows that the set of NE of G_R coincides with the set of RNE of (6.6). Let us now denote by $z_i = (x_i, \lambda_i) \in \mathbb{R}^{(n+1)N}$ the collection of the decision vector and the Lagrange multiplier for all $i \in \mathcal{N}$ and $z = \text{col}((z_i)_{i \in \mathcal{N}})$. Furthermore, denote the feasible set $Z = \{z \in \mathbb{R}^{(n+1)N} : x_i \in X_i, \lambda_i \geq \lambda_{max}(Q_i) + \zeta_i, \ \forall i \in \mathcal{N}\}$, where ζ_i is an arbitrarily small positive parameter, ensuring that the local constraint set is closed and thus game G_R, where $\lambda_i > \lambda_{\max}(Q_i)$, can be solved with any a priori defined accuracy. An exact solution is obtained when $\zeta_i \to 0^+$. The following lemma then holds:

Lemma 6.2.8. *A solution z^* of the VI problem with mapping $F(z) = \text{col}(F_i(z))_{i \in \mathcal{N}}$, where*

$$F_i(z) = \begin{pmatrix} \nabla_{x_i} f_i(x_i, x_{-i}) + \frac{1}{2K_i} \sum_{k_i=1}^{K_i} (A_i^{(i)})^\top \tilde{Q}_i(\lambda_i) \tilde{W}^{(k_i)}(x_i, x_{-i}, \lambda_i) \\ \varepsilon_i^2 - \frac{1}{K_i} \sum_{k_i=1}^{K_i} \|\xi_i^{(k_i)}\|^2 + \frac{1}{4K_i} \sum_{k_i=1}^{K_i} 4\xi_i^{(k_i)\top} \tilde{Q}_i(\lambda_i) \tilde{W}^{(k_i)}(x, \lambda_i) - \|\tilde{W}^{(k_i)}(x, \lambda_i)\|^2_{\frac{d\tilde{Q}_i}{d\lambda_i}} \end{pmatrix}. \tag{6.14}$$

over $Z \cap V(z^*)$, with $V(z^*)$ being a convex local neighbourhood around z^*, is a Nash equilibrum of G_R over Z.

Proof: The proof follows by adaptation of Proposition 6.2.7 by taking the pseudogradient of game G_R considering that $\tilde{Q}_i$ is a diagonal matrix, hence $\frac{d\tilde{Q}_i(\lambda_i)}{d\lambda_i}$ is obtained by differentiating the corresponding diagonal elements. ■

6.2.2. CHALLENGES OF DISTRIBUTIONALLY ROBUST HETEROGENEOUS GAMES COMPARED TO DRO

Let us illustrate the challenges that our game-theoretic problem entails compared to the standard DRO. Let us first define the associated distributionally robust optimization problem as follows:

$$\min_{x \in X} \max_{\mathbb{Q} \in \mathscr{P}_\varepsilon(\hat{\mathbb{P}}_K)} \mathbb{E}_{\xi \sim \mathbb{Q}}[x^\top Q x + \xi^\top Q \xi + P(x)^\top \xi], \tag{6.15}$$

First, note that the optimization problem in (6.15) is only a special case of the original game considered in (6.1). In particular, the setting in (6.1) considers more than one players, who make self-interested decisions based on their own *private samples* and *different* Wasserstein radii. The interdependence of each agent's cost to the decisions of the other agents significantly affects the resulting decision, unlike optimization where this decision is collective and only nature is viewed as an adversarial player. This implies that the pseudogradient mapping might be non-monotone; On the contrary, in a standard optimization problem, the gradient map is always monotone.

For example, consider the following distributionally robust game with 2 agents $i \in \{1,2\}$ taking scalar decisions. Specifically, agent 1 aims at solving:

$$\min_{x_1 \in X_1} \max_{\mathbb{Q}_1 \in \mathscr{P}_1} \mathbb{E}_{\xi_1 \sim \mathbb{Q}_1}[q_{11}x_1^2 + q_{12}x_1x_2 + Q_1\xi_1^2 + (a_1x_1 + a_2x_2)\xi_1],$$

while agent 2 aims at solving:

$$\min_{x_2 \in X_2} \max_{\mathbb{Q}_2 \in \mathscr{P}_2} \mathbb{E}_{\xi_2 \sim \mathbb{Q}_2}[q_{22}x_2^2 + q_{21}x_1x_2 + Q_2\xi_2^2 + (b_1x_1 + b_2x_2)\xi_2],$$

By Theorem 1, the problem above can be written as:

$$\forall\, i \in \mathcal{N} : \min_{(x_i,\lambda_i) \in X_i \times \mathbb{R}^m_{\geq 0}} J_i(x_i, x_j, \lambda_i), \tag{6.16}$$

where

$$J_1(x_1,x_2,\lambda_1) = q_{11}x_1^2 + q_{12}x_1x_2 + \lambda_1\left(\varepsilon_1^2 - \frac{1}{K_1}\sum_{k_1=1}^{K_1}(\xi_1^{(k_1)})^2\right) + \frac{1}{4K_1}\sum_{k_1=1}^{K_1}\frac{(a_1x_1 + a_2x_2 + 2\lambda_1\xi_1^{(k_1)})^2}{\lambda_1 - \lambda_{max,1}},$$

and

$$J_2(x_1,x_2,\lambda_2) = q_{22}x_2^2 + q_{21}x_1x_2 + \lambda_2\left(\varepsilon_2^2 - \frac{1}{K_2}\sum_{k_2=1}^{K_2}(\xi_2^{(k_2)})^2\right) + \frac{1}{4K_2}\sum_{k_2=1}^{K_2}\frac{(b_1x_1 + b_2x_2 + 2\lambda_2\xi_2^{(k_2)})^2}{\lambda_2 - \lambda_{max,2}}.$$

To study the monotonicity of the VI mapping associated with (6.16), we can directly study whether the game Jacobian is positive semidefinite. The following lemma shows that for this class of games, even if $K_1 = K_2$ (note that only the number is the same, thus still allowing for different samples per agent), the Jacobian is more complicated than the calculated one for standard optimization.

Lemma 6.2.9. *Consider the functions*

$$\begin{aligned}
G_1^{(k_1)}(x_1,x_2,\lambda_1) &= 2\xi_1^{(k_1)}\lambda_{1,\max} + a_1x_1 + a_2x_2,\\
G_2^{(k_2)}(x_1,x_2,\lambda_2) &= 2\xi_2^{(k_2)}\lambda_{2,\max} + b_1x_1 + b_2x_2,\\
\Delta_1 &= \lambda_1 - \lambda_{1,\max}\\
\Delta_2 &= \lambda_2 - \lambda_{2,\max},
\end{aligned} \tag{6.17}$$

corresponding to each sample $k_1 \in \{1,\dots,K_1\}$ and $k_2 \in \{1,\dots,K_2\}$. Furthermore, consider for simplicity that $K_1 = K_2 = K$ and $q_{11} = q_{12} = q_{22} = q_{21} = 0$. The symmetrized Jacobian J is then given by

$$J = \frac{1}{4K}\sum_{k=1}^{K} J_s^{(k)},$$

where

$$J_s^{(k)} = \begin{bmatrix} \frac{2a_1^2}{\Delta_1} & \frac{2a_1 G_1^{(k)}(x,\lambda_1)}{\Delta_1^2} & \frac{a_1 a_2}{\Delta_1} + \frac{b_1 b_2}{\Delta_2} & \frac{b_1 G_2^{(k)}(x,\lambda_2)}{\Delta_2^2} \\ \frac{2a_1 G_1^{(k)}(x,\lambda_1)}{\Delta_1^2} & \frac{2G_1^{(k)}(x,\lambda_1)^2}{\Delta_1^3} & \frac{a_2 G_1^{(k)}(x,\lambda_1)}{\Delta_1^2} & 0 \\ \frac{a_1 a_2}{\Delta_1} + \frac{b_1 b_2}{\Delta_2} & \frac{a_2 G_1^{(k)}(x,\lambda_1)}{\Delta_1^2} & \frac{2b_2^2}{\Delta_2} & \frac{2b_2 G_2^{(k)}(x,\lambda_2)}{\Delta_2^2} \\ \frac{b_1 G_2^{(k)}(x,\lambda_2)}{\Delta_2^2} & 0 & \frac{2b_2 G_2^{(k)}(x,\lambda_2)}{\Delta_2^2} & \frac{2G_2^{(k)}(x,\lambda_2)^2}{\Delta_2^3} \end{bmatrix}. \tag{6.18}$$

Proof: The proof is obtained by performing standard algebraic operations on the defined Jacobian of the corresponding VI, considering the functions $G^{(k_1)}$ and $G^{(k_2)}$. We then use the assumption $K_1 = K_2 = K$ to pull a common sum outside and group the samples of the two agents in exhaustive pairs. ■

Based on Lemma 6.2.9, an important observation can then be made: When we consider *only* one agent, we derive as a special case of our original problem a distributionally robust optimization program with Jacobian:

$$J_s^{(k)} = \begin{bmatrix} \frac{2a_1^2}{\Delta_1} & \frac{2a_1 G_1^{(k_1)}(x,\lambda_1)}{\Delta_1^2} \\ \frac{2a_1 G_1^{(k_1)}(x,\lambda_1)}{\Delta_1^2} & \frac{2G_1^{(k_1)}(x,\lambda_1)^2}{\Delta_1^3} \end{bmatrix}, \tag{6.19}$$

For any k, $J_s^{(k)}$ can then be shown to be positive semi-definite. Thus J is positive semi-definite as the summation of positive semi-definite matrices. In contrast, the Jacobian (6.18) for the game in (6.15) might not be positive semi-definite which implies non-monotonicity. For example, the game in (6.17) with fixed decisions $x_1 = 2$, $x_2 = 2$, $\lambda_1 = \lambda_{1,\max} + 2$ and $\lambda_2 = \lambda_{2,\max} + 2$, and parameters $\lambda_{1,\max} = 0.5$, $\lambda_{2,\max} = 0.5$, $a_1 = 1$, $a_2 = 0.5$, $b_1 = 1$, $b_2 = 0.5$, has Jacobian with at least one negative eigenvalue meaning that the corresponding pseudogradient is not monotone.

Even though the mapping can in general be nonmonotone, we illustrate how equilibria can still be computed efficiently based both on the structure of our problem, as provided by Theorem 1, and the equilibrium seeking algorithms we propose in the next section. Specifically, leveraging the structure of our reformulation, we avoid introducing epigraphic variables that render both the decision and the constraints dependent on the size of the data [199], [194]. Thus, through Theorem 1, we can obtain reformulations for the class of heterogeneous data-driven Wasserstein distributionally robust games in (6.1) that scale better with data compared to the use of epigraph forms. In the next section, we assess the computational performance of our theoretical results through an illustrative example and a risk-aware portfolio allocation game, which takes into account behavioural coupling of the investors' decisions.

6.3. NUMERICAL SIMULATIONS

In the simulation results, we use three algorithms to solve the variational inequality problems:

(i) Forward-backward splitting (FB):

$$x^{k+1} = \text{prox}_{\tau g}(x^k - \tau F(x^k)),$$

Algorithm 5 Hybrid DRNE seeking algorithm (Hybrid-Alg)[214]

Require: Choose x^0, x^1, $\tau_0 > 0$, $\bar{\tau} \gg 0$, $\alpha = (1, \frac{1+\sqrt{5}}{2}]$, $\theta_0 = 1$, $\rho = \frac{1}{\alpha} + \frac{1}{\alpha^2}$, $\bar{\phi} \gg \frac{1+\sqrt{5}}{2}$, $\text{sum}_0^1 = 0$, $\text{sum}_0^2 = 0$, $\text{flg} = 1$.

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

Find the stepsize:

$$\tau_k = \min\left\{\rho\tau_{k-1}, \frac{\alpha\theta_{k-1}}{4\tau_{k-1}}\frac{\|x^k - x^{k-1}\|^2}{\|F(x^k) - F(x^{k-1})\|^2}, \bar{\tau}\right\}$$

$$\bar{x}^k = \frac{(\phi_k - 1)x^k + \bar{x}^{k-1}}{\phi_k}$$

Update the next iteration:

$$x^{k+1} = \text{prox}_{\tau_k g}(\bar{x}^k - \tau_k F(x^k))$$

Update: $\theta_{k+1} = \dfrac{\alpha\tau_k}{\tau_{k-1}}$

compute the following summations with $\phi_{k+1} = \bar{\phi}$:

$\text{sum}_{k+1}^1 = \text{sum}_k^1 +$ (6.20)

$\text{sum}_{k+1}^2 = \text{sum}_k^2 +$ (6.21)

if $(\text{sum}_{k+1}^1 \le 0 \ \wedge \ \text{flg} = 1) \ \vee \ (\text{sum}_{k+1}^2 \le 0 \ \wedge \ \text{flg} = 0)$ **then**

$\phi_{k+1} = \bar{\phi}$, $\text{flg} = 1$

else

if $\text{flg} = 1$ **then**

$x^{k+1} = x^k$, $\bar{x}^k = \bar{x}^{k-1}$

$\phi_{k+1} = \alpha$, $\theta_k = \theta_{k-1}$, $\tau_k = \tau_{k-1}$

$\text{sum}_{k+1}^1 = 0$, $\text{sum}_{k+1}^2 = 0$, $\text{flg} = 0$

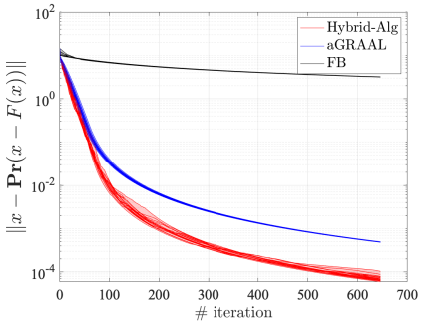
else

$\phi_{k+1} = \alpha$

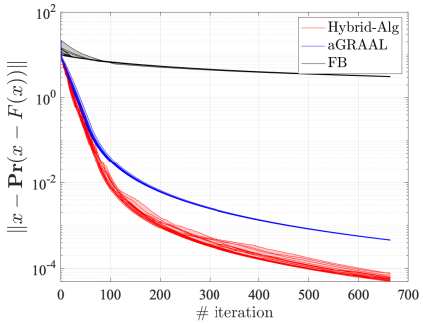
$\text{sum}_{k+1}^2 = \text{sum}_k^2 +$ ((6.21) with $\phi_{k+1} = \alpha$)

$\text{sum}_{k+1}^1 = 0$

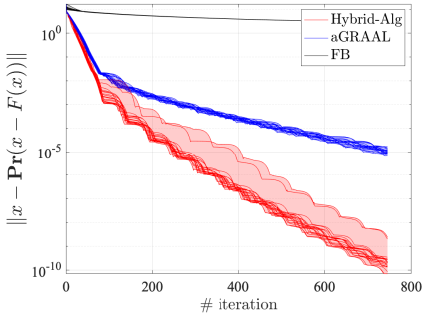
where τ is the stepsize. This method is known as a projected gradient descent algorithm as well which is the most used method in machine learning and control applications. The convergence of this method is guaranteed for strongly monotone (with a strongly monotone constant μ) and Lipschitz (with a Lipschitz constant L) operator with $\tau \in \left(0, 2\mu/L^2\right)$ [4, Theorem 12.4.6].



(a) $\varepsilon = 10^{-6}$.



(b) $\varepsilon = 10^{-3}$.



(c) $\varepsilon = 1$.

Figure 6.2: Residual of variational inequality problem with different radii ε_i per agent chosen according to the distribution $\varepsilon \cdot U[1,5]$, where ε takes values in $\{10^{-6}, 10^{-3}, 1\}$.

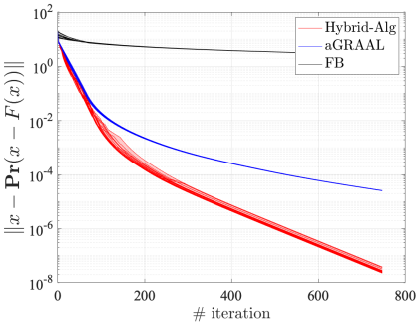
(ii) Adaptive golden ration algorithm (aGRAAL) [46]:

$$\phi \in (1, \frac{1+\sqrt{5}}{2}], \quad \rho = \frac{1}{\phi} + \frac{1}{\phi^2}$$
$$\tau_k = \min\left\{\rho\tau_{k-1}, \frac{\phi\theta_{k-1}}{4\tau_{k-1}} \frac{\|x^k - x^{k-1}\|^2}{\|F(x^k) - F(x^{k-1})\|^2}, \bar{\tau}\right\}$$
$$\bar{x}^k = \frac{(\phi-1)x^k + \bar{x}^{k-1}}{\phi}$$
$$x^{k+1} = \mathrm{prox}_{\tau_k g}(\bar{x}^k - \tau_k F(x^k)), \quad \theta_{k+1} = \frac{\phi\tau_k}{\tau_{k-1}}$$

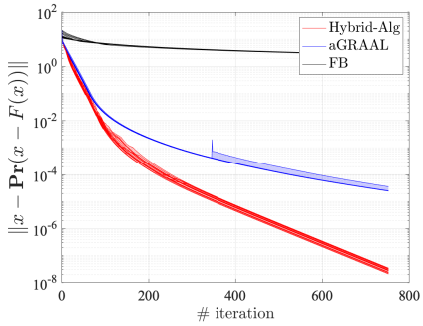
The convergence of this method is guaranteed for Lipschitz and monotone operator (with a Lipschitz constant L) [46, Lemma 3].

(iii) Hybrid method (Hybrid-Alg): The full steps of this method can be found in Algorithm 5. This algorithm is presented in the recent work [214]. However, we note

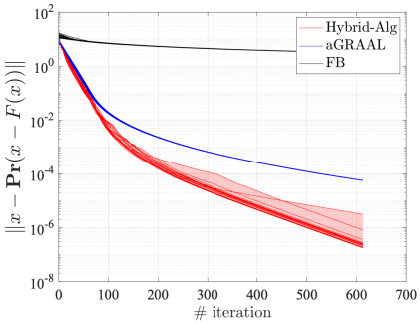
that the convergence rate proof only holds for monotone mappings and is applied on a significantly less complicated case study. The application of this algorithm on our problem shows that the equilibria of even a generally nonmonotone mapping could be numerically obtained relatively fast, even faster than the golden ratio method in practice. The Hybrid method, though similar to aGRAAL, differs significantly in the choice of the momentum parameter, as we explain later on.



(a) $K_i = \text{randi}((10,20),1,n)$.



(b) $K_i = \text{randi}((40,60),1,n)$.



(c) $K_i = \text{randi}((80,120),1,n)$.

Figure 6.3: Residual of variational inequality problem with different number of samples K_i per agent and radii ε_i chosen according to the distribution $\varepsilon \cdot U[1,5]$ with fixed radius $\varepsilon = 0.01$.

6.3.1. NUMERICAL EXAMPLE

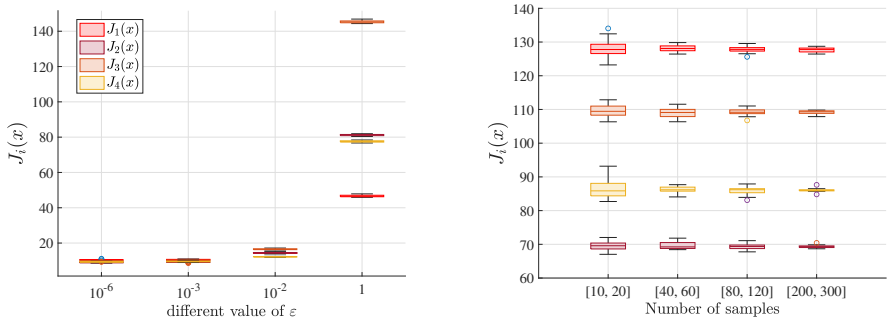
In this section, we reformulate a case study of the distributionally robust game in (6.1), under Assumption 6.2.1, as a variational inequality problem and solve it using FB (with $\tau = 0.001$), aGRAAL, and Hybrid-Alg. The key difference between aGRAAL and Hybrid-Alg algorithms is that, unlike aGRAAL, which uses a fixed momentum parameter, Hybrid-Alg employs a variable momentum parameter. We believe this is a testimony to the potential of switching the momentum parameter between a small (used in aGRAAL) and a large value, which has a significant impact on convergence speed. In particular, having

larger, variable momentum parameter in Algorithm 5 makes $\bar{x}_k$ closer to the most recent iterate, x_k, rather than $\bar{x}_{k-1}$, which allows us to estimate the local Lipschitz constant of the corresponding VI mapping F more precisely compared to aGRAAL.

Briefly speaking, the following two equations are evaluated in each iteration of HybridAlg (Algorithm 5) to assess sufficient decrease and to determine whether the large or the small momentum parameter should be used.

$$\frac{\theta_{k-1}}{2}\|x^k - x^{k-1}\|^2 - \frac{\tau_k}{\tau_{k-1}}\phi_k\|x^k - \bar{x}^k\|^2 + \Big(\frac{\tau_k}{\tau_{k-1}}\phi_k - 1 - \frac{1}{\phi_{k+1}}\Big)\|x^{k+1} - \bar{x}^k\|^2 - \Big(\frac{\tau_k}{\tau_{k-1}}\phi_k - \theta_k\Big)\|x^{k+1} - x^k\|^2 - \frac{\theta_k}{2}\|x^{k+1} - x^k\|^2. \tag{6.20}$$

$$-\frac{\tau_k\phi_k}{\tau_{k-1}}\|x^k - \bar{x}^k\|^2 + \Big(\frac{\tau_k\phi_k}{\tau_{k-1}} - 1 - \frac{1}{\phi_{k+1}}\Big)\|x^{k+1} - \bar{x}^k\|^2 - \Big(\frac{\tau_k\phi_k}{\tau_{k-1}} - \theta_k\Big)\|x^{k+1} - x^k\|^2. \tag{6.21}$$



(a) Different radii ε.

(b) Different number of samples.

Figure 6.4: Cost function values of four agents at the equilibrium for 10 different scenarios represented by box plots with (a) different radii (b) Both different samples and different number of samples per agent.

For the simulation, the parameters in the problem are generated as follows: Each drawn sample $\xi_i^{(k_i)}$ is generated from the uniform distribution with support set [0,1], while P_i is given by $P_i(x) = \sum_{i\in\mathcal{N}} a_i x_i$. The values a_i, and the eigenvalues of D_i in the reformulation (6.14) are randomized. Each agent's Wasserstein radius ε_i is chosen randomly according to the distribution $\varepsilon \cdot U[1,5]$, where ε takes fixed values in $\{10^{-6}, 10^{-3}, 10^{-2}, 1\}$ and $U[1,5]$ is a uniform discrete distribution with support set $\{1,2,\ldots,5\}$. Figure 6.2 shows the residual of the corresponding mapping F for the illustrative example for different Wasserstein radii and a fixed number of samples. We note that the convergence rate of both algorithms is almost linear, which illustrates that, even though the VI mapping can be nonmonotone, fast solutions can be obtained using both algorithms. Figure 6.3 shows the residual for an increasing number of invividual data for each agent and individual radii per agent. The number of each agent's samples for each case study is drawn from a discrete integer distribution in [10,20], [40,60] and [80,120], respectively. Note that even if we increase the number of samples, the convergence rate does not change,

thus leading to results that scale well with the sample size. Finally, Figure 6.4 illustrates how the cost of each agent at the equilibrium is affected by the Wasserstein radii and the number of samples of each agent for 10 different problem instances represented by boxplots. In Figure 6.4(a) we consider different radii ε_i per agent obtained from the distribution $\varepsilon \cdot U[1,5]$, where $\varepsilon \in \{10^{-6}, 10^{-3}, 10^{-2}, 1\}$ to investigate the effect of increasing Wasserstein radii on the cost of each agent; In Figure 6.4(b) the number of samples per agent follows a uniform distribution with support sets $\{[10,20],[40,50],[80,120],[200,300]\}$ per case study to investigate the effect an increasing number of samples has on the cost of each agent.

We observe that as we increase the value of the radii, the cost functions of each agent are higher representing a more conservative but robust behaviour against distrubutional shifts. Finally, for fixed radii, as the number of samples increases, the empirical variance of the costs decreases as well, as a result of a more accurate estimation of the probability distribution, used as the center of each ambiguity set.

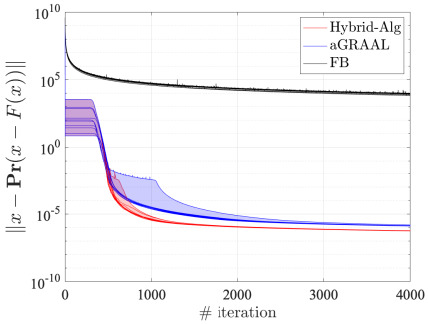
6.3.2. RISK-AWARE PORTFOLIO ALLOCATION UNDER MARKET UNCERTAINTIES AND BEHAVIOURAL INFLUENCES

We consider a multi-investor robust portfolio allocation problem, where each investor $i \in \mathcal{N}$ allocates capital seeking to maximize their profits or minimize their costs taking into account their exposure to market risks. The decision variable for each investor is their portfolio allocation $x_i \in X_i$, where X_i represents the set of feasible portfolios for investor i, normalized to a simplex representing the percentage of capital split among investments. Furthermore, we wish to model behavioural impacts of other investors onto each individual investor. Finally, we consider that agents are not only aware of the possible high variance of market uncertainties, but also aware that, when multiple investors accumulate to a single asset, this could lead to market bubbles which affects the returns from such investments. Thus, each investor's objective, given the other investors' strategies x_{-i}, is defined according to the following optimization problem:

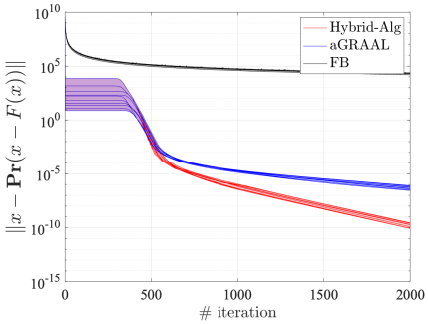
$$\min_{x_i \in X_i} \max_{\mathbb{Q}_i \in \mathscr{P}_i} \left\{ x_i^\top C_{ii} x_i + x_i^\top C_{ij} x_j - r_i^\top x_i + \mathbb{E}_{\xi_i \sim \mathbb{Q}_i} \left[\xi_i^\top Q_i \xi_i + P_i(x) \xi_i \right] \right\}.$$

The term $r_i^\top x_i$ represents the deterministic part of the returns based on the allocation of capital to assets. The quadratic deterministic terms models (possible) behavioural coupling due to competition of the investors according to performance metrics often used to make such investments.

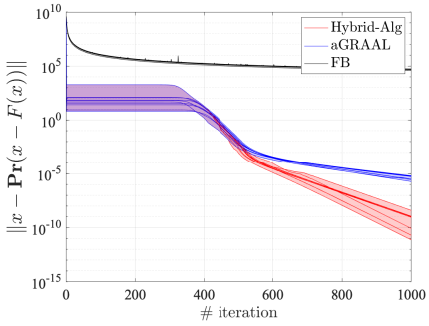
The ambiguity set $\mathscr{P}_i$ models investor i's ambiguity in the distribution of uncertain market parameters affecting the returns. The term $\xi_i^\top Q_i \xi_i$ represents i's aversion to volatility, indicating each agent's individual sensitivity to uncertain fluctuations. The term $P_i(x)\xi_i$, where $P_i(x) = \sum_{j \in \mathcal{N}} x_j$, models herding behavior, where multiple investors investing heavily in the same assets increase asset-specific risks. This crowding effect can drive prices up, raising the risk of market bubbles. In Figure 6.5, we set the Wasserstein radii at $\varepsilon \in \{10^{-6}, 0.01, 1\}$ and consider 10 different instances of the problem with different values of matrices Q_i, C_{ii}, C_{ij}, r_i and different multi-samples per agent $i \in \mathcal{N}$ obtained from different t-distributions. Note that even though the mapping is in general nonmonotone, most case studies lead to satisfactory (mostly linear) convergence results



(a) $\varepsilon = 10^{-6}$.



(b) $\varepsilon = 10^{-2}$.



(c) $\varepsilon = 1$.

Figure 6.5: Residual for different instances of the portofolio allocation problem with each agent's ε_i chosen according to $\varepsilon \cdot U[1,5]$ with $\varepsilon \in \{10^{-6}, 10^{-2}, 1\}$.

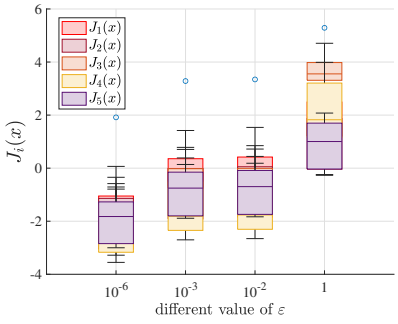


Figure 6.6: Cost function values of four agents at the equilibrium for 10 different scenarios represented by box plots for different radii ε_i per agent obtained from the uniform distribution $\varepsilon \cdot U[1,5]$, where $\varepsilon \in \{10^{-6}, 10^{-3}, 10^{-2}, 1\}$ to investigate the effect of increasing Wasserstein radii on the cost of each agent.

with both schemes, Hybrid-Alg (red lines) and aGRAAL (blue lines). In most of the case studies, the superiority of the hybrid algorithm is evident. Figure 6.6 shows the values of

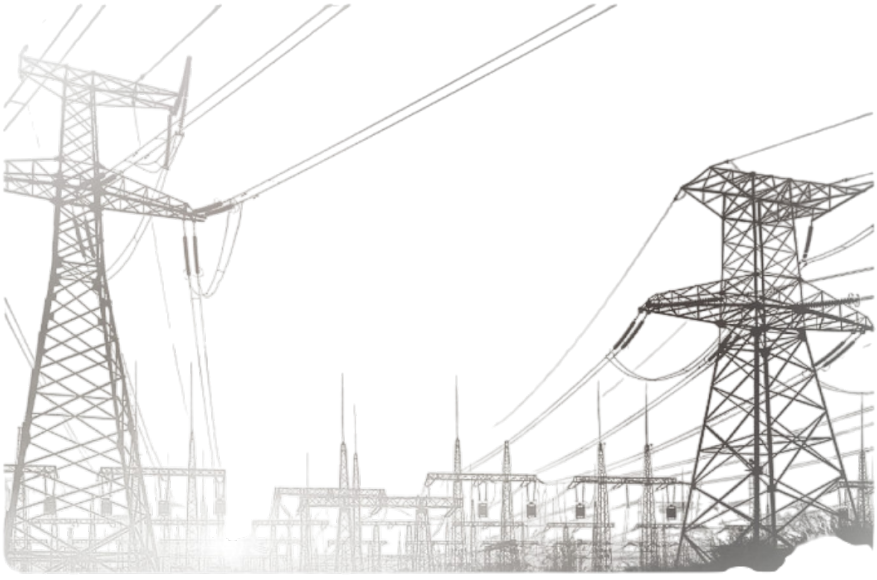
the cost functions of the agents at the equilibrium point for those 10 different instances represented by a box plot. Even though the problem is nonmonotone, increasing the Wasserstein radius of the agents leads in general to a larger value of the cost function.

6.4. CONCLUSION

This work explores data-driven distributionally robust games using individual Wasserstein ambiguity sets and private data, thus allowing agents to develop their own personalized risk-averse decisions. We reformulate a seemingly-infinite dimensional game into a data-driven finite-dimensional variational inequality problem, which enjoys data-scalability properties. Future work will focus on introducing coupling constraints to our model. Extending on that we wish to investigate this problem under the presence of distributionally robust chance constraints coupling the agents decisions and in particular, whether certain assumptions such as linearity of the constraints can aid in obtaining a satisfactory reformulation or approximation of the original game.

III

Application to Multi-Agent Optimization and Control



7

AN EMBEDDED ACCELERATED DECENTRALIZED OPTIMIZATION ALGORITHM WITH APPLICATION TO ENERGY COMMUNITIES

The previous chapters addressed various optimization problems, namely convex optimization and variational inequalities, through mathematical analysis and illustrative numerical experiments. This chapter focuses on modeling real-world applications using convex and variational-inequality formulations, and then applying iterative methods to solve the resulting problems.

In this chapter, we model energy communities using a bilevel optimization framework, analyze the methodology and solution approach, and implement and test the resulting algorithm on the real embedded system.

Energy demand on power distribution networks is steadily increasing, due to electrification trends and the widespread integration of renewable energy systems. This growth introduces new technical challenges related to grid congestion, voltage regulation, and particularly to energy losses along transmission and distribution lines, which represent a significant fraction of total energy waste worldwide. It has been demonstrated by a considerable number of studies that, if distributed generation is appropriately coordinated, it can have a significant effect in mitigating losses and enhancing overall grid efficiency. A key strategy for achieving these objectives involves the large-scale adoption of Renewable Energy Sources (RESs) and the promotion of decentralized, community-based models. Within this context, Renewable Energy Communities (RECs) have emerged as a promising framework to foster local energy production and consumption, reduce grid dependency, and support the energy transition. As defined by the European directive RED II [217], RECs are legal entities composed exclusively of renewable generation units, where partecipants (ECPs) jointly manage their energy flows to maximize local self-consumption and shared benefits. By reducing the distance between generation and demand, RECs can substantially lower transmission and distribution losses, improve local energy resilience, and enable active participation of prosumers in energy markets.

Although the structure of a REC may vary depending on the national legislative framework, a key and recurring concept is that of *shared energy*. This is typically defined as the minimum between the total production and total consumption of all ECPs within the same time interval (commonly one hour). Such energy exchanges between each ECP and the main grid are virtual as they rely on the existing grid infrastructure and do not require physical connections among members. Furthermore, to promote energy sharing, RECs are usually granted incentives proportional to the amount of shared energy. Given this definition, it becomes evident that aligning energy consumption and production is crucial for RECs to generate tangible benefits for the grid and to maximize incentives. Achieving this alignment requires the presence of flexible systems (e.g. Energy Storage Systems - ESS, Electric Vehicles - EVs, flexible loads, etc.) and an effective Energy Community Manager (ECM), capable of ensuring a robust organizational and operational framework. A representation of the REC architecture is depicted in Fig. 7.1. Even if RECs consider existing technologies, they are not free from technical, economic, regulatory and social challenges [218]. Among the most critical aspects on the management point of view, ECPs must balance their economic interests with collective goals. Current incentive schemes encourage the spread of RECs, however, maximizing shared energy alone may negatively impact ECPs' costs. For this reason, the ECM is usually subject to possibly conflicting objectives: maximizing shared energy and minimizing ECPs' costs, balancing network-level benefits with individual economic objectives. The literature primarily focuses on aligning demand and generation profiles to maximize shared energy and related incentives. For instance, [219] analyze how prioritizing economic versus technical objectives influences REC operations: cost-oriented optimization maximizes financial benefits, whereas resilience-oriented optimization represents a worst-case scenario for grid service provision. Several studies have proposed formal models for optimal REC operation. In [220] a bilevel approach is introduced: the lower level performs market clearing via MILP, and the upper level redistributes profits by maximiz-

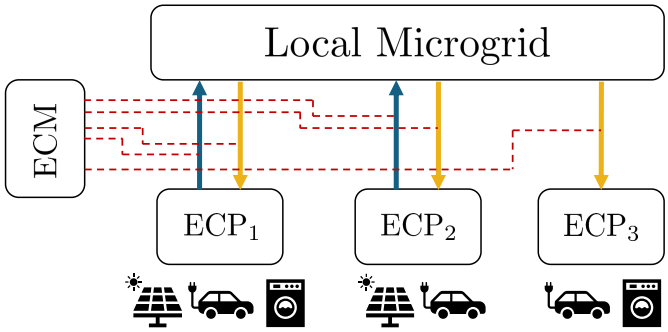


Figure 7.1: A generic representation of a REC with ECM and ECPs.

ing the minimum relative benefit among participants, while in [221], a MILP framework maximizes a social welfare function. [222] present a hierarchical mechanism in which users minimize their individual costs, followed by optimization at the ECM level of internal energy trades. Authors in [223] use a bilevel approach where the low-level problems (ECPs cost minimization) are converted into Karush–Kuhn–Tucker constraints for the high-level problem (shared energy maximization). However, this latter approach results in a centralized problem that cannot be decomposed and whose embedded implementation is significantly more challenging.

To overcome such computational and scalability limitations, recent works have explored embedded and distributed optimization methods for real-time energy management. The Alternating Direction Method of Multipliers (ADMM) and its accelerated variants enable decentralized optimization while preserving privacy and scalability [224], [225], [226], [227]. Recent studies have introduced Nesterov-accelerated ADMM and fast Quadratic Programming (QP) solvers suitable for low-resource embedded platforms, demonstrating real-time performance in microgrid and smart building applications [228], [229], [230], [231], [232].

Building on this literature, this paper introduces a novel embedded-oriented ECM framework for RECs. A multi-level architecture is proposed, where the high level focuses on maximizing shared energy, while the lower levels represent the cost minimization of individual ECPs. To enable distributed and efficient computation, the overall problem is reformulated as a compact QP problem suitable for decentralized optimization. The resulting problems are solved in parallel through an accelerated ADMM algorithm, which ensures fast convergence, scalability, and privacy preservation. The proposed approach is validated on resource-constrained embedded devices (ODROID-N2L and ODROID-H3+), demonstrating the feasibility of real-time operation and highlighting the practical relevance of lightweight optimization techniques for decentralized REC management. The selected microcontrollers feature multi-core ARM (Advanced RISC Machine) architectures and hardware-level floating-point support, making them suitable for real-time embedded control. Accordingly, the proposed optimization algorithm has been specifically designed to ensure deterministic execution times and computational efficiency under such embedded constraints, enabling its deployment in real REC management

systems.

The main contributions of this work can be summarized as follows:

- A novel decentralized ECM formulation that jointly maximizes shared energy among participants and minimizes individual user costs, effectively balancing collective welfare and economic efficiency within RECs.
- A fast, privacy-preserving accelerated ADMM algorithm for solving the QP reformulated ECM problem, designed to ensure rapid convergence, scalability, and data confidentiality, while being tailored for low-resource embedded platforms.
- An embedded implementation and experimental validation on low-power microcontrollers (ODROID-N2L and H3+), demonstrating an algorithm–hardware codesign that guarantees deterministic computation times, high numerical accuracy, and real-time feasibility in resource-constrained REC environments.

The remaining part of the paper is divided as follows. Section 7.1 describes the overall optimization problem for the REC and its reformulation as a QP. Section 7.2 details the solution approach using accelerated ADMM. In Section 7.3 the embedded implementation and case study results are presented. Finally, conclusions and future developments are presented in Section 7.4.

7.1. ENERGY COMMUNITY MODELING

The proposed architecture presents two main levels. The high level represents the ECM whose aim is to maximize the shared energy. Then for each ECP, there is a low-level problem which addresses the cost minimization. The sets of the optimization problems to be defined are:

- $\mathcal{T} = 1, 2, \ldots, T$ is the set of the time intervals.
- $\mathcal{N} = 1, 2, \ldots, N$ is the set of the ECPs.

The proposed optimization problem can be written in a general form as

$$\min_{u} \mathcal{F}(u)$$
$$\text{s.t.} \begin{cases} u \geq 0 \\ u_i \in \underset{u_i \in \mathcal{U}_i}{\arg\min} \mathcal{G}(u_i) \quad \forall i \in \mathcal{N} \end{cases} \tag{7.1}$$

where $\mathcal{F} : \mathbb{R}^n \to \mathbb{R}$ and $\mathcal{G} : \mathbb{R}^m \to \mathbb{R}$ are the objective functions of the high-level and low-level problems, respectively. Moreover, $u = \text{col}(u_i)$, with $u \in \mathbb{R}^n$, and $u_i \in \mathbb{R}^m$ which are the decision variables of the problem and where $\mathcal{U}_i \subset \mathbb{R}^m, i \in \mathcal{N}$ are the sets of the constraints of the i-th low-level problem. In the following subsections, the two levels considered in this work are described in detail.

7.1.1. THE ECM PROBLEM

As it mentioned, shared energy represent a key aspect of RECs. This quantity is important from a technical point of view because the greater the shared energy, the greater local consumption and therefore the less stress on the grid. In line with the European legislation, we assume that shared energy is defined as the "minimum between the electricity globally injected and the electricity globally withdrawn" [221], [223]. Therefore, we have

$$e_t^{sh} = \min\left(\sum_{i\in\mathcal{N}} p_{i,t}^{G,in}, \sum_{i\in\mathcal{N}} p_{i,t}^{G,out}\right)\Delta, \quad \forall t \in \mathcal{T} \tag{7.2}$$

where e_t^{sh} [kWh] is the shared energy, $p_{i,t}^{G,in}$ [kW] and $p_{i,t}^{G,out}$ [kW] are the power flows absorbed from/injected into the grid by each user i in each time interval, which length is defined by Δ [h]. A graphical representation of its calculation is provided by Fig. 7.2. However, this definition implies that to maximize the total shared energy, the overall

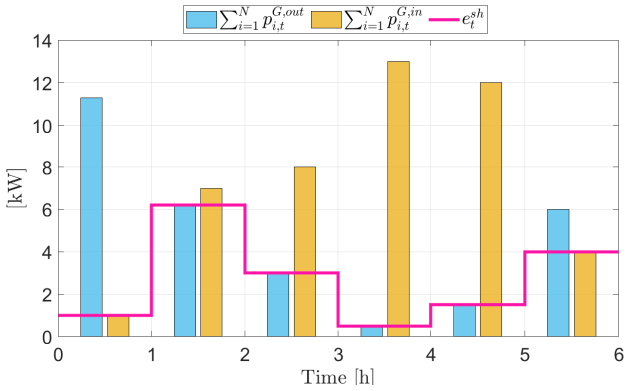


Figure 7.2: Example of shared energy evaluation.

production and consumption (considering all the users) must be the highest possible, at the same time. It follows that we want to minimize the overall exchange with the grid at all times, i.e. while maximizing each user absorption/injection. Combining the two terms, and normalizing them, we obtain the overall objective function to be minimized:

$$J^{sh} = \sum_{t\in\mathcal{T}} \left[\left(\sum_{i\in\mathcal{N}} \frac{p_{i,t}^{G,in} - p_{i,t}^{G,out}}{\overline{P}_i^G}\right)^2 - \sum_{i\in\mathcal{N}} \left(\frac{p_{i,t}^{G,out}}{\overline{P}_i^G}\right)\right]\Delta \tag{7.3}$$

where $\overline{P}_i^G$ [kW] is the maximum power that member i can exchange with the grid. Note that the only constraint that must be satisfied is

$$p_{i,t}^{G,in}, p_{i,t}^{G,out} \geq 0 \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.4}$$

7.1.2. THE ECPS' PROBLEMS

The low-level problem, to be written for each of the N ECPs, is presented next. To achieve the operating costs minimization, the objective function to be minimized is

$$J_i^{op} = \sum_{t \in \mathcal{T}} \left(C_{i,t}^{buy} p_{i,t}^{G,in} - C_{i,t}^{sell} p_{i,t}^{G,out} \right) \Delta \quad \forall i \in \mathcal{N} \tag{7.5}$$

where $C_{i,t}^{buy}$ and $C_{i,t}^{sell}$ [€/kWh] are the energy buying and selling prices of each user at each time, respectively. The power balance for each user is defined as

$$\begin{aligned} &p_{i,t}^{G,in} - p_{i,t}^{G,out} + P_{i,t}^{PV} + p_{i,t}^{S,dch} - p_{i,t}^{S,ch} = \\ &P_{i,t}^{L,fix} + p_{i,t}^{L,flex} + p_{i,t}^{EV} \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \end{aligned} \tag{7.6}$$

where $P_{i,t}^{PV}$ [kW] is power from photovoltaic plant, $p_{i,t}^{S,ch}$ [kW] and $p_{i,t}^{S,dch}$ [kW] are the power flows relevant to charging and discharging the storage, $P_{i,t}^{L,fix}$ [kW] is the fixed power load which cannot be controlled, $p_{i,t}^{L,flex}$ [kW] is the flexible power load portion, i.e. the load that can be shifted along the day, and $p_{i,t}^{EV}$ [kW] is the charging power of the electric vehicle. The constraints relevant to the energy storage system are

$$x_{i,t}^S = x_{i,t-1}^S + \frac{\Delta}{CAP^S} \left(\Gamma_i^{S,ch} p_{i,t}^{S,ch} - \frac{1}{\Gamma_i^{S,dch}} p_{i,t}^{S,dch} \right) \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.7}$$

$$0 \le p_{i,t}^{S,ch} \le \overline{P}_i^S \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.8}$$

$$0 \le p_{i,t}^{S,dch} \le \overline{P}_i^S \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.9}$$

$$X_i^S \le x_{i,t}^S \le \overline{X}_i^S \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.10}$$

where $x_{i,t}^S$ denotes the ESS state of charge, CAP^S [kWh] its nominal capacity, and $\Gamma_i^{S,ch}$ and $\Gamma_i^{S,dch}$ its charging and discharging efficiencies. Constraints (7.8)-(7.10) define the upper and lower bounds on power flows and the states of charge, namely $\overline{P}_i^S$, $\overline{X}_i^S$ and $\underline{X}_i^S$. According to current legislation, all power production must come exclusively from renewable sources, which also applies to energy stored in the ESS. Consequently, the charging power must originate solely from the photovoltaic (PV) system, and the ESS cannot be charged from the grid, as the exact energy mix cannot be guaranteed. In practice, this condition is automatically satisfied when the grid purchasing cost exceeds the combined benefits from energy sales and the REC incentive. Under these circumstances, it is never economically convenient to charge the ESS from the grid. On the other hand, charging from surplus PV generation (after serving local loads) is advantageous, as the stored energy can later be used to maximize shared energy and the corresponding incentive. The power exchange with the main grid is operated by

$$0 \le p_{i,t}^{G,out} \le \overline{P}_i^G \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.11}$$

$$0 \leq p_{i,t}^{G,in} \leq \overline{P}_i^G \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.12}$$

which upper bound the two terms defining $p_{i,t}^G$. The constraints relevant to the deferrable loads are as follows

$$\sum_{t \in \mathcal{T}} p_{i,t}^{L,flex} \Delta \geq E_i^{L,flex} \quad \forall i \in \mathcal{N} \tag{7.13}$$

$$0 \leq p_{i,t}^{L,flex} \leq \overline{P}_i^{L,flex} \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.14}$$

where (7.13) imposes that the overall deferrable loads of the i-th ECP is satisfied over the entire optimization horizon. Note that using an inequality in (7.13) is preferred over the equal to simplify the problem; this choice is further justified if one considers that the buying price from the network is always greater than the overall benefit (selling price+EC incentive) thus the solver tends to satisfy $E_i^{L,flex}$ as much as enough without exceeding. Then, the power devoted to satisfying the flexible load is subjected to bound constraints in (7.14). The constraints relevant to the electric vehicle (EV) are

$$x_{i,t}^{EV} = x_{i,t-1}^{EV} + \frac{\Delta}{CAP_i^{EV}} \Gamma_i^{EV} p_{i,t}^{EV} \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.15}$$

$$0 \leq p_{i,t}^{EV} \leq \overline{P}_i^{EV} \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.16}$$

$$\underline{X}_i^{EV} \leq x_{i,t}^{EV} \leq \overline{X}_i^{EV} \quad \forall i \in \mathcal{N}, \forall t \in \mathcal{T} \tag{7.17}$$

$$x_{i,t}^{EV} \geq X_i^{EV*} \quad \forall i \in \mathcal{N}, t = T_i^{EV*} \tag{7.18}$$

where $x_{i,t}^{EV}$ is the EV state of charge, CAP_i^{EV} [kWh] its capacity, Γ_i^{EV} its charging efficiency. Please note that in constraint (7.15) only the charging mode is considered (V2G mode is discarded) as in ECs the only energy resource allowed is from RES; therefore, EVs cannot work in V2G mode because in this case there would be no guarantee that such charging energy is fully renewable. Constraints (7.16) and (7.17) represent the upper and lower bounds of power flows and states of charge. Finally, constraint (7.18) ensures that the desired value for the EV state of charge is reached (and possibly exceeded) at the desired time instant T_i^{EV*}.

7.1.3. PROBLEM REFORMULATION

Now, the problem can be reformulated in canonical form. Note that the bold characters in the following denote vectors and matrices. Moreover, all the I matrices in the problem formulation have size $T \times T$. The considered control variables for each user are

$$u_i = \begin{bmatrix} p_i^{G,out} & p_i^{S,ch} & p_i^{S,dch} & p_i^{L,flex} & p_i^{EV} \end{bmatrix} \quad \forall i \in \mathcal{N} \tag{7.19}$$

The power balance in (7.6) and the ESS and EV state equations in (7.7) and (7.15) can be written as

$$\begin{bmatrix} I & I & -I & I & I \end{bmatrix} u_i = P_i^{PV} - P_i^{L,fix} + p_i^{G,in} \quad \forall i \in \mathcal{N} \tag{7.20}$$

$$x_i^S = \begin{bmatrix} 0 & K_i^{ch} & K_i^{dch} & 0 & 0 \end{bmatrix} u_i + 1 X_0^S \quad i \in \mathcal{N} \tag{7.21}$$

$$x_i^{EV} = \begin{bmatrix} 0 & K_i^{ch} & K_i^{dch} & 0 & 0 \end{bmatrix} u_i + 1X_{i,0}^{EV} \quad \forall i \in \mathcal{N} \tag{7.22}$$

where K_i^* is $T \times T$ matrices defined as:

$$K_i^{ch} = \begin{cases} k_{h,j} = 0, & \text{if } h < j, \\ k_{h,j} = \dfrac{\Gamma_i^{S,ch}\Delta}{CAP_i^S}, & \text{if } h \geq j, \end{cases} \quad \forall i \in \mathcal{N} \tag{7.23}$$

$$K_i^{dch} = \begin{cases} k_{h,j} = 0, & \text{if } h < j, \\ k_{h,j} = -\dfrac{\Delta}{\Gamma_i^{S,dch} CAP_i^S}, & \text{if } h \geq j, \end{cases} \quad \forall i \in \mathcal{N} \tag{7.24}$$

$$K_i^{EV} = \begin{cases} k_{h,j} = 0, & \text{if } h < j, \\ k_{h,j} = \dfrac{\Gamma_i^{EV}\Delta}{CAP_i^{EV}}, & \text{if } h \geq j, \end{cases} \quad \forall i \in \mathcal{N} \tag{7.25}$$

Note that the formulation in (7.21)-(7.25) represents the conversion of the dynamic problem to a static one. Then, making appropriate substitutions and some algebraic manipulations, inequality constraints (7.4), (7.8)-(7.14), (7.16)-(7.18) can be written in matrix form as

$$\Omega_i = \begin{bmatrix} 0 & -I & 0 & 0 & 0 \\ 0 & I & 0 & 0 & 0 \\ 0 & 0 & -I & 0 & 0 \\ 0 & 0 & I & 0 & 0 \\ 0 & -K^{ch} & -K^{dch} & 0 & 0 \\ 0 & K^{ch} & K^{dch} & 0 & 0 \\ -I & 0 & 0 & 0 & 0 \\ I & 0 & 0 & 0 & 0 \\ -I & -I & I & -I & -I \\ I & I & -I & I & I \\ 0 & 0 & 0 & -I & 0 \\ 0 & 0 & 0 & I & 0 \\ 0 & 0 & 0 & -1\Delta & 0 \\ 0 & 0 & 0 & 0 & -I \\ 0 & 0 & 0 & 0 & I \\ 0 & 0 & 0 & 0 & -K^{EV} \\ 0 & 0 & 0 & 0 & K^{EV} \\ 0 & 0 & 0 & 0 & -(K^{EV})_{T^{EV*}} \end{bmatrix}_i, \quad \Psi_i = \begin{bmatrix} 0 \\ \overline{P}^S \\ 0 \\ \overline{P}^S \\ -\underline{X}^S + 1X_0^S \\ \overline{X}^S - 1X_0^S \\ 0 \\ \overline{P}^G \\ -P^{PV} + P^{L,fix} \\ \overline{P}^G + P^{PV} - P^{L,fix} \\ 0 \\ \overline{P}^{L,flex} \\ -E^{L,flex} \\ 0 \\ \overline{P}^{EV} \\ -\underline{X}^{EV} + 1X_0^{EV} \\ -1X_0^{EV} + \overline{X}^{EV} \\ -X^{EV*} \end{bmatrix}_i \quad \forall i \in \mathcal{N} \tag{7.26}$$

Then, by substituting the equality constraints and rewriting in matrix form (7.3) and (7.5), the overall multi-level architecture can be written as in (7.1) with

$$\begin{aligned} \mathcal{F}(u) &= \frac{1}{2} u^\top Q u + q_{QP}^\top u \\ \mathcal{G}(u_i) &= \hat{q}_i^\top u_i \quad \forall i \in \mathcal{N} \end{aligned} \tag{7.27}$$

where $Q = \text{diag}(Q_i)$, $q_{QP} = \text{col}\{q_{QP,i}\}$

$$Q_i = \frac{2}{1\left(\overline{P}_i^G\right)^2}\begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & I & -I & I & I \\ 0 & -I & I & -I & -I \\ 0 & I & -I & I & I \\ 0 & I & -I & I & I \end{bmatrix} \tag{7.28}$$

$$q_{QP,i} = \frac{2}{\left(\overline{P}_i^G\right)^2}\left(P_i^{L,fix} - P_i^{PV}\right)^\top \begin{bmatrix} 0 & I & -I & I & I \end{bmatrix} - \frac{1}{\overline{P}_i^G}\begin{bmatrix} I & 0 & 0 & 0 & 0 \end{bmatrix} \tag{7.29}$$

$$\hat{q}_i = \begin{bmatrix} \Delta\left(C_i^{buy} - C_i^{sell}\right) \Delta C_i^{buy} - \Delta C_i^{buy} \Delta C_i^{buy} \Delta C_i^{buy} \end{bmatrix} \tag{7.30}$$

with $\mathscr{U}_i = \{u_i : \Omega_i u_i - \Psi_i \leq 0\} \quad \forall i \in \mathscr{N}$.

Due to the structure of the above defined matrix Q which is block diagonal, the problem (7.27) is fully separable among each ECP, by creating a fully decentralised architecture. This is one of the main contributions of this work and represents a valuable feature also from a practical perspective since the overall optmization is based only on local measurements and without any need for communication among ECPs. This also allows for keeping privacy among the participants by maintaining the optimality of the solution

In the next section, we will present an optimisation algorithm suitable for embedded applications that allows us to solve problems like (7.27) with performance levels appropriate for the proposed applications.

7.2. SOLUTION APPROACH

To assess a suitable ADMM-based optimisation algorithm for embedded devices, we first start from the optimisation problem of the single ECP

$$\begin{aligned} &\min_{u_i} \frac{1}{2} u_i^\top Q_i u_i + q_{QP,i}^\top u_i \\ &s.t. \begin{cases} u_i \geq 0 \\ u_i \in \underset{\Omega_i u_i - \Psi_i \leq 0}{\text{argmin}} \ \hat{q}_i^\top u_i \end{cases} \end{aligned} \tag{7.31}$$

That, thanks to the properties of the KKT conditions of a LP, is equivalent to the following QP

$$\min_{u_i, \lambda_i} \frac{1}{2} u_i^\top Q_i u_i + q_{QP,i}^\top u_i \tag{7.32}$$

$$s.t. \begin{cases} u_i \geq 0 \\ \lambda_i \geq 0 \\ \Omega_i u_i - \Psi_i \leq 0 \\ \hat{q}_i^\top + \Omega_i \lambda_i = 0 \\ \hat{q}_i^\top u_i + \Psi_i \lambda_i = 0 \end{cases} \tag{7.33}$$

whose standard form is given by

$$\min_x \frac{1}{2} x_i^\top P_i x_i + q_i^\top x_i$$
$$s.t. \begin{cases} G_i x_i \leq d_i \\ F_i x_i = r_i \end{cases} \tag{7.34}$$

where $x_i = [u_i, \lambda_i]$ and

$$P_i = \begin{bmatrix} Q_i & 0 \\ 0 & 0 \end{bmatrix}, \quad q_i = \begin{bmatrix} q_{QP,i} \\ 0 \end{bmatrix}, \quad F_i = \begin{bmatrix} 0 & \Omega \\ \hat{q}_i^T & \Psi_i \end{bmatrix} \tag{7.35}$$

$$r_i = \begin{bmatrix} -\hat{q}_i^T \\ 0 \end{bmatrix}, \quad G_i = \begin{bmatrix} -I & 0 \\ \Omega_i & 0 \\ 0 & -I \end{bmatrix}, \quad d_i = \begin{bmatrix} 0 \\ \Psi_i \\ 0 \end{bmatrix} \tag{7.36}$$

ADMM algorithm considers the problem of minimising composite objective functions subject to an equality constraint:

$$\min_{x \in X, z \in Z} f(x) + g(z)$$
$$\text{s.t. } Ax + Bz = c \tag{7.37}$$

where f and g are convex functions, $A \in \mathbb{R}^{m \times n_1}$, $B \in \mathbb{R}^{m \times n_2}$, $c \in \mathbb{R}^m$, $x \in X$, $z \in Z$, where $X \subset R^{n_1}$ and $Z \subset R^{n_2}$ are non-empty closed convex sets. The partial augmented Lagrangian of the problem is:

$$L_\rho(x, z, \mu) = f(x) + g(z) +$$
$$\mu^T (Ax + Bz - c) + \frac{\rho}{2} \|Ax + Bz - c\|_2^2 \tag{7.38}$$

where $\mu \in \mathbb{R}^m$ is the dual variable, and $\rho > 0$ is the penalty parameter. ADMM consists of the following updates:

$$x_{t+1} = \arg\min_{x \in X} L_\rho(x, z_t, \mu_t) \tag{7.39}$$
$$z_{t+1} = \arg\min_{z \in Z} L_\rho(x_{t+1}, z, \mu) \tag{7.40}$$
$$\mu_{t+1} = \mu_t + \rho(Ax_{t+1} + Bz_{t+1} - c) \tag{7.41}$$

Problem (7.34) can be recast in ADMM form by means of positive slack variables $z \geq 0$ and posing

$$f(x) = \frac{1}{2} x_i^\top P_i x_i + q_i^\top x_i \tag{7.42}$$
$$g(z) = I_{\mathcal{Z}}(z) \tag{7.43}$$
$$A = \begin{bmatrix} G_i \\ F_i \end{bmatrix}, \quad B = \begin{bmatrix} I \\ 0 \end{bmatrix}, \quad c = \begin{bmatrix} d_i \\ r_i \end{bmatrix} \tag{7.44}$$

where $I_{\mathcal{Z}}(z)$ denotes the indicator function for the set $\mathcal{Z} = \{z : z \geq 0\}$.

The classical ADMM formulation for QP, as proposed in [224], may be inefficient for embedded implementations like the one presented in our paper, since it employs non-sparse large-scale matrices. Inspired by the work in [233], which provides the formal steps, we propose the following Nesterov-accelerated ADMM algorithm for QP problems, suitable for embedded implementation.

$$x_{k+1} = \overline{q} + \overline{M}\left(\begin{bmatrix}\hat{z}_k \\ 0\end{bmatrix} + \hat{\mu}_k\right) \tag{7.45}$$

$$\alpha_{k+1} = \frac{1+\sqrt{1+4\alpha_k^2}}{2} \tag{7.46}$$

$$\hat{x}_{k+1} = x_{k+1} + \frac{\alpha_k - 1}{\alpha_{k+1}}(x_{k+1} - x_k) \tag{7.47}$$

$$z_{k+1} = \max\{0, -G\hat{x}_{k+1} - \hat{\mu}_k^G + d_i\} \tag{7.48}$$

$$\hat{z}_{k+1} = z_{k+1} + \frac{\alpha_k - 1}{\alpha_{k+1}}(z_{k+1} - z_k) \tag{7.49}$$

$$\mu_{k+1} = \hat{\mu}_k + A\hat{x}_{k+1} + \begin{bmatrix}\hat{z}_{k+1} \\ 0\end{bmatrix} - c \tag{7.50}$$

$$\hat{\mu}_{k+1} = \mu_{k+1} + \frac{\alpha_k - 1}{\alpha_{k+1}}\left(\mu_{k+1} - \mu_k\right) \tag{7.51}$$

where $\overline{M} = \rho M A^T$, $\overline{q} = Mq - \overline{M}c$, and $M = -(P_i + \rho A^T A)^{-1}$ are quantities that can be precomputed. It is important to note that, unlike classical accelerated ADMM algorithms such as [234], Nesterov acceleration is also applied in the x-update; this simple change allows faster convergence, as shown in the case study section. Another important issue is that the proposed algorithm employs sparse matrices, which allow faster computation at the price of storing also "accelerated" primal and dual variables, namely $\hat{x}, \hat{z}, \hat{\mu}$.

7.3. CASE STUDY IMPLEMENTATION AND TEST ON EMBEDDED SYSTEMS

The proposed optimization framework is implemented and tested on two embedded microcontrollers, namely the ODROID-N2L and the ODROID-H3+ (Fig. 7.3), both characterized by limited computational resources and memory capacity. The experiments aimed to evaluate the feasibility of running real-time optimization routines on low-power embedded platforms. The scheduling problem was formulated with a 5-minute discretization over a daily time horizon, employing a receding horizon approach with a one-day moving window to enhance adaptability to time-varying inputs. Data acquisition and control commands were managed through the Modbus [235], which ensured standardized communication with external sensors and actuators but introduced latency and synchronization challenges requiring robust handling strategies.

7.3.1. COMPARISON WITH STATE-OF-THE-ART ALGORITHMS

The proposed algorithm has been compared with two standard ADMM based algorithms for QPs, that is:

- Classical ADMM: the standard ADMM for QPs under the formulation proposed by [233]
- Fast ADMM: the previous algorithm but accelerated in the same way as proposed by [234]

Both algorithm formulations are provided in the appendix. The convergence plot in Fig. 7.4 shows that the proposed algorithm converges in fewer iterations than the state-of-the-art counterparts, particularly compared to classical ADMM (blue line). Specifically, 40% fewer iterations are required, and 25% fewer than with Fast ADMM are needed. To obtain a fair comparison, each algorithm has been implemented with the best theoretical ρ parameter according to [236]. Instead, regarding the convergence rate for this specific problem, since indicator functions are involved, both classical ADMM and fast ADMM exhibit an $\mathcal{O}(\frac{1}{k})$ rate. The implementation achieved numerical precision of 10^{-6} on both devices within less than 1 minute, including all data read and write operations via Modbus. These results demonstrate that, despite hardware constraints, embedded systems such as the ODROID-N2L and H3+ can effectively execute advanced optimisation routines when appropriately tailored. The study highlights the importance of lightweight algorithmic structures, efficient memory management, and optimised communication layers to ensure stable and accurate real-time performance on resource-constrained platforms.

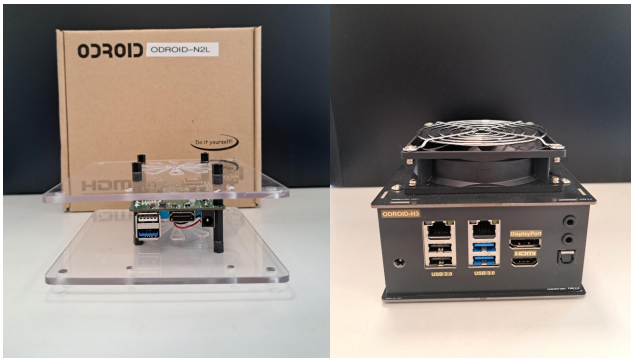


Figure 7.3: ODROID-N2L (left); ODROID-H3+ (right).

7.3.2. OPTIMAL EC SCHEDULING

A medium-size case study is tested, where an EC with 10 ECPs ($N = 10$) is considered. Table 7.1 reports the general data relevant to the ECPs. Moreover, the esteemed PV power production and the fixed power consumpion of each ECP are reported in Fig. 7.5. By analyzing data in Table 7.1 and Fig. 7.5, one can notice how generation and consumption are allocated: as an example, only ECP 4-6-9 need to charge their EVs, while ECP 2-3-6

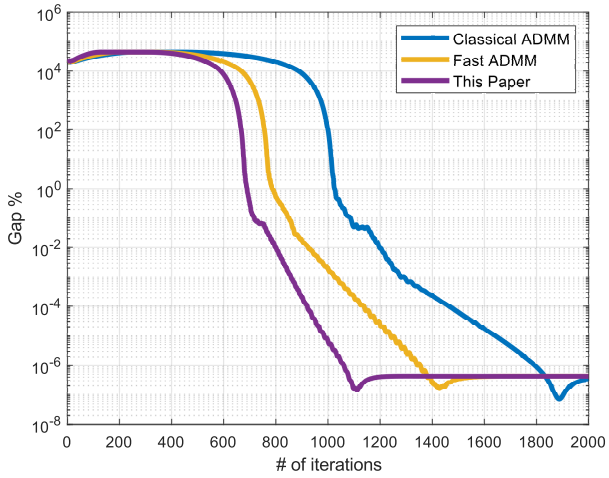


Figure 7.4: Convergence of the proposed algorithm with respect to state-of-the-art algorithms: Classical ADMM and Fast ADMM.

Table 7.1: Data of the EC Participants.

ECP_i	CAP_i^S [kWh]	$X_{i,0}^S$ [/]	$\overline{P}_i^S$ [kW]	$E_i^{L,\text{flex}}$ [kWh]	CAP_i^{EV} [kWh]	$X_{i,0}^{EV}$ [/]	$X_i^{EV^*}$ [/]	$T_i^{EV^*}$ [h]
1	/	/	/	5.14	/	/	/	/
2	15	0.4	5	3.20	/	/	/	/
3	15	0.4	5	12.76	/	/	/	/
4	/	/	/	8.31	50	0.2	0.8	15
5	/	/	/	6.56	/	/	/	/
6	15	0.4	5	5.14	50	0.3	0.9	12
7	/	/	/	3.20	/	/	/	/
8	/	/	/	12.76	/	/	/	/
9	/	/	/	6.50	50	0.35	0.8	15
10	/	/	/	13.78	/	/	/	/

are the only ones to have PV production plants and ESSs. Therefore, they are the ones that could potentially inject some power into the grid.

Further parameters considered as equal for all the users are: $\overline{P}_i^G$=20 kW, the bounds on the storages/EVs always equal to 0.2 and 1, and the charging/discharging efficiencies equal to 0.95.

In addition to the presented data, it is necessary to specify the selling and buying price for exchanging power with the main grid: C_t^{sell} is set to 0.08 [€/kWh], while the values of C_t^{buy} range between 0.53 and 0.56 [€/kWh]. Specifically, the price is 0.56 [€/kWh] from hour 1 to 8, 0.53 [€/kWh] from hour 9 to 16, and 0.54 [€/unit] from hour 17 to 24.

Fig. 7.6 illustrates the overall power exchange among all users, together with the resulting shared energy. The pink line corresponds to the definition in (7.2), representing

the minimum between the two aggregated powers at each time slot. It can be observed that, since PV systems are the sole source of generation, flexible loads have been concentrated during the central hours of the day to maximize shared energy. Further insights on the energy storage system are provided in Fig. 7.7. The figure shows that the ESS primarily acts as an additional load during daytime and is discharged during periods of higher energy prices. While this strategy would not be economically justified by price variations alone, due to charging and discharging inefficiencies, it becomes viable under the incentive mechanism for shared energy. The total shared energy amounts to 183.21 kWh. Considering an incentive value similar to the Italian one (0.108 [€/kWh]), this corresponds to a revenue of approximately 19 €. Given that the net cost of energy purchasing, after accounting for revenues from energy sales, is 120.96 €, the incentive on shared energy leads to a significant overall cost reduction of 16.36%.

7.4. CONCLUSIONS AND FUTURE DEVELOPMENTS

This work has presented a novel framework for the management and optimization of RECs, specifically tailored for embedded platforms with limited computational resources. The proposed approach addresses the dual objectives of maximizing shared energy among community participants and minimizing individual operating costs, while respecting operational constraints of energy storage systems, electric vehicles, and flexible loads. The proposed architecture allows for completely separating the optimisation problem among the ECPs without any need for variable exchange.
A compact QP formulation of the problem has been derived, enabling the deployment of an accelerated ADMM algorithm that ensures fast convergence and scalability. The

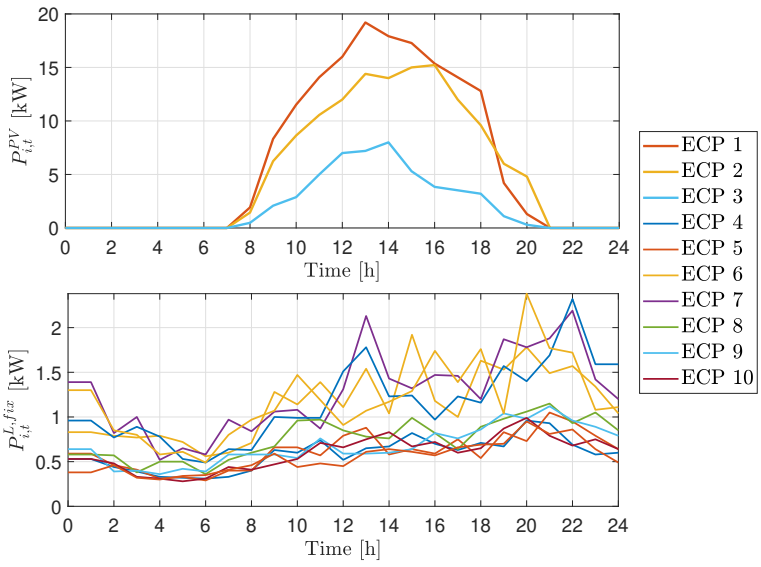


Figure 7.5: ECP profiles on PV production and consumption.

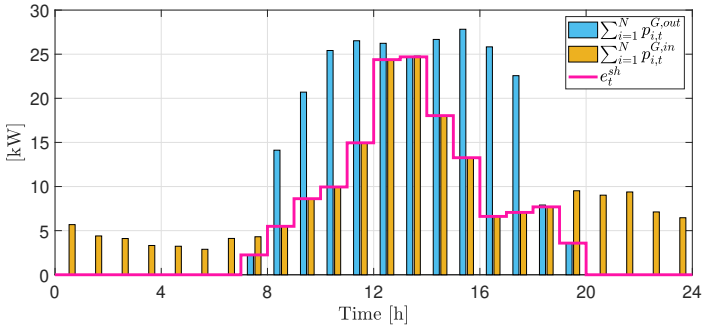


Figure 7.6: Power exchange with the grid and resulting shared energy.

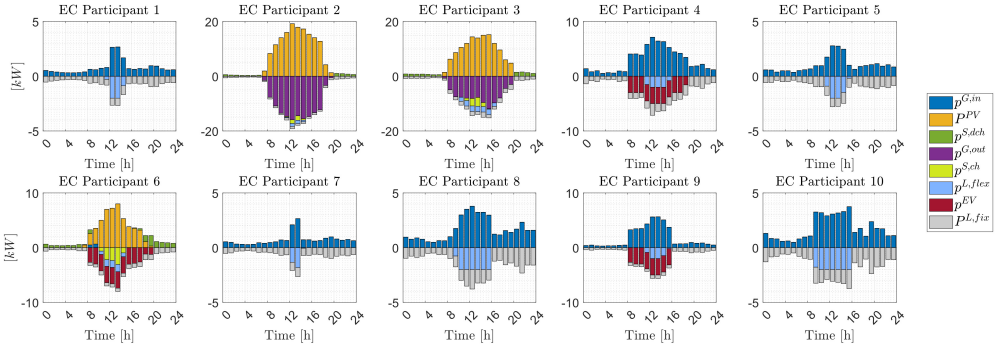


Figure 7.7: Power balance of each EC Participant.

embedded-oriented design has been validated on two low-power microcontrollers (ODROID-N2L and ODROID-H3+), demonstrating that real-time optimization of complex REC scheduling problems is feasible even on resource-constrained platforms. Numerical tests indicate that high-precision solutions can be obtained within practical time horizons, including all communication overheads, confirming the effectiveness of the proposed approach in real-world scenarios.

The case study analysis, based on a medium-sized EC with 10 participants, highlights several key outcomes. The algorithm successfully schedules energy flows to maximise shared energy while efficiently exploiting available storage and PV generation. The resulting incentive mechanism provides a tangible reduction of the overall electricity bill, demonstrating the economic benefits of coordinated operation within RECs.

Future work will focus on extending the framework to grid connected communities and integrating additional sources of flexibility as well as further remunerative policies such as demand response programs.

ALGORITHMS USED AS COMPARISON

We provide additional type algorithms for solving (34) supporting the technical as well as the numerical investigation of the paper.

CLASSICAL ADMM

$$x_{k+1} = \overline{q} + \overline{M}\left(\begin{bmatrix} z_k \\ 0 \end{bmatrix} + \mu_k\right)$$

$$z_{k+1} = \max\{0, -Gx_{k+1} - \mu_k + d_i\}$$

$$\mu_{k+1} = \mu_k + A\hat{x}_{k+1} + \begin{bmatrix} \hat{z}_{k+1} \\ 0 \end{bmatrix} - c$$

FAST ADMM

$$x_{k+1} = \overline{q} + \overline{M}\left(\begin{bmatrix} \hat{z}_k \\ 0 \end{bmatrix} + \hat{\mu}_k\right)$$

$$\alpha_{k+1} = \frac{1 + \sqrt{1 + 4\alpha_k^2}}{2}$$

$$z_{k+1} = \max\{0, -G\hat{x}_{k+1} - \hat{\mu}_k^G + d_i\}$$

$$\hat{z}_{k+1} = z_{k+1} + \frac{\alpha_k - 1}{\alpha_{k+1}}(z_{k+1} - z_k)$$

$$\mu_{k+1} = \hat{\mu}_k + A\hat{x}_{k+1} + \begin{bmatrix} \hat{z}_{k+1} \\ 0 \end{bmatrix} - c$$

$$\hat{\mu}_{k+1} = \mu_{k+1} + \frac{\alpha_k - 1}{\alpha_{k+1}}(\mu_{k+1} - \mu_k)$$

8

monviso: A Python Package for Solving Monotone VI

In this chapter, we present monviso (monotone variational inequalities solver), a novel open-source Python package for solving monotone variational inequalities. We detail the package's structure and baseline functionality, discussing a simple example that illustrates the essential methods and parameters. Moreover, we characterize how the proximal operator, which is the foundation of many iterative schemes, is handled through cvxpy, an open-source Python library for convex optimization. We list the available algorithms and describe the basic implementation of any general iterative method to enable users to build additional and (possibly new) algorithms. Finally, we illustrate several examples of possible use cases for monviso, showcasing the different applications the package can support across various fields, including control, optimization, dynamic game theory, and machine learning.

The proliferation of open-source projects that allow to easily implement control and decision-making problems is positively impacting both the academic and industrial communities. On the one hand, it allows researchers to accelerate the pace of research by providing unified frameworks for testing and benchmarking new ideas and approaches. On the other hand, it enables practitioners to solve real-world problems, often by means of quick prototyping before devising *ad-hoc* tailored implementations. Such projects serve as essential resources to democratize access to often advanced computational tools, removing the financial burden associated with proprietary software and allowing for customization and extensions aimed at satisfying specific needs. Moreover, they allow users even from outside the control and optimization community to tackle their own problems without the need of delving into technical details. Among the available mathematical tools, variational inequalities (VIs) unify many of the problems that appear in control and decision science at large, encompassing, e.g., optimal control [14], [157], optimization [237], [238], machine learning [239], game theory [240], [241], [242], and finance [15], making it a deeply studied topic of research [4]. Although the majority of existing projects strongly focus on optimization [243], and, to a lesser extent, control and dynamical systems [244], VIs have received limited attention from the perspective of systematic numerical implementations. Nevertheless, the research literature on iterative methods for solving monotone VIs keeps growing consistently. To the best of the authors' knowledge, only one package for solving VI targets Python (`imgemp/VI-Solver`), while two packages for Julia are dedicated to finite-dimensional VIs (`VariationalInequality.jl`) and complementarity problems (`Complementarity.jl`). A dedicated suite is provided by GAMS; albeit being closed source, this encompasses tools for modelling and solving VIs, quasi-VIs, and equilibrium problems. Note that we are here focusing on *modelling protocols*, i.e., the package/language that allows for declaring the problem at hand, and not *solvers*, i.e., the software that computes its solution. Except for GAMS, the surveyed pieces of software have not yet reached a mature development stage. Moreover, the number of methods and algorithms they implement for VI resolution is limited.

8

In this paper, we provide an overview of `monviso` (`monotone variational inequalities solver`), a novel Python package for solving monotone VIs. Although programming languages such as MATLAB and Julia are specifically tailored to numerical programming, Python has been chosen as the base language for `monviso` due to its ever-increasing popularity and ease of use. Essentially, `monviso` acts as a wrapper around `cvxpy` [245], one of the most used pieces of software for convex optimization, in order to simplify the process of implementing montone VIs. Differently from the aforementioned projects, `monviso` has been designed to be modular and simple to extend: the algorithms implemented in the current version, in fact, do not include all the ones existing in literature. Therefore, allowing for simply adding (possibly new) iterative methods is crucial for the package's fruibility. `monviso`, together with all the code snippets discussed in the following, is publicly available at

`github.com/nicomignoni/monviso`

Moreover, all details here reported refer to the current package version, i.e., `0.1`.

8.1. NOTATION AND PRELIMINARIES

In the sequel, set $\mathbb{R}$ and $\mathbb{N}$ represent the real and natural numbers, respectively. We denote $[\mathbf{x}_1^\top,\ldots,\mathbf{x}_N^\top]^\top =: \text{col}(\mathbf{x}_i)_{i=1}^M$ and $[\mathbf{x}_1,\ldots,\mathbf{x}_N] =: \text{row}(\mathbf{x}_i)_{i=1}^M$. The all-zeros and all-ones vectors are denoted with $\mathbf{1}$ and $\mathbf{0}$, respectively; the identity matrix of size $N\times N$ is $\mathbf{I}_N$, while vector $\mathbf{e}_{i,N}$ denotes the i-th row of $\mathbf{I}_N$. Given a square matrix $\mathbf{A}\in\mathbb{R}^{n\times n}$, its positive (negative) semidefiniteness is denoted with $\mathbf{A}\succeq 0$ ($\mathbf{A}\preceq 0$). The diagonal block matrix made by matrices $\mathbf{A}_1,\ldots,\mathbf{A}_n$ is denoted as $\text{blkdg}(\mathbf{A}_1,\ldots,\mathbf{A}_n)$. Symbols $\otimes$ and $\odot$ denote the Kronecker and Hadamard (i.e., element-wise) product, respectively. The golden ratio constant is denoted as $\varphi := \frac{1+\sqrt{5}}{2}$. The uniform distribution bounded between $a,b\in\mathbb{R}$ is denoted as $\mathscr{U}(a,b)$. Given a set $\mathscr{S}\subseteq\mathbb{R}^n$, the associated indicator function is denoted as $\iota_{\mathscr{S}}:\mathbb{R}\to\{0,+\infty\}$, so that $\iota_{\mathscr{S}}(\mathbf{x})=0$ if $\mathbf{x}\in\mathscr{S}$ and $\iota_{\mathscr{S}}(\mathbf{x})=+\infty$ otherwise. The normal cone associated with set $\mathscr{S}$ is $\mathsf{N}_{\mathscr{S}}:\mathscr{S}\rightrightarrows\mathbb{R}^n$, defined as $\mathsf{N}_{\mathscr{S}}(\mathbf{x}) := \{\mathbf{y}\in\mathbb{R}^n : \mathbf{y}^\top(\mathbf{x}-\mathbf{z})\geq 0,\ \forall\mathbf{z}\in\mathscr{S}\}$. A mapping $\mathbf{F}:\mathbb{R}^n\to\mathbb{R}^n$ is L-Lipschitz iff $\|\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{y})\|\leq L\|\mathbf{x}-\mathbf{y}\|$, for an arbitrary $L<+\infty$. Moreover, a mapping $\mathbf{F}(\cdot)$ is *monotone* if $(\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{y}))^\top(\mathbf{x}-\mathbf{y})\geq 0$ holds for all $\mathbf{x},\mathbf{y}\in\mathbb{R}^n$, and *strongly monotone* if $(\mathbf{F}(\mathbf{x})-\mathbf{F}(\mathbf{y}))^\top(\mathbf{x}-\mathbf{y})\geq\mu\|\mathbf{x}-\mathbf{y}\|^2$ holds for all $\mathbf{x},\mathbf{y}\in\mathbb{R}^n$ and some $\mu>0$. The constrained proximal operator for a given point $\mathbf{x}\in\mathbb{R}^n$ and a scalar function $g:\mathbb{R}^n\to\mathbb{R}$, with respect to set $\mathscr{S}$, is defined as

$$\text{prox}_{g,\mathscr{S}}(\mathbf{x}) = \underset{\mathbf{y}\in\mathscr{S}}{\arg\min}\left\{g(\mathbf{y})+\frac{1}{2}\|\mathbf{y}-\mathbf{x}\|^2\right\}. \tag{8.1}$$

The projection operator for point $\mathbf{x}\in\mathbb{R}^n$ with respect to set $\mathscr{S}$ can be defined as (8.1) when $g(\mathbf{y})=0$, i.e.,

$$\text{proj}_{\mathscr{S}}(\mathbf{x}) = \text{prox}_{0,\mathscr{S}}(\mathbf{x}) = \underset{\mathbf{y}\in\mathscr{S}}{\arg\min}\ \frac{1}{2}\|\mathbf{y}-\mathbf{x}\|^2. \tag{8.2}$$

Given a vector mapping $\mathbf{F}:\mathbb{R}^n\to\mathbb{R}^n$ and a scalar convex (possibly non-smooth) function $g:\mathbb{R}^n\to\mathbb{R}$, solving a VI consists of finding a point $\mathbf{x}^*\in\mathbb{R}^n$ such that the following holds:

$$\inf_{\mathbf{x}\in\mathbb{R}^n}(\mathbf{x}-\mathbf{x}^*)^\top\mathbf{F}(\mathbf{x}^*)-g(\mathbf{x})-g(\mathbf{x}^*)\geq 0. \tag{8.3}$$

The standing assumptions for (8.3) are (i) the *monotonicity* of $\mathbf{F}(\cdot)$ always holds; (ii) $g(\cdot)$ being proper convex lower-semicontinuous; the iii) the non-emptiness of the solution set of (8.3). When a VI is constrained to a set $\mathscr{S}\subset\mathbb{R}^n$, by letting $g(\mathbf{x})=\iota_{\mathscr{S}}(\mathbf{x})$, the problem in (8.3) becomes

$$\inf_{\mathbf{x}\in\mathscr{S}}(\mathbf{x}-\mathbf{x}^*)^\top\mathbf{F}(\mathbf{x}^*)\geq 0. \tag{8.4}$$

8.2. THE PACKAGE OVERVIEW

`monviso` is a novel open-source Python package for solving monotone VIs. The proposed package is a wrapper around `cvxpy` [245], which allows to conveniently define the proximal and projection operators. These operators, in fact, usually represent the most computationally intense steps of iterative schemes, being themselves fully-fledged optimization problems. Albeit some proximal operators can be explicitly expressed in analytical form, the majority requires a (usually convex) solver for retrieving the solution. A

list of *proximal-friendly* operators is reported in [246], specifically cataloging $g(\cdot)$ functions in (8.3), which yields an exact formulation for the proximity operator. Nonetheless, evaluating its analytic form, even when possible, is usually tedious and error-prone, binding the resulting iterative approach to the specific application at hand. Therefore, `monviso` follows the same philosophy of several existing optimization modelling language, which allow for a degree of expressiveness close to mathematical statements. This enables the user to input a generic form for g and $\mathscr{S}$, which will be used to automatically construct (8.1).

8.2.1. THE BASIC FUNCTIONALITY

In order to illustrate the inner working of `monviso`, let us provide a simple example as a starting point.

Example 8.2.1. *Let* $\mathbf{F}(\mathbf{x}) = \mathbf{H}\mathbf{x}$ *for some* $\mathbf{H} \succ 0$, $g(\mathbf{x}) = \|\mathbf{x}\|_1$, *and* $\mathscr{S} = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{A}\mathbf{x} \leq \mathbf{b}\}$, *for some* $\mathbf{A} \in \mathbb{R}^{m \times n}$ *and* $\mathbf{b} \in \mathbb{R}^n$. *It is straightforward to verify that* $\mathbf{F}(\cdot)$ *is strongly monotone with* $\mu = \lambda_{\min}(\mathbf{H})$ *and Lipschitz with* $L = \|\mathbf{H}\|_2$. *Therefore, the resulting VI in* (8.3) *could be solved using the well-known proximal gradient descent method [NEM83], where convergence is guaranteed for a step size* $\lambda \in \left(0, \frac{2\mu}{L^2}\right)$. *The following Python code illustrates how to solve the VI in* (8.3) *using* `monviso`.

```
import numpy as np
import cvxpy as cp

from monviso import VI

# Create the problem data
n, m = 30, 40
H = np.random.uniform(2, 10, size=(n, n))
A = np.random.uniform(45, 50, size=(m, n))
b = np.random.uniform(3, 7, size=(m,))

# Make H positive definite
H = H @ H.T

# Lipschitz and strong monotonicity constants
mu = np.linalg.eigvals(H).min()
L = np.linalg.norm(H, 2)

# Define F, g, and S
F = lambda x: H @ x
g = lambda x: cp.norm(x)
S = [lambda x: A @ x <= b]

# Define and solve the VI
vi = VI(n, F, g, S)

x0 = np.random.uniform(4, 5, n)
algorithm_params = {
    "x": x0,
    "step_size": 2 * mu / L**2
}
sol = vi.solution(
```

```
33          "pg", algorithm_params, max_iters=25,
34          eval_tol=-np.inf, log_path="result.log"
35      )
```

Except for lines 1-25, which simply define the problem parameters and needed constants, the core implementation of the VI problem is expressed by lines 27-37. In `monviso`, a VI is an instance of the class `VI()`, which takes the following objects as attributes:

- `n`: the size of the vector space n. This is necessary since `monviso` cannot infer the size of the vector space solely from `F` or `g`.
- `F`: any Python function[1] taking a `numpy` array as input, and returning another `numpy` array of the same size. It corresponds to the operator $\mathbf{F}(\cdot)$.
- `g`: a function having a single `cvxpy` `Variable` as argument and returning a scalar. Moreover, such a variable must be a 1-dimension array, i.e., a vector. This characterization corresponds to the mathematical definition of $g(\cdot)$ in (8.3). As default, `g = 0`.
- `S`: a Python `list` of callables returning a `cvxpy`'s `Constraints`. Each of them has a single `cvxpy`'s `Variable` as variable. As for `g`, such a variable must be a 1-dimension array, i.e., a vector. As default, `S = []`.

The rationale for requiring the vector space size and the variable characterizing `g` and `S` is detailed in the following. In case `g` and `S` are provided, `monviso` will check whether they are characterized by a single and identical variable vector, in order to ensure consistency with the VI definition in (8.3). An initial solution is randomly generated in line 30, while the parameters characterizing the employed algorithm are declared in line 31. The parameters to be declared depend on the chosen iterative method, although they generally comprise (at least) one starting solution and the (possibly initial) value of λ, referred to as `step_size`. The VI solution is evaluated in line 34 through the `vi.solution()` function, which takes as mandatory arguments: i) the name of the algorithm to use, ii) its parameters (i.e., the key-value map previously discussed), and iii) the maximum number of iterations. Additional optional parameters are also available:

- `eval_func`: unless the maximum number of allowed iterations is reached, iterative algorithm stop when a certain (arbitrarily small) small tolerance is reached. `monviso` allows for defining custom evaluation functions, which take the k-th iteration solution, $\mathbf{x}_k$, as single argument. As default, the *residual* magnitude $J : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is set, defined as $J(\mathbf{x}_k) := \|\mathbf{x}_k - \text{prox}_{g,\mathscr{S}}(\mathbf{x}_k - \mathbf{F}(\mathbf{x}_k))\|$
- `eval_tol`: the tolerance value for the evaluation function, under which the chosen iterative algorithm stops. As tolerance value of 1^{-9} is set as the default.
- `log_path`: the path where the algorithm log is saved. Such a log is a comma separated value file, comprising the following fields: i) `iter`, i.e., the iteration number, ii) `eval_func_value`, i.e., the value of the evaluation function, and iii) `time`, i.e., the time (in seconds) elapsed from the first to the k-th iteration.

[1] In Example 8.2.1, an anonymous function.

8.2.2. STATEFUL CONSTRAINED PROXIMAL OPERATOR

As per its definition in (8.1), the evaluation of the proximal operator corresponds to finding its minimizer. In `monviso`, this is accomplished by defining (8.1) as a `cvxpy.Problem`. Usually, optimization modelling languages that allow the user to define the problem declaratively need to parse it in order to provide the solver at hand with the accepted canonical form. This is often done through an epigraphic reformulation, which casts the constraint to a suitable conic representation. In our use case, the challenge arises because this parsing process would occur at each iteration, possibly multiple times when the proximal operator needs to be evaluated more than once (e.g., extragradient method [KOR76]). Such a repeated parsing process would dramatically worsen the overall computation time.

In order to mitigate this potential overhead, we exploited the disciplined parametrized programming (DPP) that `cvxpy` implements. DPP defines a ruleset for parsing an optimization problem in canonical form while keeping track of constant terms that might be changed from one solution instance to another. A detailed discussion of the DPP principles can be found in [247]. For our case, the term $\mathbf{x}$ is a parameter for (8.1) and (8.2): on such a basis, `monviso` implements the proximal and projection[2] operators as invariant objects upon instantiating the `VI()`. Their *statefulness* resides in the fact that the same instance is used in all the iterations for any given algorithm, with only $\mathbf{x}$ (i.e., the *state*) being updated.

Moreover, the definition of proximal operator in (8.1) includes a constraint set, differing from the usual definition where the minimization problem is defined over $\mathbb{R}^n$. The reason lies in the difficulties behind the computational instabilities that arise from implementing the indicator function. In fact, since $g(\cdot)$ might be non-smooth, one can set $g(\mathbf{x}) = \iota_{\mathscr{S}}(\mathbf{x})$ when the VI at hand is constrained by $\mathscr{S}$. However, gradient-based methods are known to be dramatically susceptible to the behavior of strongly discontinuous functions [248]. Characterizing the proximal operator as constrained counteracts such issues, with `S = []` corresponding to $\mathscr{S} = \mathbb{R}^n$.

8.2.3. THE IMPLEMENTED ALGORITHMS

Table 8.1 lists the algorithms implemented in the current version of `monviso`, along with each method's name and the required parameters. For the sake of brevity, we will not delve into the details and properties of each algorithm: the package documentation reports a brief description of each method, together with the description of the basic iterations and its convergence conditions.

Each implemented algorithm shares the same function signature, which can be summarized as follows:

```
def algorithm_name(
    x: np.ndarray,
    step_size: float,
    **other_params: dict, optional,
    **cvxpy_solve_params: dict, optional
) -> np.ndarray:
```

[2] Albeit (8.2) is a special case of (8.1), it is convenient to instantiate it as standalone operator, since it is often used for evaluations not (always) strictly related to algorithms iterates, e.g., the residual metric used for stopping the convergence.

Table 8.1: The list of implemented algorithms

Ref.	Method	Algorithm	Parameters
[NEM83]	`pg`	Proximal Gradient	$\mathbf{x}_0 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{L}\right)$
[KOR76]	`eg`	Extragradient	$\mathbf{x}_0 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{L}\right)$
[POP80]	`popov`	Popov's Method	$\mathbf{x}_0, \mathbf{y}_0 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{2L}\right)$
[TSE00]	`fbf`	Forward-Backward-Forward	$\mathbf{x}_0 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{L}\right)$
[MAL20a]	`frb`	Forward-Reflected-Backward	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{2L}\right)$
[MAL15]	`prg`	Projected Reflected Gradient	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{\sqrt{2}-1}{L}\right)$
[YOO21]	`eag`	Extra Anchored Gradient	$\mathbf{x}_0 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{\sqrt{3}L}\right)$
[CAI20]	`arg`	Accelerated Reflected Gradient	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{1}{12L}\right)$
[YOO21]	`fogda`	(Explicit) Fast OGDA	$\mathbf{x}_0, \mathbf{x}_1, \mathbf{y} \in \mathbb{R}^n$ $\lambda \in \left(0, \frac{1}{4L}\right)$, $\alpha > 2$
[SED23]	`cfogda`	Constrained Fast OGDA	$\mathbf{x}_0, \mathbf{y}_0 \in \mathbb{R}^n$, $\mathbf{x}_1 \in \mathscr{S}$ $\mathbf{z}_0 \in \mathsf{N}_{\mathscr{S}}(\mathbf{x}_1)$, $\lambda \in \left(0, \frac{1}{4L}\right)$, $\alpha > 2$
[MAL20b]	`graal`	Golden Ratio Algorithm	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda \in \left(0, \frac{\varphi}{2L}\right)$, $\phi \in (1, \varphi]$
[MAL20b]	`agraal`	Adaptive Golden Ratio Algorithm	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda_0 > 0$, $\phi \in (1, \varphi]$
[BAG26]	`hgraal_1`	Hybrid Golden Ratio Algorithm 1	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda_0 > 0$, $\phi \in (1, \varphi]$
[BAG26]	`hgraal_2`	Hybrid Golden Ratio Algorithm 2	$\mathbf{x}_0, \mathbf{x}_1 \in \mathbb{R}^n$, $\lambda_0 > 0$, $\phi \in (1, \varphi]$

```
        # (Eventual) preparatory steps
        while True:
            # Iteration steps, out of which at
            # least one generates x
            yield x
```

Each iterative algorithm is, thus, implemented as a generator, indefinitely yielding the new value of $\mathbf{x}$ at each function call. As mentioned in Example 8.2.1, each algorithm takes as a mandatory argument at least one starting point `x`, as well as the (possibly initial) value for the `step_size`. More arguments can be declared: for instance, both the Explicit and Constrained Fast Optimistic Gradient Descent-Ascent (OGDA) require a further parameter α to be set. Wherever possible, we set additional arguments with a suitable default in order to simplify the usage. The additional set of optional parameters defined by `cvxpy_solve_params` contains the arguments that can be passed to the `cxvpy.Problem.solve()` function. Specifically, since (8.1) is implemented as `cvxpy` optimization problem, as mentioned in Section 8.2.2, the user might need to pass further arguments to explicitly set `cvxpy` options on how to solve such a problem (e.g., which solver to use or parsing options). The algorithms' function signature strives to allow users to possibly implement their own iterative methods quickly and easily. In fact, one of the core concepts behind `monviso` is keeping modularity and ease of use as the package might grow in the future, anticipating future growth with the addition of more algorithms for monotone VIs.

8.3. APPLICATION EXAMPLES

In this section, we illustrate some examples related to control, optimization, game theory, and machine learning problems, which can be reduced to a VI and solved through `monviso`. For the sake of brevity, we provide a short description of the proposed examples here; further details can be found in the available repository[3].

LINEAR COMPLEMENTARITY PROBLEM [238, SECTION 3]

A common problem that can be cast to a VI is the linear complementarity problem: given $\mathbf{b} \in \mathbb{R}^n$ and $0 \preceq \mathbf{A} \in \mathbb{R}^{n \times n}$, one want to solve the following

$$\text{find } \mathbf{x} \in \mathbb{R}^n_{\geq 0} \text{ s.t. } \mathbf{y} = \mathbf{A}\mathbf{x} + \mathbf{b} \geq \mathbf{0},\ \mathbf{y}^\top \mathbf{x} = 0. \tag{8.5}$$

By setting $\mathbf{F}(\mathbf{x}) = -\mathbf{y} = -\mathbf{A}\mathbf{x} - \mathbf{b}$ and $\mathscr{S} = \mathbb{R}_{\geq 0}$ it can be readily verified that each solution for (8.4) is also a solution for (8.5). Figure 8.1a illustrates the convergence results.

TWO PLAYERS ZERO-SUM GAME [249]

Many examples of non-cooperative behavior between two adversarial agents can be modelled through zero-sum games. Let us consider vectors $\mathbf{x}_i \in \Delta_i$ as the decision variable of the i-th player, with $i \in \{1,2\}$, where $\Delta_i \subset \mathbb{R}^{n_i}$ is the simplex constraints set defined as $\Delta_i := \{\mathbf{x} \in \mathbb{R}^{n_i} : \mathbf{1}^\top \mathbf{x} = 1\}$, for all $i \in \{1,2\}$. Let $\mathbf{x} := \text{col}(\mathbf{x}_i)_{i=1}^2$. Players try to solve the following problem:

$$\min_{\mathbf{x}_1 \in \Delta_1} \max_{\mathbf{x}_2 \in \Delta_2} \Phi(\mathbf{x}_1, \mathbf{x}_2) \tag{8.6}$$

[3] github.com/nicomignoni/monviso/tree/master/examples

whose (Nash) equilibrium solution is achieved for $\mathbf{x}^*$ satisfying the following

$$\Phi(\mathbf{x}_1^*, \mathbf{x}_2) \leq \Phi(\mathbf{x}_1^*, \mathbf{x}_2^*) \leq \Phi(\mathbf{x}_1, \mathbf{x}_2^*), \quad \forall \mathbf{x} \in \Delta_1 \times \Delta_2 \tag{8.7}$$

For the sake of simplicity, we consider $\Phi(\mathbf{x}_1, \mathbf{x}_2) := \mathbf{x}_1^\top \mathbf{H}\mathbf{x}_2$, for some $\mathbf{H} \in \mathbb{R}^{n_1 \times n_2}$. In doing so, the equilibrium condition in the previous equation can be written as a VI, where $\mathscr{S} = \Delta_1 \times \Delta_2$ and the mapping $\mathbf{F} : \mathbb{R}^{n_1+n_2} \to \mathbb{R}^{n_1+n_2}$ is defined:

$$\mathbf{F}(\mathbf{x}) = \begin{bmatrix} \mathbf{H}\mathbf{x}_1 \\ -\mathbf{H}^\top \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} & \mathbf{H} \\ -\mathbf{H}^\top & \end{bmatrix} \mathbf{x} \tag{8.8}$$

The convergence results are reported in Fig. 8.1b.

FEASIBILITY PROBLEM: FINDING A POINT IN THE INTERSECTION OF M BALLS [237]

Let us consider M balls in $\mathbb{R}^n$, where the i-th ball of radius $r_i > 0$ centered in $\mathbf{c}_i \in \mathbb{R}^n$ is given by $\mathscr{B}_i(\mathbf{c}_i, r_i) \subset \mathbb{R}^n$. We are interested in finding a point belonging to their intersection, i.e., we want to solve the following:

$$\text{find } \mathbf{x} \text{ subject to } \mathbf{x} \in \bigcap_{i=1}^{M} \mathscr{B}_i(\mathbf{c}_i, r_i) \tag{8.9}$$

It is straightforward to verify that the projection of a point onto $\mathscr{B}_i(\mathbf{c}_i, r_i)$ is evaluated as:

$$\text{proj}_{\mathscr{B}_i(\mathbf{c}_i, r_i)}(\mathbf{x}) = \begin{cases} r_i \dfrac{\mathbf{x} - \mathbf{c}_i}{\|\mathbf{x} - \mathbf{c}_i\|} & \text{if } \|\mathbf{x} - \mathbf{c}_i\| > r_i \\ \mathbf{x} & \text{otherwise} \end{cases} \tag{8.10}$$

Due to the non-expansiveness of the projection in (2), one can find a solution for (1) as the fixed point of the following iterate:

$$\mathbf{x}_{k+1} = \mathsf{T}(\mathbf{x}_k) = \frac{1}{M} \sum_{i=1}^{M} \text{proj}_{\mathscr{B}_i(\mathbf{c}_i, r_i)}(\mathbf{x}_k) \tag{8.11}$$

which result from the well-known Krasnosel'skiĭ-Mann iterate. By letting $F = \mathsf{I} - \mathsf{T}$, where I denotes the identity operator, the fixed point for (3) can be treated as the canonical VI. Figure 8.1c illustrates the convergence results.

SKEW SYMMETRIC OPERATOR [133, EXAMPLE 20.35]

A simple example of a monotone operator that is not (even locally) strongly monotone is the skewed-symmetric operator, which is described as $\mathbf{F}(\mathbf{x}) = \text{blkdg}(\mathbf{A}_1, \ldots, \mathbf{A}_M)\mathbf{x}$ for some arbitrary $M \in \mathbb{N}$, where $\mathbf{A}_i = \text{triu}(\mathbf{B}_i) - \text{tril}(\mathbf{B}_i)$, for some $\mathbf{B}_i \succeq 0$, for all $i = 1, \ldots, M$. The convergence results are reported in Fig. 8.1e.

SPARSE LOGISTIC REGRESSION [239, SECTION 3]

Consider a dataset of M rows and N columns, so that $\mathbf{A} = \text{col}(\mathbf{a}_i^\top)_{i=1}^M \in \mathbb{R}^{M \times N}$ is the dataset matrix, and $\mathbf{a}_i \in \mathbb{R}^N$ is the i-th features' vector for the i-th dataset row. Moreover, let $\mathbf{b} \in \mathbb{R}^M$ be the target vector, so that $b_i \in \{-1, 1\}$ is the (binary) ground truth for

the i-th data entry. The sparse logistic regression consists of finding the weight vector $\mathbf{x} \in \mathbb{R}^N$ that minimizes the loss function $V : \mathbb{R}^n \to \mathbb{R}_{\geq 0}$, defined as follows

$$\begin{aligned} V(\mathbf{x}) &:= \sum_{i=1}^{M} \log\left(1 + \frac{1}{\exp(b_i \mathbf{a}_i^\top \mathbf{x})}\right) + \gamma \|\mathbf{x}\|_1 \\ &= \underbrace{\mathbf{1}_M^\top \log(1 + \exp(-\mathbf{b} \odot \mathbf{A}\mathbf{x}))}_{=:s(\mathbf{x})} + \underbrace{\gamma \|\mathbf{x}\|_1}_{=:g(\mathbf{x})} \end{aligned} \tag{8.12}$$

where $\gamma \in \mathbb{R}_{>0}$ is the ℓ_1-regulation strength. The gradient for $s(\cdot)$, $\nabla s_{\mathbf{x}}(\mathbf{x})$, is calculated as

$$\nabla s_{\mathbf{x}}(\mathbf{x}) = -\frac{\mathbf{A}^\top \odot (\mathbf{1}_N \otimes \mathbf{b}^\top) \odot \exp(-\mathbf{b} \odot \mathbf{A}\mathbf{x})}{1 + \exp(-\mathbf{b} \odot \mathbf{A}\mathbf{x})} \mathbf{1}_M \tag{8.13}$$

The problem of finding the minimizer for (8.12) can be cast as (8.3), with $\mathbf{F}(\mathbf{x}) := \nabla s_{\mathbf{x}}(\mathbf{x})$. The convergence trajectory is reported in Fig. 8.1d.

MARKOV DECISION PROCESS (MDP) [250, CHAPTER 3]

A stationary discrete MDP is characterized by the tuple $(\mathscr{X}, \mathscr{A}, P, r, \gamma)$, i) where $\mathscr{X}$ is the (finite countable) set of states; ii) $\mathscr{A}$ is the (finite countable) set of actions; iii) $P : \mathscr{X} \times \mathscr{A} \times \mathscr{X} \to [0,1]$ is the transition probability function, such that $P(x, a, x^+)$ is the probability of ending up in state $x^+ \in \mathscr{S}$ from state $x \in \mathscr{X}$ when taking action $a \in \mathscr{A}$; iv) $r : \mathscr{X} \times \mathscr{X} \to \mathbb{R}$ is the reward function, so that $r(x, x^+)$ returns the reward for transitioning from state $x \in \mathscr{X}$ to state $x^+ \in \mathscr{X}$; $\gamma \in \mathbb{R}_{>0}$ is a discount factor. The aim is to find a policy, i.e., a function $\pi : \mathscr{S} \to \mathscr{A}$, returning the best action for any given state. A solution concept for MDP is the *value function*, $v^\pi : \mathscr{S} \to \mathbb{R}$, defined as

$$v^\pi(x) = \overbrace{\sum_{x^+ \in \mathscr{X}} P(x, \pi(x), x^+) \left(r(x, x^+) + \gamma v(x^+)\right)}^{=: \mathsf{T}(v^\pi)} \tag{8.14}$$

returning the "goodness" of policy π. The expression in (8.14) is known as *Bellman equation*, and can be expressed as an operator of v^π, i.e., $\mathsf{T}[v^\pi(s)] =: \mathsf{T}(v^\pi)$. It can be shown that the value function yielded by the optimal policy, v^*, results from the fixed-point problem $v^* = \mathsf{T}(v^*)$. Therefore, the latter can be formulated as a canonical VI, with $F = \mathsf{I} - \mathsf{T}$. The convergence results are reported in Fig. 8.1f.

LINEAR-QUADRATIC (LQ) DYNAMIC GAMES [157]

As shown in [157, Proposition 2], the receding horizon open-loop Nash equilibria (NE) can be reformulated as a non-symmetric VI. Specifically, consider a set of agents $\mathscr{N} = \{1, \dots, N\}$ characterizing a state vector $\mathbf{x}[t] \in \mathbb{R}^n$, whose (linear) dynamics is described as

$$\mathbf{x}[t+1] = \mathbf{A}\mathbf{x}[t] + \sum_{i \in \mathscr{N}} \mathbf{B}_i \mathbf{u}_i[t] \tag{8.15}$$

for $t = 1, \dots, T$. Each agent i selfishly tries to choose $\mathbf{u}_i[t] \in \mathbb{R}^m$ in order to minimize the following cost function

$$J_i(\mathbf{u}_i | \mathbf{x}_0, \mathbf{u}_{-i}) = \frac{1}{2} \sum_{t=0}^{T-1} \|\mathbf{x}[t | \mathbf{x}_0, \mathbf{u}]\|_{\mathbf{Q}_i}^2 + \|\mathbf{u}_i[t]\|_{\mathbf{R}_i}^2 \tag{8.16}$$

for some $0 \preceq \mathbf{Q}_i \in \mathbb{R}^{n\times n}$ and $0 \prec \mathbf{R}_i \in \mathbb{R}^{m\times m}$, with $\mathbf{u}_{-i} = \text{col}(\mathbf{u}_j)_{j\in\mathcal{N}\setminus\{i\}}$ and $\mathbf{u}_j = \text{col}(\mathbf{u}_j[t])_{t=1}^{T}$. Moreover, $\mathbf{u} = \text{col}(\mathbf{u}_i)_{i\in\mathcal{N}}$. The set of feasible inputs, for each agent $i \in \mathcal{N}$, is $\mathcal{U}_i(\mathbf{x}_0, \mathbf{u}_{-i}) := \{\mathbf{u}_i \in \mathbb{R}^{mT} : \mathbf{u}_i[t] \in \mathcal{U}_i(\mathbf{u}_{-i}[t]),\ \forall t = 0,\dots,T-1;\ \mathbf{x}[t|\mathbf{x}_0,\mathbf{u}] \in \mathcal{X},\ \forall t = 1,\dots,T\}$, where $\mathcal{X} \in \mathbb{R}^n$ is the set of feasible system states. Finally, $\mathcal{U}(\mathbf{x}_0) = \{\mathbf{u} \in \mathbb{R}^{mTN} : \mathbf{u}_i \in \mathcal{U}(\mathbf{x}_0, \mathbf{u}_{-i}),\ \forall i \in \mathcal{N}\}$. Following [157, Definition 1], the sequence of input $\mathbf{u}_i^* \in \mathcal{U}_i(\mathbf{x}_0, \mathbf{u}_{-i})$, for all $i \in \mathcal{N}$, characterizes an open-loop NE iff

$$J(\mathbf{u}_i^*|\mathbf{x}_0, \mathbf{u}_{-i}^*) \leq \inf_{\mathbf{u}_i\in\mathcal{U}_i(\mathbf{x}_0,\mathbf{u}_{-i}^*)} \left\{J(\mathbf{u}_i^*|\mathbf{x}_0, \mathbf{u}_{-i})\right\} \tag{8.17}$$

which is satisfied by the fixed-point of the best response mapping of each agent, defined as

$$\mathbf{u}_i^* = \underset{\mathbf{u}_i\in\mathcal{U}(\mathbf{x}_0,\mathbf{u}_{-i}^*)}{\text{argmin}}\ J_i(\mathbf{u}_i|\mathbf{x}_0, \mathbf{u}_{-i}^*), \quad \forall i \in \mathcal{N} \tag{8.18}$$

Proposition 2 in [157] states that any solution of (8.3) is a solution for (8.18) when $\mathcal{S} = \mathcal{U}(\mathbf{x}_0)$ and $F : \mathbb{R}^{mTN} \to \mathbb{R}^{mTN}$, defined as

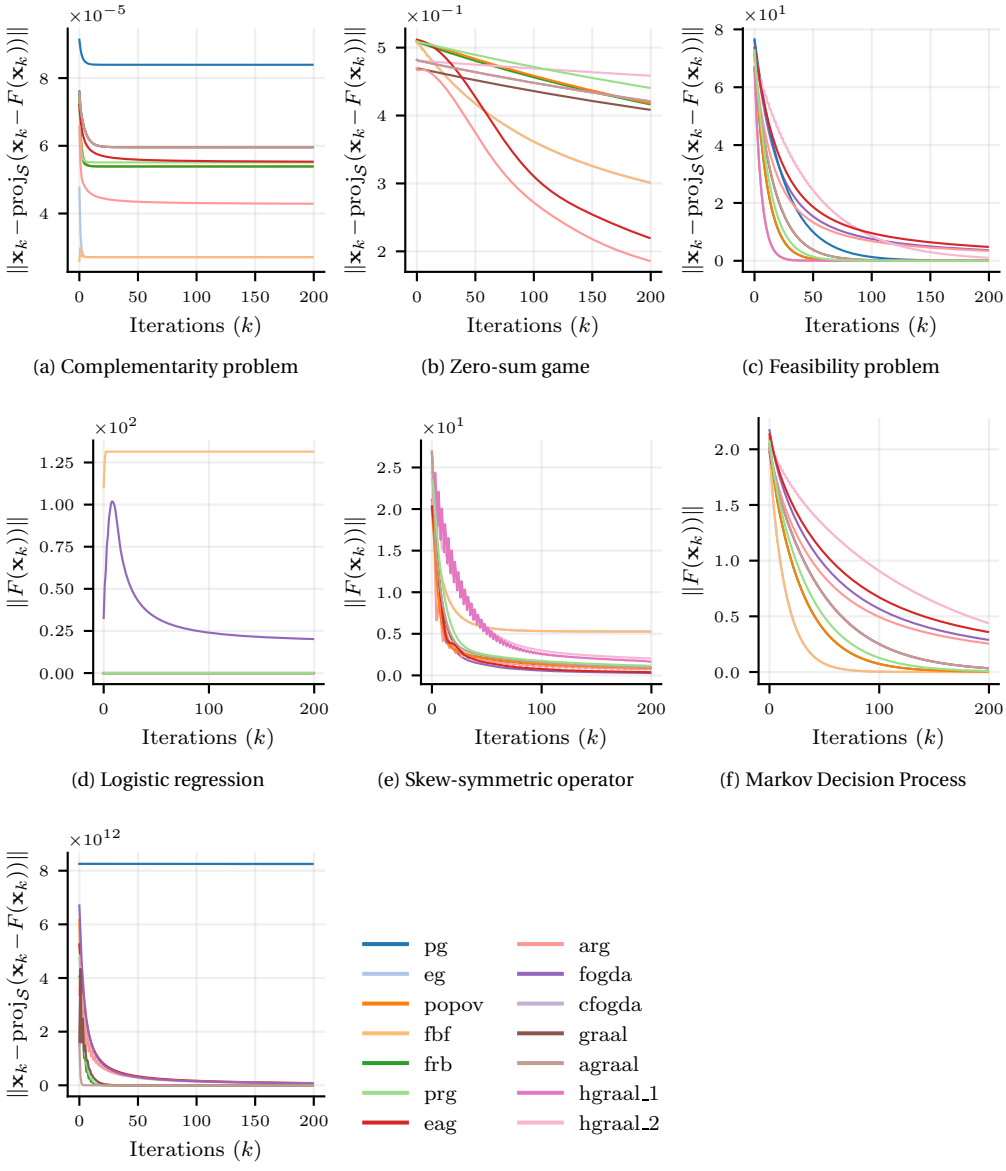
$$\begin{aligned}\mathbf{F}(\mathbf{u}) =& \text{col}(\mathbf{G}_i^\top \bar{\mathbf{Q}}_i)_{i\in\mathcal{N}}(\text{row}(\mathbf{G}_i)_{i\in\mathcal{N}}\mathbf{u} + \mathbf{H}\mathbf{x}_0) + \\ & \text{blkdg}(\mathbf{I}_T \otimes \mathbf{R}_i)_{i\in\mathcal{N}}\mathbf{u}\end{aligned} \tag{8.19}$$

where, for all $i \in \mathcal{N}$, $\bar{\mathbf{Q}}_i = \text{blkdg}(\mathbf{I}_{T-1} \otimes \mathbf{Q}_i, \mathbf{P}_i)$, $\mathbf{G}_i = \mathbf{e}_{1,T}^\top \otimes \text{col}(\mathbf{A}_i^t\mathbf{B}_i)_{t=0}^{T-1} + \mathbf{I}_T \otimes \mathbf{B}_i$ and $\mathbf{H} = \text{col}(\mathbf{A}^t)_{t=1}^{T}$. Matrix $\mathbf{P}_i$ results from the open-loop NE feedback synthesis as discussed in [157, Equation 6].

8.4. CONCLUSIONS

In this paper, we have presented `monviso`, a novel open-source Python package for solving monotone variational inequalities (VIs). We have detailed the basic functionalities of the packages through a simple introductory example and discussed how `monviso` integrates `cvxpy` to implement the proximal operator in a stateful fashion. Several examples presented the implemented algorithms, ranging from control and optimization to machine learning, showcasing `monviso` utility across different applications.

Future work will focus on improving the overall performance, possibly resorting to just-in-time compilation, as well as extending `monviso` to other commonly used programming languages, such as MATLAB and Julia.



(a) Complementarity problem

(b) Zero-sum game

(c) Feasibility problem

(d) Logistic regression

(e) Skew-symmetric operator

(f) Markov Decision Process

(g) LQ Dynamic Game

Figure 8.1: Convergence results for each example, using the currently available algorithms in `monviso`.

9

CONCLUDING REMARKS

In this thesis, we have studied convex optimization in smooth and nonsmooth settings and proposed fast solution algorithms. We further extended the analysis to variational inequalities, including algorithmic methods and applications to dynamic games and energy network systems. In this chapter, we summarize the main contributions and outlines directions for future research.

This thesis investigated the design, analysis, and application of fast first-order methods for convex optimization and equilibrium problems, with particular emphasis on adaptivity, scalability, and applicability to multi-agent and control systems. Across a broad spectrum of problem classes, ranging from smooth and nonsmooth convex optimization to variational inequalities and game-theoretic models, the common objective has been to develop algorithms that achieve strong theoretical guarantees while remaining practical for large-scale and real-world implementations. A detailed summary of the contributions is as follows.

9.1. SUMMARY

The first part of the thesis focused on adaptive algorithms for convex optimization. Chapter 2 addressed smooth and composite optimization problems and introduced a novel linesearch-based stepsize selection rule for (accelerated) first-order methods, enabling optimal convergence rates without requiring knowledge of global smoothness constants. Chapter 3 extended this line of research to fully nonsmooth convex optimization by proposing an adaptive accelerated smoothing framework, which achieves global optimal sublinear convergence and, under additional regularity, locally linear convergence. Together, these chapters demonstrate how adaptivity in stepsize selection, smoothing, and momentum can be rigorously combined to enhance both theoretical guarantees and empirical performance.

The second part of the thesis studied equilibrium computation through the lens of variational inequalities. Chapter 4 developed a projection-free algorithm for strongly monotone variational inequalities based on Frank–Wolfe oracles, targeting settings where projection operations are computationally costly. Chapter 5 considered constrained linear quadratic dynamic games, formulated them as affine variational inequalities, and proposed a tailored Douglas–Rachford splitting method with linear convergence, illustrated in an automated-driving scenario. Chapter 6 further extended the variational inequality framework to heterogeneous Wasserstein distributionally robust Nash games, showing that infinite-dimensional game formulations admit equivalent finite dimensional variational inequalities that can be solved efficiently using first-order methods, even in the presence of nonmonotonicity.

The third part of the thesis emphasized applications and implementation aspects. Chapter 7 introduced a bilevel optimization framework for decentralized energy community management and proposed an accelerated, privacy-preserving ADMM algorithm suitable for embedded platforms, with experimental validation on low-power hardware. Chapter 8 presented `monviso`, an open-source Python package for solving monotone variational inequalities, designed to support reproducible research and facilitate the deployment of equilibrium-seeking algorithms in control, optimization, and machine learning applications.

9.2. REFLECTIONS AND FUTURE RESEARCH DIRECTIONS

The work presented in this thesis opens several avenues for future research. Brief discussions of future directions are provided at the end of each chapter. Here, we highlight the main directions more comprehensively, emphasizing the potential opportunities and challenges associated with each through an illustrative example.

Illustrative example. Consider the energy network shown in Fig. 9.1, where generators are connected to multiple energy markets, and each market supplies energy to end consumers. Two classes of games can be formulated on this network.

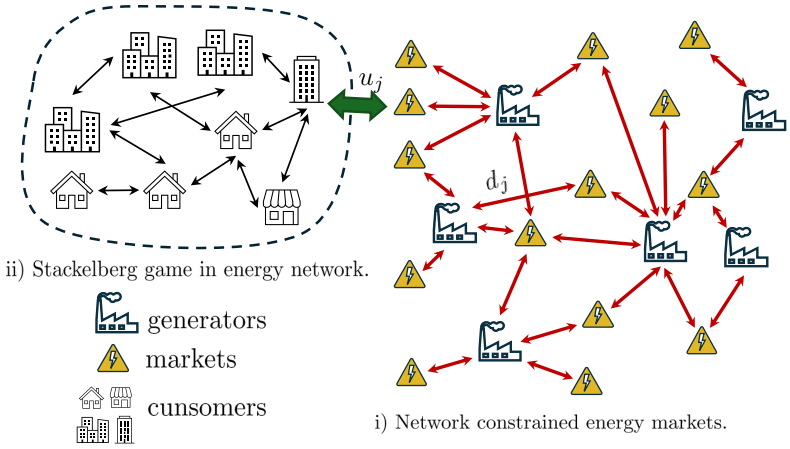


Figure 9.1: Games in energy networked systems.

i) **Cournot game [251].** Assume there are N generators and m markets. Generator i participates in n_i markets and chooses a production vector $x_i \in \Omega_i \subset \mathbb{R}^{n_i}$. The matrix $A_i \in \mathbb{R}^{m\times n_i}$ specifies the markets in which generator i operates. Each generator solves the following forward problem

$$\begin{aligned} \min_{x_i\in\Omega_i} \quad & f_i(x_i,\mathbf{x}_{-i}) := c_i(x_i) - P(A\mathbf{x})^\top A_i x_i \\ \textbf{s.t.} \quad & A\mathbf{x} \le r, \end{aligned} \tag{9.1}$$

where $A = [A_1,\dots,A_N]$, $\mathbf{x} = \mathrm{col}(x_1,\dots,x_N)$, and $r = \mathrm{col}(r_1,\dots,r_m)$ denotes the market capacity limits. The function c_i represents the generation cost of generator i. The market price vector $P(A\mathbf{x})$ depends on the aggregate supply and is defined componentwise as

$$P_j(A\mathbf{x}) = \bar{P}_j - d_j[A\mathbf{x}]_j, \quad j = 1,\dots,m, \tag{9.2}$$

where $D = \mathrm{diag}(d_1,\dots,d_m)$ is a linear inverse demand matrix.

ii) **Stackelberg game [252].** We also consider a hierarchical interaction between each market and the consumers connected to it. Each market acts as a leader by setting the energy price, while the consumers act as followers by adjusting their demand. Let $p_j \in \mathbb{R}_{\geq 0}$ denote the price set by market j, and let $y_j \in \mathbb{R}^{n_j}$ denote the demand vector of consumers connected to market j. Given p_j, the consumers solve

$$\min_{y_j \in \mathscr{Y}_j} \quad g_j(y_j, p_j) := p_j \mathbb{1}^\top y_j - u_j(y_j), \tag{9.3}$$

where u_j is a utility function and $\mathscr{Y}_j$ is the feasible demand set. Let $y_j^\star(p_j)$ denote the consumers' optimal response. Anticipating the followers' behavior, market j solves the leader problem

$$\begin{aligned} \min_{p_j \geq 0} \quad & h_j(p_j) := -p_j \mathbb{1}^\top y_j^\star(p_j \\ \textbf{s.t.} \quad & \mathbb{1}^\top y_j^\star(p_j) \leq r_j, \end{aligned} \tag{9.4}$$

where r_j denotes the available energy supply of market j. A Stackelberg equilibrium is defined as a price–demand pair $(p_j^\star, y_j^\star)$ such that $y_j^\star = y_j^\star(p_j^\star)$ and $p_j^\star$ solves the leader problem.

Based on these two problems, we are now in a position to identify the potential research directions that can be undertaken in system and control, game theory, scheduling, and management science.

ı) **Inverse Cournot game:** While the Cournot game seeks the equilibrium in (9.1), the model inputs are often difficult to estimate, even though equilibrium outcomes are typically observable. This motivates the use of inverse optimization in equilibrium settings [17], [18], [20], where observed equilibria can be exploited to infer underlying model parameters. In particular, collected equilibria in (9.1) can be used to:

(a) learn the topology/graph of the energy network (red lines in Fig. 9.1 or A_i),

(b) estimate the market demand parameters d_j in (9.2).

ıı) **Online Stackelberg game [253]:** In the Stackelberg game, the goal is to find the price demand pair $(p_j^\star, y_j^\star)$ in (9.3)–(9.4). However, iterative algorithms are often slow because the leader does not know the followers' parameters or utility functions (u_j) and only observes their responses (y_j). A promising approach to improve efficiency and convergence is to estimate the followers' parameters and utility functions from collected data, enabling the leader to solve an approximate problem centrally within a local region.

BIBLIOGRAPHY

[1] N. Pustelnik and L. Condat, "Proximity operator of a sum of functions; application to depth map estimation", *IEEE Signal Processing Letters*, vol. 24, no. 12, pp. 1827–1831, 2017.

[2] F. Abboud, E. Chouzenoux, J.-C. Pesquet, J.-H. Chenot, and L. Laborelli, "Dual block-coordinate forward-backward algorithm with application to deconvolution and deinterlacing of video sequences", *Journal of Mathematical Imaging and Vision*, vol. 59, pp. 415–431, 2017.

[3] Y.-L. Yu, "On decomposing the proximal map", *Advances in Neural Information Processing Systems*, 2013.

[4] F. Facchinei and J.-S. Pang, *Finite-dimensional variational inequalities and complementarity problems*. Springer, 2003.

[5] F. E. Browder, "A new generalization of the schauder fixed point theorem", *Mathematische Annalen*, vol. 174, no. 4, pp. 285–290, 1967.

[6] R. Rahimi Baghbadorani, S. Grammatico, and P. Mohajerin Esfahani, "Adaptive accelerated composite minimization", *preprint available at arXiv:2405.03414*, 2024.

[7] R. Rahimi Baghbadorani, S. Grammatico, and P. Mohajerin Esfahani, "Locally linear convergence for nonsmooth convex optimization via coupled smoothing and momentum", *preprint available at arXiv:2511.10239*, 2025.

[8] R. Rahimi Baghbadorani, P. Mohajerin Esfahani, and S. Grammatico, "A frank-wolfe algorithm for strongly monotone variational inequalities", *Operations Research Letters*, p. 107 388, 2025.

[9] Y. Nesterov, L. Scrimali, et al., "Solving strongly monotone variational and quasi-variational inequalities", CORE, Tech. Rep., 2006.

[10] G. Gidel, T. Jebara, and S. Lacoste-Julien, "Frank-Wolfe Algorithms for Saddle Point Problems", in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, A. Singh and J. Zhu, Eds., ser. Proceedings of Machine Learning Research, vol. 54, PMLR, 2017, pp. 362–371.

[11] D. P. Bertsekas, "A class of optimal routing algorithms for communication networks", *NASA STI/Recon Technical Report N*, vol. 81, p. 11 299, 1980.

[12] M. C. Bliemer and P. H. Bovy, "Quasi-variational inequality formulation of the multiclass dynamic traffic assignment problem", *Transportation Research Part B: Methodological*, vol. 37, no. 6, pp. 501–519, 2003.

[13] D. P. Bertsekas and E. M. Gafni, "Projection methods for variational inequalities with application to the traffic assignment problem", in *Nondifferential and Variational Techniques in Optimization*, 1982, pp. 139–159.

[14] R. Rahimi Baghbadorani, E. Benenati, and S. Grammatico, "A douglas-rachford splitting method for solving monotone variational inequalities in linear-quadratic dynamic games", *preprint available at arXiv:2504.05757*, 2025.

[15] G. Pantazis, R. R. Bahbadorani, and S. Grammatico, "Nash equilibrium seeking for a class of quadratic-bilinear wasserstein distributionally robust games", *preprint available at arXiv:2411.09636*, 2024.

[16] G. Ferro, S. Grammatico, L. Parodi, R. R. Baghbadorani, and M. Robba, "An embedded accelerated decentralized optimization algorithm with application to energy communities", *Control Engineering Practice*, vol. 172, p. 106 920, 2026.

[17] D. Bertsimas, V. Gupta, and I. C. Paschalidis, "Data-driven estimation in equilibrium using inverse optimization", *Mathematical Programming*, vol. 153, no. 2, pp. 595–633, 2015.

[18] P. Mohajerin Esfahani, S. Shafieezadeh-Abadeh, G. A. Hanasusanto, and D. Kuhn, "Data-driven inverse optimization with imperfect information", *Mathematical Programming*, vol. 167, no. 1, pp. 191–234, 2018.

[19] P. Zattoni Scroccaro, B. Atasoy, and P. Mohajerin Esfahani, "Learning in inverse optimization: Incenter cost, augmented suboptimality loss, and algorithms", *Operations Research*, vol. 73, no. 5, pp. 2661–2679, 2025.

[20] K. Ren, P. Mohajerin Esfahani, and A. Georghiou, "Inverse optimization via learning feasible regions", *preprint available at arXiv:2505.15025*, 2025.

[21] P. D. Grontas, G. Belgioioso, C. Cenedese, M. Fochesato, J. Lygeros, and F. Dörfler, "Big hype: Best intervention in games via distributed hypergradient descent", *IEEE Transactions on Automatic Control*, vol. 69, no. 12, pp. 8338–8353, 2024.

[22] C. Maheshwari, J. Cheng, S. Sastry, L. Ratliff, and E. Mazumdar, "Follower agnostic learning in stackelberg games", in *2024 IEEE 63rd Conference on Decision and Control (CDC)*, IEEE, 2024, pp. 222–228.

[23] H. Im and P. Grigas, "Stochastic first-order algorithms for constrained distributionally robust optimization", *INFORMS Journal on Computing*, vol. 37, no. 2, pp. 212–229, 2025.

[24] Y. Xu, "First-order methods for constrained convex programming based on linearized augmented lagrangian function", *INFORMS Journal on Optimization*, vol. 3, no. 1, pp. 89–117, 2021.

[25] N. Ho-Nguyen and F. Kılınç-Karzan, "Online first-order framework for robust convex optimization", *Operations Research*, vol. 66, no. 6, pp. 1670–1692, 2018.

[26] R. Battiti, "First- and second-order methods for learning: Between steepest descent and newton's method", *Neural Computation*, vol. 4, 2 1992, ISSN: 0899-7667. DOI: 10.1162/neco.1992.4.2.141.

[27] G. Lan, *First-order and stochastic optimization methods for machine learning*. Springer, 2020, vol. 1.

[28] M. Hardt, T. Ma, and B. Recht, "Gradient descent learns linear dynamical systems", *Journal of Machine Learning Research*, vol. 19, no. 29, pp. 1–44, 2018.

9

[29] M. Fazel, T. K. Pong, D. Sun, and P. Tseng, "Hankel matrix rank minimization with applications to system identification and realization", *SIAM Journal on Matrix Analysis and Applications*, vol. 34, no. 3, pp. 946–977, 2013.

[30] B. T. Polyak, "Minimization of unsmooth functionals", *USSR Computational Mathematics and Mathematical Physics*, vol. 9, 3 1969, ISSN: 00415553. DOI: 10.1016/0041-5553(69)90061-5.

[31] Y. Malitsky and K. Mishchenko, "Adaptive gradient descent without descent", in *Proceedings of the 37th International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 119, 2020, pp. 6702–6712. [Online]. Available: https://proceedings.mlr.press/v119/malitsky20a.html.

[32] J. Nocedal and S. J. Wright, *Numerical Optimization*. Springer, 2006.

[33] Y. Drori and M. Teboulle, "Performance of first-order methods for smooth convex minimization: A novel approach", *Mathematical Programming*, vol. 145, 1-2 2014, ISSN: 14364646. DOI: 10.1007/s10107-013-0653-0.

[34] L. Armijo, "Minimization of functions having lipschitz continuous first partial derivatives", *Pacific Journal of Mathematics*, vol. 16, 1 1966.

[35] B. Poliak, *Introduction to Optimization* (Translations series in mathematics and engineering). Optimization Software, Publications Division, 1987, ISBN: 9780911575149.

[36] M. Barré and A. d'Aspremont, "Polyak steps for adaptive fast gradient method", *preprint available at arXiv:1906.03056*, 2019.

[37] M. Barré, A. Taylor, and A. d'Aspremont, "Complexity guarantees for polyak steps with momentum", in *Conference on learning theory*, 2020, pp. 452–478.

[38] E. Hazan and S. Kakade, "Revisiting the polyak step size", *preprint available at arXiv:1905.00313*, 2019.

[39] Y. Nesterov, "A method of solving a convex programming problem with convergence rate $\mathcal{O}(1/k^2)$", *Soviet Mathematics Doklady*, vol. 27, 2 1983.

[40] A. S. Nemirovskij and D. B. Yudin, "Problem complexity and method efficiency in optimization", 1983.

[41] A. Beck and M. Teboulle, "Mirror descent and nonlinear projected subgradient methods for convex optimization", *Operations Research Letters*, vol. 31, no. 3, pp. 167–175, 2003.

[42] A. Yurtsever, O. Fercoq, F. Locatello, and V. Cevher, "A conditional gradient framework for composite convex minimization with applications to semidefinite programming", in *International Conference on Machine Learning*, 2018, pp. 5727–5736.

[43] T. Yang and Q. Lin, "Rsg: Beating subgradient method without smoothness and strong convexity", *The Journal of Machine Learning Research*, vol. 19, no. 1, pp. 236–268, 2018.

[44] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems", *SIAM Journal on Imaging Sciences*, vol. 2, 1 2009.

[45] M.-L. Vladarean, Y. Malitsky, and V. Cevher, "A first-order primal-dual method with adaptivity to local smoothness", *Advances in Neural Information Processing Systems*, vol. 34, pp. 6171–6182, 2021.

[46] Y. Malitsky, "Golden ratio algorithms for variational inequalities", *Mathematical Programming*, vol. 184, no. 1-2, pp. 383–410, 2020.

[47] T. Li and G. Lan, "A simple uniformly optimal method without line search for convex optimization", *preprint available at arXiv:2310.10082*, 2023.

[48] S. Bubeck, Y. T. Lee, and M. Singh, "A geometric alternative to Nesterov's accelerated gradient descent", *preprint available at arXiv:1506.08187*, 2015.

[49] Y. Nesterov et al., *Lectures on Convex Optimization*. Springer, 2018, vol. 137.

[50] S. Bubeck et al., "Convex optimization: Algorithms and complexity", *Foundations and Trends in Machine Learning*, vol. 8, no. 3-4, pp. 231–357, 2015.

[51] G. Jiang, L. J. Hong, and B. L. Nelson, "Online risk monitoring using offline simulation", *INFORMS Journal on Computing*, vol. 32, no. 2, pp. 356–375, 2020.

[52] Y. Huang, W. Zhang, H. Li, D. Ge, H. Liu, and Y. Ye, "A restarted primal-dual hybrid conjugate gradient method for large-scale quadratic programming", *INFORMS Journal on Computing*, 2025.

[53] A. Mittal, C. Gokalp, and G. A. Hanasusanto, "Robust quadratic programming with mixed-integer uncertainty", *INFORMS Journal on Computing*, vol. 32, no. 2, pp. 201–218, 2020.

[54] S. Mehrotra and J. Sun, "An algorithm for convex quadratic programming that requires $\mathcal{O}(n^3.5L)$ arithmetic operations", *Mathematics of Operations Research*, vol. 15, no. 2, pp. 342–363, 1990.

[55] G. FENG, X. LI, and Z. WANG, "Technical note—on the relation between several discrete choice models.(2017)", *Operations Research*, vol. 65, no. 6, pp. 1429–1731,

[56] Z. Qu, A. Galichon, W. Gao, and J. Ugander, "On sinkhorn's algorithm and choice modeling", *Operations Research*, 2025.

[57] N. Liu, Y. Ma, and H. Topaloglu, "Assortment optimization under the multinomial logit model with sequential offerings", *INFORMS Journal on Computing*, vol. 32, no. 3, pp. 835–853, 2020.

[58] S. Boyd and L. Vandenberghe, "Semidefinite programming relaxations of nonconvex problems in control and combinatorial optimization", in *Communications, Computation, Control, and Signal Processing: a tribute to Thomas Kailath*, Springer, 1997, pp. 279–287.

[59] Y. Nesterov, "Smoothing technique and its applications in semidefinite optimization", *Mathematical Programming*, vol. 110, no. 2, pp. 245–259, 2007.

[60] Q. Han et al., "A low-rank ADMM splitting approach for semidefinite programming", *INFORMS Journal on Computing*, 2025.

9

[61] R. Rahimi Baghbadorani, P. Mohajerin Esfahani, and S. Grammatico, "Fast adaptive first-order semidefinite programming for data-driven linear quadratic regulation", in *2024 32nd Mediterranean Conference on Control and Automation (MED)*, IEEE, 2024, pp. 771–776.

[62] M. Neykov, J. S. Liu, and T. Cai, "ℓ_1-regularized least squares for support recovery of high dimensional single index models with gaussian designs", *Journal of Machine Learning Research*, vol. 17, no. 87, pp. 1–37, 2016.

[63] I. Selesnick, "Sparse regularization via convex analysis", *IEEE Transactions on Signal Processing*, vol. 65, no. 17, pp. 4481–4494, 2017.

[64] B. R. Gaines, J. Kim, and H. Zhou, "Algorithms for fitting the constrained lasso", *Journal of Computational and Graphical Statistics*, vol. 27, no. 4, pp. 861–871, 2018.

[65] D. Bertsimas and N. Mundru, "Sparse convex regression", *INFORMS Journal on Computing*, vol. 33, no. 1, pp. 262–279, 2021.

[66] Y. Nesterov and B. T. Polyak, "Cubic regularization of newton method and its global performance", *Mathematical Programming*, vol. 108, 1 2006, ISSN: 00255610. DOI: 10.1007/s10107-006-0706-8.

[67] X. Chen, B. Jiang, T. Lin, and S. Zhang, "Accelerating adaptive cubic regularization of newton's method via random sampling", *Journal of Machine Learning Research*, vol. 23, no. 90, pp. 1–38, 2022.

[68] S. Boyd, L. E. Ghaoui, E. Feron, and V. Balakrishnan, *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics, 1994, ISBN: 978-0-89871-485-2. DOI: 10.1137/1.9781611970777.

[69] T. Tang and K.-C. Toh, "Solving graph equipartition sdps on an algebraic variety", *Mathematical Programming*, vol. 204, no. 1, pp. 299–347, 2024.

[70] P. Biswas, T.-C. Liang, K.-C. Toh, Y. Ye, and T.-C. Wang, "Semidefinite programming approaches for sensor network localization with noisy distance measurements", *IEEE transactions on automation science and engineering*, vol. 3, no. 4, pp. 360–371, 2006.

[71] C. Helmberg and F. Rendl, "A spectral bundle method for semidefinite programming", *SIAM Journal on Optimization*, vol. 10, no. 3, pp. 673–696, 2000.

[72] A. Beck, *First-order Methods in Optimization*. SIAM, 2017.

[73] Y. Nesterov, "Accelerating the cubic regularization of newton's method on convex problems", *Mathematical Programming*, vol. 112, no. 1, pp. 159–181, 2008.

[74] A. Mokhtari, A. E. Ozdaglar, and S. Pattathil, "Convergence rate of $\mathcal{O}(1/k)$ for optimistic gradient and extragradient methods in smooth convex-concave saddle point problems", *SIAM Journal on Optimization*, vol. 30, no. 4, pp. 3230–3251, 2020.

[75] T. Yoon and E. K. Ryu, "Accelerated algorithms for smooth convex-concave minimax problems with $\mathcal{O}(1/k^2)$ rate on squared gradient norm", in *International Conference on Machine Learning*, 2021, pp. 12 098–12 109.

[76] R. I. Boţ, E. R. Csetnek, and M. Sedlmayer, "An accelerated minimax algorithm for convex-concave saddle point problems with nonsmooth coupling function", *Computational Optimization and Applications*, pp. 1–42, 2022.

[77] S. Lacoste-Julien, M. Schmidt, and F. Bach, "A simpler approach to obtaining an $mathcalO(1/t)$ convergence rate for the projected stochastic subgradient method", *preprint available at arXiv:1212.2002*, 2012.

[78] A. Nemirovski, "Prox-method with rate of convergence $\mathcal{O}(1/t)$ for variational inequalities with lipschitz continuous monotone operators and smooth convex-concave saddle point problems", *SIAM Journal on Optimization*, vol. 15, no. 1, pp. 229–251, 2004.

[79] A. Juditsky, A. Nemirovski, and C. Tauvel, "Solving variational inequalities with stochastic mirror-prox algorithm", *Stochastic Systems*, vol. 1, no. 1, pp. 17–58, 2011.

[80] Y. Malitsky and M. K. Tam, "A forward-backward splitting method for monotone inclusions without cocoercivity", *SIAM Journal on Optimization*, vol. 30, no. 2, pp. 1451–1472, 2020.

[81] C. Jin, P. Netrapalli, and M. I. Jordan, "Accelerated gradient descent escapes saddle points faster than gradient descent", in *Conference On Learning Theory*, 2018, pp. 1042–1085.

[82] A. Alacaoglu, Y. Malitsky, and V. Cevher, "Convergence of adaptive algorithms for constrained weakly convex optimization", *Advances in Neural Information Processing Systems*, vol. 34, pp. 14 214–14 225, 2021.

[83] B. O'Donoghue and E. Candès, "Adaptive restart for accelerated gradient schemes", *Foundations of Computational Mathematics*, vol. 15, 3 2015, ISSN: 16153383. DOI: 10.1007/s10208-013-9150-3.

[84] D. Kim and J. A. Fessler, "Adaptive restart of the optimized gradient method for convex optimization", *Journal of Optimization Theory and Applications*, vol. 178, 1 2018, ISSN: 15732878. DOI: 10.1007/s10957-018-1287-4.

[85] J. Liang, J. Fadili, and G. Peyré, "Activity identification and local linear convergence of forward–backward-type methods", *SIAM Journal on Optimization*, vol. 27, no. 1, pp. 408–437, 2017.

[86] M. Annergren, A. Hansson, and B. Wahlberg, "An admm algorithm for solving ℓ_1 regularized mpc", in *IEEE Conference on Decision and Control*, 2012.

[87] S. K. Pakazad, H. Ohlsson, and L. Ljung, "Sparse control using sum-of-norms regularized model predictive control", in *52nd IEEE Conference on Decision and Control*, IEEE, 2013, pp. 5758–5763.

[88] J. Noom, O. Soloviev, and M. Verhaegen, "Proximal-based recursive implementation for model-free data-driven fault diagnosis", *Automatica*, vol. 165, p. 111 656, 2024.

[89] H. Fu, M. K. Ng, M. Nikolova, and J. L. Barlow, "Efficient minimization methods of mixed l2-l1 and l1-l1 norms for image restoration", *SIAM Journal on Scientific computing*, vol. 27, no. 6, pp. 1881–1902, 2006.

[90] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging", *Journal of Mathematical Imaging and Vision*, vol. 40, pp. 120–145, 2011.

[91] S. Wang et al., "Dictionary learning based impulse noise removal via l1–l1 minimization", *Signal Processing*, vol. 93, no. 9, pp. 2696–2708, 2013.

[92] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1-problems in compressive sensing", *SIAM Journal on Scientific Computing*, vol. 33, no. 1, pp. 250–278, 2011.

[93] X.-B. Hu and W.-H. Chen, "Model predictive control for constrained systems with uncertain state-delays", *International Journal of Robust and Nonlinear Control*, vol. 14, no. 17, pp. 1421–1432, 2004.

[94] Y. Nesterov, "Gradient methods for minimizing composite functions", *Mathematical Programming*, vol. 140, no. 1, pp. 125–161, 2013.

[95] H. Schramm and J. Zowe, "A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results", *SIAM Journal on Optimization*, vol. 2, no. 1, pp. 121–152, 1992.

[96] Y. Nesterov, "Smooth minimization of non-smooth functions", *Mathematical Programming*, vol. 103, pp. 127–152, 2005.

[97] B. S. B. J. Nesta, *A fast and accurate first-order method for sparse recovery*, 2009.

[98] Y. Nesterov, "Excessive gap technique in nonsmooth convex minimization", *SIAM Journal on Optimization*, vol. 16, no. 1, 2005.

[99] A. dAspremont, L. Ghaoui, M. Jordan, and G. Lanckriet, "A direct formulation for sparse pca using semidefinite programming", *Advances in Neural Information Processing Systems*, 2004.

[100] A. Beck and M. Teboulle, "Smoothing and first order methods: A unified framework", *SIAM Journal on Optimization*, vol. 22, no. 2, pp. 557–580, 2012.

[101] A. dAspremont and N. Karoui, "A stochastic smoothing algorithm for semidefinite programming", *SIAM Journal on Optimization*, vol. 24, no. 3, pp. 1138–1177, 2014.

[102] A. Yurtsever, O. Fercoq, and V. Cevher, "A conditional-gradient-based augmented lagrangian framework", *ICML*, 2019.

[103] F. Orabona, A. Argyriou, and N. Srebro, "Prisma: Proximal iterative smoothing algorithm", *preprint available at arXiv:1206.2372*, 2012.

[104] R. I. Boţ and C. Hendrich, "A variable smoothing algorithm for solving convex optimization problems", *An Official Journal of the Spanish Society of Statistics and Operations Research*, vol. 23, pp. 124–150, 2015.

[105] Q. Tran-Dinh, "Adaptive smoothing algorithms for nonsmooth composite convex minimization", *Computational Optimization and Applications*, vol. 66, pp. 425–451, 2017.

[106] C. Chaux, J.-C. Pesquet, and N. Pustelnik, "Nested iterative algorithms for convex constrained image recovery problems", *SIAM Journal on Imaging Sciences*, vol. 2, no. 2, pp. 730–762, 2009.

[107] R. A. Poliquin and R. T. Rockafellar, "Generalized hessian properties of regularized nonsmooth functions", *SIAM Journal on Optimization*, vol. 6, no. 4, pp. 1121–1137, 1996.

[108] B. Grimmer, H. Lu, P. Worah, and V. Mirrokni, "The landscape of the proximal point method for nonconvex–nonconcave minimax optimization", *Mathematical Programming*, vol. 201, no. 1, pp. 373–407, 2023.

[109] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the polyak-łojasiewicz condition", in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*, Springer, 2016, pp. 795–811.

[110] Y. Bello-Cruz, G. Li, and T. T. A. Nghia, "Quadratic growth conditions and uniqueness of optimal solution to lasso", *Journal of Optimization Theory and Applications*, vol. 194, no. 1, pp. 167–190, 2022.

[111] B. Polyak, *Introduction to Optimization*. New York, NY, USA: Optimization Software, 1987.

[112] C. Thrampoulidis, E. Abbasi, and B. Hassibi, "Lasso with non-linear measurements is equivalent to one with linear measurements", *Advances in Neural Information Processing Systems*, 2015.

[113] M. F. Sahin, A. Eftekhari, A. Alacaoglu, F. Latorre, and V. Cevher, "An inexact augmented lagrangian framework for nonconvex optimization with nonlinear constraints", *Advances in Neural Information Processing Systems*, 2019.

[114] H. Avron, S. Kale, S. Kasiviswanathan, and V. Sindhwani, "Efficient and practical stochastic subgradient descent for nuclear norm regularization", *preprint available at arXiv:1206.6384*, 2012.

[115] J. Liang, J. M. Fadili, and G. Peyré, "Local linear convergence of forward–backward under partial smoothness", *Advances in neural information processing systems*, vol. 27, 2014.

[116] P. J. Bickel, Y. Ritov, and A. B. Tsybakov, "Simultaneous analysis of lasso and dantzig selector", *The Annals of Statistic*, vol. 37, no. 4, pp. 1705–1732, 2009.

[117] M. Jaggi, M. Sulovsk, et al., "A simple algorithm for nuclear norm regularized problems", in *Proceedings of the 27th International Conference on Machine Learning*, 2010.

[118] N. Srebro, J. Rennie, and T. Jaakkola, "Maximum-margin matrix factorization", *Advances in Neural Information Processing Systems*, 2004.

[119] Y. Guo, "Robust transfer principal component analysis with rank constraints", *Advances in Neural Information Processing Systems*, 2013.

[120] Y. Sun, S. Oymak, and M. Fazel, "System identification via nuclear norm regularization", *preprint available at arXiv:2203.16673*, 2022.

[121] K. H. Johansson, A. Horch, O. Wijk, and A. Hansson, "Teaching multivariable control using the quadruple-tank process", in *38th IEEE Conference on Decision and Control*, IEEE, vol. 1, 1999.

[122] C.-C. Chang and C.-J. Lin, "Libsvm: A library for support vector machines", *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, pp. 1–27, 2011.

[123] C. Chen, B. He, Y. Ye, and X. Yuan, "The direct extension of admm for multi-block convex minimization problems is not necessarily convergent", *Mathematical Programming*, vol. 155, no. 1, pp. 57–79, 2016.

[124] J. A. Ball, J. Chudoung, and M. V. Day, "Robust optimal switching control for nonlinear systems", *SIAM Journal on Control and Optimization*, vol. 41, no. 3, pp. 900–931, 2002.

[125] G. Scutari, D. P. Palomar, F. Facchinei, and J.-S. Pang, "Convex optimization, game theory, and variational inequality theory", *IEEE Signal Processing Magazine*, vol. 27, no. 3, pp. 35–49, 2010.

[126] M. Hu and M. Fukushima, "Variational inequality formulation of a class of multi-leader-follower games", *Journal of optimization theory and applications*, vol. 151, pp. 455–473, 2011.

[127] E. Allevi, A. Gnudi, I. V. Konnov, and G. Oggioni, "Evaluating the effects of environmental regulations on a closed-loop supply chain network: A variational inequality approach", *Annals of Operations Research*, vol. 261, no. 1, pp. 1–43, 2018.

[128] G. Gidel, H. Berard, G. Vignoud, P. Vincent, and S. Lacoste-Julien, "A variational inequality perspective on generative adversarial networks", *International Conference on Learning Representations (ICLR)*, 2019.

[129] P. Harker, "A variational inequality approach for the determination of oligopolistic market equilibrium", *Mathematical Programming*, vol. 30, pp. 105–111, 1984.

[130] J.-C. Yao, "The generalized quasi-variational inequality problem with applications", *Journal of Mathematical Analysis and Applications*, vol. 158, no. 1, pp. 139–160, 1991.

[131] Y. V. Malitsky and V. Semenov, "An extragradient algorithm for monotone variational inequalities", *Cybernetics and Systems Analysis*, vol. 50, no. 2, pp. 271–277, 2014.

[132] Y. Malitsky, "Projected reflected gradient methods for monotone variational inequalities", *SIAM Journal on Optimization*, vol. 25, no. 1, pp. 502–520, 2015.

[133] H. H. Bauschke, P. L. Combettes, H. H. Bauschke, and P. L. Combettes, *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer, 2017.

[134] P. Giselsson, "Tight global linear convergence rate bounds for douglas–rachford splitting", *Journal of Fixed Point Theory and Applications*, vol. 19, no. 4, pp. 2241–2270, 2017.

[135] W. M. Moursi and L. Vandenberghe, "Douglas–rachford splitting for the sum of a lipschitz continuous and a strongly monotone operator", *Journal of Optimization Theory and Applications*, vol. 183, pp. 179–198, 2019.

[136] N. Mignoni, R. Rahimi Baghbadorani, P. Mohajerin Esfahani, R. Carli, M. Dotoli, and S. Grammatico, "`monviso`: A Python package for solving monotone variational inequalities", in *2025 European Control Conference (ECC)*, IEEE, 2025.

[137] C. Combettes and S. Pokutta, "Complexity of linear minimization and projection on some sets", *Operations Research Letters*, vol. 49, no. 4, pp. 565–571, 2021.

[138] G. Braun et al., "Conditional gradient methods", *preprint available at arXiv:2211.14103*, 2022.

[139] M. Jaggi, "Revisiting frank-wolfe: Projection-free sparse convex optimization", in *International conference on machine learning*, PMLR, 2013, pp. 427–435.

[140] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of frank-wolfe optimization variants", *Advances in neural information processing systems*, vol. 28, 2015.

[141] J. Hammond, "Solving asymmetric variational inequality problems and systems of equations with generalized nonlinear programming algorithms", Ph.D. dissertation, Massachusetts Institute of Technology, 1984.

[142] C. Chen, L. Luo, W. Zhang, and Y. Yu, "Efficient projection-free algorithms for saddle point problems", *Advances in Neural Information Processing Systems*, vol. 33, pp. 10 799–10 808, 2020.

[143] V. Kolmogorov, "Solving relaxations of map-mrf problems: Combinatorial in-face frank-wolfe directions", in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 11 980–11 989.

[144] S. Dafermos, "Traffic equilibrium and variational inequalities", *Transportation science*, vol. 14, no. 1, pp. 42–54, 1980.

[145] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "OSQP: An operator splitting solver for quadratic programs", *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.

[146] D. Liu, V. Cevher, and Q. Tran-Dinh, "A newton frank–wolfe method for constrained self-concordant minimization", *Journal of Global Optimization*, pp. 1–27, 2022.

[147] T. Lin and M. I. Jordan, "Perseus: A simple and optimal high-order method for variational inequalities", *Mathematical Programming*, pp. 1–42, 2024.

[148] T. Başar and G. J. Olsder, *Dynamic Noncooperative Game Theory* (Classics in Applied Mathematics 23), 2nd ed. Philadelphia: SIAM, 1999, ISBN: 978-0-89871-429-6.

[149] A. Haurie, J. B. Krawczyk, and G. Zaccour, *Games and dynamic games*. World Scientific Publishing Company, 2012, vol. 1.

9

[150] Y. Theodor and U. Shaked, "Output-feedback mixed H-infinity and H-2 control – a dynamic game approach", *International Journal of Control*, vol. 64, no. 2, May 1996.

[151] M. Wang, Z. Wang, J. Talbot, J. C. Gerdes, and M. Schwager, "Game-Theoretic Planning for Self-Driving Cars in Multivehicle Competitive Scenarios", *IEEE Transactions on Robotics*, vol. 37, no. 4, pp. 1313–1325, Aug. 2021.

[152] R. Spica, E. Cristofalo, Z. Wang, E. Montijano, and M. Schwager, "A Real-Time Game Theoretic Planner for Autonomous Two-Player Drone Racing", *IEEE Transactions on Robotics*, vol. 36, no. 5, pp. 1389–1403, Oct. 2020.

[153] S. Hall, G. Belgioioso, D. Liao-McPherson, and F. Dorfler, "Receding horizon games with coupling constraints for demand-side management", in *2022 IEEE 61st Conference on Decision and Control (CDC)*, 2022, pp. 3795–3800.

[154] S. Hall, L. Guerrini, F. Dörfler, and D. Liao-McPherson, "Receding horizon games for modeling competitive supply chains", in *8th IFAC Conference on Nonlinear Model Predictive Control NMPC 2024*, 2024.

[155] A. Monti, B. Nortmann, T. Mylvaganam, and M. Sassano, "Feedback and open-loop nash equilibria for lq infinite-horizon discrete-time dynamic games", *SIAM Journal on Control and Optimization*, vol. 62, no. 3, pp. 1417–1436, 2024.

[156] M. Sassano and A. Astolfi, "Constructive design of open-loop nash equilibrium strategies that admit a feedback synthesis in lq games", *Automatica*, vol. 133, p. 109 840, 2021.

[157] E. Benenati and S. Grammatico, "Linear-quadratic dynamic games as receding-horizon variational inequalities", *IEEE Transactions on Automatic Control*, pp. 1–16, 2025, Accepted.

[158] S. Le Cleac'h, M. Schwager, and Z. Manchester, "ALGAMES: A fast augmented Lagrangian solver for constrained dynamic games", *Autonomous Robots*, vol. 46, no. 1, pp. 201–215, Jan. 2022.

[159] S. Hall, D. Liao-McPherson, G. Belgioioso, and F. Dörfler, "Stability Certificates for Receding Horizon Games", *preprint available at arXiv:2404.12165*, Apr. 2024. arXiv: 2404.12165 [cs, eess]. Accessed: May 1, 2024.

[160] J. Eckstein and M. C. Ferris, "Operator-splitting methods for monotone affine variational inequalities, with a parallel application to optimal control", *INFORMS Journal on Computing*, vol. 10, no. 2, pp. 218–235, 1998.

[161] J. P. Hespanha, *Linear Systems Theory*, Second edition. Princeton Oxford: Princeton University Press, 2018, ISBN: 978-0-691-17957-5.

[162] H. Min, P. Yi, W. Wang, J. Lei, and J. Wang, "ADMM-iCLQG: A Fast Solver of Constrained Dynamic Games for Planning Multi-Vehicle Feedback Trajectory", in *2025 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, Jun. 2025, pp. 2265–2272.

[163] R. Rahimi Baghbadorani and S. Grammatico, "Fast newton methods for linear-quadratic dynamic games with application to autonomous vehicle platooning and intersection crossing", *preprint available at arXiv:2605.01898*, 2026.

[164] G. Scutari, F. Facchinei, J. Pang, and D. Palomar, "Real and complex monotone communication games", *IEEE Transactions on Information Theory*, vol. 60, pp. 4197–4231, 2014.

[165] W. Saad, Z. Han, H. Poor, and T. Basar, "Game-theoretic methods for the smart grid: An overview of microgrid systems, demand-side management, and smart grid communications", *IEEE Signal Processing Magazine*, vol. 29, no. 5, pp. 86–105, 2012.

[166] D. Acemoglu and M. Jensen, "Aggregate comparative statics", *Games and Economic Behavior*, vol. 81, pp. 27–49, 2013, ISSN: 0899-8256.

[167] A. Kajii and T. Ui, "Incomplete information games with multiple priors", *The Japanese Economic Review*, vol. 56, no. 3, pp. 332–351, Sep. 2005.

[168] E. Kalai, "Large robust games", *Econometrica*, vol. 72, no. 6, pp. 1631–1665, Nov. 2004.

[169] M. Kopel, "Simple and complex adjustment dynamics in cournot duopoly models", *Chaos, Solitons & Fractals*, vol. 7, no. 12, pp. 2031–2048, Dec. 1996.

[170] L. G. Epstein, "A definition of uncertainty aversion", *The Review of Economic Studies*, vol. 66, no. 3, pp. 579–608, Jul. 1999.

[171] T. Başar and G. Olsder, "Dynamic non-cooperative game theory", 1999.

[172] D. Paccagnan, B. Gentile, F. Parise, M. Kamgarpour, and J. Lygeros, "Nash and wardrop equilibria in aggregative games with coupling constraints", *IEEE Transactions on Automatic Control*, vol. 64, no. 4, pp. 1373–1388, 2019.

[173] M. Marinacci, "Ambiguous games", *Games and Economic Behavior*, vol. 31, no. 2, pp. 191–219, May 2000.

[174] G. P. Crespi, D. Kuroiwa, and M. Rocca, "Robust nash equilibria in vector-valued games with uncertainty", *Annals of Operations Research*, vol. 289, no. 2, pp. 185–193, Mar. 2020.

[175] G. P. Crespi, D. Radi, and M. Rocca, "Insights on the theory of robust games", *Computational Economics*, vol. 65, no. 2, pp. 717–761, Feb. 2025.

[176] P. Couchman, B. Kouvaritakis, M. Cannon, and F. Prashad, "Gaming strategy for electric power with random demand", *IEEE Transactions on Power Systems*, vol. 20, no. 3, pp. 1283–1292, 2005.

[177] O. Singh V.V. nad Jouini and A. Lisser, "Existence of nash equilibrium for chance-constrained games", *Operations Research Letters*, vol. 44, no. 5, pp. 640–644, 2016, ISSN: 0167-6377.

[178] M. Aghassi and D. Bertsimas, "Robust game theory", *Mathematical Programming*, vol. 107, no. 1-2, pp. 231–273, 2006.

[179] S. Hayashi, N. Yamashita, and M. Fukushima, "Robust Nash equilibria and second-order cone complementarity problems", *Journal of Nonlinear and Convex Analysis*, vol. 6, 2005.

[180] M. Dror, L. A. Guardiola, A. Meca, and J. Puerto, "Dynamic realization games in newsvendor inventory centralization", *International Journal of Game Theory*, vol. 37, no. 1, pp. 139–153, 2008.

[181] D. Bauso and J. B. Timmer, "Robust dynamic cooperative games", *International Journal of Game Theory*, vol. 38, no. 1, pp. 23–36, 2009, ISSN: 0020-7276.

[182] G. Pantazis, F. Fabiani, F. Fele, and K. Margellos, "Probabilistically robust stabilizing allocations in uncertain coalitional games", *IEEE Control Systems Letters*, vol. 6, pp. 3128–3133, 2022. DOI: 10.1109/LCSYS.2022.3182152.

[183] F. Fele and K. Margellos, "Probabilistic sensitivity of Nash equilibria in multi-agent games: A wait-and-judge approach", pp. 5026–5031, 2019, ISSN: 2576-2370.

[184] F. Fele and K. Margellos, "Probably approximately correct Nash equilibrium learning", *IEEE Transactions on Automatic Control*, vol. 66, no. 9, pp. 4238–4245, 2021.

[185] D. Paccagnan and M. Campi, "The scenario approach meets uncertain game theory and variational inequalities", in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 6124–6129. DOI: 10.1109/CDC40024.2019.9030247.

[186] G. Pantazis, F. Fele, and K. Margellos, "A posteriori probabilistic feasibility guarantees for Nash equilibria in uncertain multi-agent games", *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 3403–3408, 2020, 21st IFAC World Congress, ISSN: 2405-8963.

[187] G. Pantazis, F. Fele, and K. Margellos, "On the probabilistic feasibility of solutions in multi-agent optimization problems under uncertainty", *European Journal of Control*, vol. 63, pp. 186–195, 2022, ISSN: 0947-3580.

[188] M. Mammarella, V. Mirasierra, M. Lorenzen, T. Alamo, and F. Dabbene, "Chance-constrained sets approximation: A probabilistic scaling approach", *Automatica*, vol. 137, p. 110 108, 2022, ISSN: 0005-1098. DOI: https://doi.org/10.1016/j.automatica.2021.110108. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0005109821006373.

[189] G. Pantazis, F. Fele, and K. Margellos, "A priori data-driven robustness guarantees on strategic deviations from generalised nash equilibria", *IFAC Automatica*, pp. 1–16, Apr. 2023.

[190] B. Franci and S. Grammatico, "A distributed forward–backward algorithm for stochastic generalized nash equilibrium seeking", *IEEE Transactions on Automatic Control*, vol. 66, no. 11, pp. 5467–5473, 2021. DOI: 10.1109/TAC.2020.3047369.

[191] B. Franci and S. Grammatico, "Stochastic generalized nash equilibrium-seeking in merely monotone games", *IEEE Transactions on Automatic Control*, vol. 67, no. 8, pp. 3905–3919, 2022. DOI: 10.1109/TAC.2021.3108496.

[192] C. Villani, "Topics in optimal transportation", Graduate studies in mathematics, no. 58, 2016.

[193] S. Dereich, M. Scheutzow, and R. Schottstedt, "Constructive quantization: Approximation by empirical measures", *Annales de l'Institut Henri Poincaré, Probabilités et Statistiques*, vol. 49, 2011.

[194] P. Mohajerin Esfahani and D. Kuhn, "Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations", *Mathematical Programming*, vol. 171, no. 1-2, pp. 115–166, Sep. 2018. Accessed: Feb. 21, 2023.

[195] J. Dedecker and F. Merlevède, "Behavior of the empirical wasserstein distance in $\mathbb{R}^d$ under moment conditions", *Electronic Journal of Probability*, vol. 24, 2019.

[196] J. Weed and F. Bach, "Sharp asymptotic and finite-sample rates of convergence of empirical measures in wasserstein distance", *Bernoulli*, vol. 25, 2017.

[197] J. Weed and Q. Berthet, "Estimation of smooth densities in wasserstein distance", 2019.

[198] N. Fournier, "Convergence of the empirical measure in expected wasserstein distance: Non-asymptotic explicit bounds in $\mathbb{R}^d$", 2023.

[199] D. Kuhn, P. Mohajerin Esfahani, V. Nguyen, and S. Shafieezadeh-Abadeh, "Wasserstein distributionally robust optimization: Theory and applications in machine learning", *INFORMS Tutorials in Operations Research*, pp. 130–166, 2019. DOI: 10.1287/educ.2019.0198.

[200] L. Chaouach, T. Oomen, and D. Boskos, "Comparing structured ambiguity sets for stochastic optimization: Application to uncertainty quantification", *2023 62nd IEEE Conference on Decision and Control (CDC)*, pp. 8274–8279, 2023. DOI: 10.1109/CDC49753.2023.10383381.

[201] L. Chaouach, D. Boskos, and T. Oomen, "Uncertain uncertainty in data-driven stochastic optimization: Towards structured ambiguity sets", *2022 IEEE 61st Conference on Decision and Control (CDC)*, pp. 4776–4781, 2022. DOI: 10.1109/CDC51059.2022.9992405.

[202] Z. Chen, D. Kuhn, and W. Wiesemann, "Data-driven chance constrained programs over wasserstein balls", *Operations Research*, 2018.

[203] A. Hota, A. Cherukuri, and J. Lygeros, "Data-driven chance constrained optimization under wasserstein ambiguity sets", *2019 American Control Conference (ACC)*, pp. 1501–1506, 2018.

[204] M. Heinlein, T. Alamo, and S. Lucia, "On the risk levels of distributionally robust chance constrained problems", *IEEE Transactions on Automatic Control (submitted)*, 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2409.01177.

[205] S. Peng, A. Lisser, V. Singh, N. Gupta, and E. Balachandar, "Games with distributionally robust joint chance constraints", *Optimization Letters*, vol. 15, pp. 1931–1953, 2021. DOI: 10.1007/s11590-021-01700-9.

[206] Y. Liu, H. Xu, S. Yang, and J. Zhang, "Distributionally robust equilibrium for continuous games: Nash and stackelberg models", *European Journal of Operational Research*, vol. 265, no. 2, pp. 631–643, 2018, ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2017.07.050.

[207] T. Xia, J. Liu, and A. Lisser, "Distributionally robust chance constrained games under wasserstein ball", *Operations Research Letters*, vol. 51, no. 3, pp. 315–321, 2023, ISSN: 0167-6377.

[208] F. Fabiani and B. Franci, "On distributionally robust generalized nash games defined over the wasserstein ball", *Journal of Optimization Theory and Applications*, vol. 199, no. 2, pp. 298–309, Oct. 2023.

[209] Z. Chen, D. Kuhn, and W. Wiesemann, "On approximations of data-driven chance constrained programs over wasserstein balls", *Operations Research Letters*, vol. 51, no. 3, pp. 226–233, 2023, ISSN: 0167-6377. DOI: https://doi.org/10.1016/j.orl.2023.02.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167637723000317.

[210] S. Shafiee, L. Aolaritei, D. Dörfler, and D. Kuhn, "New perspectives on regularization and computation in optimal transport-based distributionally robust optimization", *preprint available at arXiv:2303.03900*, 2024. [Online]. Available: https://arxiv.org/abs/2303.03900.

[211] N. Lanzetti, S. Bolognani, and F. Dörfler, "First-order conditions for optimization in the wasserstein space", *SIAM Journal on Mathematics of Data Science*, vol. 7, no. 1, pp. 274–300, Mar. 2025.

[212] G. Pantazis, B. Franci, and S. Grammatico, "On data-driven wasserstein distributionally robust nash equilibrium problems with heterogeneous uncertainty", *preprint available at arXiv*, 2023. [Online]. Available: https://arxiv.org/abs/2312.03573.

[213] D. Boskos, J. Cortés, and S. Martinez, "High-confidence data-driven ambiguity sets for time-varying linear systems", *IEEE Transactions on Automatic Control*, vol. 69, no. 2, pp. 797–812, 2024. DOI: 10.1109/TAC.2023.3273815.

[214] R. Rahimi Baghbadorani, P. Mohajerin Esfahani, and S. Grammatico, "A hybrid algorithm for monotone variational inequalities", *European Control Conference (ECC). preprint available at arXiv:2604.03658*, 2026.

[215] L. Kantorovich and S. Rubinstein, "On a space of totally additive functions", *Vestnik Leningrad. Univ*, vol. 13, pp. 52–59, 1958.

[216] A. Nagurney, *Network Economics: A Variational Inequality Approach* (Advances in Computational Economics). Dordrecht, The Netherlands: Kluwer Academic Publishers, 1999, vol. 10.

[217] European Union, *Directive (eu) 2018/2001 of the european parliament and of the council of 11 december 2018 on the promotion of the use of energy from renewable sources*, 2018.

[218] S. Ahmed, A. Ali, and A. D'Angola, "A review of renewable energy communities: Concepts, scope, progress, challenges, and recommendations", *Sustainability*, vol. 16, no. 5, p. 1749, 2024. DOI: 10.3390/su16051749.

[219] L. Gruber, I. Kockar, and S. Wogrin, "Towards resilient energy communities: Evaluating the impact of economic and technical optimization", *International Journal of Electrical Power & Energy Systems*, vol. 155, p. 109 592, 2024. DOI: 10.1016/j.ijepes.2023.109592.

[220] G. G. Zanvettor, M. Casini, A. Giannitrapani, S. Paoletti, and A. Vicino, "Optimal management of energy communities hosting a fleet of electric vehicles", *Energies*, vol. 15, no. 22, p. 8697, 2022. DOI: 10.3390/en15228697.

[221] M. Stentati, S. Paoletti, and A. Vicino, "Optimization of energy communities in the italian incentive system", in *2022 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, Novi Sad, Serbia: IEEE, 2022, pp. 1–5. DOI: 10.1109/ISGT-Europe54678.2022.9960513.

[222] J. Faraji, Z. D. Grève, and F. Vallée, "Hierarchical energy sharing management for a renewable energy community with heterogeneous end-users", in *2023 IEEE Belgrade PowerTech*, Belgrade, Serbia: IEEE, 2023, pp. 1–8. DOI: 10.1109/PowerTech55446.2023.10202668.

[223] V. Casella, G. Ferro, L. Parodi, and M. Robba, "Energy community optimal management: A bilevel approach", in *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, IEEE, 2024, pp. 2720–2725.

[224] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers", *Foundations and Trends in Machine Learning*, vol. 3, pp. 1–122, Jan. 2011.

[225] W. Zhang, K. Ma, and Z. Yan, "Admm-based distributed optimal energy management for microgrids", *IEEE Transactions on Smart Grid*, vol. 9, no. 5, pp. 4547–4557, 2018.

[226] J. Li, S. Chen, C. Li, and F. Wang, "Distributed game strategy for formation flying of multiple spacecraft with disturbance rejection", *IEEE Transactions on Aerospace and Electronic Systems*, vol. 57, no. 1, pp. 119–128, 2020.

[227] G. Alonso and A. Parisio, "Real-time distributed mpc for multi-energy microgrids using admm", *IEEE Transactions on Smart Grid*, vol. 12, no. 4, pp. 3359–3369, 2021.

[228] W. Deng and W. Yin, "On the global linear convergence of the admm with strongly convex objectives", *Journal of Scientific Computing*, vol. 66, no. 3, pp. 889–916, 2016.

[229] A. Kadkhodaei and A. G. Richards, "Fast admm for quadratic programming using nesterov acceleration", in *2022 American Control Conference (ACC)*, 2022, pp. 3036–3041.

[230] B. Stellato, A. Bemporad, and S. Boyd, "Embedded optimization for model predictive control at megahertz rates", *IEEE Transactions on Automatic Control*, vol. 67, no. 6, pp. 2795–2810, 2020.

[231] A. Zanelli, G. Frison, R. Quirynen, and M. Diehl, "Forces pro: A general-purpose embedded optimization solver", *IEEE Control Systems Magazine*, vol. 41, no. 4, pp. 36–55, 2021.

[232] T. Kim, J. Park, and H. Choi, "Embedded optimization for real-time energy management in microgrids: A comparative study", *Control Engineering Practice*, vol. 139, p. 105 591, 2023.

[233] T. V. Dang, K. Ling, and J. Maciejowski, "Embedded admm-based qp solver for mpc with polytopic constraints", in *2015 European Control Conference (ECC)*, 2015, pp. 3446–3451. DOI: 10.1109/ECC.2015.7331067.

[234] T. Goldstein, B. O'Donoghue, S. Setzer, and R. Baraniuk, "Fast alternating direction optimization methods", *SIAM Journal on Imaging Sciences*, vol. 7, no. 3, pp. 1588–1623, 2014.

[235] Y. Lopes, N. C. Fernandes, D. C. Muchaluat-Saade, and K. Obraczka, "Smart grid communication: Requirements and scada protocols analysis", in *2018 Simposio Brasileiro de Sistemas Eletricos (SBSE)*, IEEE, 2018, pp. 1–6.

[236] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal parameter selection for the alternating direction method of multipliers (admm): Quadratic problems", *IEEE Transactions on Automatic Control*, vol. 60, no. 3, pp. 644–658, 2014.

[237] H. H. Bauschke and J. M. Borwein, "On projection algorithms for solving convex feasibility problems", *SIAM review*, vol. 38, no. 3, pp. 367–426, 1996.

[238] P. T. Harker and J. Pang, "For the linear complementarity problem", *Lectures in Applied Mathematics*, vol. 26, pp. 265–284, 1990.

[239] K. Mishchenko, "Regularized newton method with global convergence", *SIAM Journal on Optimization*, vol. 33, no. 3, pp. 1440–1462, 2023.

[240] P. Scarabaggio, R. Carli, S. Grammatico, and M. Dotoli, "Local generalized nash equilibria with nonconvex coupling constraints", *IEEE Transactions on Automatic Control*, 2024.

[241] N. Mignoni, J. Martinez-Piazuelo, R. Carli, C. Ocampo-Martinez, N. Quijano, and M. Dotoli, "A game-theoretical control framework for transactive energy trading in energy communities", in *2024 European Control Conference (ECC)*, IEEE, 2024, pp. 786–791.

[242] N. Mignoni, R. Carli, and M. Dotoli, "A noncooperative stochastic rolling horizon control framework for v1g and v2b scheduling in energy communities", in *2023 European Control Conference (ECC)*, IEEE, 2023, pp. 1–6.

[243] R. Anand, D. Aggarwal, and V. Kumar, "A comparative analysis of optimization solvers", *Journal of Statistics and Management Systems*, vol. 20, no. 4, pp. 623–635, 2017.

[244] G. Peev, N. Simidjievski, and S. Džeroski, "Modeling of dynamical systems: A survey of tools and a case study", in *20th International Multiconference Information Society-IS*, 2017, pp. 15–18.

[245] S. Diamond and S. Boyd, "CVXPY: A Python-embedded modeling language for convex optimization", *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[246] G. Chierchia, E. Chouzenoux, P. L. Combettes, and J.-C. Pesquet, "The proximity operator repository", *User's guide http://proximity-operator. net/download/guide. pdf. Accessed*, vol. 6, 2020.

[247] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable convex optimization layers", *Advances in neural information processing systems*, vol. 32, 2019.

[248] D. Jeong et al., "Numerical study of an indicator function for front-tracking methods", *Mathematical Problems in Engineering*, vol. 2022, no. 1, p. 7 381 115, 2022.

[249] C. E. Lemke and J. T. Howson Jr, "Equilibrium points of bimatrix games", *Journal of the Society for industrial and Applied Mathematics*, vol. 12, no. 2, pp. 413–423, 1964.

[250] R. S. Sutton, "Reinforcement learning: An introduction", *A Bradford Book*, 2018.

[251] P. Yi and L. Pavel, "An operator splitting approach for distributed generalized nash equilibria computation", *Automatica*, vol. 102, pp. 111–121, 2019.

[252] K. Ma, G. Hu, and C. J. Spanos, "Distributed energy consumption control via real-time pricing feedback in smart grid", *IEEE Transactions on Control Systems Technology*, vol. 22, no. 5, pp. 1907–1914, 2014.

[253] S. Cianchi, R. Rahimi Baghbadorani, A. Sanjab, and S. Grammatico, "Learning-based stackelberg equilibrium seeking with application to demand-side energy management", *preprint available at arXiv:2605.00588*, 2026.

ACKNOWLEDGEMENTS

If you have jumped directly to this section, as is often the case, I warmly welcome you and sincerely appreciate your support throughout this journey.

Navigating the challenges and limitations of my home country has not been easy, but the path has been made smoother by the people I have walked alongside. I am shaped by my community, and I dedicate this work to them as a small token of my deep gratitude.

My deepest thanks go to my family in Iran. To my mother, father, and my brothers, Mohammad and Sepehr: thank you for your unwavering patience and love. Although I am currently in the Netherlands, I send you my warmest embrace and look forward to our reunion.

Upon moving to the Netherlands, I was fortunate to find a second family. I am deeply grateful to Artemis, who has been the sister I never had; Masoud, who guided me like an older brother; and Sir Syrous, as a guiding elder. My time here was further enriched by the companionship of Darya, Matthijs, Ali, Arash, and Mohammad Bo.

I am equally grateful for the academic environment and the people who supported me throughout this journey. My sincere appreciation goes to my supervisors, Sergio and Peyman, for their exceptional guidance, generosity, and continuous support, both academically and personally. I would also like to thank Amin and Arman for their mentorship and support as senior colleagues, as well as all members of my research group and colleagues: Rayyan, Pedro, Georgios, Mattia, Wicak, Emilio, Arghya, Silvia, Luyao, Shijie, Tolga, Amin, Shabnam, Suad, Aitazaz, Gabriel, Prady, Anuj, Emrul, and the entire DCSC staff, along with others I may have unintentionally omitted.

Finally, I would like to express my special gratitude to Mohammad Kh for his support and kindness, not only as an academic mentor but also as an older brother figure. He introduced me to my current postdoctoral supervisor, Yashar, whose exceptional guidance and support I also sincerely acknowledge.

Reza Rahimi Baghbadorani
Delft, April 2026

CURRICULUM VITÆ

Reza RAHIMI BAGHBADORANI

05-07-1997 Born in Isfahan, Iran.

EDUCATION

2015–2019	B.Sc. Electrical Engineering Isfahan University of Technology
2019–2021	M.Sc. Systems and Control Sharif University of Technology
Thesis:	Vehicle Routing Problem Using Model Predictive Method
Promotor:	Prof. M. Haeri
2022–2026	Ph.D. Systems and Control Delft University of Technology
Thesis:	Fast Algorithms for Optimization, Games, and Control Applications
Promotors:	Dr. Ing. S. Grammatico & Dr. P. Mohajerin Esfahani

List of Publications

Journal articles

1. Giulio Ferro, Sergio Grammatico, Luca Parodi, **Reza Rahimi Baghbadorani**, Michela Robba, *An embedded accelerated decentralized optimization algorithm with application to energy communities*, Control Engineering Practice, 2026.
2. **Reza Rahimi Baghbadorani**, Peyman Mohajerin Esfahani, Sergio Grammatico, *A Frank-Wolfe algorithm for strongly monotone variational inequalities*, Operations Research Letters, 2025.

Conference papers

1. **Reza Rahimi Baghbadorani**, Sergio Grammatico, *Fast Newton methods for linear-quadratic dynamic games with application to autonomous vehicle platooning and intersection crossing*, IEEE International Conference on Intelligent Transportation Systems, 2026.
2. **Reza Rahimi Baghbadorani**, Emilio Benenati, Sergio Grammatico, *A Douglas-Rachford splitting method for solving monotone variational inequalities in linear-quadratic dynamic games*, IFAC World Congress, 2026.
3. **Reza Rahimi Baghbadorani**, Peyman Mohajerin Esfahani, Sergio Grammatico, *A Hybrid Algorithm for Monotone Variational Inequalities*, European Control Conference (ECC), 2026.
4. Nicola Mignoni, **Reza Rahimi Baghbadorani**, Raffaele Carli, Peyman Mohajerin Esfahani, Mariagrazia Dotoli, Sergio Grammatico, *monviso: A Python package for solving monotone variational inequalities*, European Control Conference (ECC), 2025.
5. **Reza Rahimi Baghbadorani**, Peyman Mohajerin Esfahani, Sergio Grammatico, *Fast Adaptive First-Order Semidefinite Programming for Data-Driven Linear Quadratic Regulation*, 32nd Mediterranean Conference on Control and Automation (MED), 2024.

Preprints

1. Arghya Mallick, **Reza Rahimi Baghbadorani**, Silvia Cianchi, Peyman Mohajerin Esfahani, Sergio Grammatico, *Inverse Dynamic Game Theory via Robust Receding Horizon Learning*, To be submitted, 2026.
2. Silvia Cianchi, **Reza Rahimi Baghbadorani**, Anibal Sanjab, Sergio Grammatico, *Learning-Based Stackelberg Equilibrium Seeking with Application to Demand-Side Energy Management*, preprint available at arXiv:2605.00588, 2026.
3. **Reza Rahimi Baghbadorani**, Sergio Grammatico, Peyman Mohajerin Esfahani, *Locally Linear Convergence for Nonsmooth Convex Optimization via Coupled Smoothing and Momentum*, preprint available at arXiv:2511.10239, 2025.

4. Georgios Pantazis, **Reza Rahimi Baghbadorani**, Sergio Grammatico, *Nash equilibrium seeking for a class of quadratic-bilinear Wasserstein distributionally robust games*, preprint available at arXiv:2411.09636, 2024.

5. **Reza Rahimi Baghbadorani**, Sergio Grammatico, Peyman Mohajerin Esfahani, *A New Lineserach for Accelerated Composite Minimization*, preprint available at arXiv:2405.03414, 2024.

AI REVIEWERS CONFUSED BY HUMAN RESEARCH: 'TOO SLOW' CLAIMS GPT-8

[illegible]

GRAD STUDENT UNIONS 'IS IT WORK IF WE'RE ALGORITHMS?'

[illegible]

FUNDING CRISIS!

PRODUCING LESS PAPER WORK THAN DESIRED

[illegible]

ETHICS COMMITTEES BATTLE CHATGPT CHESS-PLAYER BOT OVER SENSITIVE DATA

[illegible]

DELFT CAMPUS ANNOUNCES NEW-CHAIR OF FAST THOUGHT- FOR FAST PHD GRADUATIONS

[illegible]

CITATIONS PER GALLON: MEASURING RESEARCH EFFICACY IN THE THE POST-HUMAN ERA

[illegible]