Advances in Computer-Assisted Design and Analysis of First-Order Optimization Methods and Related Problems

by

Shuvomoy Das Gupta

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN OPERATIONS RESEARCH

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2024

Authored by: Shuvomoy Das Gupta
Operations Research Center
May 1, 2024

Certified by: Robert M. Freund
Theresa Seley Professor in Management Science
MIT Sloan School of Management
Thesis Supervisor

Certified by: Bart P.G. Van Parys
Assistant Professor, Operations Research & Statistics
MIT Sloan School of Management
Thesis Supervisor

Accepted by: Georgia Perakis
John C. Head III Dean (Interim), MIT Sloan School of Management
Professor, Operations Management, Operations Research & Statistics
Co-Director, Operations Research Center

Advances in Computer-Assisted Design and Analysis of First-Order Optimization Methods and Related Problems

by

Shuvomoy Das Gupta

Submitted to the Sloan School of Management
on May 1, 2024 in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY IN OPERATIONS RESEARCH

ABSTRACT

First-order methods are optimization algorithms that can be described and analyzed using the values and gradients of the functions to be minimized. These methods have become the main workhorses for modern large-scale optimization and machine learning due to their low iteration costs, minimal memory requirements, and dimension-independent convergence guarantees. As the data revolution continues to unfold, the pressing demand for discovering faster first-order methods and rigorous convergence analyses of existing first-order methods have become the key problem in today's big data era. To that goal, in this thesis, we make advancements in computer-assisted design and analysis of first-order methods and related problems.

The core contribution of this thesis is developing computer-assisted methodologies for analyzing and designing first-order methods using nonconvex quadratically constrained quadratic optimization problems (QCQPs). In this approach, the key idea involves posing the analysis or design of first-order methods as nonconvex but practically tractable QCQPs and then solving them to global optimality using custom spatial branch-and-bound algorithms.

In Chapter 2 of this thesis, we present Branch-and-Bound Performance Estimation Programming (BnB-PEP), a unified methodology for constructing optimal first-order methods for convex and nonconvex optimization. BnB-PEP poses the problem of finding the optimal first-order method as a nonconvex but practically tractable QCQP and solves it to certifiable global optimality using a customized branch-and-bound algorithm. Our customized branch-and-bound algorithm, through exploiting specific problem structures, outperforms the latest off-the-shelf implementations by orders of magnitude, accelerating the solution time from hours to seconds and weeks to minutes. We apply BnB-PEP to several practically relevant convex and nonconvex setups and obtain first-order methods with bounds that improve upon prior state-of-the-art results. Furthermore, we use the BnB-PEP methodology to find proofs with potential function structures, thereby systematically generating analytical convergence proofs.

We next propose a QCQP-based computer-assisted approach to the analysis of the worst-case convergence of nonlinear conjugate gradient methods (NCGMs) in Chapter 3. Those methods are known for their generally good empirical performances for large-scale optimization while having relatively incomplete analyses. Using our computer-assisted approach, we establish novel complexity bounds for the Polak-Ribière-Polyak (PRP) and the Fletcher-Reeves (FR) NCGMs for smooth strongly convex minimization. In particular, we construct mathematical proofs that establish the first non-asymptotic convergence bound for FR (which is historically the first developed NCGM), and a much improved non-asymptotic convergence bound for PRP. Additionally, we provide simple adversarial examples on which these methods do not perform better than gradient descent with exact line search, leaving very little room for improvements on the same class of problems.

In Chapter 4 of this thesis, we develop the nonconvex exterior-point optimization solver (NExOS)—a first-order algorithm tailored to sparse and low-rank optimization problems. We consider the problem of minimizing a convex function over a nonconvex constraint set, where the set can be decomposed as the intersection of a compact convex set and a nonconvex set involving sparse or low-rank constraints. Unlike the convex relaxation approaches, NExOS finds a locally optimal point of the original problem by solving a sequence of penalized problems with strictly decreasing penalty parameters by exploiting the nonconvex geometry. NExOS solves each penalized problem by applying a first-order algorithm, which converges linearly to a local minimum of the corresponding penalized formulation under regularity conditions. Furthermore, the local minima of the penalized problems converge to a local minimum of the original problem as the penalty parameter goes to zero. We then implement and test NExOS on many instances from a wide variety of sparse and low-rank optimization problems, empirically demonstrating that our algorithm outperforms specialized methods.

Thesis supervisor: Robert M. Freund

Title: Theresa Seley Professor in Management Science

MIT Sloan School of Management

Thesis supervisor: Bart P.G. Van Parys

Title: Assistant Professor, Operations Research & Statistics

MIT Sloan School of Management

Acknowledgments

First and foremost, I express my deepest gratitude to my advisors, Professor Robert M. Freund and Professor Bart Van P.G. Parys, for their support and mentorship during the last five years. Their guidance and encouragement have been fundamental to my professional and personal growth. I am so grateful to them for providing me with the independence to explore research questions that resonated deeply with me while also offering insightful direction whenever I faced challenges or uncertainties. Their ability to balance fostering academic freedom while providing guidance when needed has been instrumental in shaping my research skills and intellectual perspective. I am incredibly fortunate to have worked under their mentorship and benefited from their knowledge and experience. Their contribution to my PhD journey has been immeasurable, and I hold the utmost respect and appreciation for their mentorship.

I feel deeply indebted to my collaborator, mentor, and friend: Professor Ernest K. Ryu, who also serves on my thesis committee. Ernest has profoundly influenced my academic trajectory. I first communicated with Ernest via email when I was a Master's student at the University of Toronto regarding one of his papers on monotone operator splitting. I asked him a few questions that were very simple to figure out by myself, yet he responded to me very kindly and answered with great details. After finishing my Master's, I worked in the

industry for a few years, but occasionally I would check Ernest's new papers, and I always enjoyed reading them. So, when I started my PhD, I thought that a potential collaboration with Ernest might be a great idea. In the second year of my PhD, I started working with him on computer-assisted methodologies for optimization. We had a time zone difference, so for our meetings often, I would wake up very early in the morning, and Ernest would stay awake very late, but we had so much fun working together! Working with Ernest is definitely one of the highlights of my PhD journey.

I am very thankful to Professor Andy Sun for serving on my thesis committee and for offering great advice and feedback during my PhD. Working as a Teaching Assistant in his power system optimization course was such a great experience for me to learn about the applications of optimization theory in power system design.

I want to express my sincere gratitude to my collaborator Dr. Adrien Taylor. Adrien is one of the major figures in the performance estimation area for the fantastic series of papers he has written with his excellent collaborators. When I started working in performance estimation, I learned about the topic first by reading his PhD thesis, which, besides being a landmark dissertation in the field, also has great tutorial values for clarity, rigor, and readability. Later, I had the opportunity of working with him when I realized that not only is Adrien an amazing researcher but also a very supportive mentor.

I am so grateful to Professor James Renegar for emailing me with some very kind words after I met him at the 2023 Cornell Young Researchers Workshop. I was so moved that I printed out the email and put it in front of my work desk. Whenever I feel fatigued, I look at his email, and it always inspires me to keep working hard.

I am extremely thankful to my friends at ORC, who were always appreciative of my few good qualities and tolerant of my many flaws. In particular, I want to thank my dear friends

Moïse Blanchard and Amine Bennouna, whose friendships made my PhD journey a truly wonderful experience. Moïse has been a constant source of inspiration to me: he is one of the smartest persons I have ever met, yet he is so humble, generous, and conscientious. He has always supported me through thick and thin. I am very grateful to have witnessed in Moïse a concrete example of a world-class researcher, a great human being, and an amazing friend. Amine is someone who excels at anything (including but not limited to great research) that he puts his mind to, yet he is so down-to-earth and easy-going with a great sense of humor. I learned a lot from him about how to present a research paper, as he is an excellent communicator: I was very fortunate to have Amine as a speaker in a 2022 INFORMS session that I co-chaired with Professor Van Parys, and he gave one of the best talks ever. Whenever I doubted if I was on the right path, Amine and Moïse would always express their belief in me, and I would convince myself that if they see value in my research, it must have some value: I have such admiration for their judgment and integrity. I would like to thank Yu Ma for being a trustworthy and loyal friend and for giving me good advice in times of need: *a friend in need is a friend indeed*. I am very thankful to Vassilis Digalakis Jr for his constant encouragement during the job market and beyond. I am very thankful to Lindsey Blanks, Léonard Boussioux, Kimberly Villalobos Carballo, Ryan Cory-Wright, Samuel Humphries, Zikai Xiong, Cynthia Zeng, Emily Zhang, and Jiayu Zhao for their friendship and support.

I am incredibly grateful to my best friend Ian Ruffolo. Ian was one of the few people who encouraged me to pursue a PhD in when I was working in industry. During my PhD, he has always been there for me and constantly supported me during difficult times. His kindness, compassion, and level-headedness inspire me. He was also a patient listener to many of my research presentations and provided excellent feedback on improving my English. One of the fondest memories from my PhD years was acting as Ian's travel guide for MIT, Harvard, and the Cambridge area during the Summer of 2023.

I would like to thank the Harvard Square Homeless Shelter and the First Church of Cambridge for giving me the opportunity to volunteer at the "Saturday morning brunches" and the "Friday Café", respectively. The volunteers there are some of the kindest, most composed, and most altruistic people I have ever met, and I learned so much from them about working hard to attain a goal yet developing the equanimity to let go of things that we are not meant for.

Last but not least, some words of gratitude to my parents, Madhabi Rani Deb and Sudhendu Das Gupta. *Ma* and *Baba*, you have worked incredibly hard in life and sacrificed so much for us to ensure that we get a good education. My achievements, if any, have only been possible because of your unconditional love and support. I want to thank my younger brother, Dhrubomoy Das Gupta, for taking care of our parents during the last five years, which gave me the opportunity to do a PhD in the first place. You are the best brother ever.

I dedicate this thesis to my late grandmother, Malati Rani Deb, who raised me and instilled the belief in me that I can succeed in anything if I work hard and am willing to learn from my mistakes. If I have any good qualities, it is because of her, whereas my flaws are mine only. *Dida*, this is for you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Optimization problems pervade engineering, operations research, and machine learning, and as such, finding the fastest algorithm to solve them is of paramount importance. The efficiency of an optimization method can make a significant difference in the outcome of a problem, as well as the feasibility of solving it in practical applications. The ability to efficiently solve large-scale optimization problems has a direct impact on a wide range of areas, including operations research, management science, machine learning, and many others. In addition, optimizing the speed of an optimization method has far-reaching implications in terms of computational resources, cost-effectiveness, and energy consumption.

For these aforementioned reasons, researchers in optimization continue to seek and develop faster methods with better convergence rates, lower computational complexity, and enhanced robustness. First-order methods are optimization algorithms that can be described and analyzed using the values and gradients of the functions to be minimized. These methods have emerged as the principal tools in modern large-scale optimization and machine learning, attributed to their low iteration costs, minimal memory requirements, and dimension-

independent convergence guarantees. While other types of optimization methods such as second-order methods can offer faster convergence in certain situations, they are computationally expensive and often impractical for large-scale optimization and machine learning as these second-order methods require matrix inversion or factorization and solution of large systems of equations. In contrast, first-order methods can handle large-scale data sets and are amenable to distributed computing, making them well-suited for machine learning applications. Hence, discovery of new faster first-order methods along with tighter convergence analysis of existing first-order methods can have a significant impact on the feasibility and scalability of large-scale optimization and machine learning problems, as well as their ability to handle real-world applications in a timely and cost-effective manner.

As a result, developing methodologies that can systematically find novel faster first-order methods and discover tight convergence analyses of existing first-order methods is of great importance. To that end, this thesis makes advancements in computer-assisted methodologies for analyzing and designing first-order methods using nonconvex quadratically constrained quadratic optimization problems (QCQPs). The key idea in this approach is to pose the design and analysis of first-order methods as nonconvex but practically tractable QCQPs. By solving these QCQPs to global optimality using custom spatial branch-and-bound algorithms, we can either discover optimal first-order methods or establish tight convergence analyses for known first-order methods from the numerical solutions to the QCQPs.

The results in chapters 2, 3, 4 are included in the papers [1–3], respectively. We next provide a brief summary for each of the remaining chapters of this thesis below.

Chapter 2: Branch-and-Bound Performance Estimation Programming: A Unified Methodology for Constructing Optimal Optimization Methods. In Chapter 2 of this thesis, we present Branch-and-Bound Performance Estimation Programming (BnB-PEP), a

unified methodology for constructing optimal first-order methods for convex and nonconvex optimization.

About ten years ago, researchers showed that the performance of a *known* first-order method can be computed exactly by solving a tractable higher-level semidefinite optimization problem called a *performance estimation problem* (PEP) [4–6], and since then this framework has led to many novel methods and analyses [7–10]. However, if the goal is instead to *discover* the provably best first-order method for a given problem class, the aforementioned methodologies encounter significant limitations. BnB-PEP overcomes those limitations by posing the problem of finding the optimal first-order method as a nonconvex but dimension-independent and practically tractable QCQP and solving it to certifiable global optimality using a customized branch-and-bound algorithm. Our customized open-sourced branch-and-bound algorithm, through exploiting specific problem structures, outperforms the latest off-the-shelf implementations by orders of magnitude, accelerating the solution time from hours to seconds and weeks to minutes.

We apply BnB-PEP to several practically relevant convex and nonconvex setups and obtain first-order methods with bounds that improve upon prior state-of-the-art results. Furthermore, we use the BnB-PEP methodology to find proofs with potential function structures, thereby systematically generating analytical convergence proofs. Finally, we provide roughly 10,000 lines of code precisely describing and demonstrating our methodology:

https://github.com/Shuvomoy/BnB-PEP-code

Chapter 3: Nonlinear conjugate gradient methods: worst-case convergence rates via computer-assisted analyses. We next propose a QCQP-based computer-assisted approach to the analysis of the worst-case convergence of nonlinear conjugate gradient methods

(NCGMs) in Chapter 3. NCGMs are a class of adaptive (stepsizes are functions of the gradients of previous iterates) first-order algorithms, and notable NCGMs include Polak-Ribière-Polyak (PRP) [11, 12], Fletcher-Reeves (FR) [13], Hestenes-Stiefel method [14], the conjugate descent method due to Fletcher [15], and Dai-Yuan method [16].

In the class of NCGMs, FR and PRP play a fundamental role, as under exact line search, the other NCGMs reduce to either PRP or FR. These NCGMs exhibit excellent empirical performances for solving large-scale optimization problems but have barely any non-asymptotic convergence results in spite of being more than 50 years old. In that context, we make a two-fold contribution in Chapter 3. First, we compute non-asymptotic worst-case convergence results along with simple counter-examples in dimension 4 for PRP and FR under exact line search. Our convergence results are obtained by formulating the problems of computing worst-case scenarios as nonconvex QCQPs, and then by solving them to global optimality. Second, these computations enable us to construct mathematical proofs that establish an improved non-asymptotic convergence bound for PRP, and, to the best of our knowledge, the first non-asymptotic convergence bound for FR. Furthermore, the worst-case bounds for PRP and FR obtained numerically reveal that there are simple adversarial examples on which these methods do not perform better than gradient descent with exact line search, leaving very little room for improvements on this class of problems. Since we demonstrate that the convergence results of NCGMs associated with exact line search are already disappointing, we conclude that inexact line searches, which approximate exact line search, are unlikely to offer improvement.

We also contribute in terms of computer-assisted methodologies too in this chapter. Our approach to computing worst-case scenarios and bounds for adaptive first-order methods through nonconvex QCQPs advances the semidefinite programming-based PEP methodologies [4–6] developed for non-adaptive first-order methods. This contribution aligns with the

spirit of Chapter 2, developed for devising optimal (but non-adaptive) first-order methods. The code used to generate and validate the results in this chapter is available at:

https://github.com/Shuvomoy/NCG-PEP-code.

Chapter 4: Exterior-point Optimization for Sparse and Low-rank Optimization. In Chapter 4 of this thesis, we present the nonconvex exterior-point optimization solver (NExOS), which is a first-order algorithm tailored to sparse and low-rank optimization problems. Many optimization problems of substantial current interest can be formulated as sparse or low-rank optimization problems. Sparse optimization problems, i.e., optimization problems with sparsity constraints have found applications in gene expression analysis [17, pp. 2–4], sparse regression [18, pp. 155–157], signal transmission and recovery [19, 20], hierarchical sparse polynomial regression [21], and best subset selection [22]. On the other hand low-rank optimization problems, i.e., problems where the decision variables are low-rank matrices, have found applications in collaborative filtering [18, pp. 279-281], design of online recommendation systems [23, 24], bandit optimization [25], data compression [26–28], and low rank kernel learning [29].

In many case, these sparse and low-rank optimization problems correspond to minimizing a convex function over a nonconvex constraint set, where the set can be decomposed as the intersection of a compact convex set and a nonconvex set involving sparse or low-rank constraints. NExOS finds a locally optimal point of the original problem by solving a sequence of penalized problems with strictly decreasing penalty parameters by exploiting the nonconvex geometry of such problems. NExOS solves each penalized problem by applying a first-order algorithm (Douglas-Rachford splitting), and we show that it converges linearly to a local minimum of the corresponding penalized formulation under mild regularity conditions. We also prove that the local minima of the penalized problems converge to a local minimum of

the original problem as the penalty parameter goes to zero.

We implement NExOS in the open-source `Julia` package `NExOS.jl` and test it extensively on many synthetic and real-world instances of different sparse and low-rank optimization problems of substantial current interest: sparse regression problem, affine rank minimization problem, and low-rank factor analysis problem. We demonstrate that NExOS computes solutions very quickly, where the quality of the found solutions are competitive with or better than specialized algorithms on various performance measures. `NExOS.jl` is available at:

https://github.com/Shuvomoy/NExOS.jl

Chapter 2

Branch-and-Bound Performance Estimation Programming: A Unified Methodology for Constructing Optimal Optimization Methods

2.1 Introduction

Since the pioneering work of Nesterov and Nemirovsky on accelerated gradient methods [30] and information-based complexity [31, 32], finding efficient and optimal first-order methods has been the focus in the study of large-scale optimization. Recently, renewed vitality was injected into this classical line of research by the emergence of computer-assisted methodologies following the Performance Estimation Problem (PEP) of Drori and Teboulle [4]. The celebrated accelerated gradient method by Nesterov was improved by a constant

factor in [8, 9, 33], and entirely novel acceleration mechanisms, distinct from Nesterov's, have been discovered [34–37]. These computer-assisted methodologies, roughly speaking, pose the problem of analyzing an efficient method as a convex semidefinite program, and the convexity provides certain algorithmic guarantees.

However, the convexity in the formulation simultaneously serves as a limitation. The aforementioned works presented several ingenious changes of variables, relaxations, and reformulations to retain convexity, but such efforts cover only a handful of setups. When these techniques do not apply, the prior methodologies become inapplicable.

Contribution. This work presents the Branch-and-Bound Performance Estimation Programming (BnB-PEP), a methodology for constructing optimal first-order methods for convex and nonconvex optimization in a tractable and unified manner. We formulate the problem of finding the optimal optimization method as a nonconvex quadratically constrained quadratic problem (QCQP). By directly confronting the nonconvexity of the QCQPs in consideration, BnB-PEP offers significantly more flexibility and removes the many limitations of the prior PEP-based methodologies. We then provide a customized spatial branch-and-bound algorithm that enables us to solve such QCQPs to certifiable global optimality in a practical time scale. The customization speeds up the branch-and-bound algorithm, compared to the latest off-the-shelf implementations, by orders of magnitude, reducing runtimes from hours to seconds and weeks to minutes. We apply the BnB-PEP methodology to several setups for which the prior methodologies do not apply and construct methods with bounds improving upon prior state-of-the-art results. Finally, we use the BnB-PEP methodology to find proofs with potential function structures, thereby systematically generating analytical convergence proofs.

2.1.1 Prior work

The performance estimation methodology, initiated by Drori and Teboulle [4], formulates the worst case-performance of an optimization method as an optimization problem itself and upper bounds this performance through a semidefinite program (SDP) "relaxation". Taylor, Hendrickx, and Glineur then showed that the SDP formulation is, in fact, tight (not a relaxation) through the notion of convex interpolation [5]. Lessard, Recht, and Packard combined the notion of the performance estimation methodology with control-theoretic notions through their integral quadratic constraints (IQC) formulation [10]. Taylor, Van Scoy, and Lessard then showed that IQCs could be seen as a feasible solution to performance estimation problems finding optimal linear convergence rate through Lyapunov functions [7]. Taylor and Bach extended this observation through a methodology that uses the performance estimation approach to find the optimal sublinear rates through potential functions [38].

This advancement in the the performance estimation methodology has led to the discovery of many novel methods and analyses. Drori and Teboulle numerically constructed the optimized gradient method (OGM) [4] and Kim and Fessler found its analytical description [8]. OGM surpasses Nesterov's fast gradient method by a constant factor and Drori showed that OGM is exactly optimal through an exact matching complexity lower bound [39]. Drori and Taylor constructed efficient first-order methods that utilize 1D and 3D exact line searches using a span-search variant of the performance estimation methodology [40]. Van Scoy, Freeman, and Lynch constructed the triple momentum method, a method that surpasses the Nesterov's fast gradient method for the strongly convex setup by a constant factor, using the IQC methodology [33]. Taylor and Drori constructed the information-theoretic exact method (ITEM), further improving upon the triple momentum method [9]. Drori and Taylor showed that ITEM is exactly optimal through an exact matching complexity lower bound [41]. Kim

and Fessler constructed OGM-G, which has the best known rate for reducing the gradient magnitude in the smooth convex setup [34]. Finally, the performance estimation methodology has also been utilized for constructing methods with inexact evaluations [42–44], analyzing methods in the composite minimization setup [45, 46], analyzing methods with exact line search [47], analyzing monotone operator and splitting methods [35, 36, 48, 49], and analyzing acceleration in the mirror descent setup [50].

Prior work constructing efficient methods based on the performance estimation methodology relies on two conceptual stages. The first stage poses the *inner* problem as finding the worst-case performance of a given method and formulates the inner problem as a convex SDP through some ingenious change of variables and SDP duality. The second stage constructs an *outer* problem that minimizes the aforementioned worst-case performance as a function of the method. An equivalent view is that the inner problem finds a convergence proof and the outer problem finds the algorithm with the smallest (best) guarantee established by the convergence proof of the inner problem. While the inner optimization problem is convex, setups for which the outer minimization problem is convex are quite rare. As we detail in §2.3.1.2, prior work circumvents this nonconvexity within the scope of convex optimization through relaxations and heuristics. However, these prior techniques do not always apply, especially when the underlying optimization problem is nonconvex. The setups of §2.6.2 and §2.6.3 of this work are such examples.

2.1.2 Organization

This chapter is organized as follows. In §2.2, we present the necessary background and describe our problem setup. In §2.3, we illustrate the BnB-PEP methodology by applying it on a concrete problem instance of constructing the optimal fixed-step first-order method for reducing the gradient of a strongly convex and smooth function. Our discussion up to

§2.3.1.1 follows prior approaches, and our novel contribution starts in §2.3.1.2. In §2.4, we present customizations of the spatial branch-and-bound algorithm that enables us to solve the QCQPs that construct optimal optimization methods in a practical time scale. In §2.5, we present the generalized formulation of our methodology. In §2.6, we demonstrate the effectiveness of our methodology through several applications. In §2.6.1, we construct the optimal gradient method without momentum for reducing function value in the smooth convex setup and demonstrate that it outperforms the best known method without momentum. In §2.6.2, we construct the optimal method for reducing gradient norm of smooth nonconvex functions and demonstrate that it outperforms the prior best known method [51]. In §2.6.3, we design an optimized first-order method with respect to a suitable potential function for reducing the (sub)gradient norm of nonsmooth weakly convex functions and demonstrate that it outperforms the prior best known method [52, Theorem 3.1]. Additionally, in §2.6.3.3, we present a systematic approach to generate analytical proofs from the solutions obtained through our methodology, extending the approach of Taylor and Bach [38] to nonconvex and nonsmooth setups.

2.1.3 Computational setup

For scientific reproducibility, we open-source our codes to generate all the numerical results presented in this chapter at the link:

https://github.com/Shuvomoy/BnB-PEP-code

Unless otherwise specified, we performed our numerical experiments on a laptop computer running Windows 10 Pro with Intel Core i7-8650U CPU with 16 GB of RAM. We used `JuMP`—a domain-specific modeling language for mathematical optimization embedded in the open-source programming language `Julia` [53]—to model the optimization problems. Our

proposed algorithm uses the following solvers: `Mosek 9.3` [54] (free for academic use), `Ipopt 3.12.11` [55] (open-source), `KNITRO 13.0.0` [56] (free for academic use), and `Gurobi 10` [57] (free for academic use).

2.2 Background and problem setup

Write $\mathbb{R}^d$ for the underlying Euclidean space, even though our results and formulations extend to the setup where the underlying setup is a Hilbert space [48]. Write $\langle \cdot \mid \cdot \rangle$ and $\|\cdot\|$ to denote the standard inner product and norm on $\mathbb{R}^d$. For $a, b \in \mathbb{N}$, denote $[a : b] = \{a, a+1, a+2, \ldots, b-1, b\}$. Write $\mathbb{R}^{m\times n}$ for the set of $m \times n$ matrices, $\mathbb{S}^n$ for the set of $n \times n$ symmetric matrices, and $\mathbb{S}^n_+$ for the set of $n \times n$ positive-semidefinite matrices. We use the standard notation $e_i \in \mathbb{R}^d$ for the unit vector having a single 1 as its i-th component. Write $(\cdot \odot \cdot) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^{d\times d}$ to denote the symmetric outer product, that is, for any $x, y \in \mathbb{R}^d$:

$$x \odot y = \frac{1}{2}\left(xy^\top + yx^\top\right).$$

We follow standard convex-analytical definitions [58–61]. A set $S \subseteq \mathbb{R}^d$ is convex if for any $x, y \in S$ and $\theta \in [0, 1]$, we have $\theta x + (1-\theta)y \in S$. A function $f\colon \mathbb{R}^d \to \mathbb{R}$ is convex if

$$f\left(\theta x + (1-\theta)y\right) \leq \theta f(x) + (1-\theta) f(y)$$

for all $x, y \in \mathbb{R}^d$ and $\theta \in (0, 1)$.

The abstract subdifferential of $f\colon \mathbb{R}^d \to \mathbb{R}$ at x, denoted by $\partial f(x)$, is defined to satisfy the following properties [62]:

(i) If f is convex, then the abstract subdifferential is the usual convex subdifferential, i.e.,

$$\partial f(x) = \{g \in \mathbb{R}^n \mid f(y) \geq f(x) + \langle g \mid y - x \rangle, \, \forall \, y \in \mathbb{R}^n\}.$$

(ii) If f is continuously differentiable at x, then its abstract subdifferential at x just contains the gradient $\nabla f(x)$, i.e., $\partial f(x) = \{\nabla f(x)\}$.

(iii) If f attains a local minimum at x, then $0 \in \partial f(x)$.

(iv) For all $y \in \mathbb{R}^d$ and $\beta \in \mathbb{R}$,

$$\partial \left(f(\cdot) + \frac{\beta}{2} \| \cdot - y \|^2 \right) = \partial f(\cdot) + \beta(\cdot - y).$$

The Clarke–Rockafellar subdifferential [63, §1.2], Mordukhovich subdifferential [64, §1.3], and Fréchet subdifferential [65, page 132] are all instances of the abstract subdifferential [62, page 70]. Whenever we say *subdifferential* in this chapter, we are referring to the abstract subdifferential. Because our analyses use only the properties of the abstract subdifferential, our results apply to all instances of the abstract subdifferential. We write $f'(x)$ to denote an element of $\partial f(x)$.

We say a function f is L-smooth if it is differentiable everywhere and ∇f is L-Lipschitz continuous. We say a function f is μ-strongly convex if $f(\cdot) - (\mu/2)\| \cdot \|^2$ is convex. We say a function f is ρ-weakly convex if $f + (\rho/2)\| \cdot \|^2$ is convex. We say a function f has L-bounded subgradients if $\|g\| \leq L$ for all $g \in \partial f(x)$ and $x \in \mathbb{R}^d$.

2.2.1 Quadratically constrained quadratic program (QCQP)

A QCQP is defined as:

$$p^\star = \left(\begin{array}{lll} \underset{x \in \mathbb{R}^q}{\text{minimize}} & c^\top x + x^\top Q_0 x & \\ \text{subject to} & a_i^\top x + x^\top Q_i x \leq b_i, & i \in [1:m], \\ & a_j^\top x + x^\top Q_j x = b_j, & j \in [m+1:p], \end{array} \right) \tag{2.1}$$

where $x \in \mathbb{R}^q$ is the decision variable. The matrices $Q_0, Q_1, \ldots, Q_p \in \mathbb{R}^{q \times q}$ are symmetric, but not necessarily positive-semidefinite. Therefore, this problem is nonconvex.

Practical tractability of QCQPs. The QCQP problem class is NP-hard and therefore has no known polynomial-time algorithm [66, pp. 565–567]. However, such theoretical worst-case intractability does not necessarily imply that specific problem instances are not *practically tractable* [67, Chapter 1].

Branch-and-bound solvers have experienced astounding speedup in the past few decades. In the last thirty years, branch-and-bound solvers for mixed-integer optimization (MIO) problems have achieved an algorithmic speedup of approximately 1,250,000 and a hardware speedup of approximately 1,560,000, resulting in an overall speedup factor of approximately 2 trillion [67, page 5]. While these speedup factors are for MIO and not for QCQP, the speedup factors for QCQP solvers have followed a similar trend since the recent (2019) incorporation of QCQPs in commercial solvers [68]. For example, in less than three years, `Gurobi`'s spatial branch-and-bound algorithm has achieved a machine-independent speedup factor of 175.5 [57, 69].

This remarkable speedup has rendered previously intractable problems practically tractable. Furthermore, one can often significantly speed up the spatial branch-and-bound algorithm

by customizing it to exploit specific problem structure. We present such customizations for the BnB-PEP Algorithm in §2.4 and §2.5, and demonstrate that the speedup is absolutely essential for the BnB-PEP Algorithm to be used practically.

QCQP solvers for local and global solutions. Since QCQPs have twice-continuously differentiable objectives and constraints, one can use interior-point solvers such as `KNITRO` [56] or `Ipopt` [70] to compute locally optimal solutions under certain regularity conditions [71, Theorem 4][72, §3.2]. On the other hand, one can use the spatial branch-and-bound algorithm implemented in solvers such as `Gurobi` [57] to find globally optimal solutions of nonconvex QCQPs and to certify their optimality in finite time.

2.2.2 Problem setup

Consider the unconstrained minimization problem

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad f(x), \tag{2.2}$$

where f is smooth or nonsmooth, convex or nonconvex.

For the sake of simplicity, we assume that f has a global minimizer $x_\star$ (not necessarily unique). Optimization problems with further structure, such as problems with constraints and problems whose objective are sums of functions, can also be considered. However, we restrict our discussion to this setup of unconstrained minimization for the sake of simplicity.

Function class $\mathcal{F}$. An optimization method is usually designed for a specific class of functions. In this work, we use the BnB-PEP methodology with function classes listed in Table 2.1. More generally, we can use the BnB-PEP methodology with quadratically representable function classes, a notion we further discuss in §2.A.1 of the appendix at the

Function class	Notation
L-smooth Convex ($0 < L < \infty$)	$\mathcal{F}_{0,L}$
L-smooth and μ-strongly convex ($0 \le \mu < L < \infty$)	$\mathcal{F}_{\mu,L}$
L-smooth Nonconvex ($0 < L < \infty$)	$\mathcal{F}_{-L,L}$
ρ-weakly convex with L-bounded subgradients ($\rho > 0, L > 0$)	$\mathcal{W}_{\rho,L}$

Table 2.1: Function classes considered in this chapter.

end of this chapter.

Fixed-step first-order method $\mathcal{M}_N$. We consider fixed-step first-order methods, which include most subgradient methods and accelerated gradient methods [9]. A method is said to be a *fixed-step first-order method* (FSFOM) with N steps if it takes in a function f and a starting point $x_0 \in \mathbb{R}^d$ as input and produces its iterates with:

$$x_i = x_{i-1} - \sum_{j=0}^{i-1} s_{i,j} f'(x_j) \tag{2.3}$$

for $i \in [1:N]$, where $f'(x_j) \in \partial f(x_j)$ is a subgradient of f at x_j for $j \in [0:N-1]$. We can equivalently express the FSFOM (2.3) with

$$x_i = x_0 - \sum_{j=0}^{i-1} \overline{s}_{i,j} f'(x_j),$$

where $\{s_{i,j}\}_{0\le j<i\le N}$ and $\{\overline{s}_{i,j}\}_{0\le j<i\le N}$ are related by

$$\overline{s}_{i,j} = \begin{cases} s_{i,i-1}, & \text{if } j = i-1, \\ \overline{s}_{i-1,j} + s_{i,j}, & \text{if } j \in [0:i-2] \end{cases} \tag{2.4}$$

for $0 \le j < i \le N$. Write $s = \{s_{i,j}\}_{0\le j<i\le N}$ and $\overline{s} = \{\overline{s}_{i,j}\}_{0\le j<i\le N}$ to denote the collection of stepsizes. The stepsizes s or $\overline{s}$ may depend on the function class $\mathcal{F}$ and the value of N,

but are otherwise predetermined. In particular, they may not depend on function values or gradients observed throughout the method. Write $\mathcal{M}_N$ to denote the set of all FSFOMs with N steps. We will soon formulate the problem of finding an optimal FSFOM in $\mathcal{M}_N$ as an optimization problem itself, and the stepsizes s or $\overline{s}$ will serve as the decision variables.

The notion of fixed-step *linear* first-order methods extend these definitions to accommodate proximal methods and conditional gradient methods [73, pp. 118-119]. Our BnB-PEP methodology also directly applies to these generalizations, but we restrict our discussion to FSFOMs for the sake of simplicity.

Performance measure $\mathcal{E}$ and initial condition $\mathcal{C}$. For notational convenience, define the index sets

$$I_N = \{0, 1, \ldots, N\}, \qquad I_N^\star = \{0, 1, \ldots, N, \star\}.$$

Throughout this chapter, we will use $\star$ as the index corresponding to the optimal point. Write $\mathcal{E}$ to denote the performance measure that evaluates a method $M \in \mathcal{M}_N$ on a specific function $f \in \mathcal{F}$ with a starting point x_0. We require that $\mathcal{E}$ depends only on iterates $\{x_0, \ldots, x_N\}$, a globally optimal solution $x_\star$ to (2.2), and the values and (sub)gradients of f at the points $x_0, x_1, \ldots, x_N, x_\star$. In other words, $\mathcal{E}$ may depend on the solution $x_\star$ and zero- and first-order information the FSFOM observes, but may not depend on other unobserved information of f. Commonly considered performance measures are

$$\mathcal{E}\left(\{x_i, f'(x_i), f(x_i)\}_{i \in I_N^\star}\right) = f(x_N) - f(x_\star)$$

or

$$\mathcal{E}\left(\{x_i, \nabla f(x_i), f(x_i)\}_{i \in I_N^\star}\right) = \|\nabla f(x_N)\|^2$$

when f is differentiable.

To obtain a meaningful rate on the methods, we impose a suitable condition on the initial iterate x_0, which we abstractly express as

$$\mathcal{C}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) \le 0.$$

Commonly considered initial conditions are

$$\mathcal{C}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) = \|x_0 - x_\star\|^2 - R^2$$

or

$$\mathcal{C}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) = f(x_0) - f(x_\star) - R^2,$$

where $R > 0$.

Worst-case performance $\mathcal{R}$. The worst-case performance or the rate of the method $M \in \mathcal{M}_N$ is obtained by maximizing $\mathcal{E}$ over functions in $\mathcal{F}$. More formally, we define

$$\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) = \left(\begin{array}{l} \text{maximize} \quad \mathcal{E}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) \\ \text{subject to} \\ f \in \mathcal{F}, \\ x_\star \text{ is a globally optimal solution to (2.2)}, \\ \{x_i\}_{i\in[1:N]} \text{ is generated by FSFOM } M \text{ with initial point } x_0, \\ \mathcal{C}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) \le 0, \end{array}\right) \qquad (\mathcal{O}^{\text{inner}})$$

where f, $x_0, \ldots, x_N$, and $x_\star$ are the decision variables. We set $x_\star = 0$ and $f(x_\star) = 0$, which incurs no loss of generality because the function classes in Table 2.1 and the FSFOM in consideration are closed and invariant under shifting variables and function values.

We will soon show that evaluating the worst-case performance of a given method $M \in \mathcal{M}_N$ by solving ($\mathcal{O}^{\text{inner}}$) can be represented as a (finite-dimensional convex) semidefinite-program.

Optimal FSFOM. An *optimal* FSFOM $M_N^\star \in \mathcal{M}_N$ for a given performance measure $\mathcal{E}$ over function class $\mathcal{F}$ subject to the initial condition $\mathcal{C}$ is a solution to the following minimax optimization problem:

$$\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) = \underset{M \in \mathcal{M}_N}{\text{minimize}} \quad \mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right). \tag{$\mathcal{O}^{\text{outer}}$}$$

Finding an optimal FSFOM $M_N^\star \in \mathcal{M}_N$ by solving ($\mathcal{O}^{\text{outer}}$) is, in general, a nonconvex problem. The BnB-PEP methodology formulates ($\mathcal{O}^{\text{outer}}$) as a (nonconvex) QCQP and solves it to certifiable global optimality using a spatial branch-and-bound algorithm in a practically tractable manner. In §2.3, we illustrate our methodology by describing it for a concrete problem instance. In §2.5, we present the general form of the the methodology.

2.3 BnB-PEP for strongly convex smooth minimization

This section demonstrates the BnB-PEP methodology on a concrete instance for which prior methodologies do not apply. The general BnB-PEP methodology is presented in §2.5.

Specifically, we find the optimal FSFOM for reducing the gradient of μ-strongly convex L-smooth functions, with $0 \leq \mu < L \leq \infty$. In other words, we choose the function class $\mathcal{F} = \mathcal{F}_{\mu,L}$ and performance measure $\mathcal{E} = \|\nabla f(x_N)\|^2$. Further, we choose the initial condition $\mathcal{C} = \|x_0 - x_\star\|^2 - R^2 \leq 0$ with $R > 0$. Then, an optimal FSFOM is a solution of the following instance of ($\mathcal{O}^{\text{outer}}$):

$$\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) = \underset{M \in \mathcal{M}_N}{\text{minimize}} \quad \mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right).$$

2.3.1 Optimal optimization method from BnB-PEP-QCQP

We formulate the outer problem as a (nonconvex) QCQP, which we refer to as the BnB-PEP-QCQP, in the following two steps. In §2.3.1.1, we formulate the inner problem ($\mathcal{O}^{\text{inner}}$) as a convex SDP. This first step follows the approach of [4, 6]. Then in §2.3.1.2 formulates the outer problem ($\mathcal{O}^{\text{outer}}$) as a QCQP. This second step is is novel.

2.3.1.1 Formulating the inner problem ($\mathcal{O}^{\text{inner}}$) as a convex SDP

Infinite-dimensional inner optimization problem. By setting $s_{i,j} = \frac{h_{i,j}}{L}$ in (2.3), parameterize FSFOMs in $\mathcal{M}_N$ as

$$x_i = x_{i-1} - \frac{1}{L}\sum_{j=0}^{i-1} h_{i,j} f'(x_j) \tag{2.5}$$

for $i \in [1:N]$. Write $h = \{h_{i,j}\}_{0\leq j<i\leq N}$. Then ($\mathcal{O}^{\text{inner}}$) becomes

$$\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{lll}
\text{maximize} & \|\nabla f(x_N)\|^2 & \\
\text{subject to} & f \in \mathcal{F}_{\mu,L}, & \\
& \nabla f(x_\star) = 0, & \\
& x_i = x_{i-1} - \frac{1}{L}\sum_{j=0}^{i-1} h_{i,j} \nabla f(x_j), & i \in [1:N], \\
& \|x_0 - x_\star\|^2 \leq R^2, & \\
& x_\star = 0,\ f(x_\star) = 0, &
\end{array} \right)
\end{aligned} \tag{2.6}$$

where $f, x_0, \ldots, x_N$ are the decision variables. As is, f is an infinite-dimensional decision variable.

Reparametrization from $\mathcal{F}_{\mu,L}$ to $\mathcal{F}_{0,L-\mu}$. Next, we use the following lemma to reparameterize ($\mathcal{O}^{\text{inner}}$), defined with function class $\mathcal{F}_{\mu,L}$, into an equivalent problem with the

$(L-\mu)$-smooth convex function class $\mathcal{F}_{0,L-\mu}$. The benefit of the reparametrization is that the final problem becomes more compact.

Lemma 2.1 (Reparametrization from $\mathcal{F}_{\mu,L}$ to $\mathcal{F}_{0,L-\mu}$ [9, §3.2])**.** *Consider $f \in \mathcal{F}_{\mu,L}$ where $0 \le \mu \le L \le \infty$ with a minimizer $x_\star$. Consider an FSFOM with f and $\{h_{i,j}\}_{0\le j<i\le N}$ as defined in (2.5). Define $\widetilde{f} \coloneqq f - (\mu/2)\|\cdot - x_\star\|^2$ and an array of parameters $\{\alpha_{i,j}\}_{0\le j<i\le N}$*

$$\alpha_{i,j} = \begin{cases} h_{i,i-1}, & \text{if } j = i-1, \\ \alpha_{i-1,j} + h_{i,j} - \frac{\mu}{L}\sum_{k=j+1}^{i-1} h_{i,k}\alpha_{k,j}, & \text{if } j \in [0:i-2], \end{cases}$$

where $i \in [1:N]$ and $j \in [0:i-1]$. Then (i) $\widetilde{f} \in \mathcal{F}_{0,L-\mu}$ if and only if $f \in \mathcal{F}_{\mu,L}$, (ii) $x_\star \in \operatorname{argmin} \widetilde{f}$, and (iii) the FSFOM (2.5) is equivalent to

$$x_i = x_\star + (x_0 - x_\star)\left(1 - \frac{\mu}{L}\sum_{j=0}^{i-1}\alpha_{i,j}\right) - \sum_{j=0}^{i-1}\frac{\alpha_{i,j}}{L}\nabla\widetilde{f}(x_j)$$

for $i \in [1:N]$.

Reformulated infinite-dimensional maximization problem. Using Lemma 2.1, we reformulate ($\mathcal{O}^{\text{inner}}$) as

$$
\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{l}
\text{maximize} \quad \|\nabla \widetilde{f}(x_N)\|^2 + \mu^2 \|x_N - x_\star\|^2 + 2\mu \left\langle \nabla \widetilde{f}(x_N) \mid x_N - x_\star \right\rangle \\
\text{subject to} \\
\widetilde{f} \in \mathcal{F}_{0, L-\mu}, \\
\nabla \widetilde{f}(x_\star) = 0, \\
x_i = x_0 \left(1 - \frac{\mu}{L} \sum_{j=0}^{i-1} \alpha_{i,j}\right) - \frac{1}{L} \sum_{j=0}^{i-1} \alpha_{i,j} \nabla \widetilde{f}(x_j), \quad i \in [1:N], \\
\|x_0 - x_\star\|^2 \leq R^2, \\
x_\star = 0, \widetilde{f}(x_\star) = 0,
\end{array} \right)
\end{aligned}
$$

where $\widetilde{f}, x_0, \ldots, x_N$ are the decision variables. The decision variable $\widetilde{f} \in \mathcal{F}_{0,L-\mu}$ is still infinite-dimensional. Write $\alpha = \{\alpha_{i,j}\}_{0 \leq j < i \leq N}$.

Interpolation argument. We now convert the infinite-dimensional optimization problem into finite-dimensional one with the following lemma.

Lemma 2.2 ($\mathcal{F}_{0,L}$-interpolation [9, Theorem 2])**.** *Let I be an index set, and let $\{(x_i, g_i, f_i)\}_{i \in I} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$. Let $L > 0$. There exists $f \in \mathcal{F}_{0,L}$ satisfying $f(x_i) = f_i$ and $g_i \in \partial f(x_i)$ for all $i \in I$ if and only if* [1]

$$
f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2L} \|g_i - g_j\|^2, \quad \forall\, i, j \in I.
$$

[1] This can be viewed as a discretization of the following condition [59, Theorem 2.1.5, Equation (2.1.10)]: $f \in \mathcal{F}_{0,L}$ if and only if

$$
f(y) \geq f(x) + \langle \nabla f(x) \mid y - x \rangle + \frac{1}{2L} \|\nabla f(x) - \nabla f(y)\|^2, \quad \forall x, y \in \mathbb{R}^d.
$$

Finite-dimensional maximization problem. Using Lemma 2.2, reformulate ($\mathcal{O}^{\text{inner}}$) as

$$
\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{l}
\text{maximize} \quad \|g_N\|^2 + \mu^2 \|x_N - x_\star\|^2 - 2\mu \left\langle g_N \mid x_\star - x_N \right\rangle \\
\text{subject to} \\
f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2(L-\mu)} \|g_i - g_j\|^2, \quad i, j \in I_N^\star : i \neq j, \\
g_\star = 0, x_\star = 0, f_\star = 0, \\
x_i = x_0 \left(1 - \frac{\mu}{L} \sum_{j=0}^{i-1} \alpha_{i,j}\right) - \frac{1}{L} \sum_{j=0}^{i-1} \alpha_{i,j} g_j, \quad i \in [1:N], \\
\|x_0 - x_\star\|^2 \leq R^2.
\end{array} \right)
\end{aligned} \tag{2.7}
$$

Now, the decision variables are $\{x_i, g_i, f_i\}_{i \in I_N^\star} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$, where $I_N^\star = \{0, 1, \ldots, N, \star\}$, and the optimization problem is finite-dimensional, although nonconvex. To clarify, we are applying Lemma 2.2 on $\widetilde{f} \in \mathcal{F}_{0,L-\mu}$, rather than $f \in \mathcal{F}_{\mu,L}$. However, we use the symbols $\{f_i\}_{i \in I_N^\star}$, rather than the arguably more consistent $\{\widetilde{f}_i\}_{i \in I_N^\star}$ for the sake of notational conciseness.

Grammian formulation. Next, we formulate ($\mathcal{O}^{\text{inner}}$) as a convex SDP. Let

$$
\begin{aligned}
H &= [x_0 \mid g_0 \mid g_1 \mid \ldots \mid g_N] \in \mathbb{R}^{d \times (N+2)}, \\
G &= H^\top H \in \mathbb{S}_+^{N+2}, \\
F &= [f_0 \mid f_1 \mid \ldots \mid f_N] \in \mathbb{R}^{1 \times (N+1)}.
\end{aligned} \tag{2.8}
$$

Note that $\mathbf{rank}\, G \leq d$. Define the following notation for selecting columns and elements of H and F:

$$\begin{aligned}
&\mathbf{g}_\star = 0 \in \mathbb{R}^{N+2},\ \mathbf{g}_i = e_{i+2} \in \mathbb{R}^{N+2}, \quad i \in [0:N] \\
&\mathbf{x}_0 = e_1 \in \mathbb{R}^{N+2},\ \mathbf{x}_\star = 0 \in \mathbb{R}^{N+2}, \\
&\mathbf{x}_i = \mathbf{x}_0 \left(1 - \frac{\mu}{L} \sum_{j=0}^{i-1} \alpha_{i,j}\right) - \frac{1}{L} \sum_{j=0}^{i-1} \alpha_{i,j} \mathbf{g}_j \in \mathbb{R}^{N+2}, \quad i \in [1:N] \\
&\mathbf{f}_\star = 0 \in \mathbb{R}^{N+1},\ \mathbf{f}_i = e_{i+1} \in \mathbb{R}^{N+1}, \quad i \in [0:N].
\end{aligned} \tag{2.9}$$

This notation is defined so that

$$x_i = H\mathbf{x}_i,\ g_i = H\mathbf{g}_i,\ f_i = F\mathbf{f}_i$$

for $i \in I_N^\star$. Note that $\mathbf{x}_i$ depends on $\{\alpha_{i,j}\}_{j \in [0:i-1]}$ linearly for $i \in [1:N]$. Furthermore, for $i, j \in I_N^\star$, define

$$\begin{aligned}
&A_{i,j}(\alpha) = \mathbf{g}_j \odot (\mathbf{x}_i - \mathbf{x}_j) \in \mathbb{S}^{N+2}, \\
&B_{i,j}(\alpha) = (\mathbf{x}_i - \mathbf{x}_j) \odot (\mathbf{x}_i - \mathbf{x}_j) \in \mathbb{S}_+^{N+2}, \\
&C_{i,j} = (\mathbf{g}_i - \mathbf{g}_j) \odot (\mathbf{g}_i - \mathbf{g}_j) \in \mathbb{S}_+^{N+2}, \\
&a_{i,j} = \mathbf{f}_j - \mathbf{f}_i \in \mathbb{R}^{N+1}.
\end{aligned} \tag{2.10}$$

Note that $A_{i,j}(\alpha)$ is affine and $B_{i,j}(\alpha)$ is quadratic as functions of $\{\alpha_{i,j}\}_{i \in [1:N], j \in [0,i-1]}$. This notation is defined so that

$$\begin{aligned}
&\langle g_j \mid x_i - x_j \rangle = \mathbf{tr}\, G A_{i,j}(\alpha), \\
&\|x_i - x_j\|^2 = \mathbf{tr}\, G B_{i,j}(\alpha), \\
&\|g_i - g_j\|^2 = \mathbf{tr}\, G C_{i,j},
\end{aligned} \tag{2.11}$$

for $i, j \in I_N^\star$.

Using this notation, formulate ($\mathcal{O}^{\text{inner}}$) as

$$
\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{ll}
\text{maximize} & \mathbf{tr}\, G \left(C_{N,\star} + \mu^2 B_{N,\star}(\alpha) - 2\mu A_{\star,N}(\alpha) \right) \\
\text{subject to} & \\
F a_{i,j} + \mathbf{tr}\, G A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} \mathbf{tr}\, G C_{i,j} \leq 0, & i, j \in I_N^\star : i \neq j, \\
-G \preceq 0, \, \mathbf{rank}(G) \leq d, & \\
\mathbf{tr}\, G B_{0,\star} \leq R^2, &
\end{array} \right)
\end{aligned}
$$

where $F \in \mathbb{R}^{1 \times (N+1)}$ and $G \in \mathbb{R}^{(N+2) \times (N+2)}$ are the decision variables. The equivalence relies on the fact that given a $G \in \mathbb{S}_+^{N+2}$ satisfying $\mathbf{rank}(G) \leq d$, there exists a $H \in \mathbb{R}^{d \times (N+2)}$ such that $G = H^\top H$. The argument is further detailed in [5, §3.2]. This formulation is not yet a convex SDP due to the rank constraint $\mathbf{rank}(G) \leq d$.

SDP representation. Next, we make the following large-scale assumption.

Assumption 2.1. *We have $d \geq N + 2$.*

Under this assumption, the constraint $\mathbf{rank}\, G \leq d$ becomes vacuous, since $G \in \mathbb{S}_+^{(N+2)}$. We drop the rank constraint and formulate ($\mathcal{O}^{\text{inner}}$) as a convex SDP

$$\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C})$$

$$
= \left(\begin{array}{l} \text{maximize} \quad \mathbf{tr}\, G \left(C_{N,\star} + \mu^2 B_{N,\star}(\alpha) - 2\mu A_{\star,N}(\alpha) \right) \\ \text{subject to} \\ Fa_{i,j} + \mathbf{tr}\, G A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} \mathbf{tr}\, G C_{i,j} \leq 0, \quad i,j \in I_N^\star : i \neq j, \quad \triangleright \text{ dual var. } \lambda_{i,j} \geq 0 \\ -G \preceq 0, \quad \triangleright \text{ dual var. } Z \succeq 0 \\ \mathbf{tr}\, G B_{0,\star} \leq R^2, \quad \triangleright \text{ dual var. } \nu \geq 0 \end{array} \right) \tag{2.12}
$$

where $F \in \mathbb{R}^{1\times(N+1)}$ and $G \in \mathbb{R}^{(N+2)\times(N+2)}$ are the decision variables. We denote the corresponding dual variables on the right hand side of the constraints with $\triangleright$ dual var. for later use.

We emphasize that dropping the rank constraint is not a relaxation; the optimization problem (2.12) and its solution is independent of the dimension d, provided that the large-scale assumption $d \geq N+2$ holds. See [5, §3.3] for further discussion.

Dualization. Next we use convex duality to formulate ($\mathcal{O}^{\text{inner}}$), originally a maximization problem, as a minimization problem. Take the dual of (2.12) to get

$$
\begin{aligned} &\overline{\mathcal{R}}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\ &= \left(\begin{array}{l} \text{minimize} \quad \nu R^2 \\ \text{subject to} \\ \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} a_{i,j} = 0, \\ \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha) + 2\mu A_{\star,N}(\alpha) + \\ \quad \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left(A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\ Z \succeq 0, \\ \nu \geq 0, \ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \end{array} \right) \end{aligned} \tag{2.13}
$$

where $\nu \in \mathbb{R}$, $\lambda = \{\lambda_{i,j}\}_{i,j \in I_N^\star : i \neq j}$, and $Z \in \mathbb{S}_+^{N+2}$ are the decision variables. (Note, we write $\overline{\mathcal{R}}$ rather than $\mathcal{R}$ here.) We call λ, ν, and Z the *inner-dual variables*.

By weak duality of convex SDPs, we have

$$\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \leq \overline{\mathcal{R}}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}).$$

In convex SDPs, strong duality holds often but not always. For the sake of simplicity, we assume strong duality holds.

Assumption 2.2. *Strong duality holds between* (2.12) *and* (2.13)*, i.e.,*

$$\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) = \overline{\mathcal{R}}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}).$$

This assumption can be removed in most cases using the line of reasoning of [74, Claim 4], but the technique is tedious. Since strong duality for convex SDPs "usually" holds, we argue there is little utility in pursuing this direction. Nevertheless, in a strict sense, the assumption constitutes a gap in our mathematical arguments. We detail the implication of this gap in §2.A.2 of the appendix. In any case, strong duality holds for "generic" FSFOMs [5, Theorem 6], so one can safely use the BnB-PEP methodology with confidence that the obtained FSFOM will be optimal among the "nice" generic FSFOMs.

Now we have arrived at the end of the formulation based on prior works [4, 6], and, at this point, our formulations diverge.

2.3.1.2 Formulating the outer problem ($\mathcal{O}^{\text{outer}}$) as a QCQP

With ($\mathcal{O}^{\text{inner}}$) formulated as a minimization problem, the outer optimization problem ($\mathcal{O}^{\text{outer}}$) becomes a joint minimization over the inner dual variables and the FSFOM parameters α. However, the outer minimization problem is not convex in all of the variables, even though the inner problem is.

Prior work circumvents this nonconvexity within the scope of convex optimization. Prior work on OGM [4, 8], ITEM [9], ORC-F$_\flat$ [74, §4], and OBL-F$_\flat$ [74, §5.1] convexify ($\mathcal{O}^{\text{outer}}$) into an SDP through appropriate relaxations and changes of variables. The relaxation discards certain inequalities, a process discussed in §2.4.3. Due to the relaxation, the solution FSFOM of the relaxed SDP is not necessarily the exact optimal FSFOM. Exact optimality of OGM [39] and ITEM [41] were proved through separate exact matching complexity lower bounds. Even though ORC-F$_\flat$ and OBL-F$_\flat$ are optimal solutions of the relaxed SDP, they are not optimal FSFOMs, as their guarantees are worse than that of OGM. Prior work on OGM-G [34], M-OGM-G [75], OBL-G$_\flat$ [74], APPM [35, 36], and SM-APPM [76] formulate ($\mathcal{O}^{\text{outer}}$) as bi-convex optimization problems, as problems with bilinear matrix inequalities (BMIs), after relaxations discarding certain inequalities. Although bi-convex problems (which are nonconvex) do not have provably efficient algorithms, prior work have obtained FSFOMs using the alternating minimization heuristic. Exact optimality of APPM and SM-APPM were proved through separate exact matching complexity lower bounds [76]. OGM-G is presumed but not proven to be exactly optimal. M-OGM-G and OBL-G$_\flat$ are not optimal FSFOMs in the usual sense as their guarantees are worse than that of OGM-G. For the setups we consider, especially the setup of §2.6.2 and §2.6.3, these prior techniques do not apply and the optimization over the FSFOM cannot be formulated as a convex nor a bi-convex optimization problem (to the best of our knowledge) even after appropriate relaxations.

Formulating ($\mathcal{O}^{\text{outer}}$) as a QCQP. The nonconvex outer optimization problem ($\mathcal{O}^{\text{outer}}$) minimizes over α in addition to the inner dual variables of (2.13). We confront the nonconvexity directly by formulating ($\mathcal{O}^{\text{outer}}$) as a (nonconvex) QCQP and solving it with spatial branch-and-bound algorithms. We do not discard constraints or use any relaxation. To this end, we replace the semidefinite constraint with a quadratic constraint via the Cholesky factorization.

Lemma 2.3 ([77, Corollary 7.2.9])**.** *A matrix $Z \in \mathbb{S}^n$ is positive semidefinite if and only if it has a Cholesky factorization $PP^\top = Z$, where $P \in \mathbb{R}^{n\times n}$ is lower triangular with nonnegative diagonals.*

In raw index form, the conditions of Lemma 2.3, applied to the present setup, have the following equivalent representations:

$$\left(\begin{array}{l} P \text{ is lower triangular with nonnegative diagonals,} \\ PP^\top = Z. \end{array}\right)$$
$$\Leftrightarrow \left(\begin{array}{ll} P_{j,j} \geq 0, & j \in [1:N+2], \\ P_{i,j} = 0, & 1 \leq i < j \leq N+2, \\ \sum_{k=1}^{j} P_{i,k}P_{j,k} = Z_{i,j}, & 1 \leq j \leq i \leq N+2. \end{array}\right)$$

We now formulate ($\mathcal{O}^{\text{outer}}$), the problem of finding an optimal FSFOM, as the following QCQP

$$\mathcal{R}^\star(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C})$$

$$
= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} = 0, \\
\nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha) + 2\mu A_{\star,N}(\alpha) + \\
\qquad \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\
P \text{ is lower triangular with nonnegative diagonals}, \\
PP^\top = Z, \\
\nu \geq 0, \ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j,
\end{array} \right) \tag{2.14}
$$

where λ, ν, Z, P, and α are the decision variables. We name this optimization problem the BnB-PEP-QCQP.

2.3.2 Solving the BnB-PEP-QCQP using the BnB-PEP Algorithm

We now solve the BnB-PEP-QCQP (2.14) to certifiable optimality in a practical time scale via the BnB-PEP Algorithm, stated as Algorithm 1. The algorithm has 3 stages: Stage 1 finds a feasible point, Stage 2 uses an interior-point solver to find a locally optimal solution, and Stage 3 uses a spatial branch-and-bound solver to find a globally optimal solution.

Details of the BnB-PEP Algorithm. Stage 1 computes a feasible solution of (2.14) by taking advantage of the structure that (2.13) is a convex SDP when the FSFOM is fixed. We set the stepsize to represent gradient descent (GD)

$$
x_i = x_{i-1} - \frac{1}{L} \nabla f(x_{i-1}), \quad i \in [1:N]
$$

which is a suboptimal but reasonable algorithm. By Lemma 2.1, this corresponds to $\alpha_{i,i-1}^{\text{init}} = 1$ for $i \in [1:N]$ and $\alpha_{i,j}^{\text{init}} = 0$ for $j \neq i-1$.

Algorithm 1 BnB-PEP Algorithm: Given (μ, L, R), solves (2.14) to global optimality

Stage 1. Compute a feasible solution.

- Set $\alpha^{\text{init}}_{i,i-1} \leftarrow 1$ and $\alpha^{\text{init}}_{i,j} \leftarrow 0$ for for $i \in [1:N]$, $j \neq i-1$.
- Set $\alpha \leftarrow \alpha^{\text{init}}$ in (2.13) and solve the convex SDP. Denote the computed optimal solution to (2.13) by $\{\nu^{\text{init}}, \lambda^{\text{init}}, Z^{\text{init}}\}$.
- Compute Cholesky decomposition $Z^{\text{init}} = P^{\text{init}}(P^{\text{init}})^\top$.

Stage 2. Compute a locally optimal solution by warm-starting at Stage 1 solution.

- Warm-start (2.14) with $\{\alpha^{\text{init}}, \nu^{\text{init}}, \lambda^{\text{init}}, Z^{\text{init}}, P^{\text{init}}\}$ and solve it to local optimality using the nonlinear interior-point method. Denote the solution by $\{\alpha^{\text{lopt}}, \nu^{\text{lopt}}, \lambda^{\text{lopt}}, Z^{\text{lopt}}, P^{\text{lopt}}\}$.

Stage 3. Compute a globally optimal solution by warm-starting at Stage 2 solution.

- Warm-start (2.14) with $\{\alpha^{\text{lopt}}, \nu^{\text{lopt}}, \lambda^{\text{lopt}}, Z^{\text{lopt}}, P^{\text{lopt}}\}$ and solve it to global optimality using a customized spatial branch-and-bound algorithm described in §2.4. Denote the solution by $\{\alpha^\star, \nu^\star, \lambda^\star, Z^\star, P^\star\}$ and the optimal objective value by $p^\star$.

Return: $\{\alpha^\star, \nu^\star, \lambda^\star, Z^\star, P^\star\}$ and $p^\star$.

Stage 2 computes a locally optimal solution to (2.14) using an interior-point algorithm, warm-starting at the feasible solution corresponding to GD that was produced by Stage 1. When a good warm-starting point is provided, interior-point algorithms can quickly converge to a locally optimal solution [70, 78], [71, §3.3]. If the interior-point algorithm fails to converge, we return the feasible solution from Stage 1. Fortunately, we observe that Stage 2 consistently provides a locally optimal solution.

Stage 3 computes a globally optimal solution using a customized spatial branch-and-bound algorithm, warm-starting at the solution produced by Stage 2. We detail our BnB-PEP-QCQP-specific customization in §2.4. A good warm-starting point provides a tight upper bound of the optimal value and, therefore, significantly accelerates the spatial branch-and-bound algorithm of Stage 3. In our experience, Stage 2 often provides an excellent, even nearly optimal, warm-starting point.

In principle, one can simply use an off-the-shelf implementation of spatial branch-and-bound algorithm such as `Gurobi` [68] to solve (2.14). Spatial branch-and-bound algorithms do compute globally optimal solutions in "finite time", but the default implementation

is impractically slow, as Table 2.4 illustrates. Customizing the spatial branch-and-bound algorithm with problem-specific insights is essential, as discussed in §2.4.

Numerical results. We conduct numerical experiments on the computational environment described in §2.1.3 with parameters $\mu = 0.1$, $L = 1$, and $R = 1$. Due to the scale invariance discussed in [5, §3.5], it suffices to solve the BnB-PEP-QCQP for $L = 1$ and $R = 1$ and find the corresponding optimal stepsize vector $\alpha^\star$ (or $h^\star$) and the associated optimal worst-case performance measure $\|\nabla f(x_N)\|^2$. More specifically, for any other $L > 0$ and $R > 0$, the new optimal stepsize vector will be scaled as $\alpha^\star/L$ (or $h^\star/L$) with corresponding performance measure scaled as $L^2R^2\|\nabla f(x_N)\|^2$. The homogeneity relations for other performance measures and initial conditions can be found at [5, §3.5] and [73, §4.2.5].[2] Tables 2.2 and 2.3 present the results of the BnB-PEP Algorithm. The optimal algorithm indeed outperforms other known algorithms in terms of the worst-case performance measure $\|\nabla f(x_N)\|^2$ with initial condition $\|x_0 - x_\star\| \leq R^2$. Table 2.4 presents runtimes of the BnB-PEP Algorithm. The BnB-PEP-QCQP can be solved in a practical time scale with the BnB-PEP Algorithm, but only when we use the customized spatial branch-and-bound solver of §2.4.

A notable empirical observation is that the FSFOM produced by Stage 2, expected to be locally optimal, is often globally optimal or near-optimal. In this case, Stage 3 serves mainly to certify the global optimality of the warm-starting solution of Stage 2. This fortuitous behavior was observed consistently in our experiments of §2.6 as well. Because Stages 1 and 2 tend to be faster than Stage 3 and because the output of Stage 2 is often globally optimal, one can use the output of Stage 2 as a heuristic without running Stage 3. This can be useful when the goal is to obtain a good method, and there is no need to certify that the method is

[2]The journal version of [5, §3.5] contained a typo in the homogeneity relations, which was later corrected in a subsequent arXiv update https://arxiv.org/pdf/1502.05666.pdf.

N	# variables	# constraints	Worst-case $\|\nabla f(x_N)\|^2$			
			Optimal	GD	ITEM	OGM-$\mathcal{F}_{\mu,L}$
1	20	33	0.1473	0.2244	0.6695	0.2122
2	36	56	0.0409	0.0893	0.3770	0.0835
3	57	85	0.0145	0.0449	0.1933	0.0378
4	83	120	0.005766	0.0257	0.0945	0.0178
5	114	161	0.002459	0.0159	0.0451	0.0085
10	410	456	4.89×10^{-5}	2.58×10^{-3}	1.03×10^{-3}	1.97×10^{-4}
25	2135	2241	5.42×10^{-10}	5.89×10^{-5}	5.5×10^{-7}	7.21×10^{-8}

Table 2.2: Comparison of the optimal method obtained by solving (2.14) with the BnB-PEP Algorithm against other known methods.

optimal.

Table 2.2 compares the performance of the optimal method, obtained with the BnB-PEP Algorithm, against GD (plain gradient desent), ITEM [9], and OGM-$\mathcal{F}_{\mu,L}$ [9]. (ITEM and OGM-$\mathcal{F}_{\mu,L}$ are optimal with respect to different performance measures and therefore are suboptimal when the goal is to reduce the gradient magnitude. The stepsizes of ITEM were taken from page 21 of the first arXiv version of [9], and the stepsizes of OGM-$\mathcal{F}_{\mu,L}$ were taken from [9, §E.1].) We also show the total number of variables and constraints of (2.14) that the BnB-PEP Algorithm works with after the mathematical model described in `JuMP` gets converted to the `MathOptInterface` format [79], which is the standard data structure for representing optimization models in `JuMP`.

Table 2.3 shows the globally optimal stepsizes found by the BnB-PEP Algorithm. To clarify, we obtain the optimal $\alpha^\star$ from the BnB-PEP Algorithm, solve for $h^\star$ with Lemma 2.1, and present $h^\star$ in the table.

Table 2.4 presents the runtimes with and without the customized spatial branch-and-bound

N	$h^\star$
1	$\begin{bmatrix} 1.3837 \end{bmatrix}$
2	$\begin{bmatrix} 1.5018 & \\ 0.0494 & 1.5018 \end{bmatrix}$
3	$\begin{bmatrix} 1.5308 & & \\ 0.0889 & 1.7229 & \\ 0.0109 & 0.0889 & 1.5308 \end{bmatrix}$
4	$\begin{bmatrix} 1.5403 & & & \\ 0.1038 & 1.7926 & & \\ 0.0229 & 0.1751 & 1.7926 & \\ 0.003 & 0.0229 & 0.1038 & 1.5403 \end{bmatrix}$
5	$\begin{bmatrix} 1.5439 & & & & \\ 0.1097 & 1.8187 & & & \\ 0.0286 & 0.2132 & 1.8842 & & \\ 0.0069 & 0.0514 & 0.2132 & 1.8187 & \\ 0.0009 & 0.0069 & 0.0286 & 0.1097 & 1.5439 \end{bmatrix}$
10	$\begin{bmatrix} 1.5465 & & & & & & & & & \\ 0.1141 & 1.8377 & & & & & & & & \\ 0.033 & 0.2426 & 1.9488 & & & & & & & \\ 0.0107 & 0.0786 & 0.3072 & 1.995 & & & & & & \\ 0.0036 & 0.0265 & 0.1037 & 0.3357 & 2.0122 & & & & & \\ 0.0012 & 0.009 & 0.0352 & 0.114 & 0.3437 & 2.0122 & & & & \\ 0.0004 & 0.003 & 0.0117 & 0.0378 & 0.114 & 0.3357 & 1.995 & & & \\ 0.0001 & 0.0009 & 0.0036 & 0.0117 & 0.0352 & 0.1037 & 0.3072 & 1.9488 & & \\ 0.0 & 0.0002 & 0.0009 & 0.003 & 0.009 & 0.0265 & 0.0786 & 0.2426 & 1.8377 & \\ 0.0 & 0.0 & 0.0001 & 0.0004 & 0.0012 & 0.0036 & 0.0107 & 0.033 & 0.1141 & 1.5465 \end{bmatrix}$
25	See Supplementary Information or `Github` repository

Table 2.3: Globally optimal stepsizes obtained by solving (2.14) with the BnB-PEP Algorithm.

Algorithm	BnB-PEP Algorithm runtime			Off-the-shelf `Gurobi` runtime on `MIT Supercloud`
	Stage 1	Stage 2	Stage 3	
$N = 1$	0.004 s	0.130 s	0.081 s	5 h 17 m
$N = 2$	0.007 s	0.147 s	0.110 s	1 d 3 h
$N = 3$	0.007 s	0.153 s	0.512 s	4 d 13 h
$N = 4$	0.015 s	0.192 s	4.602 s	More than a week
$N = 5$	0.017 s	0.330 s	456.685 s	More than a week
$N = 10$	0.26 s	2 m 37 s	1 d 22 h	Does not finish in 2 weeks
$N = 25$	3.2 s	6 m 22 s	3 d 10 h	Does not finish in 2 weeks

Table 2.4: This table compares the runtimes of the BnB-PEP Algorithm executed on a standard laptop with the off-the-shelf spatial branch-and-bound algorithm of `Gurobi` executed on `MIT Supercloud` for $N = 1, \ldots, 5, 10$. For the case $N = 25$, both the BnB-PEP Algorithm and off-the-shelf `Gurobi` were executed on `MIT Supercloud`.

solver of §2.4. The off-the-shelf spatial branch-and-bound algorithm of `Gurobi` was very slow despite running on the `MIT Supercloud Computing Cluster` with 24 Intel-Xeon-Platinum-8260 nodes (has 1152 cores) and 384 GB of RAM running Ubuntu 18.04.6 LTS with Linux 4.14.250-llgrid-10ms kernel [80]. On the other hand, our BnB-PEP Algorithm ran efficiently on both a standard laptop and on `MIT Supercloud`. For $N = 25$, we run both the BnB-PEP Algorithm and the off-the-shelf `Gurobi` on the `MIT Supercloud`. The cases for which the off-the-shelf spatial branch-and-bound algorithm terminated, the results agreed with the results of the BnB-PEP Algorithm.

2.4 Efficient implementation of the BnB-PEP Algorithm

As Table 2.4 illustrates, an off-the-shelf spatial branch-and-bound algorithm applied to BnB-PEP-QCQP is very slow. In this section, we customize the spatial branch-and-bound algorithm to exploit specific problem structure and obtain a speedup that enables us to run

the BnB-PEP Algorithm on a laptop.

In §2.4.1, we briefly review the standard spatial branch-and-bound algorithm. We present the customization that provide significant speedups in §2.4.2. In §2.4.3 we show how to compute the effective index set of the inner-dual-variable and thereby reduce the size of the BnB-PEP-QCQP without losing optimality.

2.4.1 How the spatial branch-and-bound algorithm solves QCQPs

We briefly review the standard spatial branch-and-bound algorithm [68, 72, 81, 82]. We assume the optimization problem admits a finite optimal value, as this is the case in the setups we consider.[3]

The spatial branch-and-bound algorithm uses a divide-and-conquer approach to solve (2.1). The algorithm starts with the *presolve* phase, solving a linear relaxation of (2.1) to obtain valid bounds $l \leq x \leq u$, with $l, u \in \mathbb{R}^q$, that are satisfied by optimal solutions. Then the algorithm performs *branching*, partitioning the feasible region of (2.1) into a finite collection of subregions $F_1, \ldots, F_K$ and considering the subproblems

$$p_k^\star = \left(\begin{array}{lll} \underset{x \in \mathbb{R}^q}{\text{minimize}} & c^\top x + x^\top Q_0 x & \\ \text{subject to} & a_i^\top x + x^\top Q_i x \leq b_i, & i \in [1:m], \\ & a_j^\top x + x^\top Q_j x = b_j, & j \in [m+1:p], \\ & x \in F_k, & \end{array} \right)$$

for $k \in [1:K]$. The best (smallest) among the optimal values $p_1^\star, \ldots, p_K^\star$ is $p^\star$, by definition. The *bounding* part is about how to efficiently solve these subproblems via solving relaxations

[3] The setups of §2.3 and §2.6 satisfy $p^\star < \infty$, since any FSFOM (such as the method that has all 0 stepsizes and therefore does not move) achieves a finite performance measure, and $0 \leq p^\star$, since the objectives are nonnegative. In general, however, there could be pathological BnB-PEPs such that $p^\star = -\infty$ or $p^\star = \infty$.

and how to split these subproblems into smaller subproblems if necessary; we discuss this next.

The central idea is that, while solving a particular subproblem (also a QCQP albeit over a smaller region) might be as hard as solving the original problem, a lower bound and an upper bound of that subproblem is much easier to solve via linear relaxations. Using this idea, first, at the root node of the spatial branch-and-bound tree, a linear relaxation of (2.1) is constructed and solved, which gives a lower bound on $p^\star$, denoted by $\underline{p}^\star$. The tighter this relaxation, the closer $\underline{p}^\star$ is to $p^\star$. In addition to that, the user can *warm-start* the branch-and-bound algorithm by providing a known initial feasible solution to (2.1), which gives an upper-bound on $p^\star$, denoted by $\overline{p}^\star$. Efficient warm-starting procedure that exploit the problem structure can massively speed up branch-and-bound-solvers. The branch-and-bound algorithm during its execution keeps updating $\overline{p}^\star, \underline{p}^\star$. The difference $\overline{p}^\star - \underline{p}^\star$ is called the *gap*, and when this gap is equal to zero (or less than some tolerance ϵ) at some point of the algorithm, we have found the globally (near-)optimal value $p^\star$ of (2.1) along with one (approximately) optimal solution, and the algorithm is terminated. We next discuss how the gap is improved over the course of the algorithm.

Once the subregions have been created, the algorithm picks an active subregion, say F_k (which k to select can be arbitrary, though, in practice, it is usually done via different heuristics in modern solvers), and constructs two linear optimization problems on F_k. These linear formulations are constructed using the McCormick envelopes [83], which provide lower and upper bounds for the quadratic objective and constraints in (2.1), whereas the the linear constraints in (2.1) are kept unaltered. Then three types of linear cuts are added to the linear optimization problems to remove regions that are certain to not contain any optimal solutions [84–86]. Solving these linear optimization problems along with the cuts provides valid lower and upper bounds on the optimal values of (2.1) for the active subregion F_k. Solving these

linear optimization problems on F_k leads to one of the three possibilities below:

1. If the linear optimization problem associated with the lower bound is either infeasible or has an objective value greater than the global upper bound $\overline{p}^\star$, then F_k cannot contain the optimal solution to (2.1). Hence, without solving the QCQP on F_k, we can discard or *prune* the subregion. Such a pruned subregion becomes a permanent *leaf* of the branch-and-bound tree.
2. If both the lower and upper bounds for the subregion are the same, then without directly solving the QCQP on F_k, we have found this subproblem's optimal solution with optimal value $p_k^\star$. This optimal solution on F_k is a feasible solution to the main problem (2.1). It is not necessary to branch on this subregion anymore, and it becomes a permanent leaf of the search tree. If the objective value associated with this new feasible solution leads to an improved upper bound $\overline{p}^\star$ compared to the current incumbent, then the feasible solution on F_k becomes the new incumbent solution. Otherwise, updating the incumbent is not necessary and we simply proceed with the search.
3. If 1 or 2 does not happen, then the subregion F_k is partitioned into smaller subregions by branching again, which are then added to the list of active subregions.

In addition to that, at any point, the algorithm keeps an updated value of the lower bound on $\underline{p}^\star$ by taking the minimum of the best objective values of all the current leaf nodes. On the other hand, the upper bound $\overline{p}^\star$ corresponds to the incumbent solution. As the algorithm explores the active subregions, the gap $\overline{p}^\star - \underline{p}^\star$ keeps getting smaller, and once it is zero or smaller than a certain tolerance ϵ, we have found the global optimal value $p^\star$ of (2.1) along with one optimal solution subject to the tolerance, and the algorithm terminates.

2.4.2 Efficient implementation of the spatial branch-and-bound algorithm

We now customize the spatial branch-and-bound algorithm to efficiently exploit problem structure of the BnB-PEP-QCQP. Our customization of `Gurobi`'s branch-and-bound algorithm [57] uses solver-independent *callback functions*, an interface provided by `JuMP` [53].

Callback functions are user-defined functions provided to the optimization solver that query or modify the state of the optimization process of a solver. Examples of such callback functions include providing custom heuristics to compute better feasible solutions, changing the default branching decision of the branch-and-bound algorithm, or applying on-demand separators to add new constraints only if they are violated by the current solution.

We discuss the generalization of our customization of the spatial branch-and-bound algorithm for arbitrary $\mathcal{E}$, $\mathcal{F}$, and $\mathcal{C}$ in §2.5. We first present the customizations in §2.4.2.1, §2.4.2.2, and §2.4.2.3, and then discuss the observed speedups in §2.4.2.4.

2.4.2.1 Bounds on optimal solutions

Branch and bound algorithms require bounds on the optimization variables to partition the feasible region. If no bound information for a variable is provided in the original formulation (2.1), then the solver obtains a bound by solving a generic linear relaxation during the presolve phase. However, this bound can be of poor quality as a generic solver does not have any problem-specific insight, and a loose bound can cause the solver to waste time in unimportant regions. We show how to significantly speed up the branch-and-bound algorithm by exploiting the structure of (2.1) to obtain tighter bounds.

Implied linear constraints. The constraint $Z = PP^\top$ implies that Z is symmetric positive semidefinite. This in turn implies

$$\begin{aligned} &Z = Z^\top, \\ &\operatorname{diag}(Z) \geq 0, \\ &-\tfrac{Z_{i,i}+Z_{j,j}}{2} \leq Z_{i,j} \leq \tfrac{Z_{i,i}+Z_{j,j}}{2}. \end{aligned} \tag{2.15}$$

where $\operatorname{diag}(Z) \geq 0$ means the $Z_{i,i} \geq 0$ for $i \in [1 : N+2]$. To explain, $Z \succeq 0$ implies that every 1×1 principal submatrix of Z is positive-semidefinite [77, Observation 7.1.2], and this in turn implies the second constraint. Also, $Z \succeq 0$ implies that every the 2×2 principal submatrix of Z is positive-semidefinite, and this in turn implies

$$|Z_{i,j}| \leq \sqrt{Z_{i,i} Z_{j,j}} \Leftrightarrow Z_{i,j}^2 \leq Z_{i,i} Z_{j,j} \tag{2.16}$$

for $i, j \in [1 : N+2]$. Chaining the AM-GM inequality

$$\sqrt{Z_{i,i} Z_{j,j}} \leq \frac{Z_{i,i} + Z_{j,j}}{2},$$

we get the third constraint.

While these implied constraints are indeed mathematically redundant, they are algorithmically indispensable as they provide crucial information that the solver cannot deduce directly. Explicitly incorporating these implied constraints provides significant speedups. Instead of incorporating the tighter convex second-order cone (SOC) constraint (2.16), we opt for the third linear constraint (2.15) in our BnB-PEP-QCQP formulation. This choice avoids a slowdown in the spatial branch-and-bound algorithm, which solves only *linear* relaxations at each node, as detailed in §2.4.1. The linear relaxations are derived from McCormick convex envelopes, which are constructed without considering underlying convexity and are

computationally expensive [72, 81, 82]. Using the SOC constraint (2.16) would result in the spatial branch-and-bound algorithm treating it as a generic quadratic constraint and spending extra time constructing McCormick convex envelopes for it [68, pp. 10–15]. Since the positive semi-definiteness of Z is already modeled by the quadratic constraints $Z = PP^\top$, and their associated convex envelopes are tighter than the ones for SOC constraints, the additional SOC constraints would ultimately lead to a net slowdown. Conversely, the third constraint in (2.15) is linear and can be directly incorporated into the linear relaxations at the nodes without any extra processing time. These constraints differ from those automatically generated by the McCormick convex envelopes, ultimately resulting in a speed-up due to their low computational cost and provision of valuable bound information that is not automatically inferred through the McCormick convex envelopes.

Variable bounds via SDP relaxation of (2.14). Next, we compute bounds M_λ, M_ν, M_α, and M_Z such that

$$\begin{aligned}
&\lambda_{i,j} \leq M_\lambda, \quad i,j \in I_N^\star : i \neq j,\\
&|Z_{i,j}| \leq M_Z, \quad i,j \in [1:N+2],\\
&|P_{i,j}| \leq M_P, \quad i,j \in [1:N+2],\\
&|\alpha_{i,j}| \leq M_\alpha, \quad i \in [1:N],\ j \in [0:i-1],\\
&\nu \leq M_\nu,
\end{aligned} \tag{2.17}$$

are satisfied by global minimizers of (2.14).

Let $w = \mathrm{vec}(\alpha, \nu, \lambda)$ denote the column vector stacking the elements of α, ν, and λ. Let $W = ww^\top$. Then we can construct a lifted nonconvex semidefinite representation of the constraint set of (2.14), which includes the nonconvex rank-1 constraint $W = ww^\top$ [87]. The specific form is quite tedious, so we present it in §2.A.3 of the appendix. We then relax the

rank-1 constraint $W = ww^\top$ to an implied convex constraint

$$W \succeq ww^\top \Leftrightarrow \begin{bmatrix} W & w \\ w^\top & 1 \end{bmatrix} \succeq 0, \tag{2.18}$$

where we have used the Schur complement. Since any feasible (and optimal) solution of (2.14) must lie in this larger relaxed convex set, we compute bounds by optimizing over this set as follows.

The feasible point provided by Stage 1 of the BnB-PEP Algorithm establishes an upper bound $\nu \leq M_\nu = \nu^{\text{init}}$, since ν is the scaled objective function. Next, solve

$$\left(\begin{array}{ll} \text{maximize} & c_\lambda M_\lambda + c_Z M_Z + c_\alpha M_\alpha \\ \text{subject to} & \text{semidefinite relaxation of (2.14)}, \\ & \text{constraint (2.18)}, \\ & \lambda_{i,j} \leq M_\lambda, \quad i,j \in I_N^\star : i \neq j, \\ & |Z_{i,j}| \leq M_Z, \quad i,j \in [1:N+2], \\ & |\alpha_{i,j}| \leq M_\alpha, \quad i \in [1:N],\ j \in [0:i-1], \\ & \nu \leq \nu^{\text{init}}, \end{array}\right) \tag{2.19}$$

where λ, ν, Z, α, W, $M_\lambda \leq \|\lambda\|_1$, $M_Z \leq \mathbf{tr}\, Z$, and $M_\alpha \leq \|\alpha\|_1$ are the decision variables, with

$$(c_\lambda, c_Z, c_\alpha) \in \{(1,0,0), (0,1,0), (0,0,1)\}$$

to obtain M_λ, M_Z, and M_α, respectively.[4] (Since we restrict our search to points satisfying $\nu \leq \nu^{\text{init}}$, our bounds may exclude some suboptimal feasible solutions. However, all optimal

[4]As a note of caution, solving (2.19) with $(c_\lambda, c_Z, c_\alpha) = (1,1,1)$ does not provide a valid bound for all M_λ, M_Z, and M_α; maximizing $M_\lambda + M_Z + M_\alpha$ may reduce one bound below a valid threshold to increase another bound.

solutions will satisfy the bound.) Finally, we set $M_P = \sqrt{M_Z}$ based on

$$P_{i,j}^2 \leq \sum_{k=1}^{i} P_{i,k}^2 = Z_{ii} \leq M_Z \tag{2.20}$$

for all $i, j \in [1 : N + 2]$.

To clarify, this approach is a relaxation in the sense that it is guaranteed to produce variable bounds that will include all globally optimal solutions. (However, there is no guarantee on the tightness of the bounds, so the bounds could be very loose and not useful.)

Besides computing valid bounds on the variables, we also investigated the quality of the solutions of the SDP relaxations, for this setup and all other examples in this chapter. Unfortunately, we found that the solutions of the SDP relaxations to be of very poor quality in every case: the optimal value of the SDP relaxation was far from the optimal value of the BnB-PEP-QCQP. Additionally, we observed that the SDP relaxations failed to generate feasible solutions for the underlying BnB-PEP-QCQPs, even when we considered a rank-1 projection of the solution matrix. In other words, it was not possible to reconstruct valid first-order methods from the solutions of the SDP relaxations.

Heuristic bounds. However, the SDP relaxation to compute the variable bounds is quite cumbersome. Therefore, we present a simpler alternative, a heuristic that estimates the variable bounds based on the Stage 2 solution.

The premise of the heuristic is as follows. First, we make the informal assumption that the Stage 2 solution is near-optimal, which, again, happened very often in our experiments. In §2.4.3, we discuss that optimal inner-dual variables $\lambda = \{\lambda_{i,j}\}_{i,j \in I_N^\star : i \neq j}$ and Z are sometimes not unique and that sparse λ and low-rank Z are more valuable. Following the literature on sparse signal processing [17, §2], we promote sparsity of λ by reducing its ℓ_1-norm

$\sum_{i,j\in I_N^\star:i\neq j}\lambda_{i,j}$ and low rank of Z by reducing its nuclear norm $\mathbf{tr}\, Z$. To do so, we need our variable bounds to include the global solutions with the minimum ℓ_1-norm of λ and minimum nuclear norm of Z.

Based on the constraint set of (2.13), consider the following convex SDP

$$\left(\begin{array}{ll} \text{maximize} & c_\lambda \sum_{i,j\in I_N^\star:i\neq j}\lambda_{i,j} + c_Z\, \mathbf{tr}\, Z \\ \text{subject to} & \sum_{i,j\in I_N^\star:i\neq j}\lambda_{i,j}a_{i,j} = 0, \\ & \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha^{\text{lopt}}) + 2\mu A_{\star,N}(\alpha^{\text{lopt}}) + \\ & \quad \sum_{i,j\in I_N^\star:i\neq j}\lambda_{i,j}\left(A_{i,j}(\alpha^{\text{lopt}}) + \frac{1}{2(L-\mu)}C_{i,j}\right) = Z, \\ & Z \succeq 0, \\ & \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \\ & \nu \geq 0, \\ & \nu R^2 \leq \nu^{\text{lopt}} R^2, \end{array}\right) \tag{2.21}$$

where ν, λ, and Z are the decision variables, $(c_\lambda, c_Z) \in \{(1,0),(0,1)\}$, and α^{lopt} are ν^{lopt} are set to be values from the Stage 2 solution. Let $\widetilde{M}$ be a user-defined parameter greater than 1. With $(c_\lambda, c_Z) = (1,0)$, we *maximize* the ℓ_1 norm of λ and get λ^{hrstc}. Set

$$M_\lambda = \widetilde{M} \max_{i,j\in I_N^\star:i\neq j}\{\lambda_{i,j}^{\text{hrstc}}\}.$$

With $(c_\lambda, c_Z) = (0,1)$, we *maximize* the nuclear norm of Z and get Z^{hrstc}. Set

$$M_Z = \widetilde{M} \max_{i\in[1:N+2]}\{Z_{i,i}^{\text{hrstc}}\}$$

based on the reasoning that (2.15) implies that every entry of Z is bounded by the maximum

of the diagonal entries. Set M_P from M_Z using (2.20). Set

$$M_\alpha = 5\widetilde{M} \max_{0 \le j < i \le N} \{\alpha_{i,j}^{\text{lopt}}\}.$$

A note of caution is that when the Stage 2 solution is far from optimal, it is unclear whether this heuristic is even likely to produce a valid bound. When the Stage 2 solution is, in fact, near-optimal, the heuristic should help the BnB-PEP Algorithm to find the globally optimal solution quickly, and this is what we observe in our experiments.

Another note of caution is that the heuristic fails silently when it fails; there is no reliable mechanism to detect whether the heuristic bounds include or exclude global solutions. After solving (2.14) using the bounds, we verify if the solution lies within the interior of those bounds. Furthermore, empirically we always found that the solutions computed using the heuristic-based bound were well within the interior of the imposed bounds, though this is not a guarantee that the associated solution is globally optimal, as there is a possibility that a strictly better global solution lies far outside of the boundary since the BnB-PEP-QCQP is nonconvex. Finally, in all our experiments, we additionally verified that the heuristic-based bound produced the same optimal solutions as the SDP-based bounds.

Remark. We clarify that the heuristic bound offers no guarantee of correctness and that the SDP relaxation, which is guaranteed to be correct, is the superior choice. However, SDP relaxations can be cumbersome to formulate and implement. Therefore, one may first try out the heuristic bound in a prototyping phase and then decide to implement the SDP relaxation if the preliminary results are sufficiently interesting.

2.4.2.2 Tighter lower bounds via lazy callback

When an incumbent or warm-starting solution is already near-optimal, i.e., when the upper bound is already good, the work in certifying global optimality mostly lies in improving the lower bound. Indeed, in our experiments, Stage 2 of the BnB-PEP Algorithm consistently found near-optimal solutions, and Stage 3 spent most of its time improving the lower bound to certify or polish the solution of Stage 2. If we can compute a good lower bound and provide it to the spatial branch-and-bound algorithm, Stage 3 can terminate very quickly as the number of subregions to be explored is substantially reduced. To that goal, we compute a tighter lower bound of (2.14) via the *lazy constraint callback method*. Unlike normal constraints, lazy constraints are not generated upfront but are rather generated and added one by one when needed.

Consider a variant of (2.14), where we model $Z = PP^\top \Leftrightarrow Z \succeq 0$ equivalently as

$$\mathbf{tr}\left(Zyy^\top\right) \geq 0, \qquad \forall\, y \in \mathbb{R}^{N+2}.$$

Since this formulation uses an infinite set of linear constraints, we relax it with a finite set of linear constraints

$$\mathbf{tr}\left(Zyy^\top\right) \geq 0, \qquad \forall\, y \in Y, \tag{2.22}$$

where Y is initialized to be a randomly generated set of $2(N+2)^2$ unit vectors in $\mathbb{R}^{N+2}$ following the prescription of [88, §5.1]. In (2.14), we relax $Z = PP^\top$ into the constraint (2.22) and obtain a simpler QCQP. Then, update Y lazily by repeating the following steps (i)–(iii) a finite number of times (1×10^6 times in our implementation):

(i) Solve the relaxation of (2.14), where (2.22) is used instead of $Z = PP^\top$, to obtain Z and a lower bound.

(ii) Find the minimum eigenvalue $\text{eig}_{\min}(Z)$ and corresponding normalized eigenvector u of Z. If $\text{eig}_{\min}(Z) \geq 0$, terminate.

(iii) If $\text{eig}_{\min}(Z) < 0$, then add u to Y, i.e., add the constraint $\mathbf{tr}(Zuu^\top) \geq 0$. (Note, $\mathbf{tr}(Zuu^\top) < 0$. So the added constraint makes the current Z infeasible for the updated relaxation (2.22).)

In step (ii), if $\text{eig}_{\min}(Z) \geq 0$, then the solution Z of the relaxation (2.22) is in fact optimal for the original unrelaxed problem, so we terminate. We use the lazy constraint callback interface of `JuMP` to implement this scheme. After adding one additional linear constraint in step (iii), updating the solution in step (i) is efficient since `Gurobi` and all modern solvers based on the simplex algorithm can quickly update a solution when one linear constraint is added [89, pp. 205–207].

2.4.2.3 Improved upper bounds via SDP solves

As a heuristic to obtain improve upper bounds, we utilize the fact that the optimization of (2.14) reduces to an SDP when the stepsize α is fixed. This is a structure that the branch-and-bound solver cannot infer by itself.

When the branching process reaches a new node, we access (via a callback feature of `JuMP`) the solution $(\alpha^{\text{rlx}}, \nu^{\text{rlx}}, \lambda^{\text{rlx}}, Z^{\text{rlx}}, P^{\text{rlx}})$ of the relaxation and quantify its infeasibility with

$$\begin{aligned}
&\text{merit}(\alpha^{\text{rlx}}, \nu^{\text{rlx}}, \lambda^{\text{rlx}}, Z^{\text{rlx}}, P^{\text{rlx}}) \\
=&\| \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j}^{\text{rlx}} a_{i,j} \|_\infty + \left\| \nu^{\text{rlx}} B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha^{\text{rlx}}) + 2\mu A_{\star,N}(\alpha^{\text{rlx}}) \right\|_\infty \\
&+ |\min\{\text{eig}_{\min}(Z^{\text{rlx}}), 0\}|,
\end{aligned}$$

N	M_λ	M_α	M_Z	M_P	M_ν	Runtime (s)
1	1.00	2.00	1.00	1.00	0.2244	0.068
2	1.00	4.5175	1.00	1.00	0.0893	0.181
3	1.00	3.672	1.00	1.00	0.0449	0.736
4	1.00	3.5166	1.00	1.00	0.0257	3.173
5	1.00	3.7919	1.00	1.00	0.0159	11.380

Table 2.5: Valid bounds on the decision variables in (2.14) obtained via the SDP relaxation of (2.19). The runtime describes to the total time spent compute all the bounds of the row.

where $\text{eig}_{\min}(Z^{\text{rlx}})$ is the minimum eigenvalue of Z^{rlx}. If

$$\text{merit}(\alpha^{\text{rlx}}, \nu^{\text{rlx}}, \lambda^{\text{rlx}}, Z^{\text{rlx}}, P^{\text{rlx}}) \leq \epsilon,$$

then we fix the stepsize in (2.13) to α^{rlx} and solve the convex SDP. (We take $\epsilon = 0.01$ in our implementation.) We submit the solution to the SDP as a heuristic solution (via a callback feature of `JuMP`). If the heuristic solution improves the best upper bound $\overline{p}^\star$, then it is accepted by the solver, else it is rejected.

2.4.2.4 Numerical evaluation of the customizations

In our experiments, we found that that the customization of §2.4.2.1 provided the largest speedups, §2.4.2.2 substantial speedups, and §2.4.2.3 no speedups. We further describe our observations here.

Variable bounds of §2.4.2.1. Tables 2.5 and 2.6 show the bounds obtained through the SDP relaxation. As an aside, we found that these valid bounds substantially improve not only the branch-and-bound algorithm of Stage 3, but also the local solve of Stage 2.

N	M_λ	M_α	M_Z	M_P	M_ν	Runtime (s)
1	0.8789	7.6105	0.4233	0.6506	0.1473	0.082
2	0.9504	8.2597	0.1934	0.4397	0.0409	0.093
3	0.9767	9.4761	0.1009	0.3177	0.0145	0.105
4	0.9853	9.8591	0.0599	0.2448	0.005766	0.114
5	0.9886	10.3633	0.0383	0.1958	0.002459	0.121

Table 2.6: Heuristic bounds on the decision variables in (2.14) with $\widetilde{M} = 1.01$. The runtime describes the total time spent compute all the bounds of the row. Compared to the results of Table 2.5 the bounds tend to be tighter, the runtime is faster, and the implementation is much simpler. However, there is no theoretical guarantee that the bounds are valid.

N	$\underline{p}^\star$	$\overline{p}^\star$	$\overline{p}^\star - \underline{p}^\star$	Runtime (s)
1	0.1432	0.1473	0.0041	0.135
2	0.0374	0.0409	0.0035	0.232
3	0.0121	0.0145	0.0024	2.550
4	0.00178	0.005766	0.003986	72.7
5	0.000517	0.002459	0.001941	336.341

Table 2.7: Lower bound $\underline{p}^\star$ of (2.14) computed from the lazy constraint callback method. The upper bound $\overline{p}^\star$ is the objective value from Stage 2 of the BnB-PEP Algorithm.

Tighter lower bound of §2.4.2.2. Table 2.7 shows the lower-bounds for (2.14) computed from the lazy constraint callback method. The customization produces a high quality lower bound, which, combined with the near-optimal solution of Stage 2 of the BnB-PEP Algorithm, enables the branch-and-bound algorithm to terminate quickly.

Improved upper bound of §2.4.2.3. In our experiments, the submitted upper bounds were all rejected by the solver and therefore provided no speedup. This is not surprising, as it is likely due to the warm-starting solution from Stage 2 being near-optimal. To verify this

hypothesis, we ran Stage 3 without Stage 2. In this case, the submitted heuristic solution was often accepted by the solver, but the overall performance was slow as Stage 3 started from a poor warm-starting solution. We recommend that users of the BnB-PEP Algorithm always perform Stage 2 before Stage 3. However, when the warm-starting solution is not near-optimal, we can expect this customization to provide a speedup.

2.4.3 Structured inner-dual variables

The family solutions to (2.14) with $N = 1, 2, \ldots$ exhibits an exploitable structure: the optimal $\lambda^\star$ is sparse and the optimal $Z^\star$ is low-rank. A computational benefit of this structure is that it reduces the problem size of the BnB-PEP-QCQP. A theoretical benefit is that the structured inner-dual variable corresponds to simpler and therefore more analytically tractable proofs, which we seek in §2.6.3.

In this section, we describe a heuristic strategy for identifying such structure. The general idea is to solve the problem exactly for smaller values of N, say $N = 1, \ldots, 5$, and infer the pattern. This heuristic is based on the expectation that the observed patterns will continue to hold for $N = 6, 7, \ldots$.

Sparsity pattern of $\boldsymbol{\lambda}$. Denote the support of $\lambda^\star$ as

$$\operatorname{supp}(\lambda^\star) = \{(i, j) \mid i, j \in I_N^\star, i \neq j,\ \lambda_{i,j}^\star > 0\}.$$

(Note that $\lambda_{i,j}^\star \geq 0$ for all i, j.) If we know $\operatorname{supp}(\lambda^\star)$ in advance, then we can simplify (2.14) by replacing both instances of

$$\sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j}(\cdots)$$

with

$$\sum_{(i,j)\in \mathrm{supp}(\lambda^\star)} \lambda_{i,j}(\cdots)$$

and obtain a smaller QCQP.

First solve (2.14) for $N = 1, \ldots, 5$ using the BnB-PEP Algorithm. At this point, solutions may already reveal their pattern in $\mathrm{supp}(\lambda^\star)$. However, optimal inner-dual variables for a given FSFOM are not always unique (see [90] or Table 2.8), and, if so, the solution returned by the spatial branch-and-bound solver will likely not be a sparse one. Therefore, following the literature on sparse signal processing [17, §2], we promote sparsity of λ by reducing its ℓ_1-norm[5] as follows

$$\left(\begin{array}{ll} \text{minimize} & \|\lambda\|_1 = \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \\ \text{subject to} & \nu R^2 \leq p^\star, \\ & \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} a_{i,j} = 0, \\ & \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha^\star) + 2\mu A_{\star,N}(\alpha^\star) + \\ & \quad \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left(A_{i,j}(\alpha^\star) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\ & Z \succeq 0, \\ & \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \\ & \nu \geq 0, \end{array}\right) \tag{2.23}$$

where ν, λ, and $Z \in \mathbb{S}_+^{N+2}$ are the decision variables. The constraint set of (2.23) is almost identical to (2.13), except we impose the constraint $\nu R^2 \leq p^\star$ and fix $\alpha^\star$ to the optimal stepsize computed with the BnB-PEP Algorithm. This way, our search is confined to the set of optimal solutions to (2.14) and, with the FSFOM fixed, the problem is efficiently solved as an SDP. Denote the solution to (2.23) by $\{\nu^\star, \lambda^{\star,\mathrm{sparse}}, Z^{\star,\mathrm{sparse}}\}$. Hopefully, the optimal $\lambda^{\star,\mathrm{sparse}}$ for $N = 1, \ldots, 5$ are sparse and their structure reveals a pattern.

[5] One could consider further advanced approaches for promoting sparsity such as [91].

N	Optimal λ with minimum ℓ_1 norm	Optimal λ with maximum ℓ_1 norm
1	2.642	3.594
2	2.434	3.114
3	2.369	2.925
4	2.339	2.823
5	2.320	2.757

Table 2.8: Solutions to (2.23) with minimum and maximum ℓ_1-norm on optimal λ for the setup of §2.3 and §2.4. The gap demonstrates that the optimal inner-dual variable is not unique and therefore that the ℓ_1-norm minimization is necessary for obtaining a sparse solution.

Low rank of Z. When $r = \text{rank}(Z^\star)$ with $r < n$, then we can use the factorization $Z = PP^\top$, where $P \in \mathbb{R}^{n\times n}$ has $n - r$ columns constrained to be zero. Such constraints significantly reduce the effective size of the QCQP.

Lemma 2.4 ([92, Theorem 10.9])**.** *A matrix $Z \in \mathbb{S}^n$ is positive semidefinite with rank $r \leq n$ if and only if it has a Cholesky factorization $Z = PP^\top$, where $P \in \mathbb{R}^{n\times n}$ is lower-triangular, has r positive diagonal entries, and $n - r$ columns containing all zero.*

We solve (2.23) for $N = 1, \ldots, 5$ and infer the rank. (In principle, one could perform a separate nuclear norm minimization or further advanced approaches such as [93] to reduce the rank of Z, but this was not necessary in our experiments.) In our current setup, $Z^{\star,\text{sparse}}$ has rank 1, as Table 2.9 indicates. Other optimized method throughout the literature such as OGM, ITEM, OGM-G [4, 8, 9, 34] also have corresponding low-rank $Z^\star$.

For $N = 6, 7, \ldots$, we obtain $Z^\star$ and $P^\star$ from Stage 2 of the BnB-PEP Algorithm. If we expect a certain value of $r = \text{rank}(Z^\star)$, keep r columns of $P^\star$ with the largest magnitude and constrain the remaining $n - r$ columns to be 0 in the subsequent Stage 3.

Structured inner-dual variables represent simpler proofs. A feasible point of the dualized problem (2.13) can be interpreted as a convergence proof combining inequalities

$$f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2(L-\mu)} \|g_i - g_j\|^2 \tag{2.24}$$

for $i, j \in I_N^\star$, and the value of $\lambda_{i,j}^\star$ corresponds to the value used in forming a weighted combination of the inequalities [48, §3.3]. Therefore, $\lambda_{i,j}^\star = 0$ for some (i, j) is equivalent to not using the corresponding inequality in the convergence proof. A sparse λ corresponds to a proof using fewer inequalities, which tend to be simpler proofs.

On the other hand, $\operatorname{rank}(Z^\star)$ corresponds to the excess quadratic terms arising within a proof. For convergence proof of FSFOMs of the form,

$$A \leq B - \|c_1\|^2 - \ldots \|c_r\|^2 \leq B,$$

$r = \operatorname{rank}(Z^\star)$ corresponds to the number of quadratic terms $\{\|c_i\|^2\}_{i=1,\ldots,r}$, roughly speaking. Since $\operatorname{rank}(Z^\star)$ corresponds to the number of excess terms to deal with in a proof, an optimal solution $Z^\star$ with small rank tends to correspond to simpler proofs.

Numerical results. Table 2.9 presents $\operatorname{supp}(\lambda^{\star,\text{sparse}})$, $\operatorname{rank}(Z^{\star,\text{sparse}})$, and the non-zero columns of $P^{\star,\text{sparse}}$ from solving the convex SDP (2.23). We use $\mu = 0.1$, $L = 1$, and $R = 1$. From the results, we infer the pattern

$$\operatorname{supp}(\lambda^{\star,\text{sparse}}) = \{(\star, i)\}_{i\in[0:N]} \cup \{(i, i+1)\}_{i\in[0:N-1]} \cup \{N, i\}_{i\in\{\star\}\cup[0:N-1]},$$

which has only $3N + 2$ components compared to the $(N + 2)(N + 1)$ of the full index set. Furthermore, $\operatorname{rank}(Z^{\star,\text{sparse}}) = 1$. For $N = 1, \ldots, 5$, we verified that there are globally optimal

N	$\text{supp}(\lambda^\star)$	Rank of $Z^{\star,\text{sparse}}$	Index of nonzero column of $P^{\star,\text{sparse}}$	Runtime (s) for solving (2.23)
1	$\{(\star,0),(\star,1),$ $(0,1),$ $(1,\star),(1,0)\}$	1	Column # 1	0.0021
2	$\{(\star,0),(\star,1),(\star,2),$ $(0,1),(1,2),$ $(2,\star),(2,0),(2,1)\}$	1	Column # 1	0.0056
3	$\{(\star,0),(\star,1),(\star,2),(\star,3),$ $(0,1),(1,2),(2,3),$ $(3,\star),(3,0),(3,1),(3,2)\}$	1	Column # 1	0.0071
4	$\{(\star,i)\}_{i\in[0:4]}\cup$ $\{(i,i+1)\}_{i\in[0:3]}\cup$ $\{4,i\}_{i\in\{\star\}\cup[0:3]}$	1	Column # 1	0.0097
5	$\{(\star,i)\}_{i\in[0:5]}\cup$ $\{(i,i+1)\}_{i\in[0:4]}\cup$ $\{5,i\}_{i\in\{\star\}\cup[0:4]}$	1	Column # 1	0.0140

Table 2.9: Structure of the inner-dual variables obtained from the convex SDP (2.23). The last column shows the runtime to solve (2.23). (Table 2.4 shows the runtime to solve (2.14), a prerequisite for solving (2.23).)

solutions satisfying these patterns. For $N = 6, 7, \ldots, 25$, we verified that there are locally optimal solutions satisfying these patterns.

Discussion. Prior work on optimized FSFOMs such as OGM, ITEM, and OGM-G [4, 8, 9, 34] discard certain inequalities in their formulations. The choice of which inequality to discard, which is equivalent to identifying $\text{supp}(\lambda^\star)$, was likely carried out through ad-hoc trial and error. As no reasoning or intuition was provided behind the choice and as the set of discarded inequalities are different from one work to another, the process is opaque. Our approach provides a systematic process for making this choice.

To clarify, we solve the exact, unrelaxed (2.14) with BnB-PEP Algorithm for $N = 1, \ldots, 5$. The methodology for $N = 6, 7, \ldots$ is a heuristic in the sense that our solution is exactly only

under the condition that the observed sparsity pattern continues. If the pattern changes, the QCQP becomes a relaxation, and the produced FSFOM becomes suboptimal. However, one can be reasonably confident in the sparsity pattern as it is based on the exact solutions for $N = 1, \ldots, 5$.

2.5 Generalized BnB-PEP methodology

We now discuss the generalization of the BnB-PEP methodology for general $\mathcal{E}$, $\mathcal{F}$, and $\mathcal{C}$.

Generalized BnB-PEP-QCQP. The BnB-PEP-QCQP formulation for general $\mathcal{E}$, $\mathcal{F}$, and $\mathcal{C}$ follows steps analogous to those of §2.3.1.

(i) **Infinite-dimensional inner optimization problem.** Construct an infinite-dimensional representation of ($\mathcal{O}^{\text{inner}}$) analogous to (2.6) of §2.3.1. When $x_\star$ exists, set $x_\star = 0$ and $f(x_\star) = 0$ without loss of generality.

(ii) **Interpolation argument.** Using a reparametrization (if necessary) and an interpolation argument, formulate the infinite-dimensional inner problem of (i) as a finite-dimensional problem analogous to (2.7) of §2.3.1.

(iii) **Grammian formulation.** By introducing Grammian matrices and using a large-scale assumption, formulate the finite-dimensional inner maximization problem of (ii) as an SDP, analogous to the problem (2.12) in §2.3.1. When the FSOM is fixed, the SDP is a convex optimization problem.

(iv) **Dualization.** Form the dual the SDP of (iii) analogous to (2.13) of §2.3.1. Assume strong duality.

(v) **Formulating ($\mathcal{O}^{\text{outer}}$) as a QCQP.** Using Lemma 2.3, replace the SDP constraint $Z \succeq 0$ of the dual SDP with $Z = PP^\top$, where P is lower triangular with nonnegative diagonals. This formulates ($\mathcal{O}^{\text{outer}}$) as a QCQP analogous to (2.14) of §2.3.1. When f

is nonconvex, certain cubic or trilinear terms may arise, whereas for a convex f the nonlinear terms are bilinear or quadratic. If so, for such a nonconvex f, formulate such terms as quadratic or bilinear constraints by introducing dummy variables, a process illustrated in §2.6.2 and §2.6.3. We call the resultant QCQP the BnB-PEP-QCQP. The variables of the dual SDP of (iv) are present in the BnB-PEP-QCQP, and we refer to them as the inner-dual-variables.

Generalized BnB-PEP Algorithm. We solve the BnB-PEP-QCQP to certifiable global optimality with the following generalized BnB-PEP Algorithm, a generalization of Algorithm 1.

- **Stage 1: Compute a feasible solution.** Fix the stepsizes in the dual SDP of Step (iv) of the formulation of BnB-PEP-QCQP to a reasonable h^{init} and solve the resultant convex minimization problem to obtain a feasible the BnB-PEP-QCQP. Table 2.10 lists reasonable stepsizes.
- **Stage 2: Compute a locally optimal solution by warm-starting at Stage 1 solution.** Warm-start the BnB-PEP-QCQP with the feasible solution found in Stage 1 and solve the problem to local optimality using a nonlinear interior-point method.
- **Stage 3: Compute a globally optimal solution by warm-starting at Stage 2 solution.** Warm-start the BnB-PEP-QCQP with the locally optimal solution found in Stage 2 and solve the problem to global optimality using a customized spatial branch-and-bound algorithm described in the following.

Efficient implementation of the generalized BnB-PEP Algorithm. We customize the spatial branch-and-bound algorithm to exploit specific problem structure of the generalized BnB-PEP-QCQP. The techniques are analogous to those described in §2.4. We find bounds on optimal solutions through implied linear constraints, SDP relaxation, and a heuristic. We

Function class	Fixed stepsize h^{init} for Stage 1 of the BnB-PEP Algorithm
$\mathcal{F}_{0,L}$	$h^{\text{init}}_{i,j} = \begin{cases} 1/L, & \text{if } j = i-1, \\ 0, & \text{else,} \end{cases} \quad 0 \le j < i \le N.$
$\mathcal{F}_{\mu,L}$	Same as $\mathcal{F}_{0,L}$.
$\mathcal{F}_{-L,L}$	Same as $\mathcal{F}_{0,L}$.
$\mathcal{W}_{\rho,L}$	$h^{\text{init}}_{i,j} = \begin{cases} \frac{R\rho}{L}\frac{1}{\sqrt{N+1}}, & \text{if } j = i-1, \\ 0, & \text{else,} \end{cases} \quad 0 \le j < i \le N.$

Table 2.10: Fixed stepsize vector h^{init} to use in step 1 of the BnB-PEP Algorithm. For $\mathcal{F}_{0,\infty}$, and $\mathcal{W}_{\rho,L}$, $R>0$ is the upper bound associated with the initial condition.

find tighter lower bounds via lazy callback by replacing $Z = PP^\top$ with

$$\mathbf{tr}\left(Zyy^\top\right) \ge 0, \qquad \forall\, y \in Y$$

and lazily updating Y. We improve upper bounds via SDP solves by constructing a merit function to measure the infeasibility at the nodes of the branch-and-bound tree and solving the convex SDP with the stepsizes fixed when the merit function value falls below some tolerance. We exploit the structure of the inner-dual variables by observing the sparsity and low-rank pattern for small N (e.g., $N \le 5$) and extrapolating the patterns to larger N.

2.6 Applications

In this section, we demonstrate the strength of the BnB-PEP methodology by applying it to three setups for which the prior methodologies do not apply. Numerical experiments of this section were performed in the computational setup described in §2.1.3. We empirically observed that, among the two approaches of §2.4.2.1 for computing variable bounds, the heuristic-based bounds were tighter and lead to runtimes faster by factor of 2–5 compared to

using the SDP-based bounds. We report the faster runtimes in our tables. In all instances, the two approaches produced the same optimal solutions.

2.6.1 Optimal gradient method without momentum

In optimization folklore, momentum is considered essential for accelerating first-order gradient methods. Indeed, prior FSFOMs minimizing smooth convex functions such as Nesterov's method [30], OGM [4, 8], ITEM [9], and many others [94, 95] all achieve accelerated rates with momentum. However, a little known fact is that simple gradient descent, without momentum, can achieve an accelerated rate for minimizing strongly convex *quadratics* [96, 97]. Whether a similar acceleration without momentum is possible for convex non-quadratic functions is not known.

In this section, we investigate whether the simple gradient descent method

$$x_i = x_{i-1} - \frac{1}{L} h_{i-1} \nabla f(x_{i-1}) \tag{$\mathcal{G}_N$}$$

with $i \in [1:N]$ can achieve an accelerated rate for minimizing L-smooth convex functions when the stepsize $\{h_i\}_{i\in[0:N-1]}$ is chosen optimally. We denote the class of FSFOM of this form as $\mathcal{G}_N \subset \mathcal{M}_N$.

As we discuss in §2.6.1.1, it is relatively straightforward to show that the unaccelerated $\mathcal{O}(1/k)$ rate cannot be surpassed if $\{h_i\}_{i\in[0:N-1]}$ stays within the "standard" range $(0, 2)$. However, Young's method [96] uses long steps satisfying $1 < h_i < L/\mu$ for some i to achieve an accelerated rate for L-smooth and μ-strongly convex quadratics. The question is whether a similar use of long steps can provide an acceleration in the smooth convex setup.

Formally, we choose the function class

$$\mathcal{F} = \{f \mid f \in \mathcal{F}_{0,L},\ f \text{ has a minimizer } x_\star\},$$

performance measure $\mathcal{E} = f(x_N) - f(x_\star)$, and initial condition $\mathcal{C} = \|x_0 - x_\star\|^2 - R^2 \leq 0$ with $R > 0$. We solve the following instance of ($\mathcal{O}^{\text{outer}}$):

$$\mathcal{R}^\star\left(\mathcal{G}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) = \underset{M \in \mathcal{G}_N}{\text{minimize}} \quad \mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \ .$$

Derivation of BnB-PEP-QCQP. Following §2.5 Step (i), formulate the inner optimization problem ($\mathcal{O}^{\text{inner}}$) as

$$\begin{aligned}
&\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{lll}
\text{maximize} & f(x_N) - f(x_\star) & \\
\text{subject to} & f \in \mathcal{F}_{0,L}, & \\
& \nabla f(x_\star) = 0, & \\
& x_i = x_{i-1} - h_{i-1}\nabla f(x_{i-1}) & i \in [1:N], \\
& \|x_0 - x_\star\|^2 \leq R^2, & \\
& x_\star = 0,\ f(x_\star) = 0, &
\end{array} \right)
\end{aligned}$$

where f and $x_0, \ldots, x_N$ are the decision variables. Write $h = \{h_i\}_{i \in [0:N-1]}$. Following §2.5 Step (ii), use the interpolation argument to formulate the inner problem as

$$\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right)$$

$$
= \left(\begin{array}{l}
\text{maximize} \quad f_N - f_\star \\
\text{subject to} \\
f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2L}\|g_i - g_j\|^2, \quad i,j \in I_N^\star : i \neq j, \\
g_\star = 0, x_\star = 0, f_\star = 0, \\
x_i = x_0 - (1/L)\sum_{j=0}^{i-1} h_j g_j, \quad i \in [1:N], \\
\|x_0 - x_\star\|^2 \leq R^2,
\end{array} \right)
$$

where $\{x_i, g_i, f_i\}_{i \in I_N^\star} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$ are the decision variables. Following §2.5 Step (iii), implement the Grammian transformation. Define the Grammian matrices $H \in \mathbb{R}^{d \times (N+2)}$, $G \in \mathbb{S}_+^{N+2}$, and $F \in \mathbb{R}^{1 \times (N+1)}$ using the same equations in (2.8), $\{\mathbf{x}_i, \mathbf{g}_i, \mathbf{f}_i\}_{i \in I_N^\star}$ using the same encoding as (2.9), except for $\{\mathbf{x}_i\}_{i \in [1:N]}$, which we define as

$$
\mathbf{x}_i = \mathbf{x}_0 - (1/L)\sum_{j=0}^{i-1} h_j \mathbf{g}_j \in \mathbb{R}^{N+2}, \quad i \in [1:N].
$$

Note, $\mathbf{x}_i$ is linearly parameterized by h. The matrices $A_{i,j}$, $B_{i,j}$, $C_{i,j}$, and $a_{i,j}$ are the same as in (2.10) except that they are now parameterized by h. Under the large-scale assumption $d \geq N+2$, we equivalently formulate the inner problem as the SDP

$$
\begin{aligned}
&\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{maximize} \quad F a_{\star,N} \\
\text{subject to} \\
F a_{i,j} + \mathbf{tr}\, G A_{i,j}(h) + \frac{1}{2L}\mathbf{tr}\, G C_{i,j} \leq 0, \quad i,j \in I_N^\star : i \neq j, \quad \triangleright \text{ dual var. } \lambda_{i,j} \geq 0 \\
-G \preceq 0, \quad \triangleright \text{ dual var. } Z \succeq 0 \\
\mathbf{tr}\, G B_{0,\star} \leq R^2, \quad \triangleright \text{ dual var. } \nu \geq 0
\end{array} \right)
\end{aligned}
$$

where $F \in \mathbb{R}^{1\times(N+1)}$ and $G \in \mathbb{R}^{(N+2)\times(N+2)}$ are the decision variables. Following §2.5 Step (iv), construct the dual:

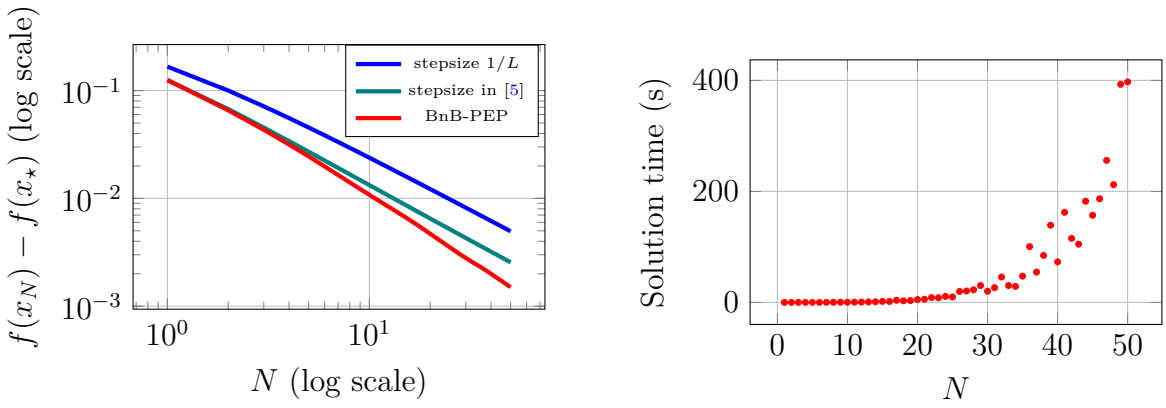
$$\begin{aligned}
&\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} a_{i,j} - a_{\star,N} = 0, \\
\nu B_{0,\star} + \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left(A_{i,j}(h) + \frac{1}{2L} C_{i,j} \right) = Z, \\
Z \succeq 0, \\
\nu \geq 0,\ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j,
\end{array} \right)
\end{aligned}$$

where ν, λ, and Z are the decision variables. Assume that strong duality holds. Finally, following §2.5 Step (v), use Lemma 2.3 to pose ($\mathcal{O}^{\text{outer}}$) as the following BnB-PEP-QCQP:

$$\begin{aligned}
&\mathcal{R}^\star\left(\mathcal{G}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} a_{i,j} - a_{\star,N} = 0, \\
\nu B_{0,\star} + \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left(A_{i,j}(h) + \frac{1}{2L} C_{i,j} \right) = Z, \\
P \text{ is lower triangular with nonnegative diagonals}, \\
PP^\top = Z, \\
\nu \geq 0,\ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j,
\end{array} \right)
\end{aligned} \tag{2.25}$$

where λ, ν, Z, P, and h are the decision variables.

Numerical results. Tables 2.11 and 2.12 present the results of solving (2.25) with $L = 1$, $R = 1$, $N = 1, \ldots, 5, 10, 25$ using the BnB-PEP Algorithm. We compare the optimal stepsize



(a) (b)

Figure 2.1: Numerical results for computing locally optimal stepsizes by solving (2.25) with the first two stages of the BnB-PEP Algorithm. Global optimality of the stepsizes are verified for $N = 1, 2, \ldots, 25$. (Left) Worst-case performance of $f(x_N) - f_\star$ vs. iteration count N. (Right) Runtimes of the BnB-PEP Algorithm (including Stages 1 and 2 but excluding Stage 3).

N	# variables	# constraints	Worst-case $f(x_N) - f(x_\star)$			Runtime of the BnB-PEP Algorithm
			Optimal	For stepsize in [5, §4.1.1]	For stepsize $h_i = 1$	
1	20	33	0.125	0.125	0.1667	0.03 s
2	34	56	0.065946	0.067355	0.1	0.252 s
3	54	85	0.042893	0.045364	0.0714	0.375 s
4	77	120	0.03117	0.033976	0.0555	17.602 s
5	104	161	0.024071	0.0270701	0.0454	86.904 s
10	365	456	0.010622	0.0132692	0.0238	1 d 18 h
25	1835	2241	0.0034757	0.0051754	0.0098	2 d 20 h

Table 2.11: Comparison between the performances of the optimal method obtained by solving (2.25) with the BnB-PEP Algorithm, the method with constant normalized stepsize $h_i = 1$, and the method with constant normalized stepsize h_i prescribed in [5, §4.1.1]. The BnB-PEP Algorithm was executed on a standard laptop for $N = 1, 2, \ldots, 10$, and on `MIT Supercloud` for $N = 25$.

N	$h^\star$
1	$\begin{bmatrix} 1.5 \end{bmatrix}$
2	$\begin{bmatrix} 1.414214 \\ 1.876768 \end{bmatrix}$
3	$\begin{bmatrix} 1.414215 \\ 2.414207 \\ 1.500001 \end{bmatrix}$
4	$\begin{bmatrix} 1.414214 \\ 1.601232 \\ 3.005144 \\ 1.5 \end{bmatrix}$
5	$\begin{bmatrix} 1.414214 \\ 2.0 \\ 1.414214 \\ 3.557647 \\ 1.5 \end{bmatrix}$
10	See Supplementary Information or `Github` repository
25	See Supplementary Information or `Github` repository

Table 2.12: Globally optimal stepsizes obtained by solving (2.25) with the BnB-PEP Algorithm.

with the constant normalized stepsize $h_i = 1$ for all i, which is known to be optimal among $h_i \in (0, 1]$ [98, Corollary 2.8, Theorem 2.9], and constant normalized stepsize h satisfying

$$\frac{1}{2Nh + 1} = (1 - h)^{2N}, \tag{2.26}$$

which is conjectured by Taylor, Hendrickx, and Glineur to be the optimal constant normalized stepsize [5, §4.1.1]. The stepsizes presented in Table 2.12 are certified to be globally optimal. Interestingly, we observe that the locally optimal stepsizes obtained at Stage 2, denoted by h^{lopt}, were already near-optimal.

This observation motivates us to apply just the first two stages of the BnB-PEP Algorithm for N upto 50. In Figure 2.1, we again compare the performance guarantees of the locally optimal stepsizes h^{lopt} with that of the constant normalized stepsizes $h_i = 1$ and the constant normalized stepsizes h_i satisfying (2.26). We verified global optimality of these stepsizes h^{lopt} for $N = 1, \ldots, 25$. While h^{lopt} for $N = 26, \ldots, 50$ are not certifiably globally optimal (although we suspect that they are near-optimal), their computed performances are certifiably accurate.

Figure 2.1 shows that the computed stepsizes h^{lopt} outperforms both constant stepsizes. Figure 2.2 presents a linear fit of the rate in the log-log scale. The fit $0.156/N^{1.178}$ indicates that the asymptotic rate may be faster than $\mathcal{O}(1/k)$.

Figure 2.3 shows h^{lopt} for $N = 5, 10, 25, 50$. The optimal stepsizes h^{lopt} for $N = 1, 2, \ldots, 50$ (global optimality verified for $N = 1, 2, \ldots, 25$) are provided as Supplementary Information and as a data file at:

https://github.com/Shuvomoy/BnB-PEP-code/blob/main/Misc/stpszs.jl

However, finding an analytical form of the computed stepsizes seems difficult. Therefore, we leave inconclusive the question of whether acceleration without momentum is possible.

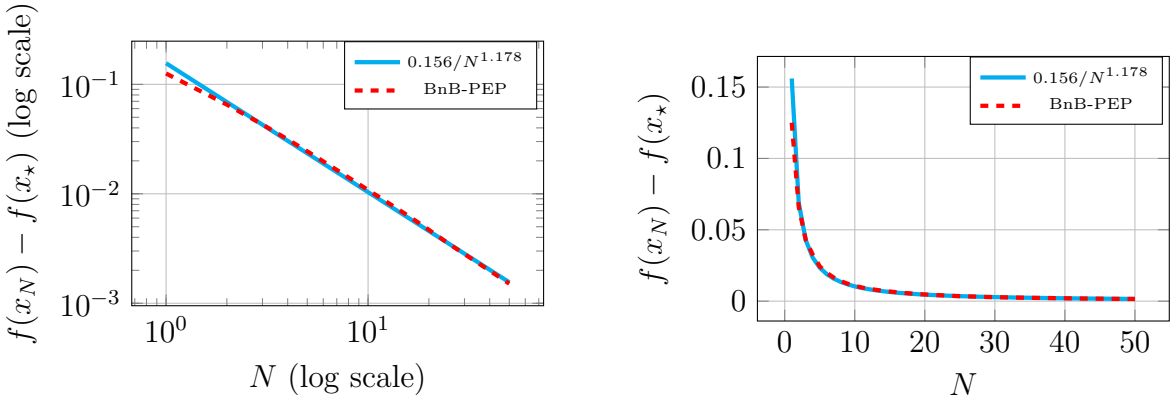


Figure 2.2: Fitting the worst-case performance of $f(x_N) - f_\star$ corresponding to the locally optimal stepsizes obtained by solving (2.25) with the first two stages of the BnB-PEP Algorithm yields $0.156/N^{1.178}$. The asymptotic rate may be faster than $\mathcal{O}(1/k)$.

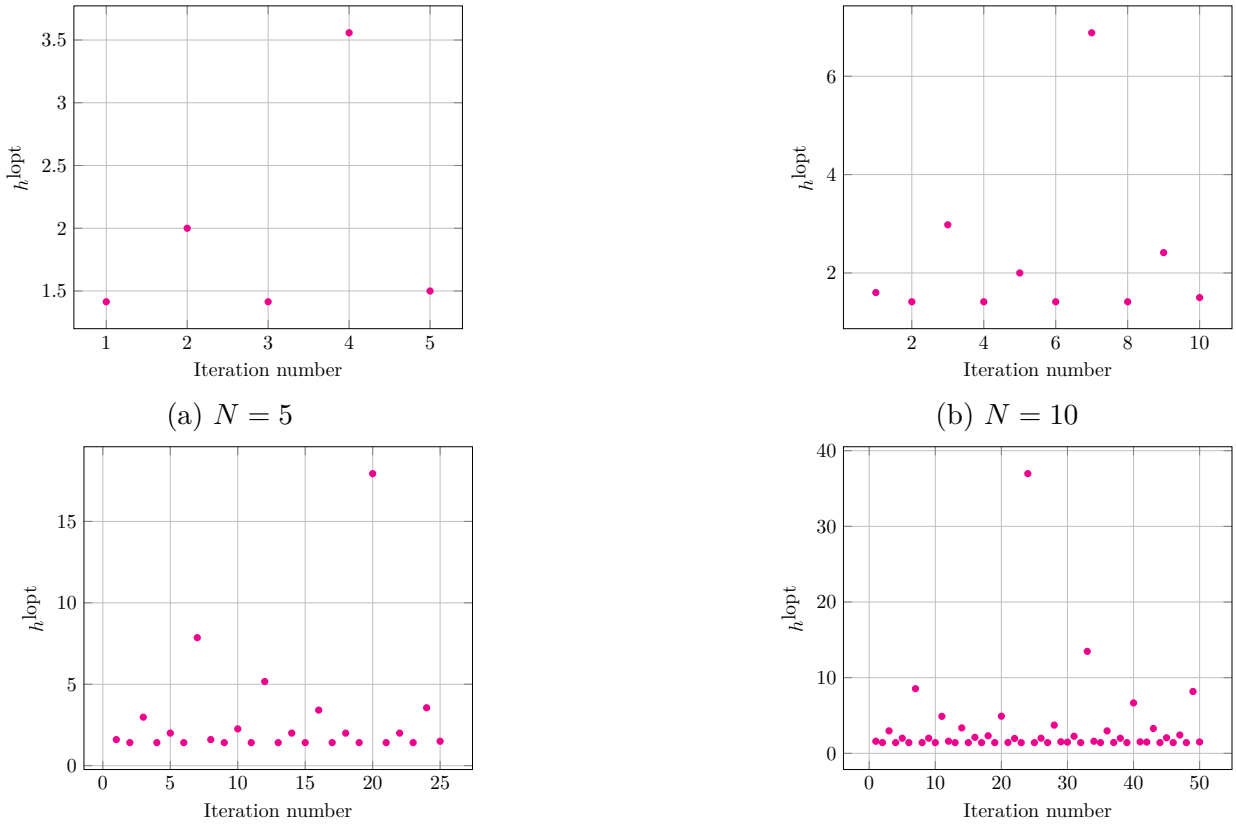
2.6.1.1 Acceleration is impossible without long steps

Consider the univariate functions

$$f_1(x) = \begin{cases} \frac{LR}{2Nh+1}|x| - \frac{LR^2}{2(2Nh+1)^2} & \text{if } |x| \geq \frac{R}{2Nh+1} \\ \frac{L}{2}x^2 & \text{otherwise} \end{cases}$$
$$f_2(x) = \frac{L}{2}x^2,$$

which are L-smooth convex functions minimized at $x_\star = 0$.

Consider gradient descent $x_i = x_{i-1} - (h_{i-1}/L)\nabla f_j(x_{i-1})$ where for $j = 1, 2$ and $i \in [1 : N]$ with starting point $x_0 = R$. For the sake of simplicity, consider the constant stepsize $h_i = h$ for all $i \in [0 : N-1]$. It is straightforward to check that the objective values at iteration N



(a) $N = 5$

(b) $N = 10$

(c) $N = 25$

(d) $N = 50$

Figure 2.3: Locally optimal stepsizes h^{lopt} vs. iteration number for $N = 5, 10, 25, 50$ with $L = 1$. These optimized methods utilize long steps $h_i > 2/L$ for some i, much alike how Young's method [96] uses long steps satisfying $1 < h_i < L/\mu$ for some i to achieve an accelerated rate for L-smooth and μ-strongly convex quadratics.

are

$$f_1(x_N) = \frac{LR^2}{2}\frac{1}{2Nh+1} \quad (\text{for } 0 \le h \le 2)$$
$$f_2(x_N) = \frac{LR^2}{2}(1-h)^2.$$

The analysis of f_1 shows that acceleration, if possible, would require the algorithm to take long steps exceeding the range $h < 2$. On the other hand, the analysis of f_2 shows that the constant-step gradient descent cannot use a stepsize exceeding $h < 2$ as otherwise one gets divergence.

For the general case when h_i is not constant, a similar line of reasoning with f_1 shows that acceleration without momentum is possible only if $\{h_i\}_{i\in[0:N-1]}$ exceeds $h_i < 2$ for some $i \in [0 : N-1]$. The reasoning and the counter examples f_1 and f_2 are based on [4, Theorem 2] and [5, §4.1.1].

2.6.2 Optimal method for reducing gradient of smooth nonconvex functions

In this section, we construct an optimal FSFOM for decreasing the gradient of L-smooth nonconvex functions. Formally, we choose the function class

$$\mathcal{F} = \{f \mid f \in \mathcal{F}_{-L,L},\ f \text{ has a global minimizer } x_\star\},$$

performance measure[6]

$$\mathcal{E} = \min_{i\in[0:N]} \|\nabla f(x_i)\|^2,$$

[6]In the nonconvex setup, performance measures such as $f(x_N) - f(x_\star)$ or $\|\nabla f(x_N)\|^2$ may not converge to zero as $N \to \infty$ [52, page 3, paragraph 2][99, Remark after Theorem 1].

and initial condition $\mathcal{C} = f(x_0) - f(x_\star) - (1/2)R^2 \leq 0$ with $R > 0$. We parameterize FSFOMs in $\mathcal{M}_N$ as

$$x_i = x_{i-1} - \sum_{j=0}^{i-1} \frac{h_{i,j}}{L} \nabla f(x_j) = x_0 - \sum_{j=0}^{i-1} \frac{\overline{h}_{i,j}}{L} \nabla f(x_j), \tag{2.27}$$

for $i \in [1:N]$. We solve the following instance of ($\mathcal{O}^{\text{outer}}$):

$$\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) = \underset{M \in \mathcal{M}_N}{\text{minimize}} \quad \mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) .$$

Derivation of BnB-PEP-QCQP. Following §2.5 Step (i), formulate the inner optimization problem ($\mathcal{O}^{\text{inner}}$) as

$$\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right)$$
$$= \left(\begin{array}{ll} \text{maximize} & \min_{i \in [0:N]} \|\nabla f(x_i)\|^2 \\ \text{subject to} & f \in \mathcal{F}_{-L,L} \\ & f(x) \geq f(x_\star), \quad \text{for all } x \in \mathbb{R}^d, \\ & x_i = x_0 - \frac{1}{L} \sum_{j=0}^{i-1} \overline{h}_{i,j} \nabla f(x_j) \quad i \in [1:N], \\ & f(x_0) - f(x_\star) \leq R^2, \\ & x_\star = 0, \ f(x_\star) = 0, \end{array} \right)$$

where f and $x_0, \ldots, x_N$ are the decision variables. Write $\overline{h} = \{\overline{h}_{i,j}\}_{0 \leq j < i \leq N}$. To follow the interpolation argument of §2.5 Step (ii), we use the following interpolation result.

Lemma 2.5 (Interpolation inequality for $\mathcal{F}_{-L,L}$)**.** *Let I be a finite index set, and let $\{(x_i, g_i, f_i)\}_{i \in I \cup \{\star\}} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$. Let $L > 0$. There exists $f \in \mathcal{F}_{-L,L}$ satisfying $f(x) \geq f(x_\star) = f_\star$ for all $x \in \mathbb{R}^d$, $f(x_i) = f_i$ for all $i \in I$, and $g_i = \nabla f(x_i)$ for all*

$i \in I$ if and only if [7]

$$
\begin{aligned}
&f_i \geq f_j - \frac{L}{4}\|x_i - x_j\|^2 + \frac{1}{2}\langle g_i + g_j \mid x_i - x_j\rangle + \frac{1}{4L}\|g_i - g_j\|^2, \quad \forall\, i, j \in I \cup \{\star\},\\
&f_\star \leq f_i - \frac{1}{2L}\|g_i\|^2, \quad \forall\, i \in I,\\
&g_\star = 0.
\end{aligned}
$$

Proof. The result follows from translating [99, Theorem 7] into the form of [6, Theorem 3.10]. (Note that journal version of [6, Theorem 3.10] has a sign error that was corrected in its updated arXiv version.) □

Now formulate the inner problem as

$$
\begin{aligned}
&\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right)\\
&= \left(\begin{array}{l}
\text{maximize} \quad t\\
\text{subject to}\\
t \leq \|\nabla f(x_i)\|^2, \quad i \in [0:N],\\
f_i \geq f_j - \frac{L}{4}\|x_i - x_j\|^2 + \frac{1}{2}\langle g_i + g_j \mid x_i - x_j\rangle + \frac{1}{4L}\|g_i - g_j\|^2, \quad i, j \in I_N^\star : i \neq j,\\
f_\star \leq f_i - \frac{1}{2L}\|g_i\|^2, \quad i \in [0:N],\\
g_\star = 0,\ x_\star = 0,\ f_\star = 0,\\
x_i = x_0 - \frac{1}{L}\sum_{j=0}^{i-1} \overline{h}_{i,j} \nabla f(x_j), \quad i \in [1:N],\\
f_0 - f_\star \leq R^2,
\end{array}\right)
\end{aligned}
$$

where $\{x_i, g_i, f_i\}_{i \in I_N} \subseteq \mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}$ and $t \in \mathbb{R}$ are the decision variables. Following §2.5 Step

[7] The first condition can be viewed as a discretization of the following condition [6, Theorem 3.10]: $f \in \mathcal{F}_{-L,L}$ if and only if

$$
f(y) \geq f(x) - \frac{L}{4}\|x - y\|^2 + \frac{1}{2}\langle \nabla f(x) + \nabla f(y) \mid y - x\rangle + \frac{1}{4L}\|\nabla f(x) - \nabla f(y)\|^2, \quad \forall x, y \in \mathbb{R}^d.
$$

(iii), implement the Grammian transformation. Define the Grammian matrices $H \in \mathbb{R}^{d\times(N+2)}$, $G \in \mathbb{S}_+^{N+2}$, and $F \in \mathbb{R}^{1\times(N+1)}$ using the same equations in (2.8), $\{\mathbf{x}_i, \mathbf{g}_i, \mathbf{f}_i\}_{i\in I_N^\star}$ using the same encoding as (2.9), except for $\{\mathbf{x}_i\}_{i\in[1:N]}$, which we define as

$$\mathbf{x}_i = \mathbf{x}_0 - \frac{1}{L}\sum_{j=0}^{i-1} \overline{h}_{i,j}\mathbf{g}_j \in \mathbb{R}^{N+2}, \quad i \in [1:N].$$

Note, $\mathbf{x}_i$ is linearly parameterized by $\overline{h}$. The matrices $B_{i,j}$, $C_{i,j}$, and $a_{i,j}$ are the same as in (2.10) except that they are now parameterized by $\overline{h}$. For $i, j \in I_N^\star$, define

$$\widetilde{A}_{i,j}(\overline{h}) = (\mathbf{g}_i + \mathbf{g}_j) \odot (\mathbf{x}_i - \mathbf{x}_j),$$

so that

$$\mathbf{tr}\, G\widetilde{A}_{i,j}(\overline{h}) = \langle g_i + g_j \mid x_i - x_j\rangle$$

holds. Under the large-scale assumption $d \geq N+2$, we equivalently formulate the inner problem as the SDP:

$$\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right)$$
$$= \left(\begin{array}{ll}
\text{maximize} & t \\
\text{subject to} & \\
t \leq \mathbf{tr}\, GC_{i,\star}, \quad i \in [0:N] & \triangleright \text{ dual var. } \eta_i \geq 0 \\
Fa_{i,j} - \frac{L}{4}\mathbf{tr}\, GB_{i,j}(\overline{h}) + \frac{1}{2}\mathbf{tr}\, G\widetilde{A}_{i,j}(\overline{h}) + \frac{1}{4L}\mathbf{tr}\, GC_{i,j} \leq 0, \quad i,j \in I_N^\star : i \neq j, & \triangleright \text{ dual var. } \lambda_{i,j} \geq 0 \\
Fa_{i,\star} + \frac{1}{2L}\mathbf{tr}\, GC_{i,\star} \leq 0, \quad i \in [0:N], & \triangleright \text{ dual var. } \tau_i \geq 0 \\
-G \preceq 0, & \triangleright \text{ dual var. } Z \succeq 0 \\
Fa_{\star,0} \leq R^2, & \triangleright \text{ dual var. } \nu \geq 0
\end{array}\right)$$

where $F \in \mathbb{R}^{1\times(N+1)}$ and $G \in \mathbb{R}^{(N+2)\times(N+2)}$ are the decision variables. Following §2.5 Step

(iv), construct the dual:

$$
\begin{aligned}
&\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} + \sum_{i \in I_N^\star} \tau_i a_{i,\star} + \nu a_{\star,0} = 0, \\
-\sum_{i \in [0:N]} \eta_i C_{i,\star} + \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(-\frac{L}{4} B_{i,j}(\overline{h}) + \frac{1}{2} \widetilde{A}_{i,j}(\overline{h}) + \frac{1}{4L} C_{i,j} \right) \\
\qquad\qquad + \frac{1}{2L} \sum_{i \in [0:N]} \tau_i C_{i,\star} = Z, \\
\sum_{i \in [0:N]} \eta_i = 1, \\
Z \succeq 0, \\
\nu \geq 0, \ \tau_i \geq 0, \ \lambda_{i,j} \geq 0, \quad i, j \in I_N^\star : i \neq j, \\
\eta_i \geq 0, \quad i \in [0:N],
\end{array} \right)
\end{aligned}
$$

where ν, λ, η, τ, and Z are the decision variables. Assume that strong duality holds. Finally, following §2.5 Step (v), use Lemma 2.3 to pose ($\mathcal{O}^{\text{outer}}$) as the following BnB-PEP-QCQP:

$$\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right)$$

$$
= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} + \sum_{i \in I_N^\star} \tau_i a_{i,\star} + \nu a_{\star,0} = 0, \\
-\sum_{i \in [0:N]} \eta_i C_{i,\star} + \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(-\frac{L}{4} \Theta_{i,j} + \frac{1}{2} \widetilde{A}_{i,j}(\overline{h}) + \frac{1}{4L} C_{i,j} \right) \\
\qquad\qquad + \frac{1}{2L} \sum_{i \in [0:N]} \tau_i C_{i,\star} = Z, \\
\sum_{i \in [0:N]} \eta_i = 1, \\
P \text{ is lower triangular with nonnegative diagonals}, \\
PP^\top = Z, \\
\Theta_{i,j} = B_{i,j}(\overline{h}), \quad i,j \in I_N^\star : i \neq j, \\
\nu \geq 0, \ \tau_i \geq 0, \ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \\
\eta_i \geq 0, \quad i \in [0:N],
\end{array} \right) \tag{2.28}
$$

where ν, λ, η, τ, Z, P, and $\{\Theta_{i,j}\}_{i,j \in I_N^\star : i \neq j}$ are the decision variables. Note that $\{\Theta_{i,j}\}_{i,j \in I_N^\star : i \neq j}$ is introduced as a separate decision variable to formulate the cubic constraints arising from $B_{i,j}(\overline{h})$ as quadratic constraints.

Numerical results. Tables 2.13 and 2.14 present the results of solving (2.28) with $L = 1$, $R = 1$, and $N = 1, \ldots, 5$ using the BnB-PEP Algorithm. We compare the computed optimal FSFOM with the FSFOMs defined by

$$
h_{i,j} = \begin{cases} 1/L & \text{if } j = i - 1, \\ 0 & \text{if } j \in [0 : i-2], \end{cases} \tag{GD}
$$

which is gradient descent and

$$h_{i,j} = \begin{cases} 2/(\sqrt{3}L) & \text{if } j = i-1, \\ 0 & \text{if } j \in [0:i-2], \end{cases} \tag{AKZ}$$

which was proposed by Abbaszadehpeivasti, de Klerk, and Zamani and has the prior state-of-the-art rate [51]. To clarify, the stepsizes $h = \{h_{i,j}\}_{0\leq j<i\leq N}$ and $\overline{h} = \{\overline{h}_{i,j}\}_{0\leq j<i\leq N}$ are as defined in (2.27). We obtain the optimal $\overline{h}^\star$ from the BnB-PEP Algorithm, solve for $h^\star$ with (2.4), and present $h^\star$ in the table. The stepsizes presented in Table 2.14 are certified to be globally optimal by Stage 3. We again observe that the stepsizes obtained at Stage 2, denoted by $\overline{h}^{\text{lopt}}$, were already near-optimal.

Figure 2.4(a) shows that the computed stepsizes $h^\star$ outperforms (AKZ) on the worst-case guarantee of $\min_{i\in[0:N]} \|\nabla f(x_i)\|^2$. To ensure the comparison is precise, we set the precision of the solver to 10^{-10}. We observe in Figure 2.4(a) that the performance improvement diminishes as N increases, which suggests that it will go to zero as $N \to \infty$. This observation leads conjecture that (AKZ) has the exact optimal constant for the leading order term.

Conjecture 2.1. *The optimal FSFOM for reducing gradient of smooth nonconvex functions satisfies*

$$\min_{i\in[0:N]} \|\nabla f(x_i)\|^2 \leq \frac{6\sqrt{3}L(f(x_0)-f_\star)}{8N+3\sqrt{3}} + o(1/N)$$

where the leading term corresponds to the rate for (AKZ) *[51, Theorem 2].*

Also, Figure 2.4(b) shows the solution time to compute the locally optimal stepsizes h^{lopt}.

Momentum form of optimal FSFOM. An interesting observation is that the optimal FSFOM computed by the BnB-PEP Algorithm can be equivalently written in the "momentum

N	# variables	# constraints	Worst-case $\min_{i\in[0:N]} \|\nabla f(x_i)\|^2$			Runtime of the BnB-PEP Algorithm
			Optimal	GD	AKZ	
1	60	70	0.7875254	0.8	0.7875254	0.04 s
2	162	177	0.4902031	0.5	0.4902920	0.41 s
3	365	386	0.3558535	0.363636	0.3559478	9.79 s
4	723	751	0.2793046	0.285714	0.2793919	69.2 s
5	1302	1338	0.2298589	0.235294	0.2299378	607.52 s
10	4138	4128	0.1219308	0.125	0.1219809	2 d 15 h
25	118653	118433	0.0506221	0.051948	0.0506457	4 d 18 h

Table 2.13: Comparison between the performances of the optimal method obtained by solving (2.28) with the BnB-PEP Algorithm, (GD), and (AKZ). The BnB-PEP Algorithm was executed on a standard laptop for $N = 1, 2, \ldots, 10$, and on `MIT Supercloud` for $N = 25$. The performance difference between the optimal method and (AKZ), while small, is genuine, as the difference is greater than precision of the solver, set to 10^{-10}.

N	$h^\star$
1	$[1.154700]$
2	$\begin{bmatrix} 1.157583 & \\ 0.023142 & 1.146857 \end{bmatrix}$
3	$\begin{bmatrix} 1.15762 & & \\ 0.023577 & 1.149576 & \\ 0.003462 & 0.021945 & 1.146719 \end{bmatrix}$
4	$\begin{bmatrix} 1.15762 & & & \\ 0.023584 & 1.149611 & & \\ 0.003535 & 0.022356 & 1.149436 & \\ 0.000549 & 0.003276 & 0.021922 & 1.146717 \end{bmatrix}$
5	$\begin{bmatrix} 1.15762 & & & & \\ 0.023586 & 1.149611 & & & \\ 0.003546 & 0.02236 & 1.149469 & & \\ 0.00061 & 0.003334 & 0.022329 & 1.149433 & \\ 0.000149 & 0.000527 & 0.003263 & 0.02192 & 1.146717 \end{bmatrix}$
10	See Supplementary Information or `Github` repository
25	See Supplementary Information or `Github` repository

Table 2.14: Globally optimal stepsizes obtained by solving (2.28) with the BnB-PEP Algorithm.

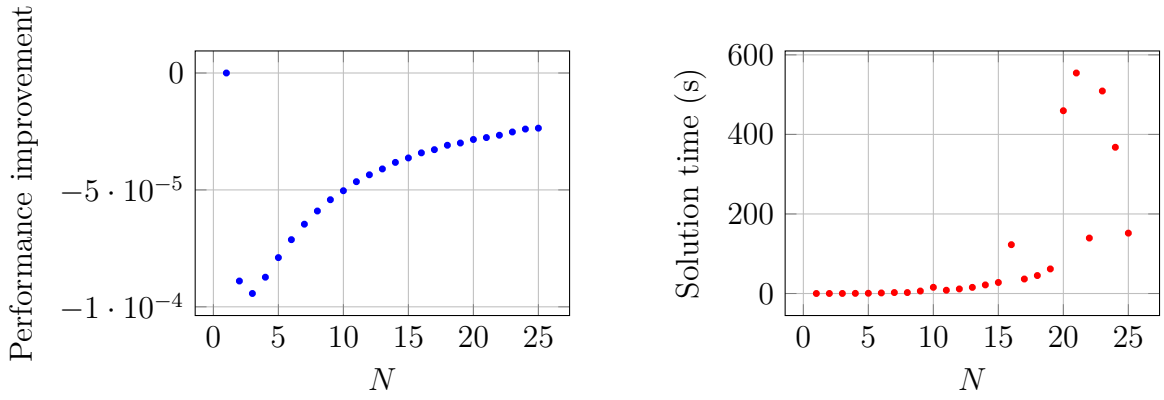


Figure 2.4: Numerical results associated with the stepsizes by solving (2.28) with the BnB-PEP Algorithm. (Left) Improvement in the worst-case guarantee of $\min_{i\in[0:N]} \|\nabla f(x_i)\|^2$ vs. iteration count N. (Right) Runtimes of the BnB-PEP Algorithm to compute locally optimal solutions (including Stages 1 and 2 but excluding Stage 3).

form":

$$\begin{aligned} y_{i+1} &= x_i - \frac{1}{L}\nabla f(x_i) \\ x_{i+1} &= y_{i+1} + \zeta_{i+1}(y_{i+1} - y_i) + \eta_{i+1}(y_{i+1} - x_i) \end{aligned} \tag{2.29}$$

for $i \in [0 : N-1]$, with coefficients $\{\zeta_i\}_{i\in[1:N]}$ and $\{\eta_i\}_{i\in[1:N]}$. Table 2.15 shows the equivalent optimal coefficients.

The class of FSFOMs in momentum form is a strict subset of the class of FSFOMs [5, §4.2]. Nesterov's fast gradient method is expressed in the momentum form (2.29) with $\eta_i = 0$ for all i. Many other accelerated gradient methods such as OGM [4, 8], OGM-G [34], Simple-OGM and SC-OGM [100], FISTA [101], FISTA-G [94], EAG [37, 102], TMM [33, 94], ITEM [9], ORC-F$_\flat$, OBL-F$_\flat$, and OBL-G$_\flat$ [74], and M-OGM-G [75, 94] can all be expressed in the momentum form (2.29).

Furthermore, we observe that the optimal FSFOMs for $N = 6, \ldots, 25$ also admit momentum forms. The list of the stepsizes in the momentum form coefficients $\{\zeta_i^\star\}_{i\in[1:N]}$ and $\{\eta_i^\star\}_{i\in[1:N]}$

N	$\zeta^\star$	$\eta^\star$
1	$[0.0]$	$[0.1547]$
2	$[0.0, 0.146858]$	$[0.157583, 0]$
3	$[0.0, 0.149583, 0.146717]$	$[0.157619, 0, 0]$
4	$[0.0, 0.149626, 0.149426, 0.146702]$	$[0.15762, 0, 0, 0]$
5	$[0.0, 0.149622, 0.149464, 0.149417, 0.146707]$	$[0.15762, 0, 0, 0, 0]$

Table 2.15: Momentum form coefficients $\{\zeta_i^\star\}_{i\in[1:N]}$ and $\{\eta_i^\star\}_{i\in[1:N]}$ for (2.29) of the optimal method obtained by solving (2.28) with the BnB-PEP Algorithm.

for $N = 6, \ldots, 25$ are provided as Supplementary Information and also as a data file at:

https://github.com/Shuvomoy/BnB-PEP-code/blob/main/Misc/zetaeta.jl

2.6.3 Efficient first-order method with respect to a potential function in weakly convex setup

Consider the problem of constructing an FSFOM that efficiently reduces the subgradient magnitude of ρ-weakly convex functions with L-bounded subgradients. Formally, we choose the function class

$$\mathcal{F} = \{f \mid f \in \mathcal{W}_{\rho,L},\ f \text{ has a global minimizer } x_\star\}.$$

Consider FSFOMs of the form

$$x_{i+1} = x_i - \frac{h}{\rho} f'(x_i) \tag{2.30}$$

for $i \in [0 : N]$, where $f'(x_i) \in \partial f(x_i)$ and $h \in \mathbb{R}$ is the stepsize to be determined. Let $\widetilde{L} = L/\rho$. Since $f \in \mathcal{W}_{\rho,L} \Leftrightarrow f/\rho \in \mathcal{W}_{1,\widetilde{L}}$, consider $f \in \mathcal{W}_{1,\widetilde{L}}$ and set $\rho = 1$ without loss of generality.

In this section, we show how to construct an efficient FSFOM by obtaining potential function analyses of FSFOMs and minimizing the guarantee. Unlike in previous sections, our goal here is not to construct an optimal FSFOM but rather is to construct an efficient FSFOM with an analytically tractable potential function analysis.

Using optimization to find potential function analyses of FSFOMs has been studied by Lessard, Recht, and Packard [10] and Taylor, Van Scoy, and Lessard [7] and the philosophy goes further back to the classical Lyapunov stability problem of control theory [103]. Our approach, in particular, closely follows and generalizes the work of Taylor and Bach [38, Appendix C], which finds potential function analyses of FSFOMs and optimize a certain span-search based relaxation of the FSFOMs. The relaxation retains convexity of the optimization, but it is restricted to the convex minimization setup with performance measure $f(x_N) - f(x_\star)$. Our proposed methodology removes this restriction; we can construct efficient FSFOMs in the convex and nonconvex setup with various performance measures. The concrete instance of this section illustrates our methodology and improves upon the prior state-of-the-art rate of Davis and Drusvyatskiy in [52, 104, 105].

2.6.3.1 Measuring stationarity via Moreau envelope

In the nonsmooth nonconvex setup, performance measures commonly used in the convex setup, such as $f(x_N) - f(x_\star)$ or $\operatorname{dist}(0; \partial f(x_N))$, may not go to zero as $N \to \infty$ [52, page 3, paragraph 2]. Therefore, we define a notion of approximate stationarity via the Moreau envelope.

Consider $f \in \mathcal{W}_{1,\widetilde{L}}$ and let $\hat{\rho} > 1$. The proximal operator and Moreau envelope of f are respectively defined as

$$\mathbf{prox}_{(1/\hat{\rho})f}(x) = \operatorname*{argmin}_{y \in \mathbb{R}^n} \left\{ f(y) + \frac{\hat{\rho}}{2}\|y - x\|^2 \right\}, \qquad f_{(1/\hat{\rho})}(x) = \min_{y \in \mathbb{R}^n} \left\{ f(y) + \frac{\hat{\rho}}{2}\|y - x\|^2 \right\}.$$

The Moreau envelope $f_{(1/\hat{\rho})}$ is global underestimator of f that is continuously differentiable: with $y = \mathbf{prox}_{(1/\hat{\rho})f}(x)$, we have

$$x - y = \frac{1}{\hat{\rho}} \nabla f_{(1/\hat{\rho})}(x) \in \partial f(y)$$

[106, (2.13), (2.17)]. If $x_\star$ is a global minimizer of f, then $f_{(1/\hat{\rho})}(x_\star) = f(x_\star)$ and $\|\nabla f_{(1/\hat{\rho})}(x_\star)\| = 0$. The gradient of the Moreau envelope serves as a measure of suboptimality since, with $y = \mathbf{prox}_{(1/\hat{\rho})f}(x)$, we have

$$\begin{aligned} \|y - x\| &= \frac{1}{\hat{\rho}} \|\nabla f_{(1/\hat{\rho})}(x)\|, \\ f(y) &\leq f(x), \\ \operatorname{dist}(0, \partial f(y)) &\leq \|\nabla f_{(1/\hat{\rho})}(x)\|, \end{aligned}$$

for any $x \in \mathbb{R}^d$ [52, page 4]. In other words, if $\|\nabla f_{(1/\hat{\rho})}(x)\|$ is small, then x is near some point y that is nearly stationary for f.

We set $\hat{\rho} = 2$, the simplest choice. For a given sequence of iterates $\{x_i\}_{i \in [0:N] \cup \{\star\}}$, define

$$\begin{aligned} y_i &= \mathbf{prox}_{(1/2)f}(x_i) \\ f'(y_i) &= 2(x_i - y_i). \end{aligned} \tag{2.31}$$

Choose the performance measure

$$\mathcal{E} = \frac{1}{N+1} \sum_{i=0}^{N} \|\nabla f_{(1/2)}(x_i)\|^2 = \frac{1}{N+1} \sum_{i=0}^{N} \|f'(y_i)\|^2,$$

which was also used in [52]. Note that

$$\min_{i\in[0:N]} \|\nabla f_{(1/2)}(x_i)\|^2 \leq \frac{1}{N+1}\sum_{i=0}^{N} \|\nabla f_{(1/2)}\left(x_i\right)\|^2,$$

so any guarantee on $\mathcal{E}$ translates to a guarantee on $\min_{i\in[0:N]} \|\nabla f_{(1/2)}(x_i)\|^2$.

Finally, we provide a few points of clarification. The parameter $\hat{\rho}$ is not used in the method (2.30) and only appears in the analysis of the algorithm. Since $\hat{\rho}$ is strictly larger than 1, the weak convexity parameter, $y = \mathbf{prox}_{(1/\hat{\rho})f}(x)$ is defined as a minimizer of a strongly convex function and therefore uniquely exists. While $f'(x_i) \in \partial f(x_i)$ is chosen arbitrarily in the method (2.30) (the i-th iteration of the method may use *any* subgradient at x_i) the choice of $f'(y_i) \in \partial f(y_i)$ (used in the analysis) is specified by (2.31) and therefore is not arbitrary.

2.6.3.2 Potential function analysis via BnB-PEP

Consider the potential function

$$\psi_k = b_k\left(f_{(1/2)}(x_k) - f_{(1/2)}(x_\star)\right) = b_k\left(f(y_k) - f(x_\star) + \|x_k - y_k\|^2\right), \qquad k \in [0:N+1],$$

where $x_\star$ is a global minimizer of f, $\{b_k\}_{k\in[0:N+1]}$ are parameters to be determined, and $\{y_k\}_{k\in[0:N+1]}$ are as defined in (2.31). Choose the initial condition $\mathcal{C}$ as

$$f_{(1/2)}(x_0) - f_{(1/2)}(x_\star) = f(y_0) - f(x_\star) + \|x_0 - y_0\|^2 \leq R^2.$$

Again, let $x_\star = 0$ and $f(x_\star) = 0$ without loss of generality. If we show

$$\|f'(y_k)\|^2 + \psi_{k+1} - \psi_k \leq c_k\|f'(x_k)\|^2 \tag{2.32}$$

for $k \in [0:N]$, where $\{b_k\}_{k\in[0:N+1]}$ and $\{c_k\}_{k\in[0:N]}$ are nonnegative parameters to be determined, then a telescoping sum provides the rate

$$\begin{aligned}\frac{1}{N+1}\sum_{i=0}^{N}\|f'(y_i)\|^2 &\leq \frac{1}{N+1}\left(\widetilde{L}^2\sum_{i=0}^{N}c_i + \psi_0 - \psi_{N+1}\right)\\ &\leq \frac{1}{N+1}\left(\widetilde{L}^2\sum_{i=0}^{N}c_i + b_0R^2\right).\end{aligned}$$

In a potential function analysis, we effectively choose to be oblivious to how x_k was generated and to the method's prior evaluations of f; we establish the potential function inequality (2.32) one iteration at a time. Due to this restriction, our efficient FSFOM is not expected to be optimal, but it is expected to have a simpler analytically tractable analysis.

Potential function analysis. Let

$$\mathcal{V}_k(h) = \left\{(b_{k+1}, b_k, c_k) \mid \|f'(y_k)\|^2 + \psi_{k+1} \leq \psi_k - c_k\|f'(x_k)\|^2,\ \forall\, x_k \in \mathbb{R}^d,\ f \in \mathcal{W}_{1,\widetilde{L}}\right\},$$

where $x_{k+1} = x_k - hf'(x_k)$, $y_k = x_k - \frac{1}{2}f'(y_k)$, and $y_{k+1} = x_{k+1} - \frac{1}{2}f'(y_{k+1})$, be the set of (b_{k+1}, b_k, c_k) such that (2.32) holds for all $x_k \in \mathbb{R}^d$ and $f \in \mathcal{W}_{1,\widetilde{L}}$. Then, one could consider optimizing the FSFOM by solving

$$\left(\begin{array}{ll}\text{minimize} & \frac{1}{N+1}\left(\widetilde{L}^2\sum_{i=0}^{N}c_i + b_0R^2\right)\\ \text{subject to} & (b_{k+1}, b_k, c_k) \in \mathcal{V}_k(h), \quad k \in [0:N+1],\end{array}\right) \tag{2.33}$$

where $\{b_i\}_{i\in[0:N+1]}$, $\{c_i\}_{i\in[0:N]}$, and $h \in \mathbb{R}$ are the decision variables.

However, checking $(b_{k+1}, b_k, c_k) \in \mathcal{V}_k(h)$ is difficult and (2.33) is difficult to solve, because $\mathcal{W}_{1,\widetilde{L}}$ is a function class without a known interpolation result. In the following, we find $\tilde{\mathcal{V}}_k(h) \subseteq \mathcal{V}_k(h)$ such that membership with respect to $\tilde{\mathcal{V}}_k(h)$ is easy to check. Then we

optimize the FSFOM by solving

$$\left(\begin{array}{ll} \text{minimize} & (1/(N+1))\left(\widetilde{L}^2 \sum_{i=0}^{N} c_i + b_0 R^2\right) \\ \text{subject to} & (b_{k+1}, b_k, c_k) \in \tilde{\mathcal{V}}_k(h), \quad k \in [0:N+1] \end{array}\right) \tag{2.34}$$

where $\{b_i\}_{i\in[0:N+1]}$, $\{c_i\}_{i\in[0:N]}$, and $h \in \mathbb{R}$ are the decision variables. Note, (2.33) is upper bounded by (2.34), since $\mathcal{V}_k(h) \supseteq \tilde{\mathcal{V}}_k(h)$.

In the following, we show that each constraint $(b_{k+1}, b_k, c_k) \in \mathcal{V}_k(h)$ in (2.34) $k \in [0:N+1]$ can be formulated into a QCQP feasibility problem. We then apply the feasibility problem formulations $k \in [0:N+1]$ to obtain the BnB-PEP-QCQP (2.42).

Sufficient SDP for potential function inequality. Note, $(b_{k+1}, b_k, c_k) \in \mathcal{V}_k(h)$ if and only if the optimal value of the following problem is less than or equal to 0:

$$\left(\begin{array}{ll} \text{maximize} & \|f'(y_k)\|^2 + b_{k+1}\left(f(y_{k+1}) - f(x_\star) + \|x_{k+1} - y_{k+1}\|^2\right) \\ & -b_k\left(f(y_k) - f(x_\star) + \|x_k - y_k\|^2\right) - c_k\|f'(x_k)\|^2 \\ \text{subject to} & x_{k+1} = x_k - hf'(x_k) \\ & y_k = x_k - \frac{1}{2}f'(y_k) \\ & y_{k+1} = x_{k+1} - \frac{1}{2}f'(y_{k+1}), \\ & f'(x_\star) = 0,\ x_\star = 0,\ f(x_\star) = 0, \\ & f(w) \geq f(x_\star), \quad w \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \\ & f \in \mathcal{W}_{1,\widetilde{L}}, \end{array}\right) \tag{2.35}$$

where $f \in \mathcal{W}_{1,\widetilde{L}}$ and $x_k, x_{k+1}, y_k, y_{k+1} \in \mathbb{R}^d$ are the decision variables.

Define $\hat{\mathcal{V}}_k(h) \subseteq \mathcal{V}_k(h)$ such that $(b_{k+1}, b_k, c_k) \in \hat{\mathcal{V}}_k(h)$ if and only if the optimal value of the

following problem is less than or equal to 0:

$$\left(\begin{array}{ll}
\text{maximize} & \|f'(y_k)\|^2 + b_{k+1}\left(f(y_{k+1}) - f(x_\star) + \|x_{k+1} - y_{k+1}\|^2\right) \\
& -b_k\left(f(y_k) - f(x_\star) + \|x_k - y_k\|^2\right) - c_k\|f'(x_k)\|^2 \\
\text{subject to} & x_{k+1} = x_k - hf'(x_k) \\
& y_k = x_k - \frac{1}{2}f'(y_k) \\
& y_{k+1} = x_{k+1} - \frac{1}{2}f'(y_{k+1}), \\
& f'(x_\star) = 0,\ x_\star = 0,\ f(x_\star) = 0, \\
& f(w) \geq f(x_\star), \quad w \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \\
& f(w') \geq f(w) + \langle f'(w) \mid w' - w\rangle - \frac{1}{2}\|w' - w\|^2, \\
& \qquad\qquad w, w' \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \\
& \|f'(w)\|^2 \leq \widetilde{L}^2, \quad w \in \{x_k, x_{k+1}, y_k, y_{k+1}\},
\end{array}\right) \tag{2.36}$$

where f, x_k, x_{k+1}, y_k, and y_{k+1} are the decision variables. Since the constraints of (2.35) imply the constraints of (2.36), the optimal value of (2.35) is upper bounded by (2.36) and $\mathcal{V}_k(h) \supseteq \hat{\mathcal{V}}_k(h)$. (Since $\mathcal{W}_{1,\widetilde{L}}$ is a function class with no known interpolation result, two optimal values and the sets are not necessarily equal.) For notational convenience, define

$$\begin{aligned}
&(x_\star, f'(x_\star), f(x_\star)) = (w_\star, g_\star, f_\star), \\
&(x_k, f'(x_k), f(x_k)) = (w_0, g_0, f_0), \\
&(x_{k+1}, f'(x_{k+1}), f(x_{k+1})) = (w_1, g_1, f_1), \\
&(y_k, f'(y_k), f(y_k)) = (w_2, g_2, f_2), \\
&(y_{k+1}, f'(y_{k+1}), f(y_{k+1})) = (w_3, g_3, f_3).
\end{aligned}$$

Then we can express (2.36) equivalently as

$$\left(\begin{array}{ll} \text{maximize} & \|g_2\|^2 + b_{k+1}\left(f_3 - f_\star + \|w_1 - w_3\|^2\right) \\ & -(b_k\left(f_2 - f_\star + \|w_0 - w_2\|^2\right) - c_k\|g_0\|^2 \\ \text{subject to} & w_1 = w_0 - hg_0 \\ & w_2 = w_0 - \tfrac{1}{2}g_2, \\ & w_3 = w_1 - \tfrac{1}{2}g_3, \\ & (w_\star, g_\star, f_\star) = (0,0,0), \\ & f(w_i) \geq f(x_\star), \quad i \in [0:3], \\ & f_i \geq f_j + \langle g_j \mid w_i - w_j\rangle - \tfrac{1}{2}\|w_i - w_j\|^2, \quad i,j \in [0:3]\cup\{\star\} : i \neq j, \\ & \|g_i\|^2 \leq \widetilde{L}^2, \qquad i \in [0:3]\cup\{\star\}, \end{array}\right) \tag{2.37}$$

where $\{w_i, g_i, f_i\}_{i\in[0:3]\cup\{\star\}}$ are the decision variables.

Next, we use the Grammian formulation to formulate (2.37) as an SDP. For $k \in [0:N]$, let

$$\begin{aligned} H^{[k]} &= [w_0 \mid g_0 \mid g_1 \mid g_2 \mid g_3] \in \mathbb{R}^{d\times 5}, \\ G^{[k]} &= H^{[k]\top} H^{[k]} \in \mathbb{S}^5_+, \\ F^{[k]} &= [f_0 \mid f_1 \mid f_2 \mid f_3] \in \mathbb{R}^{1\times 4}. \end{aligned} \tag{2.38}$$

Note that $\mathbf{rank}\, G^{[k]} \leq d$. Define the following notation for selecting columns and elements of

$H^{[k]}$ and $F^{[k]}$:

$$\mathbf{g}_\star = 0 \in \mathbb{R}^5, \mathbf{g}_i = e_{i+2} \in \mathbb{R}^5,\ i \in [0:3],$$

$$\mathbf{f}_\star = 0 \in \mathbb{R}^4, \mathbf{f}_i = e_{i+1} \in \mathbb{R}^4,\ i \in [0:3],$$

$$\mathbf{w}_\star = 0 \in \mathbb{R}^5,$$

$$\mathbf{w}_0 = e_1 \in \mathbb{R}^5,$$

$$\mathbf{w}_1 = \mathbf{w}_0 - h\mathbf{g}_0 \in \mathbb{R}^5,$$

$$\mathbf{w}_2 = \mathbf{w}_0 - \frac{1}{2}\mathbf{g}_2 \in \mathbb{R}^5,$$

$$\mathbf{w}_3 = \mathbf{w}_1 - \frac{1}{2}\mathbf{g}_3 \in \mathbb{R}^5.$$

Furthermore, define

$$A_{i,j}(h) = \mathbf{g}_j \odot (\mathbf{w}_i - \mathbf{w}_j),$$

$$B_{i,j}(h) = (\mathbf{w}_i - \mathbf{w}_j) \odot (\mathbf{w}_i - \mathbf{w}_j),$$

$$C_{i,j} = (\mathbf{g}_i - \mathbf{g}_j) \odot (\mathbf{g}_i - \mathbf{g}_j),$$

$$a_{i,j} = \mathbf{f}_j - \mathbf{f}_i,$$

for $i, j \in [0:3] \cup \{\star\}$. Note that $A_{i,j}(h)$ and $B_{i,j}(h)$ are affine and quadratic as functions of h. This notation defined so that

$$w_i = H^{[k]}\mathbf{w}_i,\ g_i = H^{[k]}\mathbf{g}_i,\ f_i = F^{[k]}\mathbf{f}_i,$$

$$\langle g_j \mid w_i - w_j \rangle = \mathbf{tr}\, G^{[k]} A_{i,j}(h),$$

$$\|w_i - w_j\|^2 = \mathbf{tr}\, G^{[k]} B_{i,j}(h),\ \text{and}$$

$$\|g_i - g_j\|^2 = \mathbf{tr}\, G^{[k,]} C_{i,j}.$$

for $i, j \in [0:3] \cup \{\star\}$. Finally, define

$$Q^{[k]} = Q^{[k]}(h, b_{k+1}, b_k, c_k) = C_{2,\star} + b_{k+1}B_{1,3}(h) - b_k B_{0,2}(h) - c_k C_{0,\star},$$

$$q^{[k]} = q^{[k]}(b_{k+1}, b_k) = b_{k+1}a_{\star,3} - b_k a_{\star,2}$$

for $k \in [0:N]$. Assume the large-scale assumption $d \geq 5$. Using the new notation, equivalently formulate (2.37) as

$$\left(\begin{array}{ll} \text{maximize} & \mathbf{tr}\, G^{[k]} Q^{[k]} + F^{[k]} q^{[k]} \\ \text{subject to} & F^{[k]} a_{i,\star} \leq 0, \quad i \in [0:3], \quad \triangleright \text{ dual var. } \tau_i^{[k]} \geq 0 \\ & F^{[k]} a_{i,j} + \mathbf{tr}\, G^{[k]} \left(A_{i,j}(h) - \frac{1}{2} B_{i,j}(h)\right) \leq 0, \quad i,j \in [0:3] \cup \{\star\} : i \neq j, \quad \triangleright \text{ dual var. } \lambda_{i,j}^{[k]} \geq 0 \\ & \mathbf{tr}\, G^{[k]} C_{i,\star} \leq \widetilde{L}^2 \quad i \in [0:3] \cup \{\star\}, \quad \triangleright \text{ dual var. } \eta_i^{[k]} \geq 0 \\ & -G^{[k]} \preceq 0, \quad \triangleright \text{ dual var. } Z \succeq 0 \end{array}\right) \tag{2.39}$$

where $G^{[k]} \in \mathbb{S}_+^5$ and $F^{[k]} \in \mathbb{R}^{1\times 4}$ are the decision variables.

Next, we dualize. Define $\tilde{\mathcal{V}}_k(h) \subseteq \mathcal{V}_k(h)$ such that $(b_{k+1}, b_k, c_k) \in \tilde{\mathcal{V}}_k(h)$ if and only if the optimal value of the following problem is less than or equal to 0:

$$\left(\begin{array}{ll} \text{minimize} & \widetilde{L}^2 \sum_{i\in[0:3]\cup\{\star\}} \eta_i^{[k]} \\ \text{subject to} & -Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} \left(A_{i,j}(h) - \frac{1}{2} B_{i,j}(h)\right) + \sum_{i\in[0:3]\cup\{\star\}} \eta_i C_{i,\star} = Z^{[k]}, \\ & -q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0, \\ & \lambda_{i,j}^{[k]} \geq 0,\ \eta_i^{[k]} \geq 0, \quad i,j \in [0:3] \cup \{\star\} : i \neq j, \\ & \tau_i^{[k]} \geq 0, \quad i \in [0:3], \\ & Z^{[k]} \succeq 0, \end{array}\right) \tag{2.40}$$

where $\{\eta^{[k]}\}_{i\in[0:3]\cup\{\star\}}$, $\{\lambda_{i,j}^{[k]}\}_{i,\in[0:3]\cup\{\star\}:i\neq j}$, $\{\tau_i^{[k]}\}_{i\in[0:3]}$, and $Z^{[k]} \in \mathbb{S}_+^5$ are the decision variables. By weak duality, the optimal value of (2.39) is upper bounded by (2.40) and $\hat{\mathcal{V}}_k(h) \supseteq \tilde{\mathcal{V}}_k(h)$. (While we expect strong duality to usually hold, we do not need to assume it.) Observe that for the optimal value of (2.40) to be less than equal to zero, we must have $\eta^{[k]} = 0$. Hence,

(2.40) simplifies into the feasibility problem

$$\left(\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & -Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} \left(A_{i,j}(h) - \frac{1}{2}B_{i,j}(h)\right) = Z^{[k]}, \\ & -q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0, \\ & \lambda_{i,j}^{[k]} \geq 0, \quad i,j \in [0:3]\cup\{\star\} : i \neq j, \\ & \tau_i^{[k]} \geq 0, \quad i \in [0:3], \\ & Z^{[k]} \succeq 0 \end{array}\right) \tag{2.41}$$

for $k \in [0:N]$.

Optimizing potential function analysis. We have shown that existence of a feasible point for (2.41) implies $(b_{k+1}, b_k, c_k) \in \tilde{\mathcal{V}}_k(h)$, which in turn implies (2.32) holds. Finally, use Lemma 2.3 to formulate (2.34) as the following BnB-PEP-QCQP:

$$\left(\begin{array}{ll} \text{minimize} & \widetilde{L}^2 \left[\frac{1}{N+1}\left(\sum_{i=0}^N c_i + \frac{R^2}{\widetilde{L}^2} b_0\right)\right] \\ \text{subject to} & -Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} \left(A_{i,j}(h) - \frac{1}{2}\Theta_{i,j}^{[k]}\right) = Z^{[k]}, \quad k \in [0:N], \\ & -q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0, \quad k \in [0:N], \\ & \lambda_{i,j}^{[k]} \geq 0, \quad i,j \in [0:3]\cup\{\star\} : i \neq j, \quad k \in [0:N], \\ & \tau_i^{[k]} \geq 0, \quad i \in [0:3], \quad k \in [0:N], \\ & P^{[k]} \text{ is lower triangular with nonnegative diagonals}, \quad k \in [0:N], \\ & P^{[k]}(P^{[k]})^\top = Z^{[k]}, \quad k \in [0:N], \\ & \Theta_{i,j}^{[k]} = B_{i,j}(h), \quad i,j \in [0:3]\cup\{\star\} : i \neq j, \quad k \in [0:N], \end{array}\right) \tag{2.42}$$

where the decision variables are $\{b_k\}_{k\in[0:N+1]}$, $\{c_k\}_{k\in[0:N]}$, $\{\lambda^{[k]}\}_{k\in[0:N]}$, $\{\tau^{[k]}\}_{k\in[0:N]}$, $\{Z^{[k]}\}_{k\in[0:N]}$, $\{P^{[k]}\}_{k\in[0:N]}$, $\{\Theta^{[k]}\}_{k\in[0:N]}$, and h. We call $\lambda^{[k]}$, $\tau^{[k]}$, and $Z^{[k]}$ the inner-dual variables. Note that $\{\Theta^{[k]}\}_{k\in[0:N]}$ is introduced as a separate decision variable to formulate the cubic con-

straints arising from $B_{i,j}(h)$ as quadratic constraints. A feasible solution of (2.42) provides a performance guarantee on the FSFOM defined by the stepsize h.

2.6.3.3 Numerical results and analytical convergence proofs

Tables 2.16 and 2.17 shows the results of solving (2.42) with $\widetilde{L} = 1$, $R = 0.1$ and $N = 1, \ldots, 5, 10, 25$ using the BnB-PEP Algorithm. Similar to our previous experiments, we empirically observe that the locally optimal stepsizes obtained at Stage 2, denoted by h^{lopt}, were already near-optimal. This motivates us to apply just the first two stages of the BnB-PEP Algorithm for $N = 26, \ldots, 100$. In Figure 2.5, we compare the locally optimal stepsizes h^{lopt} with the stepsize $h = R/(\widetilde{L}\sqrt{(N+1)})$ reported in the proof of [52, Theorem 3.1].

In the following, we use the numerical results to obtain an analytical form of the stepsize and its convergence proof.

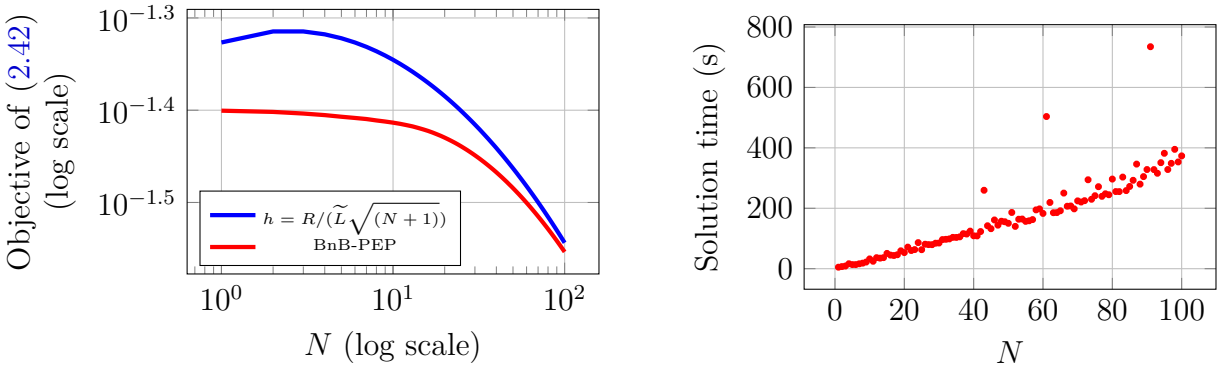


Figure 2.5: Numerical results for the locally optimal stepsizes by solving (2.42) the BnB-PEP Algorithm. We have verified global optimality of the locally optimal stepsizes for $N = 1, 2, \ldots, 25$. (Left) Objective value $\widetilde{L}^2[(\sum_{i=0}^{N} c_i + (R^2/\widetilde{L}^2)b_0)/(N+1)]$ for $h = R/(\widetilde{L}\sqrt{(N+1)})$ (as prescribed by [52]) and for the stepsizes computed by the BnB-PEP Algorithm vs. iteration count N. (Right) Runtimes of the BnB-PEP Algorithm (including Stages 1 and 2 but excluding Stage 3).

N	# variables	# constraints	Worst-case $\widetilde{L}^2[(\sum_{i=0}^{N} c_i + (R^2/\widetilde{L}^2)b_0)/(N+1)]$		Runtime of the BnB-PEP Algorithm
			Optimal	$h = \frac{R}{(\widetilde{L}\sqrt{(N+1)})}$	
1	735	780	0.0398	0.04717	0.01 s
2	1102	1170	0.0396	0.04837	0.22 s
3	1469	1560	0.0394	0.048415	0.34 s
4	1836	1950	0.0392157	0.0480887	0.81 s
5	2203	2340	0.039026	0.0476315	2.6 s
10	4038	4290	0.038113	0.0451378	8 h 40 m
25	9543	10140	0.035692	0.0397182	19 h 15 m

Table 2.16: Comparison between the performances of the optimal method obtained by solving (2.42) with the BnB-PEP Algorithm and the method with $h = R/(\widetilde{L}\sqrt{(N+1)})$ as prescribed by [52]. The BnB-PEP Algorithm was executed on a standard laptop for $N = 1, 2, \ldots, 10$, and on `MIT Supercloud` for $N = 25$.

N	$h^\star$
1	0.01
2	0.0099664
3	0.0099331
4	0.0099
5	0.00986714
10	0.0097061
25	0.0092553

Table 2.17: Globally optimal stepsize obtained by solving (2.42) with the BnB-PEP Algorithm.

Structured inner-dual variables. To find an analytical proof of an FSFOM defined by h, i.e., to find a feasible solution of (2.42), we use the methodology of §2.4.3 to we find the following pattern of the optimal inner-dual variables $\lambda^{\star[k]}$, $\tau^{\star[k]}$, and $Z^{\star[k]}$ for all $k \in [0:N]$:

- Only $\lambda^{\star[k]}_{2,3}$, $\lambda^{\star[k]}_{2,0}$, and $\lambda^{\star[k]}_{0,2}$ are nonzero. Furthermore, $\lambda^{\star[k]}_{2,0} = \lambda^{\star[k]}_{0,2}$.
- Only $\tau_2^{\star[k]}$ is nonzero.
- Only $Z^{\star[k]}_{2,2}$, $Z^{\star[k]}_{4,2} = Z^{\star[k]}_{2,4}$, $Z^{\star[k]}_{5,2} = Z^{\star[k]}_{2,5}$, $Z^{\star[k]}_{5,4} = Z^{\star[k]}_{4,5}$, $Z^{[k]}_{4,4}$, and $Z^{[k]}_{5,5}$ are nonzero. Furthermore, $Z^{\star[k]}_{5,2} = -Z^{\star[k]}_{4,2}$, $Z^{\star[k]}_{4,4} = Z^{\star[k]}_{5,5}$, and $Z^{\star[k]}_{4,4} = -Z^{\star}_{5,4}$.

Since the pattern is the same for all k, we enforce this pattern for a given k in the constraint set of (2.42). This leads to a system of equations, and after some tedious but elementary calculations, we get the simplified form

$$\begin{aligned} 4 + (1-2h)b_{k+1} &= b_k, \\ \lambda^{[k]}_{2,3} &= b_{k+1}, \\ \lambda^{[k]}_{2,0} = \lambda^{[k]}_{0,2} &= 2hb_{k+1}, \\ \tau_2^{[k]} &= b_k - b_{k+1}, \end{aligned} \tag{2.43}$$

for all $k \in [0:N]$. We share Mathematica code for calculating (2.43) symbolically as follows:

https://github.com/Shuvomoy/BnB-PEP-code/blob/main/Misc/potential.nb

The solution numerically produced by BnB-PEP Algorithm have $b_{N+1} = 0$, so we use that terminal value. Furthermore, from the numerical results, we also empirically observe that $c_k^\star$, $h^\star$, and $b^\star_{k+1}$ follow the relationship $c_k = h^2 b_{k+1}$ for all $k \in [0:N]$.

Convergence proof 1: Analytical solution to the BnB-PEP QCQP. We restrict[8] our consideration to $h \in (0, 1/2]$. The recursive relationship $4 + (1-2h)b_{k+1} = b_k$ of (2.43) with the terminal condition $b_{N+1} = 0$ implies

$$b_k = \frac{2}{h}\left(1 - (1-2h)^{N+1-k}\right) \quad \text{for } k \in [0 : N+1]. \tag{2.44}$$

This formula and the resulting values from (2.43) indeed make $\{b_k\}_{k\in[0:N+1]}$, $\{c_k\}_{k\in[0:N]}$, $\{\lambda^{[k]}\}_{k\in[0:N]}$, and $\{\tau^{[k]}\}_{k\in[0:N]}$ non-negative. Plugging values from (2.43) and $c_k = h^2 b_{k+1}$ into (2.42) we get

$$\begin{aligned}
&-q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0,\\
&-Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} \left(A_{i,j}(h) - \frac{1}{2}B_{i,j}(h)\right) = Z^{[k]},
\end{aligned}$$

$$Z^{[k]} = \begin{pmatrix}
0 & 0 & 0 & 0 & 0\\
0 & \frac{1}{2}h^2 b_{k+1} & 0 & -\frac{1}{4}hb_{k+1} & \frac{1}{4}hb_{k+1}\\
0 & 0 & 0 & 0 & 0\\
0 & -\frac{1}{4}hb_{k+1} & 0 & \frac{1}{8}b_{k+1} & -\frac{1}{8}b_{k+1}\\
0 & \frac{1}{4}hb_{k+1} & 0 & -\frac{1}{8}b_{k+1} & \frac{1}{8}b_{k+1}
\end{pmatrix}$$

for all $k \in [0 : N]$. The eigenvalues of $Z^{[k]}$ are $\{0, 0, 0, 0, (1/4)(1+2h^2)b_{k+1}\}$, so $Z^{[k]} \succeq 0$. Thus the values of $\{b, c, \lambda, \tau\}$ defined by (2.43) and (2.44) is a feasible solution of (2.42), and we have proved the following theorem.

Theorem 2.1. *Let $N \in \mathbb{N}$. Let $f \in \mathcal{W}_{1,L}$ have a global minimizer $x_\star$. Let $R > 0$, and let*

[8] While unlikely, it is possible that a stepsize $h \notin (0, 1/2]$ achieves a better performance for some parameter values. (Our numerical experiments indicate that this is not the case.) If so, our choice of $h \in (0, 1/2]$ excludes this better choice. Regardless, our resulting choice of h and its performance guarantee are valid.

$x_0 \in \mathbb{R}^d$ satisfy the initial condition $f(y_0) - f(x_\star) + \|x_0 - y_0\|^2 \leq R^2$. Consider the method

$$x_{k+1} = x_k - hf'(x_k)$$

for $k \in [0:N]$, where $f'(x_k)$ is an arbitrary subgradient of f at x_k. Let $y_k = \mathbf{prox}_{(1/2)f}(x_k)$ and

$$\psi_k = b_k \left(f(y_k) - f(x_\star) + \|x_k - y_k\|^2 \right)$$

for $k \in [0:N+1]$. If $h \in (0, 1/2]$, $4 + (1-2h)b_{k+1} = b_k$ for $k \in [0:N]$, $b_{N+1} = 0$, and $c_k = h^2 b_{k+1}$ for $k \in [0:N]$, then

$$\|f'(y_k)\|^2 + \psi_{k+1} - \psi_k \leq c_k \|f'(x_k)\|^2$$

for all $k \in [0:N]$, and

$$\frac{1}{N+1} \sum_{i=0}^{N} \|\nabla f_{(1/2)}(x_i)\|^2 \leq \frac{1}{N+1} \left(\widetilde{L}^2 \sum_{i=0}^{N} c_i + b_0 R^2 \right).$$

To find the optimal $h^\star$, one can minimize the bound of Theorem 2.1. For notational simplicity, define $\kappa = R/L$. With (2.44), the analytical performance measure minimizes is

$$\begin{aligned} &\frac{\widetilde{L}^2}{N+1} \left(\sum_{i=0}^{N} c_i + \kappa^2 b_0 \right) \\ =&\frac{\widetilde{L}^2}{N+1} \left[-1 + (1-2h)^{N+1} + 2h(N+1) + \kappa^2 \frac{2}{h} \left(1 - (1-2h)^{N+1} \right) \right]. \end{aligned} \tag{2.45}$$

As an aside, one can directly verify the nonnegativity of (2.45) with Bernoulli's lower bound inequality, which states that $(1+x)^r \geq 1 + rx$ for any positive integer $r \geq 1$ and any real $x \geq -1$ [107, page 1]. Plotting (2.45) for different values κ and N reveals that it has a unique

minimum in h. Hence, the optimal $h^\star$ can be found by setting the derivative equal to zero:

$$\frac{2\kappa^2\left((1-2h)^N-1\right)}{h^2}+\frac{4\kappa^2 N(1-2h)^N}{h}-2(N+1)\left((1-2h)^N-1\right)=0.$$

However, this equation does not seem to admit a simple algebraic solution.

To find a simpler analytical stepsize, we construct an upper bound of (2.45) that does admits a closed-form minimizer:

$$\begin{aligned}
&\frac{\widetilde{L}^2}{N+1}\left[-1+(1-2h)^{N+1}+2h(N+1)+\kappa^2\frac{2}{h}\left(1-(1-2h)^{N+1}\right)\right]\\
\overset{a)}{<}&\frac{\widetilde{L}^2}{N+1}\left[-1+\frac{1}{2(N+1)h+1}+2h(N+1)+\kappa^2\frac{2}{h}\right]\\
\overset{b)}{<}&\frac{\widetilde{L}^2}{N+1}\left[-1+\frac{1}{2(N+1)h}+2h(N+1)+\kappa^2\frac{2}{h}\right]\\
=&\frac{\widetilde{L}^2}{N+1}\left[-1+\frac{1}{2h}\left(\frac{1}{N+1}+4\kappa^2\right)+2h(N+1)\right].
\end{aligned}\tag{2.46}$$

Here, $a)$ uses $1-(1-2h)^{N+1}<1$ and

$$(1-2h)^{N+1}\leq\frac{1}{2(N+1)h+1},$$

that follows from Bernoulli's upper bound inequality $(1+a)^r\leq 1/(1-ra)$ for $a\in[-1,0], r\in\mathbb{N}$ [107, page 1] along with $h\in(0,1/2]$ and $b)$ uses $1/\left(2(N+1)h+1\right)<1/\left(2(N+1)h\right)$. The minimum of (2.46) is achieved at

$$h^\star_{\text{ub}}=\frac{\sqrt{4\kappa^2(N+1)+1}}{2(N+1)}.$$

Plugging this back into (2.46), we have the following analytical performance guarantee:

$$\frac{\widetilde{L}^2}{N+1}\left(-1+\frac{1}{2h^\star_{\mathrm{ub}}}\left(\frac{1}{N+1}+4\kappa^2\right)+2h^\star_{\mathrm{ub}}(N+1)\right)=\frac{\widetilde{L}^2\left(2\sqrt{4\kappa^2(N+1)+1}-1\right)}{N+1}.$$

We have proved the following corollary.

Corollary 2.1. *In the setup of Theorem 2.1, the choice*

$$h=\frac{\sqrt{4\kappa^2(N+1)+1}}{2(N+1)},$$

yields the rate

$$\frac{1}{N+1}\sum_{i=0}^{N}\|\nabla f_{(1/2)}(x_i)\|^2\leq\frac{\widetilde{L}^2\left(2\sqrt{4\kappa^2(N+1)+1}-1\right)}{N+1}.$$

The result of Corollary 2.1 strictly improves upon the prior state-of-the-art rate [52, Theorem 3.1]

$$\frac{1}{N+1}\sum_{i=0}^{N}\|\nabla f_{(1/2)}(x_i)\|^2\leq\widetilde{L}^2\frac{4\kappa}{\sqrt{N+1}}$$

for large enough N satisfying $N>(9-64\kappa^2)/(64\kappa^2)$. (This stated rate is obtained by plugging the stepsize found in the proof of [52, Theorem 3.1] into [52, Equation 3.4] in the noiseless setup. The claimed rate [52, Equation 3.5] has an error in the constant.)

Convergence proof 2: Classical analytical proof. While the previous analytical proof is rigorous, its reliance on the BnB-PEP-QCQP formulation makes it inaccessible to those who do not already understand the PEP methodology. Therefore, we translate the proof into a classical form that does not depend on the PEP methodology.

To clarify, the discovery of the previous proof was assisted by numerical solutions, but its correctness can be verified by humans without the aid of any numerical solvers. The benefit of the following alternate proof is its accessibility; the previous equivalent proof was equally correct and rigorous.

Alternate proof of Theorem 2.1. The proof forms nonnegative combinations of valid inequalities and organizes the terms to establish the stated result. The arguably mysterious weights of the nonnegative combinations correspond to the values of the inner-dual-variables listed in (2.43).

Note that

$$\begin{aligned} f(y_{k+1}) - f(y_k) + \langle f'(y_{k+1}) \mid y_k - y_{k+1} \rangle - \frac{1}{2}\|y_k - y_{k+1}\|^2 \leq 0, \\ f(x_k) - f(y_k) + \langle f'(x_k) \mid y_k - x_k \rangle - \frac{1}{2}\|y_k - x_k\|^2 \leq 0, \\ f(y_k) - f(x_k) + \langle f'(y_k) \mid x_k - y_k \rangle - \frac{1}{2}\|x_k - y_k\|^2 \leq 0, \end{aligned}$$

by weak convexity of f, and

$$f(x_\star) - f(y_k) \leq 0$$

by the assumption that $x_\star$ is a global minimizer. Multiplying the last four inequalities with the nonnegative weights b_{k+1}, $2hb_{k+1}$, $2hb_{k+1}$, and $b_k - b_{k+1} = 4 - 2hb_{k+1}$ (nonnegativity follows from (2.44)), respectively, and then adding them together, we obtain

$$\begin{aligned} 0 \geq & b_{k+1}\left(f(y_{k+1}) - f(y_k) + \langle f'(y_{k+1}) \mid y_k - y_{k+1} \rangle - \frac{1}{2}\|y_k - y_{k+1}\|^2\right) + \\ & 2hb_{k+1}\left(f(x_k) - f(y_k) + \langle f'(x_k) \mid y_k - x_k \rangle - \frac{1}{2}\|y_k - x_k\|^2\right) + \\ & 2hb_{k+1}\left(f(y_k) - f(x_k) + \langle f'(y_k) \mid x_k - y_k \rangle - \frac{1}{2}\|x_k - y_k\|^2\right) + \end{aligned}$$

$$
\begin{aligned}
&(4-2hb_{k+1})\left(f(x_\star)-f(y_k)\right)\\
\overset{a)}{=}&b_{k+1}(f(y_{k+1})-f(x_\star))-b_k(f(y_k)-f(x_\star))+\\
&\frac{1}{8}b_{k+1}\Big[-4\|f'(x_k)\|^2h^2-2\left\langle f'(y_k)\mid 2hf'(x_k)+f'(y_{k+1})\right\rangle\\
&+4\left\langle f'(x_k)\mid f'(y_{k+1})\right\rangle h+(4h-1)\|f'(y_k)\|^2+3\|f'(y_{k+1})\|^2\Big]\\
=&b_{k+1}(f(y_{k+1})-f(x_\star))-b_k(f(y_k)-f(x_\star))\\
&+\frac{1}{4}b_{k+1}\left(\|f'(y_{k+1})\|^2-4h^2\|f'(x_k)\|^2-(1-2h)\|f'(y_k)\|^2\right)+\\
&b_{k+1}\Big[\frac{1}{8}\Big(-4\|f'(x_k)\|^2h^2-2\left\langle f'(y_k)\mid 2hf'(x_k)+f'(y_{k+1})\right\rangle\\
&+4\left\langle f'(x_k)\mid f'(y_{k+1})\right\rangle h+(4h-1)\|f'(y_k)\|^2+3\|f'(y_{k+1})\|^2\Big)-\\
&\frac{1}{4}\left(\|f'(y_{k+1})\|^2-4h^2\|f'(x_k)\|^2-(1-2h)\|f'(y_k)\|^2\right)\Big]\\
=&b_{k+1}(f(y_{k+1})-f(x_\star))-b_k(f(y_k)-f(x_\star))\\
&+\frac{1}{4}b_{k+1}\left(\|f'(y_{k+1})\|^2-4h^2\|f'(x_k)\|^2-(1-2h)\|f'(y_k)\|^2\right)\\
&+\frac{1}{8}\|2hf'(x_k)-f'(y_k)+f'(y_{k+1})\|^2\\
\overset{b)}{\geq}&b_{k+1}(f(y_{k+1})-f(x_\star))-b_k(f(y_k)-f(x_\star))\\
&+\frac{1}{4}b_{k+1}\left(\|f'(y_{k+1})\|^2-4h^2\|f'(x_k)\|^2-(1-2h)\|f'(y_k)\|^2\right)\\
\overset{c)}{=}&b_{k+1}(f(y_{k+1})-f(x_\star))-b_k(f(y_k)-f(x_\star))+\|f'(y_k)\|^2\\
&+\frac{b_{k+1}}{4}\|f'(y_{k+1})\|^2-\frac{b_k}{4}\|f'(y_k)\|^2-c_k\|f'(x_k)\|^2\\
=&\|f'(y_k)\|_2+b_{k+1}\left(f(y_{k+1})-f(x_\star)\|+\|y_{k+1}-w_1\|^2\right)\\
&-b_k\left(f(y_k)-f(x_\star)+\|x_k-y_k\|^2\right)-c_k\|f'(x_k)\|^2\\
\overset{d)}{=}&\|f'(y_k)\|^2+\psi_{k+1}-\psi_k-c_k\|f'(x_k)\|^2,
\end{aligned}
$$

where a) uses (2.43) and

$$y_k - y_{k+1} = hf'(x_k) - \frac{1}{2}f'(y_k) + \frac{1}{2}f'(y_{k+1}),$$
$$x_k - y_k = \frac{1}{2}f'(y_k),$$

b) removes the non-negative term in the previous line (the last term), c) uses (2.43), and d) uses the definition of ψ_k. □

Remark. The convergence proof above nowhere utilizes the assumption that $x_\star$ exists; it only requires that $f_\star = \inf_x f(x) > -\infty$. There was no a priori guarantee that we would get such a proof since the BnB-PEP-QCQP was allowed to use the existence of $x_\star$. However, BnB-PEP-QCQP chose not use that assumption in producing an optimal solution.

2.7 Conclusion

The contribution of the BnB-PEP methodology is threefold. First, BnB-PEP poses the problem of finding the optimal fixed-step first-order method for convex or nonconvex, smooth or nonsmooth optimization as a nonconvex but practically tractable QCQP called BnB-PEP-QCQP. Second, our methodology presents the BnB-PEP Algorithm that solves the BnB-PEP-QCQP to certifiable global optimality. Through exploiting specific problem structures, the BnB-PEP Algorithm outperforms the latest off-the-shelf implementations by orders of magnitude, reducing the solution-time from hours to seconds and weeks to minutes. Third, we test the BnB-PEP methodology on a variety of problem setups for which the prior methodologies failed and obtain first-order methods with bounds, some numerical and some analytical, that improve upon prior state-of-the-art results.

As the BnB-PEP offers significantly more flexibility compared to the prior performance

estimation methodologies, we expect there to be many fruitful future directions of work utilizing the BnB-PEP methodology. In particular, using the BnB-PEP to analyze and generate composite optimization methods [6, 45, 46], randomized and stochastic methods [38, 108], monotone operator and splitting methods [35, 36, 48, 49], mirror descent methods [50], and adaptive methods [109] are all interesting directions of future work. Recently, novel worst-case convergence rates for nonlinear conjugate gradient methods were established in [2] using the BnB-PEP methodology.

2.A Appendix for Chapter 2

2.A.1 Function class

The BnB-PEP methodology applies to quadratically representable function classes. We say $\mathcal{F}$ is *quadratically representable* if the membership $f \in \mathcal{F}$ is defined by an inequality of the form

$$c_0 f(y) \geq c_1 f(x) + q(x, y, u, v), \quad \forall u \in \partial f(x),\, v \in \partial f(y),\, x, y \in \mathbb{R}^d,$$

where

$$\begin{aligned} q(x, y, u, v) \triangleq\, & c_2 \langle x \mid x \rangle + c_3 \langle y \mid y \rangle + c_4 \langle u \mid u \rangle + c_5 \langle v \mid v \rangle + c_6 \langle x \mid y \rangle + c_7 \langle x \mid u \rangle \\ & + c_8 \langle x \mid v \rangle + c_9 \langle y \mid u \rangle + c_{10} \langle y \mid v \rangle + c_{11} \langle u \mid v \rangle + c_{12}, \end{aligned}$$

with $c_i \in \mathbb{R}$ for $i \in [0:12]$ along with an optional inequality of the form

$$\|u\|_2 \leq M, \quad \forall u \in \partial f(x),\, x \in \mathbb{R}^d,$$

for some $M > 0$. Many of the commonly considered function classes are quadratically representable, and we list a few in the following. The class of L-smooth convex functions $\mathcal{F}_{0,L}$ satisfies [59, Theorem 2.1.5, Equation (2.1.10)]

$$f(y) \geq f(x) + \langle \nabla f(x) \mid y - x \rangle + \frac{1}{2L} \|\nabla f(x) - \nabla f(y)\|^2, \quad \forall x, y \in \mathbb{R}^d.$$

(L-smooth functions are differentiable everywhere.) The class of L-smooth μ-strongly convex functions $\mathcal{F}_{\mu,L}$ satisfies [9, Theorem 1]

$$\begin{aligned} f(y) \geq f(x) + \langle \nabla f(x) \mid y - x \rangle + \frac{1}{2(L-\mu)} \|\nabla f(x) - \nabla f(y)\|^2 \\ + \frac{L\mu}{2(L-\mu)} \|x - y\|^2 - \frac{\mu}{2(L-\mu)} \langle \nabla f(x) - \nabla f(y) \mid x - y \rangle, \quad \forall x, y \in \mathbb{R}^d. \end{aligned}$$

The class of L-smooth nonconvex functions $\mathcal{F}_{-L,L}$ satisfies [99, Theorem 6]

$$f(y) \geq f(x) + \langle \nabla f(x) \mid y - x \rangle + \frac{1}{2L} \|\nabla f(x) - \nabla f(y)\|^2 - \frac{L}{4} \|x - y - \frac{1}{L}(\nabla f(x) - \nabla f(y))\|^2, \quad \forall x, y \in \mathbb{R}^d.$$

which is also equivalent to [6, Theorem 3.10]

$$f(y) \geq f(x) - \frac{L}{4} \|x - y\|^2 + \frac{1}{2} \langle \nabla f(x) + \nabla f(y) \mid y - x \rangle + \frac{1}{4L} \|\nabla f(x) - \nabla f(y)\|^2, \quad \forall x, y \in \mathbb{R}^d.$$

The class of ρ-weakly convex functions $\mathcal{W}_{\rho,\infty}$ is satisfies [52, Lemma 2.1]

$$f(y) \geq f(x) + \langle u \mid y - x \rangle - \frac{\rho}{2} \|x - y\|^2, \quad \forall u \in \partial f(x),\ x, y \in \mathbb{R}^d.$$

The class of nonsmooth convex functions with L-bounded subgradient $\mathcal{F}_{0,\infty}^{L}$ satisfies [6, Definition 3.1]

$$f(y) \geq f(x) + \langle u \mid y - x \rangle, \quad \forall u \in \partial f(x),\, x, y \in \mathbb{R}^d,$$
$$\|u\|_2 \leq L, \quad \forall\, u \in \partial f(x),\, x \in \mathbb{R}^d.$$

The class of ρ-weakly convex functions with L-bounded subgradients $\mathcal{W}_{\rho,L}$ satisfies [52, Lemma 2.1], [6, §3.1 (c)]

$$f(y) \geq f(x) + \langle u \mid y - x \rangle - \frac{\rho}{2}\|x - y\|^2, \quad \forall u \in \partial f(x),\, x, y \in \mathbb{R}^d,$$
$$\|u\|_2 \leq L, \quad \forall\, u \in \partial f(x),\, x \in \mathbb{R}^d.$$

Some, but not all, of the quadratically representable functions classes have interpolation results analogous to Lemma 2.2. In particular, $\mathcal{F}_{0,L}$, $\mathcal{F}_{\mu,L}$, $\mathcal{F}_{-L,L}$, $\mathcal{W}_{\rho,\infty}$, and $\mathcal{F}_{0,\infty}^{L}$ have interpolation results ([5, Theorem 4], [6, Theorem 3.10], [6, Theorem 3.5]), while $\mathcal{W}_{\rho,L}$ does not. We can still use the BnB-PEP methodology without an interpolation condition, as we do in §2.6.3, but we loose the tightness guarantee.

2.A.2 Discussion on the strong duality assumption

Consider the setup of §2.3, and let us *not* assume strong duality. Then, by weak duality, we have

$$\mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C}) \leq \overline{\mathcal{R}}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C}),$$

where $\mathcal{R}$ is as given by (2.12), $\overline{\mathcal{R}}$ is as defined in (2.13), and $M(\alpha)$ is the FSFOM parameterized by the stepsize list $\alpha = \{\alpha_{i,j}\}_{0 \leq j < i \leq N}$.

Strong duality does provably hold generically. Let

$$A^{\text{nice}} = \{\alpha \,|\, \alpha_{i,i-1} \neq 0, \, \forall \, i = 1, \ldots, N\}$$

be the set of "nice" stepsize lists.

Lemma 2.6. *[5, Theorem 6] If $\alpha \in A^{\text{nice}}$, then strong duality holds, i.e.,*

$$\mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C}) = \overline{\mathcal{R}}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C}), \quad \forall \, \alpha \in A^{\text{nice}}.$$

Note that α is an $N(N+1)/2$-dimensional and the set of $\alpha \notin A^{\text{nice}}$ is lower-dimensional. In this sense, strong duality holds generically. When we solve the BnB-PEP-QCQP, we find

$$\alpha^\star \in \operatorname*{argmin}_{\alpha} \overline{\mathcal{R}}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C}).$$

In all of our experiments, we observe, a posteriori, that $\alpha^\star \in A^{\text{nice}}$. So

$$\mathcal{R}(M(\alpha^\star), \mathcal{E}, \mathcal{F}, \mathcal{C}) = \overline{\mathcal{R}}(M(\alpha^\star), \mathcal{E}, \mathcal{F}, \mathcal{C}).$$

However, the actual problem we wish to solve is

$$\operatorname*{minimize}_{\alpha} \quad \mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C}) \ ,$$

and it is possible that $\alpha^\star \notin \mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C})$. This would be the case if there is $\alpha^{\star,\text{true}} \notin A^{\text{nice}}$ such that

$$\mathcal{R}(M(\alpha^{\star,\text{true}}), \mathcal{E}, \mathcal{F}, \mathcal{C}) < \overline{\mathcal{R}}(M(\alpha^{\star,\text{true}}), \mathcal{E}, \mathcal{F}, \mathcal{C}) \tag{2.47}$$

$$\mathcal{R}(M(\alpha^{\star,\text{true}}),\mathcal{E},\mathcal{F},\mathcal{C}) < \mathcal{R}(M(\alpha^{\star}),\mathcal{E},\mathcal{F},\mathcal{C}) \tag{2.48}$$

$$\overline{\mathcal{R}}(M(\alpha^{\star}),\mathcal{E},\mathcal{F},\mathcal{C}) \leq \overline{\mathcal{R}}(M(\alpha^{\star,\text{true}}),\mathcal{E},\mathcal{F},\mathcal{C}) \tag{2.49}$$

Condition(2.47) states that for $\alpha^{\star,\text{true}}$, the duality gap is positive. Condition (2.48) states that $\alpha^{\star,\text{true}}$ has a better performance than $\alpha^{\star}$. Condition (2.49) reaffirms that $\alpha^{\star}$ is optimal for dual problem. Figure 2.6 illustrates this circumstance. While this pathology is probably extremely unlikely, its possibility is a consequence of the gap in the reasoning.

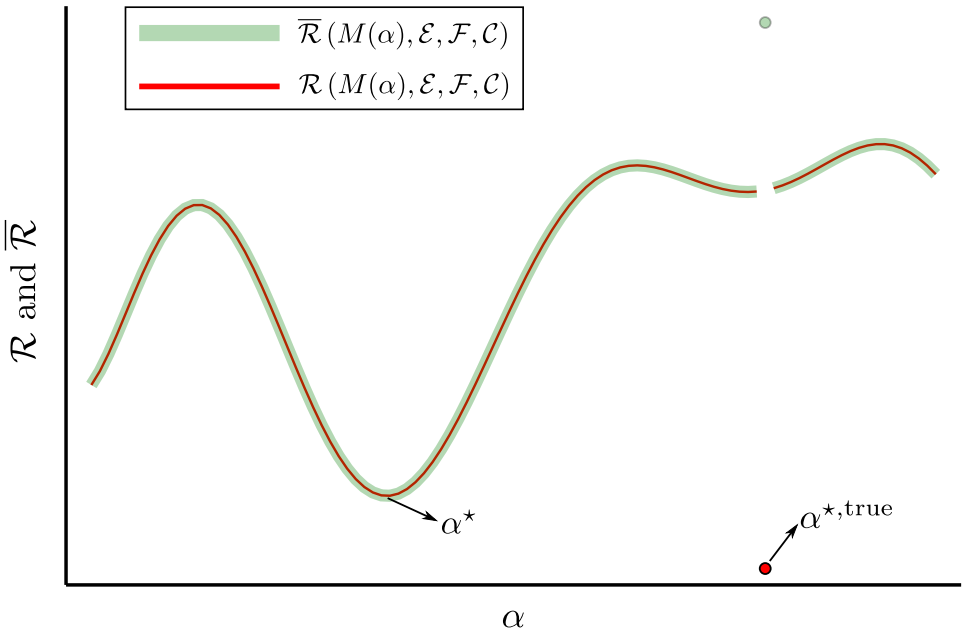


Figure 2.6: Illustration of $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ and $\overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ in a pathological setup. Even if $\alpha^{\star} \in \operatorname{argmin} \overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ satisfies $\alpha^{\star} \in A^{\text{nice}}$, it is possible that $\alpha^{\star} \notin \operatorname{argmin} \mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$.

One can rigorously exclude this pathology in most well-behaved setups using the arguments of Park and Ryu [74, Claim 4]:

(i) Prove $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ is a continuous function of α.

(ii) Observe, a posteriori, that $\alpha^{\star} \in A^{\text{nice}}$.

(One need not show that strong duality holds for $\alpha \notin A^{\text{nice}}$.) Then, we have

$$\begin{aligned}\inf_{\alpha\in\mathbb{R}^{N(N+1)/2}} \mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C}) &\overset{\text{(a)}}{=} \inf_{\alpha\in A^{\text{nice}}} \mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C}) \\ &\overset{\text{(b)}}{=} \inf_{\alpha\in A^{\text{nice}}} \overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C}) \\ &\overset{\text{(c)}}{=} \inf_{\alpha\in\mathbb{R}^{N(N+1)/2}} \overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C}) \\ &\overset{\text{(d)}}{=} \overline{\mathcal{R}}(M(\alpha^\star),\mathcal{E},\mathcal{F},\mathcal{C}).\end{aligned}$$

where (a) follows from continuity of $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ and denseness of $A^{\text{nice}} \subseteq \mathbb{R}^{N(N+1)/2}$, (b) follows from strong duality on A^{nice}, and (c) and (d) follows from the a posteriori observation that $\alpha^\star \in A^{\text{nice}}$. In this case, we would have $\alpha^\star \in \operatorname{argmin} \mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$, even if there is a duality gap, as illustrated in Figure 2.7. Showing that $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ is a continuous function of α can be done by carefully arguing that, for the particular setup, the performance continuously depends on the FSFOM's stepsize.

2.A.3 Exact rank-1 nonconvex semidefinite representation of the BnB-PEP-QCQP

In this section, we derive the exact rank-1 nonconvex semidefinite representation of (2.14). For the other BnB-PEP-QCQPs, the steps to construct such a nonconvex semidefinite representation are identical.

First, we define:

$$w = \operatorname{vec}(\alpha,\nu,\lambda), \tag{2.50}$$

which stacks the elements of α,ν,λ in a column vector w, and denote its number of elements by $|w|$. Then we can define an one-to-one and onto index selector function $\iota(\cdot)$ that takes $\alpha_{i,j}$, ν, or $\lambda_{i,j}$ as an input, and provides the unique index of that element in w with the range

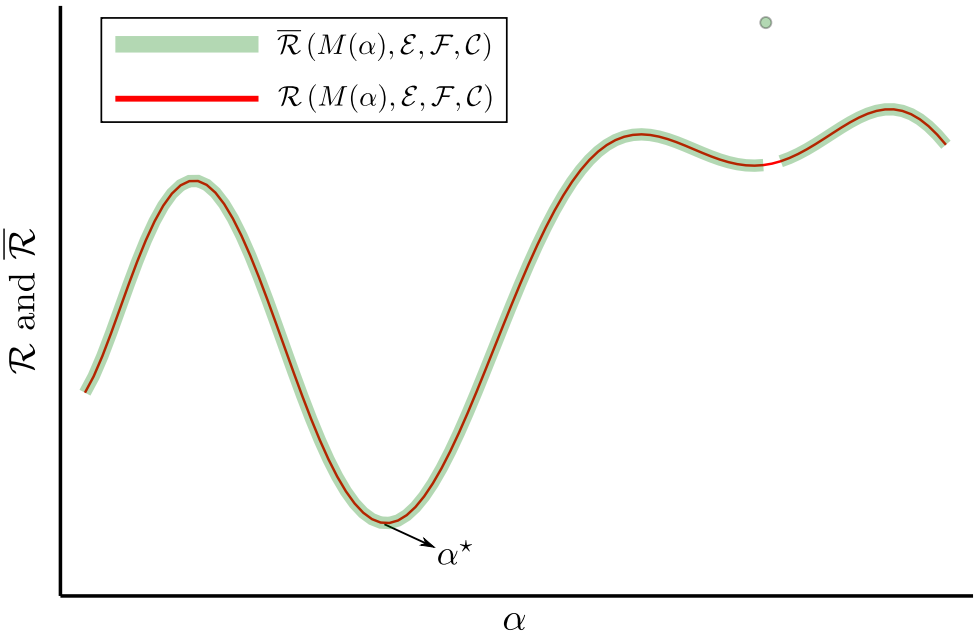


Figure 2.7: Illustration of $\mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C})$ and $\overline{\mathcal{R}}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C})$ when $\mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C})$ is continuous. If $\alpha^\star \in \operatorname{argmin} \overline{\mathcal{R}}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C})$ satisfies $\alpha^\star \in A^{\text{nice}}$, then $\alpha^\star \in \operatorname{argmin} \mathcal{R}(M(\alpha), \mathcal{E}, \mathcal{F}, \mathcal{C})$, even if there is a duality gap.

$\{1, 2, \ldots, |w|\}$ *i.e.*,

$$\alpha_{i,j} = w_{\iota(\alpha_{i,j})}, \nu = w_{\iota(\nu)}, \lambda_{i,j} = w_{\iota(\lambda_{i,j})}.$$

Next, define $W = ww^\top \in \mathbb{S}^{|w|}$. For notational convenience define, $|\mathbf{x}_0| = N + 2$. Defining the map ι mathematically can be quite tedious and does not provide any insight, but it very easy to implement through the `Julia` packages `OrderedCollections` and `JuMP`. Recall that for $i \in [1 : N]$, we have:

$$\mathbf{x}_i = \mathbf{x}_0 \left(1 - (\mu/L) \sum_{j=0}^{i-1} \alpha_{i,j}\right) - (1/L) \sum_{j=0}^{i-1} \alpha_{i,j} \mathbf{g}_j$$

$$= \underbrace{\left[I_{|\mathbf{x}_0|\times|\mathbf{x}_0|} \mid \frac{-1}{L}\left(\sum_{j=0}^{i-1}(\mu\mathbf{x}_0+\mathbf{g}_j)e_{\iota(\alpha_{i,j})}^\top\right)\right]}_{\mathfrak{J}^{[i]}\in\mathbb{R}^{|\mathbf{x}_0|\times(|\mathbf{x}_0|+|w|)}} \begin{bmatrix}\mathbf{x}_0\\ w\end{bmatrix}$$

Also, define $\mathfrak{J}^{[\star]} = \mathbf{0}_{|\mathbf{x}_0|\times(|\mathbf{x}_0|+|w|)}$, and $\mathfrak{J}^{[0]} = [I_{|\mathbf{x}_0|\times|\mathbf{x}_0|} \mid \mathbf{0}_{|\mathbf{x}_0|\times|w|}]$. Then, for all $i \in I_N^\star$, we have:

$$\mathbf{x}_i = \mathfrak{J}^{[i]} \begin{bmatrix}\mathbf{x}_0\\ w\end{bmatrix}.$$

Hence, we have for any $i, j \in I_N^\star$,

$$\begin{aligned}
\mathbf{x}_i - \mathbf{x}_j &= (\mathfrak{J}^{[i]} - \mathfrak{J}^{[j]}) \begin{bmatrix}\mathbf{x}_0\\ w\end{bmatrix}\\
&= \left[\underbrace{\mathfrak{G}^{[i,j]}}_{\in\mathbb{R}^{|\mathbf{x}_0|\times|\mathbf{x}_0|}} \mid \underbrace{\mathfrak{H}^{[i,j]}}_{\in\mathbb{R}^{|\mathbf{x}_0|\times|w|}}\right]\begin{bmatrix}\mathbf{x}_0\\ w\end{bmatrix}\\
&= \mathfrak{G}^{[i,j]}\mathbf{x}_0 + \mathfrak{H}^{[i,j]}w\\
&= \left[\underbrace{\mathfrak{g}^{[i,j][k]\top}\mathbf{x}_0}_{c^{[i,j][k]}} + \mathfrak{h}^{[i,j][k]\top}w\right]_{k=1}^{|\mathbf{x}_0|}\\
&= \left[c^{[i,j][k]} + \mathfrak{h}^{[i,j][k]\top}w\right]_{k=1}^{|\mathbf{x}_0|},
\end{aligned}$$

where $\mathfrak{h}^{[i,j][k]\top}$ and $\mathfrak{g}^{[i,j][k]\top}$ correspond to the k-th rows of $\mathfrak{H}^{[i,j]}$ and $\mathfrak{G}^{[i,j]}$, respectively. Thus, for any $i, j \in I_N^\star$, $k, \ell \in [1 : |\mathbf{x}_0|]$:

$$\begin{aligned}
[B_{i,j}(\alpha)]_{k,\ell} &= [(\mathbf{x}_i - \mathbf{x}_j) \odot (\mathbf{x}_i - \mathbf{x}_j)]_{k,\ell}\\
&= [\mathbf{x}_i - \mathbf{x}_j]_k [\mathbf{x}_i - \mathbf{x}_j]_\ell\\
&= \left[\mathfrak{G}^{[i,j]}\mathbf{x}_0 + \mathfrak{H}^{[i,j]}w\right]_k \left[\mathfrak{G}^{[i,j]}\mathbf{x}_0 + \mathfrak{H}^{[i,j]}w\right]_\ell\\
&= \left[c^{[i,j][k]} + \mathfrak{h}^{[i,j][k]\top}w\right]\left[c^{[i,j][\ell]} + \mathfrak{h}^{[i,j][\ell]\top}w\right]
\end{aligned}$$

$$
\begin{aligned}
&= c^{[i,j][k]}c^{[i,j][\ell]} + c^{[i,j][k]}\mathfrak{h}^{[i,j][\ell]\top}w + c^{[i,j][\ell]}\mathfrak{h}^{[i,j][k]\top}w \\
&\quad + (\mathfrak{h}^{[i,j][k]\top}w)(\mathfrak{h}^{[i,j][\ell]\top}w) \\
&= c^{[i,j][k]}c^{[i,j][\ell]} + c^{[i,j][k]}\mathfrak{h}^{[i,j][\ell]\top}w + c^{[i,j][\ell]}\mathfrak{h}^{[i,j][k]\top}w \\
&\quad + \sum_{\widetilde{i}=1}^{|w|}\sum_{\widetilde{j}=1}^{|w|}\mathfrak{h}_{\widetilde{i}}^{[i,j][k]}\mathfrak{h}_{\widetilde{j}}^{[i,j][\ell]}w_{\widetilde{i}}w_{\widetilde{j}} \\
&= c^{[i,j][k]}c^{[i,j][\ell]} + c^{[i,j][k]}\mathfrak{h}^{[i,j][\ell]\top}w + c^{[i,j][\ell]}\mathfrak{h}^{[i,j][k]\top}w \\
&\quad + w^\top H^{[i,j][k,\ell]}w \\
&= c^{[i,j][k]}c^{[i,j][\ell]} + c^{[i,j][k]}\mathfrak{h}^{[i,j][\ell]\top}w + c^{[i,j][\ell]}\mathfrak{h}^{[i,j][k]\top}w \\
&\quad + \mathbf{tr}\left(H^{[i,j][k,\ell]}W\right), \qquad (2.51)
\end{aligned}
$$

where $H^{[i,j][k,\ell]} \in \mathbb{S}^{|w|}$ with its entries defined by

$$
H_{\widetilde{i},\widetilde{j}}^{[i,j][k,\ell]} = \frac{1}{2}\left(\mathfrak{h}_{\widetilde{i}}^{[i,j][k]}\mathfrak{h}_{\widetilde{j}}^{[i,j][\ell]} + \mathfrak{h}_{\widetilde{j}}^{[i,j][k]}\mathfrak{h}_{\widetilde{i}}^{[i,j][\ell]}\right),
$$

for $\widetilde{i},\widetilde{j} \in [1:|w|]$. Also, for $i,j \in I_N^\star$, and $k,\ell \in [1:|\mathbf{x}_0|]$:

$$
\begin{aligned}
[A_{i,j}(\alpha)]_{k,\ell} &= [\mathbf{g}_j \odot (\mathbf{x}_i - \mathbf{x}_j)]_{k,\ell} \\
&= \left[\frac{1}{2}\mathbf{g}_j(\mathbf{x}_i - \mathbf{x}_j)^\top + \frac{1}{2}(\mathbf{x}_i - \mathbf{x}_j)\mathbf{g}_j^\top\right]_{k,\ell} \\
&= \frac{1}{2}\left[\mathbf{g}_j\right]_k\left[\mathbf{x}_i - \mathbf{x}_j\right]_\ell + \frac{1}{2}\left[\mathbf{x}_i - \mathbf{x}_j\right]_k\left[\mathbf{g}_j\right]_\ell \\
&= \frac{1}{2}\left[\mathbf{g}_j\right]_k(c^{[i,j][\ell]} + \mathfrak{h}^{[i,j][\ell]\top}w) + \frac{1}{2}(c^{[i,j][k]} + \mathfrak{h}^{[i,j][k]\top}w)\left[\mathbf{g}_j\right]_\ell \\
&= \frac{1}{2}\left[c^{[i,j][\ell]}\left[\mathbf{g}_j\right]_k + c^{[i,j][k]}\left[\mathbf{g}_j\right]_\ell\right] + \\
&\frac{1}{2}\left[\mathbf{g}_j\right]_k(\mathfrak{h}^{[i,j][\ell]\top}w) + \frac{1}{2}(\mathfrak{h}^{[i,j][k]\top}w)\left[\mathbf{g}_j\right]_\ell \\
&= \overbrace{\frac{1}{2}\left[c^{[i,j][\ell]}\left[\mathbf{g}_j\right]_k + c^{[i,j][k]}\left[\mathbf{g}_j\right]_\ell\right]}^{=\widetilde{c}^{[i,j][k,\ell]}} +
\end{aligned}
$$

$$
\begin{aligned}
&\sum_{\widetilde{i}=1}^{|w|}\left(\frac{1}{2}\left[\mathbf{g}_j\right]_k \mathfrak{h}_{\widetilde{i}}^{[i,j][\ell]}+\frac{1}{2}\left[\mathbf{g}_j\right]_\ell \mathfrak{h}_{\widetilde{i}}^{[i,j][k]}\right) w_{\widetilde{i}} \\
&=\widetilde{c}^{[i,j][k,\ell]}+\sum_{\widetilde{i}=1}^{|w|}\underbrace{\left(\frac{1}{2}\left[\mathbf{g}_j\right]_k \mathfrak{h}_{\widetilde{i}}^{[i,j][\ell]}+\frac{1}{2}\left[\mathbf{g}_j\right]_\ell \mathfrak{h}_{\widetilde{i}}^{[i,j][k]}\right)}_{=\widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]}} w_{\widetilde{i}} \\
&=\widetilde{c}^{[i,j][k,\ell]}+\sum_{\widetilde{i}=1}^{|w|}\widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]} w_{\widetilde{i}}.
\end{aligned}
$$

Next, denoting $e_{\iota(\lambda_{i,j})}=\widetilde{d}^{[i,j]}$, we have for $k,\ell\in[1:|\mathbf{x}_0|]$

$$
\begin{aligned}
&\lambda_{i,j}\left[A_{i,j}(\alpha)\right]_{k,\ell} \\
&=(\widetilde{d}^{[i,j]\top}w)\left(\widetilde{c}^{[i,j][k,\ell]}+\sum_{\widetilde{i}=1}^{|w|}\widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]}w_{\widetilde{i}}\right) \\
&=\widetilde{c}^{[i,j][k,\ell]}(\widetilde{d}^{[i,j]\top}w)+\left(\sum_{\widetilde{j}=1}^{|w|}\widetilde{d}_{\widetilde{j}}^{[i,j]}w_{\widetilde{j}}\right)\left(\sum_{\widetilde{i}=1}^{|w|}\widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]}w_{\widetilde{i}}\right) \\
&=\widetilde{c}^{[i,j][k,\ell]}(\widetilde{d}^{[i,j]\top}w)+\sum_{\widetilde{i}=1}^{|w|}\sum_{\widetilde{j}=1}^{|w|}\left[\widetilde{d}_{\widetilde{j}}^{[i,j]}\widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]}\right]w_{\widetilde{i}}w_{\widetilde{j}} \\
&=\widetilde{c}^{[i,j][k,\ell]}(\widetilde{d}^{[i,j]\top}w)+w^\top S^{[i,j][k,\ell]}w \\
&=\widetilde{c}^{[i,j][k,\ell]}(\widetilde{d}^{[i,j]\top}w)+\mathbf{tr}\left(S^{[i,j][k,\ell]}W\right),
\end{aligned}
\tag{2.52}
$$

where: $S^{[i,j][k,\ell]}\in\mathbb{S}^{|w|}$ with its entries defined by

$$
S^{[i,j][k,\ell]}[\widetilde{i},\widetilde{j}]=\frac{1}{2}\left(\left[\widetilde{d}_{\widetilde{j}}^{[i,j]}\widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]}\right]+\left[\widetilde{d}_{\widetilde{i}}^{[i,j]}\widetilde{q}_{\widetilde{j}}^{[i,j][k,\ell]}\right]\right),
$$

for $\widetilde{i},\widetilde{j}\in[1:|w|]$. Hence, we have

$$
\sum_{i,j\in I_N^\star:i\neq j}\lambda_{i,j}\left[A_{i,j}(\alpha)\right]_{k,\ell}=\left(\sum_{i,j\in I_N^\star:i\neq j}\widetilde{c}^{[i,j][k,\ell]}\widetilde{d}^{[i,j]\top}\right)w+\mathbf{tr}\left(\sum_{i,j\in I_N^\star}S^{[i,j][k,\ell]}\right)W.
$$

Using (2.51) and (2.52), we have the following nonconvex semidefinite representation of (2.14):

$$
\begin{aligned}
&\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{(i,j)\in I_N^\star} \lambda_{i,j} a_{i,j} = 0, \\
\nu \left[B_{0,\star}\right]_{k,\ell} - \left[C_{N,\star}\right]_{k,\ell} - \mu^2 \left[B_{N,\star}(\alpha)\right]_{k,\ell} + \\
\qquad 2\mu \left[A_{\star,N}(\alpha)\right]_{k,\ell} + \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} + \\
\qquad \frac{1}{2(L-\mu)} \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[C_{i,j}\right]_{k,\ell} = Z_{k,\ell}, \quad k \in [1 : |\mathbf{x}_0|],\ \ell \in [1:k], \\
\left[B_{N,\star}(\alpha)\right]_{k,\ell} = c^{[N,\star][k]} c^{[N,\star][\ell]} + c^{[N,\star][k]} \mathfrak{h}^{[N,\star][\ell]\top} w + c^{[N,\star][\ell]} \mathfrak{h}^{[N,\star][k]\top} w + \\
\qquad \mathbf{tr}\left(H^{[N,\star][k,\ell]} W\right), \quad k \in [1 : |\mathbf{x}_0|],\ \ell \in [1:k], \\
\sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} = \left(\sum_{i,j\in I_N^\star : i\neq j} \widetilde{c}^{[i,j][k,\ell]} \widetilde{d}^{[i,j]\top}\right) w + \\
\qquad \mathbf{tr}\left(\sum_{i,j\in I_N^\star} S^{[i,j][k,\ell]}\right) W, \quad k \in [1 : |\mathbf{x}_0|],\ \ell \in [1:k], \\
Z \succeq 0, \\
W = w w^\top, \\
(\forall i, j \in I_N^\star) \quad \lambda_{i,j} \geq 0,\ \nu \geq 0,
\end{array} \right)
\end{aligned}
\tag{2.53}
$$

where λ, ν, Z, and W are the decision variables, and $w = \mathrm{vec}(\alpha, \nu, \lambda)$ as defined in (2.50). The constraint $W = ww^\top$ is nonconvex, but if we replace this constraint with the implied constraint $W \succeq ww^\top$, then by using Schur complement, a convex semidefinite relaxation of (2.14) is given by:

$$
\left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j\in I_N^\star} \lambda_{i,j} a_{i,j} = 0, \\
\nu \left[B_{0,\star}\right]_{k,\ell} - \left[C_{N,\star}\right]_{k,\ell} - \mu^2 \left[B_{N,\star}(\alpha)\right]_{k,\ell} + \\
\qquad 2\mu \left[A_{\star,N}(\alpha)\right]_{k,\ell} + \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} + \\
\qquad \frac{1}{2(L-\mu)} \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[C_{i,j}\right]_{k,\ell} = Z_{k,\ell}, \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
\left[B_{N,\star}(\alpha)\right]_{k,\ell} = c^{[N,\star][k]} c^{[N,\star][\ell]} + c^{[N,\star][k]} \mathfrak{h}^{[N,\star][\ell]\top} w + c^{[N,\star][\ell]} \mathfrak{h}^{[N,\star][k]\top} w + \\
\qquad \mathbf{tr}\left(H^{[N,\star][k,\ell]} W\right), \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
\sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} = \left(\sum_{i,j\in I_N^\star : i\neq j} \widetilde{c}^{[i,j][k,\ell]} \widetilde{d}^{[i,j]\top}\right) w + \\
\qquad \mathbf{tr}\left(\sum_{i,j\in I_N^\star} S^{[i,j][k,\ell]}\right) W \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
Z \succeq 0, \\
\begin{bmatrix} W & w \\ w^\top & 1 \end{bmatrix} \succeq 0, \\
(\forall i,j \in I_N^\star) \quad \lambda_{i,j} \geq 0,\ \nu \geq 0,
\end{array}\right) \tag{2.54}
$$

where λ, ν, Z, and W are the decision variables. The optimal objective value of (2.54) will provide a lower bound to (2.53).

Chapter 3

Nonlinear conjugate gradient methods: worst-case convergence rates via computer-assisted analyses

3.1 Introduction

We consider the standard unconstrained convex minimization problem

$$f_\star \triangleq \min_{x\in\mathbb{R}^n} f(x), \tag{3.1}$$

where f is L-smooth (i.e., it has an L-Lipschitz gradient) and μ-strongly convex. We study the worst-case performances of a few famous variants of *nonlinear conjugate gradient methods* (NCGMs) for solving (3.1). More specifically, we study Polak-Ribière-Polyak (PRP) [11, 12] and Fletcher-Reeves (FR) [13] schemes with exact line search. With exact line search, many other NCGMs such as the Hestenes and Stiefel method [14], the conjugate descent method

due to Fletcher [15], and the Dai and Yuan method [16] reduce to either PRP or FR. Under exact line search, PRP and FR can be presented in the following compact form:

$$
\begin{aligned}
&\gamma_k \in \operatorname*{argmin}_{\gamma \in \mathbb{R}} f(x_k - \gamma\, d_k),\\
&x_{k+1} = x_k - \gamma_k d_k,\\
&\beta_k = \frac{\|\nabla f(x_{k+1})\|^2 - \eta\, \langle \nabla f(x_{k+1}) \mid \nabla f(x_k) \rangle}{\|\nabla f(x_k)\|^2},\\
&d_{k+1} = \nabla f(x_{k+1}) + \beta_k d_k,
\end{aligned}
\tag{$\mathcal{M}$}
$$

where PRP and FR are respectively obtained by setting $\eta = 1$ and $\eta = 0$. NCGMs have a long history (see, e.g., the survey [110] and monograph [111]), but are much less studied compared to their many first-order competitors. For instance, even though FR is generally considered the first NCGM [110, §1], we are not aware of non-asymptotic convergence results for it. On a similar note, some variants of NCGMs are known for their generally good empirical behaviors (which we illustrate in Figure 3.1) with little of them being backed-up by classical complexity analyses. In this work, we apply the performance estimation approach [4, 5] to ($\mathcal{M}$) for filling this gap by explicitly computing some worst-case convergence properties of PRP and FR with exact line search. This work focuses on exact line search, as it is arguably the most logical starting point to understand the non-asymptotic convergence behavior of NCGMs. In certain cases, the minimizer associated with exact line search has an analytical form, while in others, it can be computed efficiently [58, §9.7.1]. However, in many practical implementations, inexact line searches are employed that try to either approximately minimize $f(x_k - \gamma\, d_k)$ or even just reduce f enough along the ray $x_k - \gamma d_k$. These inexact methods can be either monotone, which ensures a decrease in f but converges slowly, or nonmonotone, which may allow faster convergence but risks nonrobust tuning [111, §1.2]. Examples of notable monotone inexact line search schemes include backtracking [112], Goldstein [113], and Wolfe line searches [114, 115], and their variants [116]. Nonmonotone schemes include [117–119] and many others;

see [111, pp. 10-14] for a brief review. Despite the computational differences between the two types of line searches, both aim to emulate the exact line search method. Consequently, when using an inexact line search process, any convergence guarantees—defined in terms of iteration numbers—are likely to be worse compared to the exact line search (neglecting the cost of performing exact line search).

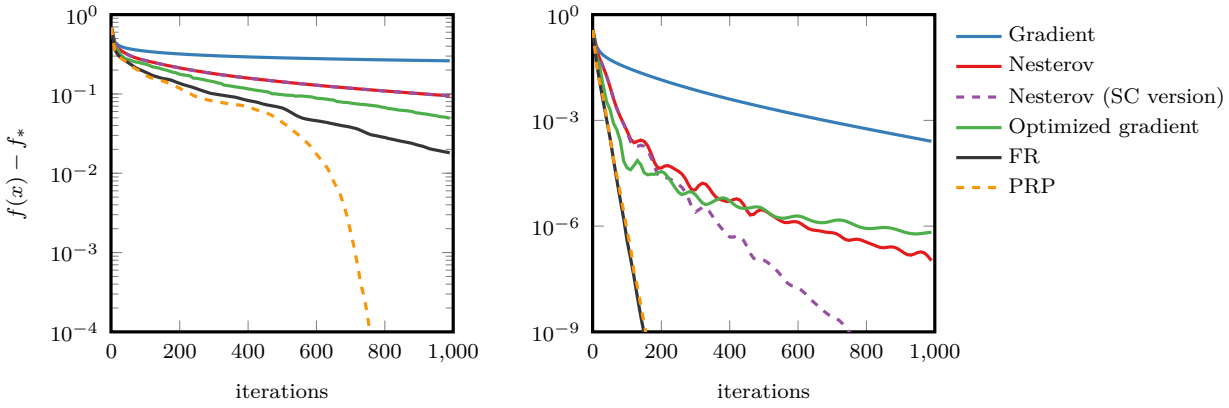


Figure 3.1: Convergence of a few first-order methods on a logistic regression problem on the small-sized Sonar dataset [120]. Experiments with normalized features (zero mean and unit variance). Left: without regularization. Right: with an ℓ_2 regularization of parameter 10^{-4}. All methods were featured with an *exact* line search (performed numerically using the bisection method with a tolerance of 10^{-8}): (i) gradient descent, (ii) Nesterov's accelerated gradient [121] (exact line search instead of backtracking), (iii) Nesterov's accelerated method for strongly convex problems, version [122, Algorithm 28] with exact line search instead of the gradient step, (iv) optimized gradient method [40, Algorithm (OGM-LS)], (v) FR, and (vi) PRP.

3.1.1 Contributions

The contribution of this chapter is twofold. First, we compute worst-case convergence bounds and counter-examples for PRP and FR. These bounds are obtained by formulating the problems of computing worst-case scenarios as nonconvex quadratically constrained quadratic optimization problems (QCQPs), and then by solving them to global optimality. Second, these computations enable us to construct mathematical proofs that establish an improved non-asymptotic convergence bound for PRP, and, to the best of our knowledge, the first

non-asymptotic convergence bound for FR. Furthermore, the worst-case bounds for PRP and FR obtained numerically reveal that there are simple adversarial examples on which these methods do not perform better than gradient descent with exact line search (GDEL), leaving very little room for improvements on this class of problems. Since we demonstrate that the convergence results of NCGMs associated with exact line search are already disappointing, we conclude that inexact line searches, which approximate exact line search, are unlikely to offer improvement.

From a methodological point of view, our approach to computing worst-case scenarios and bounds through optimization is part of what is often referred to as *performance estimation* framework. This framework models the computation of the worst-case performance of a first-order method as an optimization problem itself; such optimization problems are called performance estimation problems (PEP). While these PEPs are usually amenable to convex semidefinite programs [4–6], this is generally not the case for *adaptive* first-order methods such as PRP and FR [123, 124]. To study these methods, we evaluate the worst-case performances of ($\mathcal{M}$) by solving nonconvex QCQPs, extending the standard SDP-based approach from [4–6] developed for non-adaptive methods. This contribution aligns with the spirit of Chapter 2 of this thesis (the content of that chapter was published in Mathematical Programming in 2023 [1]), developed for devising optimal (but non-adaptive) first-order methods.

Organization. The chapter is organized as follows. In Section 3.2, we establish non-asymptotic convergence rates for PRP and FR by viewing the search direction d_k in ($\mathcal{M}$) as an approximate gradient direction. In Section 3.3, we compute the exact numerical values of the worst-case $f(x_N)-f_\star/f(x_0)-f_\star$ and $f(x_{k+N})-f_\star/f(x_k)-f_\star$ for PRP and FR by formulating the problems as nonconvex QCQPs and then solving them to certifiable global optimality using a custom spatial branch-and-bound algorithm. The solutions to these QCQPs allow us to

construct low-dimensional (dimension 4) counter-examples indicating that there is essentially no room for further improvement of the rates that we provide. In Section 3.4, we discuss the implementation details for solving the nonconvex QCQPs in this chapter.

Code. All the numerical results in this chapter were obtained on the `MIT Supercloud Computing Cluster` with Intel-Xeon-Platinum-8260 processor with 48 cores and 128 GB of RAM running Ubuntu 18.04.6 LTS with Linux 4.14.250-llgrid-10ms kernel [80]. We used `JuMP`—a domain specific modeling language for mathematical optimization embedded in the open-source programming language `Julia` [53]—to model the optimization problems. To solve the optimization problems, we use the following solvers: `Mosek 9.3` [54], `KNITRO 13.0.0` [56], and `Gurobi 10.0.0` [125], which are free for academic use. The relative feasibility tolerance and relative optimality tolerance of all the solvers are set at 1e-6. We validated the "bad" worst-case scenarios produced by our methodology using the `PEPit` package [126], which is an open-source `Python` library allowing to use the performance estimation problem (PEP) framework. A PEP is an optimization problem to compute the worst-case performance of a first-order method.

The code used to generate and validate the results in this chapter is available at:

https://github.com/Shuvomoy/NCG-PEP-code.

3.1.2 Related works

Conjugate gradient (CG) methods are particularly popular choices for solving systems of linear equations and quadratic minimization problems; in this context, they are known to be information-optimal in the class of first-order methods [32, Chapter 12 & Chapter 13] or [127, Chapter 5]. There are many extensions beyond quadratics, commonly referred to as *nonlinear conjugate gradient methods* (NCGMs). They are discussed at length in the

textbooks [128, Chapter 5 & Chapter 7] and [129, Chapter 5] and in the nice survey [110]. In particular, when exact line searches are used, many variants become equivalent and can be seen as instances of quasi-Newton methods, see [128, Chapter 7, §"Relationship with conjugate gradient methods"] or [129, Chapter 5, §5.5]. For instance, it is well known that standard variants such as Hestenes-Stiefel [14] and Dai-Yuan [16] are equivalent to ($\mathcal{M}$) when exact line searches are used, while being different in the presence of more popular line search procedures (such as Wolfe's [128, Chapter 3]). Beyond quadratics, obtaining convergence guarantees is often reduced to the problem of ensuring the search direction to be a descent direction, see for instance [127, §5.5 "Extensions to non-quadratic problems"] or [116, 130]. Without exact line searches, even when f is strongly convex, there are counter-examples showing that even popular variants may not generate descent directions [131]. Note that NCGMs are often used together with restart strategies, which we do not consider here; see, e.g., [132] and the references therein. Also, in [133, §5], the authors empirically demonstrate that NCGMs work very well in training deep learning problems.

In this work, we use the performance estimation framework, which models the computation of the worst-case performance of a first-order method as an optimization problem called PEP [4–6]. This PEP methodology is essentially mature for analyzing "fixed-step" (i.e., non-adaptive) first-order methods (and for methods whose analyses are amenable to those of fixed-step methods), whose stepsizes are essentially chosen in advance. This type of methods include many common first-order methods and operator splitting schemes, including the heavy-ball method [134] and Nesterov's accelerated gradient [101, 121]. Only very few adaptive methods were studied using the PEP methodology, namely gradient descent with exact line searches [47], greedy first-order methods [40], and Polyak stepsizes [123]. A premise to the study of NCGMs using PEPs was done in [124, §4.5.2], where an upper bound on the worst-case $(f(x_2)-f_\star)/(f(x_0)-f_\star)$ of FR was numerically computed for two iterations and

two condition number values, $q = 0.1$ and $q = 0.01$, where $q \triangleq \mu/L$. This was achieved by numerically solving an SDP relaxation through a grid search on β_k. In comparison, we compute the worst-case $(f(x_N)-f_\star)/(f(x_0)-f_\star)$ by solving the nonconvex PEPs associated with both FR and PRP to global optimality across a broader range of condition numbers over $q \in [0, 1]$ for $N = 1, 2, 3, 4$. Furthermore, for both methods, we also compute "Lyapunov"-type bounds on $(f(x_{k+N})-f_\star)/(f(x_k)-f_\star)$ that holds for any k for $N = 1, 2, 3, 4$, and also establish their analytical complexity bounds offering a more comprehensive understanding of their performance. We run our numerical experiments for $N = 1, 2, 3, 4$, as the qualitative behaviors of both PRP and FR become clear after 3-4 iterations. Our work is also closely related in spirit with the technique developed in [1] for optimizing coefficients of fixed-step first-order methods using nonconvex optimization.

3.1.3 Preliminaries

In this section, we recall the definition and a result on smooth strongly convex functions, as well as a base result on steepest descent with an exact line search.

Properties of smooth strongly convex functions. We use the standard notation $\langle \cdot \mid \cdot \rangle : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ to denote the Euclidean inner product, and the corresponding induced Euclidean norm $\|\cdot\|$. The class of L-smooth μ-strongly convex functions is standard and can be defined as follows.

Definition 3.1. Let $f : \mathbb{R}^n \to \mathbb{R}$ be a proper, closed, and convex function, and consider two constants $0 \leq \mu < L < \infty$. The function f is L-smooth and μ-strongly convex (notation $f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n)$), if

- (L-smooth) f is differentiable and for all $x, y \in \mathbb{R}^n$, it holds that $\|\nabla f(x) - \nabla f(y)\| \leq$

$L\|x-y\|$,

- (μ-strongly convex) for all $x, y \in \mathbb{R}^n$, it holds that $f(x) \geq f(y) + \langle \nabla f(y) \mid x - y \rangle + \frac{\mu}{2}\|x-y\|^2$.

We simply denote $f \in \mathcal{F}_{\mu,L}$ when the dimension is either clear from the context or unspecified. We also denote by $q \triangleq \frac{\mu}{L}$ the inverse condition number. For readability, we do not explicitly treat the (trivial) case $L = \mu$.

Smooth strongly convex functions satisfy many inequalities, see e.g., [135, Theorem 2.1.5]. For the developments below, we need only one specific inequality characterizing functions in $\mathcal{F}_{\mu,L}$. The following result can be found in [5, Theorem 4] and is key in our analysis.

Theorem 3.1. *[5, Theorem 4, $\mathcal{F}_{\mu,L}$-interpolation] Let I be an index set and $S = \{(x_i, g_i, f_i)\}_{i\in I} \subseteq \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}$ be a set of triplets. There exists $f \in \mathcal{F}_{\mu,L}$ satisfying $f(x_i) = f_i$ and $\nabla f(x_i) = g_i$ for all $i \in I$ if and only if*

$$f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2(1-\frac{\mu}{L})}\left(\frac{1}{L}\|g_i - g_j\|^2 + \mu\|x_i - x_j\|^2 - 2\frac{\mu}{L}\langle g_i - g_j \mid x_i - x_j\rangle\right) \quad (3.2)$$

holds for all $i, j \in I$.

Another related result from [41, §2.1] that we record next involves constructing a strongly-convex smooth function from a given set of triplets. In this theorem, number of elements of index set I is denoted by $|I|$.

Theorem 3.2. *[41, §2.1, strongly convex and smooth extension] Suppose I is a set of indices and $S = \{(x_i, g_i, f_i)\}_{i\in I} \subseteq \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}$ is a set of triplets such that (3.2) holds for all $i, j \in I$*

for some $0 \le \mu < L < \infty$. *Then the function* $f : \mathbb{R}^n \to \mathbb{R}$ *defined by*

$$f(y) = \max_{\alpha \in \Delta} \left[\frac{L}{2}\|y\|^2 - \frac{L-\mu}{2}\|y - \frac{1}{L-\mu}\sum_{i\in I}\alpha_i(g_i - \mu x_i)\|^2 + \sum_{i\in I}\alpha_i\Big(f_i + \frac{1}{2(L-\mu)}\|g_i - Lx_i\|^2 - \frac{L}{2}\|x_i\|^2\Big)\right] \tag{3.3}$$

where $\Delta = \{\alpha \in \mathbb{R}^{|I|} \mid \alpha \ge 0, \sum_{i=1}^n \alpha_i = 1\}$, *satisfies* $f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n)$, $f(x_i) = f_i$ *and* $\nabla f(x_i) = g_i$ *for all* $i \in I$.

Approximate steepest descent method. Consider a function $f \in \mathcal{F}_{\mu,L}$ and the approximate steepest descent method:

$$\begin{aligned} \gamma_k &= \operatorname*{argmin}_{\gamma \in \mathbb{R}} f(x_k - \gamma d_k), \\ x_{k+1} &= x_k - \gamma_k d_k, \end{aligned} \tag{$\mathcal{ASD}$}$$

where the search direction d_k satisfies a relative error criterion:

$$\|d_k - \nabla f(x_k)\| \le \epsilon \|\nabla f(x_k)\| \text{ where } \epsilon \in [0, 1). \tag{$\mathcal{REC}$}$$

Note that the relative tolerance ϵ needs to satisfy $\epsilon \in [0, 1)$ for ($\mathcal{ASD}$) to converge. If $\epsilon \ge 1$, then $d_k = 0$ becomes feasible and ($\mathcal{ASD}$) cannot be guaranteed to converge anymore, because in such a case we can select d_k to be orthogonal to $\nabla f(x_k)$ in practice [47, §5].

The iterates of ($\mathcal{ASD}$) satisfies the following two necessary (weaker) conditions for x_{k+1} to follow ($\mathcal{ASD}$):

$$\begin{aligned} \langle \nabla f(x_{k+1}) \mid d_k \rangle &= 0, \\ \langle \nabla f(x_{k+1}) \mid x_k - x_{k+1} \rangle &= 0, \end{aligned} \tag{$\mathcal{ASD}_{\text{relaxed}}$}$$

where the first condition follows from optimality of γ_k in the line search condition in ($\mathcal{ASD}$) as follows

$$\begin{aligned} 0 &= \left[\nabla_\gamma f(x_k - \gamma d_k)\right]_{\gamma=\gamma_k} \\ &= -\left\langle \nabla f(x_k - \gamma_k d_k) \mid d_k \right\rangle \\ &= -\left\langle \nabla f(x_{k+1}) \mid d_k \right\rangle, \end{aligned} \tag{3.4}$$

and the second condition comes from putting $d_k = {}^{(x_k - x_{k+1})}/_{\gamma_k}$ in (3.4).

Convergence of approximate steepest descent method. We will use the following result in our analysis. Note that similar results (without line searches) to that of Theorem 3.3 can be found in [44], which might help in future analyses of NCGMs without exact line searches.

Theorem 3.3 ([47, Theorem 5.1])**.** *Let $f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n)$, $x_\star \triangleq \operatorname{argmin}_{x\in\mathbb{R}^n} f(x)$ be the minimizer of f, and $f_\star \triangleq f(x_\star)$. For any $x_k \in \mathbb{R}^n$, search direction $d_k \in \mathbb{R}^n$, and $x_{k+1} \in \mathbb{R}^n$ computed using ($\mathcal{ASD}$) such that the relative error criterion ($\mathcal{REC}$) holds, we have*

$$f(x_{k+1}) - f_\star \leq \left(\frac{1-q_\epsilon}{1+q_\epsilon}\right)^2 \left(f(x_k) - f_\star\right), \tag{3.5}$$

where $q_\epsilon \triangleq {}^{\mu(1-\epsilon)}/_{L(1+\epsilon)}$.

Next, we show that the relative error criterion ($\mathcal{REC}$) can be interpreted in simple geometric fashion in the context of exact line searches in ($\mathcal{ASD}$). As ($\mathcal{ASD}$) uses exact line search, it is the direction of d_k that influences the convergence, not the magnitude. Scaling the magnitude of d_k by a suitable nonzero factor $\alpha = {}^{\langle \nabla f(x_k) | d_k \rangle}/_{\|d_k\|^2}$, i.e., the scaled direction being αd_k, leads to γ_k getting scaled to γ_k/α, but this scaling does not alter x_k, x_{k+1} and we have $|\sin\angle(\nabla f(x_k), \alpha d_k)| = |\sin\angle(\nabla f(x_k), d_k)|$ (details in the proof to Corollary 3.1 below);

here we have used the notation $\angle(a, b)$ to denote the angle between two vectors a, b. Hence, by appropriate scaling of search direction to αd_k, we can ensure that $|\sin\angle(\nabla f(x_k), \alpha d_k)| = {}^{\|\alpha d_k - \nabla f(x_k)\|}/_{\|\nabla f(x_k)\|}$. Since the iterates remain the same, under the angle condition, we will have the same convergence guarantees that hold for ($\mathcal{REC}$) in Theorem 3.3. In short, ($\mathcal{REC}$) is equivalent to $|\sin\theta_k| \leq \epsilon$ in the context of exact line search used in ($\mathcal{ASD}$), which we detail in Corollary 3.1 below.

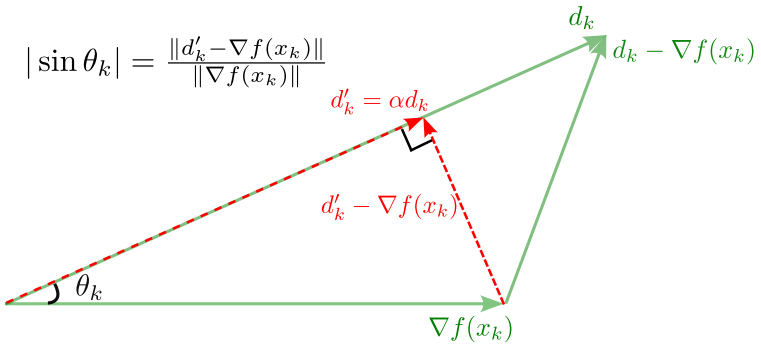


Figure 3.2: This figure illustrates how for any $x_k \in \mathbb{R}^n$, search direction $d_k \in \mathbb{R}^n$, and $x_{k+1} \in \mathbb{R}^n$ satisfying ($\mathcal{ASD}$), one can scale d_k appropriately without altering x_k, x_{k+1} such that the scaled search direction $d'_k = \alpha d_k$ with $\alpha = {}^{\langle \nabla f(x_k) | d_k \rangle}/_{\|d_k\|^2}$ ensures $|\sin\theta_k| = {}^{\|d'_k - \nabla f(x_k)\|}/_{\|\nabla f(x_k)\|}$ with θ_k being the angle between $\nabla f(x_k)$ and d_k.

Corollary 3.1. *Let $f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n)$, $x_\star \triangleq \operatorname{argmin}_{x\in\mathbb{R}^n} f(x)$ be the minimizer of f, and $f_\star \triangleq f(x_\star)$. Consider any $x_k \in \mathbb{R}^n$, search direction $d_k \in \mathbb{R}^n$, and $x_{k+1} \in \mathbb{R}^n$ computed using ($\mathcal{ASD}$) such that $|\sin\theta_k| \leq \epsilon$ with θ_k being the angle between $\nabla f(x_k)$ and d_k and $\epsilon \in [0, 1)$. Then we have*

$$f(x_{k+1}) - f_\star \leq \left(\frac{1 - q_\epsilon}{1 + q_\epsilon}\right)^2 (f(x_k) - f_\star)\,, \tag{3.6}$$

where $q_\epsilon \triangleq {}^{\mu(1-\epsilon)}/_{L(1+\epsilon)}$.

Proof. The proof sketch is as follows. As ($\mathcal{ASD}$) uses exact line search, it is only the direction of d_k that influences the convergence, not its magnitude. Hence, we can appropriately scale the

search direction d_k to $d'_k = \alpha d_k$ (with $\alpha = {\langle \nabla f(x_k) | d_k \rangle}/{\|d_k\|^2} \neq 0$) so that the algorithm iterates remain the same and we can ensure $|\sin\theta_k| = |\sin\angle(\nabla f(x_k), d'_k)| = {\|d'_k - \nabla f(x_k)\|}/{\|\nabla f(x_k)\|}$. Since the iterates remain the same, under the angle condition $|\sin\theta_k| \leq \epsilon$ we have the same convergence guarantees that hold for ($\mathcal{REC}$) in Theorem 3.3.

Now we start the proof earnestly. Consider the following method, where the search direction d_k in ($\mathcal{ASD}$) is scaled by some factor $\alpha \neq 0$ with the scaled search direction denoted by $d'_k = \alpha d_k$:

$$\begin{aligned} \gamma'_k &= \operatorname*{argmin}_{\gamma'} f(x_k - \gamma' d'_k), \\ x'_{k+1} &= x_k - \gamma'_k d'_k, \end{aligned} \tag{$\mathcal{ASD}_{\text{scaled}}$}$$

and we denote θ'_k to be the angle between $\nabla f(x_k)$ and d'_k. We now show that ($\mathcal{ASD}$) and ($\mathcal{ASD}_{\text{scaled}}$) are *equivalent in the sense that they generate an identical sequence of iterates* x_k, x_{k+1} *along with* $|\sin\theta_k| = |\sin\theta'_k|$. This is so because

$$\gamma'_k = \operatorname*{argmin}_{\gamma'} f(x_k - \gamma' d'_k) = \operatorname*{argmin}_{\gamma} f\left(x_k - \frac{\gamma_k}{\alpha} \cdot \alpha d_k\right) = \frac{\gamma_k}{\alpha},$$

i.e., the optimal stepsize γ'_k in ($\mathcal{ASD}_{\text{scaled}}$) is the optimal step-size γ_k in $\mathcal{ASD}$ scaled by ${1}/{\alpha}$, leading to

$$x'_{k+1} = x_k - \gamma'_k d'_k = x_k - \gamma_k d_k = x_{k+1}.$$

Finally,

$$|\sin\theta'_k| = \sqrt{1 - \cos^2\theta'_k}$$

$$
\begin{aligned}
&= \sqrt{1 - \frac{\langle \nabla f(x_k) \mid d'_k \rangle^2}{\|\nabla f(x_k)\|^2 \|d'_k\|^2}} \\
&= \sqrt{1 - \frac{\cancel{\alpha}^2 \langle \nabla f(x_k) \mid d_k \rangle^2}{\cancel{\alpha}^2 \|\nabla f(x_k)\|^2 \|d_k\|^2}} \\
&= \sqrt{1 - \cos^2 \theta_k} \\
&= |\sin \theta_k|.
\end{aligned}
$$

Hence to establish our convergence result (3.6), we can work with ($\mathcal{ASD}_{\text{scaled}}$). Next, we carefully select a nonzero α that ensures $\langle d'_k \mid d'_k - \nabla f(x_k) \rangle = 0$, i.e., $d'_k - \nabla f(x_k)$ would be perpendicular to d'_k (see Figure 3.2); this yields $\alpha = {}^{\langle \nabla f(x_k) | d_k \rangle}/_{\|d_k\|^2}$ which is nonzero because $\epsilon \in [0, 1)$ implies $\langle \nabla f(x_k) \mid d_k \rangle \neq 0$. For this value of α, we have $|\sin \theta_k| = {}^{\|d'_k - \nabla f(x_k)\|}/_{\|\nabla f(x_k)\|}$, which can be shown geometrically in Figure 3.2 in the right triangle (colored red) involving $\nabla f(x_k)$, d'_k, and $d'_k - \nabla f(x_k)$.

Now we are given that $|\sin \theta_k| \leq \epsilon$, hence setting $\alpha = {}^{\langle \nabla f(x_k) | d_k \rangle}/_{\|d_k\|^2}$ ensures that the relative error criterion ${}^{\|d'_k - \nabla f(x_k)\|}/_{\|\nabla f(x_k)\|} \leq \epsilon$ is satisfied for ($\mathcal{ASD}_{\text{scaled}}$). Finally by applying Theorem 3.3 to ($\mathcal{ASD}_{\text{scaled}}$), we arrive at (3.6). □

3.2 Base descent properties of NCGMs

In this section, we analyze NCGMs as approximate steepest descent methods satisfying ($\mathcal{ASD}$) through a computer-assisted approach, where only the generated search directions matter, and not their magnitudes. This renders the analysis somewhat simpler, and we argue that this is a reasonable setting for improving the analysis and understanding of NCGMs.

This section builds on the intuition that when $|\sin \theta_k|$, where θ_k is the angle between the gradient and the search direction d_k at iteration k, is upper bounded in an appropriate fashion,

one can use Theorem 3.3 for obtaining convergence guarantees. In particular, we get nontrivial convergence guarantees as soon as θ_k can be bounded away from $\pm\frac{\pi}{2}$, i.e., $\sin\theta_k$ should be bounded away from 1 for ensuring that d_k's are descent directions. Of course, viewing NCGMs as approximate steepest descent methods is adversarial by nature, as it misses the point that the directions of NCGMs are meant to be better than those of vanilla gradient descent, while such analyses can only provide worse rates. Additionally, in Section 3.2.1, we provide additional justification behind analyzing NCGMs as approximate steepest descent methods through the lens of performance estimation problem (PEP), where we formulate the process of computing the worst-case $f(x_{k+1})-f_\star/f(x_k)-f_\star$ as optimization problems.

Albeit being pessimistic by construction, the analyses of this section are, to the best of our knowledge, novel for FR (for which we provide the first non-asymptotic convergence bound) and significantly better than the state-of-the-art bound for PRP. Furthermore, we show in Section 3.3.3 and Section 3.3.4 that there is actually nearly no room for improving those analyses.

Properties of NCGMs with exact line search. Before going into the detailed approach, let us review a few properties of the iterates of $(\mathcal{M})$. Note that the iterates of $(\mathcal{M})$ satisfy the following equalities:

$$\begin{aligned}\langle \nabla f(x_{k+1}) \mid d_k\rangle &= 0,\\ \langle \nabla f(x_{k+1}) \mid x_k - x_{k+1}\rangle &= 0,\\ \langle \nabla f(x_k) \mid d_k\rangle &= \|\nabla f(x_k)\|^2,\end{aligned} \tag{3.7}$$

where the first two equalities are the same as $(\mathcal{ASD}_{\text{relaxed}})$ following from exact line search. The last equality in (3.7) follows from applying the first equality to

$$\langle \nabla f(x_k) \mid d_k\rangle = \langle \nabla f(x_k) \mid \nabla f(x_k) + \beta_{k-1} d_{k-1}\rangle = \|\nabla f(x_k)\|^2. \tag{3.8}$$

Combining (3.8) with $\langle \nabla f(x_k) \mid d_k \rangle = \|\nabla f(x_k)\| \|d_k\| \cos\theta_k$, we obtain that $\|\nabla f(x_k)\|/\|d_k\| = \cos\theta_k$, thereby reaching $\sin^2\theta_k = 1 - \|\nabla f(x_k)\|^2/\|d_k\|^2$. If we have $\|d_k\|^2/\|\nabla f(x_k)\|^2 \leq c$ ($c \geq 1$ due to (3.7)), then $\sin^2\theta_k = 1 - (\|\nabla f(x_k)\|^2/\|d_k\|^2) \leq 1 - (1/c)$, yielding

$$|\sin\theta_k| \leq \sqrt{1 - 1/c}. \tag{3.9}$$

The first two equations of (3.7), in conjunction with (3.9), satisfied by NCGMs, correspond to the same set of conditions required to apply Theorem 3.3. Thus, if we can establish an upper bound for the ratio $\|d_k\|/\|\nabla f(x_k)\|$ in the context of NCGMs, we can translate this into their worst-case convergence rates using Theorem 3.3.

Section organization. In Section 3.2.1, we provide PEP-based perspective behind analyzing NCGMs as methods satisfying ($\mathcal{ASD}$). Section 3.2.2, first frames the problems of computing the worst-case $\|d_k\|/\|\nabla f(x_k)\|$ for PRP and FR as optimization problems for obtaining the desired bounds measuring the quality of the angle θ_k as PEPs. These PEPs are nonconvex but practically tractable QCQPs and can be solved numerically to certifiable global optimality using spatial branch-and-bound algorithms (detailed in Section 3.4), which allows (i) to construct "bad" functions serving as counter-examples on which the worst-case $\|d_k\|/\|\nabla f(x_k)\|$ for PRP and FR is achieved, and (ii) to identify closed-form solutions to the PEPs leading to proofs that can be verified in a standard and mathematically rigorous way. The convergence rates for PRP and FR are provided and proved in Section 3.2.3.

3.2.1 A PEP perspective behind viewing NCGMs as approximate steepest descent method

In this section, we provide a PEP-based perspective behind analyzing NCGMs as approximate steepest descent methods satisfying ($\mathcal{ASD}$) through the lens of PEP. In this PEP approach, we formulate the problems of computing the worst-case ratios of ${f(x_{k+1})-f_\star}/{f(x_k)-f_\star}$ as the following optimization problem:

$$\left(\begin{array}{ll} \underset{\substack{f,n,x_k,x_{k+1},d_k,d_{k+1},\\ \gamma_k,\beta_k}}{\text{maximize}} & \frac{f(x_{k+1})-f_\star}{f(x_k)-f_\star} \\ \text{subject to} & n\in\mathbb{N},\ f\in\mathcal{F}_{\mu,L}(\mathbb{R}^n),\ d_k,x_k\in\mathbb{R}^n, \\ & \langle \nabla f(x_k)\mid d_k\rangle = \|\nabla f(x_k)\|^2, \\ & \|d_k\|^2\leq c\|\nabla f(x_k)\|^2, \\ & (x_{k+1},d_{k+1},\beta_k) \text{ generated by } (\mathcal{M}) \text{ from } x_k \text{ and } d_k. \end{array}\right) \tag{3.10}$$

In Section 3.3.1, we will illustrate how we can formulate and solve (3.10) by casting it as a nonconvex QCQP. Note that in (3.10), the second constraint corresponds to third equation of (3.7) and the third constraint $\|d_k\|^2\leq c\|\nabla f(x_k)\|^2$ models that if $\nabla f(x_k)=0$ then $d_k=0$ for ($\mathcal{M}$). Note that ${\|d_k\|^2}/{\|\nabla f(x_k)\|^2}\geq 1$ because $\|\nabla f(x_k)\|^2\leq\|d_k\|^2$, which follows from applying Cauchy–Schwarz inequality to (3.8).

While solving the nonconvex QCQPs equivalent to (3.10) for different values of c, μ, and L, we found that the worst-case ${f(x_{k+1})-f_\star}/{f(x_k)-f_\star}$ is strictly monotonically increasing in c. Naturally, assigning an arbitrary value to c would not be reasonable to get the best bound, because the search direction generated by ($\mathcal{M}$) may not admit such a value. For example, for PRP, c is always upper bounded by $1+{L^2}/{\mu^2}$ as ${\|d_k\|^2}/{\|\nabla f(x_k)\|^2}\leq 1+{L^2}/{\mu^2}$ for PRP [11, Theorem 2]. As we are interested in obtaining the tightest upper bound on ${f(x_{k+1})-f_\star}/{f(x_k)-f_\star}$,

the natural question is: What is the smallest admissible value of c, i.e., what is the least upper bound on the ratio $\|d_k\|^2/\|\nabla f(x_k)\|^2$ generated by $(\mathcal{M})$? To that end, we numerically computed the least upper bound on c by solving a problem similar to (3.10), except we replaced the objective $f(x_{k+1})-f_\star/f(x_k)-f_\star$ with $\|d_{k+1}\|^2/\|\nabla f(x_{k+1})\|^2$ and then replaced the indices $k, k+1$ with $k-1, k$, respectively. In Section 3.2.2, we provide the details on formulating the problems of computing the worst-case ratios of $\|d_k\|^2/\|\nabla f(x_k)\|^2$ as nonconvex QCQPs. After we computed the least upper bound on c numerically, we put them in (3.10). We then solved the associated nonconvex QCQP to global optimality, which numerically provided us with the tightest upper bound on worst-case $f(x_{k+1})-f_\star/f(x_k)-f_\star$. Remarkably, at this stage, we found that these numerically computed worst-case $f(x_{k+1})-f_\star/f(x_k)-f_\star$ for $(\mathcal{M})$ exactly matched the analytical bound prescribed in Corollary 3.1. This PEP-based observation provides us a justification for analyzing NCGMs as approximate steepest descent methods and demonstrates the viability of this approach.

3.2.2 Computing worst-case search directions

In this section, we formulate the problems of computing the worst-case ratios of $\|d_k\|/\|\nabla f(x_k)\|$. Following the classical steps introduced in [5, 6], we show that it can be cast as a nonconvex QCQP.

For doing that, we assume that at iteration $k-1$ the NCGM has not reached optimality, so $\nabla f(x_{k-1}) \neq 0$. Because $\|\nabla f(x_{k-1})\|^2 \leq \|d_{k-1}\|^2$ (follows from applying Cauchy–Schwarz inequality to (3.8)), without loss of generality we define the ratio $c_{k-1} \triangleq \|d_{k-1}\|^2/\|\nabla f(x_{k-1})\|^2$ where $c_{k-1} \geq 1$. Then, denoting by c_k the worst-case ratio $\|d_k\|^2/\|\nabla f(x_k)\|^2$ arising when applying $(\mathcal{M})$ to the minimization of an L-smooth μ-strongly convex function, we will compute c_k as a function of L, μ, and c_{k-1}. In other words, we use a *Lyapunov*-type point of view and take the stand of somewhat *forgetting* about how d_{k-1} was generated (except

through the fact that it satisfies (3.7)). Then, we compute the worst possible next search direction d_k that the algorithm could generate given that d_{k-1} satisfies a certain quality. Thereby, we obtain an upper bound on the evolution of the *quality* of the search directions (quantified by c_k) obtained throughout the iterative procedure. Formally, we compute

$$c_k(\mu, L, c_{k-1}) \triangleq \left(\begin{array}{ll} \underset{\substack{f, x_{k-1}, d_{k-1}, \\ x_k, d_k, \beta_{k-1}, n}}{\text{maximize}} & \frac{\|d_k\|^2}{\|\nabla f(x_k)\|^2} \\ \text{subject to} & n \in \mathbb{N},\ f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n),\ d_{k-1}, x_{k-1} \in \mathbb{R}^n, \\ & x_k, d_k \text{ and } \beta_{k-1} \text{ generated by } (\mathcal{M}) \text{ from } x_{k-1} \text{ and } d_{k-1}, \\ & \langle \nabla f(x_{k-1});\, d_{k-1} \rangle = \|\nabla f(x_{k-1})\|^2, \\ & \|d_{k-1}\|^2 = c_{k-1} \|\nabla f(x_{k-1})\|^2. \end{array} \right) \tag{3.11}$$

For computing $c_k(\mu, L, c_{k-1})$, we reformulate (3.11) as follows. Denote $I \triangleq \{k-1, k\}$. An appropriate sampling of the variable f (which is inconveniently infinite-dimensional) allows

us to cast (3.11) as:

$$c_k(\mu, L, c_{k-1}) = \left(\begin{array}{ll} \underset{\substack{\{d_i\}_{i\in I},\, \gamma_{k-1},\, \beta_{k-1},\\ \{(x_i,g_i,f_i)\}_{i\in I},\, n}}{\text{maximize}} & \frac{\|d_k\|^2}{\|g_k\|^2} \\ \text{subject to} & n \in \mathbb{N},\ \beta_{k-1} \in \mathbb{R},\ d_{k-1}, d_k \in \mathbb{R}^n, \\ & \{(x_i, g_i, f_i)\}_{i\in I} \subset \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}, \\ & \exists f \in \mathcal{F}_{\mu,L} : \begin{cases} f(x_i) = f_i \\ \nabla f(x_i) = g_i \end{cases} \quad \forall i \in I, \\ & \gamma_{k-1} = \underset{\gamma}{\text{argmin}}\, f(x_{k-1} - \gamma\, d_{k-1}), \\ & x_k = x_{k-1} - \gamma_{k-1} d_{k-1}, \\ & \beta_{k-1} = \frac{\|g_k\|^2 - \eta\langle g_k; g_{k-1}\rangle}{\|g_{k-1}\|^2}, \\ & d_k = g_k + \beta_{k-1} d_{k-1}, \\ & \langle g_{k-1};\, d_{k-1}\rangle = \|g_{k-1}\|^2, \\ & \|d_{k-1}\|^2 = c_{k-1}\|g_{k-1}\|^2. \end{array} \right) \tag{3.12}$$

Using Theorem 3.1, the existence constraint can be replaced by a set of linear/quadratic inequalities (3.2) for all pairs of triplets in $\{(x_i, g_i, f_i)\}_{i\in I}$ without changing the objective value. So, applying Theorem 3.1 to (3.12) followed by an homogeneity argument and a few substitutions based on (3.7), we arrive at:

$$
c_k(\mu, L, c_{k-1}) = \left(\begin{array}{ll}
\underset{\substack{\{d_i\}_{i\in I},\, \gamma_{k-1}, \beta_{k-1}, \\ \{(x_i, g_i, f_i)\}_{i\in I},\, n}}{\text{maximize}} & \|d_k\|^2 \\
\text{subject to} & n \in \mathbb{N},\ d_{k-1}, x_{k-1} \in \mathbb{R}^n, \\
& f_i \geq f_j + \langle g_j;\, x_i - x_j \rangle + \frac{1}{2(1-\frac{\mu}{L})} \Big(\frac{1}{L}\|g_i - g_j\|^2 \\
& \qquad +\mu\|x_i - x_j\|^2 - 2\frac{\mu}{L} \langle g_i - g_j;\, x_i - x_j \rangle \Big), \quad i, j \in I, \\
& \langle g_{k-1};\, d_{k-1} \rangle = \|g_{k-1}\|^2, \\
& \langle g_k;\, d_{k-1} \rangle = 0, \\
& \langle g_k;\, x_{k-1} - x_k \rangle = 0, \\
& x_k = x_{k-1} - \gamma_{k-1} d_{k-1}, \\
& \beta_{k-1} = \frac{\|g_k\|^2 - \eta \langle g_k; g_{k-1} \rangle}{\|g_{k-1}\|^2}, \\
& d_k = g_k + \beta_{k-1} d_{k-1} \\
& \|d_{k-1}\|^2 = c_{k-1}\|g_{k-1}\|^2, \\
& \|g_k\|^2 = 1.
\end{array} \right) \tag{3.13}
$$

We now show how to transform (3.13) into a finite-dimensional nonconvex QCQP based on PEP methodologies developed in [4–6]. To that goal, note that (3.13) contains function values, inner product, and norm-squared involving $\{(x_i, g_i, f_i)\}_{i\in I}$ and $\{d_i\}_{i\in I}$, to model such terms in a compact manner, we introduce the following Grammian matrices:

$$
\begin{aligned}
H &= [x_{k-1} \mid g_{k-1} \mid g_k \mid d_{k-1}] \in \mathbb{R}^{n\times 4}, \\
G &= H^\top H \in \mathbb{S}^4_+, \quad \mathbf{rank}\, G \leq n, \\
F &= [f_{k-1} \mid f_k] \in \mathbb{R}^{1\times 2}.
\end{aligned} \tag{3.14}
$$

We next define the following notation for selecting columns and elements of H and F:

$$\begin{aligned}
&\mathbf{x}_{k-1} = e_1,\ \mathbf{g}_{k-1} = e_2,\ \mathbf{g}_k = e_3,\ \mathbf{d}_{k-1} = e_4,\ \text{(all in } \mathbb{R}^4\text{)}\\
&\mathbf{f}_{k-1} = e_1,\ \mathbf{f}_k = e_2,\ \text{(all in } \mathbb{R}^2\text{)},\\
&\mathbf{x}_k = \mathbf{x}_{k-1} - \gamma_{k-1}\mathbf{d}_{k-1},\ \text{(all in } \mathbb{R}^4\text{)},\\
&\mathbf{d}_k = \mathbf{g}_k + \beta_{k-1}\mathbf{d}_{k-1},\ \text{(all in } \mathbb{R}^4\text{)}.
\end{aligned} \tag{3.15}$$

This ensures that $x_i = H\mathbf{x}_i$, $g_i = H\mathbf{g}_i$, $d_i = H\mathbf{d}_i$, $f_i = F\mathbf{f}_i$, for all $i \in I$. Next, for appropriate choices of matrices $A_{i,j}$, $B_{i,j}$, $C_{i,j}$, $\widetilde{C}_{i,j}$, $D_{i,j}$, $\widetilde{D}_{i,j}$, $E_{i,j}$, and vector $a_{i,j}$, we can ensure that the following reformulations hold for all $i, j \in I$:

$$\begin{aligned}
&\langle g_j;\, x_i - x_j\rangle = \mathbf{tr}\, GA_{i,j},\\
&\|x_i - x_j\|^2 = \mathbf{tr}\, GB_{i,j},\\
&\|g_i - g_j\|^2 = \mathbf{tr}\, GC_{i,j},\ \|g_i\|^2 = \mathbf{tr}\, GC_{i,\star},\\
&\|d_i - d_j\|^2 = \mathbf{tr}\, G\widetilde{C}_{i,j},\ \|d_i\|^2 = \mathbf{tr}\, G\widetilde{C}_{i,\star},\\
&\langle g_i;\, g_j\rangle = \mathbf{tr}\, GD_{i,j},\\
&\langle g_i;\, d_j\rangle = \mathbf{tr}\, G\widetilde{D}_{i,j},\\
&\langle g_i - g_j;\, x_i - x_j\rangle = \mathbf{tr}\, GE_{i,j},\\
&f_j - f_i = Fa_{i,j},
\end{aligned} \tag{3.16}$$

where, using (3.15), and using symmetric outer product notation $(\cdot \odot \cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^{n\times n}$

such that for any $x, y \in \mathbb{R}^n$, $x \odot y = 1/2 \left(xy^\top + yx^\top \right)$, we define

$$
\begin{aligned}
A_{i,j} &= \mathbf{g}_j \odot (\mathbf{x}_i - \mathbf{x}_j) \\
B_{i,j} &= (\mathbf{x}_i - \mathbf{x}_j) \odot (\mathbf{x}_i - \mathbf{x}_j) \\
C_{i,j} &= (\mathbf{g}_i - \mathbf{g}_j) \odot (\mathbf{g}_i - \mathbf{g}_j),\ C_{i,\star} = \mathbf{g}_i \odot \mathbf{g}_i, \\
\widetilde{C}_{i,j} &= (\mathbf{d}_i - \mathbf{d}_j) \odot (\mathbf{d}_i - \mathbf{d}_j),\ \widetilde{C}_{i,\star} = \mathbf{d}_i \odot \mathbf{d}_i, \\
D_{i,j} &= \mathbf{g}_i \odot \mathbf{g}_j, \\
\widetilde{D}_{i,j} &= \mathbf{g}_i \odot \mathbf{d}_j, \\
E_{i,j} &= (\mathbf{g}_i - \mathbf{g}_j) \odot (\mathbf{x}_i - \mathbf{x}_j), \\
a_{i,j} &= \mathbf{f}_j - \mathbf{f}_i.
\end{aligned}
\tag{3.17}
$$

Using (3.17), we can write (3.13) as a finite-dimensional optimization problem with a positive-semidefinite constraint:

$$c_k(\mu, L, c_{k-1}) = \left(\begin{array}{ll} \underset{G, F, \gamma_{k-1}, \beta_{k-1}, n}{\text{maximize}} & \mathbf{tr}\, G\widetilde{C}_{k,\star} \\ \text{subject to} & \mathbf{tr}\, G\widetilde{D}_{k-1,k-1} = \mathbf{tr}\, GC_{k-1,\star}, \\ & \mathbf{tr}\, G\widetilde{D}_{k,k-1} = 0, \\ & \mathbf{tr}\, GA_{k-1,k} = 0, \\ & \beta_{k-1} \times \mathbf{tr}\, GC_{k-1,\star} = \mathbf{tr}\, G\left(C_{k,\star} - \eta D_{k,k-1}\right), \\ & \mathbf{tr}\, G\widetilde{C}_{k-1,\star} \leq c_{k-1}\, \mathbf{tr}\, GC_{k-1,\star}, \\ & Fa_{i,j} + \mathbf{tr}\, G\Big[A_{i,j} \\ & \quad + \frac{1}{2(1-\frac{\mu}{L})}\left(\frac{1}{L}C_{i,j} + \mu B_{i,j} - 2\frac{\mu}{L}E_{i,j}\right)\Big] \leq 0,\ i, j \in I, \\ & \mathbf{tr}\, GC_{k,\star} = 1, \\ & G \in \mathbb{S}^4_+,\ \mathbf{rank}\, G \leq n. \end{array} \right) \tag{3.18}$$

In the optimization problem above, the only constraint involving n is $\mathbf{rank}\, G \leq n$, where the optimal value of the problem is monotonically nondecreasing in n. As $G \in \mathbb{S}^4_+$ (implying $\mathbf{rank}\, G \leq 4$), at the optimal solution, we have $\mathbf{rank}\, G \leq n$ satisfied automatically without impacting the optimal objective value, and the worst-case function would have a dimension of less than or equal to 4.

Next, we model the positive semidefinite constraint $G \in \mathbb{S}^4_+$ using Cholesky factorization. Recall that a matrix $Z \in \mathbb{S}^m$ is positive semidefinite if and only if it has a Cholesky factorization $P^\top P = Z$, where $P \in \mathbb{R}^{m \times m}$ [77, Corollary 7.2.9]. Hence, positive semidefiniteness of G can be reformulated as $G = \widetilde{H}^\top \widetilde{H}$, where $\widetilde{H} \in \mathbb{R}^{4\times 4}$, i.e., for $G = H^\top H$ in (3.14), we can let

$H \in \mathbb{R}^{4\times 4}$. Thus, we can write (3.18) as the following nonconvex QCQP:

$$c_k(\mu, L, c_{k-1}) = \left(\begin{array}{ll} \underset{\substack{G,F,H,\gamma_{k-1},\beta_{k-1},\\ \Theta,\{\Theta_{i,j}\}_{i,j\in I}}}{\text{maximize}} & \mathbf{tr}\, G\Theta \\ \text{subject to} & \mathbf{tr}\, G\widetilde{D}_{k-1,k-1} = \mathbf{tr}\, GC_{k-1,\star}, \\ & \mathbf{tr}\, G\widetilde{D}_{k,k-1} = 0, \\ & \mathbf{tr}\, GA_{k-1,k} = 0, \\ & \beta_{k-1} \times \mathbf{tr}\, GC_{k-1,\star} = \mathbf{tr}\, G\left(C_{k,\star} - \eta D_{k,k-1}\right), \\ & \mathbf{tr}\, G\widetilde{C}_{k-1,\star} \leq c_{k-1}\, \mathbf{tr}\, GC_{k-1,\star}, \\ & Fa_{i,j} + \mathbf{tr}\, G\Big[A_{i,j} \\ & \quad + \frac{1}{2(1-\frac{\mu}{L})}\left(\frac{1}{L}C_{i,j} + \mu\Theta_{i,j} - 2\frac{\mu}{L}E_{i,j}\right)\Big] \leq 0,\ i,j \in I, \\ & \Theta = \widetilde{C}_{k,\star}, \quad \Theta_{i,j} = B_{i,j},\ i,j \in I, \\ & G = H^\top H, \\ & \mathbf{tr}\, GC_{k,\star} = 1. \end{array}\right) \tag{$\mathcal{D}$}$$

Note that in the problem above, Θ and $\{\Theta_{i,j}\}_{i,j\in I_N^\star}$ are introduced as separate decision variables to formulate the cubic constraints arising from $\widetilde{C}_{k,\star}$ and $B_{i,j}$ as quadratic constraints, respectively. This nonconvex QCQP can be solved to certifiable global optimality using a custom spatial branch-and-bound algorithm described in Section 3.4.

Finally, we recall that numerical solutions to ($\mathcal{D}$) correspond to worst-case functions that can be obtained through the reconstruction procedure from Theorem 3.2. In addition, numerical solutions can serve as inspirations for devising rigorous mathematical proofs, as presented next.

3.2.3 Worst-case bounds for PRP and FR

In this section, we provide explicit solutions to ($\mathcal{D}$) for PRP and FR. Those results are then used for deducing simple convergence bounds through a straightforward application of Theorem 3.3.

The main benefit of our proof structures is that they are verifiable through both calculations by hand and also by symbolic computer algebra systems. Our proofs to the lemmas in this section (Lemmas 3.1, 3.2, 3.3) are obtained by linearly combining the constraints of associated performance estimation problems with appropriate weights, where the weights themselves correspond to dual variables for the performance estimation problems. This makes our proofs independently verifiable programmatically using open-source symbolic computation libraries such as `SymPy` [136] and `Wolfram Language` [137]. We have provided notebooks for the symbolic verifications of our proofs to Lemmas 3.1, 3.2, 3.3 in the `Symbolic_Verifications` folder of our open-source code available at

`https://github.com/Shuvomoy/NCG-PEP-code`.

3.2.3.1 A worst-case bound for Polak-Ribière-Polyak (PRP)

Solving ($\mathcal{D}$) with $\eta = 1$ to global optimality allows obtaining the following worst-case bound for PRP quantifying the *quality* of the search direction with respect to the gradient direction.

Lemma 3.1 (Worst-case search direction for PRP)**.** *Let $f \in \mathcal{F}_{\mu,L}$, and let $x_{k-1}, d_{k-1} \in \mathbb{R}^n$ and x_k, d_k be generated by the PRP method (i.e., ($\mathcal{M}$) with $\eta = 1$). It holds that:*

$$\frac{\|d_k\|^2}{\|\nabla f(x_k)\|^2} \leq \frac{(1+q)^2}{4q}, \tag{3.19}$$

with $q \triangleq \mu/L$. *Equivalently,* $|\sin\theta_k| \leq \epsilon$ *holds, where* θ_k *is the angle between* $\nabla f(x_k)$ *and* d_k *and* $\epsilon = (1-q)/(1+q)$.

Proof. Recall that $x_k = x_{k-1} - \gamma_{k-1}\, d_{k-1}$ and $d_k = \nabla f(x_k) + \beta_{k-1} d_{k-1}$. The proof consists of the following weighted sum of inequalities:

- optimality condition of the line search, with weight $\lambda_1 = -\beta_{k-1}^2 \frac{1+q}{L\gamma_{k-1}q}$:

$$\langle \nabla f(x_k) \mid d_{k-1} \rangle = 0,$$

- smoothness and strong convexity of f between x_{k-1} and x_k, with weight $\lambda_2 = \frac{\beta_{k-1}^2(1+q)^2}{L\gamma_{k-1}^2(1-q)q}$:

$$\begin{aligned}
f(x_{k-1}) \geq & f(x_k) + \langle \nabla f(x_k) \mid x_{k-1} - x_k \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|x_{k-1} - x_k - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2 \\
= & f(x_k) + \gamma_{k-1}\langle \nabla f(x_k) \mid d_{k-1} \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1} d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2
\end{aligned}$$

- smoothness and strong convexity of f between x_k and x_{k-1}, with weight $\lambda_3 = \lambda_2$:

$$\begin{aligned}
f(x_k) \geq & f(x_{k-1}) + \langle \nabla f(x_{k-1}) \mid x_k - x_{k-1} \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|x_{k-1} - x_k - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2 \\
= & f(x_{k-1}) - \gamma_{k-1}\langle \nabla f(x_{k-1}), d_{k-1} \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1} d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2
\end{aligned}$$

- definition of β_{k-1} with weight $\lambda_4 = \frac{\beta_{k-1}(1+q)}{L\gamma_{k-1}q}$:

$$\begin{aligned}0 &= \langle \nabla f(x_{k-1}) \mid \nabla f(x_k)\rangle - \|\nabla f(x_k)\|^2 + \beta_{k-1}\|\nabla f(x_{k-1})\|^2 \\ &= \langle \nabla f(x_{k-1}) \mid \nabla f(x_k)\rangle - \|\nabla f(x_k)\|^2 + \beta_{k-1}\langle \nabla f(x_{k-1}) \mid d_{k-1}\rangle.\end{aligned}$$

We arrive at the following weighted sum:

$$\begin{aligned}0 \geq& \lambda_1\langle \nabla f(x_k) \mid d_{k-1}\rangle \\ &+ \lambda_2\Big[f(x_k) - f(x_{k-1}) + \gamma_{k-1}\langle \nabla f(x_k) \mid d_{k-1}\rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\ &\qquad + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1}d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2\Big] \\ &+ \lambda_3\Big[f(x_{k-1}) - f(x_k) - \gamma_{k-1}\langle \nabla f(x_{k-1}) \mid d_{k-1}\rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\ &\qquad + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1}d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2\Big] \\ &+ \lambda_4\Big[\langle \nabla f(x_{k-1}) \mid \nabla f(x_k)\rangle - \|\nabla f(x_k)\|^2 + \beta_{k-1}\langle \nabla f(x_{k-1}) \mid d_{k-1}\rangle\Big]\end{aligned}$$

which can be reformulated exactly as (by expanding both expressions and then observing that all terms match, we have shown the calculation for this reformulation in Appendix 3.A.3.1)

$$\begin{aligned}0 \geq& \|d_k\|^2 - \frac{(1+q)^2}{4q}\|\nabla f(x_k)\|^2 \\ &+ \frac{4\beta_{k-1}^2 q}{(1-q)^2}\left\|d_{k-1} - \tfrac{1+q}{2L\gamma_{k-1}q}\nabla f(x_{k-1}) + \tfrac{2\beta_{k-1}(1+q) - L\gamma_{k-1}(1-q)^2}{4\beta_{k-1}L\gamma_{k-1}q}\nabla f(x_k)\right\|^2, \\ \geq& \|d_k\|^2 - \frac{(1+q)^2}{4q}\|\nabla f(x_k)\|^2,\end{aligned}$$

thereby arriving at (3.19). Finally, using (3.9), we have $|\sin\theta_k| \leq \epsilon$ where $\epsilon = {}^{(1-q)}/_{(1+q)}$. □

In Appendix 3.A.2, we numerically showcase the tightness of the worst-case bounds (3.19) for PRP. By tightness, we mean that we verified numerically that there exist $n \in \mathbb{N}$, functions

$f \in \mathcal{F}_{\mu,L}$ and $x_{k-1}, d_{k-1} \in \mathbb{R}^n$ such that $\|d_k\|^2 = ({}^{(1+q)^2}/_{4q}) \|\nabla f(x_k)\|^2$. This is done by exhibiting feasible points to ($\mathcal{D}$) (obtained by solving ($\mathcal{D}$) numerically for $\eta = 1$) for different values of the inverse condition number q and c_{k-1}. Those feasible points were verified through other (existing) software [126, 138].

The following rate is a direct consequence of Lemma 3.1 and Theorem 3.3. Perhaps surprisingly, the following guaranteed convergence rate for PRP corresponds to that of gradient descent with an exact line search (Theorem 3.3 with $\epsilon = 0$) when the condition number is squared.

Theorem 3.4 (Worst-case bound for PRP)**.** *Let $f \in \mathcal{F}_{\mu,L}$, and $x_k, d_k \in \mathbb{R}^n$ and x_{k+1}, $d_{k+1} \in \mathbb{R}^n$ be generated by respectively $k \geq 0$ and $k+1$ iterations of the PRP method (i.e., ($\mathcal{M}$) with $\eta = 1$). It holds that*

$$f(x_{k+1}) - f_\star \leq \left(\frac{1-q^2}{1+q^2}\right)^2 (f(x_k) - f_\star)\,,$$

with $q \triangleq {}^{\mu}/_{L}$.

Proof. The desired claim is a direct consequence of Corollary 3.1 with $\epsilon = \frac{1-q}{1+q}$. That is, the PRP scheme can be seen as a descent method with direction d_k satisfying $\|d_k - \nabla f(x_k)\| \leq \epsilon \|\nabla f(x_k)\|$. □

As a take-away from this theorem, we obtained an improved bound on the convergence rate of PRP, but possibly not in the most satisfying way: this analysis strategy does not allow beating steepest descent. Furthermore, this bound is tight for one iteration assuming that the current search direction satisfies ${}^{\|d_k\|^2}/_{\|\nabla f(x_k)\|^2} = {}^{(1+q)^2}/_{4q}$. However, it does not specify whether such an angle can be achieved on the same worst-case instances as those where Theorem 3.3 is achieved. In other words, there might be no worst-case instances where the bounds (3.6)

and (3.19) are tight simultaneously, possibly leaving room for improvement in the analysis of PRP. We show in Section 3.3 that we could indeed slightly improve this bound by taking into account the *history* of the method in a more appropriate way by examining multiple iterations of ($\mathcal{M}$) rather than a single one.

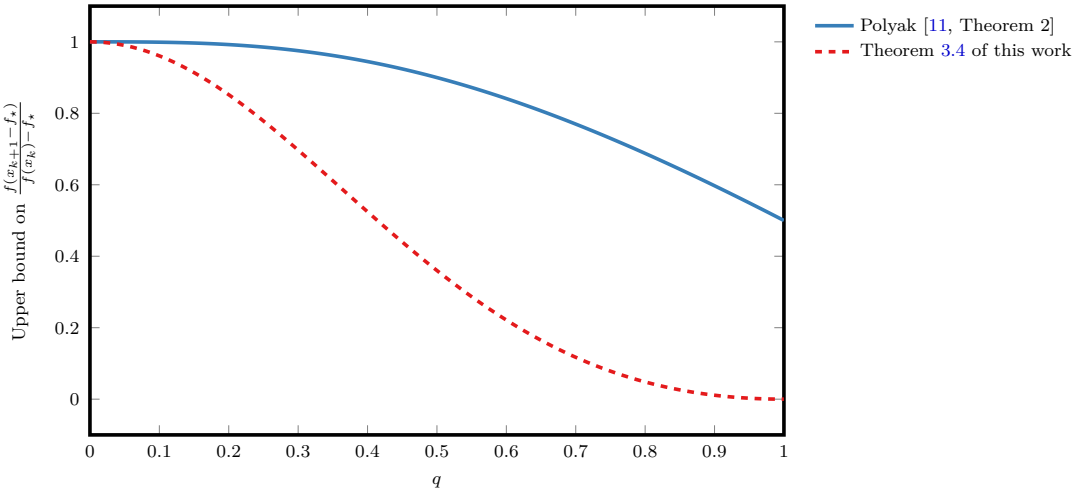


Figure 3.3: Comparison between the upper bounds on $f(x_{k+1})-f_\star / f(x_k)-f_\star$ vs. condition number $q \triangleq \mu/L$ for PRP by Polyak [11, Theorem 2] and Theorem 3.4 of this chapter.

Remark 3.1. *The only worst-case complexity result that we are aware of in the context of PRP for smooth strongly convex problems was provided by Polyak in [11, Theorem 2]:*

$$f(x_{k+1}) - f_\star \leq \left(1 - \frac{q}{1+\frac{1}{q^2}}\right)(f(x_k) - f_\star)\,.$$

Figure 3.3 shows that the upper bound on $f(x_{k+1})-f_\star / f(x_k)-f_\star$ for PRP (for different values of q) provided by [11, Theorem 2] is significantly worse compared to that of Theorem 3.4. From what we can tell, this is due to two main weaknesses in the proof of Polyak [11, Theorem 2]: a weaker analysis of gradient descent, and a weaker analysis of the direction (and in particular that $\|d_k\|^2/\|\nabla f(x_k)\|^2 \leq 1 + 1/q^2$). That is, whereas gradient descent with exact line searches is

guaranteed to achieve an accuracy $f(x_k) - f_\star \leq \varepsilon$ *in* $O(1/q \log 1/\varepsilon)$*, our analysis provides an* $O(1/q^2 \log 1/\varepsilon)$ *guarantee for PRP, where Polyak's guarantee for PRP is* $O(1/q^3 \log 1/\varepsilon)$*. As a reference, note that the lower complexity bound (achieved by a few methods, including many variations of Nesterov's accelerated gradients) is of order* $O(\sqrt{1/q} \log 1/\varepsilon)$*.*

3.2.3.2 A worst-case bound for Fletcher-Reeves (FR)

Similar to the obtaining of the bound for PRP, our bound for FR follows from solving ($\mathcal{D}$) (for $\eta = 0$) in closed-form. We start by quantifying the *quality* of the search direction with respect to the steepest descent direction. Unlike PRP, where the worst-case ratio $\|d_k\|^2/\|\nabla f(x_k)\|^2$ depends only on the condition number q, in FR, the ratio $\|d_k\|^2/\|\nabla f(x_k)\|^2$ depends also on the previous ratio $\|d_{k-1}\|^2/\|\nabla f(x_{k-1})\|^2$. To show this dependence, we first establish the following bound on the FR update parameter β_{k-1} in terms of $\|d_{k-1}\|^2/\|\nabla f(x_{k-1})\|^2$ and q.

Lemma 3.2 (Bound on β_{k-1} for FR)**.** *Let* $f \in \mathcal{F}_{\mu,L}$*, and let* $x_{k-1}, d_{k-1} \in \mathbb{R}^n$ *and* x_k*,* d_k *be generated by the FR method (i.e.,* ($\mathcal{M}$) *where* $c_{k-1} > 1$*, it holds that:*

$$0 \leq \beta_{k-1} \leq \frac{1}{c_{k-1}} \frac{\left(1 - q + 2\sqrt{(c_{k-1} - 1)q}\right)^2}{4q}, \tag{3.20}$$

where $q \triangleq \mu/L$*.*

Proof. First, note that $\beta_{k-1} \geq 0$ by definition. The other part of the proof consists of the following weighted sum of inequalities:

- relation between $\nabla f(x_{k-1})$ and d_{k-1} with weight $\lambda_1 = \gamma_{k-1}(L + \mu) - \frac{2\sqrt{\beta_{k-1}}}{\sqrt{(c_{k-1}-1)c_{k-1}}}$:

$$0 = \langle \nabla f(x_{k-1}) \mid d_{k-1} \rangle - \|\nabla f(x_{k-1})\|^2,$$

- optimality condition of the line search with weight $\lambda_2 = \frac{2}{c_{k-1}} - \gamma_{k-1}(L+\mu)$:

$$0 = \langle \nabla f(x_k) \mid d_{k-1} \rangle ,$$

- definition of β_{k-1} with weight $\lambda_3 = \frac{\sqrt{c_{k-1}-1}}{\sqrt{\beta_{k-1}c_{k-1}}}$:

$$0 = \|\nabla f(x_k)\|^2 - \beta_{k-1}\|\nabla f(x_{k-1})\|^2,$$

- initial condition on the ratio $\frac{\|d_{k-1}\|^2}{\|\nabla f(x_{k-1})\|^2}$ with weight $\lambda_4 = -\gamma_{k-1}^2 L\mu + \frac{\sqrt{\beta_{k-1}}}{c_{k-1}\sqrt{(c_{k-1}-1)c_{k-1}}}$:

$$0 = \|d_{k-1}\|^2 - c_{k-1}\|g_{k-1}\|^2$$

- smoothness and strong convexity of f between x_{k-1} and x_k, with weight $\lambda_5 = L - \mu$:

$$\begin{aligned}
0 \geq & - f(x_{k-1}) + f(x_k) + \langle \nabla f(x_k) \mid x_{k-1} - x_k \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|x_{k-1} - x_k - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2 \\
= & f(x_k) - f(x_{k-1}) + \gamma_{k-1}\langle \nabla f(x_k) \mid d_{k-1} \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1}d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2
\end{aligned}$$

  where going from the first line to the second, we used $x_{k-1} - x_k = \gamma_{k-1}d_{k-1}$,

- smoothness and strong convexity of f between x_k and x_{k-1}, with weight $\lambda_6 = \lambda_5$:

$$\begin{aligned}
0 \geq & - f(x_k) + f(x_{k-1}) + \langle \nabla f(x_{k-1}) \mid x_k - x_{k-1} \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|x_{k-1} - x_k - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2 \\
= & f(x_{k-1}) - f(x_k) - \gamma_{k-1}\langle \nabla f(x_{k-1}) \mid d_{k-1} \rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1}d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2
\end{aligned}$$

  where going from the first line to the second, we again used $x_{k-1} - x_k = \gamma_{k-1}d_{k-1}$,

The final weighted sum of inequalities is:

$$
\begin{aligned}
0 \geq\ & \lambda_1 \left[\langle \nabla f(x_{k-1}) \mid d_{k-1}\rangle - \|\nabla f(x_{k-1})\|^2\right] + \lambda_2 \left[\langle \nabla f(x_k) \mid d_{k-1}\rangle\right] \\
& + \lambda_3 \left[\|\nabla f(x_k)\|^2 - \beta_{k-1}\|\nabla f(x_{k-1})\|^2\right] + \lambda_4 \left[\|d_{k-1}\|^2 - c_{k-1}\|g_{k-1}\|^2\right] \\
& + \lambda_5 \Big[f(x_k) - f(x_{k-1}) + \gamma_{k-1}\langle \nabla f(x_k) \mid d_{k-1}\rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& \quad + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1}d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2\Big] \\
& + \lambda_6 \Big[f(x_{k-1}) - f(x_k) - \gamma_{k-1}\langle \nabla f(x_{k-1}) \mid d_{k-1}\rangle + \tfrac{1}{2L}\|\nabla f(x_{k-1}) - \nabla f(x_k)\|^2 \\
& \quad + \tfrac{\mu}{2(1-\mu/L)}\|\gamma_{k-1}d_{k-1} - \tfrac{1}{L}(\nabla f(x_{k-1}) - \nabla f(x_k))\|^2\Big],
\end{aligned}
$$

which can be reformulated exactly as (by expanding both expressions and then observing that all terms match, we have shown the calculation for this reformulation in Appendix 3.A.3.2):

$$
\begin{aligned}
0 \geq & \|\nabla f(x_k)\|^2 - \nu(\beta_{k-1}, \gamma_{k-1}, c_{k-1}, \mu, L)\|\nabla f(x_{k-1})\|^2 \\
& + \left\| \sqrt[4]{\frac{\beta_{k-1}}{(c_{k-1}-1)c_{k-1}^3}}\, d_{k-1} - \sqrt[4]{\frac{\beta_{k-1}c_{k-1}}{c_{k-1}-1}}\nabla f(x_{k-1}) + \sqrt[4]{\frac{c_{k-1}-1}{\beta_{k-1}c_{k-1}}}\nabla f(x_k) \right\|^2 \\
\geq & \|\nabla f(x_k)\|^2 - \nu(\beta_{k-1}, \gamma_{k-1}, c_{k-1}, \mu, L)\|\nabla f(x_{k-1})\|^2,
\end{aligned}
$$

where

$$
\nu(\beta_{k-1}, \gamma_{k-1}, c_{k-1}, \mu, L) = 2\sqrt{1 - \frac{1}{c_{k-1}}}\sqrt{\beta_{k-1}} - c_{k-1}\gamma_{k-1}^2 L\mu + \gamma_{k-1}(L+\mu) - 1.
$$

So, we have:

$$\begin{aligned}
\beta_{k-1} &\leq \nu(\beta_{k-1}, \gamma_{k-1}, c_{k-1}, \mu, L) \\
&\Leftrightarrow \beta_{k-1} - 2\sqrt{1-\frac{1}{c_{k-1}}}\sqrt{\beta_{k-1}} \leq -c_{k-1}\gamma_{k-1}^2 L\mu + \gamma_{k-1}(L+\mu) - 1 \\
&\Rightarrow \beta_{k-1} - 2\sqrt{1-\frac{1}{c_{k-1}}}\sqrt{\beta_{k-1}} \leq \max_{\gamma}\left(-c_{k-1}\gamma_{k-1}^2 L\mu + \gamma_{k-1}(L+\mu) - 1\right).
\end{aligned}$$

Because, $-c_{k-1}\gamma_{k-1}^2 L\mu + \gamma_{k-1}(L+\mu) - 1$ is a concave function in γ_{k-1}, its maximum can be achieved by differentiating the term with respect to γ_{k-1}, equating it to 0, and then solving for γ_{k-1}. The corresponding maximum value is equal to ${}^{(L+\mu)^2}/_{4c_{k-1}L\mu} - 1$ and achieved at $\gamma_{k-1} = {}^{(L+\mu)}/_{(2c_{k-1}L\mu)}$. Hence, the last inequality becomes:

$$\begin{aligned}
&\beta_{k-1} - 2\sqrt{1-\frac{1}{c_{k-1}}}\sqrt{\beta_{k-1}} - \frac{(L+\mu)^2}{4c_{k-1}L\mu} + 1 \leq 0 \\
&\quad\Leftrightarrow \left(\sqrt{\beta_{k-1}}\right)^2 - 2\sqrt{1-\frac{1}{c_{k-1}}}\sqrt{\beta_{k-1}} + \left(\sqrt{1-\frac{1}{c_{k-1}}}\right)^2 - \frac{(L+\mu)^2}{4c_{k-1}L\mu} - \left(\sqrt{1-\frac{1}{c_{k-1}}}\right)^2 + 1 \leq 0 \\
&\quad\Leftrightarrow \left(\sqrt{\beta_{k-1}} - \sqrt{1-\frac{1}{c_{k-1}}}\right)^2 \leq \frac{(L+\mu)^2}{4c_{k-1}L\mu} + \cancel{1} - \frac{1}{c_{k-1}} - \cancel{1} = \frac{1}{c_{k-1}}\left(\frac{(L+\mu)^2}{4L\mu} - 1\right) \\
&\quad\Leftrightarrow \sqrt{\beta_{k-1}} \leq \sqrt{1-\frac{1}{c_{k-1}}} + \sqrt{\frac{(L+\mu)^2}{4c_{k-1}L\mu} - \frac{1}{c_{k-1}}}.
\end{aligned}$$

Thereby, squaring both sides (which are nonnegative) of the last inequality and then through some algebra, we reach

$$\begin{aligned}
\beta_{k-1} &\leq 1 + \frac{(L-\mu)}{c_{k-1}}\sqrt{\frac{(c_{k-1}-1)}{\mu L}} + \frac{\mu^2 - 6\mu L + L^2}{4c_{k-1}\mu L} \\
&= \frac{1}{c_{k-1}}\frac{\left(1 - q + 2\sqrt{(c_{k-1}-1)q}\right)^2}{4q},
\end{aligned}$$

which completes the proof.

□

Next, we prove a bound quantifying the quality of the search directions of FR.

Lemma 3.3 (Worst-case search direction for FR)**.** *Let $f \in \mathcal{F}_{\mu,L}$, and let $x_{k-1}, d_{k-1} \in \mathbb{R}^n$ and x_k, d_k be generated by the FR method (i.e., ($\mathcal{M}$) with $\eta = 0$). For any $c_{k-1} \in \mathbb{R}$ such that ${\|d_{k-1}\|^2}/{\|\nabla f(x_{k-1})\|^2} = c_{k-1}$, where $c_{k-1} > 1$, it holds that:*

$$\frac{\|d_k\|^2}{\|\nabla f(x_k)\|^2} \leq c_k \triangleq 1 + \frac{\left(1 - q + 2\sqrt{(c_{k-1} - 1)q}\right)^2}{4q}, \tag{3.21}$$

with $q \triangleq {\mu}/{L}$.

Equivalently, $|\sin\theta_k| \leq \epsilon$ holds, where θ_k is the angle between $\nabla f(x_k)$ and d_k holds with $\epsilon = \sqrt{1 - {1}/{c_k}}$.

Proof. The proof consists of the following weighted sum of equalities:

- optimality condition of the line search with weight $\lambda_1 = 2\beta_{k-1}$:

$$0 = \langle \nabla f(x_k) \mid d_{k-1} \rangle,$$

- the quality of the search direction with weight $\lambda_2 = \beta_{k-1}^2$:

$$0 = \|d_{k-1}\|^2 - c_{k-1}\|\nabla f(x_{k-1})\|^2,$$

- definition of β_{k-1} with weight $\lambda_3 = -c_{k-1}\beta_{k-1}$:

$$0 = \|\nabla f(x_k)\|^2 - \beta_{k-1}\|\nabla f(x_{k-1})\|^2.$$

The weighted sum can be simplified as (calculation shown in Appendix 3.A.3.3)

$$\begin{aligned}0 =&\lambda_1 \left[\langle \nabla f(x_k); d_{k-1}\rangle\right] + \lambda_2 \left[\|d_{k-1}\|^2 - c_{k-1}\|\nabla f(x_{k-1})\|^2\right] + \lambda_3 \left[\|\nabla f(x_k)\|^2 - \beta_{k-1}\|\nabla f(x_{k-1})\|^2\right] \\ &= \|d_k\|^2 - (1 + c_{k-1}\beta_{k-1})\|\nabla f(x_k)\|^2.\end{aligned}$$

Hence,

$$\begin{aligned}\|d_k\|^2 &= (1 + c_{k-1}\beta_{k-1})\|\nabla f(x_k)\|^2 \\ &\leq \left(1 + \frac{\left(1 - q + 2\sqrt{(c_{k-1} - 1)q}\right)^2}{4q}\right)\|\nabla f(x_k)\|^2,\end{aligned}$$

where in the last line we have used the upper bound on β_{k-1} from (3.20). This gives us (3.21). Finally, using (3.9), we have $|\sin\theta_k| \leq \epsilon$, where $\epsilon = \sqrt{1 - 1/c_k}$. □

Similar to PRP, in Appendix 3.A.2, we compare this last bound with the worst example that we were able to find numerically (i.e., worst feasible points to ($\mathcal{D}$)). Thereby, we conclude tightness of the bound on the quality of the search direction (3.21). That is, we claim that for all values of q and c_{k-1}, there exist $n \in \mathbb{N}$, functions $f \in \mathcal{F}_{\mu,L}$ and $x_{k-1}, d_{k-1} \in \mathbb{R}^n$ such that the bound from Lemma 3.3 is achieved with equality.

That being said, this bound only allows obtaining unsatisfactory convergence results for FR, although not letting much room for improvements, as showed in the next sections.

Theorem 3.5 (Worst-case bound)**.** *Let $f \in \mathcal{F}_{\mu,L}$, and $x_k, d_k \in \mathbb{R}^n$ and x_{k+1}, $d_{k+1} \in \mathbb{R}^n$ be generated by respectively $k \geq 0$ and $k+1$ iterations of the FR method (i.e., ($\mathcal{M}$) with $\eta = 0$). It holds that*

$$f(x_{k+1}) - f_\star \leq \left(\frac{1 - q\frac{1-\epsilon_k}{1+\epsilon_k}}{1 + q\frac{1-\epsilon_k}{1+\epsilon_k}}\right)^2 (f(x_k) - f_\star),$$

with $\epsilon_k = \sqrt{{(1-q)^2(k-1)^2}/{4q+(1-q)^2(k-1)^2}}$.

Proof. The desired claim is a direct consequence of Corollary 3.1 with Lemma 3.3. Indeed, it follows from

$$c_k \leq 1 + \frac{\left(1 - \frac{\mu}{L} + 2\sqrt{(c_{k-1}-1)\frac{\mu}{L}}\right)^2}{\frac{4\mu}{L}}$$

(the guarantee from Lemma 3.3 for the quality of the search direction) which we can rewrite as

$$\sqrt{c_{k+1}-1} \leq \frac{1-q+2\sqrt{(c_k-1)q}}{2\sqrt{q}}$$

with $c_0 - 1 = 0$, thereby arriving to $c_k \leq 1 + k^2{(1-q)^2}/{4q}$ by recursion. For applying Theorem 3.3, we compute $\epsilon_k = \sqrt{1 - {1}/{c_k}} \leq \sqrt{{(1-q)^2k^2}/{4q+(1-q)^2k^2}}$ and reach the desired statement. □

It is clear that the statement of Theorem 3.5 is rather disappointing, as the convergence rate of the FR variation can become arbitrarily close to 1. While this guarantee clearly does not give a total and fair picture of the true behavior of FR in practice, it seems in line with the practical necessity to effectively restart the method as it runs [110].

The next section is devoted to studying the possibilities for obtaining tighter guarantees for PRP and FR beyond the simple single-iteration worst-case analyses of this section (which are tight for one iteration, but not beyond), showing that we cannot hope to improve the convergence rates for those methods without further assumptions on the problems at hand.

3.3 Obtaining better worst-case bounds for NCGMs

In the previous section, we established closed-form bounds on ratios between consecutive function values for NCGMs by characterizing worst-case search directions. Albeit being tight for the analysis of NCGMs for one iteration, the bounds that we obtained are disappointingly

inferior to those of the vanilla gradient descent. In this section, we investigate the possibility of obtaining better worst-case guarantees for NCGMs. For doing this using our framework, one natural possibility for us is to go beyond the study of a single iteration (since our results appear to be tight for this situation). Therefore, in contrast with the previous section, we now proceed only numerically and provide worst-case bounds without closed-forms.

More precisely, we solve the corresponding PEPs in two regimes. In short, the difference between the two regimes resides in the type of bounds under consideration.

1. The first type of bounds can be thought of as a "Lyapunov" approach which studies N iterations of ($\mathcal{M}$) starting at some iterate (x_k, d_k) (for which we "neglect" how it was generated). In this first setup, we numerically compute worst-case bounds on $f(x_{k+N})-f_\star / f(x_k)-f_\star$ for different values of N (namely $N = 1, 2, 3, 4$). As for the results of Section 3.2, we quantify the quality of the couple (x_k, d_k) by requiring that $\|d_k\|^2 \leq c_k \|\nabla f(x_k)\|^2$. When $N = 1$, this setup corresponds to that of Section 3.2. Stemming from the fact that the worst-case behaviors observed for $N = 1$ might not be compatible between consecutive iterations, we expect the quality of the bounds to improve with N. Of course, the main weakness of this approach is the fact that we neglect how (x_k, d_k) was generated.
2. As a natural complementary alternative, the second type of bounds studies N iterations of ($\mathcal{M}$) initiated at x_0 (with $d_0 = \nabla f(x_0)$). Whereas the first type of bounds is by construction more conservative, it has the advantage of being *recursive*: it is valid for all $k \geq 0$. On the other side, the second type of bounds is only valid for the first N iterations (the bound cannot be used recursively), but it cannot be improved at all. That is, we study *exact* worst-case ratio $f(x_N)-f_\star / f(x_0)-f_\star$ for a few different values of N (namely $N \in \{1, 2, 3, 4\}$). In this setup, we obtain worst-case bounds that are only valid close to initialization. However, it has the advantage of being unimprovable, as

we do not neglect how the search direction is generated.

Section organization. This section is organized as follows. First, in Section 3.3.1 we present the performance estimation problems for ($\mathcal{M}$) specifically for computing the worst-case ratios $^{f(x_{k+N})-f_\star}/_{f(x_k)-f_\star}$ and $^{f(x_N)-f_\star}/_{f(x_0)-f_\star}$. In Section 3.3.2, we describe the steps to arrive at the nonconvex QCQP formulations for the performance estimation problems considered. Then, Section 3.3.3 and Section 3.3.4 presents our findings for respectively PRP and FR. In Appendix 3.A.4, we discuss how to generate the counter-examples from the solutions to the nonconvex QCQPs.

3.3.1 Computing numerical worst-case scenarios

Similar to (3.11), the problem of computing the worst-case ratio $^{f(x_{k+N})-f_\star}/_{f(x_k)-f_\star}$ is framed as the following nonconvex maximization problem (for $c \geq 1$ and $q \triangleq {}^{\mu}/_{L}$):

$$\rho_N(q,c) \triangleq \left(\begin{array}{ll} \underset{\substack{f,n,\{x_{k+i}\},\{d_{k+i}\}_i,\\ \{\gamma_{k+i}\}_i,\{\beta_{k+i}\}_i}}{\text{maximize}} & \frac{f(x_{k+N})-f_\star}{f(x_k)-f_\star} \\ \text{subject to} & n \in \mathbb{N},\ f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n),\ d_k, x_k \in \mathbb{R}^n, \\ & \langle \nabla f(x_k) \mid d_k \rangle = \|\nabla f(x_k)\|^2, \\ & \|d_k\|^2 \leq c\|\nabla f(x_k)\|^2, \\ & \begin{pmatrix} x_{k+1} \\ d_{k+1} \\ \beta_k \end{pmatrix}, \ldots, \begin{pmatrix} x_{k+N} \\ d_{k+N} \\ \beta_{k+N-1} \end{pmatrix} \text{ generated by } (\mathcal{M}) \text{ from } x_k \text{ and } d_k. \end{array} \right) \tag{$\mathcal{B}_{\text{Lyapunov}}$}$$

We proceed similarly for $f(x_N)-f_\star/f(x_0)-f_\star$:

$$\rho_{N,0}(q) \triangleq \left(\begin{array}{ll} \underset{\substack{f,n,\{x_{k+i}\},\{d_{k+i}\}_i,\\ \{\gamma_{k+i}\}_i,\{\beta_{k+i}\}_i}}{\text{maximize}} & \frac{f(x_N)-f_\star}{f(x_0)-f_\star} \\ \text{subject to} & n \in \mathbb{N},\ f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n),\ x_0 \in \mathbb{R}^n, \\ & d_0 = \nabla f(x_0), \\ & \begin{pmatrix} x_1 \\ d_1 \\ \beta_0 \end{pmatrix}, \ldots, \begin{pmatrix} x_N \\ d_N \\ \beta_{N-1} \end{pmatrix} \text{ generated by } (\mathcal{M}) \text{ from } x_k \text{ and } d_k. \end{array} \right) \tag{$\mathcal{B}_{\text{exact}}$}$$

Obviously, $\rho_N(q,c) \geq \rho_{N,0}(q)$ for any $c \geq 1$. We solve the nonconvex QCQP reformulations of ($\mathcal{B}_{\text{Lyapunov}}$) and ($\mathcal{B}_{\text{exact}}$) numerically to high precision (reformulation details shown in Section 3.3.2) for $N \in \{1,2,3,4\}$ and report the corresponding results in what follows. In the numerical experiments, we fix the values of c using Lemma 3.1 for PRP in ($\mathcal{B}_{\text{Lyapunov}}$), thereby computing $\rho_N\left(q, {(1+q)^2}/{4q}\right)$ whose results are provided in Figure 3.4 of Section 3.3.3. For FR, c can become arbitrarily bad and we therefore only compute $\rho_{N,0}(q)$ via ($\mathcal{B}_{\text{exact}}$). The numerical values for $\rho_{N,0}(q)$ respectively PRP and FR are provided in Figure 3.5 and Figure 3.6, located in Section 3.3.3 and Section 3.3.4, respectively.

In the next section, we describe the nonconvex QCQP formulations for ($\mathcal{B}_{\text{Lyapunov}}$) and ($\mathcal{B}_{\text{exact}}$). Readers interested in the findings of our numerical experiments by solving the nonconvex QCQPs can skip to Section 3.3.3 (for PRP) and Section 3.3.4 (for FR).

3.3.2 Nonconvex QCQP reformulations of ($\mathcal{B}_{\text{Lyapunov}}$) and ($\mathcal{B}_{\text{exact}}$)

Similar to the reformulations from ($\mathcal{D}$), ($\mathcal{B}_{\text{Lyapunov}}$) and ($\mathcal{B}_{\text{exact}}$) can be cast as nonconvex QCQPs, where the number of nonconvex constraints grows quadratically with N. Thereby,

solving them to global optimality in reasonable time for $N = 3, 4$ is already challenging.

Therefore, rather than solving the nonconvex QCQP reformulations of ($\mathcal{B}_{\text{Lyapunov}}$) and ($\mathcal{B}_{\text{exact}}$) directly, we compute upper bounds and lower bounds by solving more tractable nonconvex QCQP formulations. We then show that the relative gap between the upper and lower bounds is less than 10% which thereby indicates that there is essentially no room for further improvement.

3.3.2.1 Nonconvex QCQP reformulation of ($\mathcal{B}_{\text{Lyapunov}}$)

This section presents the nonconvex QCQP formulations for our upper bound $\overline{\rho}_N(q, c)$ and lower bound $\underline{\rho}_N(q, c)$ on $\rho_N(q, c)$. We use the notation $[a : b] = \{a, a+1, a+2, \ldots, b-1, b\}$ where a, b are integers.

Computing $\overline{\rho}_N(q, c)$. Using (3.7), we have the following relaxation of ($\mathcal{B}_{\text{Lyapunov}}$), which provides upper bounds on $\rho_N(q, c)$:

$$
\left(\begin{array}{ll}
\underset{f,n,\{x_{k+i}\}_{i\in[0:N]},\{d_{k+i}\}_{i\in[0:N]}}{\text{maximize}} & \frac{f(x_{k+N})-f_\star}{f(x_k)-f_\star} \\
\text{subject to} & n \in \mathbb{N},\ f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n), \\
& x_{k+i},\ d_{k+i} \in \mathbb{R}^n, \quad i \in [0:N] \\
& \|d_k\|^2 \leq c\|\nabla f(x_k)\|^2, \\
& \langle \nabla f(x_{k+i+1}) \mid d_{k+i} \rangle = 0, \quad i \in [0:N-1], \\
& \langle \nabla f(x_{k+i+1}) \mid x_{k+i} - x_{k+i+1} \rangle = 0, \quad i \in [0:N-1], \\
& \langle \nabla f(x_{k+i}) \mid d_{k+i} \rangle = \|\nabla f(x_{k+i})\|^2, \quad i \in [0:N-1], \\
& d_{k+i+1} = g_{k+i+1} + \beta_{k+i} d_{k+i}, \quad i \in [0:N-2], \\
& \beta_{k+i} = \frac{\|g_{k+i+1}\|^2 - \eta\langle g_{k+i+1} | g_{k+i}\rangle}{\|g_{k+i}\|^2}, \quad i \in [0:N-2].
\end{array}\right) \tag{3.22}
$$

Using the notation $g_i \triangleq \nabla f(x_i)$ and $f_i \triangleq f(x_i)$ again, and then applying an homogeneity argument, we write (3.22) as:

$$\overline{\rho}_N(q,c) = \left(\begin{array}{ll} \text{maximize} & f_{k+N} - f_\star \\ \text{subject to} & n \in \mathbb{N},\ f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n), \\ & x_{k+i},\ d_{k+i} \in \mathbb{R}^n, \quad i \in [0:N] \\ & \|d_k\|^2 \leq c\|g_k\|^2, \\ & \langle g_{k+i+1} \mid d_{k+i} \rangle = 0, \quad i \in [0:N-1], \\ & \langle g_{k+i+1} \mid x_{k+i} - x_{k+i+1} \rangle = 0, \quad i \in [0:N-1], \\ & \langle g_{k+i} \mid d_{k+i} \rangle = \|g_{k+i}\|^2, \quad i \in [0:N-1], \\ & d_{k+i+1} = g_{k+i+1} + \beta_{k+i} d_{k+i}, \quad i \in [0:N-2], \\ & \beta_{k+i-1} = \frac{\|g_{k+i}\|^2 - \eta\langle g_{k+i} | g_{k+i-1}\rangle}{\|g_{k+i-1}\|^2}, \quad i \in [1:N-1], \\ & f_k - f_\star = 1, \end{array} \right) \tag{3.23}$$

where $f, n, \{x_{k+i}\}_{i\in[0:N]}, \{d_{k+i}\}_{i\in[0:N]}$ are the decision variables. Define $I_N^\star = \{\star, k, k+1, \ldots, k+N\}$. Next, note that the equation $d_{k+i+1} = g_{k+i+1} + \beta_{k+i} d_{k+i}$ for $i \in [0:N-2]$, can be written equivalently as the following set of equations:

$$\begin{aligned} &\chi_{j,i} = \chi_{j,i-1}\beta_{k+i-1}, \quad i \in [1:N-1], j \in [0:i-2], \\ &\chi_{i-1,i} = \beta_{k+i-1}, \quad i \in [1:N-1], \\ &d_{k+i} = g_{k+i} + \sum_{j=1}^{i-1} \chi_{j,i} g_{k+j} + \chi_{0,i} d_k, \quad i \in [1:N-1], \end{aligned} \tag{3.24}$$

where we have introduced the intermediate variables $\chi_{j,i}$, which will aid us in formulating (3.23) as a nonconvex QCQP down the line. In absence of these intermediate variables in (3.24), the resultant constraints in the final optimization problem will involve polynomials of degree three or more in the decision variables, and such optimization problems present a

significantly greater challenge in solving to global optimality compared to a QCQP. Next, using (3.24) and Theorem 3.1, we can equivalently write (3.23) as:

$$\overline{\rho}_N(q,c) = \left(\begin{array}{ll} \text{maximize} & f_{k+N} - f_\star \\ \text{subject to} & n \in \mathbb{N}, \\ & f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2(1-\frac{\mu}{L})} \Big(\frac{1}{L} \|g_i - g_j\|^2 \\ & \quad + \mu \|x_i - x_j\|^2 - 2\frac{\mu}{L} \langle g_i - g_j \mid x_i - x_j \rangle \Big), \quad i,j \in I_N^\star, \\ & \|d_k\|^2 \leq c \|g_k\|^2, \\ & \langle g_{k+i+1} \mid d_{k+i} \rangle = 0, \quad i \in [0:N-1], \\ & \langle g_{k+i+1} \mid x_{k+i} - x_{k+i+1} \rangle = 0, \quad i \in [0:N-1], \\ & \langle g_{k+i} \mid d_{k+i} \rangle = \|g_{k+i}\|^2, \quad i \in [0:N-1], \\ & \beta_{k+i-1} = \frac{\|g_{k+i}\|^2 - \eta \langle g_{k+i} | g_{k+i-1} \rangle}{\|g_{k+i-1}\|^2}, \quad i \in [1:N-1], \\ & \chi_{j,i} = \chi_{j,i-1} \beta_{k+i-1}, \quad i \in [1:N-1], j \in [0:i-2], \\ & \chi_{i-1,i} = \beta_{k+i-1}, \quad i \in [1:N-1], \\ & d_{k+i} = g_{k+i} + \sum_{j=1}^{i-1} \chi_{j,i} g_{k+j} + \chi_{0,i} d_k, \quad i \in [1:N-1], \\ & f_k - f_\star = 1, \\ & g_\star = 0,\ x_\star = 0,\ f_\star = 0, \\ & \{x_i, g_i, f_i\}_{i \in I_N^\star} \subset \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R},\ \{d_i\}_{i \in I_N^\star \setminus \{k+N\}} \subset \mathbb{R}^n, \\ & \{\beta_{k+i}\}_{i \in [0:N-2]} \subset \mathbb{R},\ \{\chi_{j,i}\}_{j \in [0:N-2], i \in [0:N-1]} \subset \mathbb{R}, \end{array} \right) \tag{3.25}$$

where $\{x_{k+i}, g_{k+i}, f_{k+i}\}_i, n, \{d_{k+i}\}_i, \{\beta_{k+i}\}_i, \{\chi_{j,i}\}_{j,i}$ are the decision variables. Note that we have set $g_\star = 0$, $x_\star = 0$, and $f_\star = 0$ without loss of generality, because both the objective and the function class are closed and invariant under shifting variables and function values. We

introduce Grammian matrices again:

$$\begin{aligned} &H = [d_k \mid g_k \mid g_{k+1} \mid g_{k+2} \mid \cdots \mid g_{k+N} \mid x_k \mid x_{k+1} \mid x_{k+2} \mid \cdots \mid x_{k+N}] \in \mathbb{R}^{n\times(2N+3)}, \\ &G = H^\top H \in \mathbb{S}_+^{(2N+3)},\ \mathbf{rank}\, G \le n, \\ &F = [f_k \mid f_{k+1} \mid \ldots \mid f_{k+N}] \in \mathbb{R}^{1\times(N+1)}. \end{aligned} \tag{3.26}$$

Using the same arguments described in Section 3.2.2, we can ignore the constraint $\mathbf{rank}\, G \le n$, and confine H to be in $\mathbb{R}^{(2N+3)\times(2N+3)}$ without loss of generality. Next, define the following notation for selecting columns and elements of H and F:

$$\begin{aligned} &\mathbf{x}_\star = \mathbf{0} \in \mathbb{R}^{2N+3},\ \mathbf{d}_k = e_1 \in \mathbb{R}^{2N+3},\ \mathbf{g}_{k+i} = e_{i+2} \in \mathbb{R}^{2N+3}, \\ &\mathbf{x}_{k+i} = e_{(N+2)+(i+1)} \in \mathbb{R}^{2N+3}, \\ &\mathbf{f}_\star = \mathbf{0},\ \mathbf{f}_{k+i} = e_{i+1} \in \mathbb{R}^{(N+1)}, \\ &\mathbf{d}_{k+i} = \mathbf{g}_{k+i} + \sum_{j=1}^{i-1} \chi_{j,i}\mathbf{g}_{k+j} + \chi_{0,i}\mathbf{d}_k \in \mathbb{R}^{2N+3}, \end{aligned} \tag{3.27}$$

where $i \in [0:N]$. This ensures that we have $x_i = H\mathbf{x}_i$, $g_i = H\mathbf{g}_i$, $d_i = H\mathbf{d}_i$, $f_i = F\mathbf{f}_i$ for all $i \in I_N^\star$. For appropriate choices of matrices $A_{i,j}$,$B_{i,j}$, $C_{i,j}$, $\widetilde{C}_{i,j}$, $D_{i,j}$, $\widetilde{D}_{i,j}$, $E_{i,j}$, and vector $a_{i,j}$ as defined in (3.16), where $\mathbf{x}_i$, $\mathbf{g}_i$, $\mathbf{f}_i$, $\mathbf{d}_i$ are taken from (3.27) now, we can ensure that the identities in (3.17) hold for all $i, j \in I_N^\star$. Using those identities and using the definition of $G = H^\top H$, where $H \in \mathbb{R}^{(2N+3)\times(2N+3)}$, we can write (3.25) as the following nonconvex

QCQP:

$$\overline{\rho}_N(q,c) = \left(\begin{array}{ll} \text{maximize} & Fa_{\star,k+N} \\ \text{subject to} & Fa_{i,j} + \mathbf{tr}\, G\left[A_{i,j} + \frac{1}{2(1-\frac{\mu}{L})}\left(\frac{1}{L}C_{i,j} + \mu B_{i,j} - 2\frac{\mu}{L}E_{i,j}\right)\right] \leq 0, \quad i,j \in I_N^\star, \\ & \mathbf{tr}\, G\widetilde{C}_{k,\star} \leq c\, \mathbf{tr}\, GC_{k,\star}, \\ & \mathbf{tr}\, G\widetilde{D}_{k+i+1,k+i} = 0, \quad i \in [0:N-1], \\ & \mathbf{tr}\, GA_{k+i,k+i+1} = 0, \quad i \in [0:N-1], \\ & \mathbf{tr}\, G\widetilde{D}_{k+i,k+i} = \mathbf{tr}\, GC_{k+i,\star} \quad i \in [0:N-1], \\ & \beta_{k+i-1} \times \mathbf{tr}\, GC_{k+i-1,\star} = \mathbf{tr}\, G\left(C_{k+i,\star} - \eta D_{k+i,k+i-1}\right), \quad i \in [1:N-1], \\ & \chi_{j,i} = \chi_{j,i-1}\beta_{k+i-1}, \quad i \in [1:N-1],\ j \in [0:i-2], \\ & \chi_{i-1,i} = \beta_{k+i-1}, \quad i \in [1:N-1], \\ & Fa_{\star,k} = 1, \\ & G = H^\top H, \\ & F \in \mathbb{R}^{N+1},\ G \in \mathbb{S}^{2N+3},\ H \in \mathbb{R}^{(2N+3)\times(2N+3)}, \\ & \{\beta_{k+i}\}_{i\in[0:N-2]} \subset \mathbb{R},\ \{\chi_{j,i}\}_{j\in[0:N-2],i\in[0:N-1]} \subset \mathbb{R}, \end{array} \right) \tag{3.28}$$

where $F, G, H, \{\chi_{j,i}\}_{j,i}, \{\beta_{k+i}\}_i$ are the decision variables.

Computing $\underline{\rho}_N(q,c)$ and corresponding counter-examples . We now discuss how we can calculate $\underline{\rho}_N(q,c)$ and construct the corresponding "bad" function. This function serves as a counter-example, illustrating scenarios where ($\mathcal{M}$) performs poorly. Once we have solved (3.28), it provides us with the corresponding CG update parameters, which we denote by $\overline{\beta}_i$. If we can solve ($\mathcal{B}_{\text{Lyapunov}}$) with the CG update parameters fixed to the $\overline{\beta}_i$ found from (3.28), then it will provide us with the lower bound $\underline{\rho}_N(\mu,L,c)$. This process also yields a "bad" function that acts as a counter-example, which we explain next. Using the notation $g_i \triangleq \nabla f(x_i)$ and $f_i \triangleq f(x_i)$, then applying the homogeneity argument, we can compute

$\underline{\rho}_N(q,c)$ by finding a feasible solution to the following optimization problem:

$$\left(\begin{array}{ll} \text{maximize} & f_{k+N} - f_\star \\ \text{subject to} & n \in \mathbb{N},\ f \in \mathcal{F}_{\mu,L}(\mathbb{R}^n), \\ & x_{k+i},\ d_{k+i} \in \mathbb{R}^n, \quad i \in [0:N] \\ & \|d_k\|^2 \leq c\|g_k\|^2, \\ & \gamma_{k+i} = \operatorname{argmin}_\gamma f(x_{k+i} - \gamma d_{k+i}), \quad i \in [0:N-1], \\ & x_{k+i+1} = x_{k+i} - \gamma_{k+i} d_{k+i}, \quad i \in [0:N-1], \\ & d_{k+i+1} = g_{k+i+1} + \overline{\beta}_{k+i} d_{k+i}, \quad i \in [0:N-2], \\ & \overline{\beta}_{k+i-1} = \frac{\|g_{k+i}\|^2 - \eta\langle g_{k+i} | g_{k+i-1}\rangle}{\|g_{k+i-1}\|^2}, \quad i \in [1:N-1], \\ & f_k - f_\star = 1, \end{array}\right) \tag{3.29}$$

where $f, n, \{x_{k+i}\}, \{d_{k+i}\}_i, \{\gamma_{k+i}\}_i$ are the decision variables. Next, note that the NCGM iteration scheme in (3.29) can be equivalently written as:

$$\begin{aligned} &\chi_{j,i} = \chi_{j,i-1}\overline{\beta}_{k+i-1}, \quad i \in [1:N-1], j \in [0:i-2] \\ &\chi_{i-1,i} = \overline{\beta}_{k+i-1}, \quad i \in [1:N-1] \\ &\alpha_{i,i-1} = \gamma_{k+i-1}, \quad i \in [1:N], \\ &\alpha_{i,j} = \gamma_{k+j} + \sum_{\ell=j+1}^{i-1} \gamma_{k+\ell}\chi_{j,\ell}, \quad i \in [1:N],\ j \in [0:i-2], \\ &x_{k+i} = x_k - \sum_{j=1}^{i-1} \alpha_{i,j} g_{k+j} - \alpha_{i,0} d_k, \quad i \in [1:N], \\ &d_{k+i} = g_{k+i} + \sum_{j=1}^{i-1} \chi_{j,i} g_{k+j} + \chi_{0,i} d_k, \quad i \in [1:N-1]. \end{aligned} \tag{3.30}$$

where we have introduced intermediate variables $\chi_{j,i}$ and $\alpha_{i,j}$ which will aid us in formulating (3.29) as a nonconvex QCQP. Define $I_N^\star = \{\star, k, k+1, \ldots, k+N\}$. Now using (3.30), Theorem

3.1, and (3.7), we can equivalently write (3.23) as:

$$
\left(\begin{array}{ll}
\text{maximize} & f_{k+N} - f_\star \\
\text{subject to} & n \in \mathbb{N}, \\
& f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2(1-\frac{\mu}{L})}\Big(\frac{1}{L}\|g_i - g_j\|^2 \\
& +\mu\|x_i - x_j\|^2 - 2\frac{\mu}{L}\langle g_i - g_j \mid x_i - x_j\rangle\Big), \quad i,j \in I_N^\star, \\
& \|d_k\|^2 \leq c\|g_k\|^2, \\
& \langle g_{k+i+1} \mid d_{k+i}\rangle = 0, \quad i \in [0:N-1], \\
& \langle g_{k+i+1} \mid x_{k+i} - x_{k+i+1}\rangle = 0, \quad i \in [0:N-1], \\
& \langle g_{k+i} \mid d_{k+i}\rangle = \|g_{k+i}\|^2, \quad i \in [0:N-1], \\
& \chi_{j,i} = \chi_{j,i-1}\overline{\beta}_{k+i-1}, \quad i \in [1:N-1], j \in [0:i-2] \\
& \chi_{i-1,i} = \overline{\beta}_{k+i-1}, \quad i \in [1:N-1] \\
& \alpha_{i,i-1} = \gamma_{k+i-1}, \quad i \in [1:N], \\
& \alpha_{i,j} = \gamma_{k+j} + \textstyle\sum_{\ell=j+1}^{i-1}\gamma_{k+\ell}\chi_{j,\ell}, \quad i \in [1:N],\ j \in [0:i-2], \\
& x_{k+i} = x_k - \textstyle\sum_{j=1}^{i-1}\alpha_{i,j}g_{k+j} - \alpha_{i,0}d_k, \quad i \in [1:N], \\
& d_{k+i} = g_{k+i} + \textstyle\sum_{j=1}^{i-1}\chi_{j,i}g_{k+j} + \chi_{0,i}d_k, \quad i \in [1:N-1]. \\
& \overline{\beta}_{k+i-1} = \frac{\|g_{k+i}\|^2 - \eta\langle g_{k+i} | g_{k+i-1}\rangle}{\|g_{k+i-1}\|^2}, \quad i \in [1:N-1], \\
& f_k - f_\star = 1, \\
& g_\star = 0,\ x_\star = 0,\ f_\star = 0, \\
& \{x_i, g_i, f_i\}_{i \in I_N^\star} \subset \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R},\ \{d_i\}_{i \in [k+1:k+N-1]} \subset \mathbb{R}^n, \\
& \{\chi_{j,i}\}_{j \in [0:N-2], i \in [0:N-1]} \subset \mathbb{R}, \\
& \{\gamma_{k+i}\}_{i \in [0:N]} \subset \mathbb{R},\ \{\alpha_{i,j}\}_{i \in [1:N], j \in [0:N-1]} \subset \mathbb{R},
\end{array}\right) \tag{3.31}
$$

where $\{x_{k+i}, g_{k+i}, f_{k+i}\}_i, n, \{\gamma_{k+i}\}_i, \{\chi_{j,i}\}_{j,i}, \{\alpha_{i,j}\}_{i,j}$ are the decision variables. We introduce the Grammian transformation:

$$
\begin{aligned}
&H = [x_k \mid g_k \mid g_{k+1} \mid \ldots \mid g_{k+N} \mid d_k] \in \mathbb{R}^{n\times(N+3)},\\
&G = H^\top H \in \mathbb{S}_+^{N+3}, \mathbf{rank}\, G \le n,\\
&F = [f_k \mid f_{k+1} \mid \ldots \mid f_{k+N}] \in \mathbb{R}^{1\times(N+1)}.
\end{aligned}
\tag{3.32}
$$

Using the same arguments described in Section 3.2.2, we again ignore the constraint $\mathbf{rank}\, G \le n$ and can let $H \in \mathbb{R}^{(N+3)\times(N+3)}$ without loss of generality. We next define the following notation for selecting columns and elements of H and F:

$$
\begin{aligned}
&\mathbf{g}_\star = 0 \in \mathbb{R}^{N+3},\ \mathbf{g}_{k+i} = e_{i+2} \in \mathbb{R}^{N+3}, \quad i \in [0:N],\\
&\mathbf{d}_k = e_{N+3} \in \mathbb{R}^{N+3},\\
&\mathbf{x}_k = e_1 \in \mathbb{R}^{N+2},\ \mathbf{x}_\star = 0 \in \mathbb{R}^{N+2},\\
&\mathbf{x}_{k+i}(\alpha) = \mathbf{x}_k - \sum_{j=1}^{i-1} \alpha_{i,j}\mathbf{g}_{k+j} - \alpha_{i,0}\mathbf{d}_k \in \mathbb{R}^{N+3}, \quad i \in [1:N],\\
&\mathbf{d}_{k+i}(\chi) = \mathbf{g}_{k+i} + \sum_{j=1}^{i-1} \chi_{j,i}\mathbf{g}_{k+j} + \chi_{0,i}\mathbf{d}_k, \quad i \in [1:N-1],\\
&\mathbf{f}_\star = 0 \in \mathbb{R}^{N+1},\ \mathbf{f}_{k+i} = e_{i+1} \in \mathbb{R}^{N+1}, \quad i \in [0:N],
\end{aligned}
\tag{3.33}
$$

which ensure $x_i = H\mathbf{x}_i,\ g_i = H\mathbf{g}_i,\ f_i = F\mathbf{f}_i,\ d_i = H\mathbf{d}_i$ for $i \in I_N^\star$. For appropriate choices of matrices $A_{i,j}$,$B_{i,j}$, $C_{i,j}$, $\widetilde{C}_{i,j}$, $D_{i,j}$, $\widetilde{D}_{i,j}$, $E_{i,j}$, and vector $a_{i,j}$ as defined in (3.16), where $\mathbf{x}_i$, $\mathbf{g}_i$, $\mathbf{f}_i$, $\mathbf{d}_i$ are from (3.33), we can ensure that the identities in (3.17) hold for all $i, j \in I_N^\star$. Using those identities and using the definition of $G = H^\top H$, where $H \in \mathbb{R}^{(N+3)\times(N+3)}$, we

can write (3.31) as the following nonconvex QCQP:

$$\left(\begin{array}{ll}
\text{maximize} & Fa_{\star,N} \\
\text{subject to} & Fa_{i,j} + \mathbf{tr}\, G\left[A_{i,j} + \frac{1}{2(1-\frac{\mu}{L})}\left(\frac{1}{L}C_{i,j} + \mu\Theta_{i,j} - 2\frac{\mu}{L}E_{i,j}\right)\right] \leq 0, \quad i,j \in I_N^\star, \\
& \Theta_{i,j} = B_{i,j}, \quad i,j \in I_N^\star, \\
& \mathbf{tr}\, G\widetilde{C}_{k,\star} \leq c\, \mathbf{tr}\, GC_{k,\star}, \\
& \mathbf{tr}\, G\widetilde{D}_{k+i+1,k+i} = 0, \quad i \in [0:N-1], \\
& \mathbf{tr}\, GA_{k+i,k+i+1} = 0, \quad i \in [0:N-1], \\
& \mathbf{tr}\, G\widetilde{D}_{k+i,k+i} = \mathbf{tr}\, GC_{k+i,\star} \quad i \in [0:N-1], \\
& \chi_{j,i} = \chi_{j,i-1}\overline{\beta}_{k+i-1}, \quad i \in [1:N-1], j \in [0:i-2] \\
& \chi_{i-1,i} = \overline{\beta}_{k+i-1}, \quad i \in [1:N-1] \\
& \alpha_{i,i-1} = \gamma_{k+i-1}, \quad i \in [1:N], \\
& \alpha_{i,j} = \gamma_{k+j} + \textstyle\sum_{\ell=j+1}^{i-1} \gamma_{k+\ell}\chi_{j,\ell}, \quad i \in [1:N],\ j \in [0:i-2], \\
& \overline{\beta}_{k+i-1} \times \mathbf{tr}\, GC_{k+i-1,\star} = \mathbf{tr}\, G\left(C_{k+i,\star} - \eta D_{k+i,k+i-1}\right), \quad i \in [1:N-1], \\
& Fa_{\star,k} = 1, \\
& G = H^\top H, \\
& F \in \mathbb{R}^{N+1},\ G \in \mathbb{S}^{N+3},\ H \in \mathbb{R}^{(N+3)\times(N+3)}, \\
& \{\chi_{j,i}\}_{j\in[0:N-2],i\in[0:N-1]} \subset \mathbb{R}, \\
& \{\gamma_{k+i}\}_{i\in[0:N]} \subset \mathbb{R},\ \{\alpha_{i,j}\}_{i\in[1:N],j\in[0:N-1]} \subset \mathbb{R},
\end{array}\right) \tag{3.34}$$

where $G, F, \{\Theta_{i,j}\}_{i,j\in I_N^\star}, H, \gamma, \alpha, \chi$ are the decision variables. Note that $\{\Theta_{i,j}\}_{i,j\in I_N^\star}$ is introduced as a separate decision variable to formulate the cubic constraints arising from $B_{i,j}$ as quadratic constraints. Also, to compute $\underline{\rho}_N(q,c)$, it suffices to find just a feasible solution to (3.34), in Section 3.4 we will discuss how to do so using our custom spatial branch-and-bound algorithm.

3.3.2.2 Nonconvex QCQP reformulation of ($\mathcal{B}_{\text{exact}}$)

Now we discuss how we compute the upper bound $\overline{\rho}_{N,0}(q)$ and lower bound $\underline{\rho}_{N,0}(q)$ to $\rho_{N,0}(q)$ defined in ($\mathcal{B}_{\text{exact}}$). The bound computation process is very similar to that of ($\mathcal{B}_{\text{Lyapunov}}$). Observe that, in ($\mathcal{B}_{\text{Lyapunov}}$), if we remove the constraint $\|d_k\|^2 \leq c\|\nabla f(x_k)\|^2$, set $k \triangleq 0$, and then add the constraint $d_0 = \nabla f(x_0)$, then it is identical to ($\mathcal{B}_{\text{exact}}$) (the constraint $\langle \nabla f(x_0) \mid d_0 \rangle = \|\nabla f(x_0)\|^2$ in ($\mathcal{B}_{\text{Lyapunov}}$) is a valid but redundant constraint for ($\mathcal{B}_{\text{exact}}$)).

So, to compute the upper bound $\overline{\rho}_{N,0}(q)$, we can follow a transformation process very similar to Section 3.3.2.1 but with a few changes. In (3.23) and (3.25), we remove the constraint $\|d_k\|^2 \leq c\|g_k\|^2$, and then add the constraint $g_k = d_k$. Second, the Grammian matrices defined in (3.26) stays the same, and in (3.27) the vectors $\{\mathbf{x}_i, \mathbf{g}_i, \mathbf{f}_i\}_{i \in I_N^\star}$ stays the same except we set $\mathbf{d}_k = \mathbf{g}_k = e_2 \in \mathbb{R}^{2N+3}$, which ensures that $d_k = F\mathbf{d}_k = g_k$. We then remove the constraint $\mathbf{tr}\, G\widetilde{C}_{k,\star} \leq c\, \mathbf{tr}\, GC_{k,\star}$ from (3.28) and finally set $k \triangleq 0$ in the resultant QCQP. The solution to the nonconvex QCQP will provide us the upper bound $\overline{\rho}_{N,0}(q)$ in ($\mathcal{B}_{\text{exact}}$).

To compute the lower bound $\underline{\rho}_{N,0}(q)$, we follow the same set of changes described in the last paragraph but to (3.29) in Section 3.3.2.1.

3.3.2.3 The relative gap between the lower bounds and upper bounds

Tables 3.1, 3.2, 3.3 record the relative gap between lower bounds and upper bounds for a few representative values of q obtained by solving the aforementioned nonconvex QCQPs associated with ($\mathcal{B}_{\text{Lyapunov}}$) and ($\mathcal{B}_{\text{exact}}$) using our custom spatial branch-and-bound algorithm described in Section 3.4. Note that the tables contain a few negative entries close to zero which are due to the absolute gap being of the same order as the accuracy of the solver ($1\text{e}-6$). For the full list for all values, we refer to our open-source code, which also allows for computing these bounds for a user-specified value of q as well. In all cases, the relative

gap is less than 10%. In most cases, it is significantly better.

$q =$	0.001	0.005	0.02	0.04	0.06	0.08	0.1	0.3	0.5
$N = 1$	3e−8	−1e−6	3e−9	6e−8	9e−8	2e−7	2e−7	1e−6	3e−7
$N = 2$	2e−6	6e−7	−3e−8	9e−8	1e−7	8e−8	3e−7	8e−3	4e−4
$N = 3$	5e−6	5e−4	7e−3	2e−2	3e−2	4e−2	2e−2	5e−2	−3e−7
$N = 4$	2e−4	3e−3	2e−2	7e−2	1e−1	3e−2	4e−2	4e−2	4e−2

Table 3.1: Relative gaps $\overline{\rho}_N(q,c) - \underline{\rho}_N(q,c) / \overline{\rho}_N(q,c)$ for PRP with $c = (1+q)^2/4q$.

$q =$	0.001	0.005	0.02	0.04	0.06	0.08	0.1	0.3	0.5
$N = 2$	7e−6	2e−4	2e−3	7e−3	1e−2	1e−2	2e−2	1e−2	1e−6
$N = 3$	5e−5	9e−4	1e−2	3e−2	5e−2	6e−2	6e−2	5e−3	−7e−6
$N = 4$	3e−4	4e−3	3e−2	4e−2	9e−2	9e−2	7e−2	3e−2	7e−2

Table 3.2: Relative gap $\overline{\rho}_{N,0}(q) - \underline{\rho}_{N,0}(q) / \overline{\rho}_{N,0}(q)$ for PRP where $N = 2, 3, 4$. The case $N = 1$ is omitted, as PRP is equivalent to GDEL in this case.

$q =$	0.001	0.005	0.02	0.04	0.06	0.08	0.1	0.3	0.5
$N = 2$	9e−6	2e−4	1e−3	7e−3	1e−2	1e−2	2e−2	1e−2	8e−7
$N = 3$	7e−5	1e−3	1e−2	2e−2	3e−2	3e−2	3e−2	3e−7	−1e−7
$N = 4$	2e−4	3e−3	2e−2	3e−2	3e−2	2e−2	1e−2	1e−6	4e−2

Table 3.3: The relative gap $\overline{\rho}_{N,0}(q) - \underline{\rho}_{N,0}(q) / \overline{\rho}_{N,0}(q)$ for FR where $N = 2, 3, 4$. The case $N = 1$ is omitted again, as in this case FR is equivalent to GDEL.

The next sections discuss and draw a few conclusions from the numerical worst-case convergence results for PRP and FR.

3.3.3 Improved worst-case bounds for PRP

Figure 3.4 reports the worst-case values of the "Lyapunov" ratio $f(x_{k+N}) - f_\star / f(x_k) - f_\star$ as a function of the inverse condition number $q \triangleq \mu/L$ and for $c = (1+q)^2/4q$ and $N = 1, 2, 3, 4$. This worst-case ratio seems to improve as N grows, but does not outperform gradient descent

with exact line search (GDEL). The diminishing improvements with N also suggests the worst-case performance of PRP in this regime might not outperform GDEL even for larger values of $N \geq 4$, albeit probably getting close to the same asymptotic worst-case convergence rate.

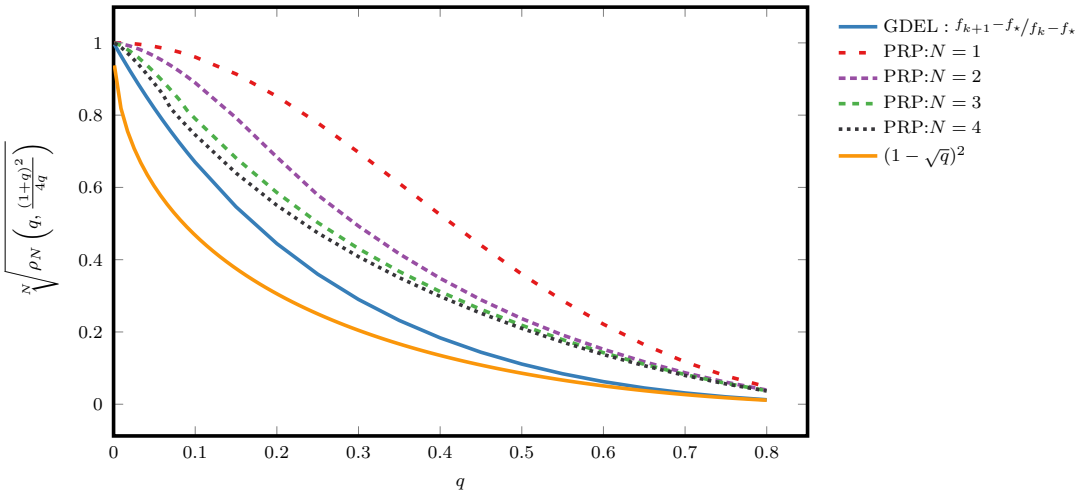


Figure 3.4: This figure reports the worst-case values for the "Lyapunov" ratio $\sqrt[N]{f(x_{k+N})-f_\star/f(x_k)-f_\star}$ vs. the condition number $q \triangleq \mu/L$ for PRP. We compute $\rho_N(q,c)$ with $c = (1+q)^2/4q$ for $N = 1,2,3,4$. As N increases, the worst-case $\sqrt[N]{f_{k+N}-f_\star/f_k-f_\star}$ improves, but remains worse than that of gradient descent with exact line search (GDEL). The curve $(1-\sqrt{q})^2$ (orange) corresponds to the rate of the lower complexity bounds for this class of problems [41].

As a complement, Figure 3.5 shows how PRP's worst-case ratio $f_N-f_\star/f_0-f_\star$ evolves as a function of q for $N = 1,2,3,4$. The worst-case performance of PRP in this setup seems to be similar to that of GDEL. Further, for small q (which is typically the only regime of interest for large-scale optimization), PRP's worst-case performance seems to be slightly better than that of GDEL. On the other hand, for larger q, PRP performs slightly worse than GDEL.

As a conclusion, we believe there is no hope to prove uniformly better worst-case bounds for PRP than those for GDEL for smooth strongly convex minimization. However, we might be able to prove improvements for small values of q at the cost of possibly very technical

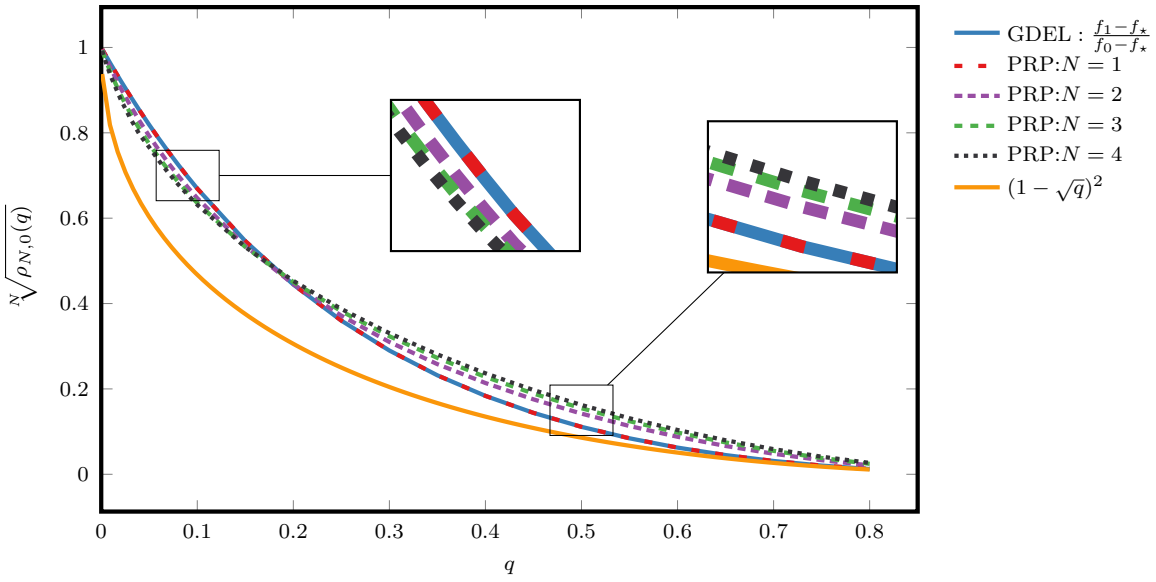


Figure 3.5: This figure reports the worst-case values for the ratio $\sqrt[N]{f_N-f_\star/f_0-f_\star}$ vs. q for PRP for $N=1,2,3,4$. For $N=1$, PRP and GDEL perform the same iteration. For $N=2,3,4$, the worst-case ratio of PRP is better than that of GDEL for $q \leq 0.1$. The curve $(1-\sqrt{q})^2$ (orange) corresponds to the rate of the lower complexity bounds for this class of problems [41].

proofs. As for the Lyapunov approach, the numerical results from this section could be improved by further increasing N, but we believe that the transient behavior does not suggest this direction to be promising. We recall that we computed the bounds by solving an optimization problem whose feasible points correspond to worst-case examples. Therefore, the numerical results provided in this section are backed-up by numerically constructed examples on which PRP behaves "badly".

3.3.4 Improved worst-case bounds for FR

Figure 3.6 reports the worst-case values for the ratio $f_N-f_\star/f_0-f_\star$ as a function of q, for $N \in \{1,2,3,4\}$. The convergence bounds appears to be marginally better than GDEL for some sufficiently small inverse condition numbers. Further, the range of values of q for which there is an improvement appears to be decreasing with $N \geq 2$. Beyond this range, the

worst-case values become significantly worse than that of GDEL. Though apparently not as dramatic as the worst-case bound from Theorem 3.5, the quality of the bound appears to be decreasing with N, which stands in line with the practical need to restart the method [110].

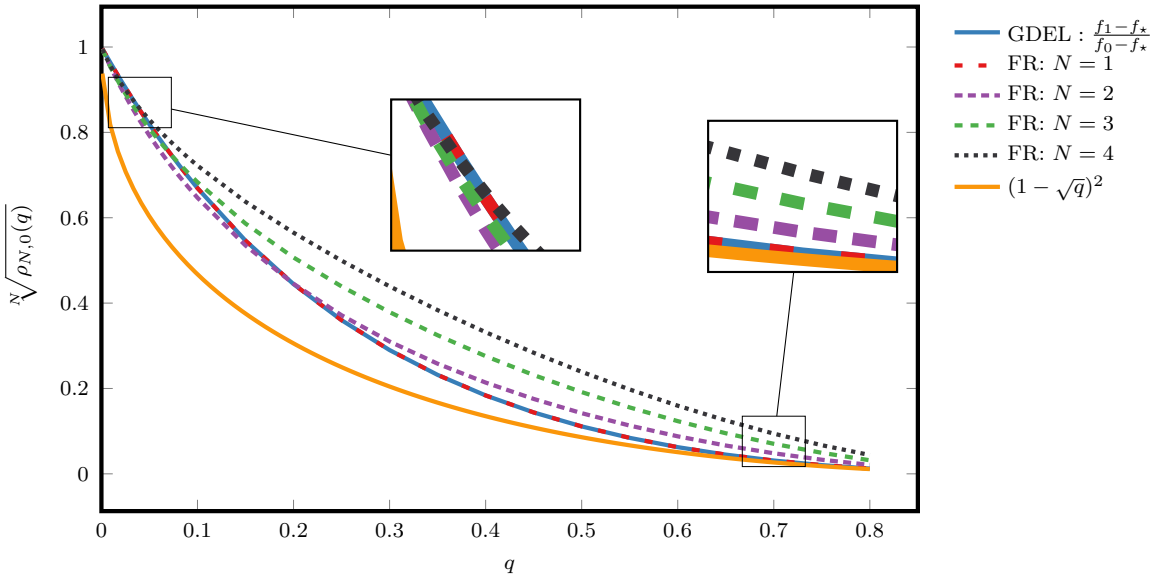


Figure 3.6: This figure reports the worst-case values for the ratio $\sqrt[N]{f_N-f_\star/f_0-f_\star}$ vs. q for FR for $N=1,2,3,4$. For $N=1$, FR and GDEL perform the same iteration. For $N=2,3,4$, the worst-case bound for FR is slightly better than that of GDEL for small enough values of q, and gets larger than GDEL for larger values of q. The range of q for which FR is better than GDEL gets smaller as $N \geq 2$ increases. The curve $(1-\sqrt{q})^2$ (orange) corresponds to the rate of the lower complexity bounds for this class of problems [41].

As in the previous section, we recall that those curves were obtained by numerically constructing "bad" worst-case counter-examples satisfying our assumptions. In other words, there is no hope to obtain better results without adding assumptions or changing the types of bounds under consideration.

3.4 Custom spatial branch-and-bound algorithm

This section discusses implementation details for solving the nonconvex QCQPs of this chapter (namely ($\mathcal{D}$), (3.28), or (3.34)) using a custom spatial branch-and-bound method. This strategy proceeds in three stages, as follows.

- **Stage 1: Compute a feasible solution.** First, we construct a feasible solution to the nonconvex QCQP. We do that by generating a random μ-strongly convex and L-smooth quadratic function, and by applying the corresponding nonlinear conjugate gradient method on it. The corresponding iterates, gradient and function values correspond to a feasible point for the nonconvex QCQPs under consideration.
- **Stage 2: Compute a locally optimal solution by warm-starting at Stage 1 solution.** Stage 2 computes a locally optimal solution to the nonconvex QCQPs using an interior-point algorithm, warm-starting at the feasible solution produced by Stage 1. When a good warm-starting point is provided, interior-point algorithms can quickly converge to a locally optimal solution under suitable regularity conditions [70, 78], [71, §3.3]. In the situation where the interior-point algorithm fails to converge, we go back to the feasible solution from Stage 1. We have empirically observed that Stage 2 consistently provides a locally optimal solution.
- **Stage 3: Compute a globally optimal solution by warm-starting at Stage 2 solution.** Stage 3 computes a globally optimal solution to the nonconvex QCQP using a spatial branch-and-bound algorithm [68, 81], warm-starting at the locally-optimal solution produced by Stage 2. For details about how spatial branch-and-bound algorithm works, we refer the reader to [1, §4.1].

Remark 3.2. *In stage 3, the most numerically challenging nonconvex quadratic constraint*

in ($\mathcal{D}$), (3.28) *or* (3.34) *is* $G = H^\top H$. *To solve those problems in reasonable times, we use the* lazy constraints *approach, [1, §4.2.5].*

In short, we replace the constraint $G = H^\top H$ *by the infinite set of linear constraints* $\mathbf{tr}\left(Gyy^\top\right) \geq 0$ *for all* y, *which we then sample to obtain a finite set of linear constraints (we recursively add additional linear constraints afterwards if need be). More precisely, we use*

$$\mathbf{tr}\left(Gyy^\top\right) \geq 0, \quad y \in Y, \tag{3.35}$$

where the initial Y *is generated randomly as a set of unit vectors following the methodology described in [88, §5.1]. By replacing* $G = H^\top H$ *by* (3.35) *we obtain a simpler (but relaxed) QCQP. Then, we update the solution* G *lazily by repeating the following steps until* $G \succeq 0$ *is satisfied subject to a termination criterion. Practically speaking, our termination criterion is that the minimal eigenvalue of* G *is larger than* $\epsilon \approx -1e-6$*; until then, we repeat the following procedure:*

1. *Solve the relaxation of the nonconvex QCQPs, where* (3.35) *is used instead of* $G = H^\top H$, *which provides us an upper bound on the original nonconvex QCQP.*
2. *Compute the minimal eigenvalue* $\mathrm{eig}_{\min}(G)$ *and the corresponding eigenvector* u *of* G. *If* $\mathrm{eig}_{\min}(G) \geq 0$, *we reached an optimal solution to the nonconvex QCQP and we terminate.*
3. *If* $\mathrm{eig}_{\min}(G) < 0$, *we add a constraint* $\mathbf{tr}(Guu^\top) \geq 0$ lazily, *which makes the current* G *infeasible for the new relaxation. We use the lazy constraint callback interface of* `JuMP` *to add constraints lazily, which means that after adding one additional linear constraint, updating the solution in step 1 is efficient since* `Gurobi` *and all modern solvers based on the simplex algorithm can quickly update a solution when only one linear constraint is added [89, pp. 205-207].*

3.5 Conclusion

This works studies the iteration complexity of two variants of nonlinear conjugate gradients, namely the Polak-Ribière-Polyak (PRP) and the Fletcher-Reeves (FR) methods. We provide novel complexity bounds for both those methods, and show that albeit unsatisfying, not much can a priori be gained from a worst-case perspective, as both methods appear to behave similar or worse to regular steepest descent in the worst-case. Further, those results suggest that explaining the good practical performances of NCGMs might be out of reach for traditional worst-case complexity analyses on classical classes of problems.

This work considers only somewhat "ideal" variants of nonlinear conjugate gradient methods, as we make explicit use of exact line search procedures. However, there is a priori no reason to believe that inexact line search procedures would improve the possibly bad worst-case behaviors. Further, the *performance estimation* methodology allows taking such inexact line search procedures into account, so the same methodology could be applied for tackling those questions. We leave such investigations for future work.

3.A Appendix for Chapter 3

3.A.1 Organization of the appendix

In what follows, we report detailed numerical results and computations that are not presented in the core of the chapter. Table 3.4 details the organization of this additional material.

Section	Content
Appendix 3.A.2	Numerical illustration of tightness of the worst-case search direction (3.19) for PRP and (3.21) for FR.
Appendix 3.A.3	Reformulation for weighted sum of inequalities for Lemmas 3.1, 3.2, 3.3
Appendix 3.A.4	Constructing counter-examples

Table 3.4: Organization of the appendix.

3.A.2 Tightness of the worst-case search directions

Figure 3.7 and Figure 3.8 illustrate the tightness of the bounds (3.19) and (3.21) for PRP and FR respectively. That is, we compare the absolute relative difference between the numerical bounds and closed-form bounds for a few different values of q and c_{k-1}, where numerical bounds are obtained by solving $(\mathcal{D})$ with $\eta = 1$ for PRP and $\eta = 0$ for FR. These numerical examples strongly suggest that our bounds cannot be improved in general.

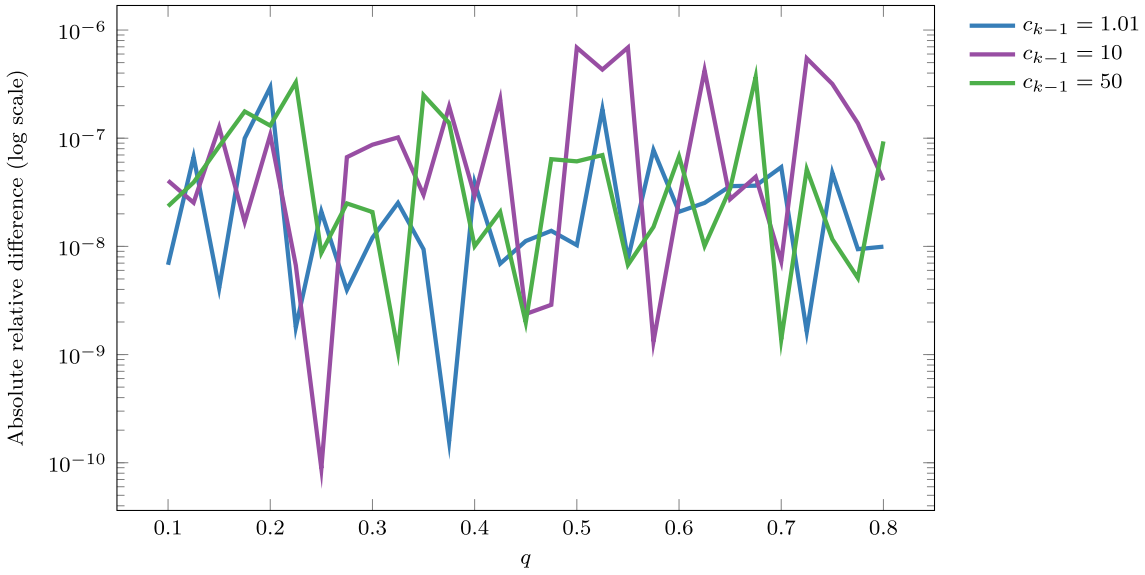


Figure 3.7: Absolute relative differences in the worst-case analytical bound (3.19) and numerical bounds from ($\mathcal{D}$) with $\eta = 1$ (for PRP) for different values of q and c_{k-1}.

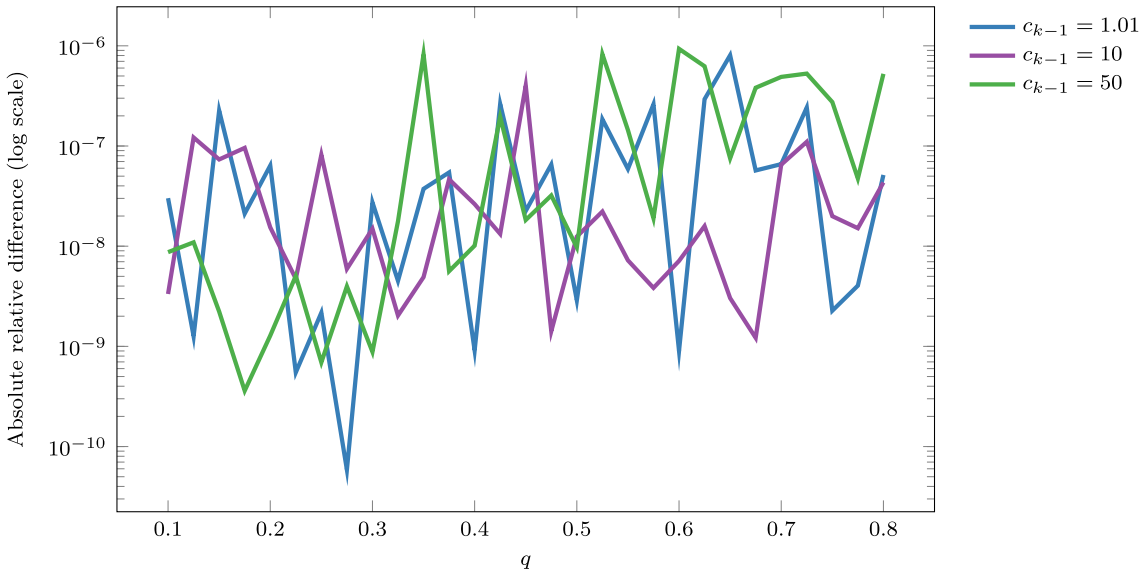


Figure 3.8: Absolute relative differences in the worst-case analytical bound (3.21) and numerical bounds from ($\mathcal{D}$) with $\eta = 0$ (for FR) for different values of q and c_{k-1}.

3.A.3 Reformulation for weighted sum of inequalities for Lemmas 3.1, 3.2, 3.3

3.A.3.1 Reformulation for weighted sum of inequalities for Lemma 3.1

For notational ease, define $f(x_k) \triangleq f_k$, $f(x_{k-1}) \triangleq f_{k-1}$,$\nabla f(x_k) \triangleq g_k$, $\nabla f(x_{k-1}) \triangleq g_{k-1}$, $\beta_{k-1} \triangleq \beta$, $\gamma_{k-1} \triangleq \gamma$, and $c_{k-1} \triangleq c$. We want to show that

$$
\begin{aligned}
&- \frac{\beta^2(q+1)}{\gamma L q}\langle g_k \mid d_{k-1}\rangle \\
&+ \frac{\beta^2(q+1)^2}{\gamma^2 L(1-q)q}\Bigg[f_k - f_{k-1} + \gamma\langle g_k \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1}-g_k\|^2 \\
&+ \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1}-g_k)\|^2\Bigg] \\
&+ \frac{\beta^2(q+1)^2}{\gamma^2 L(1-q)q}\Bigg[f_{k-1} - f_k - \gamma\langle g_{k-1} \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1}-g_k\|^2 \\
&+ \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1}-g_k)\|^2\Bigg] \\
&+ \frac{\beta(q+1)}{\gamma L q}\Big[\langle g_{k-1} \mid g_k\rangle - \|g_k\|^2 + \beta\langle g_{k-1} \mid d_{k-1}\rangle\Big]
\end{aligned} \tag{3.36}
$$

is equal to

$$
\begin{aligned}
&\|d_k\|^2 - \frac{(1+q)^2}{4q}\|g_k\|^2 \\
&+ \frac{4\beta^2 q}{(1-q)^2}\left\|d_{k-1} - \tfrac{1+q}{2L\gamma_{k-1}q}g_{k-1} + \tfrac{2\beta(1+q)-L\gamma(1-q)^2}{4\beta L\gamma q}g_k\right\|^2.
\end{aligned} \tag{3.37}
$$

We show this by expanding both terms and do term-by-term matching.

Expand the second summand of (3.36). First, we expand the second summand of (3.36) as follows.

$$
\begin{aligned}
&\frac{\beta^2(q+1)^2}{\gamma^2 L(1-q)q}\Bigg[f_k - f_{k-1} + \gamma\langle g_k \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1}-g_k\|^2 \\
&+ \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1}-g_k)\|^2\Bigg] \\
=&\frac{\beta^2(q+1)^2}{\gamma^2 L(1-q)q}\Bigg[f_k - f_{k-1} + \gamma\langle g_k \mid d_{k-1}\rangle + \tfrac{1}{2L}(\|g_{k-1}\|^2 - 2\langle g_k \mid g_{k-1}\rangle + \|g_k\|^2) \\
&+ \tfrac{\mu}{2(1-\mu/L)}\Bigg(\gamma^2\|d_{k-1}\|^2 + \frac{1}{L^2}\|g_{k-1}\|^2 + \frac{1}{L^2}\|g_k\|^2 - \frac{2}{L^2}\langle g_k \mid g_{k-1}\rangle \\
&+ \frac{2\gamma}{L}\langle d_{k-1} \mid g_k\rangle - \frac{2\gamma}{L}\langle d_{k-1} \mid g_{k-1}\rangle\Bigg)\Bigg] \\
=&\frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)}f_k - \frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)}f_{k-1} \\
&+ \frac{\beta^2(\mu+L)^2}{2(L-\mu)^2}\|d_{k-1}\|^2 + \frac{\beta^2(\mu+L)^2}{2\gamma^2\mu L(L-\mu)^2}\|g_{k-1}\|^2 + \frac{\beta^2(\mu+L)^2}{2\gamma^2\mu L(L-\mu)^2}\|g_k\|^2 \\
&- \frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)^2}\langle g_{k-1} \mid g_k\rangle + \frac{\beta^2(\mu+L)^2}{\gamma\mu(L-\mu)^2}\langle g_k \mid d_{k-1}\rangle - \frac{\beta^2(\mu+L)^2}{\gamma L(L-\mu)^2}\langle g_{k-1} \mid d_{k-1}\rangle\,, \quad (3.38)
\end{aligned}
$$

where on the second line we expand the squares and on the third line we collect the terms.

Expand the third summand of (3.36). Next, we expand the third summand of (3.36) as follows:

$$
\begin{aligned}
&\frac{\beta^2(q+1)^2}{\gamma^2 L(1-q)q}\Bigg[f_{k-1} - f_k - \gamma\langle g_{k-1} \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1}-g_k\|^2 \\
&+ \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1}-g_k)\|^2\Bigg] \\
=&\frac{\beta^2(q+1)^2}{\gamma^2 L(1-q)q}\Bigg[f_{k-1} - f_k - \gamma\langle g_{k-1} \mid d_{k-1}\rangle + \tfrac{1}{2L}(\|g_{k-1}\|^2 - 2\langle g_k \mid g_{k-1}\rangle + \|g_k\|^2) \\
&+ \tfrac{\mu}{2(1-\mu/L)}\Bigg(\gamma^2\|d_{k-1}\|^2 + \frac{1}{L^2}\|g_{k-1}\|^2 + \frac{1}{L^2}\|g_k\|^2 - \frac{2}{L^2}\langle g_k \mid g_{k-1}\rangle
\end{aligned}
$$

$$
\begin{aligned}
&+\frac{2\gamma}{L}\langle d_{k-1} \mid g_k\rangle - \frac{2\gamma}{L}\langle d_{k-1} \mid g_{k-1}\rangle\Big)\Big] \\
=&\frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)}f_{k-1} - \frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)}f_k \\
&+\frac{\beta^2(\mu+L)^2}{2(L-\mu)^2}\|d_{k-1}\|^2 + \frac{\beta^2(\mu+L)^2}{2\gamma^2\mu L(L-\mu)^2}\|g_{k-1}\|^2 + \frac{\beta^2(\mu+L)^2}{2\gamma^2\mu L(L-\mu)^2}\|g_k\|^2 \\
&-\frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)^2}\langle g_{k-1} \mid g_k\rangle + \frac{\beta^2(\mu+L)^2}{\gamma L(L-\mu)^2}\langle g_k \mid d_{k-1}\rangle - \frac{\beta^2(\mu+L)^2}{\gamma\mu(L-\mu)^2}\langle g_{k-1} \mid d_{k-1}\rangle \qquad (3.39)
\end{aligned}
$$

where again on the second line, we expand the squares and on the third line, we collect the terms.

Expanded form of (3.36). Now putting (3.38) and (3.39) in (3.36), and then collecting the terms, we arrive at the following expanded form of (3.36):

$$
\begin{aligned}
&\frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)^2}\|g_{k-1}\|^2 + \frac{\beta^2(\mu+L)^2}{(L-\mu)^2}\|d_{k-1}\|^2 \\
&+\frac{\beta(\mu+L)\left(\mu(\beta-\gamma\mu)-\gamma L^2+L(\beta+2\gamma\mu)\right)}{\gamma^2\mu L(L-\mu)^2}\|g_k\|^2 \\
&+\frac{\beta(\mu+L)\left(\mu(\gamma\mu-2\beta)+\gamma L^2-2L(\beta+\gamma\mu)\right)}{\gamma^2\mu L(L-\mu)^2}\langle g_{k-1} \mid g_k\rangle + \frac{4\beta^2(\mu+L)}{\gamma(L-\mu)^2}\langle g_k \mid d_{k-1}\rangle \\
&-\frac{4\beta^2(\mu+L)}{\gamma(L-\mu)^2}\langle g_{k-1} \mid d_{k-1}\rangle\,. \qquad (3.40)
\end{aligned}
$$

Expand the first two summands of (3.37). Now, we expand the first two summands of (3.37) as follows:

$$
\begin{aligned}
&\|d_k\|^2 - \frac{(1+(\mu/L))^2}{4(\mu/L)}\|g_k\|^2 \\
=&\|\beta d_{k-1}+g_k\|^2 - \frac{(1+(\mu/L))^2}{4(\mu/L)}\|g_k\|^2
\end{aligned}
$$

$$
\begin{aligned}
&= \beta^2 \|d_{k-1}\|^2 + \|g_k\|^2 + 2\beta \langle d_{k-1} \mid g_k \rangle - \frac{(1+(\mu/L))^2}{4(\mu/L)} \|g_k\|^2 \\
&= \beta^2 \|d_{k-1}\|^2 - \frac{(L-\mu)^2}{4\mu L} \|g_k\|^2 + 2\beta \langle d_{k-1} \mid g_k \rangle \,. \qquad (3.41)
\end{aligned}
$$

Expand the third summand of (3.37). Next, we expand the third summand of (3.37) as follows:

$$
\begin{aligned}
&\frac{4\beta^2 q}{(1-q)^2} \left\| d_{k-1} - \tfrac{1+q}{2L\gamma_{k-1}q} g_{k-1} + \tfrac{2\beta(1+q) - L\gamma(1-q)^2}{4\beta L\gamma q} g_k \right\|^2 \\
&= \frac{4\beta^2(\mu/L)}{(1-(\mu/L))^2} \Bigg[\|d_{k-1}\|^2 + \frac{\left(\mu(\gamma\mu - 2\beta) + \gamma L^2 - 2L(\beta+\gamma\mu)\right)^2}{16\beta^2\gamma^2\mu^2L^2} \|g_k\|^2 + \frac{(\mu+L)^2}{4\gamma^2\mu^2L^2} \|g_{k-1}\|^2 \\
&\quad + \frac{\left(\mu(2\beta - \gamma\mu) - \gamma L^2 + 2L(\beta+\gamma\mu)\right)}{2\beta\gamma\mu L} \langle g_k \mid d_{k-1} \rangle - \frac{(\mu+L)}{\gamma\mu L} \langle g_{k-1} \mid d_{k-1} \rangle \\
&\quad + \frac{(\mu+L)\left(\mu(\gamma\mu - 2\beta) + \gamma L^2 - 2L(\beta+\gamma\mu)\right)}{4\beta\gamma^2\mu^2L^2} \langle g_{k-1} \mid g_k \rangle \Bigg] \\
&= \frac{4\beta^2\mu L}{(L-\mu)^2} \|d_{k-1}\|^2 + \frac{\left(\mu(\gamma\mu - 2\beta) + \gamma L^2 - 2L(\beta+\gamma\mu)\right)^2}{4\gamma^2\mu L(L-\mu)^2} \|g_k\|^2 + \frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)^2} \|g_{k-1}\|^2 \\
&\quad + 2\beta \left(\frac{2\beta(\mu+L)}{\gamma(L-\mu)^2} - 1 \right) \langle g_k \mid d_{k-1} \rangle - \frac{4\beta^2(\mu+L)}{\gamma(L-\mu)^2} \langle g_{k-1} \mid d_{k-1} \rangle \\
&\quad + \frac{\beta(\mu+L)\left(\mu(\gamma\mu - 2\beta) + \gamma L^2 - 2L(\beta+\gamma\mu)\right)}{\gamma^2\mu L(L-\mu)^2} \langle g_{k-1} \mid g_k \rangle \,. \qquad (3.42)
\end{aligned}
$$

Expanded form of (3.37). Finally, putting the expanded expressions from (3.41) (3.42) in (3.37) and then collecting the terms, we get:

$$
\begin{aligned}
&\beta^2 \|d_{k-1}\|^2 - \frac{(L-\mu)^2}{4\mu L} \|g_k\|^2 + 2\beta \langle d_{k-1} \mid g_k \rangle \\
&+ \frac{4\beta^2\mu L}{(L-\mu)^2} \|d_{k-1}\|^2 + \frac{\left(\mu(\gamma\mu - 2\beta) + \gamma L^2 - 2L(\beta+\gamma\mu)\right)^2}{4\gamma^2\mu L(L-\mu)^2} \|g_k\|^2 + \frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)^2} \|g_{k-1}\|^2 \\
&+ 2\beta \left(\frac{2\beta(\mu+L)}{\gamma(L-\mu)^2} - 1 \right) \langle g_k \mid d_{k-1} \rangle - \frac{4\beta^2(\mu+L)}{\gamma(L-\mu)^2} \langle g_{k-1} \mid d_{k-1} \rangle \\
&+ \frac{\beta(\mu+L)\left(\mu(\gamma\mu - 2\beta) + \gamma L^2 - 2L(\beta+\gamma\mu)\right)}{\gamma^2\mu L(L-\mu)^2} \langle g_{k-1} \mid g_k \rangle
\end{aligned}
$$

$$
\begin{aligned}
=&\frac{\beta^2(\mu+L)^2}{\gamma^2\mu L(L-\mu)^2}\|g_{k-1}\|^2+\frac{\beta^2(\mu+L)^2}{(L-\mu)^2}\|d_{k-1}\|^2\\
&+\frac{\beta(\mu+L)\left(\mu(\beta-\gamma\mu)-\gamma L^2+L(\beta+2\gamma\mu)\right)}{\gamma^2\mu L(L-\mu)^2}\|g_k\|^2\\
&+\frac{\beta(\mu+L)\left(\mu(\gamma\mu-2\beta)+\gamma L^2-2L(\beta+\gamma\mu)\right)}{\gamma^2\mu L(L-\mu)^2}\langle g_{k-1}\mid g_k\rangle+\frac{4\beta^2(\mu+L)}{\gamma(L-\mu)^2}\langle g_k\mid d_{k-1}\rangle\\
&-\frac{4\beta^2(\mu+L)}{\gamma(L-\mu)^2}\langle g_{k-1}\mid d_{k-1}\rangle,
\end{aligned}
$$

where the last line is identical to (3.40).

The calculation shown above can also be independently verified using open-source symbolic computation libraries `SymPy` [136] and `Wolfram Language` [137] using the following notebooks available at

https://github.com/Shuvomoy/NCG-PEP-code/blob/main/Symbolic_Verifications/Verify_PRP.ipynb

and

https://github.com/Shuvomoy/NCG-PEP-code/blob/main/Symbolic_Verifications/Verify_PRP_Wolfram_Language.ipynb

, respectively.

3.A.3.2 Reformulation for weighted sum of inequalities for Lemma 3.2

For notational ease, define $f(x_k) \triangleq f_k$, $f(x_{k-1}) \triangleq f_{k-1}$,$\nabla f(x_k) \triangleq g_k$, $\nabla f(x_{k-1}) \triangleq g_{k-1}$, $\beta_{k-1} \triangleq \beta$, $\gamma_{k-1} \triangleq \gamma$, and $c_{k-1} \triangleq c$. We want to show that the weighted sum

$$
\begin{aligned}
&\left(\gamma(L+\mu) - \frac{2\sqrt{\beta}}{\sqrt{(c-1)c}}\right)\left[\langle g_{k-1} \mid d_{k-1}\rangle - \|g_{k-1}\|^2\right] \\
&+ \left(\frac{2}{c} - \gamma(L+\mu)\right)\left[\langle g_k \mid d_{k-1}\rangle\right] \\
&+ \left(\frac{\sqrt{c-1}}{\sqrt{\beta c}}\right)\left[\|g_k\|^2 - \beta\|g_{k-1}\|^2\right] \\
&+ \left(-\gamma^2 L\mu + \frac{\sqrt{\beta}}{c\sqrt{(c-1)c}}\right)\left[\|d_{k-1}\|^2 - c\|g_{k-1}\|^2\right] \\
&+ (L-\mu)\Big[f_k - f_{k-1} + \gamma\langle g_k \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1} - g_k\|^2 \\
&\qquad + \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1} - g_k)\|^2\Big] \\
&+ (L-\mu)\Big[f_{k-1} - f_k - \gamma\langle g_{k-1} \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1} - g_k\|^2 \\
&\qquad + \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1} - g_k)\|^2\Big]
\end{aligned} \tag{3.43}
$$

is equal to

$$
\begin{aligned}
&\|g_k\|^2 - \left(2\sqrt{1-\frac{1}{c}}\sqrt{\beta} - c\gamma^2 L\mu + \gamma(L+\mu) - 1\right)\|g_{k-1}\|^2 \\
&+ \left\|\sqrt[4]{\frac{\beta}{(c-1)c^3}}d_{k-1} - \sqrt[4]{\frac{\beta c}{c-1}}g_{k-1} + \sqrt[4]{\frac{c-1}{\beta c}}g_k\right\|^2
\end{aligned} \tag{3.44}
$$

We show this by expanding both terms and doing term-by-term matching.

Expand the fifth summand of (3.43). First, we expand the second summand of (3.43) as follows:

$$
\begin{aligned}
&(L-\mu)\left[f_k - f_{k-1} + \gamma\langle g_k \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1}-g_k\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1}-g_k)\|^2\right] \\
=&(L-\mu)\left[f_k - f_{k-1} + \gamma\langle g_k \mid d_{k-1}\rangle + \tfrac{1}{2L}\left(\|g_{k-1}\|^2 - 2\langle g_k \mid g_{k-1}\rangle + \|g_k\|^2\right)\right. \\
&+ \tfrac{\mu}{2(1-\mu/L)}\left(\gamma^2\|d_{k-1}\|^2 + \frac{1}{L^2}\|g_{k-1}\|^2 + \frac{1}{L^2}\|g_k\|^2 - \frac{2}{L^2}\langle g_k \mid g_{k-1}\rangle + \frac{2\gamma}{L}\langle d_{k-1} \mid g_k\rangle\right. \\
&\left.\left. - \frac{2\gamma}{L}\langle d_{k-1} \mid g_{k-1}\rangle\right)\right] \\
=&(L-\mu)f_k + (\mu-L)f_{k-1} + \frac{1}{2}\gamma^2\mu L\|d_{k-1}\|^2 + \frac{1}{2}\|g_{k-1}\|^2 + \frac{1}{2}\|g_k\|^2 \\
&- \langle g_{k-1} \mid g_k\rangle + \gamma L\langle g_k \mid d_{k-1}\rangle - \gamma\mu\langle g_{k-1} \mid d_{k-1}\rangle
\end{aligned} \tag{3.45}
$$

where on the second line, we expand the squares, and on the third line, we collect the terms.

Expand the sixth summand of (3.43). Next, we expand the sixth summand of (3.43) as follows:

$$
\begin{aligned}
&(L-\mu)\left[f_{k-1} - f_k - \gamma\langle g_{k-1} \mid d_{k-1}\rangle + \tfrac{1}{2L}\|g_{k-1}-g_k\|^2 + \tfrac{\mu}{2(1-\mu/L)}\|\gamma d_{k-1} - \tfrac{1}{L}(g_{k-1}-g_k)\|^2\right] \\
=&(L-\mu)\left[f_{k-1} - f_k - \gamma\langle g_{k-1} \mid d_{k-1}\rangle + \tfrac{1}{2L}\left(\|g_{k-1}\|^2 - 2\langle g_k \mid g_{k-1}\rangle + \|g_k\|^2\right)\right. \\
&+ \tfrac{\mu}{2(1-\mu/L)}\left(\gamma^2\|d_{k-1}\|^2 + \frac{1}{L^2}\|g_{k-1}\|^2 + \frac{1}{L^2}\|g_k\|^2 - \frac{2}{L^2}\langle g_k \mid g_{k-1}\rangle\right. \\
&\left.\left. + \frac{2\gamma}{L}\langle d_{k-1} \mid g_k\rangle - \frac{2\gamma}{L}\langle d_{k-1} \mid g_{k-1}\rangle\right)\right] \\
=&(L-\mu)f_{k-1} + (\mu-L)f_k + \frac{1}{2}\gamma^2\mu L\|d_{k-1}\|^2 + \frac{1}{2}\|g_{k-1}\|^2 + \frac{1}{2}\|g_k\|^2 \\
&- \langle g_{k-1} \mid g_k\rangle + \gamma\mu\langle g_k \mid d_{k-1}\rangle - \gamma L\langle g_{k-1} \mid d_{k-1}\rangle
\end{aligned} \tag{3.46}
$$

where again on the second line, we expand the squares and on the third line, we collect the terms.

Expanded form of (3.43). Now putting (3.45) and (3.46) in (3.43), and then collecting the terms, we arrive at the following expanded form of (3.43):

$$\frac{\sqrt{\beta}}{\sqrt{c-1}c^{3/2}}\|d_{k-1}\|^2+\left(\frac{\sqrt{c-1}}{\sqrt{c}\sqrt{\beta}}+1\right)\|g_k\|^2+\left(-\frac{\sqrt{\beta}(c-2)}{\sqrt{c-1}\sqrt{c}}+c\gamma^2\mu L-\gamma(\mu+L)+1\right)\|g_{k-1}\|^2$$
$$-2\langle g_{k-1}\mid g_k\rangle+\frac{2}{c}\langle g_k\mid d_{k-1}\rangle-\frac{2\sqrt{\beta}}{\sqrt{c-1}\sqrt{c}}\langle g_{k-1}\mid d_{k-1}\rangle \tag{3.47}$$

Expand the third summand of (3.44). Next, we expand the third summand of (3.44) as follows:

$$\begin{aligned}&\left\|\sqrt[4]{\frac{\beta}{(c-1)c}}d_{k-1}-\sqrt[4]{\frac{\beta c}{c-1}}g_{k-1}+\sqrt[4]{\frac{c-1}{\beta c}}g_k\right\|^2\\ =&\frac{\sqrt{\beta}}{\sqrt{c-1}c^{3/2}}\|d_{k-1}\|^2+\frac{\sqrt{\beta}\sqrt{c}}{\sqrt{c-1}}\|g_{k-1}\|^2+\frac{\sqrt{c-1}}{\sqrt{c}\sqrt{\beta}}\|g_k\|^2\\ &-2\langle g_{k-1}\mid g_k\rangle+\frac{2}{c}\langle g_k\mid d_{k-1}\rangle-2\frac{\sqrt{\beta}}{\sqrt{c-1}\sqrt{c}}\langle g_{k-1}\mid d_{k-1}\rangle\end{aligned} \tag{3.48}$$

Expanded form of (3.44). Finally, putting the expanded expressions from (3.47) in (3.44) and then collecting the terms, we get:

$$\|g_k\|^2-\left(2\sqrt{1-\frac{1}{c}}\sqrt{\beta}-c\gamma^2L\mu+\gamma(L+\mu)-1\right)\|g_{k-1}\|^2$$
$$+\left\|\sqrt[4]{\frac{\beta}{(c-1)c}}d_{k-1}-\sqrt[4]{\frac{\beta c}{c-1}}g_{k-1}+\sqrt[4]{\frac{c-1}{\beta c}}g_k\right\|^2$$

$$
\begin{aligned}
=&\|g_k\|^2 - \left(2\sqrt{1-\frac{1}{c}}\sqrt{\beta} - c\gamma^2 L\mu + \gamma(L+\mu) - 1\right)\|g_{k-1}\|^2 \\
&+ \frac{\sqrt{\beta}}{\sqrt{c-1}c^{3/2}}\|d_{k-1}\|^2 + \frac{\sqrt{\beta}\sqrt{c}}{\sqrt{c-1}}\|g_{k-1}\|^2 + \frac{\sqrt{c-1}}{\sqrt{c}\sqrt{\beta}}\|g_k\|^2 \\
&- 2\langle g_{k-1} \mid g_k\rangle + \frac{2}{c}\langle g_k \mid d_{k-1}\rangle - 2\frac{\sqrt{\beta}}{\sqrt{c-1}\sqrt{c}}\langle g_{k-1} \mid d_{k-1}\rangle \\
=&\left(\frac{\sqrt{c-1}}{\sqrt{c}\sqrt{\beta}} + 1\right)\|g_k\|^2 + \left(c\gamma^2 L\mu - \gamma(L+\mu) + 1 + \sqrt{\beta}\frac{\sqrt{c}}{\sqrt{c-1}} - 2\sqrt{\beta}\frac{\sqrt{c-1}}{\sqrt{c}}\right)\|g_{k-1}\|^2 \\
&+ \frac{\sqrt{\beta}}{\sqrt{c-1}c^{3/2}}\|d_{k-1}\|^2 - 2\langle g_{k-1} \mid g_k\rangle + \frac{2}{c}\langle g_k \mid d_{k-1}\rangle - 2\frac{\sqrt{\beta}}{\sqrt{c-1}\sqrt{c}}\langle g_{k-1} \mid d_{k-1}\rangle \\
=&\frac{\sqrt{\beta}}{\sqrt{c-1}c^{3/2}}\|d_{k-1}\|^2 + \left(\frac{\sqrt{c-1}}{\sqrt{c}\sqrt{\beta}} + 1\right)\|g_k\|^2 \\
&+ \left(-\frac{\sqrt{\beta}(c-2)}{\sqrt{c-1}\sqrt{c}} + c\gamma^2\mu L - \gamma(\mu+L) + 1\right)\|g_{k-1}\|^2 \\
&- 2\langle g_{k-1} \mid g_k\rangle + \frac{2}{c}\langle g_k \mid d_{k-1}\rangle - \frac{2\sqrt{\beta}}{\sqrt{c-1}\sqrt{c}}\langle g_{k-1} \mid d_{k-1}\rangle
\end{aligned}
$$

where the last line is identical to (3.47).

This symbolical calculation shown above can be independently verified using open-source symbolic computation libraries `SymPy` [136] and `Wolfram Language` [137] using the following notebooks available at

https://github.com/Shuvomoy/NCG-PEP-code/blob/main/Symbolic_Verifications/Verify_FR.ipynb

and

https://github.com/Shuvomoy/NCG-PEP-code/blob/main/Symbolic_Verifications/Verify_FR.ipynb

(in the cells titled `Lemma 2.2` of the notebooks), respectively.

3.A.3.3 Reformulation for weighted sum of inequalities for Lemma 3.3

For notational ease, define $\nabla f(x_k) \triangleq g_k$, $\nabla f(x_{k-1}) \triangleq g_{k-1}$, $\beta_{k-1} \triangleq \beta$, and $c_{k-1} \triangleq c$. The reformulation of the weighted sum is as follows:

$$
\begin{aligned}
&2\beta \langle d_{k-1} \mid g_k \rangle + \beta^2 \left(\|d_{k-1}\|^2 - c\|g_{k-1}\|^2 \right) - c\beta \left(\|g_k\|^2 - \beta \|g_{k-1}\|^2 \right) \\
=&2\beta \langle d_{k-1} \mid g_k \rangle + \beta^2 \|d_{k-1}\|^2 - \cancel{c\beta^2 \|g_{k-1}\|^2} - c\beta \|g_k\|^2 + \cancel{c\beta^2 \|g_{k-1}\|^2} \\
=&2\beta \langle d_{k-1} \mid g_k \rangle + \beta^2 \|d_{k-1}\|^2 - c\beta \|g_k\|^2 \\
=&2 \langle \beta d_{k-1} \mid g_k \rangle + \|\beta d_{k-1}\|^2 - c\beta \|g_k\|^2 \\
=&2 \langle d_k - g_k \mid g_k \rangle + \|d_k - g_k\|^2 - c\beta \|g_k\|^2 \\
=&\cancel{2 \langle d_k \mid g_k \rangle} - 2\|g_k\|^2 + \|d_k\|^2 - \cancel{2 \langle d_k \mid g_k \rangle} + \|g_k\|^2 - c\beta \|g_k\|^2 \\
=&\|d_k\|^2 - \|g_k\|^2 - c\beta \|g_k\|^2 \\
=&\|d_k\|^2 - (1 + c\beta) \|g_k\|^2,
\end{aligned}
$$

thus arriving at the simplified form used in the proof.

This symbolical calculation shown above can be independently verified using open-source symbolic computation libraries `SymPy` [136] and `Wolfram Language` [137] using the following notebooks available at

https://github.com/Shuvomoy/NCG-PEP-code/blob/main/Symbolic_Verifications/Verify_FR.ipynb

and

https://github.com/Shuvomoy/NCG-PEP-code/blob/main/Symbolic_Verifications/Verify_FR.ipynb

(in the cells titled `Lemma 2.3` of the notebooks), respectively.

3.A.4 Constructing counter-examples

Once we have solved the nonconvex QCQPs associated with ($\mathcal{B}_{\text{Lyapunov}}$) or ($\mathcal{B}_{\text{exact}}$), we can construct the associated triplets $\{x_i, g_i, f_i\}_{i\in I_N^\star}$ and then apply Theorem 3.2 to construct the corresponding "bad" function. This "bad" function serves as a counter-example, illustrating scenarios where ($\mathcal{M}$) performs poorly. One can access the numerically constructed triplets $\{x_i, g_i, f_i\}_{i\in I_N^\star}$ associated with the counter-examples by following the instructions provided in our github repository. Next, we provide a concrete example of how to construct a "bad" function for ($\mathcal{B}_{\text{Lyapunov}}$) from our provided code and datasets located in the folder titled '`Code_for_NCG_PEP`' of the github repository. Constructing counter-examples for ($\mathcal{B}_{\text{exact}}$) is analogous.

$x_\star$	x_0	x_1	x_2
$\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}1.67262\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}0.354109\\-0.810313\\0.0775561\\0.000222477\end{bmatrix}$	$\begin{bmatrix}-0.140817\\-0.322955\\-0.138845\\0.000244795\end{bmatrix}$

Table 3.5: Numerical values of $\{x_i\}_{i\in\{\star,0,1,2\}}$ for constructing the counter-example of Example 3.1.

$g_\star$	g_0	g_1	g_2
$\begin{bmatrix}0\\0\\0\\0\end{bmatrix}$	$\begin{bmatrix}1.08734\\0.237212\\0\\0\end{bmatrix}$	$\begin{bmatrix}0.303362\\-0.47567\\0.187527\\0\end{bmatrix}$	$\begin{bmatrix}-0.158567\\-0.205519\\-0.100196\\0.000166564\end{bmatrix}$

Table 3.6: Numerical values of $\{g_i\}_{i\in\{\star,0,1,2\}}$ for constructing the counter-example of Example 3.1.

Example 3.1 (How to construct counter-examples for ($\mathcal{B}_{\text{Lyapunov}}$))**.** *Suppose we are inter-*

$f_\star$	f_0	f_1	f_2
0	1	0.267353	0.056104

Table 3.7: Numerical values of $\{f_i\}_{i\in\{\star,0,1,2\}}$ for constructing the counter-example of Example 3.1.

ested in constructing a "bad" function aka counter-example for the worst-case bound on $f(x_{k+2})-f_\star/f(x_k)-f_\star$ *(steps for other values of* N *are identical) for PRP with* $q \triangleq \mu/L = 0.5$*. The resultant "bad" function from* $\mathbb{R}^4$ *to* $\mathbb{R}$ *is completely characterized by the triplets* $\{x_i, g_i, f_i\}_{i\in\{\star,0,1,2\}}$*, where the triplets can be generated or accessed in two ways:*

1. *we can run '*`1.Example_Julia.ipynb`*' with the input parameters and generate the function by solving the nonconvex QCQP directly and generate the triplets, or*
2. *we can directly access the triplets from the saved datasets in the folder* `Saved_Output_Files` *with instructions provided in the file '*`2.Using_the_saved_datasets_Julia.ipynb`*'.*

For the sake of completeness, we provide the numerical values of $\{x_i\}_{i\in\{\star,0,1,2\}}$, $\{g_i\}_{i\in\{\star,0,1,2\}}$, *and* $\{f_i\}_{i\in\{\star,0,1,2\}}$ *of the function in this setup in Table 3.5, Table 3.6, and Table 3.7, respectively. From the numerical values of the triplets, we can construct the "bad" function using Theorem 3.2. For this constructed function, we have the performance guarantee* $f(x_{k+2})-f_\star/f(x_k)-f_\star \geq 0.056104$*, which closely matches the bound provided in Figure 3.4. Additionally, this guarantee can be verified through other existing open-source software [126, 138]; we provide code for this independent verification in the file called*

'`3.PEPIt_verification_Python.ipynb`*'.*

Chapter 4

Exterior-point Optimization for Sparse and Low-rank Optimization

4.1 Introduction

This chapter studies optimization problems involving a strongly convex and smooth cost function over a closed nonconvex constraint set $\mathcal{X}$ involving sparse or low-rank constraints. We propose a first-order algorithm *nonconvex exterior-point optimization solver* (NExOS) to solve such problems numerically. We can write such problems as:

$$\begin{array}{ll} \text{minimize} & f(x) + (\beta/2)\|x\|^2 \\ \text{subject to} & x \in \mathcal{X}, \end{array} \tag{$\mathcal{P}$}$$

where x takes value in a finite-dimensional vector space $\mathbb{E}$ over the reals, f is a strongly convex and smooth function. In Appendix 4.A.2.1, we generalize our framework to the case when f is non-smooth convex.

The regularization parameter $\beta > 0$ is commonly introduced in statistics and machine learning problems to reduce the generalization error without increasing the training error [139, §5.2.2]. In this chapter, there is also a theoretical consideration behind including the term $\frac{\beta}{2}\|x\|^2$ in problem ($\mathcal{P}$). NExOS finds a locally optimal point of problem ($\mathcal{P}$) by solving a sequence of penalized subproblems with strictly decreasing penalty parameters, where each penalized subproblem is solved by a first-order algorithm. Under the presence of $\frac{\beta}{2}\|x\|^2$ with $\beta > 0$, we can prove that each penalized subproblem is locally strongly convex and smooth admitting a unique local minimum (see Proposition 4.1), which in turn ensure linear convergence of the first-order method to that local minimum (see Theorem 4.1). In the Numerical Experiments section, we demonstrate that β can be set to a value as small as 10^{-8}. This empirical evidence suggests that, in practice, the impact of β on the objective value can be made negligible, yet one can still reap the theoretical benefits. Therefore, while β plays a crucial role in the theoretical aspects of our algorithm, its influence on the problems considered in the Numerical Experiments section is minimal and can be adjusted as per the problem's requirements.

Furthermore, $\mathbb{E}$ is equipped with inner product $\langle \cdot \mid \cdot \rangle$ and norm $\|\cdot\| = \sqrt{\langle x \mid x \rangle}$. For $\mathbb{E} = \mathbf{R}^d$, we have $\langle x \mid y \rangle = x^\top y$ for $x, y \in \mathbf{R}^d$, and for $\mathbb{E} = \mathbf{R}^{m\times n}$, we have $\langle X \mid Y \rangle = \mathbf{tr}(X^\top Y)$, for $X, Y \in \mathbf{R}^{m\times n}$. The constraint set $\mathcal{X}$ is closed and nonconvex and can be decomposed as the intersection of a compact convex set and a nonconvex set involving sparse or low-rank constraints. Sparse and low-rank constraint sets are very important in modeling many machine learning problems, because they allow for high interpretability, speed-ups in computation, and reduced memory requirements [18].

Sparsity-constrained optimization. Sparsity constraints have found applications in many practical settings, *e.g.*, gene expression analysis [17, pp. 2–4], sparse regression [18, pp. 155–157], signal transmission and recovery [19, 20], hierarchical sparse polynomial regression

[21], and best subset selection [22], just to name a few. In these problems, the constraint set $\mathcal{X}$ decomposes as $\mathcal{X} = \mathcal{C} \bigcap \mathcal{N}$, where $\mathcal{C}$ is a compact convex set, and

$$\mathcal{N} = \{x \in \mathbb{R}^d \mid \mathbf{card}(x) \leq k\}, \tag{4.1}$$

where $\mathbf{card}(x)$ counts the number of nonzero elements in x. In these optimization problems, $\mathcal{C}$ can be a polyhedron, infinity-norm ball, box constraint set, or probability simplex; these sets usually show up in applications involving econometrics, housing price prediction, air-quality prediction, signal processing, and meteorology [140–143].

In this chapter, we apply NExOS to solve the sparse regression problem for both synthetic and real-world datasets in §4.4.1, which is concerned with approximating a vector $b \in \mathbf{R}^m$ with a linear combination of at most k columns of a matrix $A \in \mathbf{R}^{m \times d}$ with bounded coefficients. This problem has the form:

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2^2 + (\beta/2)\|x\|_2^2 \\ \text{subject to} & \mathbf{card}(x) \leq k, \quad \|x\|_\infty \leq \Gamma, \end{array} \tag{SR}$$

where $x \in \mathbf{R}^d$ is the decision variable, and $A \in \mathbf{R}^{m \times d}$, $b \in \mathbf{R}^m$, and $\Gamma > 0$ are problem data.

Low-rank optimization. We can write low-rank optimization problems in the form of problem ($\mathcal{P}$), which are common in machine learning applications such as collaborative filtering [18, pp. 279-281], design of online recommendation systems [23, 24], bandit optimization [25], data compression [26–28], and low rank kernel learning [29]. In these applications, the constraint set $\mathcal{X}$ decomposes as $\mathcal{X} = \mathcal{C} \bigcap \mathcal{N}$, where $\mathcal{C}$ is a compact convex set, and

$$\mathcal{N} = \{X \in \mathbb{R}^{m \times d} \mid \mathbf{rank}(X) \leq r\}. \tag{4.2}$$

In these optimization problems, $\mathcal{C}$ can be matrix-norm ball, Frobenius-norm ball, hyperplane/half-space induced by trace [144, 145]. In this chapter, we apply NExOSto solve the affine rank minimization problem:

$$\begin{array}{ll} \text{minimize} & \|\mathcal{A}(X) - b\|_2^2 + (\beta/2)\|X\|_F^2 \\ \text{subject to} & \mathbf{rank}(X) \leq r, \quad \|X\|_2 \leq \Gamma, \end{array} \tag{RM}$$

where $X \in \mathbf{R}^{m\times d}$ is the decision variable, $b \in \mathbf{R}^k$ is noisy measurement data, and $\mathcal{A} : \mathbf{R}^{m\times d} \to \mathbf{R}^k$ is a linear map. The parameter $\Gamma > 0$ is the upper bound for the spectral norm of X. The affine map $\mathcal{A}$ is determined by k matrices $A_1, \ldots, A_k$ in $\mathbf{R}^{m\times d}$ where $\mathcal{A}(X) = (\mathbf{tr}(A_1^T X), \ldots, \mathbf{tr}(A_k^T X))$. We present several numerical experiments to solve (RM) using NExOSfor both synthetic and real-world datasets in §4.4.2.

4.1.1 Related work

Convex relaxation approach. Due to the presence of the nonconvex set $\mathcal{X}$, the nonconvex problem ($\mathcal{P}$) is $\mathcal{NP}$-hard [146]. A common way to deal with this issue is to avoid this inherent nonconvexity altogether by convexifying the original problem. The relaxation of the sparsity constraint leads to the popular LASSO formulation and its variants [17], whereas relaxation of the low-rank constraints produces the nuclear norm based convex models [147].

The basic advantage of the convex relaxation technique is that, in general, a globally optimal solution to a convex problem can be computed reliably and efficiently [58, §1.1], whereas for nonconvex problems a local optimal solution is often the best one can hope for. Furthermore, if certain statistical assumptions on the data generating process hold, then it is possible to recover exact solutions to the original nonconvex problems with high probability by solving the convex relaxations (see [17] and the references therein).

However, when stringent assumptions do not hold, then solutions to the convex formulations can be of poor quality and may not scale very well [18, §6.3 and §7.8]. In this situation, the nonconvexity of the original problem must be confronted directly, because such nonconvex formulations capture the underlying problem structures more accurately than their convex counterparts.

First-order methods. To that goal, first-order algorithms such as hard thresholding algorithms, *e.g.*, IHT [148], NIHT [149], HTP [150], CGIHT [151], address nonconvexity in sparse and low-rank optimization by implementing variants of projected gradient descent with projection taken onto the sparse and/or low-rank set.

While these first-order methods have been successful in recovering low-rank and sparse solutions in underdetermined linear systems, they too require assumptions on the data such as the *restricted isometry property* for recovering true solutions [18, §7.5]. Furthermore, to converge to a local minimum, hard thresholding algorithms require the spectral norm of the measurement matrix to be less than one, which is a restrictive condition [148].

Besides hard thresholding algorithms, heuristics based on first-order algorithms such as the alternating direction method of multipliers ADMM have gained a lot of traction in the last few years. Though ADMM was originally designed to solve convex optimization problems, since the idea of implementing this algorithm as a general purpose heuristic to solve nonconvex optimization problems was introduced in [152, §9.1-9.2], ADMM-based heuristics have been applied successfully to approximately solve nonconvex problems in many different application areas [141, 153].

However, the biggest drawback of these heuristics based on first-order methods comes from the fact that they take an algorithm designed to solve convex problems and apply it verbatim

to a nonconvex setup. As a result, these algorithms often fail to converge, and even when they do, it need not be a local minimum, let alone a global one [154, §2.2]. Also, empirical evidence suggests that the iterates of these algorithms may diverge even if they come arbitrarily close to a locally optimal solution during some iteration. The main reason is that these heuristics do not establish a clear relationship between the local minimum of problem ($\mathcal{P}$) and the fixed point set of the underlying operator that controls the iteration scheme. An alternative approach that has been quite successful empirically in finding low-rank solutions is to consider an unconstrained problem with Frobenius norm penalty and then using alternating minimization to compute a solution [155]. However, the alternating minimization approach may not converge to a solution and should be considered a heuristic [155, §2.4].

Discrete optimization approach. For these reasons above, in the last few years, there has been significant interest in addressing the nonconvexity present in many optimization problems directly via a discrete optimization approach. In this way, a particular nonconvex optimization problem is formulated exactly using discrete optimization techniques and then specialized algorithms are developed to find a certifiably optimal solution. This approach has found considerable success in solving machine learning problems with sparse and low-rank optimization [67, 156]. A mixed integer optimization approach to compute near-optimal solutions for sparse regression problem, where problem dimension $d = 1000$, is computed in [22]. In [142], the authors propose a cutting plane method for a similar problem, which works well with mild sample correlations and a sufficiently large dimension. In [157], the authors design and implement fast algorithms based on coordinate descent and local combinatorial optimization to solve sparse regression problem with a three-fold speedup where $d \approx 10^6$. In [144], the authors propose a framework for modeling and solving low-rank optimization problems to certifiable optimality via symmetric projection matrices.

However, the runtime of these discrete optimization based algorithms can often become prohibitively long as the problem dimensions grow [22]. Also, these discrete optimization algorithms have efficient implementations only for a narrow class of loss functions and constraint sets; they do not generalize well if a minor modification is made to the problem structure, and in such a case they often fail to find a solution point in a reasonable amount of time even for smaller dimensions [67]. Furthermore, one often relies on commercial softwares, such as `Gurobi`, `Mosek`, or `Cplex` to solve these discrete optimization problems, thus making the solution process somewhat opaque [22, 156].

4.1.2 Contributions

The main contribution of this work is to propose NExOS: a first-order algorithm tailored for nonconvex optimization problems of the form ($\mathcal{P}$). The term *exterior-point* originates from the fact that the iterates approach a local minimum from outside of the feasible region; it is inspired by the convex exterior-point method first proposed by Fiacco and McCormick in the 1960s [71, §4]. By exploiting the underlying geometry of the constraint set, we construct an iterative method that finds a locally optimal point of the original problem via an outer loop consisting of increasingly accurate penalized formulations of the original problem by reducing only one penalty parameter. Each penalized problem is then solved by applying an inner algorithm that implements a variant of the Douglas-Rachford splitting algorithm.

We prove that NExOS, besides avoiding the drawbacks of convex relaxation and discrete optimization approach, has the following favorable features. First, the penalized problem has strong convexity and smoothness around local minima, but can be made arbitrarily close to the original nonconvex problem by reducing the penalty parameter. Second, under mild regularity conditions, the inner algorithm finds local minima for the penalized problems at a linear convergence rate, and as the penalty parameter goes to zero, the local minima of

the penalized problems converge to a local minimum of the original problem. Furthermore, we show that, when those regularity conditions do not hold, the inner algorithm is still guaranteed to subsequentially converge to a first-order stationary point of the penalized problem at the rate $o(1/\sqrt{k})$.

We implement NExOS in the open-source `Julia` package `NExOS.jl` and test it extensively on many synthetic and real-world instances of different nonconvex optimization problems of substantial current interest. We demonstrate that NExOS very quickly computes solutions that are competitive with or better than specialized algorithms on various performance measures. `NExOS.jl` is available at https://github.com/Shuvomoy/NExOS.jl.

Organization of the chapter. The rest of the chapter is organized as follows. We describe our NExOS framework in §4.2. We provide convergence analysis of the algorithm in §4.3. Then we demonstrate the performance of our algorithm on several nonconvex optimization problems of significant current interest in §4.4. The concluding remarks are presented in §4.5.

4.2 Our approach

The backbone of our approach is to address the nonconvexity by working with an asymptotically exact nonconvex penalization of problem ($\mathcal{P}$), which enjoys local convexity around local minima. We use the notation $\iota_{\mathcal{X}}(x)$ that denotes the indicator function of the set $\mathcal{X}$ at x, which is 0 if $x \in \mathcal{X}$ and ∞ else. Using this, we can write problem ($\mathcal{P}$) as an unconstrained optimization problem, where the objective is $f(x) + (\beta/2)\|x\|^2 + \iota_{\mathcal{X}}(x)$. In our penalization, we replace the indicator function ι with its *Moreau envelope* with positive parameter μ:

$$^{\mu}\iota(x) = \min_{y}\{\iota(y) + (1/2\mu)\|y - x\|^2\} = (1/2\mu)d^2(x), \tag{4.3}$$

where $d(x)$ is the Euclidean distance of the point x from the set $\mathcal{X}$.

Properties of Moreau envelope for a nonconvex set. The function ${}^{\mu}\iota$, though nonconvex, has many desirable attributes that greatly simplify design and convergence analysis of our algorithm. We summarize these properties below; See [158, Proposition 12.9] for the first four properties, and Proposition 4.1 in §4.3 for the last one.

1. **Bounded.** The function ${}^{\mu}\iota$ is bounded on every compact set. In contrast, ι is an extended valued function that takes the value $+\infty$ outside the set $\mathcal{X}$.
2. **Finite and jointly continuous.** For every $\mu > 0$ and $x \in \mathbb{E}$, the function ${}^{\mu}\iota(x)$ is jointly continuous and finite. Therefore, ${}^{\mu}\iota$ is continuous on $\mathbb{E}$. In contrast, the indicator function ι is not continuous.
3. **Accuracy of approximation controlled by μ.** With decreasing μ, the approximation ${}^{\mu}\iota$ monotonically increases to ι, *i.e.*, for any positive μ_1, μ_2 such that $0 \leq \mu_1 \leq \mu_2$, we have

$$0 \leq {}^{\mu_2}\iota(x) \leq {}^{\mu_1}\iota(x) \leq \iota(x)$$

   for any $x \in \mathbf{E}$.
4. **Asymptotically equal to ι.** The approximation ${}^{\mu}\iota$ is asymptotically equal to ι as μ goes to zero, *i.e.*, we have the point-wise limit

$$\lim_{\mu \downarrow 0} {}^{\mu}\iota(x) = \iota(x)$$

   for all $x \in E$.
5. **Local convexity and differentiability around points of interest.** Adding any quadratic regularizer to ${}^{\mu}\iota$ makes the sum locally convex and differentiable around points of interest. To be precise, if at x, the set $\mathcal{X}$ is prox-regular, then for any value of

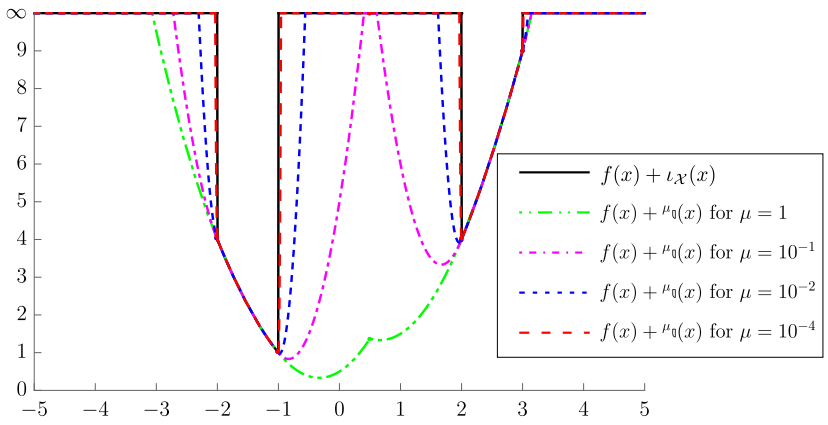


Figure 4.1: An illustration of how the penalized cost function in problem ($\mathcal{P}_\mu$) compares against the original cost function in problem ($\mathcal{P}$) for different values of μ. Note that the regularization parameter β is kept fixed at its initial value 1 throughout.

$\beta > 0$, the function ${}^{\mu}\iota(x) + \frac{\beta}{2}\|x\|^2$ is convex and differentiable on a neighborhood of x.

The favorable features of ${}^{\mu}\iota$ motivate us to consider the following penalization formulation of problem ($\mathcal{P}$) denoted by problem ($\mathcal{P}_\mu$), where the subscript μ indicates the penalty parameter:

$$\text{minimize} \quad f(x) + {}^{\mu}\mathbb{1}(x), \tag{$\mathcal{P}_\mu$}$$

where ${}^{\mu}\mathbb{1} \equiv {}^{\mu}\iota + (\beta/2)\|\cdot\|^2$, $x \in \mathbb{E}$ is the decision variable, and μ is a positive *penalty parameter*. We call the cost function in problem ($\mathcal{P}_\mu$) an *exterior-point minimization function*; the term is inspired by [71, §4.1]. The notation ${}^{\mu}\mathbb{1} \equiv {}^{\mu}\iota + (\beta/2)\|\cdot\|^2$ introduced in problem ($\mathcal{P}_\mu$) not only reduces notational clutter, but also alludes to a specific way of splitting the objective into two summands f and ${}^{\mu}\mathbb{1}$, which will ultimately allow us to establish convergence of our algorithm in §4.3. Because ${}^{\mu}\iota$ is an asymptotically exact approximation of $\iota_{\mathcal{X}}$ as $\mu \to 0$, solving problem ($\mathcal{P}_\mu$) for a small enough value of the penalty parameter μ suffices for all practical purposes.

To provide intuition on how the exterior-point minimization function in problem ($\mathcal{P}_\mu$) com-

Algorithm 2 Nonconvex Exterior-point Optimization Solver (NExOS). Here $\mathbf{\Pi}(x)$ denotes the Euclidean projection of x on the nonconvex set $\mathcal{X}$.

given: regularization parameter $\beta > 0$, an initial point z_{init}, initial penalty parameter μ_{init}, minimum penalty parameter $\mu_{\min}$, tolerance for the fixed point gap ϵ for each inner iteration, tolerance for stopping criterion δ for the outer iteration, and multiplicative factor $\rho \in (0, 1)$.

Initialization. $\mu := \mu_{\text{init}}$, *and* $z^0 := z_{\text{init}}$. *Outer iteration.* **while** *stopping criterion is not met* **do**

> *Inner iteration.* Using Algorithm 3, compute x_μ, y_μ, and z_μ that solve problem $(\mathcal{P}_\mu)$ with tolerance ϵ, where $z_\mu^0 := z^0$ is input as the initial point. *Stopping criterion.* **quit** *if* $|(f(\mathbf{\Pi}\, x_\mu) + (\beta/2)\|\mathbf{\Pi}_{\mathcal{X}} x_\mu\|^2) - (f(x_\mu) + {}^{\mu}\mathbb{1}(x_\mu))| \leq \delta$.
> *Set initial point for next inner iteration.* $z^0 := z_\mu$. *Update* μ. $\mu := \rho\mu$.

end

return x_μ, y_μ, and z_μ

pares against the original minimization function in problem $(\mathcal{P})$, we provide an illustrative one-dimensional example in Figure 4.1. Figure 4.1 captures all the key properties of our penalization scheme. In this figure, $f = (1/2)(\cdot)^2$, $\beta = 1$, $\mathcal{X} = [-2, -1] \bigcup [2, 3]$. The problem has two local minima, one at -1 and one at -2. We see that for larger values of μ, problem $(\mathcal{P}_\mu)$ is not a good approximation of problem $(\mathcal{P})$, but around each local minimum there is a relatively large region where $f + {}^{\mu}\mathbb{1}$ is strongly convex and smooth. As μ gets smaller, problem $(\mathcal{P}_\mu)$ becomes a more accurate approximation of problem $(\mathcal{P})$, though the regions of convexity and smoothness around local minima shrink. For $\mu = 10^{-4}$, problem $(\mathcal{P}_\mu)$ is identical to problem $(\mathcal{P})$ for all practical purposes. Note that the regularization parameter β is kept fixed at its initial value 1 throughout.

Now that we have intuitively justified intuition behind working with $(\mathcal{P}_\mu)$, we are in a position to present our algorithm.

Algorithm description. Algorithm 2 outlines NExOS. The main part is an outer loop that solves a sequence of penalized problems of the form problem $(\mathcal{P}_\mu)$ with strictly decreasing penalty parameter μ, until the termination criterion is met, at which point the exterior-point

Algorithm 3 Inner Algorithm for problem $(\mathcal{P}_\mu)$. Here $\mathbf{\Pi}(x)$ denotes the Euclidean projection of x on the nonconvex set $\mathcal{X}$, and $\mathbf{prox}_{\gamma f}$ denotes the proximal operator of f with parameter $\gamma > 0$ as defined in (4.4).

given: starting point z^0, tolerance for the fixed point gap ϵ, and proximal parameter $\gamma > 0$.
Initialization. $n := 0$, $\kappa := 1/(\beta\gamma + 1)$, $\theta := \mu/(\gamma\kappa + \mu)$.
while $\|x^n - y^n\| > \epsilon$ **do**
  Compute $x^{n+1} := \mathbf{prox}_{\gamma f}(z^n)$.
  Compute $\tilde{y}^{n+1} := \kappa(2x^{n+1} - z^n)$.
  Compute $y^{n+1} := \theta\tilde{y}^{n+1} + (1-\theta)\mathbf{\Pi}(\tilde{y}^{n+1})$.
  Compute $z^{n+1} := z^n + y^{n+1} - x^{n+1}$.
  Update $n := n + 1$.
end
return x^n, y^n, and z^n.

minimization function is a sufficiently close approximation of the original cost function. For each μ, problem $(\mathcal{P}_\mu)$ is solved by an inner algorithm, denoted by Algorithm 3.

One can derive Algorithm 3 by applying Douglas-Rachford splitting (DRS) [158, page 401] to problem $(\mathcal{P}_\mu)$ as follows. If we apply Douglas-Rachford splitting [158, page 401] to problem $(\mathcal{P}_\mu)$ with penalty parameter μ, we have the following variant with three sub-iterations:

$$\begin{aligned} x^{n+1} &= \mathbf{prox}_{\gamma f}(z^n) \\ y^{n+1} &= \mathbf{prox}_{\gamma \mu_{\mathbb{Q}}}\left(2x^{n+1} - z^n\right) \\ z^{n+1} &= z^n + y^{n+1} - x^{n+1}. \end{aligned} \tag{DRS}$$

The computational cost for $\mathbf{prox}_{\gamma \mu_{\mathbb{Q}}}$ is the same as computing a projection onto the constraint set $\mathcal{X}$, as stated in Lemma 4.1 below; this result follows from [159, Theorem 6.13, Theorem 6.63]. It should be noted that [159, Theorem 6.13, Theorem 6.63] assume convexity of the functions in the theorem statements, but its proof does not require convexity and works for nonconvex functions as well.

Lemma 4.1 (Computing $\mathbf{prox}_{\gamma \mu_{\mathbb{Q}}}(x)$)**.** *Consider the nonconvex compact constraint set $\mathcal{X}$ in problem (P). Denote $\kappa = 1/(\beta\gamma + 1) \in [0, 1]$ and $\theta = \mu/(\gamma\kappa + \mu) \in [0, 1]$. Then, for any $x \in \mathbf{E}$, and for any $\mu, \beta, \gamma > 0$, we have $\mathbf{prox}_{\gamma \mu_{\mathbb{Q}}}(x) = \theta\kappa x + (1 - \theta)\, \mathbf{\Pi}\,(\kappa x)$.*

Finally, combining (DRS), [159, Theorem 6.13], and Lemma 4.1, we arrive at Algorithm 3.

Algorithm subroutines. The inner algorithm requires two subroutines, evaluating (i) $\mathbf{prox}_{\gamma f}(x)$, which is the proximal operator of the convex function f at the input point x, and (ii) $\mathbf{\Pi}(x)$, which is a projection of x on the nonconvex set $\mathcal{X}$. We discuss now how we compute them in our implementation. To that goal, we recall that, for a function g (not necessarily convex) its proximal operator $\mathbf{prox}_{\gamma g}$ and Moreau envelope ${}^{\gamma}g$, where $\gamma > 0$, are defined as:

$$\begin{aligned} \mathbf{prox}_{\gamma g}(x) &= \operatorname*{argmin}_{y \in \mathbf{E}} \left(g(y) + (1/2\gamma)\|y - x\|^2 \right), \\ {}^{\gamma}g(x) &= \min_{y \in \mathbf{E}} \left(g(y) + (1/2\gamma)\|y - x\|^2 \right). \end{aligned} \tag{4.4}$$

Computing proximal operator of f. For the convex function f, $\mathbf{prox}_{\gamma f}$ is always single-valued and computing it is equivalent to solving a convex optimization problem, which often can be done in closed form for many relevant cost functions in machine learning [159, pp. 449-450]. If the proximal operator of f does not admit a closed form solution, then we solve the corresponding convex optimization problem (4.4) to a high precision solution. For this purpose, we can select any convex optimization solver supported by `MathOptInterface`, which is the abstraction layer for optimization solvers in `Julia`.

Computing projection onto $\mathcal{X}$. The notation $\mathbf{\Pi}(x)$ denotes the *projection operator* of x onto the constraint set $\mathcal{X}$, defined as

$$\mathbf{\Pi}_{\mathcal{X}}(x) = \mathbf{prox}_{\gamma \iota_{\mathcal{X}}}(x) = \operatorname{argmin}_{y \in \mathcal{X}}(\|y - x\|^2).$$

A list of nonconvex sets that are easy to project onto can be found in [141, §4], this includes nonconvex sets such as boolean vectors with fixed cardinality, vectors with bounded cardinality, quadratic sets, matrices with bounded singular values, matrices with bounded rank etc. If $\mathcal{X}$ is in this list, then we project onto $\mathcal{X}$ directly.

Now consider the case where the constraint set $\mathcal{X}$ decomposes as $\mathcal{X} = \mathcal{C} \bigcap \mathcal{N}$, where $\mathcal{N}$ is a nonconvex set with tractable projection and $\mathcal{C}$ is any compact convex set. In this setup, let $\iota_{\mathcal{C}}$ and $\iota_{\mathcal{N}}$ be the indicator functions of $\mathcal{C}$ and $\mathcal{N}$, respectively. Defining $\phi = f + \iota_{\mathcal{C}}$, we write problem ($\mathcal{P}$) as: $\min_{x \in \mathbf{E}} \phi(x) + (\beta/2)\|x\|^2 + \iota_{\mathcal{N}}(x)$.

For any convex function ϕ, its Moreau envelope ${}^{\nu}\phi$, for any $\nu > 0$, has the following three desirable features.

1. For every $x \in \mathbb{E}$ we have ${}^{\nu}\phi(x) \leq \phi(x)$ and ${}^{\nu}\phi(x) \to \phi(x)$ as $\nu \to 0$ [60, Theorem 1.25].
2. we have $x^\star \in \operatorname{argmin}_{x \in \mathbb{E}} \phi(x)$ if and only if $x^\star \in \operatorname{argmin}_{x \in \mathbb{E}} {}^{\nu}\phi(x)$ with the minimizer $x^\star$ satisfying $\phi(x^\star) = {}^{\nu}\phi(x^\star)$ [158, Corollary 17.5].
3. the Moreau envelope ${}^{\nu}\phi$ is convex, and smooth (*i.e.*, it is differentiable and its gradient is Lipschitz continuous) everywhere irrespective of the differentiability or smoothness of the original function ϕ. The gradient is: ${}^{\nu}\phi(x) = \left(x - \mathbf{prox}_{\nu\phi}(x)\right)/\nu$, which is $(1/\nu)-$Lipschitz continuous [158, Proposition 12.29].

These properties make ${}^{\nu}\phi$ a smooth approximation of ϕ for a small enough ν. Hence, we work with the following approximation of the original problem: $\min_x {}^{\nu}\phi + (\beta/2)\|x\|^2 + \iota_{\mathcal{N}}(x)$, where we replace f with ${}^{\nu}\phi$ and $\iota_{\mathcal{X}}$ with $\iota_{\mathcal{N}}$ in Algorithms 2 and 3. The proximal operator of ${}^{\nu}\phi$ can be computed using $\mathbf{prox}_{\gamma\,{}^{\nu}\phi}(x) = x + (\gamma/(\gamma+\nu))(\mathbf{prox}_{(\gamma+\nu)\phi}(x) - x)$, where computing $\mathbf{prox}_{(\gamma+\nu)\phi}(x)$ corresponds to solving the following convex optimization problem $\operatorname{argmin}_{y \in C} \phi(y) + 1/(2(\gamma+\nu))\|y - x\|^2$, which follows from [158, Proposition 24.8].

Remark 4.1. *Problem* ($\mathcal{P}_\mu$) *involves minimizing the sum of two functions: a convex function* f *and a nonconvex function* $\mu_{\square}$. *As the objective is split into two parts in problem* ($\mathcal{P}_\mu$), *selecting any other two-operator splitting algorithm (*e.g., *forward-backward splitting [160, page 25], Chambolle-Pock algorithm [160, page 32], ADMM [161]* etc.*) can work as the inner algorithm in principle. However, in the context of our problem setup, Douglas-Rachford splitting might be the most suitable choice for the following reasons.*

1. *We have picked Douglas-Rachford splitting over ADMM, because Douglas-Rachford operates on the original nonconvex problem, whereas ADMM can be viewed as Douglas-Rachford splitting on the dual of the original nonconvex problem [162]. As strong duality usually does not hold when the primal problem is nonconvex, it seems more intuitive to work with the nonconvex problem directly over its dual.*
2. *We favored Douglas-Rachford splitting over proximal gradient method, because even when the problem is convex, Douglas-Rachford splitting converges under more general conditions, whereas proximal gradient method require more restrictive conditions to converge [61, page 49]. Hence, we believe that Douglas-Rachford splitting represents the most natural choice for the inner algorithm over the proximal gradient method.*
3. *Douglas-Rachford splitting is favorably unique in contrast with other two-operator splitting methods, as Douglas-Rachford splitting is the only two-operator splitting method that satisfies the following properties simultaneously [163]: (i) it is constructed only with scalar multiplication, addition, and proximal operators, (ii) it computes proximal operators only once every iteration, (iii) it converges unconditionally for maximally monotone operators, and (iv) it does not increase the problem size.*

   In §4.3, some of these desirable properties of Douglas-Rachford splitting are exploited to establish convergence. While other operator splitting algorithms may work to establish convergence as well, some of the unique features of Douglas-Rachford splitting will be

lost [163].

4.3 Convergence analysis

This section is organized as follows. We start with the definition of the key geometry property of sets involving sparse and low-rank optimization problems. Then we define the local minima of such problems, followed by the assumptions we use in our convergence analysis. We next discuss the convergence roadmap, where the first step involves showing that the exterior point minimization function is locally strongly convex and smooth around local minima, and the second step entails connecting the local minima with the underlying operator controlling NExOS. Then, we present the main result, which shows that, under mild regularity conditions, the inner algorithm of NExOS finds local minima for the penalized problems at a linear convergence rate, and as the penalty parameter goes to zero, the local minima of the penalized problems converge to a local minimum of the original problem. Furthermore, we show that, when those regularity conditions do not hold, the inner algorithm is still guaranteed to subsequentially converge to a first-order stationary point at the rate $o(1/\sqrt{k})$.

The key geometric property of sparse and low-rank constraint sets that we use in our convergence analysis is *prox-regularity at local minima*, *i.e.*, having single-valued Euclidean projection around local minima [164]. Prox-regularity of a set at a point is defined as follows.

Definition 4.1 (Prox-regular set [164])**.** A nonempty closed set $\mathcal{S} \subseteq \mathbf{E}$ is prox-regular at a point $x \in \mathcal{S}$ if projection onto $\mathcal{S}$ is single-valued on a neighborhood of x. The set $\mathcal{S}$ is prox-regular if it is prox-regular at every point in the set.

If the constraint set $\mathcal{X}$ decomposes as $\mathcal{X} = \mathcal{C} \bigcap \mathcal{N}$, where $\mathcal{C}$ is a compact convex set, and $\mathcal{N}$ is prox-regular around local minima, then the feasible set $\mathcal{X}$ inherits the prox-regularity

property around local minima from the set $\mathcal{N}$ (see Lemma 4.4 in §4.3). The set $\mathcal{N}$ in (4.2) is a prox-regular set at any point $X \in \mathbf{R}^{m\times d}$ where $\mathbf{rank}(X) = r$ [165, Proposition 3.8]. One can show that $\mathcal{X}$ inherits the prox-regularity property at any X with $\mathbf{rank}(X) = r$ from the set $\mathcal{N}$; a formal proof is given in Lemma 4.4 in Appendix 4.A.1.1. Similarly, $\mathcal{N}$ in (4.1) is prox-regular at any point x satisfying $\mathbf{card}(x) = k$ because we can write $\mathbf{card}(x) \leq k$ as a special case of the low-rank constraint by embedding the components of x in the diagonal entries of a matrix and then using the prox-regularity of low-rank constraint set.

In our convergence analysis, we use the prox-regularity property of sparse and low-rank optimization to establish our convergence results, hence NExOS can be applied to problems involving other constraint sets that are prox-regular at local minimal. Some other notable prox-regular sets are as follows. Closed convex sets are prox-regular everywhere [60, page 612]. Examples of well-known prox-regular sets that are not convex include sets involving bilinear constraints [166], weakly convex sets [167], proximally smooth sets [168], strongly amenable sets [60, page 612], and sets with Shapiro property [169]. Also, a nonconvex set defined by a system of finitely many inequality and equality constraints for which a basic constraint qualification holds is prox-regular [106, page 10].

We next provide the definition of local minimum of problem ($\mathcal{P}$). Recall that, according to our setup the set $\mathcal{X}$ is prox-regular at local minimum.

Definition 4.2 (Local minimum of problem ($\mathcal{P}$))**.** A point $\bar{x} \in \mathcal{X}$ is a local minimum of problem ($\mathcal{P}$) if the set $\mathcal{X}$ is prox-regular at $\bar{x}$, and there exists a closed ball with center $\bar{x}$ and radius r, denoted by $\overline{B}(\bar{x}; r)$ such that for all $y \in \mathcal{X} \cap \overline{B}(\bar{x}; r) \setminus \{\bar{x}\}$, we have $f(\bar{x}) + (\beta/2)\|\bar{x}\|^2 < f(y) + (\beta/2)\|y\|^2$.

In the definition above, the strict inequality is due to the strongly convex nature of the

objective $f + (\beta/2)\|\cdot\|^2$ and follows from [170, Proposition 2.1] and [60, Theorem 6.12]. We now state and justify the assumptions used in our convergence analysis.

Assumption 4.1 (Strong convexity and smoothness of f)**.** *The function f in problem ($\mathcal{P}_\mu$) is α-strongly convex and L-smooth where $L > \alpha > 0$, i.e., $f - (\alpha/2)\|\cdot\|^2$ is convex and $f - (L/2)\|\cdot\|^2$ is concave.*

Assumption 4.2 (Problem ($\mathcal{P}$) is not trivial)**.** *The unique solution to the unconstrained strongly convex problem $\min_x f(x) + (\beta/2)\|x\|^2$ does not lie in $\mathcal{X}$.*

Assumption 4.1 corresponds to the function $f + (\beta/2)\|\cdot\|^2$ being $(\alpha+\beta)$-strongly convex and $(L+\beta)$-smooth. In our convergence analysis, $\beta > 0$ can be arbitrarily small, so it does not fall outside the setup described in §4.1. The L-smoothness in f is equivalent to its gradient ∇f being $L-$Lipschitz everywhere on $\mathbb{E}$ [158, Theorem 18.15]. In our convergence analysis, this assumption is required in establishing linear convergence of the inner algorithms of NExOS.

Assumption 4.2 imposes that a local minimum of problem ($\mathcal{P}$) is not the global minimum of its unconstrained convex relaxation, which does not incur any loss of generality. We can solve the unconstrained strongly convex optimization problem $\min_x f(x) + (\beta/2)\|x\|^2$ and check if the corresponding minimizer lies in $\mathcal{X}$; if that is the case, then that minimizer is also the global minimizer of problem ($\mathcal{P}$), and there is no point in solving the nonconvex problem. This can be easily checked by solving an unconstrained convex optimization problem, so Assumption 4.2 does not cause any loss of generality.

To discuss our convergence roadmap, we introduce some standard operator theoretic notions as follows. A set-valued operator $\mathbb{A} : \mathbb{E} \rightrightarrows \mathbb{E}$ maps an element x in $\mathbb{E}$ to a set $\mathbb{A}(x)$ in $\mathbb{E}$; its domain is defined as $\mathbf{dom}\,\mathbb{A} = \{x \in \mathbb{E} \mid \mathbb{A}(x) \neq \emptyset\}$, its range is defined as

$\mathbf{ran}\,\mathbb{A} = \bigcup_{x\in\mathbb{E}} \mathbb{A}(x)$, and it is completely characterized by its graph: $\mathbf{gra}\,\mathbb{A} = \{(u,x) \in \mathbb{E}\times\mathbb{E} \mid u \in \mathbb{A}(x)\}$. Furthermore, we define $\mathbf{fix}\,\mathbb{A} = \{x \in \mathbb{E} \mid x \in \mathbb{A}(x)\}$, and $\mathbf{zer}\,\mathbb{A} = \{x \in \mathbb{E} \mid 0 \in \mathbb{A}(x)\}$. For every x, addition of two operators $\mathbb{A}_1, \mathbb{A}_2 : \mathbf{E} \rightrightarrows \mathbf{E}$, denoted by $\mathbb{A}_1+\mathbb{A}_2$, is defined as $(\mathbb{A}_1+\mathbb{A}_2)(x) = \mathbb{A}_1(x)+\mathbb{A}_2(x)$, subtraction is defined analogously, and composition of these operators, denoted by $\mathbb{A}_1\mathbb{A}_2$, is defined as $\mathbb{A}_1\mathbb{A}_2(x) = \mathbb{A}_1(\mathbb{A}_2(x))$; note that order matters for composition. Also, if $\mathcal{S} \subseteq \mathbf{E}$ is a nonempty set, then $\mathbb{A}(\mathcal{S}) = \cup\{\mathbb{A}(x) \mid x \in \mathcal{S}\}$.

We next discuss our convergence roadmap. Convergence of NExOS is controlled by the DRS operator of problem ($\mathcal{P}_\mu$):

$$\mathbb{T}_\mu = \mathbf{prox}_{\gamma\mu_{\emptyset}}\left(2\mathbf{prox}_{\gamma f} - \mathbb{I}\right) + \mathbb{I} - \mathbf{prox}_{\gamma f}, \tag{4.5}$$

where $\mu > 0$, and $\mathbb{I}$ stands for the identity operator in $\mathbb{E}$, *i.e.*, for any $x \in \mathbb{E}$, we have $\mathbb{I}(x) = x$. Using $\mathbb{T}_\mu$, the inner algorithm—Algorithm 3—can be written as

$$z^{n+1} = \mathbb{T}_\mu\left(z^n\right) \tag{$\mathcal{A}_\mu$}$$

where μ is the penalty parameter and z^n is initialized at the fixed point from the previous inner algorithm.

To show the convergence of NExOS, we first show that for some $\mu_{\max} > 0$, for any $\mu \in (0, \mu_{\max}]$, the exterior point minimization function $f + {}^{\mu}{\emptyset}$ is strongly convex and smooth on some open ball with center x and radius $r_{\max}$, denoted by $B(\bar{x}; r_{\max})$, where it will attain a unique local minimum x_μ. Then we show that for $\mu \in (0, \mu_{\max}]$, the operator $\mathbb{T}_\mu(x)$ will be contractive in x and Lipschitz continuous in μ, and connects its fixed point set $\mathbf{fix}\,\mathbb{T}_\mu$ with the local minima x_μ, via the relationship $x_\mu = \mathbf{prox}_{\gamma f}(\mathbf{fix}\,\mathbb{T}_\mu)$. In the main convergence result, we show that for a sequence of penalty parameters $\mathfrak{M} = \{\mu_1, \mu_2, \mu_3, \ldots, \mu_N\}$ and under proper

initialization, if we apply NExOSto $\mathfrak{M}$, then for all $\mu_m \in \mathfrak{M}$,the inner algorithm will linearly converge to x_{μ_m}, and as $\mu_N \to 0$, we will have $x_{\mu_N} \to \bar{x}$. Finally, we show that, when the regularity conditions of the prior result do not hold, the inner algorithm is still guaranteed to subsequentially converge to a first-order stationary point (not necessarily a local minimum) at the rate $o(1/\sqrt{k})$.

We next present a proposition that shows that the exterior point minimization function in problem ($\mathcal{P}_\mu$) will be locally strongly convex and smooth around local minima for our selection of penalty parameters, even though problem ($\mathcal{P}$) is nonconvex. Furthermore, as the penalty parameter goes to zero, the local minimum of problem ($\mathcal{P}_\mu$) converges to the local minimum of the original problem ($\mathcal{P}$). So, under proper initialization, NExOS can solve the sequence of penalized problems $\{\mathcal{P}_\mu\}_{\mu\in(0,\mu_{\text{init}}]}$ similar to convex optimization problems; we will prove this in our main convergence result (Theorem 4.1).

Proposition 4.1 (Attainment of local minimum by $f + {}^{\mu}\mathbb{0}$)**.** *Let Assumptions 4.1 and 4.2 hold for problem ($\mathcal{P}$), and let $\bar{x}$ be a local minimum to problem ($\mathcal{P}$). Then the following hold.*

(i) There exist $\mu_{\max} > 0$ and $r_{\max} > 0$ such that for any $\mu \in (0, \mu_{\max}]$, the exterior point minimization function $f + {}^{\mu}\mathbb{0}$ in problem ($\mathcal{P}_\mu$) is strongly convex and smooth in the open ball $B(\bar{x}; r_{\max})$ and will attain a unique local minimum x_μ in this ball.

(ii) As $\mu \to 0$, this local minimum x_μ will go to $\bar{x}$ in limit, i.e., $x_\mu \to \bar{x}$.

Proof. See Appendix 4.A.2.2. □

Because the exterior point minimization function is locally strongly convex and smooth, intuitively the DRS operator of problem ($\mathcal{P}_\mu$) would behave similar to that of a DRS operator of a composite convex optimization problem, but locally. When we minimize a sum of two

convex functions where one of them is strongly convex and smooth, the corresponding DRS operator is contractive [171, Theorem 1]. So, we can expect that the DRS operator for problem ($\mathcal{P}_\mu$) would be locally contractive around a local minimum, which indeed turns out to be the case as proven in the next proposition. Furthermore, the next proposition shows that $\mathbb{T}_\mu(x)$ is locally Lipschitz continuous in the penalty parameter μ around a local minimum for fixed x. As $\mathbb{T}_\mu(x)$ is locally contractive in x and Lipschitz continuous in μ, it ensures that as we reduce the penalty parameter μ, the local minimum x_μ of problem ($\mathcal{P}_\mu$) found by NExOSdoes not change abruptly.

Proposition 4.2 (Characterization of $\mathbb{T}_\mu$)**.** *Let Assumptions 4.1 and 4.2 hold for problem ($\mathcal{P}$), and let $\bar{x}$ be a local minimum to problem ($\mathcal{P}$). Then the following hold.*

(i) There exists a contraction factor $\kappa' \in (0,1)$ such that for any $x_1, x_2 \in B(\bar{x}; r_{\max})$ and $\mu \in (0, \mu_{\max}]$, we have $\|\mathbb{T}_\mu(x_1) - \mathbb{T}_\mu(x_2)\| \leq \kappa' \|x_1 - x_2\|$.

(ii) For any $x \in B(\bar{x}; r_{\max})$, the operator $\mathbb{T}_\mu(x)$ is Lipschitz continuous in μ, i.e., there exists an $\ell > 0$ such that for any $\mu_1, \mu_2 \in (0, \mu_{\max}]$ and $x \in B(\bar{x}; r_{\max})$, we have $\|\mathbb{T}_{\mu_1}(x) - \mathbb{T}_{\mu_2}(x)\| \leq \ell \|\mu_1 - \mu_2\|$.

Proof. See Appendix 4.A.2.3. □

If the inner algorithm ($\mathcal{A}_\mu$) converges to a point z_μ, then z_μ would be a fixed point of the DRS operator $\mathbb{T}_\mu$. Establishing the convergence of NExOS necessitates connecting the local minimum x_μ of problem ($\mathcal{P}_\mu$) to the fixed point set of $\mathbb{T}_\mu$, which is achieved by the next proposition. Because our DRS operator locally behaves in a manner similar to the DRS operator of a convex optimization problem as shown by Proposition 4.2, it is natural to expect that the connection between x_μ and z_μ in our setup would be similar to that of a convex setup, but in a local sense. This indeed turns out to be the case as proven in the next

proposition. The statement of this proposition is structurally similar to [158, Proposition 25.1(ii)] that establishes a similar relationship globally for a convex setup, whereas our result is established around the local minima of problem ($\mathcal{P}_\mu$).

Proposition 4.3 (Relationship between local minima of problem ($\mathcal{P}$) and $\mathbf{fix}\,\mathbb{T}_\mu$)**.** *Let Assumptions 4.1 and 4.2 hold for problem ($\mathcal{P}$). Let $\bar{x}$ be a local minimum to problem ($\mathcal{P}$), and $\mu \in (0, \mu_{\max}]$. Then, $x_\mu = \operatorname{argmin}_{B(\bar{x};r_{\max})} f(x) + {}^{\mu}\mathbb{I}(x) = \mathbf{prox}_{\gamma f}(\mathbf{fix}\,\mathbb{T}_\mu)$, where the sets $\mathbf{fix}\,\mathbb{T}_\mu$, and $\mathbf{prox}_{\gamma f}(\mathbf{fix}\,\mathbb{T}_\mu)$ are singletons over $B(\bar{x}; r_{\max})$.*

Proof. See Appendix 4.A.2.4. □

Before we present the main convergence result, we provide a helper lemma, which shows how the distances between x_μ, z_μ and $\bar{x}$ change as μ is varied in Algorithm 2. Additionally, this lemma provides the range for the proximal parameter γ. If $\mathcal{X}$ is a bounded set satisfying $\|x\| \leq D$ for all $x \in \mathcal{X}$, then term $\max_{x \in B(\bar{x};r_{\max})} \|\nabla f(x)\|$ in this lemma can be replaced with $L \times D$.

Lemma 4.2 (Distance between local minima of problem ($\mathcal{P}$) with local minima of problem ($\mathcal{P}_\mu$))**.** *Let Assumptions 4.1 and 4.2 hold for problem ($\mathcal{P}$), and let $\bar{x}$ be a local minimum to problem ($\mathcal{P}$) over $B(\bar{x}; r_{\max})$. Then the following hold.*

(i) For any $\mu \in (0, \mu_{\max}]$, the unique local minimum x_μ of problem ($\mathcal{P}_\mu$) over $B(\bar{x}; r_{\max})$ satisfies $\|x_\mu - \bar{x}\| < r_{\max}/\eta'$ for some $\eta' > 1$.

(ii) Let z_μ be the unique fixed point of $\mathbb{T}_\mu$ over $B(\bar{x}; r_{\max})$ corresponding to x_μ. Then for any $\mu \in (0, \mu_{\max}]$, we have $r_{\max} - \|x_\mu - \bar{x}\| > (\eta' - 1) r_{\max}/\eta'$ and $r_{\max} - \|z_\mu - \bar{x}\| > \psi$, where $\psi = (\eta' - 1) r_{\max}/\eta' - \gamma \max_{x \in B(\bar{x};r_{\max})} \|\nabla f(x)\| > 0$ with the proximal parameter

γ taken to satisfy

$$0 < \gamma < (\eta' - 1) r_{\max} / \left(\eta' \max_{x \in B(\bar{x}; r_{\max})} \|\nabla f(x)\| \right).$$

Furthermore, $\min_{\mu \in (0, \mu_{\max}]} \{(r_{\max} - \|z_\mu - \bar{x}\|) - \psi\} > 0$.

Proof. See Appendix 4.A.2.5. □

We now present our main convergence results for NExOS. For convenience, we denote the n-th iterates of the inner algorithm of NExOSfor penalty parameter μ by $\{x_\mu^n, y_\mu^n, z_\mu^n\}$. In the theorem, an ϵ-approximate fixed point $\widetilde{z}$ of $\mathbb{T}_\mu$ is defined by $\max\{\|\widetilde{z} - \mathbb{T}_\mu(\widetilde{z})\|, \|z_\mu - \widetilde{z}\|\} \leq \epsilon$, where z_μ is the unique fixed point of $\mathbb{T}_\mu$ over $B(\bar{x}; r_{\max})$. Furthermore, define:

$$\bar{\epsilon} := \min\{ \min_{\mu \in (0, \mu_{\max}]} ((r_{\max} - \|z_\mu - \bar{x}\|) - \psi)/2, (1 - \kappa')\psi \} > 0, \tag{4.6}$$

where $\kappa' \in (0, 1)$ is the contraction factor of $\mathbb{T}_\mu$ for any $\mu > 0$ (cf. Proposition 4.2) and the right-hand side is positive due to the third and fifth equations of Lemma 4.2(ii). Theorem 4.1 states that if we have a good initial point z_{init} for the first penalty parameter μ_{init}, then NExOS will construct a finite sequence of penalty parameters such that all the inner algorithms for these penalty parameters will linearly converge to the unique local minima of the corresponding inner problems.

Theorem 4.1 (Convergence result for NExOS)**.** *Let Assumptions 4.1 and 4.2 hold for problem (P), and let $\bar{x}$ be a local minimum to problem (P). Suppose that the fixed-point tolerance ϵ for Algorithm 3 satisfies $\epsilon \in [0, \bar{\epsilon})$, where $\bar{\epsilon}$ is defined in (4.6). The proximal parameter γ is selected to satisfy the fourth equation of Lemma 4.2(ii). In this setup, NExOS will construct a finite sequence of strictly decreasing penalty parameters $\mathfrak{M} = \{\mu_1 := \mu_{\text{init}}, \mu_2 = \rho\mu_1, \mu_3 = \rho\mu_2, \ldots\}$,*

with $\mu_{\text{init}} \leq \mu_{\max}$ *and* $\rho \in (0,1)$*, such that we have the following recursive convergence property.*

For any $\mu \in \mathcal{M}$*, if an* ϵ*-approximate fixed point of* $\mathbb{T}_\mu$ *over* $B(\bar{x}; r_{\max})$ *is used to initialize the inner algorithm for penalty parameter* $\rho\mu$*, then the corresponding inner algorithm iterates* $z^n_{\rho\mu}$ *linearly converges to* $z_{\rho\mu}$ *that is the unique fixed point of* $\mathbb{T}_{\rho\mu}$ *over* $B(\bar{x}, r_{\max})$*, and the iterates* $x^n_{\rho\mu}, y^n_{\rho\mu}$ *linearly converge to* $x_{\rho\mu} = \mathbf{prox}_{\gamma f}(z_{\rho\mu})$*, which is the unique local minimum to* $(\mathcal{P}_{\rho\mu})$ *over* $B(\bar{x}; r_{\max})$*.*

Proof. See Appendix 4.A.2.6. □

From Theorem 4.1, we see that an ϵ-approximate fixed point of $\mathbb{T}_{\rho\mu}$ over $B(\bar{x}; r_{\max})$ can be computed and then used to initialize the next inner algorithm for penalty parameter $\rho^2\mu$; this chain of logic makes each inner algorithm linearly converge to the corresponding locally optimal solution. Finally, for the convergence of the first inner algorithm we have the following result, which states that if the initial point z_{init} is not "too far away" from $B(\bar{x}; r_{\max})$, then the first inner algorithm of NExOSfor penalty parameter μ_1 converges to a locally optimal solution of $(\mathcal{P}_{\mu_1})$.

Lemma 4.3 (Convergence of the first inner algorithm)**.** *Let* $\bar{x}$ *be a local minimum to problem* $(\mathcal{P})$*, where Assumptions 4.1 and 4.2 hold. Let* z_{init} *be the chosen initial point for* $\mu_1 := \mu_{\text{init}}$ *such that* $\overline{B}(z_{\mu_1}; \|z_{\text{init}} - z_{\mu_1}\|) \subseteq B(\bar{x}; r_{\max})$*, where* z_{μ_1} *be the corresponding unique fixed point of* $\mathbb{T}_{\mu_1}$*. Then,* $z^n_{\mu_1}$ *linearly converges to* z_{μ_1} *and both* $x^n_{\mu_1}$ *and* $y^n_{\mu_1}$ *linearly converge to the unique local minimum* x_{μ_1} *of* $(\mathcal{P}_{\mu_1})$ *over* $B(\bar{x}; r_{\max})$*.*

Proof. See Appendix 4.A.2.7. □

We now discuss what can be said if the initial point z_{init} does not necessarily satisfy the conditions stated in Theorem 4.1 or Lemma 4.3. Unfortunately, in such a situation, we can only show subsequential convergence of the iterates.

Theorem 4.2 (Convergence result for NExOS for z_{init} that is far away from $B(\bar{x}; r_{\max})$)**.** *Suppose, the proximal parameter γ is selected to satisfy $0 < \gamma < 1/L$ and let z_{init} be the any arbitrarily chosen initial point that does not satisfy the conditions of Lemma 4.3. Then, in this setup, NExOS will construct a finite sequence of strictly decreasing penalty parameters $\mathfrak{M} = \{\mu_1 := \mu_{\text{init}}, \mu_2 = \rho\mu_1, \mu_3 = \rho\mu_2, \ldots\}$, and $\rho \in (0, 1)$, such that we have the following recursive convergence property. For any $\mu \in \mathcal{M}$, if an ϵ-approximate fixed point of $\mathbb{T}_\mu$ over $B(\bar{x}; r_{\max})$ is used to initialize the inner algorithm for penalty parameter $\rho\mu$, then the corresponding inner algorithm iterates $z^n_{\rho\mu}$ subsequentially converges to $z_{\rho\mu}$ that is a fixed point of $\mathbb{T}_{\rho\mu}$, and the iterates $x^n_{\rho\mu}, y^n_{\rho\mu}$subsequentially converge to a first-order stationary point to $(\mathcal{P}_{\rho\mu})$ denoted by $x_{\rho\mu} = \mathbf{prox}_{\gamma f}(z_{\rho\mu})$ with the rate $\min_{n \le k} \|\nabla (f + {}^{\mu}\mathbb{1})(x^n_{\rho\mu})\| \le \frac{1-\gamma L}{2L} o(1/\sqrt{k})$.*

Proof. See Appendix 4.A.2.8. □

4.4 Numerical experiments

In this section, we apply NExOS to the following nonconvex optimization problems of substantial current interest for both synthetic and real-world datasets: sparse regression problem in §4.4.1, affine rank minimization problem in §4.4.2, and low-rank factor analysis problem in §4.4.3. We illustrate that NExOS produces solutions that are either competitive or better in comparison with the other approaches on different performance measures. We have implemented NExOS in `NExOS.jl` solver, which is an open-source software package written in the `Julia` programming language. `NExOS.jl` can address any optimization problem of

the form of problem ($\mathcal{P}$). The code and documentation are available online at: https://github.com/Shuvomoy/NExOS.jl.

In our numerical experiments, we present a comprehensive evaluation of NExOS, showing both statistical and optimization-theoretic evaluations. This dual approach is deliberate—while our primary contribution is in developing optimization methodology, the optimization problems considered in this section—such as sparse regression, affine rank minimization, matrix completion, and factor analysis—are deeply rooted in the fields of statistics and machine learning [17, 18, 147, 172–174]. Therefore, our numerical experiments are constructed not only to demonstrate NExOS efficiently computing local minima for nonconvex problems but also to highlight its ability to provide statistically robust solutions, which are also important in the application context. This dual capacity is of paramount importance for practical applications in statistics and machine learning, underlining the algorithm's versatility and effectiveness. By addressing these aspects, we aim to illustrate the broad applicability of NExOS across optimization-theoretic and applied statistical or learning domains.

To compute the proximal operator of a function f with closed form or easy-to-compute solution, `NExOS.jl` uses the open-source package ProximalOperators.jl [175]. When f is a constrained convex function (*i.e.*, a convex function over some convex constraint set) with no closed form proximal map, `NExOS.jl` computes the proximal operator by using the open-source `Julia` package `JuMP` [53] and any of the commercial or open-source solver supported by it. The set $\mathcal{X}$ can be any prox-regular nonconvex set fitting our setup. Our implementation is readily extensible using `Julia` abstract types so that the user can add support for additional convex functions and prox-regular sets. The numerical study is executed on a MacBook Pro laptop with Apple M1 Max chip with 32 GB memory. The datasets considered in this section, unless specified otherwise, are available online at: http://tinyurl.com/NExOSDatasets.

In applying NExOS, we use the following values that we found to be the best performing based on our empirical observations. We take the starting value of μ to be 2, and reduce this value with a multiplicative factor of 0.5 during each iteration of the outer loop until the termination criterion is met. The value of the proximal parameter γ is chosen to be 10^{-3}. We initialize our iterates at $\mathbf{0}$. Maximum number of inner iterations for a fixed value of μ is taken to be 1000. The tolerance for the fixed point gap for each penalized problem is taken to be 10^{-4} and the tolerance for the termination criterion is taken to be 10^{-6}.

Value of β is taken to be 10^{-8} for the following reasons. In §4.3, we showed that the presence of $\beta > 0$, ensures that each penalized subproblem is locally strongly convex and smooth, having a unique local minimum. This, in turn, helps to establish linear convergence of the inner algorithm for each subproblem. We empirically demonstrate in this section that the impact of the condition $\beta > 0$, despite being critical in the theoretical analysis of our algorithm, seems to only be marginal as it can be made to be as small as 10^{-8}. We use this extremely small value of β to stress-test NExOS empirically and show that even for such a small value of β, our algorithm still works well in practice.

4.4.1 Sparse regression

In (SR), we set $\mathcal{X} := \{x \mid \|x\|_\infty \leq \Gamma, \mathbf{card}(x) \leq k\}$, and $f(x) := \|Ax - b\|_2^2$. A projection onto $\mathcal{X}$ can be computed using the formula in [18, §2.2], whereas the proximal operator for f can be computed using the formula in [161, §6.1.1]. Now we are in a position to apply NExOS to this problem.

4.4.1.1 Synthetic dataset: comparison with elastic net and `Gurobi`

We compare the solution found by NExOS with the solutions found by elastic net (glmnet used for the implementation) and spatial branch-and-bound algorithm (Gurobi used for the

implementation). elastic net is a well-known method for computing an approximate solution to the regressor selection problem (SR), which empirically works extremely well in recovering support of the original signal. On the other hand, Gurobi's spatial branch-and-bound algorithm is guaranteed to compute a globally optimal solution to (SR). NExOS is guaranteed to provide a locally optimal solution under regularity conditions; so to investigate how close NExOS can get to the globally minimum value we consider a parallel implementation of NExOS running on multiple (20) worker processes, where each process runs NExOS with different random initialization, and we take the solution associated with the least objective value.

Elastic net. elastic net is a well-known method for solving the regressor selection problem, that computes an approximate solution as follows. First, elastic net solves:

$$\text{minimize} \quad \|Ax - b\|_2^2 + \lambda\|x\|_1 + (\beta/2)\|x\|_2^2, \tag{4.7}$$

where λ is a parameter that is related to the sparsity of the decision variable $x \in \mathbf{R}^d$. To solve (4.7), we have used the open-source `R` pacakge glmnet [176].

To compute λ corresponding to $\mathbf{card}(x) \leq k$ we follow the method proposed in [172, §3.4] and [58, Example 6.4]. We solve the problem (4.7) for different values of λ, and find the smallest value of λ for which $\mathbf{card}(x) \leq k$, and we consider the sparsity pattern of the corresponding solution $\tilde{x}$. Let the index set of zero elements of $\tilde{x}$ be $\mathcal{Z}$, where $\mathcal{Z}$ has $d - k$ elements. Then the elastic net solves:

$$\begin{array}{ll} \text{minimize} & \|Ax - b\|_2^2 + (\beta/2)\|x\|_2^2 \\ \text{subject to} & (\forall j \in \mathcal{Z}) \quad x_j = 0, \end{array} \tag{4.8}$$

where $x \in \mathbf{R}^d$ is the decision variable. Solving this problem corresponds to solving a positive

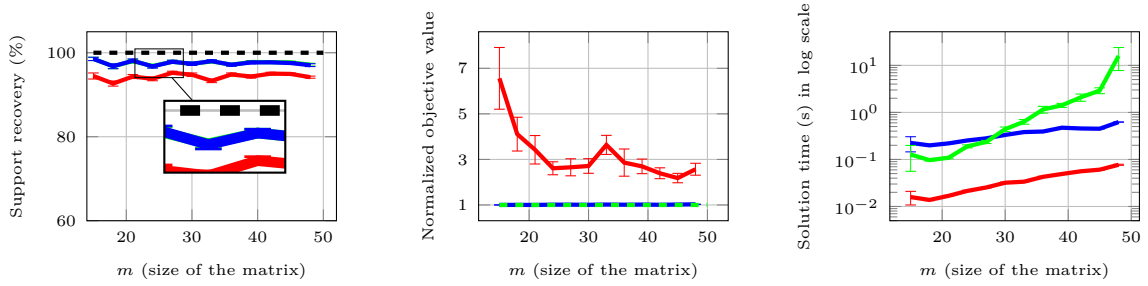
semidefinite linear system, which we solve using the built-in `LinearAlgebra` package in `Julia`.

Spatial branch-and-bound algorithm. The problem (SR) can also be modeled equivalently as the following mixed integer quadratic optimization problem [22]:

$$\begin{array}{ll} \text{minimize} & \|Ax-b\|_2^2 + (\beta/2)\|x\|_2^2 \\ \text{subject to} & |x_i| \leq \Gamma y_i, \quad i = 1, \ldots, d \\ & \sum_{i=1}^d y_i \leq k, \quad x \in \mathbf{R}^d, \quad y \in \{0,1\}^d, \end{array}$$

which can be solved to a certifiable global optimality using Gurobi's spatial branch-and-bound algorithm.

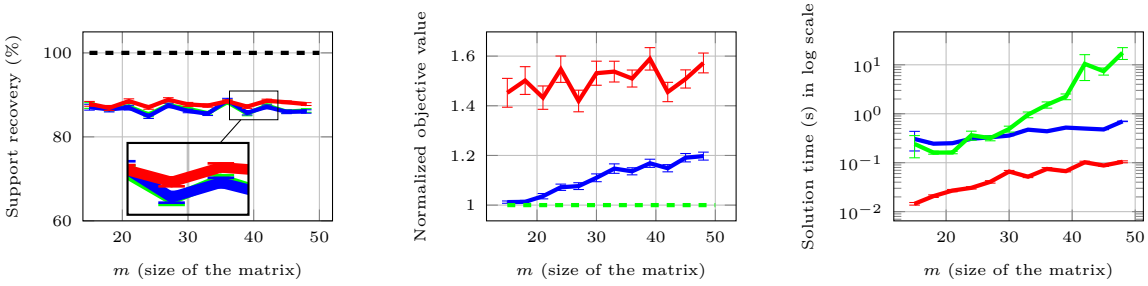
Data generation process and setup. The data generation procedure is similar to [141] and [173]. We consider two signal-to-noise ratio (SNR) regimes: SNR 1 and SNR 6, where for each SNR, we vary m from 25 to 50, and for each value of m, we generate 50 random problem instances. We limit the size of the problems because the solution time by Gurobi's spatial branch-and-bound algorithm becomes too large for comparison if we go beyond the aforementioned size. For a certain value of m, the matrix $A \in \mathbf{R}^{m\times 2m}$ is generated from an independent and identically distributed normal distribution with $\mathcal{N}(0,1)$ entries. We choose $b = A\widetilde{x} + v$, where $\widetilde{x}$ is drawn uniformly from the set of vectors satisfying $\mathbf{card}(\widetilde{x}) \leq \lfloor m/5 \rceil$ and $\|\widetilde{x}\|_\infty \leq \Gamma$ with $\Gamma = 1$. The vector v corresponds to noise, and is drawn from the distribution $\mathcal{N}(0, \sigma^2 I)$, where $\sigma^2 = \|A\widetilde{x}\|_2^2/(\mathrm{SNR}^2/m)$, which keeps the signal-to-noise ratio to approximately equal to SNR. We consider a parallel implementation of NExOS where we have 100 runs of NExOS distrubuted over 20 independent worker processes on 10 cores. Each run is initialized with a random initial points chosen from the uniform distribution over the interval $[-\Gamma, \Gamma]$. Gurobi's spatial branch-and-bound algorithm also uses 10 cores.



(a) Support recovery (SNR 6)

(b) Objective value (SNR 6)

(c) Solution time (s) (SNR 6)



(d) Support recovery (SNR 1)

(e) Objective value (SNR 1)

(f) Solution time (s) (SNR 1)

Figure 4.2: Sparse regression problem: comparison between NExOS (shown in blue), glmnet (shown in red), and Gurobi (shown in green). The first and second rows correspond to SNR 6 and SNR 1, respectively. For each SNR, the first column compares support recovery, the second column shows how close the objective value of the solution found by each algorithm gets to the optimal objective value (normalized as 1), and the third column shows the solution time (s) of each algorithm.

Results. Figure 4.2 compares NExOS (shown in blue), glmnet (shown in red) and Gurobi (shown in green) for solving (SR). The results displayed in the figures are averaged over 50 simulations for each value of m, and also show one-standard-error bands that represent one standard deviation confidence interval around the mean.

Figures 4.2(a) and 4.2(d) show the support recovery (%) of the solutions found by NExOS, glmnet, and Gurobi for SNR 6 and SNR 1, respectively. Given a solution x and true signal x^{True}, the support recovery is defined as $\sum_{i=1}^{d} 1_{\{\text{sign}(x_i)=\text{sign}(x_i^{\text{True}})\}}/d$, where $1_{\{.\}}$ evaluates to 1 if $(\cdot)$ is true and 0 else, and $\text{sign}(t)$ is 1 for $t > 0$, -1 for $t < 0$, and 0 for $t = 0$. So, higher the support recovery, better is the quality of the found solution. For both SNRs, NExOS and Gurobi have almost identical support recovery. For the high SNR, NExOS recovers most of the original signal's support and is better than glmnet consistently. On average, NExOS recovers 4% more of the support than glmnet. However, this behaviour changes for the low SNR, where glmnet recovers 1.26% more of the support than NExOS. This differing behavior in low and high SNR is consistent with the observations made in [173].

Figures 4.2(b) and 4.2(e) compare the quality of the solution found by the algorithms in terms of the normalized objective value (the objective value of the found solution divided by the otimal objective value) for SNR 6 and SNR 1, respectively. As Gurobi's spatial branch-and-bound algorithm finds certifiably globally optimal solution to (SR), its normalized objective value is always 1, though the runtime is orders of magnitude slower than glmnet and NExOS (see the next paragraph). The closer the normalized objective value is to 1, better is the quality of the solution in terms of minimizing the objective value. We see that for the high SNR, on average NExOS is able to find a solution that is very close to the globally optimal solution, whereas the solution found by glmnet has worse objective value on average. For the low SNR, on average the normalized objective values of the solutions found by both NExOS and glmnet get worse, though NExOS does better than glmnet in this case as well.

Finally, in Figures 4.2(c) and 4.2(f), we compare the solution times (in seconds and on log scale) of the algorithms for SNR 6 and SNR 1, respectively. We see that glmnet is slightly faster than NExOS. This slower performance is due to the fact that NExOS is a general purpose method, whereas glmnet is specifically optimized for the convexified sparse regression problem with a specific cost function. For smaller problems, Gurobi is somewhat faster than NExOS, however once we go beyond $m \geq 27$, the solution time by Gurobi starts to increase drastically. Beyond $m \geq 50$, comparing the solution times is not meaningful as Gurobi cannot find a solution in 2 minutes, whereas NExOS takes less than 30 seconds.

4.4.1.2 Experiments and results for real-world dataset

Description of the dataset. We now investigate the performance of our algorithm on a real-world, publicly available dataset called the `weather prediction dataset`, where we consider the problem of predicting the temperature half a day in advance in 30 US and Canadian Cities along with 6 Israeli cities. The dataset contains hourly measurements of weather attributes *e.g.,* temperature, humidity, air pressure, wind speed, and so on. The dataset has $m = 45,231$ instances along with $d = 1,800$ attributes. The dataset is preprocessed in the same manner as described in [140, §8.3]. Our goal is to predict the temperature half a day in advance as a linear function of the attributes, where at most k attributes can be nonzero. We include a bias term in our model, *i.e.*, in (SR) we set $A = [\bar{A} \mid \mathbf{1}]$. We randomly split 80% of the data into the training set and 20% of the data into the test set.

Results. Figure 4.3 shows the RMS error for the training datasets and the test datasets for both NExOS and glmnet. The results for training and test datasets are reasonably similar for each value of k. This gives us confidence that the sparse regression model will have similar performance on new and unseen data. This also suggests that our model does not suffer from over-fitting. We also see that, for $k \geq 20$ and $k \geq 5$, none of the errors for NExOS and

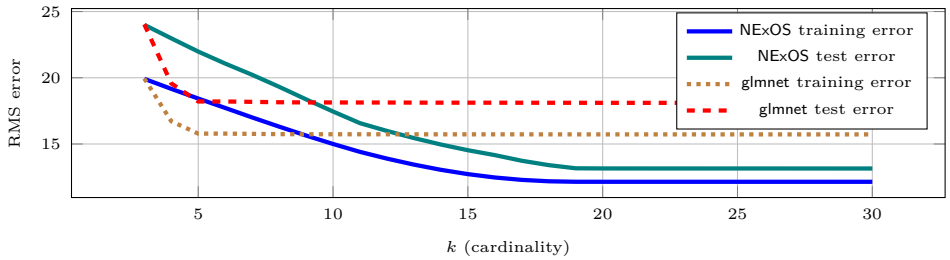


Figure 4.3: RMS error vs k (cardinality) for the weather prediction problem.

glmnet drop significantly, respectively. For smaller $k \leq 10$, glmnet does better than NExOS, but beyond $k \geq 10$, NExOS performs better than glmnet.

4.4.2 Affine rank minimization problem

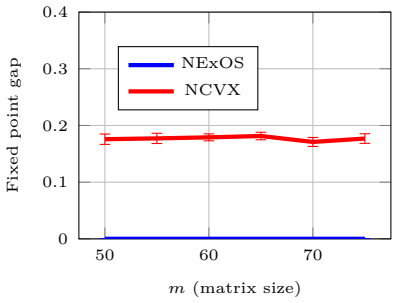
Problem description. In (SR), we set $\mathcal{X} := \{X \in \mathbf{R}^{m\times d} \mid \mathbf{rank}(X) \leq r, \|X\|_2 \leq \Gamma\}$, and $f(X) := \|\mathcal{A}(X) - b\|_2^2$. To compute the proximal operator of f, we use the formula in [161, §6.1.1]. Finally, we use the formula in [141, page 14] for projecting onto $\mathcal{X}$. Now we are in a position to apply the NExOS to this problem.

Summary of the experiments performed. *First,* we apply NExOS to solve (RM) for synthetic datasets, where we observe how the algorithm performs in recovering a low-rank matrix given noisy measurements and also compare NExOS with NCVX—an ADMM-based algorithm [141]. *Second,* we apply NExOS to a real-world dataset (`MovieLens 1M Dataset`) to see how our algorithm performs in solving a matrix-completion problem).

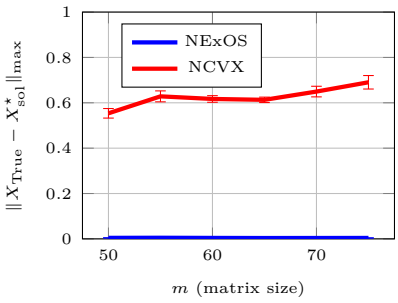
4.4.2.1 Experiments and results for synthetic dataset

Data generation process and setup. We generate the data as follows similar to [141]. We vary m (number of rows of the decision variable X) from 50 to 75 with a linear spacing of 5, where we take $d = 2m$, and rank to be equal to $m/10$ rounded to the nearest integer. For each value of m, we create 25 random instances as follows. The operator $\mathcal{A}$ is drawn from

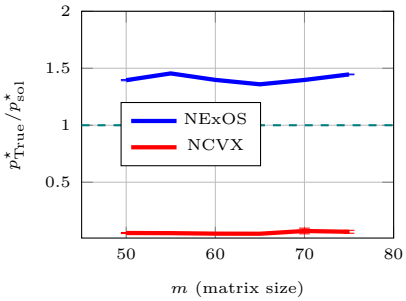
an iid normal distribution with $\mathcal{N}(0,1)$ entries. Similarly, we create the low rank matrix X_{True} with rank r, first drawn from an iid normal distribution with $\mathcal{N}(0,1)$ entries, and then truncating the singular values that exceed Γ to 0. Signal-to-noise ratio is taken to be around 20 by following the same method described for the sparse regression problem.



(a) Fixed point gap representing convergence of the iterates vs m



(b) Maximum absolute error in recovering the original matrix vs m



(c) Ratio of training losses of the true matrix X_{True} and the solution found by NExOS vs m

Figure 4.4: Affine rank minimization problem: comparison between solutions found by NExOS and NCVX algorithm by [141].

Results. The results displayed in Figure 4.4 average over 50 simulations for each value of m and also show one standard error band. We compare NExOS, with NCVX—an ADMM-based algorithm [141].

Fig 4.4a plots the normalized fixed point gap of the iterates for both algorithms computed by

$\|X^\star_{\text{Alg}} - Y^\star_{\text{Alg}}\|/\|X_{\text{True}}\|$with Alg $\in \{NExOS, NCVX\}$ and $X^\star_{\text{Alg}}, Y^\star_{\text{Alg}}$ representing the final iterates produced by the algorithms. This plot shows that NCVX iterates have a fixed point gap larger than 0.17, *i.e.*, the iterates do not converge within a reasonable fixed point gap. On the other hand, NExOS iterates converge with a normalized fixed-point gap reaching the desired tolerance of less than or equal to 10^{-4} for each instance.

Figure 4.4b shows how well NExOS and NCVX recovers the original matrix X_{True}. To quantify the recovery, we compute the max norm of the difference matrix $\|X_{\text{True}} - X^\star_{\text{Alg}}\|_{\max} = \max_{i,j} |X_{\text{True}}(i,j) - X^\star_{\text{Alg}}(i,j)|$, where the solution found by Alg is denoted by $X^\star_{\text{Alg}}$. We see that the worst-case component-wise error is very small (smaller than 0.005 for each instance) in all the cases for NExOS, but for NCVX, it is larger than 0.5 for each instance. In other words, the solution found by NExOS is much closer to the ground truth as compared to NCVX.

Finally, we show how the training loss of the solutions computed by NExOS and NCVX compare with the original matrix X_{True} in Figure 4.4c. Note that for NExOS, the ratio $p^\star_{\text{True}}/p^\star_{\text{sol}}$ is larger than one in most cases, *i.e.*, NExOS find a solutions with smaller cost compared to X_{True}. This is due to the fact that under the signal-to-noise ratio that we consider, the problem data can be explained better by another matrix with a lower training loss. That being said, $X^\star_{NExOS}$ is not too far from X_{True} component-wise as we saw in Figure 4.4b. On the other hand, for NCVX algorithm, the ratio $p^\star_{\text{True}}/p^\star_{\text{sol}}$ is smaller than 0.05 for each instance, *i.e.*, the objective value of the solutions is 20 times worse than that of the original signal.

4.4.2.2 Experiments and results for real-world dataset: matrix completion problem

Description of the dataset. To investigate the performance of our problem on a real-world dataset, we consider the publicly available `MovieLens 1M Dataset`. This dataset contains

1,000,023 ratings for 3,706 unique movies; these recommendations were made by 6,040 MovieLens users. The rating is on a scale of 1 to 5. If we construct a matrix of movie ratings by the users (also called the preference matrix), denoted by Z, then it is a matrix of 6,040 rows (each row corresponds to a user) and 3,706 columns (each column corresponds to a movie) with only 4.47% of the total entries are observed, while the rest being missing. Our goal is to complete this matrix, under the assumption that the matrix is low-rank. For more details about the model, see [18, §8.1].

To gain confidence in the generalization ability of this model, we use an out-of-sample validation process. By random selection, we split the available data into a training set (80% of the total data) and a test set (20% of the total data). We use the training set as the input data for solving the underlying optimization process, and the held-out test set is used to compute the test error for each value of r. The best rank r corresponds to the point beyond which the improvement is rather minor. We tested rank values r ranging in $\{1, 3, 5, 7, 10, 20, 25, 30, 35\}$. We compute the RMS error as follows. Let Ω_{test} be the index set corresponding to the test data. If $X_N ExOS^\star$ is the matrix returned by NExOS, then the corresponding RMS error is computed by using the formula

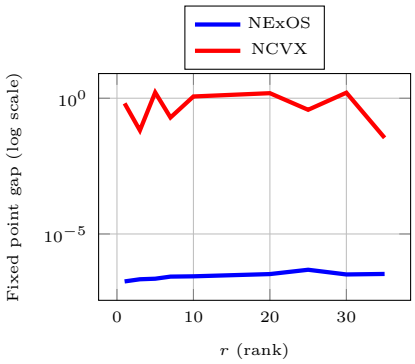
$$\text{RMS} = \sqrt{\frac{\sum_{(i,j)\in\Omega_{\text{test}}} \left((X_N ExOS^\star)_{ij} - Z_{ij} \right)^2}{|\Omega_{\text{test}}|}},$$

where $|\Omega_{\text{test}}|$ is the number of elements in Ω_{test}.

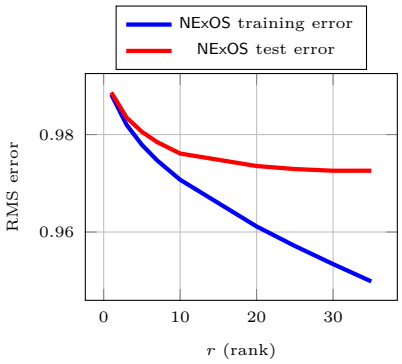
Matrix completion problem. The matrix completion problem is:

$$\begin{array}{ll} \text{minimize} & \sum_{(i,j)\in\Omega}(X_{ij} - Z_{ij})^2 + (\beta/2)\|X\|_F^2 \\ \text{subject to} & \mathbf{rank}(X) \leq r, \quad \|X\|_2 \leq \Gamma, \end{array} \tag{MC}$$

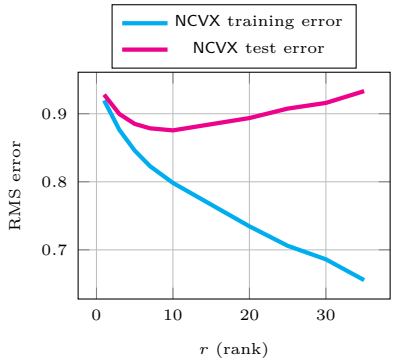
where $Z \in \mathbf{R}^{m \times d}$ is the matrix whose entries Z_{ij} are observable for $(i, j) \in \Omega$. Based on these observed entries, our goal is to construct a matrix $X \in \mathbf{R}^{m \times d}$ that has rank r. The problem above can be written as a special case of affine rank minimization problem (RM).



(a) Fixed point gap representing convergence of the iterates vs r



(b) Training and test error for NExOS vs r



(c) Training and test error for NCVX vs r

Figure 4.5: Matrix completion problem: comparison between solutions found by NExOS and NCVX algorithm by [141].

Results. Figure 4.5 compares the solutions found by NExOS and NCVX.

Fig 4.5a plots the normalized fixed point gap of the iterates for both algorithms calculated by $\|X^\star_{\text{Alg}} - Y^\star_{\text{Alg}}\|/\|X_{\text{True}}\|$with Alg $\in \{NExOS, NCVX\}$ and $X^\star_{\text{Alg}}, Y^\star_{\text{Alg}}$ representing the final iterates produced by the algorithms. This plot shows that NCVX iterates do not converge within a reasonable fixed point gap, whereas NExOS iterates converge for all the instances

with a normalized fixed-point gap less than or equal to 10^{-6} for each instance.

Figure 4.5b shows the RMS error of NExOS for the training datatest and test dataset for each value of rank r. The results for training and test datasets are reasonably similar for each value of r. We observe that beyond rank 15, the reduction in the test error is rather minor and going beyond this rank provides only diminishing returns, which is a common occurrence for low-rank matrix approximation [177, §7.1]. Thus we can choose the optimal rank to be 15 for all practical purposes.

Figure 4.5c shows the RMS error of NCVX for the training dataset and test dataset for each value of rank r. We see that, unlike NExOS, the test error for NCVX keeps increasing with r, whereas the training error NCVX is smaller. Here we note that, because NCVX iterates do not reach a reasonable fixed point gap, the training or test error of NCVX may not provide meaningful information.

4.4.3 Factor analysis problem

Problem description. The factor analysis model with sparse noise (also known as low-rank factor analysis model) involves decomposing a given positive semidefinite matrix as a sum of a low-rank positive semidefinite matrix and a diagonal matrix with nonnegative entries [17, page 191]. It can be posed as [174]:

$$\begin{array}{ll} \text{minimize} & \|\Sigma - X - D\|_F^2 + (\beta/2)\left(\|X\|_F^2 + \|D\|_F^2\right) \\ \text{subject to} & D = \mathbf{diag}(d), \quad d \geq 0, \quad X \succeq 0, \quad \mathbf{rank}(X) \leq r \\ & \Sigma - D \succeq 0, \quad \|X\|_2 \leq \Gamma, \end{array} \tag{FA}$$

where $X \in \mathbf{S}^p$ and the diagonal matrix $D \in \mathbf{S}^p$ with nonnegative entries are the decision variables, and $\Sigma \in \mathbf{S}_+^p$, $r \in \mathbb{Z}_+$, and $\Gamma \in \mathbb{R}_{++}$ are the problem data. A proper solution for

(FA) requires that both X and D are positive semidefinite. The term $\Sigma - D$ has to be positive semidefinite, else statistical interpretations of the solution is not impossible [178, page 326].

In (FA), we set $\mathcal{X} := \{(X, D) \in \mathbf{S}^p \times \mathbf{S}^p \mid \|X\|_2 \leq \Gamma, \mathbf{rank}(X) \leq r, D = \mathbf{diag}(d), d \geq 0\}$, and $f(X, D) \coloneqq \|\Sigma - X - D\|_F^2 + I_{\mathcal{P}}(X, D)$,where $I_{\mathcal{P}}$ denotes the indicator function of the convex set $\mathcal{P} = \{(X, D) \in \mathbf{S}^p \times \mathbf{S}^p \mid X \succeq 0, D = \mathbf{diag}(d), d \geq 0, d \in \mathbb{R}^p\}$. To compute the projection onto $\mathcal{X}$, we use the formula in [141, page 14] and the fact that $\mathbf{\Pi}_{\{y|y\geq 0\}}(x) = \max\{x, 0\}$, where pointwise max is used. The proximal operator for f at (X, D) can be computed by solving:

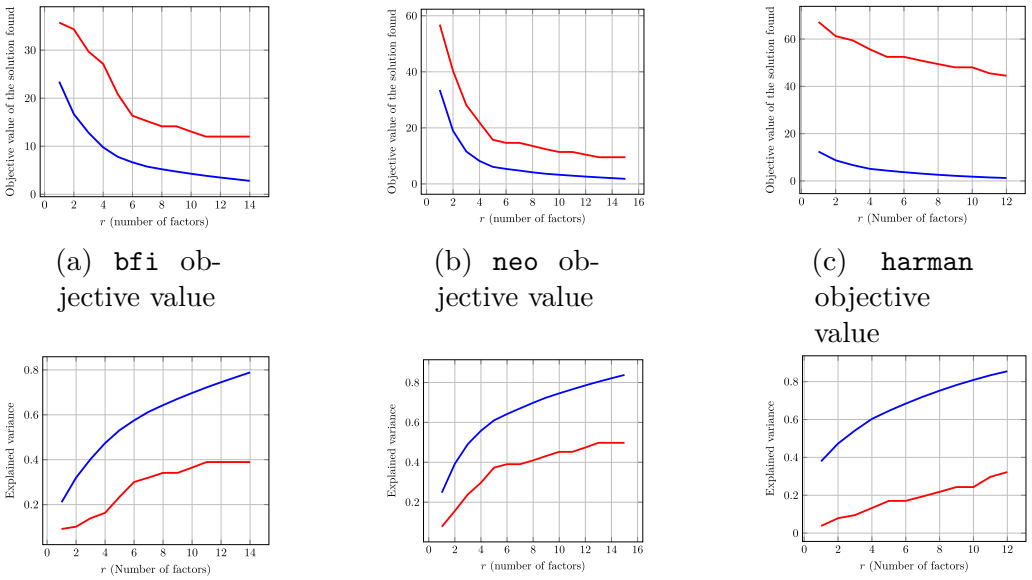
$$\begin{array}{ll} \text{minimize} & \|\Sigma - \widetilde{X} - \widetilde{D}\|_F^2 + (1/2\gamma)\|\widetilde{X} - X\|_F^2 + (1/2\gamma)\|\widetilde{D} - D\|_F^2 \\ \text{subject to} & \widetilde{X} \succeq 0, \quad \widetilde{D} = \mathbf{diag}(\tilde{d}), \quad \Sigma - \widetilde{D} \succeq 0, \quad \tilde{d} \geq 0, \end{array}$$

where $\widetilde{X} \in \mathbf{S}_+^p$, and $\tilde{d} \in \mathbb{R}_+^p$ (*i.e.*, $\widetilde{D} = \mathbf{diag}(\tilde{d})$) are the optimization variables. Now we are in a position to apply NExOS to this problem.

Comparison with nuclear norm heuristic. We compare the solution provided by NExOS to that of the nuclear norm heuristic, which isthe most well-known heuristic to approximately solve (FA) [179] via following convex relaxation:

$$\begin{array}{ll} \text{minimize} & \|\Sigma - X - D\|_F^2 + \lambda \|X\|_* \\ \text{subject to} & D = \mathbf{diag}(d), \quad d \geq 0, \quad X \succeq 0, \\ & \Sigma - D \succeq 0, \quad \|X\|_2 \leq \Gamma, \end{array} \tag{4.9}$$

where λ is a positive parameter that is related to the rank of the decision variable X. Note that, as X is positive semidefinite, we have its nuclear norm $\|X\|_* = \mathbf{tr}(X)$.



(a) `bfi` objective value

(b) `neo` objective value

(c) `harman` objective value

(d) `bfi` explained variance

(e) `neo` explained variance

(f) `harman` explained variance

Figure 4.6: Figure showing performance of NExOS in solving factor analysis problem for different datasets. Each column represents one dataset. The first and second row compares training loss and proportion of the variance explained of the solutions found by NExOS (shown in blue) and the nuclear norm heuristic (shown in red).

Performance measures. We consider two performance measures. First, we compare the training loss $\|\Sigma - X - D\|_F^2$ of the solutions found by NExOS and the nuclear norm heuristic. As both NExOS and the nuclear norm heuristic provide a point from the feasible set of (FA), such a comparison of training losses tells us which algorithm is providing a better quality solution. Second, we compute the *proportion of explained variance*, which represents how well the r-common factors explain the residual covariance, *i.e.*, $\Sigma - D$. For a given r, input proportion of variance explained by the r common factors is given by: $\sum_{i=1}^{r} \sigma_i(X) / \sum_{i=1}^{p} \sigma_i(\Sigma - D)$, where X, D are inputs, that correspond to solutions found by NExOS or the nuclear norm heuristic. As r increases, the explained variance increases to 1. The higher the value of the explained variance for a certain solution, the better is the quality of the solution.

Description of the datasets. We consider three different real-world bench-mark datasets that are popularly used for factor analysis. The `bfi`, `neo` , and `Harman74` datasets contain (2800 observations, 28 variables), (1000 observations, 30 variables), and (145 observations, 24 variables), respectively.

Setup. In applying NExOS for the factor analysis problem, we initialize our iterates with $Z_0 := \Sigma$ and $z_0 := \mathbf{0}$. All the other parameters are kept at their default values as stated in the beginning of §4.4. For each dataset, we vary the number of factors from 1 to $\lfloor p/2 \rfloor$, where p is the size of the underlying matrix Σ.

Results. Figure 4.6 shows performance of NExOS in solving the factor analysis problem for different datasets, with each row representing one dataset. The first row compares the training loss of the solution found by NExOS and the nuclear norm heuristic. We see that for all the datasets, NExOS finds a solution with a training loss that is considerably smaller than that of the nuclear norm heuristic. The second row shows the proportion of variance

explained by the algorithms considered for the datasets considered (higher is better). We see that in terms of the proportion of explained variance, NExOS delivers larger values than that of the nuclear norm heuristic for different values of r, which is indeed desirable. NExOS consistently provides solutions with better objective value and explained variance compared to the nuclear norm heuristic.

4.5 Conclusion

In this chapter, we have presented NExOS, a first-order algorithm to solve optimization problems with convex cost functions over nonconvex constraint sets— a problem structure that is satisfied by a wide range of nonconvex optimization problems including sparse and low-rank optimization. We have shown that, under mild technical conditions, NExOS is able to find a locally optimal point of the original problem by solving a sequence of penalized problems with strictly decreasing penalty parameters. We have implemented our algorithm in the `Julia` package `NExOS.jl` and have extensively tested its performance on a wide variety of nonconvex optimization problems. We have demonstrated that NExOS is able to compute high quality solutions at a speed that is competitive with tailored algorithms.

4.A Appendix for Chapter 4

4.A.1 Proof and derivation to results in §4.1

4.A.1.1 Lemma regarding prox-regularity of intersection of sets

Lemma 4.4. *Consider the nonempty constraint set $\mathcal{X} = \mathcal{C} \bigcap \mathcal{N} \subseteq \mathbb{E}$, where $\mathcal{C}$ is compact and convex, and $\mathcal{N}$ is prox-regular at $x \in \mathcal{X}$. Then $\mathcal{X}$ is prox-regular at x.*

Proof to Lemma 4.4. To prove this result we record the following result from [180], where by $d_{\mathcal{S}}(x)$ we denote the Euclidean distance of a point x from the set $\mathcal{S}$, and $\overline{\mathcal{S}}$ denotes closure of a set $\mathcal{X}$.

Lemma 4.5 (Intersection of prox-regular sets [180, Corollary 7.3(a)])**.** *Let $\mathcal{S}_1, \mathcal{S}_2$ be two closed sets in $\mathbb{E}$, such that $\mathcal{S} = \mathcal{S}_1 \bigcap \mathcal{S}_2 \neq \emptyset$ and both $\mathcal{S}_1, \mathcal{S}_2$ are prox-regular at $x \in \mathcal{S}$. If $\mathcal{S}$ is metrically calm at x, i.e., if there exist some $\varsigma > 0$ and some neighborhood of x denoted by $\mathcal{B}$ such that $d_{\mathcal{S}}(y) \leq \varsigma(d_{\mathcal{S}_1}(y) + d_{\mathcal{S}_2}(y))$ for all $y \in \mathcal{B}$, then $\mathcal{S}$ is prox-regular at x.*

Proof. (proof to Lemma 4.4) By definition, projection onto $\mathcal{N}$ is single-valued on some open ball $B(x; a)$ with center x and radius $a > 0$ [164, Theorem 1.3]. The set $\mathcal{C}$ is compact and convex, hence projection onto $\mathcal{C}$ is single-valued around every point, hence single-valued on $B(x; a)$ as well [158, Theorem 3.14, Remark 3.15]. Note that for any $y \in B(x; a)$, $d_{\mathcal{X}}(y) = 0$ if and only if both $d_{\mathcal{C}}(y)$ and $d_{\mathcal{N}}(y)$ are zero. Hence, for any $y \in B(x; a) \bigcap \mathcal{X}$, the metrically calmness condition is trivially satisfied. Next, recalling that the distance from a closed set is continuous [60, Example 9.6], over the compact set $\overline{B(x; a) \setminus \mathcal{X}}$, define the function h, such that $h(y) = 1$ if $y \in \mathcal{X}$, and $h(y) = d_{\mathcal{X}}(y)/(d_{\mathcal{C}}(y) + d_{\mathcal{N}}(y))$ else. The function h is upper-semicontinuous over $\overline{B(x; a) \setminus \mathcal{X}}$, hence it will attain a maximum $\varsigma > 0$ over $\overline{B(x; a) \setminus \mathcal{X}}$ [181, Theorem 4.16], thus satisfying the metrically calmness condition on $B(x; a) \setminus \mathcal{X}$ as well. Hence, using Lemma 4.5, the constraint set $\mathcal{X}$ is prox-regular at x. □

4.A.2 Proofs and derivations to the results in §4.3

4.A.2.1 Modifying NExOSfor nonsmooth and convex loss function

We now discuss how to modify NExOS when the loss function is nonsmooth and convex. The key idea is working with a strongly convex, smooth, and arbitrarily close approximation of f;

such smoothing techniques are very common in optimization [159, 182]. The optimization problem in this case, where the positive regularization parameter is denoted by $\widetilde{\beta}$, is given by: $\min_x \phi(x) + (\widetilde{\beta}/2)\|x\|^2 + \iota_{\mathcal{X}}(x)$, where the setup is same as problem ($\mathcal{P}$), except the function $\phi : \mathbb{E} \to \mathbf{R} \cup \{+\infty\}$ is lower-semicontinuous, proper (its domain is nonempty), and convex. Let $\beta := \widetilde{\beta}/2$. For a ν that is arbitrarily small, define the following β strongly convex and $(\nu^{-1} + \beta)$-smooth function: $f := {}^{\nu}\phi(\cdot) + (\beta/2)\|\cdot\|^2$ where ${}^{\nu}\phi$ is the Moreau envelope of ϕ with paramter ν. Following the properties of the Moreau envelope of a convex function discussed in §4.2, the following optimization problem acts as an arbitrarily close approximation to the first nonsmooth convex problem: $\min_x f + (\beta/2)\|x\|^2 + \iota_{\mathcal{X}}(x)$, which has the same setup as problem ($\mathcal{P}$).

We can compute $\mathbf{prox}_{\gamma f}(x)$ using the formula in by [159, Theorem 6.13, Theorem 6.63]. Then, we apply NExOS to $\min_x f + (\beta/2)\|x\|^2 + \iota_{\mathcal{X}}(x)$ and proceed in the same manner as discussed earlier.

4.A.2.2 Proof to Proposition 4.1

Proof to Proposition 4.1(i). We prove (i) in three steps. In the *first step*, we show that for any $\mu > 0$, $f + {}^{\mu}\mathbb{0}$ will be differentiable on some $B(\bar{x}; r_{\text{diff}})$ with $r_{\text{diff}} > 0$. In the *second step*, we then show that, for any $\mu \in (0, 1/\beta]$, $f + {}^{\mu}\mathbb{0}$ will be strongly convex and differentiable on some $B(\bar{x}; r_{\text{cvxdiff}})$. In the *third step*, we will show that there exist $\mu_{\max} > 0$ such that for any $\mu \in (0, \mu_{\max}]$, $f + {}^{\mu}\mathbb{0}$ will be strongly convex and smooth on some $B(\bar{x}; r_{\max})$ and will attain the unique local minimum x_μ in this ball.

Proof of the first step. To prove the first step, we start with the following lemma regarding differentiability of ${}^{\mu}\iota$.

Lemma 4.6 (Differentiability of ${}^{\mu}\iota$)**.** *Let $\bar{x}$ be a local minimum to problem (P), where Assumptions 4.1 and 4.2 hold. Then there exists some $r_{\text{diff}} > 0$ such that for any $\mu > 0$: (i) the function ${}^{\mu}\iota$ is differentiable on $B(\bar{x}; r_{\text{diff}})$ with derivative $\nabla\, {}^{\mu}\iota = (1/\mu)(\mathbb{I} - \mathbf{\Pi})$, and (ii) the projection operator $\mathbf{\Pi}$ onto $\mathcal{X}$ is single-valued and Lipschitz continuous on $B(\bar{x}; r_{\text{diff}})$.*

Proof. From [164, Theorem 1.3(e)], there exists some $r_{\text{diff}} > 0$ such that the function d^2 is differentiable on $B(\bar{x}; r_{\text{diff}})$. As ${}^{\mu}\iota = (1/2\mu)d^2$ from (4.3), it follows that for any $\mu > 0$, ${}^{\mu}\iota$ is differentiable on $B(\bar{x}; r_{\text{diff}})$ which proves the first part of (i). The second part of (i) follows from the fact that $\nabla d^2(x) = 2\,(x - \mathbf{\Pi}(x))$ whenever d^2 is differentiable at x [164, page 5240]. Finally, from [164, Lemma 3.2], whenever d^2 is differentiable at a point, projection $\mathbf{\Pi}$ is single-valued and Lipschitz continuous around that point, and this proves (ii). □

Due to the lemma above, $f + {}^{\mu}\mathbb{1}$ will be differentiable on $B(\bar{x}; r_{\text{diff}})$ with $r_{\text{diff}} > 0$, as f and $(\beta/2)\|\cdot\|^2$ are differentiable. Also, due to Lemma 4.6(ii), projection operator $\mathbf{\Pi}$ is $\widetilde{L}$-Lipschitz continuous on $B(\bar{x}; r_{\text{diff}})$ for some $\widetilde{L} > 0$. This proves the first step.

Proof of the second step. To prove this step, we are going to record: (1) the notion of general subdifferential of a function, followed by (2) the definition of prox-regularity of a function and its connection with prox-regular set, and (3) a helper lemma regarding convexity of the Moreau envelope under prox-regularity.

Definition 4.3 (Fenchel, Fréchet, and general subdifferential)**.** For any lower-semicontinuous function $h : \mathbb{R}^n \to \mathbb{R} \cup \{\infty\}$, its Fenchel subdifferential ∂h is defined as [183, page 1]: $u \in \partial h(x) \Leftrightarrow h(y) \geq h(x) + \langle u \mid y - x\rangle$ for all $y \in \mathbb{R}^n$. For the function h, its Fréchet subdifferential $\partial^F h$ (also known as regular subdifferential) at a point x is defined as [183, Definition 2.5]: $u \in \partial^F h(x) \Leftrightarrow \liminf_{y \to 0} (h(x + y) - h(x) - \langle u \mid y\rangle)/\|y\| \geq 0$. Finally,

the general subdifferential of h, denoted by $\partial^G h$, is defined as [106, Equation (2.8)]: $u \in \partial^G h(x) \Leftrightarrow u_n \to u, x_n \to x, f(x_n) \to f(x)$, for some $(x_n, u_n) \in \mathbf{gra}\, \partial^F h$. If h is additionally convex, then $\partial h = \partial^F h = \partial^G h$ [183, Property (2.3), Property 2.6].

Definition 4.4 (Connection between prox-regularity of a function and a set [184, Definition 1.1])**.** A function $h : \mathbf{R}^n \to \mathbf{R} \cup \{\infty\}$ that is finite at $\tilde{x}$ is prox-regular at $\tilde{x}$ for $\tilde{\nu}$, where $\tilde{\nu} \in \partial^G h(\tilde{x})$, if h is locally l.s.c. at $\tilde{x}$ and there exist a distance $\sigma > 0$ and a parameter $\rho > 0$ such that whenever $\|x' - \tilde{x}\| < \sigma$ and $\|x - \tilde{x}\| < \sigma$ with $x' \neq x$, $\|h(x) - h(\tilde{x})\| < \sigma$, $\|\nu - \tilde{\nu}\| < \sigma$ with $\nu \in \partial^G h(x)$, we have $h(x') > h(x) + \langle \nu \mid x' - x \rangle - (\rho/2)\|x' - x\|^2$. Also, a set $\mathcal{S}$ is prox-regular at $\tilde{x}$ for $\tilde{\nu}$ if we have the indicator function $\iota_{\mathcal{S}}$ is prox-regular at $\tilde{x}$ for $\tilde{\nu} \in \partial^G \iota_{\mathcal{S}}(\tilde{x})$ [184, Proposition 2.11]. The set $\mathcal{S}$ is prox-regular at $\tilde{x}$ if it is prox-regular at $\tilde{x}$ for all $\tilde{\nu} \in \partial^G \iota_{\mathcal{S}}(\tilde{x})$ [60, page 612].

We have the following helper lemma from [184].

Lemma 4.7 ([184, Theorem 5.2])**.** *Consider a function h which is lower semicontinuous at 0 with $h(0) = 0$ and there exists $\rho > 0$ such that $h(x) > -(\rho/2)\|x\|^2$ for any $x \neq 0$. Let h be prox-regular at $\tilde{x} = 0$ and $\tilde{\nu} = 0$ with respect to σ and ρ (σ and ρ as described in Definition 4.4), and let $\lambda \in (0, 1/\rho)$. Then, on some neighborhood of 0, the function*

$$ {}^{\lambda}h + \rho/(2 - 2\lambda\rho)\|\cdot\|^2 \tag{4.10} $$

is convex, where ${}^{\lambda}h$ is the Moreau envelope of h with parameter λ.

Now we start proving step 2 earnestly. To prove this result, we assume $\bar{x} = 0$. This does not cause any loss of generality because this is equivalent to transferring the coordinate origin to the optimal solution and prox-regularity of a set and strong convexity of a function is

invariant under such a coordinate transformation.

First, note that the indicator function of our constraint closed set $\mathcal{X}$ is lower semicontinuous due to [60, Remark after Theorem 1.6, page 11], and as $\bar{x}$, the local minimizer lies in $\mathcal{X}$, we have $\iota_{\mathcal{X}}(\bar{x}) = 0$. The set $\mathcal{X}$ is prox-regular at $\bar{x}$ for all $\nu \in \partial^G \iota_{\mathcal{X}}(x)$ per our setup, so using Definition 4.4, we have $\iota_{\mathcal{X}}$ prox-regular at $\bar{x} = 0$ for $\bar{\nu} = 0 \in \partial^G \iota_{\mathcal{X}}(\bar{x})$ (because $\bar{x} \in \mathcal{X}$, we will have 0 as a subgradient of $\partial \iota_{\mathcal{X}}(\bar{x})$) with respect to some distance $\sigma > 0$ and parameter $\rho > 0$.

Note that the indicator function satisfies $\iota_{\mathcal{X}}(x) = c\iota_{\mathcal{X}}(x)$ for any $c > 0$ due to its definition, so $u \in \partial^G \iota_{\mathcal{X}}(x) \Leftrightarrow cu \in c\partial^G \iota_{\mathcal{X}}(x) = \partial(c\iota_{\mathcal{X}}^G(x)) = \partial \iota_{\mathcal{X}}^G(x)$ [60, Equation 10(6)] In our setup, we have $\mathcal{X}$ prox-regular at $\bar{x}$. So, setting $h \coloneqq \iota_m athcal X, \tilde{x} \coloneqq \bar{x} = 0$, $\tilde{\nu} \coloneqq \bar{\nu} = 0$, and $\nu \coloneqq u/(\beta/2\rho)$ in Definition 4.4, we have $\iota_{\mathcal{X}}$ is also prox-regular at $\bar{x} = 0$ for $\bar{\nu} = 0$ with respect to distance $\sigma \min\{1, \beta/2\rho\}$ and parameter $\beta/2$.

Next, because the range of the indicator function is $\{0, \infty\}$, we have $\iota_{\mathcal{X}}(x) > -(\rho/2)\|x\|^2$ for any $x \neq 0$. So, we have all the conditions of Theorem 4.7 satisfied. Hence, applying Lemma 4.7, we have $(1/2\mu)\left(d^2 + \beta\mu/(2-\beta\mu)\|\cdot\|^2\right)$ convex and differentiable on

$$B\left(\bar{x}; \min\left\{\sigma \min\{1, \beta/2\rho\}, r_{\text{diff}}\right\}\right)$$

for any $\mu \in (0, 2/\beta)$, where r_{diff} comes from Lemma 4.6. As r_{diff} in this setup does not depend on μ, the ball does not depend on μ either. Finally, note that in our exterior-point minimization function we have ${}^{\mu}\square = (1/2\mu)\left(d^2 + \beta\mu\|\cdot\|^2\right)$.

So if we take $\mu \leq \frac{1}{\beta}$, then we have $(\beta/2)\mu/\left(1 - \mu(\beta/2)\right) \leq \beta\mu$, and on the ball $B\left(\bar{x}; \min\left\{\sigma \min\{1, \beta/2\rho\}, r_{\text{diff}}\right.\right.$ the function ${}^{\mu}\square$ will be convex and differentiable. But f is strongly-convex and smooth, so $f + {}^{\mu}\square$ will be strongly convex and differentiable on $B\left(\bar{x}; \min\left\{\sigma \min\{1, \beta/2\rho\}, r_{\text{diff}}\right\}\right)$ for $\mu \in (0, 1/\beta]$. This proves step 2.

Proof of the third step. As point $\bar{x} \in \mathcal{X}$ is a local minimum of problem ($\mathcal{P}$), from Definition 4.2, there is some $r > 0$ such that for all $y \in \overline{B}(\bar{x}; r)$, we have $f(\bar{x}) + (\beta/2)\|\bar{x}\|^2 < f(y) + (\beta/2)\|y\|^2 + \iota(y)$.

Then, due to the first two steps, for any $\mu \in (0, 1/\beta]$, the function $f +^{\mu}\!\!\mathbb{0}$ will be strongly convex and differentiable on $B\left(\bar{x}; \min\left\{\sigma \min\{1, \beta/2\rho\}, r_{\text{diff}}\right\}\right)$. For notational convenience, denote $r_{\max} \coloneqq \min\left\{\sigma \min\{1, \beta/2\rho\}, r_{\text{diff}}\right\}$, which is a constant. As $f +^{\mu}\!\!\mathbb{0}$ is a global underestimator of and approximates the function $f + (\beta/2)\|\cdot\|^2 + \iota$ with arbitrary precision as $\mu \to 0$, the previous statement and [60, Theorem 1.25] imply that there exist some $0 < \mu_{\max} \leq 1/\beta$ such that for any $\mu \in (0, \mu_{\max}]$, the function $f +^{\mu}\!\!\mathbb{0}$ will achieve a local minimum x_μ over $B(\bar{x}; r_{\max})$ where $\nabla(f +^{\mu}\!\!\mathbb{0})$ vanishes, *i.e.*,

$$\nabla(f +^{\mu}\!\!\mathbb{0})(x_\mu) = \nabla f(x_\mu) + \beta x_\mu + (1/\mu)\left(x_\mu - \mathbf{\Pi}\left(x_\mu\right)\right) = 0 \tag{4.11}$$

$$\Rightarrow x_\mu = (1/(\beta\mu + 1))\left(\mathbf{\Pi}(x_\mu) - \mu\nabla f(x_\mu)\right). \tag{4.12}$$

As the right hand side of the last equation is a singleton, this minimum must be unique. Finally to show the smoothness $f +^{\mu}\!\!\mathbb{0}$, for any $x \in B(\bar{x}; r_{\max})$, we have

$$\nabla\left(f +^{\mu}\!\!\mathbb{0}\right)(x) \overset{a)}{=} \nabla f(x) + (\beta + (1/\mu))\, x - (1/\mu)\, \mathbf{\Pi}(x), \tag{4.13}$$

where $a)$ uses Lemma 4.6. Thus, for any $x_1, x_2 \in B(\bar{x}; r_{\max})$ we have $\|\nabla(f + (\beta/2)\|\cdot\|^2 + {}^{\mu}\iota)(x_1) - \nabla(f + (\beta/2)\|\cdot\|^2 + {}^{\mu}\iota)(x_2)\| \leq (L + \beta + (1/\mu) + \widetilde{L})\|x_1 - x_2\|$, where we have used the following: ∇f is L-Lipschitz everywhere due to f being an $L-$smooth function in $\mathbb{E}$ ([158, Theorem 18.15]), and $\mathbf{\Pi}$ is $\widetilde{L}$-Lipschitz continuous on $B(\bar{x}; r_{\max})$, as shown in step 1. This completes the proof for (i).

(ii): Using [60, Theorem 1.25], as $\mu \to 0$, we have $x_\mu \to \bar{x}$, and $(f + {}^{\mu}\mathbb{1})(x_\mu) \to f(\bar{x}) + (\beta/2)\|\bar{x}\|^2$. Note that x_μ reaches $\bar{x}$ only in limit, as otherwise Assumption 4.2 will be violated.

4.A.2.3 Proof to Proposition 4.2

Proof to Proposition 4.2(i). We will need the notions of nonexpansive and firmly nonexpansive operators in this proof. An operator $\mathbb{A} : \mathbb{E} \to \mathbb{E}$ is nonexpansive on some set $\mathcal{S}$ if it is Lipschitz continuous with Lipschitz constant 1 on $\mathcal{S}$; the operator is contractive if the Lipschitz constant is strictly smaller than 1. On the other hand, $\mathbb{A}$ is firmly nonexpansive on $\mathcal{S}$ if and only if its reflection operator $2\mathbb{A} - \mathbb{I}$ is nonexpansive on $\mathcal{S}$. A firmly nonexpansive operator is always nonexpansive [158, page 59].

We next introduce the following definition.

Definition 4.5 (Resolvent and reflected resolvent [158, pages 333, 336])**.** For a lower-semicontinuous, proper, and convex function h, the resolvent and reflected resolvent of its subdifferential operator are defined by $\mathbb{J}_{\gamma\partial h} = (\mathbb{I} + \gamma\partial h)^{-1}$ and $\mathbb{R}_{\gamma\partial h} = 2\mathbb{J}_{\gamma\partial h} - \mathbb{I}$, respectively.

The proof of (i) is proven in two steps. First, we show that the reflection operator of $\mathbb{T}_\mu$, defined by

$$\mathbb{R}_\mu = 2\mathbb{T}_\mu - \mathbb{I}, \tag{4.14}$$

is contractive on $B(\bar{x}, r_{\max})$, and using this we show that $\mathbb{T}_\mu$ in also contractive there in the second step. To that goal, note that $\mathbb{R}_\mu$ can be represented as:

$$\mathbb{R}_\mu = (2\mathbf{prox}_{\gamma\,{}^{\mu}\mathbb{1}} - \mathbb{I})(2\mathbf{prox}_{\gamma f} - \mathbb{I}), \tag{4.15}$$

which can be proven by simply using (4.5) and (4.14) on the left-hand side and by expanding the factors on the right-hand side. Now, the operator $2\mathbf{prox}_{\gamma f} - \mathbb{I}$ associated with the

α-strongly convex and L-smooth function f is a contraction mapping for any $\gamma > 0$ with the contraction factor $\kappa = \max\{(\gamma L - 1)/(\gamma L + 1), (1 - \gamma\alpha)/(\gamma\alpha + 1)\} \in (0, 1)$, which follows from [171, Theorem 1]. Next, we show that $2\mathbf{prox}_{\gamma\, {}^{\mu}\mathbb{0}} - \mathbb{I}$ is nonexpansive on $B(\bar{x}; r_{\max})$ for any $\mu \in (0, \mu_{\max}]$. For any $\mu \in (0, \mu_{\max}]$, define the function g as follows. We have $g(y) = {}^{\mu}\mathbb{0}(y)$ if $y \in B(\bar{x}; r_{\max})$,$g(y) = \liminf_{\tilde{y}\to y} {}^{\mu}\mathbb{0}(\tilde{y})$ if $\|y - \bar{x}\| = r_{\max}$, and $g(y) = \infty$ else. The function g is lower-semicontinuous, proper, and convex everywhere due to [158, Lemma 1.31 and Corollary 9.10]. As a result for $\mu \in (0, \mu_{\max}]$, we have $\mathbf{prox}_{\gamma g} = \mathbb{J}_{\gamma\partial g}$ on $\mathbf{E}$ and $\mathbf{prox}_{\gamma g}$ is firmly nonexpansive and single-valued everywhere, which follows from [158, Proposition 12.27, Proposition 16.34, and Example 23.3]. But, for $y \in B(\bar{x}; r_{\max})$, we have ${}^{\mu}\mathbb{0}(y) = g(y)$ and $\nabla\, {}^{\mu}\mathbb{0}(y) = \partial g(y)$. Thus, on $B(\bar{x}; r_{\max})$, the operator $\mathbf{prox}_{\gamma\, {}^{\mu}\mathbb{0}} = \mathbb{J}_{\gamma\nabla\, {}^{\mu}\mathbb{0}}$, and it is firmly nonexpansive and single-valued for $\mu \in (0, \mu_{\max}]$. Any firmly nonexpansive operator $\mathbb{A}$ has a nonexpansive reflection operator $2\mathbb{A} - \mathbb{I}$ on its domain of firm nonexpansiveness [158, Proposition 4.2]. Hence, on $B(\bar{x}; r_{\max})$, for $\mu \in (0, \mu_{\max}]$ the operator $2\mathbf{prox}_{\gamma\, {}^{\mu}\mathbb{0}} - \mathbb{I}$ is nonexpansive using (4.15).

Now we show that $\mathbb{R}_\mu$ is contractive for every $x_1, x_2 \in B(\bar{x}; r_{\max})$ and $\mu \in (0, \mu_{\max}]$, we have $\|\mathbb{R}_\mu(x_1) - \mathbb{R}_\mu(x_2)\| \leq \|(2\mathbf{prox}_{\gamma f} - \mathbb{I})(x_1) - (2\mathbf{prox}_{\gamma f} - \mathbb{I})(x_2)\| \leq \kappa\|x_1 - x_2\|$ where the last inequality uses κ-contractiveness of $2\mathbf{prox}_{\gamma f} - \mathbb{I}$ thus proving that $\mathbb{R}_\mu$ acts as a contractive operator on $B(\bar{x}; r_{\max})$ for $\mu \in (0, \mu_{\max}]$. Similarly, for any $x_1, x_2 \in B(\bar{x}; r_{\max})$, using $(\mathcal{A}_\mu)$ and the triangle inequality we have $\|\mathbb{T}_\mu(x_1) - \mathbb{T}_\mu(x_2)\| \leq (1+\kappa)/2\|x_1 - x_2\|$ and as $\kappa' = (1 + \kappa)/2 \in [0, 1)$; the operator $\mathbb{T}_\mu$ is $\kappa'-$contractive on on $B(\bar{x}; r_{\max})$, for $\mu \in (0, \mu_{\max}]$.

Proof to Proposition 4.2(ii). Recalling $\mathbb{T}_\mu = (1/2)\mathbb{R}_\mu + (1/2)\mathbb{I}$ from (4.14), using (4.15), and then expanding, and finally using Lemma 4.1 and triangle inequality, we have for any $\mu, \tilde{\mu} \in (0, \mu_{\max}]$, $x \in B(\bar{x}; r_{\max})$, and $y = 2\mathbf{prox}_{\gamma f}(x) - x$:

$$\|\mathbb{T}_\mu(x) - \mathbb{T}_{\tilde{\mu}}(x)\| \leq \|(\mu/(\gamma + \mu(\beta\gamma + 1)) - \tilde{\mu}/(\gamma + \tilde{\mu}(\beta\gamma + 1)))\|\, \|y\|$$

$$+ \|(\gamma/(\gamma+\mu(\beta\gamma+1)) - \gamma/(\gamma+\tilde{\mu}(\beta\gamma+1)))\| \, \|\mathbf{\Pi}\,(y/(\beta\gamma+1))\|\,. \quad (4.16)$$

Now, in (4.16), the coefficient of $\|y\|$ satisfies $\|\mu/(\gamma+\mu(\beta\gamma+1)) - \tilde{\mu}/(\gamma+\tilde{\mu}(\beta\gamma+1))\| \leq (1/\gamma)\|\mu-\tilde{\mu}\|$

and similarly the coefficient of $\|\,\mathbf{\Pi}\,(y/(\beta\gamma+1))\,\|$ satisfies

$$\|\gamma/(\gamma+\mu(\beta\gamma+1)) - \gamma/(\gamma+\tilde{\mu}(\beta\gamma+1))\| \leq (\beta+(1/\gamma))\|\mu-\tilde{\mu}\|.$$

Putting the last two inequalities in (4.16), and then replacing $y = 2\mathbf{prox}_{\gamma f}(x) - x$, we have for any $x \in \mathcal{B}$, and for any $\mu, \tilde{\mu} \in \mathbb{R}_{++}$,

$$\begin{aligned}\|\mathbb{T}_\mu(x) - \mathbb{T}_{\tilde{\mu}}(x)\| &\leq (1/\gamma)\,\|\mu-\tilde{\mu}\|\,\|y\| + (\beta+(1/\gamma))\,\|\mu-\tilde{\mu}\|\,\|\mathbf{\Pi}\,(y/(\beta\gamma+1))\| \\ =&\{(1/\gamma)\|2\mathbf{prox}_{\gamma f}(x)-x\| + (\beta+(1/\gamma))\|\,\mathbf{\Pi}((2\mathbf{prox}_{\gamma f}(x)-x)/(\beta\gamma+1))\|\}\|\mu-\tilde{\mu}\|. \quad (4.17)\end{aligned}$$

Now, as $B(\bar{x}; r_{\max})$ is a bounded set and $x \in \mathcal{B}$, norm of the vector $y = 2\mathbf{prox}_{\gamma f}(x) - x$ can be upper-bounded over $B(\bar{x}; r_{\max})$ because $2\mathbf{prox}_{\gamma f} - \mathbb{I}$ is continuous (in fact contractive) as shown in (i). Similarly, $\|\,\mathbf{\Pi}\left((2\mathbf{prox}_{\gamma f}(x)-x)/(\beta\gamma+1)\right)\|$ can be upper-bounded on $B(\bar{x}; r_{\max})$. Combining the last two-statements, it follows that there exists some $\ell > 0$ such that

$$\sup_{x \in B(\bar{x}; r_{\max})} (1/\gamma)\|2\mathbf{prox}_{\gamma f}(x)-x\| + (\beta+1/\gamma)\left\|\mathbf{\Pi}\left((2\mathbf{prox}_{\gamma f}(x)-x)/(\beta\gamma+1)\right)\right\| \leq \ell,$$

and putting the last inequality in (4.17), we arrive at the claim.

4.A.2.4 Proof to Proposition 4.3

The structure of the proof follows that of [158, Proposition 25.1(ii)]. Let $\mu \in (0, \mu_{\max}]$. Recalling Definition 4.5, and due to Proposition 4.1(i), $x_\mu \in B(\bar{x}; r_{\max})$ satisfies

$$\begin{aligned} x_\mu &= \underset{B(\bar{x};r_{\max})}{\operatorname{argmin}} \; f(x) + {}^{\mu}\mathbb{0}(x) = \mathbf{zer}(\nabla f + \nabla\, {}^{\mu}\mathbb{0}) \\ &\overset{a)}{\Leftrightarrow} (\exists y \in \mathbf{E}) \;\; x_\mu = \mathbb{J}_{\gamma \nabla\, {}^{\mu}\mathbb{0}} \mathbb{R}_{\gamma \nabla f}(y) \text{ and } x_\mu = \mathbb{J}_{\gamma \nabla f}(y), \end{aligned} \tag{4.18}$$

where $a)$ uses the facts (shown in the proof to Proposition 4.2) that: (i) $\mathbb{J}_{\gamma\nabla f}$ is a single-valued operator everywhere, whereas $\mathbb{J}_{\gamma \nabla\, {}^{\mu}\mathbb{0}}$ is a single-valued operator on the region of convexity $B(\bar{x}; r_{\max})$, and (ii) $x_\mu = \mathbb{J}_{\gamma\nabla f}(y)$ can be expressed as $x_\mu = \mathbb{J}_{\gamma\nabla f}(y) \Leftrightarrow 2x_\mu - y = (2\mathbb{J}_{\gamma\nabla f} - \mathbb{I})\, y = \mathbb{R}_{\gamma\nabla f}(y)$. Also, using the last expression, we can write the first term of (4.18) as $\mathbb{J}_{\gamma \nabla\, {}^{\mu}\mathbb{0}} \mathbb{R}_{\gamma\nabla f}(y) = x_\mu \Leftrightarrow y \in \mathbf{fix}(\mathbb{R}_{\gamma \nabla\, {}^{\mu}\mathbb{0}} \mathbb{R}_{\gamma\nabla f})$. Because for lower-semicontinuous, proper, and convex function, the resolvent of the subdifferential is equal to its proximal operator [158, Proposition 12.27, Proposition 16.34, and Example 23.3], we have $\mathbb{J}_{\gamma\partial f} = \mathbf{prox}_{\gamma f}$ with both being single-valued. Using the last fact along with (4.18), $y \in \mathbf{fix}(\mathbb{R}_{\gamma \nabla\, {}^{\mu}\mathbb{0}} \mathbb{R}_{\gamma\nabla f})$, we have $x_\mu \in \mathbf{prox}_{\gamma f}\left(\mathbf{fix}\left(\mathbb{R}_{\gamma \nabla\, {}^{\mu}\mathbb{0}} \mathbb{R}_{\gamma\partial f}\right)\right)$, but x_μ is unique due to Proposition 4.1, so the inclusion can be replaced with equality. Thus x_μ, satisfies $x_\mu = \mathbf{prox}_{\gamma f}\left(\mathbf{fix}\left(\mathbb{R}_{\gamma \nabla\, {}^{\mu}\mathbb{0}} \mathbb{R}_{\gamma\partial f}\right)\right)$ where the sets are singletons due to Proposition 4.1 and single-valuedness of $\mathbf{prox}_{\gamma f}$. Also, because $\mathbb{T}_\mu$ in (4.5) and $\mathbb{R}_\mu$ in (4.14) have the same fixed point set (follows from (4.14)), using (4.15), we arrive at the claim.

4.A.2.5 Proof to Lemma 4.2

(i): This follows directly from the proof to Proposition 4.1.

(ii): From Lemma 4.2(i), and recalling that $\eta' > 1$, for any $\mu \in (0, \mu_{\max}]$, we have the first equation. Recalling Definition 4.5, and using the fact that for lower-semicontinuous, proper,

and convex function, the resolvent of the subdifferential is equal to its proximal operator [158, Proposition 12.27, Proposition 16.34, and Example 23.3], we have $\mathbb{J}_{\gamma\partial f} = \mathbf{prox}_{\gamma f}$ with both being single-valued. So, from Proposition 4.3: $x_\mu = \mathbf{prox}_{\gamma f}(z_\mu) = (\mathbb{I} + \gamma\partial f)^{-1}(z_\mu) \Leftrightarrow z_\mu = x_\mu + \gamma\nabla f(x_\mu)$. Hence, for any $\mu \in (0, \mu_{\max}]$:

$$\begin{aligned}
&\|z_\mu - \bar{x}\| = \|x_\mu + \gamma\nabla f(x_\mu) - \bar{x}\| \leq \|x_\mu - \bar{x}\| + \gamma\|\nabla f(x_\mu)\| \\
\Leftrightarrow &r_{\max} - \|z_\mu - \bar{x}\| \geq r_{\max} - \|x_\mu - \bar{x}\| - \gamma\|\nabla f(x_\mu)\| \overset{a)}{\geq} (\eta' - 1)r_{\max}/\eta' - \gamma\|\nabla f(x_\mu)\|,
\end{aligned}$$

where $a)$ uses the first equation of Lemma 4.2(ii). Because, for the strongly convex and smooth function f, its gradient is bounded over a bounded set $B(\bar{x}; r_{\max})$ [185, Lemma 1, §1.4.2], then for γ satisfying the fourth equation of Lemma 4.2(ii) and the definition of ψ in the third equation of Lemma 4.2(ii), we have the second equation of Lemma 4.2(ii) for any $\mu \in (0, \mu_{\max}]$. To prove the final equation of Lemma 4.2(ii), note that

$$\begin{aligned}
&\lim_{\mu\to 0}(r_{\max} - \|z_\mu - \bar{x}\|) - \psi \\
&\overset{a)}{=} \lim_{\mu\to 0}(r_{\max} - \|x_\mu + \gamma\nabla f(x_\mu) - \bar{x}\|) - (\eta' - 1)r_{\max}/\eta' + \gamma \max_{x\in B(\bar{x};r_{\max})}\|\nabla f(x)\| \\
&\overset{b)}{=} (r_{\max} - \|\bar{x} + \gamma\nabla f(\bar{x}) - \bar{x}\|) - (\eta' - 1)r_{\max}/\eta' + \gamma \max_{x\in B(\bar{x};r_{\max})}\|\nabla f(x)\| \\
&= (1/\eta')r_{\max} + \gamma\left(\max_{x\in B(\bar{x};r_{\max})}\|\nabla f(x)\| - \|\nabla f(\bar{x})\|\right) > 0, \qquad (4.19)
\end{aligned}$$

where in $a)$ we have used $z_\mu = x_\mu + \gamma\nabla f(x_\mu)$ and the third equation of Lemma 4.2(ii), in $b)$ we have used smoothness of f along with Proposition 4.1(ii). Inequality (4.19) along with the second equation of Lemma 4.2(ii) implies the final equation of Lemma 4.2(ii).

4.A.2.6 Proof to Theorem 4.1

We use the following result from [186] in proving Theorem 4.1.

Theorem 4.3 (Convergence of local contraction mapping [186, pp. 313-314]). *Let $\mathbb{A} : \mathbb{E} \to \mathbb{E}$ be some operator. If there exist $\tilde{x}$, $\omega \in (0,1)$, and $r > 0$ such that (a) $\mathbb{A}$ is ω-contractive on $B(\tilde{x}; r)$, i.e., for all x_1, x_2 in $B(\tilde{x}; r)$, and (b) $\|\mathbb{A}(\tilde{x}) - \tilde{x}\| \leq (1-\omega)r$. Then $\mathbb{A}$ has a unique fixed point in $B(\tilde{x}; r)$ and the iteration scheme $x_{n+1} = \mathbb{A}(x_n)$ with the initialization $x_0 := \tilde{x}$ linearly converges to that unique fixed point.*

Furthermore, recall that NExOS (Algorithm 2) can be compactly represented using ($\mathcal{A}_\mu$) as follows. For any $m \in \{1, 2, \ldots, N\}$ (equivalently for each $\mu_m \in \{\mu_1, \ldots, \mu_N\}$),

$$z_{\mu_m}^{n+1} = \mathbb{T}_{\mu_m}\left(z_{\mu_m}^{n}\right), \tag{4.20}$$

where $z_{\mu_m}^0$ is initialized at $z_{\mu_{m-1}}$. From Proposition 4.2, for any $\mu \in \mathfrak{M}$, the operator $\mathbb{T}_\mu$ is a κ'-contraction mapping over the region of convexity $B(\bar{x}; r_{\max})$, where $\kappa' \in (0,1)$. From Proposition 4.1, there will be a unique local minimum x_μ of problem ($\mathcal{P}_\mu$) over $B(\bar{x}; r_{\max})$. Suppose, instead of the exact fixed point $z_{\mu_{m-1}} \in \mathbf{fix}\, \mathbb{T}_{\mu_{m-1}}$, we have computed $\widetilde{z}$, which is an ϵ-approximate fixed point of $\mathbb{T}_{\mu_{m-1}}$ in $B(\bar{x}; r_{\max})$, i.e., $\|\widetilde{z} - \mathbb{T}_{\mu_{m-1}}(\widetilde{z})\| \leq \epsilon$ and $\|\widetilde{z} - z_{\mu_{m-1}}\| \leq \epsilon$, where $\epsilon \in [0, \bar{\epsilon})$. Then, we have:

$$\|\mathbb{T}_{\mu_{m-1}}(\widetilde{z}) - z_{\mu_{m-1}}\| = \|\mathbb{T}_{\mu_{m-1}}(\widetilde{z}) - \mathbb{T}_{\mu_{m-1}}(z_{\mu_{m-1}})\| \overset{a)}{\leq} \kappa' \underbrace{\|\widetilde{z} - z_{\mu_{m-1}}\|}_{\leq \epsilon} \leq \epsilon, \tag{4.21}$$

where $a)$ uses κ'-contractive nature of $\mathbb{T}_{\mu_{m-1}}$ over $B(\bar{x}; r_{\max})$. Hence, using triangle inequality,

$$\|\widetilde{z} - \bar{x}\| \overset{a)}{\leq} \|\widetilde{z} - \mathbb{T}_{\mu_{m-1}}(\widetilde{z})\| + \|\mathbb{T}_{\mu_{m-1}}(\widetilde{z}) - z_{\mu_{m-1}}\| + \|z_{\mu_{m-1}} - \bar{x}\| \overset{b)}{\leq} 2\epsilon + \|z_{\mu_{m-1}} - \bar{x}\|,$$

where $a)$ uses triangle inequality and $b)$ uses (4.21). As $\epsilon \in [0, \bar{\epsilon})$, where $\bar{\epsilon}$ is defined in (4.6), due to the second equation of Lemma 4.2(ii), we have $r_{\max} - \|\widetilde{z} - \bar{x}\| > \psi$.

Define $\Delta = ((1-\kappa')\psi - \epsilon)/\ell$, which will be positive due to $\epsilon \in [0, \bar{\epsilon})$ and (4.6). Next, select $\theta \in (0,1)$ such that $\overline{\Delta} = \theta\Delta < \mu_1$, hence there exists a $\rho \in (0,1)$ such that $\overline{\Delta} = (1-\rho)\mu_1$. Now reduce the penalty parameter using

$$\mu_m = \mu_{m-1} - \rho^{m-2}\overline{\Delta} = \rho\mu_{m-1} = \rho^{m-1}\mu_1 \tag{4.22}$$

for any $m \geq 2$. Next, we initialize the iteration scheme $z_{\mu_m}^{n+1} = \mathbb{T}_{\mu_m}\left(z_{\mu_m}^n\right)$ at $z_{\mu_m}^0 := \widetilde{z}$. Around this initial point, let us consider the open ball $B(\widetilde{z}, \psi)$. For any $x \in B(\widetilde{z}; \psi)$, we have $\|x - \bar{x}\| \leq \|x - \widetilde{z}\| + \|\widetilde{z} - \bar{x}\| < \psi + \|\widetilde{z} - \bar{x}\| < r_{\max}$, where the last inequality follows from $r_{\max} - \|\widetilde{z} - \bar{x}\| > \psi$. Thus we have shown that $B(\widetilde{z}; \psi) \subseteq B(\bar{x}; r_{\max})$. Hence, from Proposition 4.2, on $B(\widetilde{z}; \psi)$, the Douglas-Rachford operator $\mathbb{T}_{\mu_m}$ is contractive. Next, we have $\|\mathbb{T}_{\mu_m}(\widetilde{z}) - \widetilde{z}\| \leq (1-\kappa')\psi$, because $\|\mathbb{T}_{\mu_m}(\widetilde{z}) - \widetilde{z}\| \overset{a)}{\leq} \|\mathbb{T}_{\mu_m}(\widetilde{z}) - \mathbb{T}_{\mu_{m-1}}(\widetilde{z})\| + \|\mathbb{T}_{\mu_{m-1}}(\widetilde{z}) - \widetilde{z}\| \overset{b)}{\leq} \ell\|\mu_m - \mu_{m-1}\| + \epsilon \overset{c)}{\leq} \epsilon + \ell\Delta \overset{d)}{\leq} (1-\kappa')\psi$, where $a)$ triangle inequality, $b)$ uses Proposition 4.2(ii) and $\|\widetilde{z} - \mathbb{T}_{\mu_{m-1}}(\widetilde{z})\| \leq \epsilon$, $c)$ uses (4.22) and $\|\mu_m - \mu_{m-1}\| \leq \overline{\Delta} \leq \Delta$ $d)$ uses the definition of Δ. Thus, both conditions of Theorem 4.3 are satisfied, and $z_{\mu_m}^n$ in (4.20) will linearly converge to the unique fixed point $z_{\mu_m}$of the operator $\mathbb{T}_{\mu_m}$, and $x_{\mu_m}^n, y_{\mu_m}^n$ will linearly converge to x_{μ_m}. This completes the proof.

4.A.2.7 Proof to Lemma 4.3

First, we show that, for the given initialization of z_{init}, the iterates $z_{\mu_1}^n$ stay in $\overline{B}(z_{\mu_1}; \|z_{\text{init}} - z_{\mu_1}\|)$ for any $n \in \mathbb{N}$ via induction. The base case is true via given. Let, $z_{\mu_1}^n \in \overline{B}(z_{\mu_1}; \|z_{\text{init}} - z_{\mu_1}\|)$. Then, $\|z_{\mu_1}^{n+1} - z_{\mu_1}\| \overset{a)}{=} \|\mathbb{T}_{\mu_1}(z_{\mu_1}^n) - \mathbb{T}_{\mu_1}(z_{\mu_1})\| \overset{b)}{\leq} \kappa'\|z_{\mu_1}^n - z_{\mu_1}\| \overset{c)}{\leq} \kappa'\|z_{\text{init}} - z_{\mu_1}\|$, where $a)$ uses $z_{\mu_1} \in \mathbf{fix}\,\mathbb{T}_\mu$, and $b)$ uses Proposition 4.2, and $c)$ uses $\|z_{\mu_1}^n - z_{\mu_1}\| \leq \|z_{\text{init}} - z_{\mu_1}\|$. So, the iterates $z_{\mu_1}^n$ stay in $\overline{B}(z_{\mu_1}; \|z_{\text{init}} - z_{\mu_1}\|)$. As, $\kappa' \in (0,1)$, this inequality also implies that z_μ^n linearly converges to z_μ with the rate of at least κ'. Then using similar reasoning presented in the proof to Theorem 4.1, we have x_μ^n and y_μ^n linearly converge to the unique local minimum

x_μ of problem ($\mathcal{P}_\mu$). This completes the proof.

4.A.2.8 Proof to Theorem 4.2

The proof is based on the results in [187, Theorem 4] and [162, Theorem 4.3]. The function f is L-Lipschitz continuous and strongly smooth, hence f is a coercive function satisfying $\liminf_{\|x\|\to\infty} f(x) = \infty$ and is bounded below [158, Corollary 11.17]. Also, ${}^{\mu}\!\square(x)$ is jointly continuous hence lower-semicontinuous in x and μ and is bounded below by definition. Let the proximal parameter γ be smaller than or equal to $1/L$. Then due to [187, (14), (15) and Theorem 4], $\{x_\mu^n, y_\mu^n, z_\mu^n\}$ (iterates of the inner algorithm of NExOS for any penalty parameter μ) will be bounded. This boundedness implies the existence of a cluster point of the sequence, which allows us to use [187, Theorem 4 and Theorem 1] to show that for any z_{init}, the iterates x_μ^n and y_μ^n subsequentially converges to a first-order stationary point x_μ satisfying $\nabla\left(f + {}^{\mu}\!\square\right)(x_\mu) = 0$. The rate $\min_{n\le k} \|\nabla\left(f + {}^{\mu}\!\square\right)(x_{\rho\mu}^n)\| \le ((1-\gamma L)/2L)o(1/\sqrt{k})$ is a direct application of [162, Theorem 4.3] as our setup satisfies all the conditions to apply it.

References

[1] Shuvomoy Das Gupta, Bart P.G. Van Parys, and Ernest K. Ryu. "Branch-and-bound performance estimation programming: a unified methodology for constructing optimal optimization methods". *Mathematical Programming* (2023), pp. 1–73.

[2] Shuvomoy Das Gupta, Robert M. Freund, Xu Andy Sun, and Adrien B. Taylor. "Nonlinear conjugate gradient methods: worst-case convergence rates via computer-assisted analyses". *arXiv preprint arXiv:2301.01530* (2023).

[3] Shuvomoy Das Gupta, Bartolomeo Stellato, and Bart P.G. Van Parys. "Exterior-point optimization for sparse and low-rank optimization". *arXiv preprint arXiv:2011.04552* (2023).

[4] Yoel Drori and Marc Teboulle. "Performance of first-order methods for smooth convex minimization: A novel approach". *Mathematical Programming* 145.1-2 (2014), pp. 451–482.

[5] Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. "Smooth strongly convex interpolation and exact worst-case performance of first-order methods". *Mathematical Programming* 161.1 (2017), pp. 307–345.

[6] Adrien B. Taylor, Julien M Hendrickx, and François Glineur. "Exact worst-case performance of first-order methods for composite convex optimization". *SIAM Journal on Optimization* 27.3 (2017), pp. 1283–1313.

[7] Adrien B. Taylor, Bryan Van Scoy, and Laurent Lessard. "Lyapunov functions for first-order methods: Tight automated convergence guarantees". *International Conference on Machine Learning* (2018).

[8] Donghwan Kim and Jeffrey A Fessler. "Optimized first-order methods for smooth convex minimization". *Mathematical Programming* 159.1 (2016), pp. 81–107.

[9] Adrien B. Taylor and Yoel Drori. "An optimal gradient method for smooth strongly convex minimization". *Mathematical Programming* (2022), pp. 1–38.

[10] Laurent Lessard, Benjamin Recht, and Andrew Packard. "Analysis and design of optimization algorithms via integral quadratic constraints". *SIAM Journal on Optimization* 26.1 (2016), pp. 57–95.

[11] Boris T. Polyak. "The conjugate gradient method in extremal problems". *USSR Computational Mathematics and Mathematical Physics* 9.4 (1969), pp. 94–112.

[12] Elijah Polak and Gérard Ribière. "Note sur la convergence de méthodes de directions conjuguées". *Revue française d'informatique et de recherche opérationnelle. Série rouge* 3.16 (1969), pp. 35–43.

[13] Roger Fletcher and Colin M. Reeves. "Function minimization by conjugate gradients". *The computer journal* 7.2 (1964), pp. 149–154.

[14] Magnus R. Hestenes and Eduard Stiefel. "Methods of conjugate gradients for solving linear systems". *Journal of Research of the National Bureau of Standards* 49.6 (1952), p. 409.

[15] Roger Fletcher. *Practical Methods of Optimization Volume 1: Unconstrained Optimization*. John Wiley & Sons, 1987.

[16] Yu-Hong Dai and Yaxiang Yuan. "A nonlinear conjugate gradient method with a strong global convergence property". *SIAM Journal on Optimization* 10.1 (1999), pp. 177–182.

[17] Trevor Hastie, Robert Tibshirani, and Martin Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman and Hall/CRC, 2019.

[18] Prateek Jain and Purushottam Kar. "Non-convex optimization for machine learning". *Foundations and Trends® in Machine Learning* 10.3-4 (2017), pp. 142–336.

[19] Emmanuel Candès, Michael B Wakin, and Stephen Boyd. "Enhancing sparsity by reweighted l1 minimization". *Journal of Fourier Analysis and Applications* (2008). ISSN: 10695869.

[20] Joel A. Tropp. "Just relax: Convex programming methods for identifying sparse signals in noise". *IEEE Transactions on Information Theory* (2006). ISSN: 00189448.

[21] Dimitris Bertsimas and Bart P.G. Van Parys. "Sparse hierarchical regression with polynomials". *Machine Learning* (2020). ISSN: 15730565. DOI: 10.1007/s10994-020-05868-6. eprint: 1709.10030.

[22] Dimitris Bertsimas, Angela King, and Rahul Mazumder. "Best subset selection via a modern optimization lens". *The Annals of Statistics* 44.2 (2016), pp. 813–852.

[23] Rahul Mazumder, Trevor Hastie, and Robert Tibshirani. "Spectral regularization algorithms for learning large incomplete matrices". *Journal of Machine Learning Research* (2010). ISSN: 15324435.

[24] Emmanuel Candès and Benjamin Recht. "Exact matrix completion via convex optimization". *Foundations of Computational Mathematics* (2009). ISSN: 16153375.

[25] Kwang-Sung Jun, Rebecca Willett, Stephen Wright, and Robert Nowak. "Bilinear bandits with low-rank structure". In: *International Conference on Machine Learning*. PMLR. 2019, pp. 3163–3172.

[26] Aubrey Gress and Ian Davidson. "A flexible framework for projecting heterogeneous data". In: *CIKM 2014 - Proceedings of the 2014 ACM International Conference on Information and Knowledge Management*. 2014. ISBN: 9781450325981. DOI: 10.1145/2661829.2662030.

[27] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. "Distributed representations of words and phrases and their compositionality". In: *Advances in Neural Information Processing Systems*. 2013. eprint: 1310.4546.

[28] Vivek Srikumar and Christopher D Manning. "Learning distributed representations for structured output prediction". In: *Advances in Neural Information Processing Systems*. 2014.

[29] Francis Bach. "Sharp analysis of low-rank kernel matrix approximations". *Journal of Machine Learning Research* (2013). eprint: 1208.2015.

[30] Yurii Nesterov. "A method of solving a convex programming problem with convergence rate $O(1/k^2)$". *Soviet Mathematics Doklady* 27.2 (1983), pp. 372–376.

[31] Arkadi Nemirovsky. "Information-based complexity of linear operator equations". *Journal of Complexity* 8.2 (1992), pp. 153–175.

[32] Arkadi Nemirovski. "Information-based complexity of convex programming". *Lecture Notes, Technion - Israel Institute of Technology* (1995).

[33] Bryan Van Scoy, Randy A Freeman, and Kevin M Lynch. "The fastest known globally convergent first-order method for minimizing strongly convex functions". *IEEE Control Systems Letters* 2.1 (2017), pp. 49–54.

[34] Donghwan Kim and Jeffrey A Fessler. "Optimizing the efficiency of first-order methods for decreasing the gradient of smooth convex functions". *Journal of Optimization Theory and Applications* 188.1 (2021), pp. 192–219.

[35] Donghwan Kim. "Accelerated proximal point method for maximally monotone operators". *Mathematical Programming* 190.1–2 (2021), pp. 57–87.

[36] Felix Lieder. "On the convergence rate of the Halpern-iteration". *Optimization Letters* 15.2 (2021), pp. 405–418.

[37] TaeHo Yoon and Ernest K. Ryu. "Accelerated Algorithms for Smooth Convex-Concave Minimax Problems with $\mathcal{O}(1/k^2)$ Rate on Squared Gradient Norm". *International Conference on Machine Learning* (2021).

[38] Adrien B. Taylor and Francis Bach. "Stochastic first-order methods: Non-asymptotic and computer-aided analyses via potential functions". *Conference on Learning Theory* (2019).

[39] Yoel Drori. "The exact information-based complexity of smooth convex minimization". *Journal of Complexity* 39 (2017), pp. 1–16.

[40] Yoel Drori and Adrien B. Taylor. “Efficient first-order methods for convex minimization: A constructive approach”. *Mathematical Programming* 184.1 (2020), pp. 183–220.

[41] Yoel Drori and Adrien B. Taylor. “On the oracle complexity of smooth strongly convex minimization”. *Journal of Complexity* 68 (2022), p. 101590.

[42] Saman Cyrus, Bin Hu, Bryan Van Scoy, and Laurent Lessard. “A Robust Accelerated Optimization Algorithm for Strongly Convex Functions”. *American Control Conference* (2018).

[43] M. Barré, A. B. Taylor, and F. Bach. “Principled Analyses and Design of First-Order Methods with Inexact Proximal Operators”. *Mathematical Programming* (2022).

[44] Etienne de Klerk, François Glineur, and Adrien B. Taylor. “Worst-case convergence analysis of inexact gradient and Newton methods through semidefinite programming performance estimation”. *SIAM Journal on Optimization* 30.3 (2020), pp. 2053–2082.

[45] Donghwan Kim and Jeffrey A Fessler. “Another look at the fast iterative shrinkage/thresholding algorithm (FISTA)”. *SIAM Journal on Optimization* 28.1 (2018), pp. 223–250.

[46] Adrien B. Taylor, Julien M Hendrickx, and François Glineur. “Exact worst-case convergence rates of the proximal gradient method for composite convex minimization”. *Journal of Optimization Theory and Applications* 178.2 (2018), pp. 455–476.

[47] Etienne de Klerk, François Glineur, and Adrien B. Taylor. “On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions”. *Optimization Letters* 11.7 (2017), pp. 1185–1199.

[48] Ernest K. Ryu, Adrien B. Taylor, Carolina Bergeling, and Pontus Giselsson. “Operator splitting performance estimation: Tight contraction factors and optimal parameter selection”. *SIAM Journal on Optimization* 30.3 (2020), pp. 2251–2271.

[49] Guoyong Gu and Junfeng Yang. “Tight Sublinear Convergence Rate of the Proximal Point Algorithm for Maximal Monotone Inclusion Problems”. *SIAM Journal on Optimization* 30.3 (2020), pp. 1905–1921.

[50] R.-A. Dragomir, Adrien B. Taylor, A. d’Aspremont, and J. Bolte. “Optimal Complexity and Certification of Bregman First-Order Methods”. *Mathematical Programming* 194.1–2 (2022), pp. 41–83.

[51] Hadi Abbaszadehpeivasti, Etienne de Klerk, and Moslem Zamani. “The exact worst-case convergence rate of the gradient method with fixed step lengths for L-smooth functions”. *Optimization Letters* 16.6 (2022), pp. 1649–1661.

[52] Damek Davis and Dmitriy Drusvyatskiy. “Stochastic model-based minimization of weakly convex functions”. *SIAM Journal on Optimization* 29.1 (2019), pp. 207–239.

[53] Iain Dunning, Joey Huchette, and Miles Lubin. "JuMP: A Modeling Language for Mathematical Optimization". *SIAM Review* 59.2 (2017), pp. 295–320. DOI: 10.1137/15M1020575.

[54] MOSEK ApS. *MOSEK Optimizer API for C 9.3.6*. 2019. URL: https://docs.mosek.com/latest/capi/index.html.

[55] Andreas Wächter and Lorenz T. Biegler. "Line Search Filter Methods for Nonlinear Programming: Local Convergence". *SIAM Journal on Optimization* 16.1 (2005), pp. 32–48. DOI: 10.1137/S1052623403426544.

[56] Richard H Byrd, Jorge Nocedal, and Richard A Waltz. "KNITRO: An integrated package for nonlinear optimization". In: *Large-Scale Nonlinear Optimization*. Ed. by G. Di Pillo and M. Roma. Springer, 2006, pp. 35–59.

[57] "Gurobi 10: New Advances" (2021). https://www.gurobi.com/products/gurobi-optimizer/whats-new-current-release/.

[58] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[59] Yurii Nesterov. *Lectures on Convex Optimization*. second. Vol. 137. 2018.

[60] Ralph T Rockafellar and Roger J-B Wets. *Variational Analysis*. Springer, 2009.

[61] Ernest K. Ryu and Wotao Yin. *Large-scale Convex Optimization: Algorithms & Analyses via Monotone Operators*. Cambridge University Press, 2022.

[62] H. H. Bauschke, W. M. Moursi, and X. Wang. "Generalized monotone operators and their averaged resolvents". *Mathematical Programming* 189.1–2 (2021), pp. 55–74.

[63] Frank H Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.

[64] Boris S Mordukhovich. *Variational Analysis and Generalized Differentiation I: Basic Theory*. Springer, 2006.

[65] Jonathan Borwein and Adrian S Lewis. *Convex Analysis and Nonlinear Optimization, 2nd Edition*. Springer, 2006.

[66] Dimitris Bertsimas and Robert Weismantel. *Optimization over Integers*. Vol. 13. Dynamic Ideas Belmont, MA, 2005.

[67] Dimitris Bertsimas and Jack Dunn. *Machine Learning Under a Modern Optimization Lens*. Dynamic Ideas, MA, 2019.

[68] Tobias Achterberg and Eli Towle. "Non-Convex Quadratic Optimization: Gurobi 9.0" (2020). https://www.gurobi.com/resource/non-convex-quadratic-optimization/.

[69] Tobias Achterberg. "Non-Convex MIQCP in Gurobi 9.1: New Advances" (2020). https://cdn.gurobi.com/wp-content/uploads/2020/12/Non-Convex-MIQCP-in-Gurobi-9.1-New-Advances.pdf.

[70] Andreas Wächter and Lorenz T Biegler. "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming". *Mathematical Programming* 106.1 (2006), pp. 25–57.

[71] Anthony V Fiacco and Garth P McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. SIAM, 1990.

[72] Leo Liberti. "Introduction to global optimization". *Ecole Polytechnique* (2008).

[73] Adrien B. Taylor. "Convex Interpolation and Performance Estimation of First-Order Methods for Convex Optimization". PhD thesis. Catholic University of Louvain, Louvain-la-Neuve, Belgium, 2017.

[74] Chanwoo Park and Ernest K. Ryu. "Optimal First-Order Algorithms as a Function of Inequalities". *arXiv preprint arXiv:2110.11035* (2021).

[75] Kaiwen Zhou, Lai Tian, Anthony Man-Cho So, and James Cheng. "Practical Schemes for Finding Near-Stationary Points of Convex Finite-Sums". *International Conference on Artificial Intelligence and Statistics* (2022).

[76] Jisun Park and Ernest K. Ryu. "Exact Optimal Accelerated Complexity for Fixed-Point Iterations". *International Conference on Machine Learning* (2022).

[77] Roger A Horn and Charles R Johnson. *Matrix Analysis*. Cambridge University Press, 2012.

[78] Richard H Byrd, Guanghui Liu, and Jorge Nocedal. "On the local behavior of an interior point method for nonlinear programming". *Numerical Analysis* 1997 (1997), pp. 37–56.

[79] Benoît Legat, Oscar Dowson, Joaquim Dias Garcia, and Miles Lubin. "MathOptInterface: A Data Structure for Mathematical Optimization Problems". *INFORMS Journal on Computing* (2021).

[80] Albert Reuther et al. "Interactive supercomputing on 40,000 cores for machine learning and data analysis". In: *2018 IEEE High Performance extreme Computing Conference (HPEC)*. IEEE. 2018, pp. 1–6.

[81] Marco Locatelli and Fabio Schoen. *Global Optimization: Theory, Algorithms, and Applications*. SIAM, 2013.

[82] Reiner Horst and Hoang Tuy. *Global Optimization: Deterministic Approaches*. Springer Science & Business Media, 2013.

[83] Garth P McCormick. "Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems". *Mathematical Programming* 10.1 (1976), pp. 147–175.

[84] Hanif D Sherali and Warren P Adams. "A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems". *SIAM Journal on Discrete Mathematics* 3.3 (1990), pp. 411–430.

[85] Manfred Padberg. "The boolean quadric polytope: some characteristics, facets and relatives". *Mathematical Programming* 45.1 (1989), pp. 139–172.

[86] Hanif D Sherali and Barbara MP Fraticelli. "Enhancing RLT relaxations via a new class of semidefinite cuts". *Journal of Global Optimization* 22.1 (2002), pp. 233–261.

[87] Zhi-Quan Luo, Wing-Kin Ma, Anthony Man-Cho So, Yinyu Ye, and Shuzhong Zhang. "Semidefinite relaxation of quadratic optimization problems". *IEEE Signal Processing Magazine* 27.3 (2010), pp. 20–34.

[88] Hande Y Benson and Robert J Vanderbei. "Solving problems with semidefinite and related constraints using interior-point methods for nonlinear programming". *Mathematical Programming* 95.2 (2003), pp. 279–302.

[89] Dimitris Bertsimas and John N Tsitsiklis. *Introduction to Linear Optimization*. Vol. 6. Athena Scientific, 1997.

[90] Adrien B. Taylor. "Computer-aided analyses in optimization" (2020). https://francisbach.com/computer-aided-analyses/.

[91] Emmanuel J. Candès, Michael B. Wakin, and Stephen P. Boyd. "Enhancing Sparsity by Reweighted ℓ_1 Minimization". *Journal of Fourier Analysis and Applications* 14.5 (2008), pp. 877–905.

[92] Nicholas J Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[93] M. Fazel, H. Hindi, and S.P. Boyd. "Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices". *American Control Conference* (2003).

[94] J. Lee, C. Park, and E. K. Ryu. "A Geometric Structure of Acceleration and Its Role in Making Gradients Small Fast". *Neural Information Processing Systems* (2021).

[95] Baptiste Goujaud, Damien Scieur, Aymeric Dieuleveut, Adrien B. Taylor, and Fabian Pedregosa. "Super-Acceleration with Cyclical Step-sizes". *International Conference on Artificial Intelligence and Statistics* (2022).

[96] David Young. "On Richardson's Method for Solving Linear Systems with Positive Definite Matrices". *Journal of Mathematics and Physics* 32.1–4 (1953), pp. 243–255.

[97] Fabian Pedregosa. "On the link between optimization and polynomials: acceleration without Momentum" (2021). http://fa.bianp.net/blog/2021/no-momentum/.

[98] Yoel Drori. "Contributions to the Complexity Analysis of Optimization Algorithms." PhD thesis. Tel-Aviv University, Tel Aviv, Israel, 2014.

[99] Yoel Drori and Ohad Shamir. "The complexity of finding stationary points with stochastic gradient descent". *International Conference on Machine Learning* (2020).

[100] C. Park, J. Park, and E. K. Ryu. "Factor-$\sqrt{2}$ Acceleration of Accelerated Gradient Methods". *Applied Mathematics & Optimization* (2023).

[101] Amir Beck and Marc Teboulle. "A fast iterative shrinkage-thresholding algorithm for linear inverse problems". *SIAM Journal on Imaging Sciences* 2.1 (2009), pp. 183–202.

[102] Quoc Tran-Dinh. "The Connection Between Nesterov's Accelerated Methods and Halpern Fixed-Point Iterations". *arXiv preprint arXiv:2203.04869* (2022).

[103] A. M. Lyapunov. "The General Problem of the Stability of Motion". *Communications of the Mathematical Society of Kharkov* (1892).

[104] Damek Davis and Dmitriy Drusvyatskiy. "Stochastic subgradient method converges at the rate $O(k^{-1/4})$ on weakly convex functions". *arXiv preprint arXiv:1802.02988* (2018).

[105] Damek Davis and Dmitriy Drusvyatskiy. "Stochastic model-based minimization of weakly convex functions". *arXiv preprint arXiv:1803.06523* (2018).

[106] Ralph T Rockafellar. "Characterizing firm nonexpansiveness of prox mappings both locally and globally". *Journal of Nonlinear and Convex Analysis* 22.5 (2021).

[107] László Kozma. "Useful Inequalities" (2021). https://www.lkozma.net/inequalities_cheat_sheet/ineq.pdf.

[108] Ziqiang Shi and Rujie Liu. "Better Worst-Case Complexity Analysis of the Block Coordinate Descent Method for Large Scale Machine Learning". *International Conference on Machine Learning and Applications* (2017).

[109] Mathieu Barré, Adrien Taylor, and Alexandre d'Aspremont. "Complexity Guarantees for Polyak Steps with Momentum". *Conference on Learning Theory* (2020).

[110] William W Hager and Hongchao Zhang. "A survey of nonlinear conjugate gradient methods". *Pacific Journal of Optimization* 2.1 (2006), pp. 35–58.

[111] Neculai Andrei. *Nonlinear Conjugate Gradient Methods for Unconstrained Optimization*. Springer, 2020.

[112] Larry Armijo. "Minimization of functions having Lipschitz continuous first partial derivatives". *Pacific Journal of Mathematics* 16.1 (1966), pp. 1–3.

[113] Allen A Goldstein. "On steepest descent". *Journal of the Society for Industrial and Applied Mathematics, Series A: Control* 3.1 (1965), pp. 147–151.

[114] Philip Wolfe. "Convergence conditions for ascent methods". *SIAM Review* 11.2 (1969), pp. 226–235.

[115] Philip Wolfe. "Convergence conditions for ascent methods II: Some corrections". *SIAM Review* 13.2 (1971), pp. 185–188.

[116] William W. Hager and Hongchao Zhang. "A new conjugate gradient method with guaranteed descent and an efficient line search". *SIAM Journal on Optimization* 16.1 (2005), pp. 170–192.

[117] Luigi Grippo, Francesco Lampariello, and Stephano Lucidi. "A nonmonotone line search technique for Newton's method". *SIAM Journal on Numerical Analysis* 23.4 (1986), pp. 707–716.

[118] Hongchao Zhang and William W Hager. "A nonmonotone line search technique and its application to unconstrained optimization". *SIAM journal on Optimization* 14.4 (2004), pp. 1043–1056.

[119] Shuai Huang, Zhong Wan, and Xiaohong Chen. "A new nonmonotone line search technique for unconstrained optimization". *Numerical Algorithms* 68.4 (2015), pp. 671–689.

[120] R. Paul Gorman and Terrence J. Sejnowski. "Analysis of hidden units in a layered network trained to classify sonar targets". *Neural Networks* 1.1 (1988), pp. 75–89.

[121] Yurii Nesterov. "A method for solving the convex programming problem with convergence rate $O(1/k^2)$". *Doklady Akademii Nauk* (1983).

[122] Alexandre d'Aspremont, Damien Scieur, and Adrien B. Taylor. "Acceleration methods". *Foundations and Trends® in Optimization* 5.1-2 (2021), pp. 1–245.

[123] Mathieu Barré, Adrien B. Taylor, and Alexandre d'Aspremont. "Complexity guarantees for Polyak steps with momentum". In: *Conference on Learning Theory*. 2020.

[124] Mathieu Barré. "Worst-Case Analysis of Efficient First-Order Methods". PhD thesis. Université Paris Sciences & Lettres, 2021.

[125] Gurobi Optimization, LLC. *Gurobi Optimizer Reference Manual*. 2023. URL: https://www.gurobi.com.

[126] Baptiste Goujaud, Céline Moucer, François Glineur, Julien M. Hendrickx, Adrien B. Taylor, and Aymeric Dieuleveut. "PEPit: computer-assisted worst-case analyses of first-order optimization methods in Python". *arXiv:2201.04040* (2022).

[127] Arkadi Nemirovski. "Optimization II: Numerical Methods for Nonlinear Continuous Optimization". *Lecture notes, http://www2.isye.gatech.edu/~nemirovs/Lect_OptII.pdf* (1999).

[128] Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer, 2006.

[129] Joseph-Frédéric Bonnans, Jean-Charles Gilbert, Claude Lemaréchal, and Claudia A. Sagastizábal. *Numerical Optimization: Theoretical and Practical Aspects*. Springer, 2006.

[130] Mehiddin Al-Baali. "Descent property and global convergence of the Fletcher—Reeves method with inexact line search". *IMA Journal of Numerical Analysis* 5.1 (1985), pp. 121–124.

[131] Yu-Hong Dai. "Analysis of conjugate gradient methods". *Institute of Computational Mathematics and Scientific/Engineering Computing. Chinese Academy of Science (in Chinese)* (1997).

[132] Rémi Chan–Renous-Legoubin and Clément W. Royer. "A nonlinear conjugate gradient method with complexity guarantees and its application to nonconvex regression". *EURO Journal on Computational Optimization* 10 (2022), p. 100044. ISSN: 2192-4406.

[133] Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. ""Convex Until Proven Guilty": Dimension-Free Acceleration of Gradient Descent on Non-Convex Functions". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 654–663.

[134] Boris T. Polyak. *Introduction to Optimization.* Optimization Software, Inc., New York, 1987.

[135] Yurii Nesterov. *Introductory Lectures on Convex Optimization: A Basic Course*. Applied optimization. Springer, 2004.

[136] Aaron Meurer et al. "SymPy: symbolic computing in Python". *PeerJ Computer Science* 3 (Jan. 2017), e103. ISSN: 2376-5992. DOI: 10.7717/peerj-cs.103. URL: https://doi.org/10.7717/peerj-cs.103.

[137] Wolfram Research Inc. *Wolfram Language, Version 14.0*. Champaign, IL, 2024. URL: https://www.wolfram.com/mathematica.

[138] Adrien B. Taylor, Julien M. Hendrickx, and François Glineur. "Performance estimation toolbox (PESTO): Automated worst-case analysis of first-order optimization methods". *Conference on Decision and Control* (2017).

[139] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[140] Dimitris Bertsimas, Vassilis Digalakis Jr, Michael Lingzhi Li, and Omar Skali Lami. "Slowly varying regression under sparsity". *Operations Research* (2024).

[141] Steven Diamond, Reza Takapoui, and Stephen Boyd. "A general system for heuristic minimization of convex functions over non-convex sets". *Optimization Methods and Software* 33.1 (2018), pp. 165–193.

[142] Dimitris Bertsimas and Bart P.G. Van Parys. "Sparse high-dimensional regression: Exact scalable algorithms and phase transitions". *The Annals of Statistics* 48.1 (2020), pp. 300–323.

[143] Dimitris Bertsimas and Ryan Cory-Wright. "A scalable algorithm for sparse portfolio selection". *Informs Journal on Computing* 34.3 (2022), pp. 1489–1511.

[144] Dimitris Bertsimas, Ryan Cory-Wright, and Jean Pauphilet. "Mixed-projection conic optimization: A new paradigm for modeling rank constraints". *Operations Research* 70.6 (2022), pp. 3321–3344.

[145] Dimitris Bertsimas, Ryan Cory-Wright, Sean Lo, and Jean Pauphilet. "Optimal Low-Rank Matrix Completion: Semidefinite Relaxations and Eigenvector Disjunctions". *arXiv preprint arXiv:2305.12292* (2023).

[146] Moritz Hardt, Raghu Meka, Prasad Raghavendra, and Benjamin Weitz. "Computational limits for matrix completion". *Journal of Machine Learning Research* (2014).

[147] M. Fazel, E. Candes, B. Recht, and P. Parrilo. "Compressed sensing and robust recovery of low rank matrices". In: *2008 42nd Asilomar Conference on Signals, Systems and Computers*. 2008, pp. 1043–1047.

[148] Thomas Blumensath and Mike E Davies. "Iterative thresholding for sparse approximations". *Journal of Fourier Analysis and Applications* 14.5-6 (2008), pp. 629–654.

[149] Thomas Blumensath and Mike E. Davies. "Normalized Iterative Hard Thresholding: Guaranteed Stability and Performance". *IEEE Journal of Selected Topics in Signal Processing* 4.2 (2010), pp. 298–309.

[150] Simon Foucart. "Hard thresholding pursuit: an algorithm for compressive sensing". *SIAM Journal on Numerical Analysis* 49.6 (2011), pp. 2543–2563.

[151] Jeffrey D Blanchard, Jared Tanner, and Ke Wei. "CGIHT: Conjugate gradient iterative hard thresholding for compressed sensing and matrix completion". *Information and Inference: A Journal of the IMA* 4.4 (2015), pp. 289–327.

[152] Stephen Boyd, Neal Parikh, Eric Chu, Borja Peleato, and Jonathan Eckstein. "Distributed optimization and statistical learning via the alternating direction method of multipliers". *Foundations and Trends® in Machine learning* 3.1 (2011), pp. 1–122.

[153] Reza Takapoui, Nicholas Moehle, Stephen Boyd, and Alberto Bemporad. "A simple effective heuristic for embedded mixed-integer quadratic programming". *International Journal of Control* (2017), pp. 1–11.

[154] Reza Takapoui. "The Alternating Direction Method of Multipliers for Mixed-integer Optimization Applications". PhD thesis. Stanford University, 2017.

[155] Madeleine Udell, Corinne Horn, Reza Zadeh, Stephen Boyd, et al. "Generalized low rank models". *Foundations and Trends® in Machine Learning* 9.1 (2016), pp. 1–118.

[156] Andreas M Tillmann, Daniel Bienstock, Andrea Lodi, and Alexandra Schwartz. "Cardinality Minimization, Constraints, and Regularization: A Survey". *arXiv preprint arXiv:2106.09606* (2021).

[157] Hussein Hazimeh and Rahul Mazumder. "Fast best subset selection: Coordinate descent and local combinatorial optimization algorithms". *Operations Research* 68.5 (2020), pp. 1517–1537.

[158] Heinz H Bauschke and Patrick L Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Vol. 408. Springer, 2011.

[159] Amir Beck. *First-Order Methods in Optimization*. Vol. 25. SIAM, 2017.

[160] Ernest K. Ryu and Stephen Boyd. "Primer on monotone operator methods". *Applied And Computational Mathematics* 15.1 (2016), pp. 3–43.

[161] Neal Parikh and Stephen Boyd. "Proximal algorithms". *Foundations and Trends® in optimization* 1.3 (2014), pp. 127–239.

[162] Andreas Themelis and Panagiotis Patrinos. "Douglas-Rachford Splitting and ADMM for Nonconvex Optimization: Tight Convergence Results". *SIAM Journal on Optimization* 30.1 (2020), pp. 149–181.

[163] Ernest K. Ryu. "Uniqueness of DRS as the 2 operator resolvent-splitting and impossibility of 3 operator resolvent-splitting". *Mathematical Programming* 182.1 (2020), pp. 233–273.

[164] René Poliquin, Ralph T Rockafellar, and Lionel Thibault. "Local differentiability of distance functions". *Transactions of the American Mathematical Society* 352.11 (2000), pp. 5231–5249.

[165] D. Russell Luke. "Prox-Regularity of Rank Constraint Sets and Implications for Algorithms". *Journal of Mathematical Imaging and Vision* 47.3 (2013), pp. 231–238. ISSN: 1573-7683. DOI: 10.1007/s10851-012-0406-3.

[166] Heinz H Bauschke, Manish Krishan Lal, and Xianfu Wang. "Projections onto hyperbolas or bilinear constraint sets in Hilbert spaces". *Journal of Global Optimization* (2022), pp. 1–12.

[167] Jean-Philippe Vial. "Strong and weak convexity of sets and functions". *Mathematics of Operations Research* 8.2 (1983), pp. 231–259.

[168] Francis H Clarke, Ronald J Stern, and Peter R Wolenski. "Proximal smoothness and the lower-$\mathcal{C}^2$ property". *Journal of Convex Analysis* 2.1-2 (1995), pp. 117–144.

[169] Alexander Shapiro. "Existence and differentiability of metric projections in Hilbert spaces". *SIAM Journal on Optimization* 4.1 (1994), pp. 130–141.

[170] Alfred Auslender. "Stability in mathematical programming with nondifferentiable data". *SIAM Journal on Control and Optimization* 22.2 (1984), pp. 239–254.

[171] Pontus Giselsson and Stephen Boyd. "Linear convergence and metric selection for Douglas-Rachford splitting and ADMM". *IEEE Transactions on Automatic Control* 62.2 (2017), pp. 532–544. ISSN: 0018-9286.

[172] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY, USA: Springer, 2001.

[173] Trevor Hastie, Robert Tibshirani, and Ryan J Tibshirani. "Extended comparisons of best subset selection, forward stepwise selection, and the lasso". *arXiv preprint arXiv:1707.08692* (2017).

[174] Dimitris Bertsimas, Martin S Copenhaver, and Rahul Mazumder. "Certifiably optimal low rank factor analysis". *Journal of Machine Learning Research* 18.1 (2017), pp. 907–959.

[175] Lorenzo Stella et al. *JuliaFirstOrder/ProximalOperators.jl: v0.16.1.* https://doi.org/10.5281/zenodo.10048760. Version v0.16.1. 2023. DOI: 10.5281/zenodo.10048760.

[176] Jerome Friedman, Trevor Hastie, Rob Tibshirani, et al. "glmnet: Lasso and elastic-net regularized generalized linear models". *R package version* 1.4 (2009), pp. 1–24.

[177] Joonseok Lee, Seungyeon Kim, Guy Lebanon, Yoram Singer, and Samy Bengio. "LLORMA: Local low-rank matrix approximation". *Journal of Machine Learning Research* 17.1 (2016), pp. 442–465.

[178] Jos MF ten Berge. "Some recent developments in factor analysis and the search for proper communalities". In: *Advances in Data Science and Classification*. Springer, 1998, pp. 325–334.

[179] James Saunderson, Venkat Chandrasekaran, Pablo Parrilo, and Alan S Willsky. "Diagonal and low-rank matrix decompositions, correlation matrices, and ellipsoid fitting". *SIAM Journal on Matrix Analysis and Applications* 33.4 (2012), pp. 1395–1416.

[180] Frédéric Bernard, Lionel Thibault, and Nadia Zlateva. "Prox-regular sets and epigraphs in uniformly convex Banach spaces: various regularities and other properties". *Transactions of the American Mathematical Society* 363.4 (2011), pp. 2211–2247.

[181] Walter Rudin. *Principles of Mathematical Analysis*. McGraw-Hill New York, 1986.

[182] Yurii Nesterov. "Smooth minimization of non-smooth functions". *Mathematical Programming* 103.1 (2005), pp. 127–152.

[183] Rafael Correa, Alejandro Jofre, and Lionel Thibault. "Characterization of lower semicontinuous convex functions". *Proceedings of the American Mathematical Society* (1992), pp. 67–72.

[184] René Poliquin and Ralph T Rockafellar. "Prox-regular functions in variational analysis". *Transactions of the American Mathematical Society* 348.5 (1996), pp. 1805–1838.

[185] Boris T Polyak. *Introduction to Optimization*. Optimization Software, 1987.

[186] Asen L Dontchev and Ralph T Rockafellar. *Implicit Functions and Solution Mappings*. Vol. 543. Springer, 2009.

[187] Guoyin Li and Ting Kei Pong. "Douglas-Rachford splitting for nonconvex optimization with application to nonconvex feasibility problems". *Mathematical Programming* 159.1 (2016), pp. 371–401.