Branch-and-bound performance estimation programming: a unified methodology for constructing optimal optimization methods

Citation: Das Gupta, S., Van Parys, B.P.G. & Ryu, E.K. Branch-and-bound performance estimation programming: a unified methodology for constructing optimal optimization methods. Math. Program. 204, 567–639 (2024).

Published Version: 10.1007/s10107-023-01973-1

Publisher: Springer Science and Business Media LLC

Permanent Link: https://hdl.handle.net/1721.1/153536

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Branch-and-bound performance estimation programming: a unified methodology for constructing optimal optimization methods

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

Shuvomoy Das Gupta
MIT Operations Research Center
sdgupta@mit.edu

Bart P.G. Van Parys
MIT Sloan School of Management
vanparys@mit.edu

Ernest K. Ryu
Seoul National University
ernestryu@snu.ac.kr

Abstract

We present the 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 optimization method as a nonconvex but practically tractable quadratically constrained quadratic optimization problem and solves it to certifiable global optimality using a customized branch-and-bound algorithm. By directly confronting the nonconvexity, BnB-PEP offers significantly more flexibility and removes the many limitations of the prior methodologies. 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 setups for which the prior methodologies do not apply and obtain methods with bounds that improve 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.

Contents

1 Introduction

Since the pioneering work of Nesterov and Nemirovsky on accelerated gradient methods [60] and information-based complexity [59, 58], 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 [32]. The celebrated accelerated gradient method by Nesterov was improved by a constant factor in [43, 84, 78], and entirely novel acceleration mechanisms, distinct from Nesterov's, have been discovered [45, 42, 51, 87]. 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.

1.1 Prior work

The performance estimation methodology, initiated by Drori and Teboulle [32], 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 [80]. Lessard, Recht, and Packard combined the notion of the performance estimation methodology with control-theoretic notions through their integral quadratic constraints (IQC) formulation [49]. 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 [82]. Taylor and Bach extended this observation through a methodology that uses the performance estimation approach to find the optimal sublinear rates through potential functions [77].

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) [32] and Kim and Fessler found its analytical description [43]. 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 [28]. 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 [30]. 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 [84]. Taylor and Drori constructed the information-theoretic exact method (ITEM), further improving upon the triple momentum method [78]. Drori and Taylor showed that ITEM is exactly optimal through an exact matching complexity lower bound [31]. Kim and Fessler constructed OGM-G, which has the best known rate for reducing the gradient magnitude in the smooth convex setup [45]. Finally, the performance estimation methodology has also been utilized for constructing methods with inexact evaluations [19, 5, 25], analyzing methods in the composite minimization setup [44, 81], analyzing methods with exact line search [24], analyzing monotone operator and splitting methods [70, 37, 51, 42], and analyzing acceleration in the mirror descent setup [26].

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 §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 §6.2 and §6.3 of this work are such examples.

1.2 Organization

This paper is organized as follows. In §2, we present the necessary background and describe our problem setup. In §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 §3.1.1 follows prior approaches, and our novel contribution starts in §3.1.2. In §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 §5, we present the generalized formulation of our methodology. In §6, we demonstrate the effectiveness of our methodology through several applications. In §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 §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 [2]. In §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 [23, Theorem 3.1]. Additionally, in §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 [77] to nonconvex and nonsmooth setups.

1.3 Computational setup

For scientific reproducibility, we open-source our codes to generate all the numerical results presented in this paper 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` [33]—to model the optimization problems. Our proposed algorithm uses the following solvers: `Mosek 9.3` [57] (free for academic use), `Ipopt 3.12.11` [86] (open-source), `KNITRO 13.0.0` [16] (free for academic use), and `Gurobi 10` [1] (free for academic use).

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 [70]. 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 [14, 61, 69, 71]. 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

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

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 [7]:

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

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

(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 $0 \in \partial f(x)$.

5

(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 [18, §1.2], Mordukhovich subdifferential [56, §1.3], and Fréchet subdifferential [13, page 132] are all instances of the abstract subdifferential [7, page 70]. Whenever we say *subdifferential* in this paper, 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 $x \in \mathbb{R}^d$.

2.1 Quadratically constrained quadratic program (QCQP)

A QCQP is defined as:

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

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 [12, pp. 565–567]. However, such theoretical worst-case intractability does not necessarily imply that specific problem instances are not *practically tractable* [10, 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 [10, 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 [4]. For example, in less than three years, `Gurobi`'s spatial branch-and-bound algorithm has achieved a machine-independent speedup factor of 175.5 [3, 1].

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 §4 and §5, and demonstrate that the speedup is absolutely essential for the BnB-PEP Algorithm to be used practically.

6

Function class	Notation
L-smooth Convex $(0 < L < \infty)$	$\mathcal{F}_{0,L}$
L-smooth and μ-strongly convex $(0 \leq \mu < L < \infty)$	$\mathcal{F}_{\mu,L}$
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 1: Function classes considered in this paper.

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` [16] or `Ipopt` [85] to compute locally optimal solutions under certain regularity conditions [35, Theorem 4][50, §3.2]. On the other hand, one can use the spatial branch-and-bound algorithm implemented in solvers such as `Gurobi` [1] to find globally optimal solutions of nonconvex QCQPs and to certify their optimality in finite time.

2.2 Problem setup

Consider the unconstrained minimization problem

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad f(x), \tag{2}$$

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 1. More generally, we can use the BnB-PEP methodology with quadratically representable function classes, a notion we further discuss in §8.1 of the appendix.

Fixed-step first-order method $\mathcal{M}_N$. We consider fixed-step first-order methods, which include most subgradient methods and accelerated gradient methods [78]. 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:

$$x_i = x_{i-1} - \sum_{j=0}^{i-1} s_{i,j} f'(x_j) \tag{3}$$

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 (3) with

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

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

$$\overline{s}_{i,j} = \begin{cases} s_{i,i-1}, & \text{if } j = i-1, \\ \overline{s}_{i-1,j} + s_{i,j}, & \text{if } j \in [0:i-2] \end{cases} \tag{4}$$

for $0 \leq j < i \leq N$. Write $s = \{s_{i,j}\}_{0\leq j<i\leq N}$ and $\overline{s} = \{\overline{s}_{i,j}\}_{0\leq j<i\leq N}$ 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 [75, 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 paper, 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), 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

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

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

$$\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) = \left(\begin{array}{ll} \text{maximize} & \mathcal{E}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) \\ \text{subject to} & \\ & f \in \mathcal{F}, \\ & x_\star \text{ is a globally optimal solution to (2)}, \\ & \{x_i\}_{i\in[1:N]} \text{ is generated by FSFOM } M \text{ with initial point } x_0, \\ & \mathcal{C}\left(\{x_i, f'(x_i), f(x_i)\}_{i\in I_N^\star}\right) \leq 0, \end{array} \right) \qquad (\mathcal{O}^{\text{inner}})$$

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 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:

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

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 §3, we illustrate our methodology by describing it for a concrete problem instance. In §5, we present the general form of the the methodology.

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 §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 $(\mathcal{O}^{\text{outer}})$:

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

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 §3.1.1, we formulate the inner problem $(\mathcal{O}^{\text{inner}})$ as a convex SDP. This first step follows the approach of [32, 79]. Then in §3.1.2 formulates the outer problem $(\mathcal{O}^{\text{outer}})$ as a QCQP. This second step is is novel.

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 (3), parameterize FS-FOMs in $\mathcal{M}_N$ as

$$x_i = x_{i-1} - \frac{1}{L}\sum_{j=0}^{i-1} h_{i,j} f'(x_j) \tag{5}$$

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

$$\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{ll} \text{maximize} & \|\nabla f(x_N)\|^2 \\ \text{subject to} & f \in \mathcal{F}_{\mu,L}, \\ & \nabla f(x_\star) = 0, \\ & x_i = x_{i-1} - \frac{1}{L}\sum_{j=0}^{i-1} h_{i,j}\nabla f(x_j), \quad i \in [1:N], \\ & \|x_0 - x_\star\|^2 \le R^2, \\ & x_\star = 0,\ f(x_\star) = 0, \end{array} \right)
\end{aligned} \tag{6}$$

where $f, x_0, \dots, x_N$ 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 1 (Reparametrization from $\mathcal{F}_{\mu,L}$ to $\mathcal{F}_{0,L-\mu}$ [78, §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* (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 (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 $i \in [1:N]$.

10

Reformulated infinite-dimensional maximization problem. Using Lemma 1, we reformulate ($\mathcal{O}^{\text{inner}}$) as

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

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 ($\mathcal{F}_{0,L}$-interpolation [78, 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]

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

Finite-dimensional maximization problem. Using Lemma 2, reformulate ($\mathcal{O}^{\text{inner}}$) as

$$\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{ll}
\text{maximize} & \|g_N\|^2 + \mu^2 \|x_N - x_\star\|^2 - 2\mu \langle g_N \mid x_\star - x_N \rangle \\
\text{subject to} & \\
& f_i \geq f_j + \langle g_j \mid x_i - x_j \rangle + \frac{1}{2(L-\mu)} \|g_i - g_j\|^2, \quad i, j \in I_N^\star : i \neq j, \\
& g_\star = 0, x_\star = 0, f_\star = 0, \\
& x_i = x_0 \left(1 - \frac{\mu}{L} \sum_{j=0}^{i-1} \alpha_{i,j}\right) - \frac{1}{L} \sum_{j=0}^{i-1} \alpha_{i,j} g_j, \quad i \in [1:N], \\
& \|x_0 - x_\star\|^2 \leq R^2.
\end{array} \right)
\end{aligned} \tag{7}$$

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 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.

[1] This can be viewed as a discretization of the following condition [61, Theorem 2.1.5, Equation (2.1.10)]: $f \in \mathcal{F}_{0,L}$ if and only if

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

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

$$\begin{aligned} H &= [x_0 \mid g_0 \mid g_1 \mid \ldots \mid g_N] \in \mathbb{R}^{d\times(N+2)}, \\ G &= H^\top H \in \mathbb{S}_+^{N+2}, \\ F &= [f_0 \mid f_1 \mid \ldots \mid f_N] \in \mathbb{R}^{1\times(N+1)}. \end{aligned} \tag{8}$$

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

$$\begin{aligned} &\mathbf{g}_\star = 0 \in \mathbb{R}^{N+2},\ \mathbf{g}_i = e_{i+2} \in \mathbb{R}^{N+2}, \quad i \in [0:N] \\ &\mathbf{x}_0 = e_1 \in \mathbb{R}^{N+2},\ \mathbf{x}_\star = 0 \in \mathbb{R}^{N+2}, \\ &\mathbf{x}_i = \mathbf{x}_0\left(1 - \frac{\mu}{L}\sum_{j=0}^{i-1}\alpha_{i,j}\right) - \frac{1}{L}\sum_{j=0}^{i-1}\alpha_{i,j}\mathbf{g}_j \in \mathbb{R}^{N+2}, \quad i \in [1:N] \\ &\mathbf{f}_\star = 0 \in \mathbb{R}^{N+1},\ \mathbf{f}_i = e_{i+1} \in \mathbb{R}^{N+1}, \quad i \in [0:N]. \end{aligned} \tag{9}$$

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

$$\begin{aligned} A_{i,j}(\alpha) &= \mathbf{g}_j \odot (\mathbf{x}_i - \mathbf{x}_j) \in \mathbb{S}^{N+2}, \\ B_{i,j}(\alpha) &= (\mathbf{x}_i - \mathbf{x}_j) \odot (\mathbf{x}_i - \mathbf{x}_j) \in \mathbb{S}_+^{N+2}, \\ C_{i,j} &= (\mathbf{g}_i - \mathbf{g}_j) \odot (\mathbf{g}_i - \mathbf{g}_j) \in \mathbb{S}_+^{N+2}, \\ a_{i,j} &= \mathbf{f}_j - \mathbf{f}_i \in \mathbb{R}^{N+1}. \end{aligned} \tag{10}$$

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

$$\begin{aligned} \langle g_j \mid x_i - x_j\rangle &= \mathbf{tr}\, GA_{i,j}(\alpha), \\ \|x_i - x_j\|^2 &= \mathbf{tr}\, GB_{i,j}(\alpha), \\ \|g_i - g_j\|^2 &= \mathbf{tr}\, GC_{i,j}, \end{aligned} \tag{11}$$

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

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

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

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 [80, §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 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 $(\mathcal{O}^{\mathrm{inner}})$ as a convex SDP

$$\begin{aligned}
&\mathcal{R}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{ll}
\text{maximize} & \mathbf{tr}\, G \left(C_{N,\star} + \mu^2 B_{N,\star}(\alpha) - 2\mu A_{\star,N}(\alpha) \right) \\
\text{subject to} & \\
& F a_{i,j} + \mathbf{tr}\, G A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} \mathbf{tr}\, G C_{i,j} \leq 0, \quad i,j \in I_N^\star : i \neq j, \quad \triangleright \text{ dual var. } \lambda_{i,j} \geq 0 \\
& -G \preceq 0, \quad \triangleright \text{ dual var. } Z \succeq 0 \\
& \mathbf{tr}\, G B_{0,\star} \leq R^2, \quad \triangleright \text{ dual var. } \nu \geq 0
\end{array} \right)
\end{aligned} \tag{12}$$

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 (12) and its solution is independent of the dimension d, provided that the large-scale assumption $d \geq N+2$ holds. See [80, §3.3] for further discussion.

Dualization. Next we use convex duality to formulate $(\mathcal{O}^{\mathrm{inner}})$, originally a maximization problem, as a minimization problem. Take the dual of (12) to get

$$\begin{aligned}
&\overline{\mathcal{R}}(M, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\
&= \left(\begin{array}{ll}
\text{minimize} & \nu R^2 \\
\text{subject to} & \\
& \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} = 0, \\
& \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha) + 2\mu A_{\star,N}(\alpha) + \\
& \quad \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\
& Z \succeq 0, \\
& \nu \geq 0,\ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j,
\end{array} \right)
\end{aligned} \tag{13}$$

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. *Strong duality holds between* (12) *and* (13)*, i.e.,*

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

13

This assumption can be removed in most cases using the line of reasoning of [64, 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 §8.2 of the appendix. In any case, strong duality holds for "generic" FSFOMs [80, 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 [32, 79], and, at this point, our formulations diverge.

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 [32, 43], ITEM [78], ORC-$\text{F}_\flat$ [64, §4], and OBL-$\text{F}_\flat$ [64, §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 §4.3. Due to the relaxation, the solution FSFOM of the relaxed SDP is not necessarily the exact optimal FSFOM. Exact optimality of OGM [28] and ITEM [31] were proved through separate exact matching complexity lower bounds. Even though ORC-$\text{F}_\flat$ and OBL-$\text{F}_\flat$ 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 [45], M-OGM-G [89], OBL-$\text{G}_\flat$ [64], APPM [42, 51], and SM-APPM [65] 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 [65]. OGM-G is presumed but not proven to be exactly optimal. M-OGM-G and OBL-$\text{G}_\flat$ 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 §6.2 and §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 (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 3 ([40, 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 3, applied to the present setup, have the following

14

equivalent representations:

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

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

$$\begin{aligned} &\mathcal{R}^\star(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}) \\ &= \left(\begin{array}{l} \text{minimize} \quad \nu R^2 \\ \text{subject to} \\ \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} = 0, \\ \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha) + 2\mu A_{\star,N}(\alpha) + \\ \qquad \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(A_{i,j}(\alpha) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\ P \text{ is lower triangular with nonnegative diagonals,} \\ PP^\top = Z, \\ \nu \geq 0, \ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \end{array} \right) \end{aligned} \tag{14}$$

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

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

We now solve the BnB-PEP-QCQP (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.

Algorithm 1 BnB-PEP Algorithm: Given (μ, L, R), solves (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 (13) and solve the convex SDP. Denote the computed optimal solution to (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 (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 (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 §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 $p^\star$.

15

Details of the BnB-PEP Algorithm. Stage 1 computes a feasible solution of (14) by taking advantage of the structure that (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 1, this corresponds to $\alpha^{\text{init}}_{i,i-1} = 1$ for $i \in [1:N]$ and $\alpha^{\text{init}}_{i,j} = 0$ for $j \neq i-1$.

Stage 2 computes a locally optimal solution to (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 [15, 85], [35, §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 §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` [4] to solve (14). Spatial branch-and-bound algorithms do compute globally optimal solutions in "finite time", but the default implementation is impractically slow, as Table 4 illustrates. Customizing the spatial branch-and-bound algorithm with problem-specific insights is essential, as discussed in §4.

Numerical results. We conduct numerical experiments on the computational environment described in §1.3 with parameters $\mu = 0.1$, $L = 1$, and $R = 1$. Due to the scale invariance discussed in [80, §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 [80, §3.5] and [75, §4.2.5].[2] Tables 2 and 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 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 §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 §6 as well. Because Stages 1 and 2 tend to be faster than Stage

[2]The journal version of [80, §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: Comparison of the optimal method obtained by solving (14) with the BnB-PEP Algorithm against other known methods.

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 optimal.

Table 2 compares the performance of the optimal method, obtained with the BnB-PEP Algorithm, against GD (plain gradient desent), ITEM [78], and OGM-$\mathcal{F}_{\mu,L}$ [78]. (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 [78], and the stepsizes of OGM-$\mathcal{F}_{\mu,L}$ were taken from [78, §E.1].) We also show the total number of variables and constraints of (14) that the BnB-PEP Algorithm works with after the mathematical model described in `JuMP` gets converted to the `MathOptInterface` format [48], which is the standard data structure for representing optimization models in `JuMP`.

Table 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 1, and present $h^\star$ in the table.

Table 4 presents the runtimes with and without the customized spatial branch-and-bound solver of §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 [67]. 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.

4 Efficient implementation of the BnB-PEP Algorithm

As Table 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

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 3: Globally optimal stepsizes obtained by solving (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 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`.

18

specific problem structure and obtain a speedup that enables us to run the BnB-PEP Algorithm on a laptop.

In §4.1, we briefly review the standard spatial branch-and-bound algorithm. We present the customization that provide significant speedups in §4.2. In §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.

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

We briefly review the standard spatial branch-and-bound algorithm [4, 52, 41, 50]. 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 (1). The algorithm starts with the *presolve* phase, solving a linear relaxation of (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 (1) into a finite collection of subregions $F_1, \ldots, F_K$ and considering the subproblems

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

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 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 (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 (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 (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

[3] The setups of §3 and §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 $p^\star = \infty$.

19

constructed using the McCormick envelopes [55], which provide lower and upper bounds for the quadratic objective and constraints in (1), whereas the the linear constraints in (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 [72, 62, 73]. Solving these linear optimization problems along with the cuts provides valid lower and upper bounds on the optimal values of (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 (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 (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 (1) along with one optimal solution subject to the tolerance, and the algorithm terminates.

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 [1] uses solver-independent *callback functions*, an interface provided by `JuMP` [33].

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 §5. We first present the customizations in §4.2.1, §4.2.2, and §4.2.3, and then discuss the observed speedups in §4.2.4.

20

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 (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 (1) to obtain tighter bounds.

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

$$\begin{aligned} &Z = Z^\top, \\ &\operatorname{diag}(Z) \geq 0, \\ &-\tfrac{Z_{i,i}+Z_{j,j}}{2} \leq Z_{i,j} \leq \tfrac{Z_{i,i}+Z_{j,j}}{2}. \end{aligned} \tag{15}$$

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 [40, 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

$$|Z_{i,j}| \leq \sqrt{Z_{i,i} Z_{j,j}} \Leftrightarrow Z_{i,j}^2 \leq Z_{i,i} Z_{j,j} \tag{16}$$

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 (16), we opt for the third linear constraint (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 §4.1. The linear relaxations are derived from McCormick convex envelopes, which are constructed without considering underlying convexity and are computationally expensive [52, 41, 50]. Using the SOC constraint (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 [4, 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 (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 (14)**.** Next, we compute bounds M_λ, M_ν, M_α, and M_Z such that

$$\begin{array}{l} \lambda_{i,j} \leq M_\lambda, \quad i,j \in I_N^\star : i \neq j, \\ |Z_{i,j}| \leq M_Z, \quad i,j \in [1:N+2], \\ |P_{i,j}| \leq M_P, \quad i,j \in [1:N+2], \\ |\alpha_{i,j}| \leq M_\alpha, \quad i \in [1:N],\ j \in [0:i-1], \\ \nu \leq M_\nu, \end{array} \tag{17}$$

are satisfied by global minimizers of (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 (14), which includes the nonconvex rank-1 constraint $W = ww^\top$ [53]. The specific form is quite tedious, so we present it in §8.3 of the appendix. We then relax the rank-1 constraint $W = ww^\top$ to an implied convex constraint

$$W \succeq ww^\top \Leftrightarrow \begin{bmatrix} W & w \\ w^\top & 1 \end{bmatrix} \succeq 0, \tag{18}$$

where we have used the Schur complement. Since any feasible (and optimal) solution of (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

$$\left(\begin{array}{ll} \text{maximize} & c_\lambda M_\lambda + c_Z M_Z + c_\alpha M_\alpha \\ \text{subject to} & \text{semidefinite relaxation of (14)}, \\ & \text{constraint (18)}, \\ & \lambda_{i,j} \leq M_\lambda, \quad i,j \in I_N^\star : i \neq j, \\ & |Z_{i,j}| \leq M_Z, \quad i,j \in [1:N+2], \\ & |\alpha_{i,j}| \leq M_\alpha, \quad i \in [1:N],\ j \in [0:i-1], \\ & \nu \leq \nu^{\text{init}}, \end{array}\right) \tag{19}$$

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

$$P_{i,j}^2 \leq \sum_{k=1}^{i} P_{i,k}^2 = Z_{ii} \leq M_Z \tag{20}$$

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.)

[4] As a note of caution, solving (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.

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 paper. 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 §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 [38, §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 (13), consider the following convex SDP

$$\left(\begin{array}{ll} \text{maximize} & c_\lambda \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} + c_Z\, \mathbf{tr}\, Z \\ \text{subject to} & \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} a_{i,j} = 0, \\ & \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha^{\text{lopt}}) + 2\mu A_{\star,N}(\alpha^{\text{lopt}}) + \\ & \quad \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left(A_{i,j}(\alpha^{\text{lopt}}) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\ & Z \succeq 0, \\ & \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \\ & \nu \geq 0, \\ & \nu R^2 \leq \nu^{\text{lopt}} R^2, \end{array}\right) \tag{21}$$

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 (15) implies that every entry of Z is bounded by the maximum of the diagonal entries. Set M_P from M_Z using (20). Set

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

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 (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.

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 (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 (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

$$\mathbf{tr}\left(Zyy^\top\right) \geq 0, \qquad \forall\, y \in Y, \tag{22}$$

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 [9, §5.1]. In (14), we relax $Z = PP^\top$ into the constraint (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 (14), where (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 (22).)

In step (ii), if $\text{eig}_{\min}(Z) \geq 0$, then the solution Z of the relaxation (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 [11, pp. 205–207].

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 (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

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

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 (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.

4.2.4 Numerical evaluation of the customizations

In our experiments, we found that that the customization of §4.2.1 provided the largest speedups, §4.2.2 substantial speedups, and §4.2.3 no speedups. We further describe our observations here.

Variable bounds of §4.2.1. Tables 5 and 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.

Tighter lower bound of §4.2.2. Table 7 shows the lower-bounds for (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.

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 5: Valid bounds on the decision variables in (14) obtained via the SDP relaxation of (19). The runtime describes to the total time spent compute all the bounds of the row.

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 6: Heuristic bounds on the decision variables in (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 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 7: Lower bound $\underline{p}^\star$ of (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.

26

Improved upper bound of §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.

4.3 Structured inner-dual variables

The family solutions to (14) with $N = 1, 2, \dots$ 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 §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 $N = 1, \dots, 5$, and infer the pattern. This heuristic is based on the expectation that the observed patterns will continue to hold for $N = 6, 7, \dots$.

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 (14) by replacing both instances of

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

with

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

and obtain a smaller QCQP.

First solve (14) for $N = 1, \dots, 5$ using the BnB-PEP Algorithm. At this point, solutions may already reveal their pattern in $\operatorname{supp}(\lambda^\star)$. However, optimal inner-dual variables for a given FSFOM are not always unique (see [76] or Table 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

27

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 8: Solutions to (23) with minimum and maximum ℓ_1-norm on optimal λ for the setup of §3 and §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.

processing [38, §2], we promote sparsity of λ by reducing its ℓ_1-norm[5] as follows

$$\left(\begin{array}{ll} \text{minimize} & \|\lambda\|_1 = \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \\ \text{subject to} & \nu R^2 \leq p^\star, \\ & \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} a_{i,j} = 0, \\ & \nu B_{0,\star} - C_{N,\star} - \mu^2 B_{N,\star}(\alpha^\star) + 2\mu A_{\star,N}(\alpha^\star) + \\ & \quad \sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left(A_{i,j}(\alpha^\star) + \frac{1}{2(L-\mu)} C_{i,j} \right) = Z, \\ & Z \succeq 0, \\ & \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \\ & \nu \geq 0, \end{array}\right) \tag{23}$$

where ν, λ, and $Z \in \mathbb{S}_+^{N+2}$ are the decision variables. The constraint set of (23) is almost identical to (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 (14) and, with the FSFOM fixed, the problem is efficiently solved as an SDP. Denote the solution to (23) by $\{\nu^\star, \lambda^{\star,\text{sparse}}, Z^{\star,\text{sparse}}\}$. Hopefully, the optimal $\lambda^{\star,\text{sparse}}$ for $N = 1, \ldots, 5$ are sparse and their structure reveals a pattern.

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 4 ([39, 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 (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 [34] 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 9 indicates. Other optimized method throughout the literature such as OGM, ITEM, OGM-G [32, 43, 78, 45] also have corresponding low-rank $Z^\star$.

[5] One could consider further advanced approaches for promoting sparsity such as [17].

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 (13) can be interpreted as a convergence proof combining inequalities

$$f_i \geq f_j + \langle g_j, x_i - x_j \rangle + \frac{1}{2(L-\mu)} \|g_i - g_j\|^2 \tag{24}$$

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 [70, §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, $\text{rank}(Z^\star)$ corresponds to the excess quadratic terms arising within a proof. For convergence proof of FSFOMs of the form,

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

$r = \text{rank}(Z^\star)$ corresponds to the number of quadratic terms $\{\|c_i\|^2\}_{i=1,\ldots,r}$, roughly speaking. Since $\text{rank}(Z^\star)$ 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 9 presents $\text{supp}(\lambda^{\star,\text{sparse}})$, $\text{rank}(Z^{\star,\text{sparse}})$, and the non-zero columns of $P^{\star,\text{sparse}}$ from solving the convex SDP (23). We use $\mu = 0.1$, $L = 1$, and $R = 1$. From the results, we infer the pattern

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

which has only $3N+2$ components compared to the $(N+2)(N+1)$ of the full index set. Furthermore, $\text{rank}(Z^{\star,\text{sparse}}) = 1$. For $N = 1, \ldots, 5$, we verified that there are globally optimal 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 [32, 43, 78, 45] 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 (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 $N = 1, \ldots, 5$.

N	$\mathrm{supp}(\lambda^\star)$	Rank of $Z^{\star,\mathrm{sparse}}$	Index of nonzero column of $P^{\star,\mathrm{sparse}}$	Runtime (s) for solving (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 9: Structure of the inner-dual variables obtained from the convex SDP (23). The last column shows the runtime to solve (23). (Table 4 shows the runtime to solve (14), a prerequisite for solving (23).)

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 §3.1.

(i) **Infinite-dimensional inner optimization problem.** Construct an infinite-dimensional representation of ($\mathcal{O}^{\mathrm{inner}}$) analogous to (6) of §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 (7) of §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 (12) in §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 (13) of §3.1. Assume strong duality.

(v) **Formulating ($\mathcal{O}^{\mathrm{outer}}$) as a QCQP.** Using Lemma 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 ($\mathcal{O}^{\mathrm{outer}}$) as a QCQP analogous to (14) of §3.1. When f is nonconvex, certain cubic or trilinear terms may arise, whereas for a convex f the nonlinear terms are bilinear

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 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.

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 §6.2 and §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 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 §4. We find bounds on optimal solutions through implied linear constraints, SDP relaxation, and a heuristic. We 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.

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 §1.3. We empirically observed that, among the two approaches of §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.

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 [60], OGM [32, 43], ITEM [78], and many others [47, 36] 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* [88, 66]. 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 §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 [88] 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 $(\mathcal{O}^{\text{outer}})$:

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

Derivation of BnB-PEP-QCQP. Following §5 Step (i), formulate the inner optimization problem $(\mathcal{O}^{\text{inner}})$ as

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

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

where f and $x_0, \ldots, x_N$ are the decision variables. Write $h = \{h_i\}_{i \in [0:N-1]}$. Following §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 §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 (8), $\{\mathbf{x}_i, \mathbf{g}_i, \mathbf{f}_i\}_{i \in I_N^\star}$ using the same encoding as (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 (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

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

where $F \in \mathbb{R}^{1 \times (N+1)}$ and $G \in \mathbb{R}^{(N+2) \times (N+2)}$ are the decision variables. Following §5 Step (iv), construct the dual:

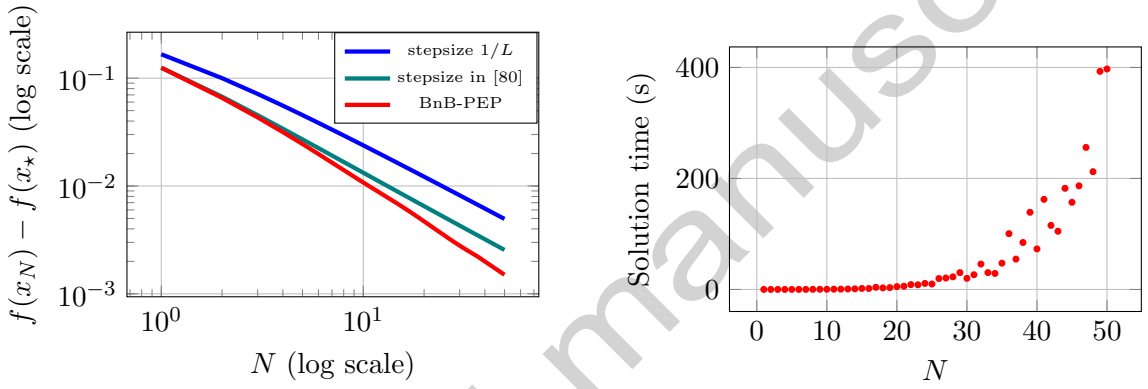
$$\mathcal{R}\left(M, \mathcal{E}, \mathcal{F}, \mathcal{C}\right)$$
$$= \left(\begin{array}{l} \text{minimize} \quad \nu R^2 \\ \text{subject to} \\ \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} - a_{\star,N} = 0, \\ \nu B_{0,\star} + \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(A_{i,j}(h)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

33

where ν, λ, and Z are the decision variables. Assume that strong duality holds. Finally, following §5 Step (v), use Lemma 3 to pose $(\mathcal{O}^{\text{outer}})$ as the following BnB-PEP-QCQP:

$$\begin{aligned}
&\mathcal{R}^\star\left(\mathcal{G}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} - a_{\star,N} = 0, \\
\nu B_{0,\star} + \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(A_{i,j}(h) + \frac{1}{2L} C_{i,j} \right) = Z, \\
P \text{ is lower triangular with nonnegative diagonals}, \\
PP^\top = Z, \\
\nu \geq 0, \ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j,
\end{array} \right)
\end{aligned} \tag{25}$$

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



(a) **(b)**

Figure 1: Numerical results for computing locally optimal stepsizes by solving (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).

Numerical results. Tables 11 and 12 present the results of solving (25) with $L = 1$, $R = 1$, $N = 1, \ldots, 5, 10, 25$ using the BnB-PEP Algorithm. We compare the optimal stepsize with the constant normalized stepsize $h_i = 1$ for all i, which is known to be optimal among $h_i \in (0, 1]$ [27, Corollary 2.8, Theorem 2.9], and constant normalized stepsize h satisfying

$$\frac{1}{2Nh + 1} = (1 - h)^{2N}, \tag{26}$$

which is conjectured by Taylor, Hendrickx, and Glineur to be the optimal constant normalized stepsize [80, §4.1.1]. The stepsizes presented in Table 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 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

N	# variables	# constraints	Worst-case $f(x_N) - f(x_\star)$			Runtime of the BnB-PEP Algorithm
			Optimal	For stepsize in [80, §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 11: Comparison between the performances of the optimal method obtained by solving (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 [80, §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 12: Globally optimal stepsizes obtained by solving (25) with the BnB-PEP Algorithm.

35

h_i satisfying (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 1 shows that the computed stepsizes h^{lopt} outperforms both constant stepsizes. Figure 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 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.

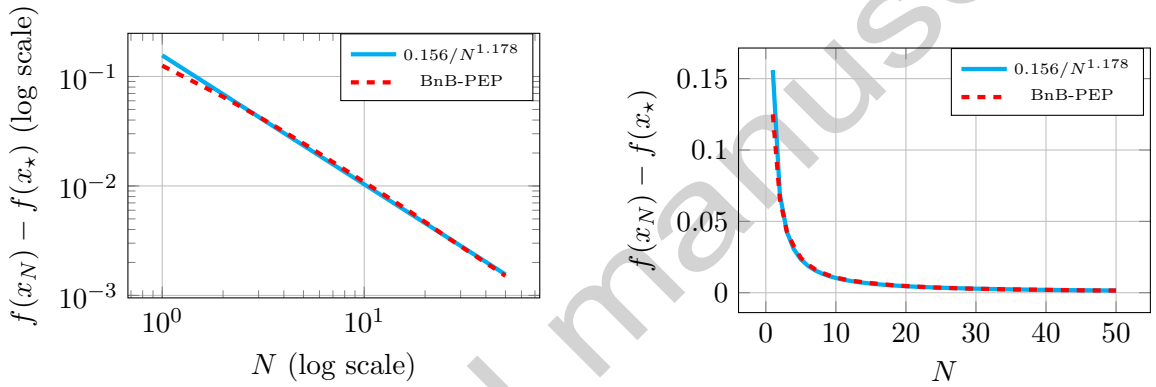


Figure 2: Fitting the worst-case performance of $f(x_N) - f_\star$ corresponding to the locally optimal stepsizes obtained by solving (25) with the first two stages of the BnB-PEP Algorithm yields $0.156/N^{1.178}$. The asymptotic rate may be faster than $\mathcal{O}(1/k)$.

6.1.1 Acceleration is impossible without long steps

Consider the univariate functions

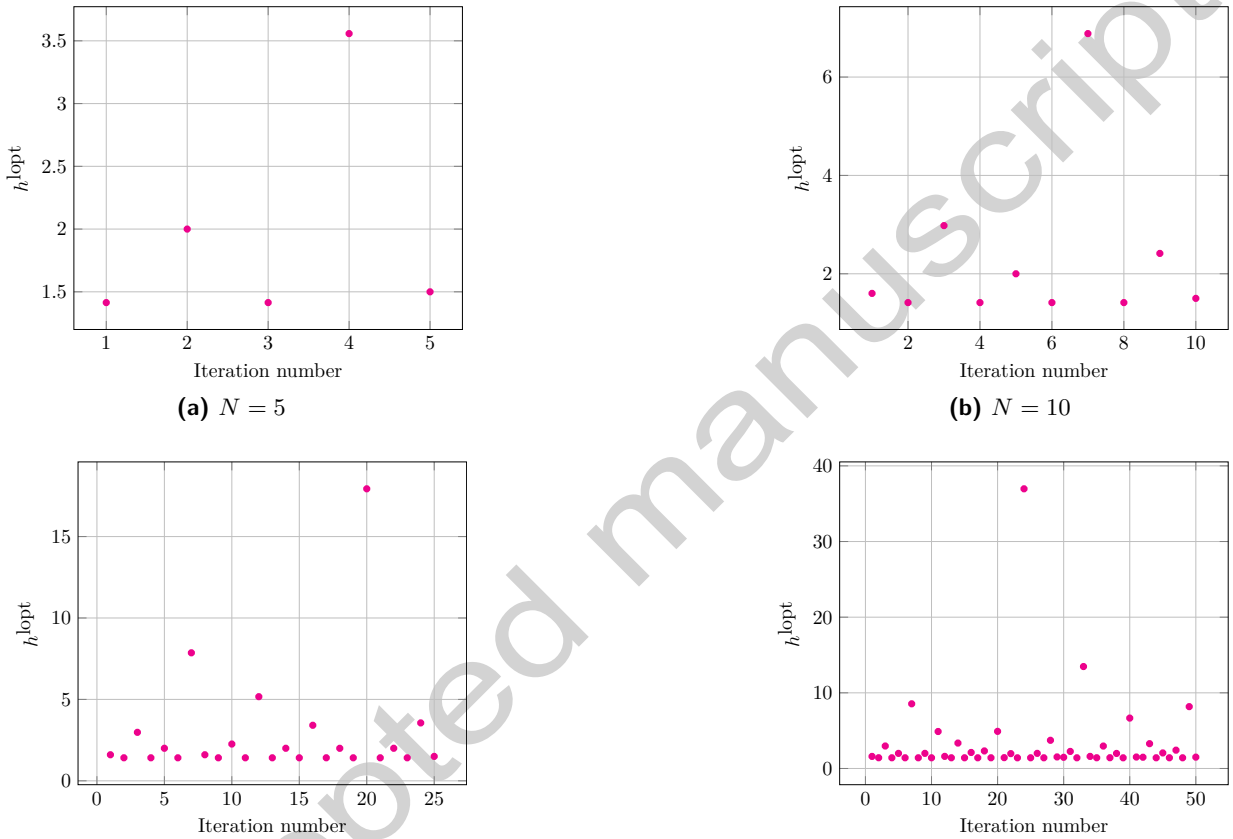
$$f_1(x) = \begin{cases} \frac{LR}{2Nh+1}|x| - \frac{LR^2}{2(2Nh+1)^2} & \text{if } |x| \geq \frac{R}{2Nh+1} \\ \frac{L}{2}x^2 & \text{otherwise} \end{cases}$$

$$f_2(x) = \frac{L}{2}x^2,$$

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 are

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



(a) $N = 5$ **(b)** $N = 10$

(c) $N = 25$ **(d)** $N = 50$

Figure 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 [88] 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.

37

$$f_2(x_N) = \frac{LR^2}{2}(1-h)^2.$$

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 [32, Theorem 2] and [80, §4.1.1].

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]

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

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

$$x_i = x_{i-1} - \sum_{j=0}^{i-1} \frac{h_{i,j}}{L}\nabla f(x_j) = x_0 - \sum_{j=0}^{i-1} \frac{\overline{h}_{i,j}}{L}\nabla f(x_j), \tag{27}$$

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

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

Derivation of BnB-PEP-QCQP. Following §5 Step (i), formulate the inner optimization problem $(\mathcal{O}^{\text{inner}})$ as

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

where f and $x_0, \dots, x_N$ are the decision variables. Write $\overline{h} = \{\overline{h}_{i,j}\}_{0\le j<i\le N}$. To follow the interpolation argument of §5 Step (ii), we use the following interpolation result.

[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 $N \to \infty$ [23, page 3, paragraph 2][29, Remark after Theorem 1].

38

Lemma 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 [29, Theorem 7] into the form of [79, Theorem 3.10]. (Note that journal version of [79, 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 §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 (8), $\{\mathbf{x}_i, \mathbf{g}_i, \mathbf{f}_i\}_{i\in I_N^\star}$ using the same encoding as (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 (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

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

[7] The first condition can be viewed as a discretization of the following condition [79, Theorem 3.10]: $f \in \mathcal{F}_{-L,L}$ if and only if

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

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

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

where $F \in \mathbb{R}^{1 \times (N+1)}$ and $G \in \mathbb{R}^{(N+2)\times(N+2)}$ are the decision variables. Following §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 §5 Step (v), use Lemma 3 to pose $(\mathcal{O}^{\text{outer}})$ as the following BnB-PEP-QCQP:

$$
\begin{aligned}
&\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} a_{i,j} + \sum_{i \in I_N^\star} \tau_i a_{i,\star} + \nu a_{\star,0} = 0, \\
-\sum_{i \in [0:N]} \eta_i C_{i,\star} + \sum_{i,j \in I_N^\star : i \neq j} \lambda_{i,j} \left(-\frac{L}{4} \Theta_{i,j} + \frac{1}{2} \widetilde{A}_{i,j}(\overline{h}) + \frac{1}{4L} C_{i,j} \right) \\
\qquad\qquad + \frac{1}{2L} \sum_{i \in [0:N]} \tau_i C_{i,\star} = Z, \\
\sum_{i \in [0:N]} \eta_i = 1, \\
P \text{ is lower triangular with nonnegative diagonals}, \\
P P^\top = Z, \\
\Theta_{i,j} = B_{i,j}(\overline{h}), \quad i,j \in I_N^\star : i \neq j, \\
\nu \geq 0,\ \tau_i \geq 0,\ \lambda_{i,j} \geq 0, \quad i,j \in I_N^\star : i \neq j, \\
\eta_i \geq 0, \quad i \in [0:N],
\end{array} \right)
\end{aligned}
\tag{28}
$$

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 13 and 14 present the results of solving (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 [2]. To clarify, the stepsizes $h = \{h_{i,j}\}_{0 \le j < i \le N}$ and $\overline{h} = \{\overline{h}_{i,j}\}_{0 \le j < i \le N}$ are as defined in (27). We obtain the optimal $\overline{h}^\star$ from the BnB-PEP Algorithm, solve for $h^\star$ with (4), and present $h^\star$ in the table. The stepsizes presented in Table 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 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 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 1. *The optimal FSFOM for reducing gradient of smooth nonconvex functions satisfies*

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

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

Also, Figure 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 form":

$$\begin{aligned} y_{i+1} &= x_i - \frac{1}{L}\nabla f(x_i) \\ x_{i+1} &= y_{i+1} + \zeta_{i+1}(y_{i+1} - y_i) + \eta_{i+1}(y_{i+1} - x_i) \end{aligned} \tag{29}$$

for $i \in [0 : N-1]$, with coefficients $\{\zeta_i\}_{i \in [1:N]}$ and $\{\eta_i\}_{i \in [1:N]}$. Table 15 shows the equivalent optimal coefficients.

The class of FSFOMs in momentum form is a strict subset of the class of FSFOMs [80, §4.2]. Nesterov's fast gradient method is expressed in the momentum form (29) with $\eta_i = 0$ for all i. Many other accelerated gradient methods such as OGM [32, 43], OGM-G [45], Simple-OGM and SC-OGM [63], FISTA [8], FISTA-G [47], EAG [87, 83], TMM [84, 47], ITEM [78], ORC-F$_\flat$, OBL-F$_\flat$, and OBL-G$_\flat$ [64], and M-OGM-G [89, 47] can all be expressed in the momentum form (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 $\{\eta_i^\star\}_{i \in [1:N]}$ for $N = 6, \ldots, 25$ are provided as Supplementary Information and also as a data file at:

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 13: Comparison between the performances of the optimal method obtained by solving (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 14: Globally optimal stepsizes obtained by solving (28) with the BnB-PEP Algorithm.

42

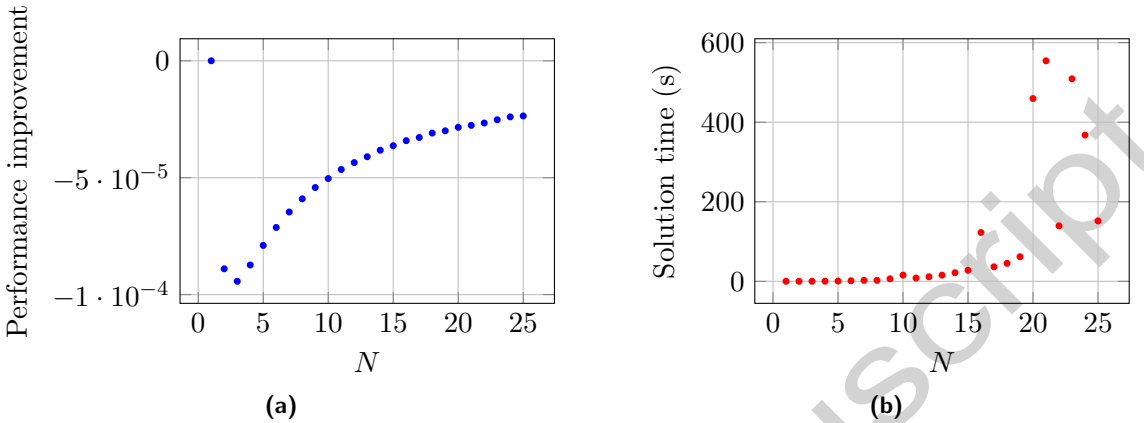


Figure 4: Numerical results associated with the stepsizes by solving (28) with the BnB-PEP Algorithm. (Left) Improvement in the worst-case guarantee of $\min_{i\in[0:N]} \|\nabla f(x_i)\|^2$ vs. iteration count N. (Right) Runtimes of the BnB-PEP Algorithm to compute locally optimal solutions (including Stages 1 and 2 but excluding Stage 3).

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 15: Momentum form coefficients $\{\zeta_i^\star\}_{i\in[1:N]}$ and $\{\eta_i^\star\}_{i\in[1:N]}$ for (29) of the optimal method obtained by solving (28) with the BnB-PEP Algorithm.

43

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

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

$$x_{i+1} = x_i - \frac{h}{\rho} f'(x_i) \tag{30}$$

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 [49] and Taylor, Van Scoy, and Lessard [82] and the philosophy goes further back to the classical Lyapunov stability problem of control theory [54]. Our approach, in particular, closely follows and generalizes the work of Taylor and Bach [77, 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 [22, 21, 23].

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 $N \to \infty$ [23, 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

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

[68, (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 $x \in \mathbb{R}^d$ [23, 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

$$\begin{aligned}y_i &= \mathbf{prox}_{(1/2)f}(x_i)\\ f'(y_i) &= 2(x_i - y_i).\end{aligned} \tag{31}$$

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 [23]. Note that

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

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 (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 (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 (31) and therefore is not arbitrary.

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 (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

$$\|f'(y_k)\|^2 + \psi_{k+1} - \psi_k \leq c_k\|f'(x_k)\|^2 \tag{32}$$

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 (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

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

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 (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

$$\left(\begin{array}{ll}\text{minimize} & \frac{1}{N+1}\left(\widetilde{L}^2\sum_{i=0}^{N}c_i+b_0R^2\right)\\ \text{subject to} & (b_{k+1},b_k,c_k)\in\mathcal{V}_k(h),\quad k\in[0:N+1],\end{array}\right) \tag{33}$$

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 (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

$$\left(\begin{array}{ll}\text{minimize} & (1/(N+1))\left(\widetilde{L}^2\sum_{i=0}^{N}c_i+b_0R^2\right)\\ \text{subject to} & (b_{k+1},b_k,c_k)\in\tilde{\mathcal{V}}_k(h),\quad k\in[0:N+1]\end{array}\right) \tag{34}$$

where $\{b_i\}_{i\in[0:N+1]}$, $\{c_i\}_{i\in[0:N]}$, and $h\in\mathbb{R}$ are the decision variables. Note, (33) is upper bounded by (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 (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 (42).

46

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:

$$\left(\begin{array}{ll} \text{maximize} & \|f'(y_k)\|^2 + b_{k+1}\left(f(y_{k+1}) - f(x_\star) + \|x_{k+1} - y_{k+1}\|^2\right) \\ & -b_k\left(f(y_k) - f(x_\star) + \|x_k - y_k\|^2\right) - c_k\|f'(x_k)\|^2 \\ \text{subject to} & x_{k+1} = x_k - hf'(x_k) \\ & y_k = x_k - \frac{1}{2}f'(y_k) \\ & y_{k+1} = x_{k+1} - \frac{1}{2}f'(y_{k+1}), \\ & f'(x_\star) = 0,\ x_\star = 0,\ f(x_\star) = 0, \\ & f(w) \geq f(x_\star), \quad w \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \\ & f \in \mathcal{W}_{1,\widetilde{L}}, \end{array}\right) \tag{35}$$

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:

$$\left(\begin{array}{ll} \text{maximize} & \|f'(y_k)\|^2 + b_{k+1}\left(f(y_{k+1}) - f(x_\star) + \|x_{k+1} - y_{k+1}\|^2\right) \\ & -b_k\left(f(y_k) - f(x_\star) + \|x_k - y_k\|^2\right) - c_k\|f'(x_k)\|^2 \\ \text{subject to} & x_{k+1} = x_k - hf'(x_k) \\ & y_k = x_k - \frac{1}{2}f'(y_k) \\ & y_{k+1} = x_{k+1} - \frac{1}{2}f'(y_{k+1}), \\ & f'(x_\star) = 0,\ x_\star = 0,\ f(x_\star) = 0, \\ & f(w) \geq f(x_\star), \quad w \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \\ & f(w') \geq f(w) + \langle f'(w) \mid w' - w\rangle - \frac{1}{2}\|w' - w\|^2, \\ & \qquad\qquad w, w' \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \\ & \|f'(w)\|^2 \leq \widetilde{L}^2, \quad w \in \{x_k, x_{k+1}, y_k, y_{k+1}\}, \end{array}\right) \tag{36}$$

where f, x_k, x_{k+1}, y_k, and y_{k+1} are the decision variables. Since the constraints of (35) imply the constraints of (36), the optimal value of (35) is upper bounded by (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

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

47

Then we can express (36) equivalently as

$$\left(\begin{array}{ll} \text{maximize} & \|g_2\|^2 + b_{k+1}\left(f_3 - f_\star + \|w_1 - w_3\|^2\right) \\ & -(b_k\left(f_2 - f_\star + \|w_0 - w_2\|^2\right) - c_k\|g_0\|^2 \\ \text{subject to} & w_1 = w_0 - hg_0 \\ & w_2 = w_0 - \frac{1}{2}g_2, \\ & w_3 = w_1 - \frac{1}{2}g_3, \\ & (w_\star, g_\star, f_\star) = (0,0,0), \\ & f(w_i) \geq f(x_\star), \quad i \in [0:3], \\ & f_i \geq f_j + \langle g_j \mid w_i - w_j\rangle - \frac{1}{2}\|w_i - w_j\|^2, \quad i,j \in [0:3]\cup\{\star\} : i \neq j, \\ & \|g_i\|^2 \leq \widetilde{L}^2, \qquad i \in [0:3]\cup\{\star\}, \end{array}\right) \tag{37}$$

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 (37) as an SDP. For $k \in [0:N]$, let

$$\begin{aligned} H^{[k]} &= [w_0 \mid g_0 \mid g_1 \mid g_2 \mid g_3] \in \mathbb{R}^{d\times 5}, \\ G^{[k]} &= H^{[k]\top} H^{[k]} \in \mathbb{S}^5_+, \\ F^{[k]} &= [f_0 \mid f_1 \mid f_2 \mid f_3] \in \mathbb{R}^{1\times 4}. \end{aligned} \tag{38}$$

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

$$\begin{aligned} &\mathbf{g}_\star = 0 \in \mathbb{R}^5, \mathbf{g}_i = e_{i+2} \in \mathbb{R}^5,\ i \in [0:3], \\ &\mathbf{f}_\star = 0 \in \mathbb{R}^4, \mathbf{f}_i = e_{i+1} \in \mathbb{R}^4,\ i \in [0:3], \\ &\mathbf{w}_\star = 0 \in \mathbb{R}^5, \\ &\mathbf{w}_0 = e_1 \in \mathbb{R}^5, \\ &\mathbf{w}_1 = \mathbf{w}_0 - h\mathbf{g}_0 \in \mathbb{R}^5, \\ &\mathbf{w}_2 = \mathbf{w}_0 - \frac{1}{2}\mathbf{g}_2 \in \mathbb{R}^5, \\ &\mathbf{w}_3 = \mathbf{w}_1 - \frac{1}{2}\mathbf{g}_3 \in \mathbb{R}^5. \end{aligned}$$

Furthermore, define

$$\begin{aligned} A_{i,j}(h) &= \mathbf{g}_j \odot (\mathbf{w}_i - \mathbf{w}_j), \\ B_{i,j}(h) &= (\mathbf{w}_i - \mathbf{w}_j) \odot (\mathbf{w}_i - \mathbf{w}_j), \\ C_{i,j} &= (\mathbf{g}_i - \mathbf{g}_j) \odot (\mathbf{g}_i - \mathbf{g}_j), \\ a_{i,j} &= \mathbf{f}_j - \mathbf{f}_i, \end{aligned}$$

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

$$\begin{aligned} &w_i = H^{[k]}\mathbf{w}_i,\ g_i = H^{[k]}\mathbf{g}_i,\ f_i = F^{[k]}\mathbf{f}_i, \\ &\langle g_j \mid w_i - w_j\rangle = \mathbf{tr}\, G^{[k]} A_{i,j}(h), \\ &\|w_i - w_j\|^2 = \mathbf{tr}\, G^{[k]} B_{i,j}(h), \text{ and} \\ &\|g_i - g_j\|^2 = \mathbf{tr}\, G^{[k,]} C_{i,j}. \end{aligned}$$

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 (37) as

$$\left(\begin{array}{ll} \text{maximize} & \mathbf{tr}\, G^{[k]}Q^{[k]} + F^{[k]}q^{[k]} \\ \text{subject to} & F^{[k]}a_{i,\star} \leq 0, \quad i \in [0:3], \quad \triangleright \text{ dual var. } \tau_i^{[k]} \geq 0 \\ & F^{[k]}a_{i,j} + \mathbf{tr}\, G^{[k]}\left(A_{i,j}(h) - \frac{1}{2}B_{i,j}(h)\right) \leq 0, \quad i,j \in [0:3] \cup \{\star\} : i \neq j, \quad \triangleright \text{ dual var. } \lambda_{i,j}^{[k]} \geq 0 \\ & \mathbf{tr}\, G^{[k]}C_{i,\star} \leq \widetilde{L}^2 \quad i \in [0:3] \cup \{\star\}, \quad \triangleright \text{ dual var. } \eta_i^{[k]} \geq 0 \\ & -G^{[k]} \preceq 0, \quad \triangleright \text{ dual var. } Z \succeq 0 \end{array}\right) \tag{39}$$

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:

$$\left(\begin{array}{ll} \text{minimize} & \widetilde{L}^2 \sum_{i\in[0:3]\cup\{\star\}} \eta_i^{[k]} \\ \text{subject to} & -Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]}\left(A_{i,j}(h) - \frac{1}{2}B_{i,j}(h)\right) + \sum_{i\in[0:3]\cup\{\star\}} \eta_i C_{i,\star} = Z^{[k]}, \\ & -q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0, \\ & \lambda_{i,j}^{[k]} \geq 0,\ \eta_i^{[k]} \geq 0, \quad i,j \in [0:3] \cup \{\star\} : i \neq j, \\ & \tau_i^{[k]} \geq 0, \quad i \in [0:3], \\ & Z^{[k]} \succeq 0, \end{array}\right) \tag{40}$$

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 (39) is upper bounded by (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 (40) to be less than equal to zero, we must have $\eta^{[k]} = 0$. Hence, (40) simplifies into the feasibility problem

$$\left(\begin{array}{ll} \text{minimize} & 0 \\ \text{subject to} & -Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]}\left(A_{i,j}(h) - \frac{1}{2}B_{i,j}(h)\right) = Z^{[k]}, \\ & -q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0, \\ & \lambda_{i,j}^{[k]} \geq 0, \quad i,j \in [0:3] \cup \{\star\} : i \neq j, \\ & \tau_i^{[k]} \geq 0, \quad i \in [0:3], \\ & Z^{[k]} \succeq 0 \end{array}\right) \tag{41}$$

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

Optimizing potential function analysis. We have shown that existence of a feasible point for (41) implies $(b_{k+1}, b_k, c_k) \in \tilde{\mathcal{V}}_k(h)$, which in turn implies (32) holds. Finally, use Lemma 3 to formulate

(34) as the following BnB-PEP-QCQP:

$$\left(\begin{array}{ll} \text{minimize} & \widetilde{L}^2\left[\frac{1}{N+1}\left(\sum_{i=0}^{N} c_i + \frac{R^2}{\widetilde{L}^2} b_0\right)\right] \\ \text{subject to} & -Q^{[k]} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]}\left(A_{i,j}(h) - \frac{1}{2}\Theta_{i,j}^{[k]}\right) = Z^{[k]}, \quad k\in[0:N], \\ & -q^{[k]} + \sum_{i\in[0:3]} \tau_i^{[k]} a_{i,\star} + \sum_{i,j\in[0:3]\cup\{\star\}:i\neq j} \lambda_{i,j}^{[k]} a_{i,j} = 0, \quad k\in[0:N], \\ & \lambda_{i,j}^{[k]} \geq 0, \quad i,j\in[0:3]\cup\{\star\} : i\neq j, \quad k\in[0:N], \\ & \tau_i^{[k]} \geq 0, \quad i\in[0:3], \quad k\in[0:N], \\ & P^{[k]} \text{ is lower triangular with nonnegative diagonals}, \quad k\in[0:N], \\ & P^{[k]}(P^{[k]})^\top = Z^{[k]}, \quad k\in[0:N], \\ & \Theta_{i,j}^{[k]} = B_{i,j}(h), \quad i,j\in[0:3]\cup\{\star\} : i\neq j, \quad k\in[0:N], \end{array}\right) \tag{42}$$

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 constraints arising from $B_{i,j}(h)$ as quadratic constraints. A feasible solution of (42) provides a performance guarantee on the FSFOM defined by the stepsize h.

6.3.3 Numerical results and analytical convergence proofs

Tables 16 and 17 shows the results of solving (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 5, we compare the locally optimal stepsizes h^{lopt} with the stepsize $h = R/(\widetilde{L}\sqrt{(N+1)})$ reported in the proof of [23, Theorem 3.1].

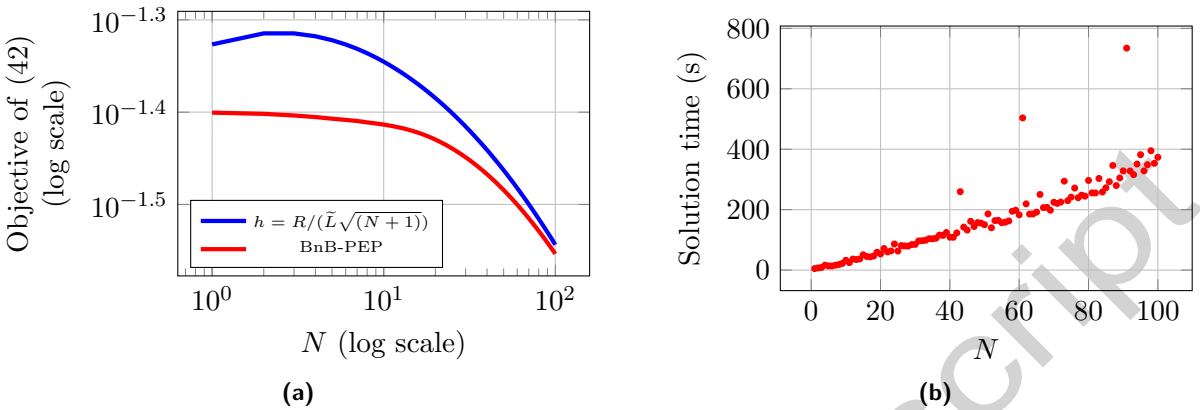
In the following, we use the numerical results to obtain an analytical form of the stepsize and its convergence proof.

Structured inner-dual variables. To find an analytical proof of an FSFOM defined by h, i.e., to find a feasible solution of (42), we use the methodology of §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 $\lambda_{2,3}^{\star[k]}$, $\lambda_{2,0}^{\star[k]}$, and $\lambda_{0,2}^{\star[k]}$ are nonzero. Furthermore, $\lambda_{2,0}^{\star[k]} = \lambda_{0,2}^{\star[k]}$.
- Only $\tau_2^{\star[k]}$ is nonzero.
- Only $Z_{2,2}^{\star[k]}$, $Z_{4,2}^{\star[k]} = Z_{2,4}^{\star[k]}$, $Z_{5,2}^{\star[k]} = Z_{2,5}^{\star[k]}$, $Z_{5,4}^{\star[k]} = Z_{4,5}^{\star[k]}$, $Z_{4,4}^{[k]}$, and $Z_{5,5}^{[k]}$ are nonzero. Furthermore, $Z_{5,2}^{\star[k]} = -Z_{4,2}^{\star[k]}$, $Z_{4,4}^{\star[k]} = Z_{5,5}^{\star[k]}$, and $Z_{4,4}^{\star[k]} = -Z_{5,4}^{\star}$.

Since the pattern is the same for all k, we enforce this pattern for a given k in the constraint set of (42). This leads to a system of equations, and after some tedious but elementary calculations, we

50



(a) **(b)**

Figure 5: Numerical results for the locally optimal stepsizes by solving (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 [23]) 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 $\widetilde{L}^2[(\sum_{i=0}^{N} c_i + (R^2/\widetilde{L}^2)b_0)/(N+1)]$ / Optimal	Worst-case $\widetilde{L}^2[(\sum_{i=0}^{N} c_i + (R^2/\widetilde{L}^2)b_0)/(N+1)]$ / $h = \frac{R}{(\widetilde{L}\sqrt{(N+1)})}$	Runtime of the BnB-PEP Algorithm
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 16: Comparison between the performances of the optimal method obtained by solving (42) with the BnB-PEP Algorithm and the method with $h = R/(\widetilde{L}\sqrt{(N+1)})$ as prescribed by [23]. The BnB-PEP Algorithm was executed on a standard laptop for $N = 1, 2, \ldots, 10$, and on `MIT Supercloud` for $N = 25$.

51

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 17: Globally optimal stepsize obtained by solving (42) with the BnB-PEP Algorithm.

get the simplified form

$$\begin{aligned} &4 + (1-2h)b_{k+1} = b_k, \\ &\lambda_{2,3}^{[k]} = b_{k+1}, \\ &\lambda_{2,0}^{[k]} = \lambda_{0,2}^{[k]} = 2hb_{k+1}, \\ &\tau_2^{[k]} = b_k - b_{k+1}, \end{aligned} \tag{43}$$

for all $k \in [0:N]$. We share Mathematica code for calculating (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 $b_{k+1}^\star$ 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 (43) with the terminal condition $b_{N+1} = 0$ implies

$$b_k = \frac{2}{h}\left(1 - (1-2h)^{N+1-k}\right) \quad \text{for } k \in [0:N+1]. \tag{44}$$

This formula and the resulting values from (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 (43) and $c_k = h^2 b_{k+1}$ into (42) we get

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

[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.

52

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

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 (43) and (44) is a feasible solution of (42), and we have proved the following theorem.

Theorem 1. *Let $N \in \mathbb{N}$. Let $f \in \mathcal{W}_{1,\widetilde{L}}$ have a global minimizer $x_\star$. Let $R > 0$, and let $x_0 \in \mathbb{R}^d$ 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 1. For notational simplicity, define $\kappa = R/L$. With (44), the analytical performance measure minimizes is

$$\begin{aligned} &\frac{\widetilde{L}^2}{N+1} \left(\sum_{i=0}^{N} c_i + \kappa^2 b_0 \right) \\ =&\frac{\widetilde{L}^2}{N+1} \left[-1 + (1-2h)^{N+1} + 2h(N+1) + \kappa^2 \frac{2}{h} \left(1 - (1-2h)^{N+1} \right) \right]. \end{aligned} \tag{45}$$

As an aside, one can directly verify the nonnegativity of (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 $x \geq -1$ [46, page 1]. Plotting (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.

53

To find a simpler analytical stepsize, we construct an upper bound of (45) that does admits a closed-form minimizer:

$$\begin{aligned}
&\frac{\widetilde{L}^2}{N+1}\left[-1+(1-2h)^{N+1}+2h(N+1)+\kappa^2\frac{2}{h}\left(1-(1-2h)^{N+1}\right)\right]\\
\overset{a)}{<}&\frac{\widetilde{L}^2}{N+1}\left[-1+\frac{1}{2(N+1)h+1}+2h(N+1)+\kappa^2\frac{2}{h}\right]\\
\overset{b)}{<}&\frac{\widetilde{L}^2}{N+1}\left[-1+\frac{1}{2(N+1)h}+2h(N+1)+\kappa^2\frac{2}{h}\right]\\
=&\frac{\widetilde{L}^2}{N+1}\left[-1+\frac{1}{2h}\left(\frac{1}{N+1}+4\kappa^2\right)+2h(N+1)\right]. \qquad (46)
\end{aligned}$$

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 $a\in[-1,0], r\in\mathbb{N}$ [46, 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 (46) is achieved at

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

Plugging this back into (46), we have the following analytical performance guarantee:

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

We have proved the following corollary.

Corollary 1. *In the setup of Theorem 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 1 strictly improves upon the prior state-of-the-art rate [23, 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 [23, Theorem 3.1] into [23, Equation 3.4] in the noiseless setup. The claimed rate [23, 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 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 (43).

Note that

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

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 (44)), respectively, and then adding them together, we obtain

$$\begin{aligned} 0 \geq & b_{k+1}\left(f(y_{k+1}) - f(y_k) + \left\langle f'(y_{k+1}) \mid y_k - y_{k+1} \right\rangle - \frac{1}{2}\|y_k - y_{k+1}\|^2 \right) + \\ & 2hb_{k+1}\left(f(x_k) - f(y_k) + \left\langle f'(x_k) \mid y_k - x_k \right\rangle - \frac{1}{2}\|y_k - x_k\|^2 \right) + \\ & 2hb_{k+1}\left(f(y_k) - f(x_k) + \left\langle f'(y_k) \mid x_k - y_k \right\rangle - \frac{1}{2}\|x_k - y_k\|^2 \right) + \\ & (4 - 2hb_{k+1})\left(f(x_\star) - f(y_k)\right) \\ \overset{a)}{=} & b_{k+1}(f(y_{k+1}) - f(x_\star)) - b_k(f(y_k) - f(x_\star)) + \\ & \frac{1}{8}b_{k+1}\Big[-4\|f'(x_k)\|^2 h^2 - 2\left\langle f'(y_k) \mid 2hf'(x_k) + f'(y_{k+1}) \right\rangle \\ & + 4\left\langle f'(x_k) \mid f'(y_{k+1}) \right\rangle h + (4h-1)\|f'(y_k)\|^2 + 3\|f'(y_{k+1})\|^2 \Big] \\ = & b_{k+1}(f(y_{k+1}) - f(x_\star)) - b_k(f(y_k) - f(x_\star)) \\ & + \frac{1}{4}b_{k+1}\left(\|f'(y_{k+1})\|^2 - 4h^2\|f'(x_k)\|^2 - (1-2h)\|f'(y_k)\|^2 \right) + \\ & b_{k+1}\Big[\frac{1}{8}\Big(-4\|f'(x_k)\|^2 h^2 - 2\left\langle f'(y_k) \mid 2hf'(x_k) + f'(y_{k+1}) \right\rangle \\ & + 4\left\langle f'(x_k) \mid f'(y_{k+1}) \right\rangle h + (4h-1)\|f'(y_k)\|^2 + 3\|f'(y_{k+1})\|^2 \Big) - \end{aligned}$$

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

where $a)$ uses (43) and

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

$b)$ removes the non-negative term in the previous line (the last term), $c)$ uses (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.

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 [79, 81, 44], randomized and stochastic methods [74, 77], monotone operator and splitting methods [70, 37, 51, 42], mirror descent methods [26], and adaptive methods [6] are all interesting directions of future work. Recently, novel worst-case convergence rates for nonlinear conjugate gradient methods were established in [20] using the BnB-PEP methodology.

Acknowledgements

EKR were supported by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government (MSIP) [NRF-2022R1C1C1010010] and the Samsung Science and Technology Foundation (Project Number SSTF-BA2101-02). We thank Yoel Drori, Robert M. Freund, Baptiste Goujaud, and Adrien Taylor for careful reading of the manuscript and constructive feedback.

References

[1] Gurobi 10: New Advances. 2021. `https://www.gurobi.com/products/gurobi-optimizer/whats-new-current-release/`.

[2] H. Abbaszadehpeivasti, E. de Klerk, and M. Zamani. The exact worst-case convergence rate of the gradient method with fixed step lengths for l-smooth functions. *Optimization Letters*, 16(6):1649–1661, 2022.

[3] T. 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`.

[4] T. Achterberg and E. Towle. Non-Convex Quadratic Optimization: Gurobi 9.0. 2020. `https://www.gurobi.com/resource/non-convex-quadratic-optimization/`.

[5] M. Barré, A. B. Taylor, and F. Bach. Principled analyses and design of first-order methods with inexact proximal operators. *Mathematical Programming*, 2022.

[6] M. Barré, A. Taylor, and A. d'Aspremont. Complexity guarantees for Polyak steps with momentum. *Conference on Learning Theory*, 2020.

[7] H. H. Bauschke, W. M. Moursi, and X. Wang. Generalized monotone operators and their averaged resolvents. *Mathematical Programming*, 189(1–2):55–74, 2021.

[8] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

[9] H. Y. Benson and R. J. Vanderbei. Solving problems with semidefinite and related constraints using interior-point methods for nonlinear programming. *Mathematical Programming*, 95(2):279–302, 2003.

[10] D. Bertsimas and J. Dunn. *Machine Learning Under a Modern Optimization Lens*. Dynamic Ideas LLC, 2019.

[11] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*, volume 6. Athena Scientific Belmont, MA, 1997.

[12] D. Bertsimas and R. Weismantel. *Optimization Over Integers*, volume 13. Dynamic Ideas Belmont, MA, 2005.

[13] J. Borwein and A. S. Lewis. *Convex Analysis and Nonlinear Optimization, 2nd Edition*. Springer, New York, 2006.

[14] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[15] R. H. Byrd, G. Liu, and J. Nocedal. On the local behavior of an interior point method for nonlinear programming. *Numerical Analysis*, 1997:37–56, 1997.

[16] R. H. Byrd, J. Nocedal, and R. A. Waltz. KNITRO: An integrated package for nonlinear optimization. In G. D. Pillo and M. Roma, editors, *Large-Scale Nonlinear Optimization*, pages 35–59. Springer, 2006.

[17] E. J. Candès, M. B. Wakin, and S. P. Boyd. Enhancing sparsity by reweighted ℓ_1 minimization. *Journal of Fourier Analysis and Applications*, 14(5):877–905, 2008.

[18] F. H. Clarke. *Optimization and Nonsmooth Analysis*. SIAM, 1990.

[19] S. Cyrus, B. Hu, B. Van Scoy, and L. Lessard. A robust accelerated optimization algorithm for strongly convex functions. *American Control Conference*, 2018.

[20] S. Das Gupta, R. M. Freund, X. A. Sun, and A. B. Taylor. Nonlinear conjugate gradient methods: worst-case convergence rates via computer-assisted analyses. *arXiv preprint arXiv:2301.01530*, 2023.

[21] D. Davis and D. Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *arXiv preprint arXiv:1803.06523*, 2018.

[22] D. Davis and D. Drusvyatskiy. Stochastic subgradient method converges at the rate $o(k^{-1/4})$ on weakly convex functions. *arXiv preprint arXiv:1802.02988*, 2018.

[23] D. Davis and D. Drusvyatskiy. Stochastic model-based minimization of weakly convex functions. *SIAM Journal on Optimization*, 29(1):207–239, 2019.

[24] E. de Klerk, F. Glineur, and A. B. Taylor. On the worst-case complexity of the gradient method with exact line search for smooth strongly convex functions. *Optimization Letters*, 11(7):1185–1199, 2017.

[25] E. de Klerk, F. Glineur, and A. B. Taylor. Worst-case convergence analysis of inexact gradient and Newton methods through semidefinite programming performance estimation. *SIAM Journal on Optimization*, 30(3):2053–2082, 2020.

[26] R.-A. Dragomir, A. B. Taylor, A. d'Aspremont, and J. Bolte. Optimal complexity and certification of Bregman first-order methods. *Mathematical Programming*, 194(1–2):41–83, 2022.

[27] Y. Drori. *Contributions to the Complexity Analysis of Optimization Algorithms*. PhD thesis, Tel-Aviv University, Tel Aviv, Israel, 2014.

[28] Y. Drori. The exact information-based complexity of smooth convex minimization. *Journal of Complexity*, 39:1–16, 2017.

[29] Y. Drori and O. Shamir. The complexity of finding stationary points with stochastic gradient descent. *International Conference on Machine Learning*, 2020.

[30] Y. Drori and A. B. Taylor. Efficient first-order methods for convex minimization: A constructive approach. *Mathematical Programming*, 184(1):183–220, 2020.

[31] Y. Drori and A. B. Taylor. On the oracle complexity of smooth strongly convex minimization. *Journal of Complexity*, 68:101590, 2022.

[32] Y. Drori and M. Teboulle. Performance of first-order methods for smooth convex minimization: A novel approach. *Mathematical Programming*, 145(1-2):451–482, 2014.

[33] I. Dunning, J. Huchette, and M. Lubin. JuMP: A modeling language for mathematical optimization. *SIAM Review*, 59(2):295–320, 2017.

[34] M. Fazel, H. Hindi, and S. Boyd. Log-det heuristic for matrix rank minimization with applications to Hankel and Euclidean distance matrices. *American Control Conference*, 2003.

[35] A. V. Fiacco and G. P. McCormick. *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*. SIAM, 1990.

[36] B. Goujaud, D. Scieur, A. Dieuleveut, A. B. Taylor, and F. Pedregosa. Super-acceleration with cyclical step-sizes. *International Conference on Artificial Intelligence and Statistics*, 2022.

[37] G. Gu and J. Yang. Tight sublinear convergence rate of the proximal point algorithm for maximal monotone inclusion problems. *SIAM Journal on Optimization*, 30(3):1905–1921, 2020.

[38] T. Hastie, R. Tibshirani, and M. Wainwright. *Statistical Learning with Sparsity: The Lasso and Generalizations*. Chapman and Hall/CRC, 2019.

[39] N. J. Higham. *Accuracy and Stability of Numerical Algorithms*. SIAM, 2002.

[40] R. A. Horn and C. R. Johnson. *Matrix Analysis*. Cambridge University Press, 2012.

[41] R. Horst and H. Tuy. *Global Optimization: Deterministic Approaches*. Springer Science & Business Media, 2013.

[42] D. Kim. Accelerated proximal point method for maximally monotone operators. *Mathematical Programming*, 190(1–2):57–87, 2021.

[43] D. Kim and J. A. Fessler. Optimized first-order methods for smooth convex minimization. *Mathematical Programming*, 159(1):81–107, 2016.

[44] D. Kim and J. A. Fessler. Another look at the fast iterative shrinkage/thresholding algorithm (FISTA). *SIAM Journal on Optimization*, 28(1):223–250, 2018.

[45] D. Kim and J. 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):192–219, 2021.

[46] L. Kozma. Useful Inequalities. 2021. `https://www.lkozma.net/inequalities_cheat_sheet/ineq.pdf`.

[47] 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.

[48] B. Legat, O. Dowson, J. D. Garcia, and M. Lubin. MathOptInterface: A data structure for mathematical optimization problems. *INFORMS Journal on Computing*, 2021.

[49] L. Lessard, B. Recht, and A. Packard. Analysis and design of optimization algorithms via integral quadratic constraints. *SIAM Journal on Optimization*, 26(1):57–95, 2016.

[50] L. Liberti. Introduction to global optimization. *Ecole Polytechnique*, 2008.

[51] F. Lieder. On the convergence rate of the halpern-iteration. *Optimization Letters*, 15(2):405–418, 2021.

[52] M. Locatelli and F. Schoen. *Global Optimization: Theory, Algorithms, and Applications*. SIAM, 2013.

[53] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine*, 27(3):20–34, 2010.

[54] A. M. Lyapunov. The general problem of the stability of motion. *Communications of the Mathematical Society of Kharkov*, 1892.

[55] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I—Convex underestimating problems. *Mathematical Programming*, 10(1):147–175, 1976.

[56] B. S. Mordukhovich. *Variational Analysis and Generalized Differentiation I: Basic Theory*. Springer, 2006.

[57] MOSEK ApS. *MOSEK Optimizer API for C 9.3.6*, 2019.

[58] A. Nemirovski. Information-based complexity of convex programming. *Lecture Notes, Technion - Israel Institute of Technology*, 1995.

[59] A. Nemirovsky. Information-based complexity of linear operator equations. *Journal of Complexity*, 8(2):153–175, 1992.

[60] Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372–376, 1983.

[61] Y. Nesterov. *Lectures on Convex Optimization*, volume 137. second edition, 2018.

[62] M. Padberg. The boolean quadric polytope: some characteristics, facets and relatives. *Mathematical Programming*, 45(1):139–172, 1989.

[63] C. Park, J. Park, and E. K. Ryu. Factor-$\sqrt{2}$ acceleration of accelerated gradient methods. *Applied Mathematics & Optimization*, 2023.

[64] C. Park and E. K. Ryu. Optimal first-order algorithms as a function of inequalities. *arXiv preprint arXiv:2110.11035*, 2021.

[65] J. Park and E. K. Ryu. Exact optimal accelerated complexity for fixed-point iterations. *International Conference on Machine Learning*, 2022.

60

[66] F. Pedregosa. On the link between optimization and polynomials: acceleration without Momentum. 2021. http://fa.bianp.net/blog/2021/no-momentum/.

[67] A. Reuther, J. Kepner, C. Byun, S. Samsi, W. Arcand, D. Bestor, B. Bergeron, V. Gadepally, M. Houle, M. Hubbell, M. Jones, A. Klein, L. Milechin, J. Mullen, A. Prout, A. Rosa, C. Yee, and P. Michaleas. Interactive supercomputing on 40,000 cores for machine learning and data analysis. In *2018 IEEE High Performance extreme Computing Conference (HPEC)*, pages 1–6. IEEE, 2018.

[68] R. T. Rockafellar. Characterizing firm nonexpansiveness of prox mappings both locally and globally. *Journal of Nonlinear and Convex Analysis*, 22(5), 2021.

[69] R. T. Rockafellar and R. J.-B. Wets. *Variational Analysis*. Springer Science & Business Media, 2009.

[70] E. K. Ryu, A. B. Taylor, C. Bergeling, and P. Giselsson. Operator splitting performance estimation: Tight contraction factors and optimal parameter selection. *SIAM Journal on Optimization*, 30(3):2251–2271, 2020.

[71] E. K. Ryu and W. Yin. *Large-Scale Convex Optimization via Monotone Operators*. Cambridge University Press, 2022.

[72] H. D. Sherali and W. 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):411–430, 1990.

[73] H. D. Sherali and B. M. Fraticelli. Enhancing RLT relaxations via a new class of semidefinite cuts. *Journal of Global Optimization*, 22(1):233–261, 2002.

[74] Z. Shi and R. 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.

[75] A. 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.

[76] A. B. Taylor. Computer-aided analyses in optimization. 2020. https://francisbach.com/computer-aided-analyses/.

[77] A. B. Taylor and F. Bach. Stochastic first-order methods: Non-asymptotic and computer-aided analyses via potential functions. *Conference on Learning Theory*, 2019.

[78] A. B. Taylor and Y. Drori. An optimal gradient method for smooth strongly convex minimization. *Mathematical Programming*, pages 1–38, 2022.

[79] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Exact worst-case performance of first-order methods for composite convex optimization. *SIAM Journal on Optimization*, 27(3):1283–1313, 2017.

[80] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Smooth strongly convex interpolation and exact worst-case performance of first-order methods. *Mathematical Programming*, 161(1–2):307–345, 2017.

[81] A. B. Taylor, J. M. Hendrickx, and F. Glineur. Exact worst-case convergence rates of the proximal gradient method for composite convex minimization. *Journal of Optimization Theory and Applications*, 178(2):455–476, 2018.

[82] A. B. Taylor, B. Van Scoy, and L. Lessard. Lyapunov functions for first-order methods: Tight automated convergence guarantees. *International Conference on Machine Learning*, 2018.

[83] Q. Tran-Dinh. The connection between Nesterov's accelerated methods and Halpern fixed-point iterations. *arXiv preprint arXiv:2203.04869*, 2022.

[84] B. Van Scoy, R. A. Freeman, and K. M. Lynch. The fastest known globally convergent first-order method for minimizing strongly convex functions. *IEEE Control Systems Letters*, 2(1):49–54, 2017.

[85] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming*, 106(1):25–57, 2006.

[86] A. Wächter and L. T. Biegler. Line search filter methods for nonlinear programming: Local convergence. *SIAM Journal on Optimization*, 16(1):32–48, 2005.

[87] T. Yoon and E. K. Ryu. Accelerated algorithms for smooth convex-concave minimax problems with $\mathcal{O}(1/k^2)$ rate on squared gradient norm. *International Conference on Machine Learning*, 2021.

[88] D. Young. On Richardson's method for solving linear systems with positive definite matrices. *Journal of Mathematics and Physics*, 32(1–4):243–255, 1953.

[89] K. Zhou, L. Tian, A. M.-C. So, and J. Cheng. Practical schemes for finding near-stationary points of convex finite-sums. *International Conference on Artificial Intelligence and Statistics*, 2022.

8 Appendix

8.1 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

$$q(x, y, u, v) \triangleq c_2 x^\top x + c_3 y^\top y + c_4 u^\top u + c_5 v^\top v + c_6 x^\top y + c_7 x^\top u + c_8 x^\top v + c_9 y^\top u + c_{10} y^\top v + c_{11} u^\top v + c_{12},$$

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

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

62

for some $M > 0$. Many of the commonly considered the 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 [61, 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 [78, 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 [29, 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 [79, 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 [23, 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 [79, Definition 3.1]

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

The class of ρ-weakly convex functions with L-bounded subgradients $\mathcal{W}_{\rho,L}$ satisfies [23, Lemma 2.1], [79, §3.1 (c)]

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

Some, but not all, of the quadratically representable functions classes have interpolation results analogous to Lemma 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 ([80, Theorem 4], [79, Theorem 3.10], [79, 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 §6.3, but we loose the tightness guarantee.

63

8.2 Discussion on the strong duality assumption

Consider the setup of §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 (12), $\overline{\mathcal{R}}$ is as defined in (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 6. *[80, 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

$$\mathcal{R}(M(\alpha^{\star,\text{true}}),\mathcal{E},\mathcal{F},\mathcal{C}) < \overline{\mathcal{R}}(M(\alpha^{\star,\text{true}}),\mathcal{E},\mathcal{F},\mathcal{C}) \tag{47}$$

$$\mathcal{R}(M(\alpha^{\star,\text{true}}),\mathcal{E},\mathcal{F},\mathcal{C}) < \mathcal{R}(M(\alpha^\star),\mathcal{E},\mathcal{F},\mathcal{C}) \tag{48}$$

$$\overline{\mathcal{R}}(M(\alpha^\star),\mathcal{E},\mathcal{F},\mathcal{C}) \leq \overline{\mathcal{R}}(M(\alpha^{\star,\text{true}}),\mathcal{E},\mathcal{F},\mathcal{C}) \tag{49}$$

Condition(47) states that for $\alpha^{\star,\text{true}}$, the duality gap is positive. Condition (48) states that $\alpha^{\star,\text{true}}$ has a better performance than $\alpha^\star$. Condition (49) reaffirms that $\alpha^\star$ is optimal for dual problem. Figure 6 illustrates this circumstance. While this pathology is probably extremely unlikely, its possibility is a consequence of the gap in the reasoning.

One can rigorously exclude this pathology in most well-behaved setups using the arguments of Park and Ryu [64, 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 $\alpha^\star \in A^{\text{nice}}$.

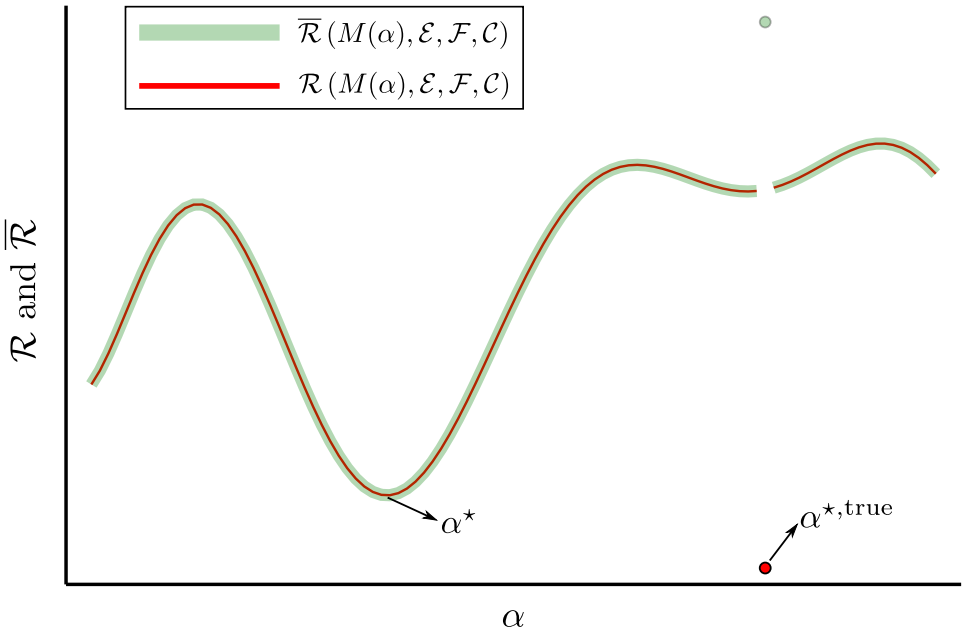


Figure 6: Illustration of $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ and $\overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ in a pathological setup. Even if $\alpha^\star \in \operatorname{argmin} \overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ 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 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 $\alpha^\star \in A^{\text{nice}}$. 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.

8.3 Exact rank-1 nonconvex semidefinite representation of the BnB-PEP-QCQP

In this section, we derive the exact rank-1 nonconvex semidefinite representation of (14). For the other BnB-PEP-QCQPs, the steps to construct such a nonconvex semidefinite representation are identical.

First, we define:

$$w = \operatorname{vec}(\alpha,\nu,\lambda), \tag{50}$$

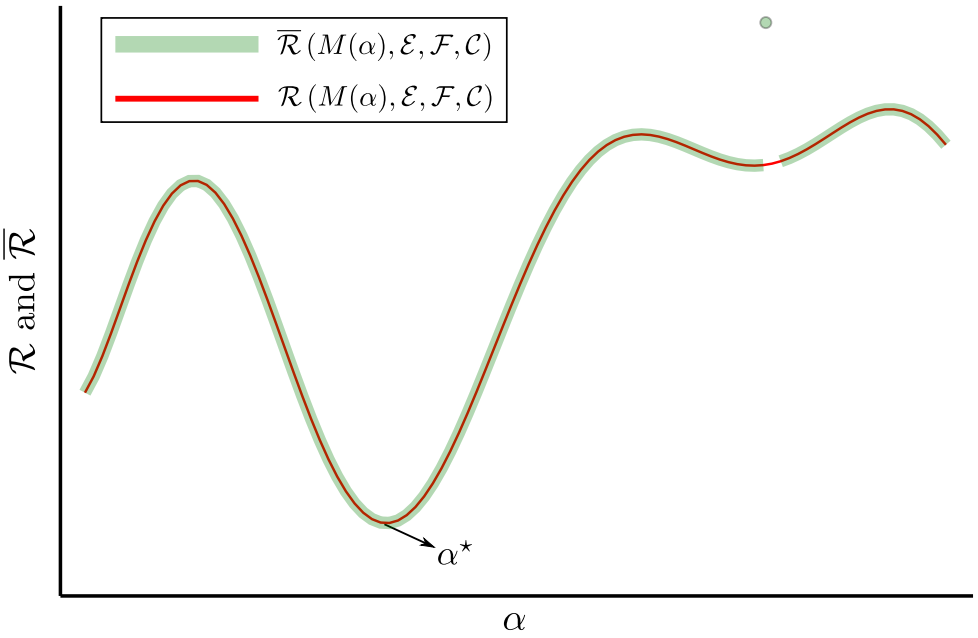


Figure 7: Illustration of $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ and $\overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ when $\mathcal{R}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ is continuous. If $\alpha^\star \in \operatorname{argmin}\overline{\mathcal{R}}(M(\alpha),\mathcal{E},\mathcal{F},\mathcal{C})$ 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.

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 $\{1, 2, \ldots, |w|\}$ *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:

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

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 $i, j \in I_N^\star$, $k, \ell \in [1 : |\mathbf{x}_0|]$:

$$\begin{aligned}
[B_{i,j}(\alpha)]_{k,\ell} &= [(\mathbf{x}_i - \mathbf{x}_j) \odot (\mathbf{x}_i - \mathbf{x}_j)]_{k,\ell} \\
&= [\mathbf{x}_i - \mathbf{x}_j]_k [\mathbf{x}_i - \mathbf{x}_j]_\ell \\
&= \left[\mathfrak{G}^{[i,j]} \mathbf{x}_0 + \mathfrak{H}^{[i,j]} w \right]_k \left[\mathfrak{G}^{[i,j]} \mathbf{x}_0 + \mathfrak{H}^{[i,j]} w \right]_\ell \\
&= \left[c^{[i,j][k]} + \mathfrak{h}^{[i,j][k]\top} w \right] \left[c^{[i,j][\ell]} + \mathfrak{h}^{[i,j][\ell]\top} w \right] \\
&= c^{[i,j][k]} c^{[i,j][\ell]} + c^{[i,j][k]} \mathfrak{h}^{[i,j][\ell]\top} w + c^{[i,j][\ell]} \mathfrak{h}^{[i,j][k]\top} w \\
&\quad + (\mathfrak{h}^{[i,j][k]\top} w)(\mathfrak{h}^{[i,j][\ell]\top} w) \\
&= c^{[i,j][k]} c^{[i,j][\ell]} + c^{[i,j][k]} \mathfrak{h}^{[i,j][\ell]\top} w + c^{[i,j][\ell]} \mathfrak{h}^{[i,j][k]\top} w \\
&\quad + \sum_{\widetilde{i}=1}^{|w|} \sum_{\widetilde{j}=1}^{|w|} \mathfrak{h}_{\widetilde{i}}^{[i,j][k]} \mathfrak{h}_{\widetilde{j}}^{[i,j][\ell]} w_{\widetilde{i}} w_{\widetilde{j}} \\
&= c^{[i,j][k]} c^{[i,j][\ell]} + c^{[i,j][k]} \mathfrak{h}^{[i,j][\ell]\top} w + c^{[i,j][\ell]} \mathfrak{h}^{[i,j][k]\top} w \\
&\quad + w^\top H^{[i,j][k,\ell]} w \\
&= c^{[i,j][k]} c^{[i,j][\ell]} + c^{[i,j][k]} \mathfrak{h}^{[i,j][\ell]\top} w + c^{[i,j][\ell]} \mathfrak{h}^{[i,j][k]\top} w \\
&\quad + \mathbf{tr}\left(H^{[i,j][k,\ell]} W \right),
\end{aligned} \tag{51}$$

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 $k, \ell \in [1 : |\mathbf{x}_0|]$:

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

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

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

$$
\begin{aligned}
&\lambda_{i,j}\left[A_{i,j}(\alpha)\right]_{k,\ell} \\
&= \left(\widetilde{d}^{[i,j]\top} w\right)\left(\widetilde{c}^{[i,j][k,\ell]} + \sum_{\widetilde{i}=1}^{|w|} \widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]} w_{\widetilde{i}}\right) \\
&= \widetilde{c}^{[i,j][k,\ell]}\left(\widetilde{d}^{[i,j]\top} w\right) + \left(\sum_{\widetilde{j}=1}^{|w|} \widetilde{d}_{\widetilde{j}}^{[i,j]} w_{\widetilde{j}}\right)\left(\sum_{\widetilde{i}=1}^{|w|} \widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]} w_{\widetilde{i}}\right) \\
&= \widetilde{c}^{[i,j][k,\ell]}\left(\widetilde{d}^{[i,j]\top} w\right) + \sum_{\widetilde{i}=1}^{|w|}\sum_{\widetilde{j}=1}^{|w|}\left[\widetilde{d}_{\widetilde{j}}^{[i,j]} \widetilde{q}_{\widetilde{i}}^{[i,j][k,\ell]}\right] w_{\widetilde{i}} w_{\widetilde{j}} \\
&= \widetilde{c}^{[i,j][k,\ell]}\left(\widetilde{d}^{[i,j]\top} w\right) + w^\top S^{[i,j][k,\ell]} w \\
&= \widetilde{c}^{[i,j][k,\ell]}\left(\widetilde{d}^{[i,j]\top} w\right) + \mathbf{tr}\left(S^{[i,j][k,\ell]} W\right),
\end{aligned} \tag{52}
$$

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 (51) and (52), we have the following nonconvex semidefinite representation of (14):

$$
\begin{aligned}
&\mathcal{R}^\star\left(\mathcal{M}_N, \mathcal{E}, \mathcal{F}, \mathcal{C}\right) \\
&= \left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{(i,j)\in I_N^\star} \lambda_{i,j} a_{i,j} = 0, \\
\nu \left[B_{0,\star}\right]_{k,\ell} - \left[C_{N,\star}\right]_{k,\ell} - \mu^2 \left[B_{N,\star}(\alpha)\right]_{k,\ell} + \\
\qquad 2\mu \left[A_{\star,N}(\alpha)\right]_{k,\ell} + \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} + \\
\qquad \frac{1}{2(L-\mu)} \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[C_{i,j}\right]_{k,\ell} = Z_{k,\ell}, \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
\left[B_{N,\star}(\alpha)\right]_{k,\ell} = c^{[N,\star][k]} c^{[N,\star][\ell]} + c^{[N,\star][k]} \mathfrak{h}^{[N,\star][\ell]\top} w + c^{[N,\star][\ell]} \mathfrak{h}^{[N,\star][k]\top} w + \\
\qquad \mathbf{tr}\left(H^{[N,\star][k,\ell]} W\right), \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
\sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} = \left(\sum_{i,j\in I_N^\star : i\neq j} \widetilde{c}^{[i,j][k,\ell]} \widetilde{d}^{[i,j]\top} \right) w + \\
\qquad \mathbf{tr}\left(\sum_{i,j\in I_N^\star} S^{[i,j][k,\ell]} \right) W, \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
Z \succeq 0, \\
W = w w^\top, \\
(\forall i,j \in I_N^\star) \quad \lambda_{i,j} \geq 0,\ \nu \geq 0,
\end{array} \right)
\end{aligned}
\tag{53}
$$

where λ, ν, Z, and W are the decision variables, and $w = \mathrm{vec}(\alpha, \nu, \lambda)$ as defined in (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 (14) is given by:

$$
\left(\begin{array}{l}
\text{minimize} \quad \nu R^2 \\
\text{subject to} \\
\sum_{i,j\in I_N^\star} \lambda_{i,j} a_{i,j} = 0, \\
\nu \left[B_{0,\star}\right]_{k,\ell} - \left[C_{N,\star}\right]_{k,\ell} - \mu^2 \left[B_{N,\star}(\alpha)\right]_{k,\ell} + \\
\qquad 2\mu \left[A_{\star,N}(\alpha)\right]_{k,\ell} + \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} + \\
\qquad \frac{1}{2(L-\mu)} \sum_{(i,j)\in I_N^\star} \lambda_{i,j} \left[C_{i,j}\right]_{k,\ell} = Z_{k,\ell}, \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
\left[B_{N,\star}(\alpha)\right]_{k,\ell} = c^{[N,\star][k]} c^{[N,\star][\ell]} + c^{[N,\star][k]} \mathfrak{h}^{[N,\star][\ell]\top} w + c^{[N,\star][\ell]} \mathfrak{h}^{[N,\star][k]\top} w + \\
\qquad \mathbf{tr}\left(H^{[N,\star][k,\ell]} W\right), \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
\sum_{i,j\in I_N^\star : i\neq j} \lambda_{i,j} \left[A_{i,j}(\alpha)\right]_{k,\ell} = \left(\sum_{i,j\in I_N^\star : i\neq j} \widetilde{c}^{[i,j][k,\ell]} \widetilde{d}^{[i,j]\top} \right) w + \\
\qquad \mathbf{tr}\left(\sum_{i,j\in I_N^\star} S^{[i,j][k,\ell]} \right) W \quad k \in [1:|\mathbf{x}_0|],\ \ell \in [1:k], \\
Z \succeq 0, \\
\begin{bmatrix} W & w \\ w^\top & 1 \end{bmatrix} \succeq 0, \\
(\forall i,j \in I_N^\star) \quad \lambda_{i,j} \geq 0,\ \nu \geq 0,
\end{array} \right)
\tag{54}
$$

where λ, ν, Z, and W are the decision variables. The optimal objective value of (54) will provide a lower bound to (53).

69