EXPLOITING PROBLEM STRUCTURE FOR FASTER OPTIMIZATION: A TRILINEAR SADDLE POINT APPROACH

A Dissertation
Presented to
The Academic Faculty

By

Zhe (Jimmy) Zhang

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in Operations Research
H. Milton Stewart School of Industrial and Systems Engineering

Georgia Institute of Technology

August 2023

EXPLOITING PROBLEM STRUCTURE FOR FASTER OPTIMIZATION: A TRILINEAR SADDLE POINT APPROACH

Thesis committee:

Dr. Guanghui Lan (Advisor)
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Alexander Shapiro
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Siva Theja Manguluri
School of Industrial and Systems Engineering
Georgia Institute of Technology

Dr. Richard Tapia
Department of Computational Applied Mathematics and Operations Research
Rice Univertsity

Dr. Arkadi Nemirovski
School of Industrial and Systems Engineering
Georgia Institute of Technology

Date approved: July 17 2023

Not all those who wander are lost.

J.R.R. Tolkien

To my parents

ACKNOWLEDGMENTS

I am profoundly grateful for the enriching six years I've spent at Georgia Tech, immersing myself in operations research and becoming part of its warm and friendly community. The stimulating discussions in offices and hallways are the highlight of this journey, and I will deeply miss them.

Throughout this academic pursuit, which included navigating through the challenges of the pandemic that spanned over half of my journey, I received invaluable support from numerous individuals. Their assistance and encouragement have made this accomplishment possible, and I sincerely appreciate each and every one of them.

At the forefront, my deepest gratitude goes to my advisor, Dr. Guanghui (George) Lan, whose commitment to both rigor and beauty has inspired me to strive for excellence. His wisdom has been an invaluable guiding light, and his unwavering support has carried me through difficult times. I am genuinely thankful for his care and intellectual guidance.

I also want to extend my thanks to the late Dr. Shabbir Ahmed, who initially guided me into my current research direction. His sharp intellect and wisdom will forever remain in my memory. Working with Dr. Siva Theja Manguluri has been a true privilege; his unyielding curiosity and enthusiasm have been a constant source of inspiration. I am indebted to Dr. Richard Tapia for his instrumental role in helping me gain entry into the PhD program. His remarkable life story, courage, and tenacity continue to inspire me.

My gratitude also extends to my committee members, Dr. Siva Theja Manguluri, Dr. Arkadi Nemirovski, Dr. Alex Shapiro, and Dr. Richard Tapia, for generously providing their time and valuable feedback.

I am deeply appreciative of the support and camaraderie offered by my friends, with special thanks to Yijiang Li, Xiaoyi Gu, Sanchong Mou, Sheng Zhang, Hongzhen Tian, Ray Liu, Kaizhao Sun, Shixuan Zhang, Jiaming Liang, Chongzhang Li, Sajad Khodadadian, Sushil Varma, Zhen Zhong, Tianhang Zhu, Chongzhang Li, Wenbo Yang, Hao Yu, and

Yuan Gao. The discussion with my lab mates, including Digivijay Boob, Georgios Kotsalis, Yi Cheng, Tianjiao Li, Caleb Ju, and Yan Li, has always been exciting. I want to give special thanks to Georgios, who kept his fighting spirit alive despite losing all the time on the squash court.

Lastly, no words can express the depth of my gratitude to my parents, Anren Zhang and Fei Luo, for their unwavering love and support throughout this journey and beyond. Special thanks also go to my girlfriend, Shulin Zhu, whose encouragement and assistance have been instrumental in achieving all that I have.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

SUMMARY

Optimization is vital in operations research, encompassing model fitting and decision-making. The exponential growth of data holds promise for realistic models and intelligent decision-making. However, the sheer volume and the exceptional dimension of big data make computations prohibitively expensive and time-consuming. In this thesis, we propose a trilinear saddle point approach to tackle some challenges in big-data optimization. By effectively leveraging problem structure, our approach significantly improves computation complexities for a few important problem classes in stochastic programming and non-linear programming. This offers valuable insights into the intrinsic computational hardness.

In Chapter Two, we consider a distributionally robust two-stage stochastic optimization problem with discrete scenario support. While much research effort has been devoted to tractable reformulations for DRO problems, especially those with continuous scenario support, few efficient numerical algorithms are developed, and most of them can neither handle the nonsmooth second-stage cost function nor the large number of scenarios K effectively. We fill the gap by reformulating the DRO problem as a trilinear min-max-max saddle point problem and developing novel algorithms that can achieve an $O(1/\epsilon)$ iteration complexity which only mildly depends on the scenario number . The major computations involved in each iteration of these algorithms can be conducted in parallel if necessary. Besides, for solving an important class of DRO problems with the Kantorovich ball ambiguity set, we propose a slight modification of our algorithms to avoid the expensive computation of the probability vector projection. Finally, preliminary numerical experiments are conducted to demonstrate the empirical advantages of the proposed algorithms.

In Chapter Three, we study the convex nested stochastic composite optimization (NSCO) problem, which finds applications in reinforcement learning and risk-averse optimization. Existing NSCO algorithms exhibit significantly worse stochastic oracle complexities compared to those without nested structures, and they require all outer-layer functions

to be smooth. To address these challenges, we propose a stochastic trilinear (multi-linear) saddle point formulation that enables the design of order-optimal algorithms for general convex NSCO problems. When all outer-layer functions are smooth, we propose a stochastic sequential dual (SSD) method to achieve an oracle complexity of $O(1/\epsilon^2)$ ($O(1/\epsilon)$) when the problem is non-strongly (strongly) convex. In cases where there are structured non-smooth or general non-smooth outer-layer functions, we propose a nonsmooth stochastic sequential dual (nSSD) method, achieving an oracle complexity of $O(1/\epsilon^2)$. Notably, we prove that this $O(1/\epsilon^2)$ complexity is unimprovable even under a strongly convex setting. These results demonstrate that the convex NSCO problem shares similar oracle complexities as those without nested compositions, except for strongly convex and outer-non-smooth problems.

In Chapter Four, we investigate the communication complexity of convex risk-averse optimization over a network. The problem generalizes the well-studied risk-neutral finite-sum distributed optimization problem and its importance stems from the need to handle risk in an uncertain environment. For algorithms in the literature, there exists a gap in communication complexities for solving risk-averse and risk-neutral problems. To address this gap, we utilize a trilinear saddle point reformulation to design two distributed algorithms: the distributed risk-averse optimization (DRAO) method and the distributed risk-averse optimization with sliding (DRAO-S) method. The single-loop DRAO method involves solving potentially complex subproblems, while the more sophisticated DRAO-S method requires only simple computations. We establish lower complexity bounds to show their communication complexities to be unimprobvable, and conduct numerical experiments to illustrate the encouraging empirical performance of the DRAO-S method.

In Chapter Five, we utilize the trilinear saddle point approach to develop new complexity results for classic nonlinear function-constrained optimization. We introduce the single-loop Accelerated Constrained Gradient Descent (ACGD) method, which modifies Nesterov's celebrated Accelerated Gradient Descent (AGD) method by incorporating a

linearly-constrained descent step. Lower complexity bounds are provided to establish the tightness of ACGD's complexity bound under a specific optimality regime. To enhance efficiency for large-scale problems, we propose the ACGD with Sliding (ACGD-S) method. ACGD-S replaces computationally demanding constrained descent steps with basic matrix-vector multiplications. ACGD-S shares the same oracle complexity as ACGD and achieves an unimprovable computation complexity measured by the number of matrix-vector multiplications. These advancements offer insights into complexity and provide efficient solutions for nonlinear function-constrained optimization, catering to both general and large-scale scenarios.

CHAPTER 1
INTRODUCTION AND BACKGROUND

Intelligent decision-making has the potential to revolutionize productivity across numerous industries, but unlocking its full benefits requires tackling complex optimization problems on a massive scale. Machine learning and artificial intelligence (AI) have achieved remarkable accomplishments in processing vast amounts of data. However, their broader implementation is hindered by concerns surrounding safety and reliability. In order to ensure the deployment of secure AI systems, it becomes imperative to incorporate safety constraints and risk considerations, thereby introducing large-scale complex optimization problems. On the other hand, operations research (OR) excels in handling uncertainty, risk, and constraints. However, OR's effectiveness in the era of big data also necessitates solving complex optimization problems on a large scale. In this thesis, we propose a trilinear saddle point approach to tackle some of the challenges encountered in solving such large-scale complex optimization problems.

1.1 Background

An optimization problem can be formalized as

$$\min_{x \in X} f(x), \tag{1.1}$$

where X represents the feasibility region and f denotes the objective function. We focus on convex optimization [1], that is, f is a convex function and X is a convex set.

Our goal is to characterize the complexity, i.e., the theoretical computation cost associated with finding a "good" solution for a class of problems, $\mathcal{P}$. Since computing the exact optimal solution x^* to (Equation 1.1) is in general intractable, we focus on finding an

ϵ-optimal solution $\bar{x}$ satisfying $f(\bar{x}) - f(x^*) \leq \epsilon$. Since evaluating the Hessian could be prohibitively expensive, the large-scale setting necessities us to consider first-order (FO) methods which utilize only gradients and function values of f to produce the candidate solution in an iterative fashion.

We call one evaluation of function value and (sub-)gradient at a point x, $(f(x), \nabla f(x))$, one FO oracle evaluation. One natural measure of computation cost is the optimal FO oracle complexity $N^*(\epsilon; \mathcal{P})$, the number of FO oracle evaluations required to find an ϵ-optimal solution. When an algorithm can find an ϵ-optimal solution in $\bar{N}(\epsilon, \mathcal{P})$ oracle evaluations for every problem in $\mathcal{P}$, we call $\bar{N}(\epsilon, \mathcal{P})$ an upper oracle complexity bound because $\bar{N}(\epsilon, \mathcal{P}) \geq N^*(\epsilon, \mathcal{P})$. When there exists a hard problem in $\mathcal{P}$ for which no algorithm can find an ϵ-optimal solution in $\underline{N}(\epsilon, \mathcal{P})$ oracle evaluations, we call $\underline{N}(\epsilon, \mathcal{P})$ a lower oracle complexity bound because $\underline{N}(\epsilon, \mathcal{P}) \leq N^*(\epsilon, \mathcal{P})$. When $\bar{N}(\epsilon, \mathcal{P})$ matches $\underline{N}(\epsilon, \mathcal{P})$ up to some universal constant, $\bar{N}(\epsilon, \mathcal{P})$ characterizes both the sufficient and necessary number of FO oracle evaluations for optimizing problems in $\mathcal{P}$ (up to constants). In that case, we refer to the associated algorithm as optimal because no algorithm could improve on it (up to constants).

The study of oracle complexity traces its roots back to the pioneering work of Nemirovsky and Yudin in the late 1970s [2]. They established fundamental oracle complexities for convex optimization, as outlined below:

- The class of M-Lipschitz continuous objective functions (non-smooth $\mathcal{P}$): if the objective function f satisfies $f(x) - f(y) \leq M \|x - y\| \, \forall x, y \in X$, the optimal oracle complexity $N^*(\epsilon, \mathcal{P})$ is $O(M^2/\epsilon^2)$.

- The class of L-Lipschitz smooth objective functions (smooth $\mathcal{P}$): if $X = \mathbb{R}^n$, and the objective function f is differentiable and satisfies $\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|$ $\forall x, y \in \mathbb{R}^n$, the optimal oracle complexity $N^*(\epsilon, \mathcal{P})$ is $O(\sqrt{L}/\sqrt{\epsilon})$.

Furthermore, if the objective function f is also μ-strongly convex, satisfying

$$f(y) - f(x) - \langle f'(x), y - x \rangle \geq \frac{\mu}{2} \|y - x\|^2 \, \forall x, y \in X$$

where $f'(\mathbf{x}) \in \partial f(\mathbf{x})$ represents an arbitrary subgradient of f at $\mathbf{x}$, the above complexities can be further improved. Moreover, in an important contribution [3], Nesterov proposed the highly acclaimed Accelerated Gradient Descent (AGD) method, which achieves the same $O(1/\sqrt{\epsilon})$ oracle complexity without requiring $X = \mathbb{R}^n$. These complexity results are summarized in Table Table 1.1. Therefore, different types of convex functions have their intrinsic hardness as characterized by their optimal oracle complexities. For example, since it requires less oracle evaluations, we can say the smooth problem is easier than the non-smooth problem from an optimization perspective.

Table 1.1: Classical FO Oracle Complexity Results for Convex Optimization[①]

Problem Class $\mathcal{P}$	Convex	μ-Strongly Convex $\mu > 0$
M-Lipschitz Continuous	$O(M^2/\epsilon^2)$	$O(M^2/\epsilon\mu)$
L-Lipschitz Smooth	$O(\sqrt{L}/\sqrt{\epsilon})$	$O(\sqrt{L/\mu}\log(1/\epsilon))$

Recently, there has been an increasing focus on investigating the oracle complexities associated with solving complex problems that involve different types of functions. In such cases, the complexity of the composed function is typically no better than the worst complexity among the composing functions. For instance, combining a smooth function with a non-smooth Lipschitz-continuous function results in a composed function that is only non-smooth Lipschitz-continuous, rather than smooth. Consequently, applying algorithms specifically designed for solving a single type of function to the composed function can lead to significant computational costs. This inefficiency has prompted researchers to develop new algorithms that are better suited for addressing these composed problems.

In the last two decades, there has been significant research on additive composite optimization problems, driven by applications in compressed sensing, image processing, and

[①]We omitted dependence on the initial distance $\left\|x^0 - x^*\right\|$ for brevity.

machine learning. These problems involve minimizing the sum of a smooth convex loss function $h(x)$ and a non-smooth regularization penalty function $g(x)$ with specific structures,

$$\min_{x \in X} \{f(x) := g(x) + h(x)\}.$$

Optimal first-order methods have been developed under different assumptions about the properties of g [4, 5, 6, 7, 8]. These algorithms can efficiently solve composite optimization problems by capitalizing on the smoothness of h and exploiting the structural properties of g.

A more challenging and general problem involves the nested composition of different types of functions:

$$\min_{x \in X} \{f(x) := h \circ g(x)\}. \tag{1.2}$$

Here, h and g correspond to different types of convex functions. The problem can arise in optimization problems involving safety constraints or risk considerations. Let's explore two examples to illustrate this.

Example 1.1 *Consider function-constrained optimization of the form*

$$\begin{aligned} \min_{x \in X} \quad & g_0(x) \\ s.t. \quad & g_i(x) \leq 0 \quad i \in [m], \end{aligned} \tag{1.3}$$

where $g_0, g_1, ..., g_m$ are smooth convex functions. It is equivalent to

$$\min_{x \in X} \max_{\lambda \in 1 \times \mathbb{R}^m_+} \textstyle\sum_{i=0}^m \lambda_i g_i(x). \tag{1.4}$$

With $g \equiv [g_0, g_1, g_2, ..., g_m]$ and $h(y) := \max_{\lambda \in 1 \times \mathbb{R}^m_+} \sum_{i=0}^m \lambda_i y$, (Equation 1.3) *reduces to the nested composite problem in* (Equation 1.2)*. Notice that g is smooth while h is structured non-smooth.*

Example 1.2 *In risk-averse stochastic programming [9], considering a finite scenario support, the optimization problem can be formulated as:*

$$\min_{x \in X} \rho(g(x, \xi_1), g(x, \xi_2), ..., g(x, \xi_m)), \tag{1.5}$$

where ξ_i denotes the i^{th} scenario. In this case, ρ and $[g(\cdot, \xi_1), ..., g(\cdot, \xi_m)]$ corresponds to h and g in (Equation 1.2) *respectively. Here, h and g can represent different types of convex functions.*

The oracle complexity for solving the nested composite optimization problem in (Equation 1.2) remains unclear. This thesis aims to address this gap by introducing a trilinear saddle point problem as a reformulation framework to tackle the nested composite problem. We leverage this framework to develop optimal algorithms under various settings, such as deterministic, stochastic, and distributed scenarios. Through these algorithmic advancements, we aim to enhance our understanding of the inherent complexity of the nested composite optimization problem. By filling this knowledge gap, our research contributes to the broader understanding of the challenges and intricacies associated with solving nested composite optimization problems.

1.2 The Trilinear Saddle Point Problem

We introduce a novel problem called the trilinear saddle point problem, defined as:

$$\min_{x \in X} \max_{p \in P} \max_{\pi \in \Pi} \{\mathcal{L}(x; p, \pi) := \sum_i p_i \left(\langle A_i x, \pi_i \rangle - g_i^*(\pi_i)\right) - h^*(p) + u(x)\}. \tag{1.6}$$

Here, $u(x)$ represents a simple convex function, A_i denotes a linear operator, and g_i^* and $h^*(x)$ are convex functions. We assume that X, P, and Π are convex sets. The name "trilinear" refers to the term $\phi(x; p, \pi) = \sum_i p_i \langle A_i x, \pi_i \rangle$ in (Equation 1.6), which is linear with respect to the third variable when the other two variables are fixed, i.e., $\phi(x; \bar{p}, \bar{\pi})$

is linear with respect to x. However, the trilinear saddle point problem is generally not tractable due to the joint maximization of p and π in the bilinear form $\sum_i p_i \pi_i$. To address this, we make the following non-negativity assumption on P:

Assumption 1 *P in* (Equation 1.6) *satisfies* $P \subset \mathbb{R}^m_+$.

This assumption ensures that the maximization of π_i is independent of the choice of p, allowing for efficient joint maximization. Furthermore, we can demonstrate that $f(x) := \max_{p \in P} \max_{\pi \in \Pi} \Lambda(x; p, \pi)$ is a convex function with respect to x.

Let's examine how Example 1.1 and Example 1.2 can be reformulated as trilinear saddle point problems:

- Function Constraint Example (1.1): By utilizing the bi-conjugation identity $g_i(x) \equiv \max_{\pi_i} \langle \pi_i, x \rangle - g_i^*(\pi_i)$, where g_i^* represents the Fenchel conjugate of g_i [10], the problem in (Equation 1.4) can be reformulated into the trilinear saddle point form in (Equation 1.6) by introducing $\pi = [\pi_0, \ldots, \pi_m]$ and $p = [1, \lambda]$. Notably, Assumption 1 is satisfied since the Lagrange multiplier λ is always non-negative.

- Risk-Averse Example (1.2): By applying the bi-conjugate identity to reformulate both ρ and the scenario costs $g(x, \xi_i)$, the problem in (Equation 1.5) can be transformed into the trilinear saddle point form in (Equation 1.6). Additionally, if the risk measure is monotone [9], Assumption 1 is automatically fulfilled.

In summary, both examples can be effectively reformulated as trilinear saddle point problems (Equation 1.6). This demonstrates the versatility and applicability of the trilinear saddle point framework in capturing and solving nested composite optimization problems arising from various contexts.

To facilitate algorithm design, we utilize a gap function Q to measure the quality of a feasible solution $z = (x, p, \pi)$ with respect to a feasible reference point $\hat{z} = (\hat{x}; \hat{p}, \hat{\pi})$. The

gap function Q is defined as:

$$Q(z;\hat{z}) := \Lambda(x;\hat{p},\hat{\pi}) - \Lambda(\hat{x};p,\pi). \tag{1.7}$$

Note that z is a saddle point of problem (Equation 1.6) if and only if $Q(z;\hat{z}) \leq 0$ for all feasible reference points $\hat{z} \in X \times P \times \Pi$. By analyzing and minimizing the gap function, we can devise effective algorithms to find desirable solutions to the trilinear saddle point problem.

Towards that end, one particularly useful property of the Q function is its decomposition into three sub-gap functions, which allows us to focus on individual components related to x, p, and π. This decomposition is given by:

$$Q(\bar{z};\hat{z}) = Q_x(\bar{z};\hat{z}) + Q_p(\bar{z};\hat{z}) + Q_\pi(\bar{z};\hat{z})$$

with

$$\begin{aligned}
Q_\pi(\bar{z};\hat{z}) &:= \mathcal{L}(\bar{x};\hat{p},\hat{\pi}) - \mathcal{L}(\bar{x};\hat{p},\bar{\pi}) = \textstyle\sum_{i=1}^m \hat{p}_i \left[\langle A_i\bar{x}, \hat{\pi}_i - \bar{\pi}_i\rangle - g_i^*(\hat{\pi}_i) + g_i^*(\bar{\pi}_i)\right]. \\
Q_p(\bar{z};\hat{z}) &:= \mathcal{L}(\bar{x};\hat{p},\bar{\pi}) - \mathcal{L}(\bar{x};\bar{p},\bar{\pi}) \\
&\qquad = \textstyle\sum_{i=1}^m (\hat{p}_i - \bar{p}_i)[\langle A_i\bar{x}, \bar{\pi}_i\rangle - g_i^*(\bar{\pi}_i)] - (h^*(\hat{p}) - h^*(\bar{p})). \\
Q_x(\bar{z};\hat{z}) &:= \mathcal{L}(\bar{x};\bar{p},\bar{\pi}) - \mathcal{L}(\hat{x};\bar{p},\bar{\pi}) = \langle \textstyle\sum_{i=1}^m \bar{p}_i A_i^\top \bar{\pi}_i, \bar{x} - \hat{x}\rangle + u(\bar{x}) - u(\hat{x}).
\end{aligned} \tag{1.8}$$

In our approach, we typically begin by designing update steps that reduce the corresponding sub-gap function while keeping the other two variables fixed. We then combine these steps in an appropriate manner to derive an iterative algorithm that effectively minimizes the overall Q function. This decomposition allows for a systematic approach in updating each variable, leading to an efficient optimization algorithm.

CHAPTER 2

DISTRIBUTIONALLY ROBUST TWO-STAGE LINEAR PROGRAM WITH A FINITE SCENARIO SUPPORT

2.1 Background and Our Contribution

Two-stage stochastic programming (SP) problems are the most widely used stochastic optimization models in practice [9]. In this paper, we consider a distributionally robust two-stage stochastic convex optimization problem with a finite set of scenarios $\{\xi_i\}_{i=1}^K$,

$$\min_{x\in X}\left\{f(x) := f_0(x) + \max_{p\in P}\sum_{k=1}^K p_k g(x,\xi_k) - \phi^*(p)\right\}, \tag{2.1}$$

where $X \subset \mathbb{R}^n$ is a convex and compact feasible region for the first-stage decision variable x, and $P \subset \mathbb{R}^K$ is a convex and compact ambiguity set for the scenario probability vector $p \in \mathbb{R}^K$. We assume that the first-stage cost function $f_0(\cdot)$ and the second-stage cost functions $g(\cdot,\xi_k)$ are proper closed convex (p.c.c.) and Lipschitz continuous, and that ϕ^*[①] is a simple p.c.c. function of p. The goal is to minimize the expected cost with respect to the worst probability vector in P.

Such a problem arises naturally under the following situations.

- **Data driven SP with finite scenario support.** We want to minimize the expected cost with respect to the true distribution p^*. However, p^* is usually unknown, and only partial information about it can be obtained from either historical observations or simulation. In this case, one can construct an $1-\alpha$ confidence ambiguity set P_α, i.e., $p^* \in P_\alpha$ with a probability of at least $1-\alpha$, and solve for the DRO problem

[①]Notice that ϕ^* is usually identically zero in DRO problems, however we include it to handle some non-coherent risk measures for risk-averse stochastic programming problem.

associated with P_α. The true cost for DRO solution $\hat{x}$ would be less than the DRO cost with a probability of at least $1-\alpha$. There exist an expansive literature on such confidence ambiguity sets, including the Phi-divergence ball [11, 12], the ζ-distance ball [13], and the hypothesis testing set [14].

- **Data driven SP with continuous scenario support.** An important metric-based ambiguity set is the Kantorovich ball. This is because when $g(x,\xi)$ is Lipschitz continuous in ξ for all x, the expected cost $\mathbb{E}_p[g(x,\xi)]$ is Lipschitz continuous in p with respect to the Kantorovich distance. In two-stage stochastic programming, the radius δ for the Kantorovich P_α ball [15, 16], the sufficient conditions for the Lipschitz-continuity of $g(x,\xi)$ and the convergence of DRO solutions to true solutions [17] are well studied. However, computing the DRO solution remains challenging because it involves finding the maximal in the infinite dimensional space of distributions. One approach to address such a difficulty is to use a duality argument to simplify the problem to

$$\min_{x\in X,\lambda\geq 0} f_0(x) + \lambda\delta + \frac{1}{N}\sum_{i=1}^{N} \max_{\xi\in\Xi} g(x,\xi) - \lambda d(\xi_i,\xi),$$

where λ is the Lagrange multiplier for the total transportation cost constraint and d is the distance function. To solve the simplified problem, [18] reformulates it to a large deterministic convex problem, [19] suggests using the mirror-prox algorithm and [15] suggests using the Benders decomposition algorithm. The successes of [19, 18] hinge on the concavity of $g(x,\xi)$ with respect to ξ, while [15] carries some other structural assumptions on $g(x,\xi)$ and Ξ. These requirements can be restrictive. For example, the concavity of $g(x,\xi)$ is not satisfied even for a two-stage linear stochastic program with right-hand side uncertainty. A more general approach is to use a discrete grid of scenarios Ξ_K to approximate the whole scenario space Ξ and solve the DRO problem restricted to Ξ_K [16, 13]. The approximation error can be

bounded by the Hausdorff distance between Ξ_K and Ξ, so a fine grid, i.e., a large number of scenarios, is necessary for a moderately accurate solution.

- **Risk-averse SP with finite scenario support:** In finance, the preference for less risk can be formulated using a risk measure ϕ, so the goal is to find a decision x with minimal ϕ. For example, in portfolio selection [20], given a finite number of scenarios about possible returns $\{g(x, \xi_k)\}$, we want to select a portfolio x with minimum $\phi(g(x, \xi_1), ..., g(x, \xi_K))$. If such a risk measure is p.c.c. and monotone, say the piecewise linear dis-utility function, then we can use bi-conjugation [10] to rewrite the problem as $\min_{x \in X} \max_{p \in \mathbb{R}^K_+} \sum_{k=1}^K p_k g_k(x) - \phi^*(p)$. In addition, if ϕ is a coherent risk measure [21], for example the average value-at-risk (AVaR), then $\phi^* \equiv 0$ and p's domain must be a subset of the probability simplex.

Now returning to (Equation 2.1), we can simply denote the $g(x, \xi_k)$ by $g_k(x)$. In many cases, the function $g(x, \xi_k)$ may involve a linear transformation T_k on x, for example, the technology matrix in stochastic programming. Then it is often desirable to process such a linear transformation differently from other nonlinear components of $g(x, \xi_k)$ in the design of algorithms. Therefore, we rewrite $g(x, \xi_k)$ as $g_k(T_k x)$ to arrive at the following equivalent reformulation of (Equation 2.1),

$$\min_{x \in X} \left\{ f(x) := f_0(x) + \max_{p \in P} \sum_{k=1}^K p_k g_k(T_k x) - \phi^*(p) \right\}. \tag{2.2}$$

Apparently, if one does not need to process T_k separately or such a linear transformation does not exist, we can simply set $T_k = I$ in (Equation 2.2).

Problem (Equation 2.2) is a convex-concave saddle point problem and can be solved by the mirror descent method [24] or the bundle level method [23] directly. However g_k is often non-smooth, for example, the minimum objective of a linear program. So direct applications of these methods would lead to an $\mathcal{O}(1/\epsilon^2)$ iteration complexity bound, which is independent of the number of scenarios K. In each iteration, the function values and

Table 2.1: Theoretical Performance Comparison for Major Deterministic Algorithms

Algorithm	Iter Complexity	Most Expensive Computation[1]
Benders Decomposition [13, 22]	$\mathcal{O}(1/\epsilon^n)$	K separable $m \times n$ LPs in parallel
Bundle Level [23]	$\mathcal{O}(1/\epsilon^2)$	K separable $m \times n$ LPs in parallel
Mirror Descent [24]	$\mathcal{O}(1/\epsilon^2)$	K separable $m \times n$ LPs in parallel
Constraint PDHG [25]	$\mathcal{O}(1/\epsilon)$	One large-scale and non-separable QP
Separabale PDHG [16]	$\mathcal{O}(K/\epsilon)$	K separable $m \times n$ QPs in parallel
Euclidean SD & SSL	$\mathcal{O}(\sqrt{K}/\epsilon)$	K separable $m \times n$ QPs in parallel
Entropy SD & SSL	$\mathcal{O}(\sqrt{\log K}/\epsilon)$	K separable $m \times n$ QPs in parallel

[1] Based on solving distributionally robust two-stage LP.
[2] The complexity of Benders decomposition (or Kelley's cutting plane method) was established in [22] with n being the dimension of the problem.

sub-gradients for $\{g_k\}$ can be computed in parallel.

To improve the iteration complexity bound, Liu et al. put the second-stage cost functions in the constraint to obtain a composite bilinear saddle point problem in [25],

$$\min_{x\in X, v_k \geq g_k(T_k x)} \max_{p\in P} f_0(x) + \textstyle\sum_{k=1}^{K} p_k v_k - \phi^*(p). \tag{2.3}$$

They applied the primal-dual hybrid gradient (PDHG) algorithm in [26] to obtain an $\mathcal{O}(1/\epsilon)$ iteration complexity bound. However, this algorithm may not be practical because each iteration involves projecting (x, v) onto a jointly constrained set, $\{v_k \geq g_k(T_k x), \forall k\}$. More recently, Chen et al. [16] address the non-separability issue by introducing a copy of x for each scenario, $\{x_k\}$, and uses Lagrange multipliers $\{\lambda_k\}$ to enforce their consensus to arrive at the following reformulation:

$$\min_{x_0, x_k \in X, v_k \geq g_k(T_k x_k)} \max_{p\in P} \max_{\lambda_k \in R^n} \langle v, p\rangle + f_0(x_0) + \textstyle\sum_{k=1}^{K} \langle x_0 - x_k, \lambda_k\rangle - \phi^*(p). \tag{2.4}$$

The objective is jointly concave (linear) with respect to (p, λ), so (Equation 2.4) is again a bilinear saddle point problem to which the PDHG algorithm can be applied. Moreover, the (x_k, v_k) projections can be performed in parallel if needed. However such an ap-

proach still has two major limitations. Firstly, the combined (p, λ) dual block prevents us from exploiting the special geometry of P, a subset of the probability simplex, to improve the iteration complexity bound's dependence on K. More specifically, since the Euclidean Bregman distance is used in [16], the radii of both the primal feasibility region for $\{x_0, (x_1, v_1)...(x_K, v_K)\}$ and the dual feasibility region for $\{(p_1, p_2...p_K); \lambda_1; \lambda_2; ...\lambda_K\}$ are $\mathcal{O}(\sqrt{K})$. So it follows from [26] that the iteration complexity bound is $\mathcal{O}(K/\epsilon)$. Secondly, the projection onto a non-smooth function constrained set $\{v_k \geq g_k(T_k x_k)\}$ in each iteration could be computationally expensive.

An interesting research problem is whether there exists an $\mathcal{O}(1/\epsilon)$ algorithm which can handle both the large number of scenarios and the non-smooth second-stage cost $g_k(T_k x)$ effectively. Towards this end, we use bi-conjugation [10] to reformulate the non-smooth $g_k(T_k x)$ as $\max_{\pi_k \in \Pi(k)} \langle \pi_k, T_k x \rangle - g_k^*(\pi_k)$ to arrive at a trilinear saddle point problem,

$$\min_{x \in X} f_0(x) + \underbrace{\max_{p \in P} \sum_{k=1}^{K} \max_{\pi_k \in \Pi(k)} p_k(\langle T_k x, \pi_k \rangle - g_k^*(\pi_k)) - \phi^*(p)}_{F(x)}, \tag{2.5}$$

where $\Pi(k)$ is the domain of the conjugate function $g_k^*(\pi_k)$, $\Pi(k) := \{\pi_k \mid g_k^*(\pi_k) < \infty\}$[②]. As compared to (Equation 2.4), (Equation 2.5) is no longer jointly concave in p and $\{\pi_k\}$, and the projection in $(p, \{\pi_k\})$ is not parallelizable. So the simple reduction to a convex-concave saddle-point problem is not possible. However, because p is non-negative, the non-concave maximization in (Equation 2.5) can be evaluated efficiently in a sequential manner: given a $x \in X$, first maximize $\{\pi_k\}$ in parallel and then maximize p.

In this paper, we take advantage of such a sequential structure by treating p and $\{\pi_k\}$ as separate dual blocks and develop two new algorithms: a simple sequential dual (SD) method and a complicated but more efficient sequential smoothing level (SSL) method. The SD method extends the popular primal-dual method; it has a novel momentum step

[②]Notice that if g_k is a p.c.c. function, then g_k^* must also be p.c.c. , so $\Pi(k)$ is closed and convex. Moreover, if g_k is Lipschitz continuous, then $\Pi(k)$ must be bounded, i.e., $\Pi(k)$ is compact.

and an additional p-projection step. The SSL algorithm extends Nesterov's smoothing scheme to build a two-layer smooth approximation of (Equation 2.5) and then applies the accelerated prox-level method in [27] to an adaptively smoothed approximation of f. The SSL algorithm is parameter-free. It is worth noting that bundle-level type methods are classical methods for solving two-stage stochastic programming problems, but they have not been studied for solving distributionally robust problems before.

In addition, since P is now a standalone block, we have more flexibility to exploit its favorable geometry to obtain either a better iteration complexity or cheaper computations in each iteration. More specifically, if P is simple, we can use entropy p projection to reduce the iteration complexity bound to $\mathcal{O}(\sqrt{\log K}/\epsilon)$. If P is the computationally challenging Kantorovich ball, we can substitute the expensive p projection with a cheaper joint probability matrix projection at the price of increasing the iteration complexity to $\mathcal{O}(\sqrt{K}/\epsilon)$. Due to the separation of the p-block from the other blocks, only stepsize modifications are needed for our SD and SSL methods. To the best of our knowledge, all these complexity results appear to be new for solving trilinear saddle point problems given in the form of (Equation 2.5).

The paper is organized as follows. Section 2 proposes the simple sequential dual (SD) algorithm, and Section 3 develops the parameter-free sequential smoothing level (SSL) method. Section 4 introduces the specialized modifications of the SD and SSL algorithms for the challenging Kantorovich ball. Finally, encouraging numerical results are presented in Section 5 and concluding remarks are made in section 6.

2.1.1 Notations and Assumptions

Throughout the paper, we use x^* denote an arbitrary optimal solution to (Equation 2.2). For any convex function f defined on $\mathcal{X}$, we use $\partial f(x)$ to denote the set of all sub-gradients and use $f'(x)$ to denote an arbitrary element in $\partial f(x)$. If the set $\mathcal{X}$ is associated with some norm $\|\cdot\|_{\mathcal{X}}$, we use $\|\cdot\|_{\mathcal{X}^*}$ to denote its dual norm. Moreover, we call f L-smooth if it

satisfies $f(x_1) - f(x_2) - \langle x_1 - x_2, f'(x_2)\rangle \leq \frac{L}{2}\|x_1 - x_2\|_{\mathcal{X}}^2$ for all $x_1, x_2 \in \mathcal{X}$, and we call f μ-strongly convex if it satisfies $f(x_1) - f(x_2) - \langle x_1 - x_2, f'(x_2)\rangle \geq \frac{\mu}{2}\|x_1 - x_2\|_{\mathcal{X}}^2$ for all $x_1, x_2 \in \mathcal{X}$.

To take advantage of the geometry of P, we need the Bregman distance function. Given a closed and convex set Y[3], let $F : Y \to \mathbb{R}$ be differentiable and convex, and 1-strongly convex over $\text{dom}(\partial F) := \{y \in Y : \partial F(y) \neq \emptyset\}$ with respect to some $\|\cdot\|_F$, the Bregman distance function $d_F : \text{dom}(\partial F) \times Y \to \mathbb{R}$ is defined as

$$d_F(y_1, y_2) = F(y_2) - F(y_1) - \langle F'(y_1), y_1 - y_2\rangle.$$

In the following analysis, we will consider a general Bregman distance function $W(\cdot, \cdot)$ for P. Distance functions of practical interests consist of the Euclidean $W(p_1, p_2) := \|p_1 - p_2\|^2$ and the entropy $W(p_1, p_2) := \sum_{i=1}^K p_{1,i} \log(p_{2,i}/p_{1,i})$, which are 1-strongly convex with respect to $\|\cdot\|_2$ and $\|\cdot\|_1$ respectively. For X and $\Pi(k)$, we will use the Euclidean distance functions $V(x_1, x_2) := \|x_1 - x_2\|_2^2/2$ and $U(\pi_{1,k}, \pi_{2,k}) := \|\pi_{1,k} - \pi_{2,k}\|_2^2/2$ for simplicity.

To facilitate analyzing how our algorithms scale with K, we need some scenario independent radii and operator norms. Let $\Omega_X^2 := \max_{x \in X} V(x_0, x)$, $\Omega_P^2 := \max_{p \in P} W(p_0, p)$ for some initial points x_0 and p_0. Ω_X is independent of K, but Ω_P can depend on K. More specifically, if p_0 is the empirical distribution and P is the whole probability simplex, then Ω_P is $\mathcal{O}(1)$ for Euclidean W and $\mathcal{O}(\sqrt{\log K})$ for entropy W.

For the multi-block $\{\pi_k\}$, we use boldface letters to denote the concatenation: $\boldsymbol{\pi} := [\pi_1, \pi_2, \ldots, \pi_K]$, $\boldsymbol{T} := [T_1; T_2; \ldots; T_K]$, $\boldsymbol{g}^*(\boldsymbol{\pi}) := [g_1^*(\pi_1), \ldots, g_K^*(\pi_K)]$, and $\boldsymbol{\Pi} := \Pi(1) \times \Pi(2) \cdots \times \Pi(K)$. We use the following shorthand notations for multi-scenario functions: $p\boldsymbol{T}\pi := \sum_{k=1}^K p_k T_k \pi_k$, $\langle x, \boldsymbol{\pi}\rangle_{\boldsymbol{T}} := [\langle T_1 x, \pi_1\rangle, \langle T_2 x, \pi_2\rangle, ..., \langle T_K x, \pi_K\rangle]$ and $\boldsymbol{U}(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) := [U(\pi_{1,1}, \pi_{2,1}), U(\pi_{1,2}, \pi_{2,2}), ..., U(\pi_{1,K}, \pi_{2,K})]$ and their k-th components:

[3] In general the Bregman distance function can be defined over any set, not necessary a closed and convex set. For the general definition, please refer to [28].

$\langle x, \boldsymbol{\pi} \rangle_{T_k} := \langle T_k x, \pi_k \rangle$ and $U_k(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2) := U(\pi_{1,k}, \pi_{2,k})$. Let the multi-block (2,q)-norm be $\|\boldsymbol{\pi}\|_{2,q} := \|[\|\pi_1\|_2, \|\pi_2\|_2, ..., \|\pi_K\|_2]\|_q$, then the scenario independent radius and operator norm for $\boldsymbol{\pi}$ and $\boldsymbol{T}$ are defined as :

$$\begin{aligned} \Omega_{\boldsymbol{\Pi}}^2 &:= \max_{k \in K} \max_{\boldsymbol{\pi} \in \boldsymbol{\Pi}} U_k(\boldsymbol{\pi}_0, \boldsymbol{\pi}) \text{ for some initial } \boldsymbol{\pi}_0 \in \boldsymbol{\Pi}, \\ M_{\boldsymbol{T}} &:= \max_{k \in [K]} \|T_k\|_{2,2} \text{ and } M_{\boldsymbol{\Pi}} := \max_{\boldsymbol{\pi} \in \boldsymbol{\Pi}} \|\boldsymbol{\pi}\|_{2,\infty}. \end{aligned} \tag{2.6}$$

Because $g_k(\cdot)$ is p.c.c. and Lipschitz-continuous, every $\Pi(k)$ is a convex closed and bounded, so $M_{\boldsymbol{\Pi}} < \infty$.

2.2 Sequential Dual Algorithm

In this section, we consider (Equation 2.5) from a saddle point perspective:

$$\min_{x \in X} \max_{(p, \boldsymbol{\pi}) \in P \times \boldsymbol{\Pi}} \{\mathcal{L}(x, p, \boldsymbol{\pi}) := f_0(x) + \textstyle\sum_{k=1}^K p_k(\langle T_k x, \pi_k \rangle - g_k^*(\pi_k)) - \phi^*(p)\}. \tag{2.7}$$

One challenge is the non-concavity of $\mathcal{L}$ with respect to $(p, \boldsymbol{\pi})$, so existing saddle point algorithms cannot be directly applied. However, upon a closer inspection, we find the ingredients required to design a primal-dual saddle point algorithm [28] still applicable due to the non-negativity of the p-block. More specifically, in Subsection 2.1, we show a duality relationship between $\mathcal{L}$ in (Equation 2.7) and f in (Equation 2.2), and a conversion from the primal-dual gap (see Definition 2.1) to the functional optimality gap. Then in Subsection 2.2, we present a decomposition of the primal-dual gap into individual optimality gaps of the $x,\ p$ and $\boldsymbol{\pi}$ blocks. These individual optimality gaps are composite linear, i.e., of the form $\langle \cdot, y \rangle + h(\cdot)$, where $h(\cdot)$ are some simple convex functions. So, as is standard in first-order methods [28], these quantities can be gradually decreased by iterative proximal updates. We introduce some novel momentum terms in these proximal updates, which then leads to the SD method.

2.2.1 Duality and Primal-Dual Function

The following duality relationship between f and $\mathcal{L}$ is straightforward because it boils down to switching the order of a non-negative weighted summation and a maximization.

Proposition 2.1 *Let f and $\mathcal{L}$ be defined in* (Equation 2.2) *and* (Equation 2.7)*, then the following statements hold for all $x \in X$.*

a) Weak Duality: $f(x) \geq \mathcal{L}(x, p, \boldsymbol{\pi})$ for all $p \in P, \boldsymbol{\pi} \in \boldsymbol{\Pi}$.

b) Strong Duality: $f(x) = \mathcal{L}(x, \bar{p}, \bar{\boldsymbol{\pi}})$ for some $\bar{p} \in P, \bar{\boldsymbol{\pi}} \in \boldsymbol{\Pi}$.

Proof: We consider the strong duality first. Pick $\bar{\pi}_k \in \partial g_k(T_k x)$ such that $\langle T_k x, \bar{\pi}_k \rangle - g_k^*(\bar{\pi}_k) = g_k(T_k x)$ and $\bar{p} \in \arg\max_{p \in P} \sum_{k=1}^K p_k g_k(T_k x) - \phi^*(p)$, then it is easy to verify $\mathcal{L}(x, \bar{p}, \bar{\boldsymbol{\pi}}) = f(x)$.

Next, we show the weak duality. Notice that

$$\begin{aligned} f(x) =& f_0(x) + \max_{p \in P} \textstyle\sum_{k=1}^K p_k [\max_{\pi_k \in \Pi(k)} \langle T_k x, \pi_k \rangle - g_k^*(\pi_k)] - \phi^*(p) \\ \overset{(a)}{=}& f_0(x) + \max_{p \in P} \max_{\pi_k \in \Pi(k)} \textstyle\sum_{k=1}^K p_k [\langle T_k x, \pi_k \rangle - g_k^*(\pi_k)] - \phi^*(p) \\ =& \max_{(p, \boldsymbol{\pi}) \in P \times \boldsymbol{\Pi}} \mathcal{L}(x, p, \boldsymbol{\pi}), \end{aligned}$$

where (a) follows from the non-negativity of P. So $f(x) = \max_{(p, \boldsymbol{\pi}) \in P \times \boldsymbol{\Pi}} \mathcal{L}(x, p, \boldsymbol{\pi}) \geq \mathcal{L}(x, p, \boldsymbol{\pi})$ for any feasible p and $\boldsymbol{\pi}$. □

We define a primal-dual gap function [28] for analyzing the saddle point problem (Equation 2.7) as follows.

Definition 2.1 *Let $z := (x, p, \boldsymbol{\pi}) \in Z := X \times P \times \boldsymbol{\Pi}$ and $u := (u_x, u_p, u_{\boldsymbol{\pi}}) \in Z$. Then the primal-dual gap function is given by*

$$Q(z, u) := \mathcal{L}(x, u_p, u_{\boldsymbol{\pi}}) - \mathcal{L}(u_x, p, \boldsymbol{\pi}).$$

$Q(z,u)$ measures the saddle point optimality of z in comparison to some u; if z is a saddle point, then $Q(z,u) \leq 0$ for all feasible u. In our analysis, we use $Q(z,u)$ as an upper bound for the functional optimality gap, $f(x) - f(x^*)$. With a carefully chosen u, we can show $\sum_{t=1}^{N} Q(z^t,u)$ providing an upper bound for the optimality gap of an ergodic average solution $\bar{x}^t$, which is illustrated in the following proposition.

Proposition 2.2 *Let $u := (x^*, u_p, u_{\boldsymbol{\pi}})$ and a feasible sequence $\{z^t := (x^t, p^t, \boldsymbol{\pi}_t)\}$ be given. If $\max_{(u_p,u_{\boldsymbol{\pi}})\in P\times\boldsymbol{\Pi}} \sum_{t=1}^{N} Q(z^t,u) \leq B$ for some finite B. Then the ergodic solution $\bar{x}^t := \sum_{t=1}^{N} x^t/N$ satisfies*

$$f(\bar{x}^t) - f(x^*) \leq \tfrac{B}{N}.$$

Proof: From the strong duality result in Proposition 2.1, we have $\mathcal{L}(\bar{x}^t, \bar{p}_N, \bar{\boldsymbol{\pi}}_N) = f(\bar{x}^t)$ for some $\bar{p}_N$ and $\bar{\boldsymbol{\pi}}_N$. But $\mathcal{L}(x, \bar{p}_N, \bar{\boldsymbol{\pi}}_N)$ is convex with respect to x, so it follows from the Jensen's inequality that $Nf(\bar{x}^t) = N\mathcal{L}(\bar{x}^t, \bar{p}_N, \bar{\boldsymbol{\pi}}_N) \leq \sum_{t=1}^{N} \mathcal{L}(x^t, \bar{p}_N, \bar{\boldsymbol{\pi}}_N)$. Moreover, the weak duality in Proposition 2.1 implies that $f(x^*) \geq \mathcal{L}(x^*, p^t, \boldsymbol{\pi}_t)\ \forall t$, so $Nf(x^*) \geq \sum_{t=1}^{N} \mathcal{L}(x^*, p^t, \boldsymbol{\pi}_t)$. Therefore we have

$$N(f(\bar{x}^t) - f(x^*)) \leq \sum_{t=1}^{N} \mathcal{L}(x^t, \bar{p}_N, \bar{\boldsymbol{\pi}}_N) - \mathcal{L}(x^*, p^t, \boldsymbol{\pi}_t) = \sum_{t=1}^{N} Q(z^t, (x^*, \bar{p}_N, \bar{\boldsymbol{\pi}}_N)) \leq B.$$

Dividing both sides by N, we get the desired result. □

2.2.2 The Sequential Dual Method

The development of the sequential dual method (see Algorithm Algorithm 1) is inspired by the following decomposition of $Q(z;u)$: $Q(z;u) \equiv Q_x(z;u) + Q_p(z;u) + Q_\pi(z;u)$ where

$$\begin{aligned}
Q_\pi(z;u) &:= \mathcal{L}(x, u_p, u_{\boldsymbol{\pi}}) - \mathcal{L}(x, u_p, \boldsymbol{\pi}) = \langle u_p, \langle x, u_{\boldsymbol{\pi}} \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(u_{\boldsymbol{\pi}}) \rangle \\
&\qquad\qquad \underbrace{-\langle u_p, \langle x, \boldsymbol{\pi} \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}) \rangle}_{\boldsymbol{\pi} \text{ gap}}. \\
Q_p(z;u) &:= \mathcal{L}(x, u_p, \boldsymbol{\pi}) - \mathcal{L}(x, p, \boldsymbol{\pi}) = \langle u_p, \langle x, \boldsymbol{\pi} \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}) \rangle - \phi^*(u_p) \\
&\qquad\qquad \underbrace{-\langle p, \langle x, \boldsymbol{\pi} \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}) \rangle + \phi^*(p)}_{p \text{ gap}}. \\
Q_x(z;u) &:= \mathcal{L}(x, p, \boldsymbol{\pi}) - \mathcal{L}(u_x, p, \boldsymbol{\pi}) = \underbrace{f_0(x) + \langle x, p\boldsymbol{T}\boldsymbol{\pi} \rangle}_{x \text{ gap}} - (f_0(u_x) + \langle u_x, p\boldsymbol{T}\boldsymbol{\pi} \rangle).
\end{aligned} \tag{2.8}$$

Observe that inside each $Q_{(\cdot)}$ function, the under-braced terms associated with the $(\cdot)$ argument are of the form $\langle y, \cdot \rangle + h(\cdot)$ for some simple convex $h(\cdot)$, e.g., $-\langle u_p, \langle x, \cdot \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\cdot) \rangle$ in the $\boldsymbol{\pi}$ gap. So we can use proximal updates to decrease them iteratively. However, at least two of $(x, p, \boldsymbol{\pi})$ appear together in every decomposed gap term. Thus we need to use some *guesses* for the other blocks if they have not been evaluated in the sequential update scheme, and care must be taken in designing those *guesses* to ensure the cancellation of the consequent prediction errors. More specifically, given a sequence $\{z_i \equiv (x_i, p_i, \boldsymbol{\pi}_i)\}_{i=0}^t$, we propose the following sequential proximal update for the $\boldsymbol{\pi},\ p$ and x blocks (in that order) to obtain a possibly smaller $Q_x(z_{t+1};u)$, $Q_p(z_{t+1};u)$ and $Q_\pi(z_{t+1};u)$.

1. **$\boldsymbol{\pi}$ block:** we need to decrease the value of $-\langle u_p, \langle x_{t+1}, \boldsymbol{\pi} \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}) \rangle$ in (Equation 2.8). But since u_p is non-negative, we might as well reduce every component of the vector $-\langle x_{t+1}, \boldsymbol{\pi} \rangle_{\boldsymbol{T}} + \boldsymbol{g}^*(\boldsymbol{\pi})$ separately. Moreover, x_{t+1} is currently unknown, so we use the *guess* $x_t + (x_t - x_{t-1})$ to arrive at the following $\boldsymbol{\pi}-$proximal update step,

i.e., Line 4 in Algorithm Algorithm 1,

$$\pi_{k,t+1} = \arg\min_{\pi_k \in \Pi(k)} -\langle 2x_t - x_{t-1}, \boldsymbol{\pi} \rangle_{T_k} + g_k^*(\boldsymbol{\pi}) + \sigma U_k(\boldsymbol{\pi}_t, \boldsymbol{\pi}).$$

2. p **block:** we wish to decrease the value of $-\langle p, \langle x_{t+1}, \boldsymbol{\pi}_{t+1} \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}_{t+1}) \rangle + \phi^*(p)$ in (Equation 2.8). Again, the information about $\langle x_{t+1}, \boldsymbol{\pi}_{t+1} \rangle_{\boldsymbol{T}}$ is unavailable, so we use the *guess* $\langle x_t, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}} + (\langle x_t, \boldsymbol{\pi}_{t+1} \rangle_{\boldsymbol{T}} - \langle x_{t-1}, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}})$ to obtain the following $p-$proximal update step, i.e., Line 5 in Algorithm Algorithm 1,

$$\begin{aligned} p_{t+1} = \arg\min_{p \in P} & - \langle p, \langle x_t, \boldsymbol{\pi}_{t+1} \rangle_{\boldsymbol{T}} + \langle x_t - x_{t-1}, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}_{t+1}) \rangle \\ & + \phi^*(p) + \tau W(p_t, p). \end{aligned}$$

3. x **block:** We intend to decrease the value of $f_0(x) + \langle x, p_{t+1} \boldsymbol{T} \boldsymbol{\pi}_{t+1} \rangle$. But since we already know $(p_{t+1}, \boldsymbol{\pi}_{t+1})$ from the previous two updates, the x-proximal update step, Line 6 in Algorithm Algorithm 1, is simply

$$x_{t+1} = \arg\min_{x \in X} \langle x, p_{t+1} \boldsymbol{T} \boldsymbol{\pi}_{t+1} \rangle + f_0(x) + \eta V(x_t, x).$$

The algorithm is named sequential dual method because both the $\boldsymbol{\pi}$ and p blocks can be viewed as dual blocks and they need to be updated sequentially before the primal x block can be updated.

Our goal in the remaining part of this section is to analyze the convergence properties of the SD method. To highlight the dependence of the iteration complexity bound on K, we need to relate the dual norm $\|\boldsymbol{\pi}\|_{2,W^*}$, which possibly depends on K, to $\|\boldsymbol{\pi}\|_{2,\infty}$, which is independent of K.

Definition 2.2 *Let $\|\cdot\|_W$ be the norm associated with P, we call any $C_p \geq 0$ a norm adjustment constant for the ambiguity set $\boldsymbol{P}$ if it satisfies $C_p \|\boldsymbol{\pi}\|_{2,\infty} \geq \|\boldsymbol{\pi}\|_{2,W^*}$ for all*

Algorithm 1 Sequential Dual Algorithm

Input: $(x_0, p_0, \boldsymbol{\pi}_0) \in X \times P \times \boldsymbol{\Pi}$ and stepsizes $\sigma, \tau, \eta > 0$
Output: $(\bar{x}^N, \bar{p}^N, \bar{\boldsymbol{\pi}}^N)$

1: **Initialization** set $x_{-1} = x_0$.
2: **for** $t = 1, 2, 3...N$ **do**
3: set $\widetilde{x}^t = 2 * x^{t-1} - x_{t-2}$.
4: set $\pi_{k,t} = \arg\max_{\pi_k \in \Pi(k)}(\langle T_k \widetilde{x}^t, \pi_k \rangle - g_k^*(\pi_k, \xi_k)) - \sigma U_k(\pi_{k,t-1}, \pi_k), \quad \forall k \in [K]$.
5: set $\widetilde{f}_{t,k} = \langle x^{t-1}, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}} + \langle (x^{t-1} - x_{t-2}), \boldsymbol{\pi}_{t-1} \rangle_{\boldsymbol{T}} - g_k^*(\pi_{k,t})$.
6: set $p^t = \arg\max_{p \in P} \langle p, \widetilde{f}_{t,k} \rangle - \phi^*(p) - \tau W(p^{t-1}, p)$.
7: set $x^t = \arg\min_{x \in X} f_0(x) + \langle x, p_t \boldsymbol{T} \boldsymbol{\pi}_t \rangle + \eta V(x^{t-1}, x)$.
8: **end for**
9: Return $\bar{x}^N = \sum_{t=1}^N \frac{x^t}{N}$.

$\boldsymbol{\pi} \in \boldsymbol{\Pi}$.

In the following analysis, we use some specific choices of norm adjustment constants to make explicit dependence of the iteration complexity bound on K.

a) When $\|\cdot\|_1$ and entropy W are used for P, we fix C_p =1.

b) When $\|\cdot\|_2$ and Euclidean W are used for P, we fix $C_p = \sqrt{K}$.

Proposition 2.3 below shows that the SD method achieves an $\mathcal{O}(1/N)$ reduction in $Q(z_N; u)$.

Proposition 2.3 *If the non-negative stepsizes satisfy*

$$\eta \geq \frac{C_p^2 M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2}{\tau} + \frac{M_{\boldsymbol{T}}^2}{\sigma}, \tag{2.9}$$

where $M_{\boldsymbol{T}}$, $M_{\boldsymbol{\Pi}}$ and C_p are defined in Section subsection 2.1.1 and Definition 2.2, then the following inequality holds for all $u \in Z$,

$$\sum_{t=1}^N Q(z_t; u) \leq \sigma \langle u_p, U_k(\boldsymbol{\pi}_0, u_{\boldsymbol{\pi}}) \rangle + \tau W(p_0, u_p) + \eta V(x_0, u_x). \tag{2.10}$$

Proof: First, consider the three projection steps of Algorithm Algorithm 1 for a fixed iteration $t \geq 1$. In the $\boldsymbol{\pi}$ update step, it follows from the standard three point inequality of proximal update, e.g., Lemma 3.4 in [28], that for a fixed k scenario,

$$-\langle 2x_t - x_{t-1}, \boldsymbol{\pi}_{t+1}\rangle_{T_k} + g_k^*(\boldsymbol{\pi}_{t+1}) + \sigma(U_k(\boldsymbol{\pi}_t, \boldsymbol{\pi}_{t+1}) + U_k(\boldsymbol{\pi}_{t+1}, u_{\boldsymbol{\pi}})) \\ \leq -\langle 2x_t - x_{t-1}, u_{\boldsymbol{\pi}}\rangle_{T_k} + g_k^*(u_{\boldsymbol{\pi}}) + \sigma U_k(\boldsymbol{\pi}_t, u_{\boldsymbol{\pi}}),$$

or equivalently,

$$\langle x_{t+1}, u_{\boldsymbol{\pi}} - \boldsymbol{\pi}_{t+1}\rangle_{T_k} - g_k^*(u_{\boldsymbol{\pi}}) + g_k^*(\boldsymbol{\pi}_{t+1}) \leq \sigma(U_k(\boldsymbol{\pi}_t, u_{\boldsymbol{\pi}}) - U_k(\boldsymbol{\pi}_{t+1}, u_{\boldsymbol{\pi}})) \\ + (\sigma U_k(\boldsymbol{\pi}_t, \boldsymbol{\pi}_{t+1}) + \langle x_{t+1} - (2x_t - x_{t-1}), u_{\boldsymbol{\pi}} - \boldsymbol{\pi}_{t+1}\rangle_{T_k}).$$

Summing up both sides with weight u_p, we get

$$Q_\pi(z_{t+1}; u) \leq \sigma\langle \boldsymbol{U}(\boldsymbol{\pi}_t, u_{\boldsymbol{\pi}}) - \boldsymbol{U}(\boldsymbol{\pi}_{t+1}, u_{\boldsymbol{\pi}}), u_p\rangle \\ + \langle\langle x_{t+1} - x_t, u_{\boldsymbol{\pi}} - \boldsymbol{\pi}_{t+1}\rangle_{\boldsymbol{T}} - \langle x_t - x_{t-1}, u_{\boldsymbol{\pi}} - \boldsymbol{\pi}_t\rangle_{\boldsymbol{T}}, u_p\rangle_{\boldsymbol{T}} + \epsilon_\pi(\boldsymbol{\pi}_{t+1}), \tag{2.11}$$

where

$$\epsilon_\pi(\boldsymbol{\pi}_{t+1}) = \langle u_p, \underbrace{\langle x_t - x_{t-1}, \boldsymbol{\pi}_{t+1} - \boldsymbol{\pi}_t\rangle_{\boldsymbol{T}} - \sigma \boldsymbol{U}(\boldsymbol{\pi}_t, \boldsymbol{\pi}_{t+1})}_{\leq \frac{1}{2\sigma}\|x_t - x_{t-1}\|_2^2 M_{\boldsymbol{T}}^2 \text{ for each component}}\rangle \leq \tfrac{1}{2\sigma}\|x_t - x_{t-1}\|_2^2 M_{\boldsymbol{T}}^2. \tag{2.12}$$

Next in the p update step, again it follows from Lemma 3.4 in [28] that

$$\langle u_p - p_{t+1}, \langle x_t, \boldsymbol{\pi}_{t+1}\rangle_{\boldsymbol{T}} + \langle x_t - x_{t-1}, \boldsymbol{\pi}_t\rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}_{t+1})\rangle + \\ \phi^*(p_{t+1}) - \phi^*(u_p) + \tau(W(p_t, p_{t+1}) + W(p_{t+1}, u_p)) \leq \tau W(p_t, u_p).$$

After adding $\langle u_p - p_{t+1}, \langle x_{t+1}, \boldsymbol{\pi}_{t+1}\rangle_{\boldsymbol{T}}\rangle$ to both sides of the inequality, we have

$$\begin{aligned} Q_p(z_{t+1};u) &\leq \langle u_p - p_{t+1}, \langle x_{t+1}-x_t, \boldsymbol{\pi}_{t+1}\rangle_{\boldsymbol{T}} - \langle x_t - x_{t-1}, \boldsymbol{\pi}_t\rangle_{\boldsymbol{T}}\rangle + \\ &\quad \tau(W(p_t,u_p) - W(p_{t+1},u_p) - W(p_t,p_{t+1})) \\ &\leq \tau(W(p_t,u_p) - W(p_{t+1},u_p)) + \epsilon_p(p_{t+1}) \\ &\quad + (\langle u_p - p_{t+1}, \langle x_{t+1}-x_t, \boldsymbol{\pi}_{t+1}\rangle_{\boldsymbol{T}}\rangle - \langle u_p - p_t, \langle x_t - x_{t-1}, \boldsymbol{\pi}_t\rangle_{\boldsymbol{T}}\rangle), \end{aligned} \tag{2.13}$$

where

$$\begin{aligned} \epsilon_p(p_{t+1}) &= \langle p_{t+1}-p_t, \langle x_t - x_{t-1}, \boldsymbol{\pi}_t\rangle_{\boldsymbol{T}}\rangle - \tau W(p_t,p_{t+1}) \\ &\leq \|p_{t+1}-p_t\|_W \|x_t - x_{t-1}\|_2 \left\|[\|T_1\boldsymbol{\pi}_{t,1}\|_2, ..., \|T_K\boldsymbol{\pi}_{t,K}\|_2]\right\|_{W^*} - \tau W(p_t,p_{t+1}) \\ &\leq \tfrac{1}{2\tau}\|x_t - x_{t-1}\|_2^2 (CpM_{\boldsymbol{T}}M_{\boldsymbol{\Pi}})^2. \end{aligned} \tag{2.14}$$

Moreover, when computing x_{t+1} in x update step, we can obtain the following simple inequality

$$\begin{aligned} Q_x(z_{t+1};u) &= \langle x_{t+1}-u_x, p_{t+1}\boldsymbol{T}\boldsymbol{\pi}_{t+1}\rangle + f_0(x_{t+1}) - f_0(u_x) \\ &\leq \eta(V(x_t,u_x) - V(x_{t+1},u_x)) - \eta V(x_t,x_{t+1}). \end{aligned} \tag{2.15}$$

Finally, summing up (paragraph 2.11), (paragraph 2.13), (paragraph 2.15)
for $t = 1, 2, 3, \ldots, N$ and applying the telescoping cancellation, we have

$$\begin{aligned} \textstyle\sum_{t=0}^{N-1} Q(z_{t+1};u) &\leq \sigma\langle u_p, \boldsymbol{U}(\boldsymbol{\pi}_0, u_{\boldsymbol{\pi}})\rangle + \tau W(p_0,u_p) + \eta V(x_0,u_x) - \eta V(x_N,u_x) \\ &\quad + \langle u_p, \langle x_N - x_{N-1}, u_{\boldsymbol{\pi}} - \boldsymbol{\pi}_N\rangle_{\boldsymbol{T}} - \sigma\boldsymbol{U}(\boldsymbol{\pi}_N, u_{\boldsymbol{\pi}})\rangle + (\langle u_p - p_N, \langle x_N - x_{N-1}, \boldsymbol{\pi}_N\rangle_{\boldsymbol{T}}\rangle \\ &\quad - \tau W(p_N,u_p)) + \underbrace{\textstyle\sum_{t=1}^{N-1}(\epsilon_p(p_{t+1}) + \epsilon_\pi(\boldsymbol{\pi}_{t+1}) - \eta V(x_{t-1},x_t))}_{\text{Notice that } \epsilon_p(p_1)=0 \text{ and } \epsilon_\pi(\boldsymbol{\pi}_1)=0 \text{ because } x_0 = x_{-1}.} - \eta V(x_{N-1},x_N). \end{aligned} \tag{2.16}$$

Observe that the stepsize requirement $\eta \geq C_p^2 M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2/\tau + M_{\boldsymbol{T}}^2/\sigma$ implies that the following parts of (paragraph 2.16) are smaller than 0:

$$\begin{aligned}
&\textstyle\sum_{t=2}^{N} \epsilon_p(p_t) + \epsilon_\pi(\boldsymbol{\pi}_t) - \eta V(x_{t-1}, x_t) \\
&\quad \leq \textstyle\sum_{t=2}^{N} \frac{1}{2\tau} \left\|x_t - x_{t-1}\right\|_2^2 (CpM_{\boldsymbol{T}} M_{\boldsymbol{\Pi}})^2 + \frac{1}{2\sigma} \left\|x_t - x_{t-1}\right\|_2^2 M_{\boldsymbol{T}}^2 - \frac{\eta}{2} \left\|x_t - x_{t-1}\right\|_2^2 \leq 0. \\
&\langle u_p, \langle x_N - x_{T-1}, u_{\boldsymbol{\pi}} - \boldsymbol{\pi}_N \rangle_{\boldsymbol{T}} - \sigma \boldsymbol{U}(\boldsymbol{\pi}_N, u_{\boldsymbol{\pi}}) \rangle \\
&\quad + (\langle u_p - p_N, \langle x_N - x_{T-1}, \boldsymbol{\pi}_N \rangle_{\boldsymbol{T}} \rangle - \tau W(p_N, u_p)) - \eta V(x_{N-1}, x_N) \leq 0.
\end{aligned}$$

So (Equation 2.10) follows by substituting the previous two inequalities and $-\eta V(x_N, u_x) \leq 0$ into (paragraph 2.16). □

The next theorem suggests a stepsize choice for Algorithm Algorithm 1 and shows its convergence in terms of function value gap.

Theorem 2.1 *If we set*

$$\sigma = M_{\boldsymbol{T}} \frac{\Omega_X}{\Omega_{\boldsymbol{\Pi}}}, \tau = M_{\boldsymbol{T}} M_{\boldsymbol{\Pi}} C_p \frac{\Omega_X}{\Omega_P}, \text{ and } \eta = M_{\boldsymbol{T}} M_{\boldsymbol{\Pi}} C_p \frac{\Omega_P}{\Omega_X} + M_{\boldsymbol{T}} \frac{\Omega_{\boldsymbol{\Pi}}}{\Omega_X}, \tag{2.17}$$

then after N iterations of Algorithm Algorithm 1, we have

$$f(\bar{x}^t) - f(x^*) \leq \frac{2\Omega_X M_{\boldsymbol{T}}}{N} (\Omega_{\boldsymbol{\Pi}} + C_p M_{\boldsymbol{\Pi}} \Omega_P). \tag{2.18}$$

Proof: Observe that the stepsize choices in (Equation 2.17) satisfies the requirement in (Equation 2.9) and $\Omega_{\boldsymbol{\Pi}}^2$, Ω_P^2, and Ω_X^2 are upper bounds for $\langle u_p, \boldsymbol{U}(\boldsymbol{\pi}_0, u_{\boldsymbol{\pi}}) \rangle, W(p_0, u_p)$ and $V(x_0, x^*)$ for any feasible $u_p, u_{\boldsymbol{\pi}}$. So it follows from Proposition 2.3 that

$$\textstyle\sum_{t=1}^{N} Q(z^t, (x^*, u_p, u_{\boldsymbol{\pi}})) \leq \sigma \Omega_{\boldsymbol{\Pi}}^2 + \tau \Omega_P^2 + \eta \Omega_X^2 \quad \forall (u_p, u_{\boldsymbol{\pi}}) \in P \times \boldsymbol{\Pi}.$$

Thus Proposition 2.2 implies that $f(\bar{x}^t) - f(x^*) \leq \frac{\sigma \Omega_{\boldsymbol{\Pi}}^2 + \tau \Omega_P^2 + \eta \Omega_X^2}{N}$. The bound (Equation 2.18) then follows from substituting the stepsize choices into the preceding in-

equality. □

We remark here that, by using $\sqrt{2}M_{\mathbf{\Pi}}$ as an upper bound for $\Omega_{\mathbf{\Pi}}$, the above convergence rate could be further simplified to $\Omega_X M_{\boldsymbol{T}} M_{\mathbf{\Pi}}(\sqrt{2} + C_p\Omega_P)/N$, i.e., $\mathcal{O}((1 + \Omega_P C_p)/N)$ if we ignore constants independent of K. Then substituting in the values of C_p and Ω_P, the iteration complexity bounds become $\mathcal{O}(\sqrt{\log K}/\epsilon)$ for entropy W and $\mathcal{O}(\sqrt{K}/\epsilon)$ for Euclidean W.

It is also worth noting that the aforementioned rate of convergence for SD seems to be tight for solving problem (Equation 2.7) since the $\mathcal{O}(1/N)$ rate of convergence is not improvable even for solving the simpler convex-concave bilinear saddle point problems [29, 30].

2.3 Sequential Smooth Level Method

In this section, we view (Equation 2.5) from the perspective of a structured non-smooth problem,

$$F(x) := \max_{p \in P} \sum_{k=1}^{K} \max_{\pi_k \in \Pi(k)} p_k(\langle T_k x, \pi_k \rangle - g_k^*(\pi_k)) - \phi^*(p). \tag{2.19}$$

(Equation 2.19) contains an additional maximization layer than those considered by Nesterov in [31]. Moreover, these two maximization layers cannot be combined because of non-separability and non-concavity issues. So the current smoothing technique are not directly applicable. To address such a difficulty, Subsection 3.1 extends the Nesterov's framework to build a two-layer smoothing scheme for F and analyzes its smooth approximation properties with respect to a sequence of points. Such a sequence-based approach helps us to determine a suitable smoothing scheme for the encountered points, rather than for the whole feasible region.

Another challenge is deciding the smoothing parameters to balance the conflicting goals of a small approximation gap (for a sound solution) and a small Lipschitz smoothness constant (for fast convergence). In fact, to calculate an optimal choice of those parameters

for a fixed smoothing scheme, we would need to know the distance to the output solution even before the algorithm is run, which is preposterous. Subsection 3.2 resolves such a difficulty by introducing a parameter-free bundle level type algorithm that operates on a dynamically smoothed F, where the smoothness parameters adjust in an on-line fashion to the encountered points.

2.3.1 Sequential Smoothing Scheme

By a smooth approximation for a non-smooth function f, we mean a convex function $\tilde{f}$ which is both L-smooth and close to f everywhere on its domain.

Definition 2.3 *Let f be a convex function on $\mathcal{X} \subset R^n$ equipped with norm $\|\cdot\|_{\mathcal{X}}$. We call a convex function $\tilde{f}$ its (α, β)-domain smooth approximation if*

a) $\left\|\nabla \tilde{f}(x_1) - \nabla \tilde{f}(x_2)\right\|_{\mathcal{X}^*} \leq \alpha \|x_1 - x_2\|_{\mathcal{X}} \quad \forall x_1, x_2 \in \mathcal{X}$,

b) $\tilde{f}(x) \leq f(x) \leq \tilde{f}(x) + \beta \quad \forall x \in \mathcal{X}$.

For our purpose of designing an adaptive smoothing algorithm, we need a weaker notion of smooth approximation. More specifically, since we use the accelerated proximal level (APL) method in [27] as the backbone of the SSL algorithm, it is useful to note that the L-smoothness constant is only used to bound the upper curvature constants associated with the linearization centers $\{x_t^l\}$ and the search points $\{x_t^{md}\}$. So we should focus on the upper curvature constant and the approximation gap associated with these points and define an (α, β)-*sequence smooth approximation*.

Definition 2.4 *Let f be a convex function on $\mathcal{X} \in R^n$ equipped with norm $\|\cdot\|_{\mathcal{X}}$ and let $\{(x_t^l, x_t^{md})\}_{t=1}^N$ be some sequence of points in $\mathcal{X}$. Then we call a convex differentiable function $\tilde{f}$ an (α, β)-sequence smooth approximation of f over $\{(x_t^l, x_t^{md})\}_{t=1}^N$ if the following conditions hold.*

a) $\widetilde{f}(x_t^{md}) - \widetilde{f}(x_t^l) - \langle \nabla \widetilde{f}(x_t^l), x_t^{md} - x_t^l \rangle \leq \frac{\alpha}{2} \left\|x_t^{md} - x_t^l\right\|_{\mathcal{X}}^2$.

b) $\tilde{f}(x_t^{md}) \leq f(x_t^{md}) \leq \tilde{f}(x_t^{md}) + \beta \ \ \forall t \in [N]$.

It is worth noting that if $\widetilde{f}$ is an (α, β)-*domain smooth approximation*, then it must be an (α, β)-*sequence smooth approximation* for all sequences. Moreover, if $\widetilde{f}$ is an (α, β)-*sequence smooth approximation* for all singleton sequences $\{x_t^l, x_t^{md}\}_{t=1}^1$, then it must be an (α, β)-*domain smooth approximation*. Because of such a close relationship, we use the generic name "smooth approximation" when referring to both of them.

Now we develop the two-layer smooth approximation scheme for (Equation 2.5). Let us briefly review Nesterov's smoothing scheme in [31] for the following structured non-smooth function $H : \mathcal{X} \to \mathbb{R}$,

$$H(x) = \max_{y \in Y} \langle x, Ay \rangle - \psi(y), \tag{2.20}$$

where $\psi(y)$ is some simple p.c.c. function defined on Y. Nesterov suggests adding a μ-multiple of some 1-strongly convex term ω to the inner y-maximization to obtain

$$H_\mu(x) = \max_{y \in Y} \langle x, Ay \rangle - \psi(y) - \mu\omega(y). \tag{2.21}$$

The following properties of H_μ are established in Theorem 1 of [31].

Lemma 2.1 *Let ω be 1-strongly convex with respect to some $\|\cdot\|_\omega$, then the following statements hold for H_μ defined in* (Equation 2.21).

a) *$H_\mu(\cdot)$ is convex and continuously differentiable with gradient $H'_\mu(x) = A^T\hat{y}$, where $\hat{y}$ is the unique solution to the maximization problem in $H_\mu(x)$.*

b) *For any $x_1, x_2 \in \mathcal{X}$ and their corresponding maximizers in $H_\mu(\cdot)$, $\hat{y}_1, \hat{y}_2$, we have $\langle A(x_1 - x_2), \hat{y}_1 - \hat{y}_2 \rangle \geq \mu \langle \nabla\omega(\hat{y}_1) - \nabla\omega(\hat{y}_2), \hat{y}_1 - \hat{y}_2 \rangle \geq \frac{\mu}{2} \|\hat{y}_1 - \hat{y}_2\|_\omega^2$.*

c) *If $\Omega_Y^2 := \max_{y \in Y} \omega(y)$, $H_\mu(\cdot)$ is an $(\frac{\|A\|_{\omega,X}^2}{\mu}, \mu\Omega_y^2)$-domain smooth approximation of $H(x)$.*

Returning to our problem (Equation 2.5), the subgradient of $F(x)$ is $p\boldsymbol{T}\boldsymbol{\pi}$. So to make it Lipschitz continuous, we can consider the following product rule type decomposition[④]:

$$p_1\boldsymbol{T}\boldsymbol{\pi}_1 - p_2\boldsymbol{T}\boldsymbol{\pi}_2 = \underbrace{(p_1 - p_2)}_{p \text{ smoothing}}\boldsymbol{T}\boldsymbol{\pi}_1 + p_2\boldsymbol{T}\underbrace{(\boldsymbol{\pi}_1 - \boldsymbol{\pi}_2)}_{\boldsymbol{\pi} \text{ smoothing}}. \tag{2.22}$$

If we smooth both the p-block and the $\boldsymbol{\pi}$-block, $p\boldsymbol{T}\boldsymbol{\pi}$ should be a Lipschitz continuous function of x. More specifically, for some $\bar{p} \in P$, we consider the following $F_{\mu_{\boldsymbol{\pi}},\mu_p}(x)$ smooth approximation,

$$\begin{aligned} g_{\mu_{\boldsymbol{\pi}},k}(x) &:= \max_{\pi_k \in \Pi(k)} \langle \pi_k, T_k x\rangle - g_k^*(\pi_k) - \mu_{\boldsymbol{\pi}} U(0,\pi_k), && (\boldsymbol{\pi} \text{ smoothing}) \\ F_{\mu_{\boldsymbol{\pi}},\mu_p}(x) &:= \max_{p\in P} \textstyle\sum_{k=1}^K p_k g_{\mu_{\boldsymbol{\pi}},k}(x) - \phi^*(p) - \mu_p W(\bar{p},p). && (p \text{ smoothing}) \end{aligned} \tag{2.23}$$

Notice that proxy center for $U(\bar{\pi}_k, \pi_k)$ is set to $\bar{\pi}_k := 0$. Such a choice allows us to use $M_{\boldsymbol{\Pi}}/\sqrt{2}$ to bound $\Omega_{\boldsymbol{\Pi}}$ so that we need to dynamically estimate only two problem parameters, $M_{\boldsymbol{\Pi}}$ and Ω_P.

Now we analyze the properties of the proposed smooth approximation (Equation 2.23). The following domain smooth approximation properties of (Equation 2.23) are direct consequences of Lemma 2.1.

Lemma 2.2 *The following statements hold for $F_{\mu_{\boldsymbol{\pi}},\mu_p}$ in* (Equation 2.23).

a) As a function of x, $g_{\mu_{\boldsymbol{\pi}},k}$ is a $(M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}}, \mu_{\boldsymbol{\pi}} M_{\boldsymbol{\Pi}}^2/2)$-domain smooth approximation of $g_k(T_k x)$.

b) As a function of $\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)$, $F_{\mu_p,\mu_{\boldsymbol{\pi}}}(\cdot)$ is a $(\|I\|_{g,W}/\mu_p, \mu_p\Omega_P^2)$-domain smooth approximation of $F(\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)) := \max_{p\in P} \sum_{k=1}^K p_k g_{\mu_{\boldsymbol{\pi}},k}(x) - \phi^*(p)$.

[④]Recall that $p\boldsymbol{T}\boldsymbol{\pi}$ is not matrix multiplication; it is merely a short hand for $\sum_{i=1}^K p_k T_k \pi_k$. However the decomposition in (Equation 2.22) is valid because $\sum_{i=1}^K p_k T_k \pi_k$ is linear with respect to p and $\boldsymbol{\pi}$.

Proof: Part b) is clear.

For part a), Lemma 2.1 implies that $g_{\mu_{\boldsymbol{\pi}},k}$ is a $(\|T_k\|_{2,2}^2/\mu_{\boldsymbol{\pi}}, \mu_{\boldsymbol{\pi}}(\max_{\pi_k\in\Pi(k)} U(0,\pi_k))$-*domain smooth approximation* of g_k. But $M_{\boldsymbol{T}}$ and $M_{\boldsymbol{\Pi}}^2/2$ are upper bounds for $\|T_k\|_{2,2}$ and $\max_{\pi_k\in\Pi(k)} U(0,\pi_k)$ for all k, so a) follows immediately. □

Just like the chain rule in calculus, we need the following technical result to reduce the above p-block L-smoothness property with respect to $\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)$ to that with respect to x.

Lemma 2.3 *Let $\|\cdot\|_{W^*}$ be the dual norm of the p-block. Then for any feasible (x_1, x_2) and their corresponding maximizers in the definition of $\mathbf{g}_{\mu_{\boldsymbol{\pi}}}$, $(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2)$, we have*

$$\|\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x_1) - \mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x_2)\|_{W^*} \le C_p M_{\boldsymbol{T}} \max\{\|\boldsymbol{\pi}_1\|_{2,\infty}, \|\boldsymbol{\pi}_2\|_{2,\infty}\} \|x_1 - x_2\|_2 . \tag{2.24}$$

Proof: First, we derive the following Lipschitz-continuity constant for each $g_{\mu_{\boldsymbol{\pi}},k}$:

$$|g_{\mu_{\boldsymbol{\pi}},k}(x_1) - g_{\mu_{\boldsymbol{\pi}},k}(x_2)| \le \max\{\|T_k^\intercal \pi_{1,k}\|_2, \|T_k^\intercal \pi_{2,k}\|_2\} \|x_1 - x_2\|_2 .$$

Because $|g_{\mu_{\boldsymbol{\pi}},k}(x_1) - g_{\mu_{\boldsymbol{\pi}},k}(x_2)|$ is the difference of two maximal values attained over the same domain, we can use the maximizer of the larger value in place of the maximizer of the smaller value to derive an upper bound. More specifically, if $g_{\mu_{\boldsymbol{\pi}},k}(x_1) \ge g_{\mu_{\boldsymbol{\pi}},k}(x_2)$, then

$$\begin{aligned}
&g_{\mu_{\boldsymbol{\pi}},k}(x_1) - g_{\mu_{\boldsymbol{\pi}},k}(x_2) \\
&:= \langle \pi_{1,k}, T_k x_1\rangle - g_k^*(\pi_{1,k}) - \mu_{\boldsymbol{\pi}} U(0,\pi_{1,k}) - \max_{\pi_k\in\Pi(k)} (\langle \pi_k, T_k x_2\rangle - g_k^*(\pi_k) - \mu_{\boldsymbol{\pi}} U(0,\pi_k)) \\
&\le \langle T_k \pi_{1,k}, x_1 - x_2\rangle \le \|x_1 - x_2\|_2 \|T_k \pi_{1,k}\|_2 .
\end{aligned}$$

A similar bound can also be obtained when $g_{\mu_{\boldsymbol{\pi}},k}(x_1) \le g_{\mu_{\boldsymbol{\pi}},k}(x_2)$. So we have

$$\begin{aligned}
&|g_{\mu_{\boldsymbol{\pi}},k}(x_1) - g_{\mu_{\boldsymbol{\pi}},k}(x_2)| \le \max\{\|T_k \pi_{1,k}\|_2, \|T_k \pi_{2,k}\|_2\} \|x_1 - x_2\|_2 \\
&\le M_{\boldsymbol{T}} \max\{\|\boldsymbol{\pi}_1\|_{2,\infty}, \|\boldsymbol{\pi}_2\|_{2,\infty}\} \|x_1 - x_2\|_2 .
\end{aligned}$$

Finally (Equation 2.24) follows from the definition of C_p in Definition 2.2. $\square$

Combining the previous two results, we obtain the following *sequence smooth approximation* property of (Equation 2.23).

Proposition 2.4 *Let $\{x_t^l, x_t^{md}\}_{t=1}^N$ be given. Let $\{\hat{p}_t^u, \hat{\pi}_t^u\}$ be the maximizers for $\{F(x_t^{md})\}$ in (Equation 2.19), and let $\{p_t^l, \pi_t^l\}$ and $\{p_t^u, \pi_t^u\}$ be the maximizers for $\{F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_t^l)\}$ and $\{F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_t^{md})\}$ in* (Equation 2.23).
If $\bar{\Omega}_p^2 \geq \max_{t\in[N]} W(\bar{p}_t, p)$ and $\bar{M}_{\boldsymbol{\Pi}} \geq \max_{t\in[N]} \max\{\|\hat{\pi}_t^u\|_{2,\infty}, \|\boldsymbol{\pi}_t^u\|_{2,\infty}, \left\|\boldsymbol{\pi}_t^l\right\|_{2,\infty}\}$, then $F_{\mu_{\boldsymbol{\pi}},\mu_p}$ is a $(2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}} + 2C_p^2\bar{M}_{\boldsymbol{\Pi}}^2 M_{\boldsymbol{T}}^2/\mu_p, \mu_p\bar{\Omega}_p^2 + \mu_{\boldsymbol{\pi}}\bar{M}_{\boldsymbol{\Pi}}^2/2)$-sequence smooth approximation of F on $\{x_t^l, x_t^{md}\}_{t=1}^N$.

Proof: Let a $t \in [N]$ be given. For notation convenience, we use x_1 and x_2 to denote x_t^{md} and x_t^l and use $(p_1, \boldsymbol{\pi}_1)$ and $(p_2, \boldsymbol{\pi}_2)$ to denote their corresponding maximizers in $F_{\mu_{\boldsymbol{\pi}},\mu_p}$ (Equation 2.23). Denoting $\tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x, p, \boldsymbol{\pi}) := \sum_{k=1}^K p_k(\langle \boldsymbol{\pi}, x\rangle_{\boldsymbol{T}} - g_k^*(\boldsymbol{\pi}) - \mu_{\boldsymbol{\pi}} V(\bar{\pi}_k, \boldsymbol{\pi})) - \phi^*(p) - \mu_p W(\bar{p}, p)$, we have the following decomposition for the upper curvature error,

$$
\begin{aligned}
&F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_1) - F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_2) - \langle \nabla F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_2), x_1 - x_2\rangle \\
&= \tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x_1, p_1, \boldsymbol{\pi}_1) - \max_{p,\boldsymbol{\pi}} \tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x_2, p, \boldsymbol{\pi}) - \langle p_2 \boldsymbol{T}\boldsymbol{\pi}_2, x_1 - x_2\rangle \\
&\overset{(a)}{\leq} \tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x_1, p_1, \boldsymbol{\pi}_1) - \tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x_2, p_1, \boldsymbol{\pi}_1) - \langle p_2 \boldsymbol{T}\boldsymbol{\pi}_2, x_1 - x_2\rangle \\
&= \langle p_1 \boldsymbol{T}\boldsymbol{\pi}_1, x_1 - x_2\rangle - \langle p_2 \boldsymbol{T}\boldsymbol{\pi}_2, x_1 - x_2\rangle \\
&= \langle p_1 \boldsymbol{T}\boldsymbol{\pi}_1 - p_2 \boldsymbol{T}\boldsymbol{\pi}_2, x_1 - x_2\rangle \\
&= \underbrace{\langle p_1 \boldsymbol{T}(\boldsymbol{\pi}_1 - \boldsymbol{\pi}_2), x_1 - x_2\rangle}_{A} + \underbrace{\langle (p_1 - p_2)\boldsymbol{T}\boldsymbol{\pi}_2, x_1 - x_2\rangle}_{B},
\end{aligned}
$$

where (a) uses $\tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x_2, p_1, \boldsymbol{\pi}_1)$ as a lower bound for $\max_{p,\boldsymbol{\pi}} \tilde{F}_{\mu_p,\mu_{\boldsymbol{\pi}}}(x_2, p, \boldsymbol{\pi})$. To bound A, we conclude from Lemma 2.2.a) that

$$
A \leq \|x_1 - x_2\|_2 \sum_{k=1}^K p_{k,1} \max_{k\in K} \|T_k'(\pi_{1,k} - \pi_{2,k})\|_2 \leq \tfrac{M_{\boldsymbol{T}}^2}{\mu_{\boldsymbol{\pi}}} \|x_1 - x_2\|_2^2 . \tag{2.25}
$$

To bound B, we use Lemma 2.2.b) and Lemma 2.3 to obtain

$$\begin{aligned}\|p_1 - p_2\|_W &\leq \tfrac{\|I\|_{W^*,W}}{\mu_p} \|\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x_1) - \mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x_2)\|_{W^*} \\ &\leq \tfrac{1}{\mu_p} C_p M_{\boldsymbol{T}} \max\{\|\boldsymbol{\pi}_1\|_{2,\infty}, \|\boldsymbol{\pi}_2\|_{2,\infty}\} \|x_1 - x_2\|_2 \leq \tfrac{1}{\mu_p} C_p M_{\boldsymbol{T}} \bar{M}_{\boldsymbol{\Pi}} \|x_1 - x_2\|_2,\end{aligned}$$

which implies that

$$\begin{aligned}B &= \langle p_1 - p_2, \langle \boldsymbol{\pi}_2, x_1 - x_2 \rangle_{\boldsymbol{T}} \rangle \\ &\leq \|p_1 - p_2\|_W \left\| [\|T_1^\intercal \pi_{2,1}\|_2 \|x_1 - x_2\|_2, ..., \|T_K^\intercal \pi_{2,K}\|_2 \|x_1 - x_2\|_2] \right\|_{W^*} \\ &\leq \tfrac{1}{\mu_p} C_p M_{\boldsymbol{T}} \bar{M}_{\boldsymbol{\Pi}} \left\| [\|T_1^\intercal \pi_{2,1}\|_2, ..., \|T_K^\intercal \pi_{2,K}\|_2] \right\|_{W^*} \|x_1 - x_2\|_2^2 \\ &\overset{(b)}{\leq} \tfrac{1}{\mu_p} (C_p M_{\boldsymbol{T}} \bar{M}_{\boldsymbol{\Pi}})^2 \|x_1 - x_2\|_2^2,\end{aligned} \tag{2.26}$$

where (b) follows from the the definition of C_p in Definition 2.2. Combining (Equation 2.25) and (Equation 2.26), we obtain the desired upper-curvature constant of $2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}} + 2C_p^2 \bar{M}_{\boldsymbol{\Pi}}^2 M_{\boldsymbol{T}}^2/\mu_p$. Moreover, it is easy to see that for a given x_t^{md}, we have

$$g_{\mu_{\boldsymbol{\pi}},k}(x_t^{md}) \leq g_k(x_t^{md}) \leq g_{\mu_{\boldsymbol{\pi}},k}(x_t^{md}) + \mu_{\boldsymbol{\pi}} U(0, \hat{\pi}_{t,k}^u) \leq g_{\mu_{\boldsymbol{\pi}},k}(x_t^{md}) + \mu_{\boldsymbol{\pi}} \tfrac{\bar{M}_{\boldsymbol{\Pi}}^2}{2},$$

and hence

$$\begin{aligned}F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_t^{md}) \leq F(x_t^{md}) &\leq F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_t^{md}) + \textstyle\sum_{k=1}^K \hat{p}_{t,k}^u \mu_{\boldsymbol{\pi}} \tfrac{\bar{M}_{\boldsymbol{\Pi}}^2}{2} + \mu_p W(\bar{p}, \hat{p}_t^u) \\ &\leq F_{\mu_{\boldsymbol{\pi}},\mu_p}(x_t^{md}) + \mu_p \bar{\Omega}_p^2 + \mu_{\boldsymbol{\pi}} \tfrac{\bar{M}_{\boldsymbol{\Pi}}^2}{2}.\end{aligned}$$

□

Using $M_{\boldsymbol{\Pi}}$ and Ω_P as upper bounds for $\bar{M}_{\boldsymbol{\Pi}}$ and $\bar{\Omega}_p$ for any $(x_1, x_2) \in X \times X$, we obtain the following *domain smooth approximation* properties of (Equation 2.23) below as an immediate corollary.

Corollary 2.1 *$F_{\mu_{\boldsymbol{\pi}},\mu_p}$ is a $(2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}} + 2C_p^2 M_{\boldsymbol{\Pi}}^2 M_{\boldsymbol{T}}^2/\mu_p, \mu_p \Omega_P^2 + \mu_{\boldsymbol{\pi}} M_{\boldsymbol{\Pi}}^2/2)$-domain smooth approximation of F.*

The need to select two smoothing parameters, μ_p and $\mu_{\boldsymbol{\pi}}$, makes (Equation 2.23) rather complicated. The next result shows a reduction to a single-parameter smoothing scheme by fixing an optimal ratio between μ_p and $\mu_{\boldsymbol{\pi}}$.

Lemma 2.4 *Let $F_{\mu_{\boldsymbol{\pi}},\mu_p}$ be a $(2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}} + 2C_p^2\bar{M}_{\boldsymbol{\Pi}}^2 M_{\boldsymbol{T}}^2/\mu_p, \mu_p\bar{\Omega}_p^2 + \mu_{\boldsymbol{\pi}}\bar{M}_{\boldsymbol{\Pi}}^2/2)$-smooth approximation of F, then the optimal ratio is*

$$\frac{\mu_p}{\mu_{\boldsymbol{\pi}}} = \frac{C_p\bar{M}_{\boldsymbol{\Pi}}^2}{\sqrt{2}\bar{\Omega}_p}$$

.

Proof: To achieve the smallest gap while maintaining a Lipschitz constant at $1/\mu$, we solve the following optimization problem analytically by the KKT condition,

$$\min_{\mu_p,\mu_{\boldsymbol{\pi}}\geq 0}\left\{\mu_p\bar{\Omega}_p^2 + \mu_{\boldsymbol{\pi}}\frac{M_{\boldsymbol{\Pi}}^2}{2} : \frac{2C_p^2M_{\boldsymbol{T}}^2\bar{M}_{\boldsymbol{\Pi}}^2}{\mu_p} + \frac{2M_{\boldsymbol{T}}^2}{\mu_{\boldsymbol{\pi}}} = \frac{1}{\mu}\right\}.$$

□

Using the above optimal ratio, F_μ defined below is then a $(M_{\boldsymbol{T}}^2/\mu, (1+\sqrt{2}C_p\bar{\Omega}_p)^2\bar{M}_{\boldsymbol{\Pi}}^2\mu)$-*sequence smooth approximation* of F:

$$F_\mu(x) = F_{\bar{\mu}_p,\bar{\mu}_\pi}(x) \text{ with } \bar{\mu}_\pi := \mu(2+2\sqrt{2}C_p\bar{\Omega}_p) \text{ and } \bar{\mu}_p := \mu(\sqrt{2}+2C_p\bar{\Omega}_p)\bar{M}_{\boldsymbol{\Pi}}^2C_p/\bar{\Omega}_p. \tag{2.27}$$

Moreover, if we replace $\bar{\Omega}_{\boldsymbol{\pi}}$ and $\bar{M}_{\boldsymbol{\Pi}}$ with their uniform upper bounds, Ω_P and $M_{\boldsymbol{\Pi}}$, then (Equation 2.27) must be a $(M_{\boldsymbol{T}}^2/\mu, (1+\sqrt{2}C_p\Omega_P)^2M_{\boldsymbol{\Pi}}^2\mu)$-*domain smooth approximation* of F. Observe that the smooth approximation properties of F_μ in (Equation 2.27) and H_μ in (Equation 2.21) studied by Nesterov [31] differ only by a constant factor, therefore any variant of Nesterov's accelerated gradient method could be applied to a fixed $f_\mu := f_0 + F_\mu$ to achieve an $\mathcal{O}(C_p\Omega_P/\epsilon)$ iteration complexity bound. However, this approach suffers from the same drawback as Nesterov's smoothing scheme in [31], i.e., one has to use

conservative estimates of $\Omega_{\mathbf{\Pi}}$ and Ω_P to guarantee an $\mathcal{O}(\epsilon/2)$ uniform approximation gap. This usually leads to a large L-smoothness constant for F_μ, and thus a slow convergence. To address this shortcoming, we present in the next subsection a novel SSL algorithm which operates on an adaptively smoothed F_μ.

2.3.2 Sequential Smooth Level Method

Algorithm 2 SSL Phase

Input: $\bar{x}, \mathrm{lb}, \bar{M}^2_{\mathbf{\Pi}}, \bar{\Omega}^2_p, \bar{\lambda}$

Output: $\tilde{x}, \tilde{\mathrm{lb}}, \tilde{M}^2_\pi, \tilde{\Omega}^2_p, \tilde{\lambda}$

1: **Initialization:** set $x_0^u := \bar{x},\ \bar{v}_0 := f(x_0^u),\ \underline{v}_0 := \mathrm{lb},\ l := \frac{1}{2}(\underline{v}_0 + \bar{v}_0), \theta := \frac{1}{2}$, and $\mu := \frac{\theta(\bar{v}_0 - l)}{\bar{M}^2_{\mathbf{\Pi}}(1+\sqrt{2}\bar{\Omega}_p C_p)^2\bar{\lambda}}$. Set the initial localizer $X'_0 := X$ and $t := 1$.

2: **while** True **do**

3: ***Update the lower bound***: set $x_t^l := (1-\alpha_t)x_{t-1}^u + (\alpha_t)x_{t-1}$. Evaluate f_μ at x_t^l to get $(p_t^l, \boldsymbol{\pi}_t^l)$ and construct a supporting function $s(x_t^l, x) := f_0(x) + F_\mu(x_t^l) + \langle \nabla F_\mu(x_t^l), x - x_t^l\rangle$. Let $s_t := \arg\min_{x\in X'_{t-1}} s(x_t^l, x)$ and $\underline{v}_t := \max\{\underline{v}_{t-1}, \min\{s_t, l\}\}$.
If $\underline{v}_t \geq l - \theta(l - \underline{v}_0)$, **return** $(x_{t-1}^u, \underline{v}_t, \bar{M}^2_{\mathbf{\Pi}}, \bar{\Omega}^2_p, \bar{\lambda})$.

4: ***Update the prox center***: set $x_t := \arg\min_{x\in X'_{t-1}, s(x_t^l,x)\leq l}\{V(x_0, x)\}$.

5: ***Update the upper bound***: set $x_t^{md} := (1-\alpha_t)x_t^u + \alpha_t x_t$ and evaluate f and f_μ at x_t^{md} to get $(\hat{p}_t^{md}, \hat{\boldsymbol{\pi}}_t^{md})$ and $(p_t^{md}, \boldsymbol{\pi}_t^{md})$. Set $\bar{v}_t := \min\{\bar{v}_{t-1}, f(x_t^{md})\}$ and choose x_t^u such that $f(x_t^u) = \bar{v}_t$.
If $\bar{v}_t \leq l + \theta(\bar{v}_0 - l)$, **return** $(x_t^u, \underline{v}_t, \bar{M}^2_{\mathbf{\Pi}}, \bar{\Omega}^2_p, \bar{\lambda})$.

6: ***Check π radius***: let $\tilde{M}^2_{\boldsymbol{\pi}} := \max_{k\in K}\max\{U(0, \pi_{k,t}^l), U(0, \hat{\pi}_{k,t}^{md}), U(0, \pi_{t,k}^{md})\}$.
If $\tilde{M}^2_{\boldsymbol{\pi}} > \bar{M}^2_{\mathbf{\Pi}}$, **return** $(x_t^u, \underline{v}_t, 2\tilde{M}^2_{\boldsymbol{\pi}}, \bar{\Omega}^2_p, \bar{\lambda})$.

7: ***Check p radius***: let $\tilde{\Omega}^2_p := W(\bar{p}, p_t^{md})$. If $\tilde{\Omega}^2_p > \bar{\Omega}^2_p$, **return** $(x_t^u, \underline{v}_t, \bar{M}^2_{\mathbf{\Pi}}, 2\tilde{\Omega}^2_p, \bar{\lambda})$.

8: ***Check aggressiveness param*** λ: if $f_\mu(x_t^{md}) \leq l + \frac{\theta}{2}(\bar{v}_0 - l)$, **return** $(x_t^u, \underline{v}_t, \bar{M}^2_{\mathbf{\Pi}}, \bar{\Omega}^2_p, 2\bar{\lambda})$.

9: ***Update the localizer***: choose an arbitrary X'_t such that $\underline{X}_t \subset X'_t \subset \bar{X}_t$ where $\underline{X}_t := \{x \in X'_{t-1} : s(x_t^l, x) \leq l\}$ and $\bar{X}_t := \{x \in X : \langle \nabla_{x=x_k} V(x_0, x), x - x_k\rangle \geq 0\}$.

10: Set $t := t + 1$.

11: **end while**

The bundle level method maintains both an upper and a lower bound on f_*. The upper bound $\bar{f}$ is the minimum function value of all the encountered points, while the lower bound $\underline{f}$ is the minimum value of a lower approximation model $h(x)$, namely bundle, consisted

of all evaluated cutting planes for f. In each iteration, $\bar{f}$ and $\underline{f}$ are used to construct a level set, say $\{x : h(x) \leq l := (\bar{f} + \underline{f})/2\}$, in which the next search point and the next cutting plane will be found. By repeating this process many times, the gap between such lower and upper bounds can be decreased to ϵ, upon which an ϵ-optimal solution must have been found.

To build an adaptive smoothing algorithm, we follow [32, 27] to partition the iterations into phases, inside which some important parameters are fixed. In [32], the constant l for defining a level set is fixed to allow the use of a restricted memory localizer. A phase of the NERML algorithm in [32] is terminated only when the upper bound or the lower bound has made enough progress to warrant a new l for the next phase. In our SSL algorithm, similar to [27], we fix both l and the smooth approximation function F_μ in a phase. The smoothing parameters $(\bar{\mu}_p, \bar{\mu}_\pi)$ in (Equation 2.27) are computed using current radii estimates. If these radii estimates are violated by a new point, we also terminate the current phase such that a more appropriate smoothing scheme can be constructed for the next phase. So each phase has two goals: to reduce the gap between the lower and upper bounds, and to update the radii estimates and hence the smoothing scheme.

- Radius Update: Line 6, 7, and 8 of the SSL Phase in Algorithm Algorithm 2. For each phase, we should construct a *sequence smooth approximation* F_μ with the smallest possible upper curvature constant for fast termination. In the USL method in [27], the L-smoothness constant of the smooth approximation H_μ is $\mathcal{O}(\bar{\Omega}_Y)$ and the estimate of $\bar{\Omega}_Y$ is updated only when it is absolutely necessary; the objective value achieved by the smooth approximation is well below the upper bound termination threshold, i.e., $H_\mu(x_t^u) \leq l + \theta(\bar{v}_0 - l)/2$, while the true objective value is above the upper bound termination threshold, i.e., $H(x_t^u) \geq l + \theta(\bar{v}_0 - l)$. In this way, [27] underestimates $\bar{\Omega}_Y$ to encourage an aggressively small upper curvature constant. Our situation is different because we need both accurate estimates of radii $\bar{M}_{\mathbf{\Pi}}$ and $\bar{\Omega}_p$ to determine the optimal ratio between μ_p and $\mu_{\boldsymbol{\pi}}$ in Lemma 2.4 and an aggressively

small upper curvature constant for fast convergence. So we create a separate variable λ to control the aggressiveness of the smooth approximation and use $\bar{M}_{\mathbf{\Pi}}$ and $\bar{\Omega}_p$ for estimating $M_{\mathbf{\Pi}}$ and Ω_P only. The radius update block in Algorithm Algorithm 2 thus has two components: 1) Line 6 and 7 check our estimates against the distances of encountered points to the fixed smoothing centers, $\bar{p}$ and 0. Once we find any violations, the violated radius estimate is doubled and the phase is terminated so that the next phase can construct a more appropriate smooth approximation. 2) Line 8 updates the aggressiveness parameter λ in the same fashion as the $\bar{\Omega}_Y$ update in the USL method. It is doubled only when the objective value achieved by the smoothed approximation is well below the upper bound termination threshold, $f_\mu(x_t^u) \leq l + \theta(\bar{v}_0 - l)/2$, while the true objective value is above the upper bound termination threshold, $f(x_t^u) \geq l + \theta(\bar{v}_0 - l)$, i.e., the approximation gap is too large.

- Gap Reduction: Line 3, 4, 5, and 9 of the SSL Phase in Algorithm Algorithm 2. This is essentially the composite accelerated proximal level (APL) method [27] applied to the composite smooth approximation function $f_\mu := f_0 + F_\mu$. Notice that, similar to Nesterov's accelerated gradient method [33], we use three sequences of points $\{x_t^l\}$, $\{x_l^{md}\}$ and $\{x_t\}$; we pick $x_t^l := (1 - \alpha_t)x_{t-1}^u + (\alpha_t)x_{t-1}$ to construct the composite cutting plane model and $x_t^{md} := (1 - \alpha_t)x_t^u + \alpha_t x_t$ to evaluate the objective value. It is shown in [27] that the following convergence result holds for any composite smooth function, and our f_μ in particular.

Lemma 2.5 *Let $\alpha_t = 2/(t+1)$, and also let $\{x_t^l\}$, $\{x_t^{md}\}$ and $\{x_t^u\}$ be the sequences of points generated by Algorithm Algorithm 2 before it terminates. If $\{x_t^l, x_t^{md}\}_{t=1}^N$ satisfy $f_\mu(x_t^{md}) - s(x_t^l; x_t^{md}) \leq \frac{M}{2}\left\|x_t^{md} - x_t^l\right\|^2$ for some $M \geq 0$, then we have*

$$f_\mu(x_N^u) - l \leq \frac{M\Omega_X^2}{N^2}.$$

Before Algorithm Algorithm 2 terminates, our estimates $\bar{M}_{\mathbf{\Pi}}$ and $\bar{\Omega}_p$ satisfy assump-

tions in Proposition 2.4, so F_μ in (Equation 2.27) is a $(M_{\boldsymbol{T}}^2/\mu, (1+\sqrt{2}C_p\bar{\Omega}_p)^2\bar{M}_{\boldsymbol{\Pi}}^2\mu)$-sequence smooth approximation of F. Therefore our choice of $\mu := \frac{\theta(\bar{v}_0-l)}{\bar{M}_{\boldsymbol{\Pi}}^2(1+\sqrt{2}\bar{\Omega}_pC_p)^2\bar{\lambda}}$ in Algorithm Algorithm 2 implies that

$$F_\mu \text{ is a } (\tfrac{M_{\boldsymbol{T}}^2\bar{M}_{\boldsymbol{\Pi}}^2(1+\sqrt{2}\bar{\Omega}_pC_p)^2\boldsymbol{\lambda}}{\theta(v_0-l)}, \tfrac{\theta(v_0-l)}{\boldsymbol{\lambda}})\text{-}\textit{sequence smooth approximation} \text{ of } F \text{ over } \{x_t^l, x_t^{md}\}. \tag{2.28}$$

By substituting $M = \frac{M_{\boldsymbol{T}}^2\bar{M}_{\boldsymbol{\Pi}}^2(1+\sqrt{2}\bar{\Omega}_pC_p)^2\boldsymbol{\lambda}}{\theta(v_0-l)}$ into Lemma 2.5, we can obtain the following bound on the number of iterations performed by the SSL Phase in Algorithm Algorithm 2.

Proposition 2.5 *Let* $\alpha_t := 2/(t+1)$ *and* $\Delta_0 := f(\bar{x}) - \mathrm{lb}$. *The SSL Phase in Algorithm Algorithm 2 terminates in at most* $(4\sqrt{2}\Omega_X M_{\boldsymbol{T}}\bar{M}_{\boldsymbol{\Pi}}\sqrt{\boldsymbol{\lambda}}(1+\sqrt{2}C_p\bar{\Omega}_p))/\Delta_0$ *iterations.*

Proof: Assuming that all other termination conditions have not been reached, then Algorithm Algorithm 2 will terminate in Line 8 if $f_\mu(x_N^{md}) - l \le \frac{1}{2}\theta(\bar{v}_0 - l) := \frac{1}{8}\Delta_0$. So it follows from (Equation 2.28) and Lemma 2.5 that the maximum number of iterations, N_{SSL}, is bounded by

$$\tfrac{1}{8}\Delta_0 \le \frac{M_{\boldsymbol{T}}^2\bar{M}_{\boldsymbol{\Pi}}^2(1+\sqrt{2}\bar{\Omega}_PC_p)^2\boldsymbol{\lambda}\Omega_X^2}{\frac{1}{4}\Delta_0}\frac{1}{N_{SSL}^2}.$$

After some simplification, we obtain the desired finite termination bound. □

Algorithm 3 Sequential Smoothing Level Method

Input: $\bar{x}_0 \in X$, tolerance $\epsilon > 0$, initial estimate $\bar{\Omega}_{p,0}^2 \in (0, \Omega_P^2], \bar{\Omega}_{\pi,0}^2 \in (0, \Omega_{\boldsymbol{\Pi}}^2], \bar{q}_0 \in (0,1]$ and $\lambda_0 \in (0,1)$

Output: $\bar{x}$, an $\epsilon-$suboptimal solution

1: **Initialization** Set $\bar{x}_1 = \arg\min_{x\in X}\{h(\bar{x}_0, x) = f_0(x) + F(\bar{x}_0) + \langle \nabla F(\bar{x}_0), x - \bar{x}_0\rangle\}$, $\mathrm{lb}_1 = h(\bar{x}_0, \bar{x}_1)$ and $\mathrm{ub}_1 = \min\{f(\bar{x}_0), f(\bar{x}_1)\}$. Set $s = 0$.
2: **while** True **do**
3: If $\mathrm{ub}_s - \mathrm{lb}_s \le \epsilon$, **terminate** with $\bar{x} = \bar{x}_s$.
4: Set $(\bar{x}_{s+1}, \mathrm{lb}_{s+1}, \bar{M}_{\pi,s+1}^2, \bar{\Omega}_{p,s+1}^2, \boldsymbol{\lambda}_{s+1}) =$ SSL-Phase$(\bar{x}_s, \mathrm{lb}_s, \bar{M}_{\pi,s}^2, \bar{\Omega}_{p,s}^2, \boldsymbol{\lambda}_s)$ and set $\mathrm{ub}_{s+1} = f(\bar{x}_{s+1})$.
5: Set $s = s + 1$.
6: **end while**

There are two ways for the SSL Phase Algorithm Algorithm 2 to terminate. If it terminates in Line 3 or Line 5, the gap between the lower and upper bounds is reduced by

a factor of at least $1/4$. So we call it a gap reduction phase. Otherwise, if it terminates in Line 6, 7, or 8, then one of the estimates $\bar{\Omega}_p^2$, $\bar{M}_{\mathbf{\Pi}}^2$ and $\boldsymbol{\lambda}$ must have been enlarged by a factor of two. So we call it an estimate enlargement phase. Because $\bar{\Omega}_p^2$ or $\bar{M}_{\mathbf{\Pi}}^2$ is doubled only when a p or a $\boldsymbol{\pi}$ exceeding its current radius estimate is found, $\bar{\Omega}_p^2$ and $\bar{M}_{\mathbf{\Pi}}^2$ are upper bounded by $2\Omega_P^2$ and $2M_{\mathbf{\Pi}}^2$ respectively. Similarly, since the difference between f and f_μ on observed points x_t^{md} is at most $\theta(v_0 - l)/(\boldsymbol{\lambda})$ (by (Equation 2.28)), the termination condition, $f(x_t^{md}) > l + \theta(v_0 - l)$ and $f_u(x_t^{md}) < l + \frac{\theta}{2}(v_0 - l)$ in Line 8 can be satisfied only if $\boldsymbol{\lambda} < 2$, i.e., $\boldsymbol{\lambda}$ must be bounded by 4. Therefore, if we repeat the SSL Phase Algorithm with updated lb, $\bar{x}$, $\bar{M}_{\mathbf{\Pi}}$, $\bar{\Omega}_p$ and $\boldsymbol{\lambda}$ in Algorithm Algorithm 3, there will only be a finite number of estimate enlargement phases, and the gap reduction phases should reduce the gap to ϵ eventually. Thus we have the following iteration complexity result for the SSL Algorithm.

Theorem 2.2 *Let $\alpha_t := 2/(t+1)$ and f_0 be Lipschitz continuous with constant M_0. To obtain an ϵ-suboptimal solution, the SSL algorithm requires at most $\mathcal{P}_s = \log_{4/3}(2\Omega_X(\sqrt{2}M_0 + M_{\boldsymbol{T}}\Omega_{\mathbf{\Pi}}(\sqrt{2} + C_p\Omega_P))/\epsilon)$ gap reduction phases and $\mathcal{P}_N = \log_2(\Omega_{\mathbf{\Pi}}^2/\bar{\Omega}_{\pi,0}^2) + \log_2(\Omega_P^2/\bar{\Omega}_{p,0}^2) + \log_2(1/\bar{q}_0) + 4$ parameter enlargement phases. In total, the number of iterations performed by Algorithm Algorithm 2 can be bounded by*

$$16(7\sqrt{2} + \log_2 \tfrac{\Omega_P^2}{\bar{\Omega}_{p,0}^2}\sqrt{2} + 2)\tfrac{\Omega_X M_{\boldsymbol{T}}\Omega_{\mathbf{\Pi}}(1+2C_p\Omega_P)}{\epsilon}.$$

Proof: Firstly, let us consider the gap reduction phases. A bound for the initial gap is

$$\begin{aligned}\mathrm{ub}_1 - \mathrm{lb}_1 &\le f(\bar{x}_1) - f(\bar{x}_0) - \langle f'(\bar{x}_0), \bar{x}_1 - \bar{x}_0\rangle \le \langle f'(\bar{x}_1) - f'(\bar{x}_0), \bar{x}_1 - \bar{x}_0\rangle \\ &= \underbrace{\langle f_0'(\bar{x}_1) - f_0'(\bar{x}_0), \bar{x}_1 - \bar{x}_0\rangle}_{A} + \underbrace{\langle \nabla F(\bar{x}_1) - \nabla F(\bar{x}_0), \bar{x}_1 - \bar{x}_0\rangle}_{B}.\end{aligned}$$

By the Cauchy Schwartz inequality and the triangle inequality, the following bounds on A and B hold,

$$\begin{aligned}
A &\leq \left(\|f_0'(\bar{x}_1)\|_2 + \|f_0'(\bar{x}_0)\|_2\right) \|\bar{x}_1 - \bar{x}_0\|_2 \leq 2M_0\sqrt{2}\Omega_X = 2\sqrt{2}M_0\Omega_X, \\
B &\leq \|\bar{x}_1 - \bar{x}_0\|_2 \left(\|(p_1 - p_2)\boldsymbol{T}\boldsymbol{\pi}_1\|_2 + \|p_2\boldsymbol{T}(\boldsymbol{\pi}_1 - \boldsymbol{\pi}_2)\|\right) \\
&\leq \sqrt{2}\Omega_X(\sqrt{2}M_{\boldsymbol{T}}\Omega_P M_{\boldsymbol{\Pi}} C_p + 2M_{\boldsymbol{T}}M_{\boldsymbol{\Pi}}) = 2M_{\boldsymbol{T}}M_{\boldsymbol{\Pi}}\Omega_X(\sqrt{2} + C_p\Omega_P).
\end{aligned}$$

So we have $\mathrm{ub}_1 - \mathrm{lb}_1 \leq 2\Omega_X(\sqrt{2}M_0 + M_{\boldsymbol{T}}\Omega_{\boldsymbol{\Pi}}(\sqrt{2} + C_p\Omega_P))$, and that number of gap reduction phases are bounded by $\mathcal{P}_s = \log_{4/3}(2\Omega_X(\sqrt{2}M_0 + M_{\boldsymbol{T}}\Omega_{\boldsymbol{\Pi}}(\sqrt{2} + C_p\Omega_P))/\epsilon)$. For the estimate enlargement phases, as discussed before, the upper bounds for $\bar{M}^2_{\boldsymbol{\Pi}},\ \bar{\Omega}^2_p$ and $\boldsymbol{\lambda}$ are $2\Omega^2_{\boldsymbol{\Pi}},\ 2\Omega^2_P$ and 4 respectively, hence there are at most $\mathcal{P}_N = \log_2(\Omega^2_{\boldsymbol{\Pi}}/\bar{\Omega}^2_{\pi,0}) + \log_2(\Omega^2_P/\bar{\Omega}^2_{p,0}) + \log_2(1/\boldsymbol{\lambda}_0) + 4$ phases.

Next, we develop separate bounds on the total number of iterations required for the gap reduction phases, $\bar{M}^2_{\boldsymbol{\Pi}}$ enlargement phases, $\bar{\Omega}^2_p$ enlargement phases and $\boldsymbol{\lambda}$ enlargement phases. For the gap reduction phases, let $g_1 \leq g_2 \leq g_3 \leq ... \leq g_S$ be their indices in Algorithm Algorithm 3. Then by the construction of Algorithm Algorithm 3, the initial gap for each phase $\Delta_s := f(x_s) - \mathrm{lb}_s$ must satisfy $\Delta_{g_i} \geq \epsilon(\frac{3}{4})^{i-S}$. Thus it follows from Proposition 2.5 and the relations $\bar{M}_{\pi,s} \leq \sqrt{2}M_{\boldsymbol{\Pi}}$, $\bar{\Omega}_{p,s} \leq \sqrt{2}\Omega_P$ and $\boldsymbol{\lambda}_s \leq 4\ \forall s$ that the total number of iterations in the gap reduction phases is bounded by

$$\begin{aligned}
&\textstyle\sum_{i=1}^{S} \frac{8\sqrt{2}\Omega_X M_{\boldsymbol{T}}(\sqrt{2}M_{\boldsymbol{\Pi}})(1+2C_p\Omega_P)\sqrt{2}}{\frac{3}{4}^{i-S}\epsilon} \\
&\leq \textstyle\sum_{j=0}^{\infty}(\frac{3}{4})^j \frac{8\sqrt{2}\Omega_X M_{\boldsymbol{T}}(\sqrt{2}M_{\boldsymbol{\Pi}})(1+2C_p\Omega_P)\sqrt{2}}{\epsilon} \leq 8\sqrt{2}(8)\frac{\Omega_X M_{\boldsymbol{T}}(M_{\boldsymbol{\Pi}})(1+2C_p\Omega_P)}{\epsilon}.
\end{aligned}$$

For the $\bar{M}_{\boldsymbol{\Pi}}$ enlargement phases, let $s_1 \leq s_2 \leq ... \leq s_L$ be their indices in Algorithm Algorithm 3. Similar to the previous analysis, we use the geometric upper bound $\bar{M}_{\pi,s_i} \leq M_{\boldsymbol{\Pi}}(1/\sqrt{2})^{L-i}$ and uniform upper bounds 4, $\sqrt{2}\Omega_P$, $1/\epsilon$ for $\boldsymbol{\lambda}_s, \bar{\Omega}_{p,s}, 1/\Delta_s, \forall s$ to conclude

that the number iterations in the $\bar{M}_{\mathbf{\Pi}}$ enlargement phases is bounded by

$$\begin{aligned}&\textstyle\sum_{i=1}^{L} \frac{8\sqrt{2}\Omega_X M_{\boldsymbol{T}}(M_{\mathbf{\Pi}})(1+2C_p\Omega_P)\sqrt{2}}{\epsilon}(\frac{1}{\sqrt{2}})^{L-i}\\ &\leq \textstyle\sum_{j=0}^{\infty}(\frac{1}{\sqrt{2}})^{j}\frac{8\sqrt{2}\Omega_X M_{\boldsymbol{T}}(M_{\mathbf{\Pi}})(1+2C_p\Omega_P)\sqrt{2}}{\epsilon} \leq 16(\sqrt{2}+1)\frac{\Omega_X M_{\boldsymbol{T}}(M_{\mathbf{\Pi}})(1+2C_p\Omega_P)}{\epsilon}.\end{aligned}$$

Similarly, the number of iterations in the $\boldsymbol{\lambda}$ enlargement phases can be bounded by $16\sqrt{2}(\Omega_X M_{\boldsymbol{T}}(M_{\mathbf{\Pi}})(1+2C_p\Omega_P))/\epsilon$. Next, since there are at most $(\log_2(\Omega_P^2/\bar{\Omega}_{p,0}^2)+1)$ $\bar{\Omega}_p$-enlargement phases and the number of iterations in each phase is bounded uniformly by $16\sqrt{2}(\Omega_X M_{\boldsymbol{T}}(\sqrt{2}M_{\mathbf{\Pi}})(1+2C_p\Omega_P))/\epsilon$, the number of iterations in the $\bar{\Omega}_p$ enlargement phases should be bounded by

$$16(\log_2 \tfrac{\Omega_P^2}{\bar{\Omega}_{p,0}^2}\sqrt{2}+\sqrt{2})\tfrac{\Omega_X M_{\boldsymbol{T}}(M_{\mathbf{\Pi}})(1+2C_p\Omega_P)}{\epsilon}.$$

The desired iteration complexity bound follows by adding up these individual bounds. □

We remark here that the above iteration complexity bound has the same dependence on ϵ and K as that of the SD algorithm, i.e., $\mathcal{O}((1+C_p\Omega_P)/\epsilon)$, which does not seem to be improvable for solving general trilinear saddle point problems.

2.4 Adaptation For Kantorovich Ball

In the previous sections, we assumed P being simple such that the p proximal update is easy. However, this is not always the case; when P is the Kantorovich ball, a projection onto it is expensive. To avoid such an expensive computation, we propose to use the joint probability matrix projection instead. Because of the standalone P block in our reformulation (Equation 2.5), such an alternative update can be incorporated into the SD and SSL algorithms with only a change of stepsizes.

2.4.1 Kantorovich Ball and Joint Probability Matrix Proximal Update

Given K scenarios and a distance matrix $D \in R_+^{K\times K}$, i.e., $D_{i,j} = d(\xi_i, \xi_j)$, the δ-Kantorovich ball around the empirical distribution vector $\bar{p} = [\frac{1}{K}, \frac{1}{K} ... \frac{1}{K}]$ is

$$
\begin{aligned}
P_\delta := \big\{ p \in R_+^K \; s.t \;\; \exists\, H \in \mathbb{R}_+^{K\times K}, & \\
\bar{p}_i = \textstyle\sum_{j=1}^K H_{i,j}, \; \forall i, & \qquad \text{(source constraints)} \\
p_j = \textstyle\sum_{i=1}^K H_{i,j}, \; \forall j, & \qquad \text{(target constraints)} \\
\langle D, H \rangle_F \leq \delta \big\}, & \qquad \text{(transportation cost constraint)}
\end{aligned}
\tag{2.29}
$$

where $\langle D, H \rangle_F$ represents the Frobenius inner product, $\sum_{i=1}^K \sum_{j=1}^K D_{i,j} H_{i,j}$. Since every row and every column of the joint probability matrix H is constrained by a linear equality, the computation for the p proximal update, $\arg\max_{p\in P_\delta} \langle c, p \rangle + W(\bar{p}, p)$[5], is not separable across scenarios. In particular, when W is the Euclidean distance function, we have to solve a quadratic program (QP) with $\mathcal{O}(K^2)$ variables and $\mathcal{O}(K)$ linear constraints, and when W is the entropy distance function, we have to solve an exponential cone problem of the same size. In fact, even checking whether a given p is inside P_δ involves solving an expensive optimal transport problem.

Alternatively, we can remove the target constraints in (Equation 2.29) by representing p in terms of H and consider a proximal update of H. Moreover, the rows of H, i.e., $\{H_i\}$, would become separable after we dualize the single transportation cost constraint.

More specifically, to implement a separable H proximal update, we need a row separable Bregman distance function $\boldsymbol{W}$ for H constructed from the Bregman distance function W for p,

$$\boldsymbol{W}(\bar{H}, H) := \textstyle\sum_{i=1}^K W(\bar{H}_i, H_i).$$

[5] Notice that $\phi^* \equiv 0$ for Kantorovich ball ambiguity set.

Notice that $\boldsymbol{W}$ is 1-strongly convex with respect to $\|H\|_{\boldsymbol{W}} := \sqrt{\sum_{i=1}^{K} \|H_i\|_W^2}$. Moreover, by fixing $\bar{H}$ for the SSL algorithm and H_0 for the SD algorithm to be a uniform matrix with $1/K^2$ on every entry, the radii $\Omega_H^2 := \max_{H \in H_\delta} \boldsymbol{W}(\bar{H}, H)$ are bounded by $1/K$ for the Euclidean $\boldsymbol{W}$ and $\log(K)$ for the entropy $\boldsymbol{W}$. In the later analysis, to emphasize the relationship between Ω_H^2 and Ω_p^2, we define another constant $\widetilde{\Omega}_p^2$ which has approximately the same range as Ω_P^2:

a) When Euclidean $\boldsymbol{W}$ is used, set $\widetilde{\Omega}_p^2 := K\Omega_H^2$.

b) When entropy $\boldsymbol{W}$ is used, set $\widetilde{\Omega}_p^2 := \Omega_H^2$.

Now if $\mathcal{H}_\delta := \{H \geq 0 \mid \bar{p}_i = \sum_{j=1}^{K} H_{i,j} \; \forall i, \langle H, D \rangle_F \leq \delta\}$ denote the feasibility region of H, the proximal update for H using $\boldsymbol{W}$ and the consequent update for the probability vector q are:

$$\hat{H} := \underset{H \in \mathcal{H}_\delta}{\arg\max} \langle c, \textstyle\sum_{i=1}^{K} H_i \rangle - \mu_q \boldsymbol{W}(\bar{H}, H), \text{ and } \hat{q} := \textstyle\sum_{i=1}^{K} \hat{H}_i. \tag{2.30}$$

To differentiate it from the usual probability vector proximal update, we refer to (Equation 2.30) as the q-update.

By dualizing the $\langle H, D \rangle_F \leq \delta$ constraint, (Equation 2.30) becomes

$$\min_{\lambda \geq 0} \lambda\delta + \textstyle\sum_{i=1}^{K} \max_{H_i \geq 0, \langle H_i, e \rangle = \bar{p}_i} \langle c, H_i \rangle - \mu_q W(H_{t,i}, H_i) - \lambda \langle H_i, D_i \rangle. \tag{2.31}$$

Notice that for a fixed λ, the inner maximization problem consists of K independent simplex projection sub-problems, so it requires $\mathcal{O}(K^2)$ algebraic operations. If the bisection method is used to search for the optimal scalar λ^*, we can find an ϵ_λ-suboptimal $\hat{\lambda}$ and $\hat{H}$ in roughly $\mathcal{O}(K^2 \log(1/\epsilon_\lambda))$ algebraic operations, a significant improvement over the original QP and the exponential cone problem. As shown in Table Table 2.2, our numerical experiments written in MATLAB 2017a (with Mosek 8.1 as the QP/exponential cone solver)

and tested on a Macbook Pro with 2.40GHz Intel Core i5 processor and 8GB of 1600MHz DDR3 memory demonstrate the significant performance improvement for the q-update.

Table 2.2: Typical Projection Time for P_δ(Sec)

	Modified		Original	
#Scenarios	Entropy	Euclidean	Entropy	Euclidean
20	.0011	.019	0.180	0.140
100	.0028	.030	0.538	0.228
500	.047	.16	16.15	6.615
1000	.16	.97	93.38	37.54
5000	7.58	20.72	Out.Mem	Out.Mem

2.4.2 Modified SD Method

To use the more efficient q-update, we need to replace the update of p_t in Line 6 of Algorithm Algorithm 1 by

$$H_t := \arg\max_{H\in\mathcal{H}_\delta} \langle \textstyle\sum_{i=1}^k H_i, \widetilde{f}_{t,k}\rangle - \tau_q \boldsymbol{W}(H_{t-1}, H), \tag{2.32}$$

and use $q_t := \sum_{i=1}^K H_{t,i}$ in place of p_t in all other parts of the algorithm.

Now we modify the arguments in Section 2 to establish the convergence properties of the modified SD method and suggest some stepsize choices. Recall that the analysis in Section 2 revolves around solving the saddle point problem $\min_{x\in X}\max_{(p,\boldsymbol{\pi})\in P\times\boldsymbol{\Pi}} \mathcal{L}(x,p,\boldsymbol{\pi})$. Here we consider a modified saddle point problem associated with H instead, i.e.,

$$\min_{x\in x}\max_{(H,\boldsymbol{\pi})\in H_\delta\times\boldsymbol{\Pi}} \{\mathcal{L}(x,H,\boldsymbol{\pi}) := f_0(x) + \langle H, (\langle x,\boldsymbol{\pi}\rangle_{\boldsymbol{T}} - \boldsymbol{g}^*(\boldsymbol{\pi}))e^\intercal\rangle_F\}, \tag{2.33}$$

where $e^\intercal \in \mathbb{R}^K$ is a row vector of ones, $[1,1,\ldots,1]$. Similar to Proposition 2.1 and 2.2, the non-negativity of H implies the duality results between $\mathcal{L}$ and f. Then if the gap function Q in Definition 2.1 is constructed from $\mathcal{L}(x,H,\boldsymbol{\pi})$ in (Equation 2.33), we have $N(f(\bar{x}^t) - f(x^*)) \le \max_{u_H,u_{\boldsymbol{\pi}}} \sum_{t=1}^N Q\left[(x_t,H_t,\boldsymbol{\pi}_t),(x^*,u_H,u_{\boldsymbol{\pi}})\right]$. Next, similar to Proposition 2.3

and Theorem Theorem 2.1, the following convergence bounds of $\sum_{t=1}^{N} Q(z^t, u)$ and the function value $f(\bar{x}^t)$ hold.

Proposition 2.6 *If the non-negative stepsizes η, τ and σ satisfy*

$$\eta \geq \frac{C_p^2 M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2 K}{\tau_q} + \frac{M_{\boldsymbol{T}}^2}{\sigma}, \tag{2.34}$$

then for any $u \in Z := X \times H \times \boldsymbol{\Pi}$, we have

$$\textstyle\sum_{t=1}^{N} Q(z_t; u) \leq \sigma \langle u_p, U_k(\boldsymbol{\pi}_0, u_{\boldsymbol{\pi}}) \rangle + \tau_q \boldsymbol{W}(H_0, u_H) + \eta V(x_0, u_x). \tag{2.35}$$

Moreover, for $\bar{x}_N = \sum_{t=1}^{N} \frac{x^t}{N}$ we have

$$f(\bar{x}^t) - f(x^*) \leq \frac{\sigma \Omega_{\boldsymbol{\Pi}}^2 + \tau_q \Omega_H^2 + \eta \Omega_X^2}{N}, \tag{2.36}$$

where $\Omega_H^2 := \max_{H \in H_\delta} \boldsymbol{W}(H_0, H)$.

Proof: We only need to modify the inequalities, (paragraph 2.13) and (paragraph 2.14), related to the p-update. The modified Line 6 for the q-update in (Equation 2.32) implies that

$$\begin{aligned}
&Q_H(z_{t+1}; u) \\
&\leq \langle u_H - H_{t+1}, (\langle x_{t+1} - x_t, \boldsymbol{\pi}_{t+1} \rangle_{\boldsymbol{T}} - \langle x_t - x_{t-1}, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}}) e^{\mathsf{T}} \rangle_F + \\
&\qquad \tau_q(\boldsymbol{W}(H_t, u_H) - \boldsymbol{W}(H_{t+1}, u_H) - \boldsymbol{W}(H_t, H_{t+1})) \\
&\leq \tau_q(\boldsymbol{W}(H_t, u_H) - \boldsymbol{W}(H_{t+1}, u_H)) + [\langle u_H - H_{t+1}, \langle x_{t+1} - x_t, \boldsymbol{\pi}_{t+1} \rangle_{\boldsymbol{T}} e^{\mathsf{T}} \rangle_F \\
&\quad - \langle u_H - H_t, \langle x_t - x_{t-1}, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}} e^{\mathsf{T}} \rangle_F] + \epsilon_p(H_{t+1}),
\end{aligned} \tag{2.37}$$

where

$$\begin{aligned}\epsilon_p(H_{t+1}) &= \langle \textstyle\sum_{i=1}^K H_{t+1,i} - H_{t,i}, \langle x_t - x_{t-1}, \boldsymbol{\pi}_t \rangle_{\boldsymbol{T}} \rangle + \tau_q \boldsymbol{W}(H_t, H_{t+1}) \\ &\overset{(a)}{\leq} \sqrt{K} \left\| H_{t+1} - H_t \right\|_{\boldsymbol{W}} \left\| x_t - x_{t-1} \right\|_2 \left\| [\|T_1 \boldsymbol{\pi}_{t,1}\|_2, ..., \|T_K \boldsymbol{\pi}_{t,K}\|_2] \right\|_{W^*} \\ &\qquad + \tau_q \boldsymbol{W}(H_t, H_{t+1}) \\ &\leq \tfrac{1}{2\tau_q} (Cp M_{\boldsymbol{T}} M_{\boldsymbol{\Pi}})^2 K \left\| x_t - x_{t-1} \right\|_2^2 . \end{aligned} \tag{2.38}$$

Here, (a) follows from the algebraic fact $\left\| \sum_{i=1}^K H_{t,i} - H_{t+1,i} \right\|_W \leq \sqrt{K} \left\| H_t - H_{t+1} \right\|_{\boldsymbol{W}}$. The rest of the proof for (Equation 2.35) is the same as that for (Equation 2.10). Finally, (Equation 2.36) follows directly from (Equation 2.35) and the relation $N(f(\bar{x}^t) - f(x^*)) \leq \max_{u_H, u_{\boldsymbol{\pi}}} \sum_{t=1}^N Q(z^t, (x^*, u_H, u_{\boldsymbol{\pi}}))$. □

Observe that the stepsize requirement (Equation 2.34) and the convergence result (Equation 2.36) are exactly the same as their counterparts, (Equation 2.9) and (Theorem 2.1) in Section 2, except for some constant factor. So we can apply a change of variables to reuse the stepsize policy developed in Theorem Theorem 2.1. More specifically, if

a) $\widetilde{\tau} := \tau_q$ and $\widetilde{C_p} := \sqrt{K}$ for entropy $\boldsymbol{W}$;

b) $\widetilde{\tau} := \tau_q / K$ and $\widetilde{C_p} := \sqrt{K}$ for Euclidean $\boldsymbol{W}$,

then we have $\tau_q \Omega_H^2 = \widetilde{\tau} \widetilde{\Omega}_p^2$ and $C_p^2 k / \tau_q = \widetilde{C_p}^2 / \widetilde{\tau}$. So the following convergence result and stepsize choice follow immediately from Proposition 2.6 and Theorem Theorem 2.1.

Corollary 2.2 *For either the entropy $\boldsymbol{W}$ or the Euclidean $\boldsymbol{W}$, if the non-negative stepsizes satisfy $\eta \geq \frac{\widetilde{C_p}^2 M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2}{\widetilde{\tau}} + \frac{M_{\boldsymbol{T}}^2}{\sigma}$, then we have*

$$f(\bar{x}^t) - f(x^*) \leq \tfrac{\sigma \Omega_{\boldsymbol{\Pi}}^2 + \widetilde{\tau} \widetilde{\Omega}_p^2 + \eta \Omega_X^2}{N}.$$

In particular, if we choose $\sigma := M_{\boldsymbol{T}} \frac{\Omega_X}{\Omega_{\boldsymbol{\Pi}}}, \widetilde{\tau} := M_{\boldsymbol{T}} M_{\boldsymbol{\Pi}} \widetilde{C_p} \frac{\Omega_X}{\widetilde{\Omega}_p}$, and $\eta := M_{\boldsymbol{T}} M_{\boldsymbol{\Pi}} \widetilde{C_p} \frac{\widetilde{\Omega}_p}{\Omega_X} +$

$M_{\boldsymbol{T}}\frac{\Omega_{\boldsymbol{\Pi}}}{\Omega_X}$, *then*

$$f(\bar{x}^t) - f(x^*) \leq \frac{2\Omega_X M_{\boldsymbol{T}}}{N}(\Omega_{\boldsymbol{\Pi}} + \widetilde{C_p}\Omega_{\boldsymbol{\Pi}}\widetilde{\Omega}_p).$$

2.4.3 Modified SSL Algorithm

We replace the p-smoothing in (Equation 2.23) with a q-smoothing to obtain a modified smooth approximation $\widetilde{F}_{\mu_q,\mu_{\boldsymbol{\pi}}}(x)$ given by

$$\begin{aligned} g_{\mu_{\boldsymbol{\pi}},k}(x) &:= \max_{\pi_k \in \Pi(k)} \langle \pi_k, T_k x\rangle - g_k^*(\pi_k) - \mu_{\boldsymbol{\pi}} U(0,\pi_k), && (\boldsymbol{\pi} \text{ smoothing}) \\ \widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}(x) &:= \max_{H \in \mathcal{H}_\delta} \langle H, \mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x) e^\intercal\rangle_F - \mu_q \boldsymbol{W}(\bar{H}, H) \text{ for some } \bar{H} \in \mathcal{H}_\delta. && (q \text{ smoothing}) \end{aligned} \tag{2.39}$$

To establish the (α,β)-smooth approximation properties of $\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}(x)$, we need the following *domain smooth approximation* properties of the q-smoothing as a counterpart to Lemma 2.2.b).

Lemma 2.6 *As a function of* $\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)$, $\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}$ *is a* $(K\|I\|_{g,W}/\mu_q, \mu_q\Omega_H^2)$*-domain smooth approximation of* $F(\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)) := \max_{H\in\mathcal{H}_\delta}\langle H, \mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)e^\intercal\rangle_F$, *where* $\Omega_H^2 := \max_{H\in H_\delta} \boldsymbol{W}(\bar{H}, H)$.

Proof: Let $\mathbf{g}_1 := \mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x_1)$ and $\mathbf{g}_2 := \mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x_2)$ be given, and let $\hat{H}_1$ and $\hat{H}_2$ be the corresponding maximizers in (Equation 2.39). Then we have

$$\begin{aligned} \|\hat{q}_1 - \hat{q}_2\|_W^2 := \left\|\textstyle\sum_{i=1}^K \hat{H}_{1,i} - \sum_{i=1}^K \hat{H}_{2,i}\right\|_W^2 &\overset{(a)}{\leq} K\left\|\hat{H}_1 - \hat{H}_2\right\|_{\boldsymbol{W}}^2 \\ &\overset{(b)}{\leq} \tfrac{K}{\mu_q}\langle \hat{H}_1 - \hat{H}_2, (\mathbf{g}_1 - \mathbf{g}_2)e^\intercal\rangle_F \\ &= \tfrac{K}{\mu_q}\langle \hat{q}_1 - \hat{q}_2, \mathbf{g}_1 - \mathbf{g}_2\rangle \\ &\overset{(c)}{\leq} \tfrac{K}{\mu_q}\|\hat{q}_1 - \hat{q}_2\|_W \|I\|_{g,W} \|\mathbf{g}_1 - \mathbf{g}_2\|_g, \end{aligned}$$

where (a) follows from the algebraic fact that $\left\|\sum_{i=1}^K H_i\right\|_W^2 \leq (\sum_{i=1}^K \|H_i\|_W)^2 \leq K\|H\|_{\boldsymbol{W}}^2$, (b) follows from Lemma 2.1.b), and (c) follows from the definition of the op-

erator norm $\|I\|_{g,W}$. Dividing both sides by $\|\hat{q}_1 - \hat{q}_2\|_W$, we conclude that $\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}$ is a Lipschitz smooth function of $\mathbf{g}_{\mu_{\boldsymbol{\pi}}}(x)$ with constant $K\|I\|_{g,W}/\mu_q$. The approximation gap follows from the definition of $\Omega_H^2 := \max_{H\in H_\delta} \boldsymbol{W}(\bar{H}, H)$. □

The other parts needed to derive the smooth approximation properties of $\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}$, including the $\boldsymbol{\pi}$ smooth approximation properties and the Lipschitz continuity constant of $g_{\mu_{\boldsymbol{\pi}},k}$, are exactly the same as those in Section 3. Therefore Corollary 2.3 below follows as an immediate consequence of Lemma 2.6 and Proposition 2.4.

Corollary 2.3 *$\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}$ is a $(2C_p^2 K M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2/\mu_q + 2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}},\ \mu_q\Omega_H^2 + \mu_{\boldsymbol{\pi}}\Omega_{\boldsymbol{\pi}}^2)$-smooth approximation of F.*

Similar to the analysis of the modified SD algorithm, we can define a change of variables to simplify the above smooth approximation properties to the same form as that of $F_{\mu_{\boldsymbol{\pi}},\mu_p}$ in (Equation 2.23) such that the SSL algorithm can be applied readily. More specifically, we set

a) $\widetilde{C_p} := \sqrt{k}$ and $\widetilde{\mu}_p := \mu_q$ for entropy $\boldsymbol{W}$;

b) $\widetilde{C_p} := \sqrt{k}$ and $\widetilde{\mu}_p := \mu_q/k$ for Euclidean $\boldsymbol{W}$,

such that $KC_p^2/\mu_q = \widetilde{C_p}^2/\widetilde{\mu}_p$ and $\mu_q\Omega_H^2 = \widetilde{\mu}_p\widetilde{\Omega}_p^2$. Then $\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}(x)$ is a $(2\widetilde{C_p}^2 M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2/\widetilde{\mu}_p + 2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}},\ \widetilde{\mu}_p\widetilde{\Omega}_p^2 + \mu_{\boldsymbol{\pi}}\Omega_{\boldsymbol{\pi}}^2)$-smooth approximation of F[6], which is almost the same as $F_{\mu_{\boldsymbol{\pi}},\mu_p}$ being a $(2C_p^2 M_{\boldsymbol{T}}^2 M_{\boldsymbol{\Pi}}^2/\mu_p + 2M_{\boldsymbol{T}}^2/\mu_{\boldsymbol{\pi}},\ \mu_p\widetilde{\Omega}_p^2 + \mu_{\boldsymbol{\pi}}\Omega_{\boldsymbol{\pi}}^2)$ smooth approximation (shown in Proposition 2.4). Since both the optimal smooth ratio (Lemma 2.4) and the SSL algorithm's convergence analysis depend only on those smooth approximation properties, we conclude from Theorem Theorem 2.2 that the SSL algorithm applied to $\widetilde{F}_{\mu_{\boldsymbol{\pi}},\mu_q}(x)$ has an iteration complexity of $\mathcal{O}(\Omega_X M_{\boldsymbol{T}}\Omega_{\boldsymbol{\Pi}}(1 + 2\widetilde{C_p}\widetilde{\Omega}_p)/\epsilon)$.

[6] The *sequence smooth approximation* properties of $\widetilde{F}_{\mu_q,\mu_{\boldsymbol{\pi}}}$ can also be derived in a similar fashion.

Table 2.3: Theoretical Performance Comparison for $\mathcal{O}(1/\epsilon)$ Algorithms for Kantorovich P Problem

Algorithm	Iter Complexity	Computation Required for p Update
Separable PDHG[16]	$\mathcal{O}(K/\epsilon)$	Solving a $2K \times K^2$ QP
Euclidean SD/SSL	$\mathcal{O}(\sqrt{K}/\epsilon)$	Solving a $2K \times K^2$ QP
Entropy SD/SSL	$\mathcal{O}(\sqrt{\log K}/\epsilon)$	Solving a $2K \times K^2$ Conic Program
Modified Euclidean SD/SSL	$\mathcal{O}(\sqrt{K}/\epsilon)$	$\mathcal{O}(K^2 \log(1/\lambda_\epsilon))$ Algebraic Computations
Modified Entropy SD/SSL	$\mathcal{O}(\sqrt{K \log K}/\epsilon)$	$\mathcal{O}(K^2 \log(1/\lambda_\epsilon))$ Algebraic Computations

[1] We set λ_ϵ to the machine precision.

2.4.4 Iteration complexity

Both the modified SD and the modified SSL algorithms have the same iteration complexity bound of $\mathcal{O}((1 + \widetilde{C}_p\widetilde{\Omega}_p)/\epsilon)$, i.e., $\mathcal{O}(\sqrt{K}/\epsilon)$ for Euclidean $\boldsymbol{W}$ and $\mathcal{O}(\sqrt{K \log K}/\epsilon)$ for entropy $\boldsymbol{W}$. It is worth noting that the extra $\sqrt{K}$ factor for entropy $\boldsymbol{W}$ arises because the entropy radius scales sub-linearly, i.e. $\Omega_{\Delta/K} = \Omega_\Delta/\sqrt{K}$ while the Euclidean radius scales linearly, i.e. $\Omega_{\Delta/K} = \Omega_\Delta/K$. Although the iteration complexity for the entropy $\boldsymbol{W}$ is $\mathcal{O}(\sqrt{\log K})$ larger than that for the Euclidean $\boldsymbol{W}$, it is still preferable in practice because each entropy projection is cheaper (shown in Table Table 2.2).

2.5 Numerical Studies

We use distributionally robust two-stage linear programs to demonstrate the empirical performance of our algorithms .

Firstly, we test our algorithms by measuring their average performance on some randomly generated instances of a synthetic problem. We consider the following capacity

installation problem of an electricity utility company.

$$\begin{aligned} \min_{x \in R^n} \quad & c^\intercal x + \max_{p \in P} \textstyle\sum_{k=1}^{K} p_k g_k(T_k x) \\ & s.t.\ 0 \leq x_i \leq U \ \forall i, \\ \text{where} \quad & g_k(T_k x) := \min_{y_k \in \mathbb{R}^m_+} y_k^\intercal e_k \\ & \qquad s.t.\ R y_k \geq d_k - T_k x. \end{aligned} \tag{2.40}$$

The company is planning for the capacities of n technologies, $x \in \mathbb{R}^n$, to be installed for the coming year, with a unit cost vector $c \in R^n$. Moreover, being the sole provider of electricity in the region, it has to satisfy all demands in different periods of the year, $d \in \mathbb{R}^m$, using a combination of power generated by those installed capacities, with an availability factor of $T \in [0,1]^{m \times n}$, and power purchased from the outside grid at a unit cost of $e \in R^m$. The stochastic parameters $e, d,$ and T are unavailable at the planning time, so the company needs to find either a data-driven or risk-averse solution. In our experiments, we set $m = 20,\ n = 40$ and generate random instances in the following fashion.

1. c generated entry-wise IID from Unif[0.5 1].
2. $e_k \in R^m$ generated entry-wise IID from Unif[2,4].
3. $d_k \in R^m$ generated entry-wise IID from Unif[50, 100].
4. $T_k \in R^{m \times n}$ generated entry-wise IID from Unif[0.5 1].
5. $R = I_{m,m}$ is the simple complete recourse matrix.

Since R is the identity matrix, the scenario sub-problems are simple. They admit closed-form solutions for a given x, and each $\Pi(k)$ is a box in (Equation 2.5), so the $\boldsymbol{\pi}$-proximal update is also simple. However, in reformulation (Equation 2.4) of [16], the projection of (x_k, v_k) onto a non-smooth functional constrained feasibility set, $\{(x_k, v_k) | v_k \geq$

$g_k(T_k x_k), x_k \in X\}$, is more difficult as we have to solve a quadratic program (QP).

We also verify our results on a real-world test instance, namely the telecommunication network expansion problem with uncertain demands, SSN(50) in [34]. However, rather than the original expected total unfilled demand, we use some risk-averse function of the total unfilled demands as the objective function. Moreover, since we have to find the optimal network flow for each demand scenario, the scenario sub-problems are more difficult (R is not the simple identity matrix and $\Pi(k)$s are not boxes). We have to use LP and QP solvers for finding the optimal flows and computing the proximal updates, respectively.

2.5.1 Implementation Details

The numerical experiments are implemented in MATLAB 2017b with Mosek 8.1 as the optimization solver and are tested on an Alienware Desktop with 4.20GHz Intel Core i7 processor and 16GB of 2400MHz DDR4 memory. The x proximal updates and level set projection problems are solved using Mosek QP and the $\boldsymbol{\pi}$ proximal updates are solved using closed-form solutions for the synthetic problem and using Mosek QP for the network expansion problem. The p proximal updates are solved to machine accuracy using a binary search for the Lagrange multipliers associated with the coupling constraints in P. Their computation complexities are listed in Table Table 2.4.

Given a test instance, SD and PDHG are first fine-tuned by selecting among a few parameter choices the one achieving the smallest objective value in 100 iterations, $f(\bar{x}_{100})$(see Table Table 2.5 for these parameter choices). Next, the fine-tuned SD and PDHG and the parameter-free SSL are used to solve the instance. We record the number of iterations and the wall clock time required for these algorithms to achieve a relative optimality gap of $\epsilon \in \{10\%, 1\%, 0.1\%\}$, i.e., $f(x_t) - f_* \leq \epsilon f_*$. If the target accuracy is not reached after 2,000 seconds, we record both the number of iterations and the time as NA. To obtain an estimate of the true objective f^*, we use the parameter-free SSL algorithm and terminate only when the absolute gap between the lower and upper bound decreases to $1e^{-3}$.

Table 2.4: # Algebraic Operations Required for p proximal update

Distance Function W & Ambiguity Set	Constraints in P	# Algebraic Operations
Euclidean or Entropy W & Simplex	Box + One Linear	$O(K\log(1/\epsilon))$
Euclidean or Entropy W & AVaR	Box + One Linear	$O(K\log(1/\epsilon))$
Euclidean W & Modifed X^2	Box + One Linear + One Quadratic	$O(K\log^2(1/\epsilon))$
Modified Entropy or Euclidean $\boldsymbol{W}$ & Kantorovich	See Section 4	$O(K^2\log(1/\epsilon))$

Table 2.5: Parameter Selections

Algorithm	#	Step-sizes
PDHG	27	Over-relaxation parameter $\rho = 2$. $\eta \in \{10^{-3}, 10^{-2}, ..., 10^{2}, 10^{3}\}$. $\tau \in \{10^{-2}, 10^{-1}, 1\}/\eta$.
SD	16	$(\sigma, \tau, \eta) \in \{.1\bar{\sigma}, \bar{\sigma}\} \times \{.1\bar{\tau}, \bar{\tau}\} \times \{.1\bar{\eta}, \bar{\eta}\}$, where $\bar{\sigma}, \bar{\tau}$ are $\bar{\eta}$ calculated using the stepsize choice in Theorem 2.1 with conservative estimates of $\Omega_X, \Omega_P, \Omega_{\mathbf{\Pi}}$ and with $\{.1M_{\mathbf{\Pi}}, M_{\mathbf{\Pi}}\}$.
SSL	1	$\lambda_0 = 2^{-6}$, $\Omega_{p,0}^2 = W(\bar{p}, p_0)$, $M_{\pi,0}^2 = 2\max_k U_k(\bar{\boldsymbol{\pi}}, \boldsymbol{\pi}_0)$, where $\bar{p}, \boldsymbol{\pi}_0$ are the maximizers for x_0 in (Equation 2.5).

Note that the parameter estimation for the PDHG algorithm is difficult because both the primal and the dual feasibility region are unbounded.

2.5.2 Synthetic Problem: Probability Simplex Ambiguity Set

Notice that both SD and SSL have the same iteration complexity bound of $\mathcal{O}(1 + C_p\Omega_P/\epsilon)$. So to best illustrate how they scale with K, we conduct experiments on the probability simplex, which has the largest Ω_P. We make a few remarks about the result obtained in Table Table 2.6.

1. In general when the number of scenarios is large, both SSL and SD show significant improvement over PDHG in both computation time and iteration number. This is consistent with numerical experiments in [16], where a toy example (with $m = 3$ and $n = 2$) takes a significant amount of time even for a small number of scenarios, $K \leq 200$. Besides, SSL seems to outperform SD in finding solutions with high

accuracy.

2. Dependence on accuracy ϵ: both SD and PDHG match the theoretical complexity guarantee of $\mathcal{O}(1/\epsilon)$. In contrast, SSL enjoys a linear rate of convergence in practice, i.e., $\mathcal{O}(c^t)$, for some $c < 1$. Such a behavior is often observed for bundle level methods [32, 27, 23], but there is no rigorous theoretical explanation to the best of our knowledge.

3. Dependence on the number of scenarios K: both the computation time and the number of iterations required for PDHG increase quickly with K. However, the numbers of iterations required for entropy SD and SSL are nearly scenario independent. In fact, they seem to decrease slightly with increasing K. One plausible explanation is that more scenarios make f smoother, thus our accelerated algorithms might converge faster. However for Euclidean W, the number of iterations required for SD increases for large K while that for SSL stays the same.

4. Per iteration computation time: the per iteration computation time of PDHG is larger than that of SD and SSL. Moreover, the projection of x onto a level set in SSL is more expensive than the simple x-proximal update in SD. So when the number of scenarios is small ($K = 50 \sim 2000$) such that the level set projection dominates computation cost, SD seems to be faster than SSL for finding $1\%, 10\%$-suboptimal solutions, even though its numbers of iterations required are larger. However, when the number of scenarios is large and the $\boldsymbol{\pi}$ projection dominates the computation cost, SSL is faster.

2.5.3 Synthetic Problem: Risk-Averse AVaR Ambiguity Set

Given the empirical probability vector $\bar{p}$, we use the following reformulation in [35] of AVaR risk measure in our experiments.

$$\begin{aligned} AVaR_{1-\alpha}[g_1(x), ..., g_K(x)] = & \max_{p \geq 0} \langle p, \mathbf{g}(x) \rangle \\ & s.t \quad \sum_{k=1}^{K} p_k = 1 \\ & \quad 0 \leq p_k \leq \tfrac{1}{\alpha} \bar{p}_k. \end{aligned}$$

Observe that results shown in Table Table 2.7 are consistent with our findings in Subsection 5.2. In addition, both the iteration numbers and the computation times for all algorithms increase slightly in the 97.5% AVaR quantile case because of the larger Ω_P.

2.5.4 Synthetic Problem: Modified X^2 Ambiguity Set

The modified X^2 in [16] is defined as

$$\mathcal{P}_r := \left\{ p \in \mathbb{R}^k_+ : \left\| p - [\tfrac{1}{K}, ..., \tfrac{1}{K}] \right\|_2^2 \leq r, \textstyle\sum_{i=1}^{K} p_i = 1 \right\}.$$

Since the entropy projection onto a quadratically constrained $\mathcal{P}_r$ is difficult, we conduct experiments using only Euclidean W. The obtained result in Table Table 2.8 is consistent with our previous findings.

2.5.5 Synthetic Problem: Kantorovich Ball Ambiguity Set

We test the modified SD and modified SSL algorithms developed in Section 4 for the more challenging Kantorovich ball. The results are presented in Table Table 2.9. When $K = 200$, the computation time in each iteration due to the Euclidean p-update in PDHG is 0.2 second, while that for the entropy q-update in both SD and SSL algorithms is 0.02 second. When K is larger, the saving from the q-update is even more significant.

2.5.6 Real-world instance: SSN(50)

We conduct tests on the SSN(50) problem in [34] with all the above-mentioned ambiguity sets. The obtained results in Table Table 2.10 show that our SD and SSL algorithms significantly outperform the PDHG algorithm in computation time. Notice that the number of iterations of SD is comparable to that of PDHG and its saving derives mainly from easier $\boldsymbol{\pi}$ proximal updates (as compared to the joint (x, v_k, y_k) epigraph projection in PDHG). For problems with a large number of scenarios, we expect our scenario-independent algorithms to have a more significant advantage over PDHG in iteration number as well.

2.5.7 Comparison with the Benders Decomposition Algorithm

Finally, we compare the SSL algorithm with another frequently used cutting plane method, the Benders decomposition [36]. Our implementation considers the following master problem,

$$\min_{x\in X,\Psi,v_k} \; c^\intercal x + \Psi$$

$$s.t.\ \Psi \geq \phi(v_1, ..., v_K), \tag{2.41}$$

$$v_k \geq g_k(T_k x), \quad \forall k \in [K]. \tag{2.42}$$

In each iteration, the algorithm first computes $(x^t, \Psi_t, \{v_{k,t}\})$ by minimizing a master model, and then adds optimality cuts for the risk function in (Equation 2.41) and for the scenario cost functions in (Equation 2.42) to the master model.

Using 10 minutes as the cutoff time, we test these two algorithms on the synthetic problem with both 50 and 1000 scenarios and on the SSN(50) problem, and the results are listed in Table Table 2.11, Table 2.12, and Table 2.13. Clearly, the Benders decomposition algorithm outperforms our SSL algorithm when the number of scenarios is small and the scenario sub-problems are simple (Table Table 2.11). However, when either the number of scenarios is large (Table Table 2.12) or the scenario sub-problems are difficult (Table

Table 2.13), the SSL algorithm converges much faster. Additionally, notice that the Benders algorithm is unable to reach the 10% optimality gap for most SSN(50) tests. This is possibly because SSN is a known ill-conditioned problem (see, e.g., Section 9 of [37]).

To sum up, our experiments demonstrate that the proposed SD and SSL algorithms show significant performance improvement over the PDHG algorithm, especially for problems with a large number of scenarios. Between SSL and SD, SSL seems to be a better choice because it does not require any parameter tuning and it has a linear rate of convergence in practice. However, the SD algorithm is simpler to implement and may have some performance advantages over SSL for small problems with a low accuracy requirement. Moreover, the flexibility to choose a Bregman distance appropriate for the P geometry has a significant influence on the per iteration computation time, which is evident in the Kantorovich ball experiment.

Table 2.6: Numerical Experiment: Simplex Ambiguity Set
mean number of iterations and time(sec) to reach desired relative optimality gap

#Scenarios	Gap	PDHG	SD Euclid	SD Entropy	SSL Euclid	SSL Entropy
20	10%	333, 11.3s	268, 0.18s	200, 0.13s	74, 0.28s	74, 0.26s
	1%	3940, 146s	4060, 3.04s	2510, 1.57s	153, 0.60s	142, 0.58s
	0.1%	NA, NA	NA, NA	23600, 16.1s	260, 1.09s	246, 1.02s
200	10%	NA, NA	62, 0.35s	44, 0.27s	94, 1.12s	94, 1.14s
	1%	NA, NA	602, 3.26s	476, 2.65s	181, 2.43s	181, 2.45s
	0.1%	NA, NA	6010, 32.1s	4810, 26.0s	307, 4.32s	311, 4.50s
1000	10%	NA, NA	48, 1.02s	44, 0.94s	101, 7.49s	91, 6.75s
	1%	NA, NA	471, 10.4s	394, 8.62s	184, 15.0s	177, 14.3s
	0.1%	NA, NA	4710, 102s	3840, 83.3s	293, 24.7s	291, 24.7s
20000	10%	NA, NA	123, 46.4s	34, 14.6s	86, 64.0s	94, 76.1s
	1%	NA, NA	1210, 461s	220, 92.8s	168, 139s	178, 160s
	0.1%	NA, NA	NA, NA	2020, 821s	285, 248s	285, 274s

Table 2.7: Numerical Experiment: AVaR risk Measure

#Scenarios	Gap	PDHG	SD Entropy	SSL Entropy
95% AVaR quantile				
50	10%	1170, 104s	111, 0.19s	63, 0.59s
	1%	12100, 1040s	1340, 2.61s	115, 0.90s
	0.1%	NA, NA	13700, 23.6s	225, 1.98s
200	10%	NA, NA	30,0.15s	57, 0.49s
	1%	NA, NA	362, 2.08s	120, 1.22s
	0.1%	NA, NA	3570, 19.8s	233, 2.55s
1000	10%	NA ,NA	9, 0.15s	40, 0.94s
	1%	NA, NA	122, 2.70s	69, 1.94s
	0.1%	NA, NA	1180, 25.8s	120, 3.90s
97.5% AVaR quantile				
50	10%	1390, 108s	118, 0.16s	70, 0.32s
	1%	14400, 1140s	1410, 1.98s	154, 0.76s
	0.1%	NA, NA	14600, 21.3s	290, 1.49s
200	10%	4140, 1250s	34, 0.29s	60, 0.48s
	1%	NA, NA	410, 2.28s	139, 1.36s
	0.1%	NA, NA	4090, 21.6s	259, 2.76s
1000	10%	NA, NA	24, 0.51s	47, 1.11s
	1%	NA, NA	205, 4.34s	88, 2.60s
	0.1%	NA, NA	2030, 43.2s	176, 5.85s

Table 2.8: Numerical Experiment: Modified X^2 Ambiguity Set

#Scenarios	Gap	PDHG	SD Euclid	SSL Euclid
		$r = 0.01$		
50	10%	514, 40.2s	112, 0.18s	43, 0.22s
	1%	4120, 330s	1530, 1.96s	82, 0.60s
	0.1%	NA, NA	15200, 18.0s	157, 1.32s
200	10%	1040, 314s	26, 0.19s	40, 0.38s
	1%	4570, 1370s	268, 1.94s	70, 0.92s
	0.1%	NA, NA	2110, 11.8s	116, 1.77s
1000	10%	NA, NA	20, 0.47s	43, 1.50s
	1%	NA, NA	95, 2.64s	67, 2.97s
	0.1%	NA, NA	330, 8.80s	99, 4.89s
		$r = 0.1$		
50	10%	996, 76.1s	163, 0.19s	59, 0.37s
	1%	9970, 804s	2340, 2.69s	121, 0.94s
	0.1%	NA, NA	20400, 21.6s	245, 2.14s
200	10%	3390, 997s	68, 0.29s	62, 0.76s
	1%	NA, NA	860, 3.81s	128, 1.91s
	0.1%	NA, NA	8920, 39.0s	238, 3.81s
1000	10%	NA, NA	70, 1.48s	64, 2.83s
	1%	NA, NA	717, 15.3s	126, 6.37s
	0.1%	NA, NA	7160, 151s	230, 12.2s

Table 2.9: Numerical Experiment: Kantorovich Ball Ambiguity Set

#Scenarios	Gap	PDHG	Modified SD Entropy	Modified SSL Entropy
$\delta = 0.01 \times$ median distance in D				
50	10%	247, 27.0s	89, 0.18s	43, 0.33s
	1%	1560, 181s	846, 1.96s	70, 0.66s
	0.1%	NA, NA	8170, 19.2s	130, 1.41s
200	10%	498, 297s	16, 0.28s	38, 1.69s
	1%	1590, 883s	163, 3.05s	55, 2.90s
	0.1%	NA, NA	1180, 23.0s	90, 5.74s
1000	10%	NA, NA	16, 13.6s	36, 76.5s
	1%	NA, NA	131, 106s	50, 127s
	0.1%	NA, NA	602, 443s	73, 221s
$\delta = 0.1 \times$ median distance in D				
50	10%	420, 49.9s	93, 0.23s	55, 0.59s
	1%	2940, 363s	847, 2.10s	101, 1.30s
	0.1%	NA, NA	8270, 17.9s	175, 2.45s
200	10%	756, 557s	20, 0.43s	52, 3.79s
	1%	NA, NA	111, 2.62s	87, 7.24s
	0.1%	NA, NA	564, 13.0s	136, 12.1s
1000	10%	NA, NA	20, 20.6s	51, 171s
	1%	NA, NA	96, 96.0s	78, 297s
	0.1%	NA, NA	358, 334s	117, 485s

Table 2.10: Numerical Experiment: SSN(50) Data on Different Ambiguity Set

Ambiguity Set	Gap	PDHG[1]	SD Euclid	SD Entropy	SSL Euclid	SSL Entropy
Simplex	10%	420, 873s	360, 63.3s	309, 54.4s	142, 40.0s	107, 31.3s
	1%	NA, NA	996, 176s	631, 111s	195, 55.2s	187, 54.0s
AVaR	10%	1116, 2310s	366, 64.4s	233, 40.8s	69, 19.3s	135, 40.3s
95%	1%	NA, NA	1016, 179.5s	640, 113s	148, 41.6s	211, 61.8s
AVaR	10%	420, 880s	355, 62.9s	310, 55.1s	94, 26.5s	117, 36.0s
97.5%	1%	NA, NA	818, 146s	634, 113s	152, 43.0s	190, 57.0s
X^2	10%	167, 337s	61, 10.7s		144, 44.3s	
$r = 0.01$	1%	NA, NA	129, 22.8s		181, 55.0s	
X^2	10%	582, 1210s	96, 17.3s		128, 40.6s	
$r = 0.1$	1%	NA, NA	477, 84.9s		192, 60.3s	
Kantorovich[2]	10%	179, 373s	285, 119s	784, 137s	119, 73.1s	90, 25.0s
$\delta = 0.01$	1%	837, 1750s	768, 306s	2133, 382s	177, 110s	135, 43.1s
Kantorovich	10%	210, 445s	235, 107s	1105, 198s	108, 66.7s	104, 29.2s
$\delta = 0.1$	1%	NA, NA	533, 234s	3281, 595s	167, 105s	179, 51.0s

[1] Both SD and PDHG use the best iterate encountered (instead of the ergodic average) to measure the optimality gap for faster convergence.
[2] Modified Euclidean and entropy projections used for SD and SSL.

Table 2.11: Numerical Experiment: Comparison Between SSL and the Benders Method on Synthetic (50)

Ambiguity Set	Gap	SSL[1]	Benders
Simplex	10%	101, 0.48s	37, 0.36s
	1%	202, 1.08s	38, 0.38s
AVaR	10%	79, 0.35s	14, 0.08s
95%	1%	129, 0.64s	15, 0.09s
AVaR	10%	64, 0.27s	22, 0.17s
97.5%	1%	150, 0.77s	23, 0.19s
X^2	10%	46, 0.19s	7, 0.03s
$r = 0.01$	1%	96, 0.52s	10, 0.07s
X^2	10%	49, 0.22s	10, 0.07s
$r = 0.1$	1%	103, 0.57s	11, 0.08s
Kantorovich	10%	47, 0.31s	8, 0.08s
$\delta = 0.01$	1%	71, 0.67s	19, 0.22s
Kantorovich	10%	56, 0.49s	5, 0.04s
$\delta = 0.1$	1%	107, 1.10s	7, 0.07s

[1] SSL uses entropy projection for all but the X^2 ambiguity set.

Table 2.12: Numerical Experiment: Comparison Between SSL and the Benders Method on Synthetic (1000)

Ambiguity Set	Gap	SSL$^{(1)}$	Benders
Simplex	10%	103, 3.04s	101, 58.0s
	1%	184, 5.76s	106, 60.5s
AVaR	10%	48, 1.15s	12, 6.23s
95%	1%	88, 2.44s	19, 7.96s
AVaR	10%	48, 1.16s	17, 7.02s
97.5%	1%	92, 2.66s	26, 8.89s
X^2	10%	35, 0.79s	8, 5.06s
$r = 0.01$	1%	64, 2.04s	22, 8.36s
X^2	10%	49, 1.37s	26, 10.3s
$r = 0.1$	1%	95, 3.19s	37, 12.4s
Kantorovich	10%	42, 60.0s	5, 16.6s
$\delta = 0.01$	1%	52, 96.8s	6, 19.0s
Kantorovich	10%	50, 101s	5, 15.8s
$\delta = 0.1$	1%	76, 187s	16, 40.0s

Table 2.13: Numerical Experiment: Comparison Between SSL and the Benders Method on SSN(50)

Ambiguity Set	Gap	SSL$^{(1)}$	Benders
Simplex	10%	106, 30.4s	NA, NA
	1%	185, 52.4s	NA, NA
AVaR	10%	135, 39.4s	NA, NA
95%	1%	211, 60.4s	NA, NA
AVaR	10%	117, 35.1s	NA, NA
97.5%	1%	174, 51.1s	NA, NA
X^2	10%	144, 44.1s	975, 578s
$r = 0.01$	1%	179, 54.2s	NA, NA
X^2	10%	128, 39.7s	NA, NA
$r = 0.1$	1%	187, 57.4s	NA, NA
Kantorovich	10%	90, 24.7s	NA, NA
$\delta = 0.01$	1%	145, 40.2s	NA, NA
Kantorovich	10%	104, 29.0s	NA, NA
$\delta = 0.1$	1%	159, 44.7s	NA, NA

CHAPTER 3

NESTED STOCHASTIC COMPOSITE OPTIMIZATION

3.1 Introduction

3.1.1 Motivation

Composite optimization has attracted considerable interest for its applications in compressed sensing, image processing and machine learning. Many algorithmic studies have been focused on composite optimization of the form $\min_{x \in X} f(x) + g(x)$, where f is a smooth convex function and g is a nonsmooth function with certain special structures. Optimal first-order methods have been developed in [4, 5, 6, 7, 8] for solving these problems under different assumptions about g. In the stochastic setting, Lan [38, 39] presented an accelerated stochastic approximation method that can achieve the optimal iteration/sampling complexity when one only has access to stochastic (sub)gradients of the objective function (see, e.g., [40, 41, 42] for extensions).

The study of composite optimization naturally extends to more complex nested composition problems. For the deterministic setting, Lewis and Wright [43] developed a globally convergent algorithm for solving $\min_{x \in X} f(g(x))$, where the outer layer function f can be non-smooth, non-convex and extended-valued. Lan [27] also investigated the complexity of these problems when f is relatively simple. For the stochastic setting where both f and g are defined as expectation functions, Ermoliev [44, 45] developed two-timescale algorithms and established their asymptotic convergence. More recently, Wang et al. [46] studied finite-time convergence properties for these types of algorithms. Owing to its broad applications in reinforcement learning [46], meta-learning, and risk-averse optimization [47], nested stochastic composite optimization (NSCO) has become an important research topic.

A key challenge in NSCO is the lack of unbiased gradient estimators for the nested function. This issue can be illustrated with a simple two-layer problem, $\min_{x\in X}\{f(x) := f_1(f_2(x))\}$. Using the chain rule, the (sub)gradient of f can be expressed as $f'(x) = f_1'(f_2(x))f_2'(x)$[①] However, the evaluation point for f_1', which is $f_2(x)$, is inaccessible because only unbiased estimators $(f_2(x,\xi), f_2'(x,\xi))$ can be obtained from the stochastic first-order oracle. If the sub-gradient is evaluated instead at some $f_2(x,\xi)$ instead, the nested estimator is biased, except for an affine f_1, i.e.,

$$\mathbb{E}[f_1'(f_2(x,\xi))] \neq \nabla f_1'(f_2(x)). \tag{3.1}$$

Thus, it is not possible to obtain an unbiased estimator for $f_1'(f_2(x))$ or $f_1'(f_2(x))f_2'(x)$.

To resolve the aforementioned challenge, the current approach is to use gradient approximation, which controls the bias, $\mathbb{E}[f_1'(\bar{y}(\xi)) - f_1'(f_2(x))]$, by evaluating f_1' at some "close enough" estimate $\bar{y}(\xi)$ to $f_2(x)$. For example, the empirical average from $N = \mathcal{O}(1/\epsilon^2)$ samples, $\bar{y}(\xi) := \sum_{i=1}^N f_2(x,\xi_i)/N$, ensures $\mathbb{E}[\|\bar{y}(\xi) - f_2(x)\|] \leq \mathcal{O}(\epsilon)$. If f_1 is L_1-Lipschitz smooth, we have

$$\mathbb{E}[\|f_1'(\bar{y}(\xi)) - f_1'(f_2(x))\|] \leq \mathcal{O}(L_1\epsilon).$$

Therefore, using $f_1'(\bar{y}(\xi))f_2'(x)$ as the gradient proxy, the stochastic gradient descent method can find an ϵ-optimal solution, i.e., $\mathbb{E}[f(x^N) - f(x^*)] \leq \epsilon$, with $\mathcal{O}(1/\epsilon^4)$ calls to the stochastic oracle. Indeed, this simple idea provides the intuition to understand the two-timescale stochastic compositional gradient descent (SCGD) method in [48], and it achieves the same oracle complexity as the SCGD method.

Comparing this result to those for solving the simple one-layer stochastic optimization problem without the nested structure, we can observe two significant differences. First,

[①]For notation simplicity, especially when deriving the gradient for a nested composite function, We use the Jacobian matrix $f_i'(y) \in R^{n\times m}$ to represent the (sub)-gradient of $f_i : \mathbb{R}^m \to \mathbb{R}^n$.

the gradient-approximation type methods require the outer-layer function to be smooth. Second, their oracle complexities are worse than the $\mathcal{O}(1/\epsilon^2)$ complexity for the one-layer problem. These observations raise the question:

In the non-convex setting, recent developments provide a partially positive answer to this question. Here, the goal is to find a δ-stationary solution $\bar{x}$, i.e.,

$$\mathbb{E}[\|\nabla f(\bar{x})\|^2] \leq \delta. \tag{3.2}$$

For the one-layer smooth problem without nested composition, we know it takes $\mathcal{O}(1/\delta^2)$ [49] queries to the stochastic oracle to find an ϵ-stationary solution. When all the layer functions are smooth, Ghadimi et al. [50] developed an $\mathcal{O}(1/\delta^2)$ algorithm for the two-layer problem, and they extended it in [51, 47] to solve the multi-layer problem with the same oracle complexity of $\mathcal{O}(1/\delta^2)$. Additionally, under some stronger smoothness assumptions, a variance reduction algorithm proposed in [52] can improve the oracle complexity further to $\mathcal{O}(1/\delta^{1.5})$. However, if any layer function is non-smooth, [47] provided only an asymptotically convergent algorithm.

In the convex setting, the question remains open. Here, the goal is to find an ϵ-optimal solution $\bar{x}$,

$$\mathbb{E}[f(\bar{x}) - f(x^*)] \leq \epsilon. \tag{3.3}$$

For the one-layer problem without nested composition, Nemirovsky and Yudin [2] established the order-optimal oracle complexity of $\mathcal{O}(1/\epsilon^2)$, which can be further improved to $\mathcal{O}(1/\epsilon)$ when f is also strongly convex. However, the existing results for NSCO fail to match them. To the best of our knowledge, finite time convergence bounds for convex NSCO problems are available only when outer-layer functions are all smooth (see [48, 53, 46]). These works use the gradient approximation idea; they use moving averages to track function values of the inner layers and apply multi-timescale schemes to ensure their faster convergence. The complexities for these methods are summarized in Table Table 3.1 and

Table 3.1: Two-Layer Oracle Complexity

Problem	Type	In the Literature	SSD/SSDp
Convex	Outer Nonsmooth	N.A.	$O(1/\epsilon^2)$
	Outer Smooth	$\mathcal{O}(1/\epsilon^4)$ [48]	
	All Smooth	$\mathcal{O}(1/\epsilon^{3.5})$ [46]	
Strongly Convex	Outer Nonsmooth	N.A.	$\mathcal{O}(1/\epsilon^2)$
	Outer Smooth	$\mathcal{O}(1/\epsilon^{1.5})$ [48]	$\mathcal{O}(1/\epsilon\ )$
	All Smooth	$\mathcal{O}(1/\epsilon^{1.25})$ [46]	$\mathcal{O}(1/\epsilon\ )$

Table 3.2: k-Layer

Problem	Type	In the Literature	SSD/SSDp
Convex	Outer Nonsmooth	N.A.	$\mathcal{O}(1/\epsilon^2)$
	Outer Smooth	$\mathcal{O}(1/\epsilon^{2k})$[53]	
	All Smooth	$\mathcal{O}(1/\epsilon^{(7+k)/4})$[53]	
Strongly Convex	Outer Nonsmooth	N.A.	$\mathcal{O}(1/\epsilon^2)$
	Outer Smooth	N.A.	$\mathcal{O}(1/\epsilon\ )$
	All Smooth	$\mathcal{O}(1/\epsilon^{(3+k)/4})$[53]	$\mathcal{O}(1/\epsilon\ )$

Table 3.2. For the two-layer problem, if the innermost layer function is non-smooth, the oracle complexity is $\mathcal{O}(1/\epsilon^4)$. With an additional smoothness assumption for f_2, the oracle complexity can be improved to $\mathcal{O}(1/\epsilon^{3.5})$ and to $\mathcal{O}(1/\epsilon^{1.25})$ if the problem is also strongly convex. For the multi-layer problem, the complexities are even worse as they depend exponentially on the number of layers k. Additionally, it is worth noting that a direct application of the non-convex methods in [50, 51, 47] would lead to a suboptimal $O(1/\epsilon^4)$ complexity. This is because only a δ-stationary point with $\delta = \epsilon^2$ (Equation 3.2) can guarantee an $O(\epsilon)$-optimal solution.

3.1.2 Our Contributions

As shown in Table Table 3.1, if the outer-layer layer function f_1 is smooth, the oracle complexity for the two-layer problem has the same order as the one-layer problem, i.e., $\mathcal{O}(1/\epsilon^2)$ in the non-strongly convex case and $\mathcal{O}(1/\epsilon)$ in the strongly convex case. Moreover, if f_1 is either general non-smooth or structured non-smooth, the oracle complexity is $\mathcal{O}(1/\epsilon^2)$

in both the non-strongly convex and the strongly convex cases. Such a worse complexity in the strongly convex case is not improvable and shows that the nested structure can make a stochastic composite optimization problem intrinsically more difficult. Additionally, the complexities for the multi-layer problem in Table Table 3.2 have the same order as those for the two-layer problem. Thus, the result establishes the orders of problem complexity for convex NSCO optimization under different settings and refutes the belief that solving the NSCO problem is as simple as the usual stochastic optimization [54].

In this paper, we answer the question by developing efficient methods to achieve order-optimal oracle complexities under some mild assumptions. First, we study the two-layer nested problem with a (possibly strongly) convex regularization term $u(x)$ given by

$$\min_{x\in X}\{f(x) := f_1 \circ f_2(x) + u(x)\}, \tag{3.4}$$

where X is a compact and convex set with a finite radius, i.e., $\max_{x,\bar{x}\in X} \|x - \bar{x}\| \leq \mathcal{D}_X < \infty$. We impose the following *compositional convexity* assumption throughout the paper.

Assumption 2 *A nested function $f_1 \circ f_2(x)$ in* (Equation 3.4) *is said to satisfy the compositional convexity assumption if*

- *every layer function $f_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_{i-1}}$ is proper closed and convex.*
- *f_1 is component-wise non-decreasing if f_2 is not affine.*

The above monotonicity and the layerwise convexity assumption together form the classic sufficient condition for the nested function to be convex (e.g. see [1]). Compared to [46, 48, 53], Assumption 2 is stronger than their assumption of the nested function f being convex, but we do not require the outer-layer function f_1 to be smooth.

The layerwise convexity assumption implies a bi-conjugate reformulation [10] important for our development. Specifically, if f_i^* denotes the Fenchel conjugate to f_i and

$\Pi_i = \text{dom}(f_i^*) := \{\pi_i \in \mathbb{R}^{n_{i-1} \times n_i} : f_{i,j}^*(\pi_{i,j}) < \infty \; \forall j \in [n_{i-1}]\}$, we have

$$f_i(y_i) := \max_{\pi_i \in \Pi_i} \pi_i y_i - f_i^*(\pi_i). \tag{3.5}$$

This relationship motivates us to introduce a novel reformulation approach to the NSCO problem. We propose a $\min - \max - \max$ saddle-point reformulation by replacing every layer function with its bi-conjugate (see (Equation 3.5)), such that the dual variables π_1 and π_2 represent the (sub)gradients, f_1' and f_2'. The tight coupling between x and the dual variables is then relaxed, allowing for more flexibility in the choice of dual iterates (π_1^t, π_2^t). In our algorithm, we select an easier π_1^t for which an unbiased estimator is readily available, rather than $\pi_1^t = f_1'(f_2(x^t))$, for which the unbiased estimator is impossible. This allows us to update x^t with unbiased arguments to achieve order-optimal oracle complexities. Specifically, our development can be summarized into four steps.

First, we consider a simple two-layer problem with smooth layer functions (Equation 3.4). Linearizing both layer functions using the bi-conjugate in (Equation 3.5), we arrive at the "$\min_x - \max_{\pi_1} - \max_{\pi_2}$" saddle-point reformulation, similar to [55]. We extend the deterministic sequential dual (SD) method proposed in [55] to a stochastic sequential dual (SSD) method, which consists of computing prox-mappings for π_2 and π_1 and then computing the prox-mapping for x at each iteration. To implement the method, we observe the equivalence [56] between the dual prox-mappings and gradient evaluations when the prox functions are chosen to be Bregman distance functions generated by the Fenchel conjugates. This allows us to obtain unbiased estimators for the dual iterates (π_1^t, π_2^t) from querying the stochastic first-order oracle, enabling us to implement the SSD method in a primal form. The resulting oracle complexity is $\mathcal{O}(1/\epsilon^2)$ $(\mathcal{O}(1/\epsilon))$ under the non-strongly convex (strongly convex) setting . As illustrated in Table Table 3.1, these complexities are of the same order as those for solving the simpler one-layer problem, significantly improving upon the existing oracle complexities in the literature. In fact, even if the goal is

to find a δ-station point (Equation 3.2) under the non-strongly convex and smooth setting, combining our SSD method with the iterative regularization technique in [57] can improve the oracle complexity to $O(1/\delta)$. Moreover, in the deterministic setting, the SSD method is also optimal because its oracle complexities match those of the Nesterov accelerated gradient method [1] up to all problem parameters. Additionally, we extend the SSD method to solve a closely related problem with f_1 being structured non-smooth [31]. We modify the SSD method by selecting $\|\cdot\|^2$ as the prox-function when computing the π_1-prox mapping. The resulting method achieves an oracle complexity of $\mathcal{O}(1/\epsilon^2)$, which is order-optimal for the non-strongly convex case. Somewhat surprisingly, we demonstrate that this bound is also order-optimal for the strongly convex case by developing a matching lower complexity bound.

Second, we consider a more complicated two-layer problem where both f_1 and f_2 are general non-smooth functions. The non-smoothness of f_1 makes the problem particularly difficult. From the (sub)gradient approximation perspective, estimating $f_1'(f_2(x))$ is not possible because f_1' may not be continuous with respect to its argument. From the SSD perspective, computing the π_1-prox mapping with any strongly convex prox-function is difficult. To resolve the difficulty, we introduce a tri-conjugate reformulation to f_1, $f_1(y_1) = \min_{v_1} \max_{\pi_1} \langle \pi_1, y_1 - v_1 \rangle + f_1(v_1)$, where v_1 denotes an auxiliary primal variable. The prox-mappings for both v_1 and π_1 can be computed efficiently with $\|\cdot\|^2$ as the prox-function. Accordingly, we reformulate the nested problem as a $\min_x \min_{v_1} \max_{\pi_1} \max_{\pi_2}$ saddle point problem, and design a non-smooth stochastic sequential dual (nSSD) method, which computes prox-mappings for π_2 and π_1 and then prox-mappings for v_1 and x in each iteration. The nSSD method achieves an oracle complexity of $\mathcal{O}(1/\epsilon^2)$. To the best of our knowledge, it is the first method to achieve any finite-time convergence guarantee when the NSCO problem has a non-smooth outer-layer function. This complexity bound is optimal for the non-strongly convex case since it matches the order for solving the simpler one-layer problem. Moreover, we show that the complexity bound also has the optimal order in

the strongly convex case by developing a matching lower complexity bound.

Third, we extend our methods to the multi-layer setting. We consider a k-layer problem constructed from an arbitrary nested composition of smooth, structured non-smooth, and general non-smooth layer functions. In this setting, there exist complex stochastic dependencies among the sequentially sampled stochastic estimators. When constructing unbiased arguments for the prox-mappings, we resolve this issue by introducing a novel repeated sampling scheme to ensure conditional independence among these estimators. This leads to the multi-layer SSD and nSSD methods. As shown in Table Table 3.2, when all the outer-layer functions are smooth, the proposed method achieves the order-optimal oracle complexities of $\mathcal{O}(1/\epsilon^2)$ ($\mathcal{O}(1/\epsilon)$) in the non-strongly convex (strongly convex) setting, significantly improving upon the existing exponential dependence on k in [53]. If any outer-layer function is either structured non-smooth or general non-smooth, the multi-layer SSD/nSSD method achieves an order-optimal oracle complexity of $\mathcal{O}(1/\epsilon^2)$. This appears to be the first finite-time convergence guarantee for the multi-layer NSCO problem with any non-smooth outer-layer functions. Moreover, our method is easy to implement because the oracle complexity of $\mathcal{O}(1/\epsilon^2)$ can be achieved with parameter-free stepsizes.

Fourth, we demonstrate the effectiveness of our SSD and nSSD methods on two interesting examples: a) minimizing the mean-upper-semideviation risk of order 1 for a two-stage stochastic program, and b) minimizing the maximum loss associated with a system of stochastic composite functions. A direct application of our methods leads to order-optimal oracle complexities. The constant dependence of our methods can be further improved if we modify the SSD/nSSD method to take advantage of some separable problem structures. The resulting convergence bounds offer new insights into the difficulty of risk-averse optimization.

The rest of the paper is organized as follows. Section 2 introduces the SSD method for the smooth and structured-nonsmooth two-layer problem. Section 3 introduces the nSSD method for the general non-smooth two-layer problem. Section 4 extends these methods to

the multi-layer setting and Section 5 provides two concrete applications. Some concluding remarks are made in Section 6.

3.1.3 Notations & Assumptions

The following notations and assumptions are used throughout the paper.

- The feasible region X is convex and compact with $\mathcal{D}_X := \max_{x_1,x_2\in X} \|x_1 - x_2\|_2 < \infty$. We assume the solution set $X^* := \arg\min_{x\in X} f(x)$ to be non-empty and use $x^* \in X^*$ to denote an arbitrary optimal solution.

- Every layer function $f_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_{i-1}}$ is closed, convex and proper. We use the notation $\nabla f_i(y_i) \in \mathbb{R}^{n_{i-1}} \times \mathbb{R}^{n_i}$ ($f_i'(y_i) \in \mathbb{R}^{n_{i-1}} \times \mathbb{R}^{n_i}$) to denote the Jocabian (sub-gradient) matrix.

- There exists a stochastic first-order oracle $\mathcal{SO}_i$ associated with every layer function f_i. When queried at some $y_i \in Y_i$, the $\mathcal{SO}_i$ returns a pair of unbiased estimators, $(f_i(y_i,\xi_i), f_i'(y_i,\xi_i))$ for $(f_i(y_i), f_i'(y_i))$. Results returned by different queries to $\mathcal{SO}_i$ are independent, and all $\mathcal{SO}_i$'s are independent.

- The Fenchel conjugate of a convex function $g(x)$ is defined as $g^*(\pi) := \max_{x\in\mathbb{R}^n} \langle x,\pi\rangle - g(x)$. The Bregman distance function (or prox-function) associated with a convex function g is defined as $D_g(x;y) = g(x) - g(y) - \langle g'(y), x-y\rangle$. For a m-dimensional vector valued function $g(x)$, its dual variable $\pi := [\pi_1;\pi_2;\ldots;\pi_m]$ is an $m\times n$ matrix, and g^* and D_{g^*} are m-dimensional vector functions. Specifically, the j^{th} entries of g^* and D_{g^*} are defined according to $g_j^*(\pi_j) := \max_{x\in\mathbb{R}^n} \langle x,\pi_j\rangle - g_j(x)$ and $D_{g^*,j}(\pi;\bar\pi) = g_j^*(\pi_j) - g_j^*(\bar\pi_j) - \langle g_j^{*\prime}(\bar\pi_j), \pi_j - \bar\pi_j\rangle$.

- We use the term prox-mapping of h to refer to the following type of computation:

$$y^t \leftarrow \arg\min_{y\in Y} \langle y, g\rangle + h(y) + \eta V(y;\bar y), \tag{3.6}$$

where $Y \subset \mathbb{R}^n$ is a closed and convex domain, h is a convex function, and $V(\cdot;\cdot)$ is some Bregman distance function. When h is an m-dimensional vector function and $y \subset \mathbb{R}^{m \times n}$ is a matrix, we use the term prox-mapping of h to refer to computing the prox-mapping for every row of y, i.e., $y^t := [y_1^t; \ldots; y_m^t]$ with

$$y_i^t \leftarrow \arg\min_{y_i \in Y_i} \langle y_i, g \rangle + h_i(y_i) + \eta V_i(y_i; \bar{y}_i) \; \forall i \in [m].$$

In both cases, we call g the argument, $\bar{y}$ the prox center, V the prox-function, and η the stepsize parameter. Moreover, we also use the term prox-mapping for y to denote the prox-mapping of h if the associated function $h(y)$ is clear from context, e.g., prox-mapping for x refers to the prox-mapping of $u(x)$.

- $\|\cdot\|$ denotes the l_2 (operator) norm unless specified otherwise.
- We use the notation $[\cdot]^+$ to denote projection onto the positive orthant, i.e., $[x]^+ := \max\{x, 0\}$, and the notation $[\cdot]_Y^+$ to denote the projection onto set Y.

3.2 Smooth and Structured Non-smooth Two Layer Problems

In this section, we present the SSD method for the two-layer problem in (Equation 3.4). We assume f_2, the inner layer function, to be smooth and f_1, the outer layer function, to be either smooth or structured non-smooth. Specifically, for a function $g(y)$ defined on Y, we call it smooth if its gradient $\nabla g(y)$ is Lipschitz continuous, i.e., $\|\nabla g(y) - \nabla g(\bar{y})\| \leq L \|y - \bar{y}\| \;\; \forall y, \bar{y} \in Y$. We call it structured non-smooth if there is some convex closed set Π and convex closed and proper function g^* such that

$$g(y) = \max_{\pi \in \Pi} \langle \pi, y \rangle - g^*(\pi), \forall y \in Y. \tag{3.7}$$

We assume the prox-mapping of g^* with $\|\cdot\|_2^2$ as the prox-function to be efficiently computable (see subsection 3.1.3). Notice such a definition differs from the one proposed by

Nesterov [31] where the inner product contains a general linear operator A, i.e., $\langle \pi, Ay \rangle$. However, our definition is not restrictive in NSCO because Ay can be regarded as the output from an inner linear layer function $A(\cdot)$.

The coming subsections are organized as follows. subsection 3.2.1 introduces the SSD method, followed by its convergence results in subsection 3.2.2. Next, subsection 3.2.3 presents a lower complexity bound for the structured non-smooth problem and subsection 3.2.4 presents the detailed convergence analysis.

3.2.1 The SSD Method

As suggested in subsection 3.1.2, the development of the SSD method is inspired by a $\min - \max - \max$ reformulation of (Equation 3.4) given by

$$\min_{x \in X} \max_{\pi_1 \in \Pi_1} \max_{\pi_2 \in \Pi_2} \{\mathcal{L}(x; \pi_1, \pi_2) := \mathcal{L}_1(x; \pi_1, \pi_2) + u(x)\}, \tag{3.8}$$

where Π_1 and Π_2 are respective domains of f_1^* and f_2^* (see (Equation 3.5)) and the compositional Lagrangian functions are defined according to

$$\mathcal{L}_2(x; \pi_2) = \pi_2 x - f_2^*(\pi_2), \text{ and } \mathcal{L}_1(x; \pi_1, \pi_2) = \pi_1 \mathcal{L}_2(x; \pi_2) - f_1^*(\pi_1). \tag{3.9}$$

For simplicity, we will use the notation $z := (x; \pi_1, \pi_2)$ and $Z := X \times \Pi_1 \times \Pi_2$ for the rest of the section. Since $\mathcal{L}_1(x; \pi_1, \pi_2)$ can be interpreted as the nested composition of a lower linear approximation to f_2, specified by π_2 and $f_2^*(\pi_2)$, and a lower linear approximation to f_1, specified by π_1 and $f_1^*(\pi_1)$, a certain duality relationship holds between $\mathcal{L}$ (c.f. (Equation 3.8)) and the original problem (Equation 3.4).

Lemma 3.1 *Let f and $\mathcal{L}$ be defined in* (Equation 3.4) *and* (Equation 3.8)*. Then*

a) Weak duality: $f(x) \geq \mathcal{L}(x; \pi_1, \pi_2) \ \ \forall (\pi_1, \pi_2) \in \Pi_1 \times \Pi_2, \forall x \in X$.

b) Strong duality: for a given $x \in X$, $f(x) = \mathcal{L}(x; \hat{\pi}_1, \hat{\pi}_2)$ for any $\hat{\pi}_2 \in \partial f_2(x)$ and

$\hat{\pi}_1 \in \partial f_1(f_2(x))$.

c) There exists a pair (π_1^*, π_2^*) *such that* $z^* := (x^*; \pi_1^*, \pi_2^*)$ *is a saddle point, i.e.,*

$$\mathcal{L}(x^*; \pi_1, \pi_2) \leq \mathcal{L}(x^*; \pi_1^*, \pi_2^*) \leq \mathcal{L}(x; \pi_1^*, \pi_2^*) \; \forall (x, \pi_1, \pi_2) \in Z.$$

d) For any $(x; \pi_1, \pi_2) \in Z$*, an upper bound on the optimality gap of* x *is given by:*

$$f(x) - f(x^*) \leq \max_{\bar{\pi}_1 \in \Pi_1, \bar{\pi}_2 \in \Pi_2} \mathcal{L}(x; \bar{\pi}_1, \bar{\pi}_2) - \mathcal{L}(x^*; \pi_1, \pi_2) \tag{3.10}$$

Proof: First, for Part b), let $x \in X$ be given. It follows from $\hat{\pi}_1 \in \partial f_1(f_2(x))$ and $\hat{\pi}_2 \in \partial f_2(x)$ (see Theorem 23.5 in [58]) that

$$f_2(x) = \mathcal{L}_2(x; \hat{\pi}_2), \text{ and } f_1(f_2(x)) = \mathcal{L}_1(x; \hat{\pi}_1, \hat{\pi}_2).$$

Thus $f(x) = \mathcal{L}_1(x; \hat{\pi}_1, \hat{\pi}_2) + u(x)$. Regarding Part a), for any $x \in X$, the following decomposition is valid for any $(\pi_1, \pi_2) \in \Pi_1 \times \Pi_2$:

$$\begin{aligned}\mathcal{L}(x; \hat{\pi}_1, \hat{\pi}_2) - \mathcal{L}(x; \pi_1, \pi_2) &= \mathcal{L}(x; \hat{\pi}_1, \hat{\pi}_2) - \mathcal{L}(x; \pi_1, \hat{\pi}_2) + \mathcal{L}(x; \pi_1, \hat{\pi}_2) - \mathcal{L}(x; \pi_1, \pi_2) \\ &= \underbrace{\mathcal{L}_1(x; \hat{\pi}_1, \hat{\pi}_2) - \mathcal{L}_1(x; \pi_1, \hat{\pi}_2)}_{A} + \underbrace{\pi_1(\mathcal{L}_2(x; \hat{\pi}_2) - \mathcal{L}_2(x; \pi_2))}_{B},\end{aligned}$$

where $A \geq 0$ because $\hat{\pi}_1 \in \arg\max_{\pi_1 \in \Pi_1} \{\pi_1 f_2(x) - f_1^*(\pi_1) \equiv \mathcal{L}_1(\pi_1, \hat{\pi}_2; x)\}$. If f_2 is affine, Π_2 is a singleton set such that $B = 0$. Otherwise, Π_1 is non-negative (Assumption 2). $\mathcal{L}_2(\hat{\pi}_2; x) - \mathcal{L}_2(\pi_2; x) \geq 0$ implies $B \geq 0$. Therefore we get $f(x) = \mathcal{L}(x; \hat{\pi}_1, \hat{\pi}_2) \geq \mathcal{L}(x; \pi_1, \pi_2)$.

As for Part c), the first-order optimality condition implies that there exist some $\pi_1^* \in \partial f_1(f_2(x^*))$, $\pi_2^* \in \partial f_2(x^*)$ and $u' \in \partial u(x^*)$ such that $\pi_1^* \pi_2^* (x - x^*) + \langle u', x - x^* \rangle \geq 0 \, \forall x \in$

X. Thus, with $D_u(x;x^*) := u(x) - u(x^*) - \langle u', x - x^*\rangle \geq 0$, we get

$$\mathcal{L}(x;\pi_1^*,\pi_2^*) - \mathcal{L}(x^*;\pi_1^*,\pi_2^*) = \pi_1^*\pi_2^*(x-x^*) + \langle u', x-x^*\rangle + D_u(x;x^*) \geq 0.$$

The relation $\mathcal{L}(x^*;\pi_1,\pi_2) \leq \mathcal{L}(x^*;\pi_1^*,\pi_2^*)$ is a direct consequence of the strong duality. So $z^* := (x^*;\pi_1^*,\pi_2^*)$ is a saddle point.

Part d) is a direct consequence of Parts a) and b). □

Now we introduce a Q-gap function which is often-used for saddle point problems (e.g. see [28] and [55]). For a point $z^t := (x^t;\pi_1^t,\pi_2^t) \in Z$, the Q-gap function, defined with respect to some reference point $z \in Z$, is given by

$$Q(z^t,z) := \mathcal{L}(x^t;\pi_1,\pi_2) - \mathcal{L}(x;\pi_1^t,\pi_2^t). \tag{3.11}$$

The Q-gap function plays a central role in our development for two reasons. First, it provides an upper bound to the optimality gap (see Lemma 4.1.d)), so reducing the Q-gap also reduces the optimality gap. Second, the Q-gap function admits a decomposition conducive to algorithm design. Specifically, the development of the SSD method is motivated by the following decomposition:

$$Q(z^t,z) = Q_2(z^t,z) + Q_1(z^t,z) + Q_0(z^t,z), \tag{3.12}$$

$$Q_2(z^t,z) := \mathcal{L}(x^t;\pi_1,\pi_2) - \mathcal{L}(x^t;\pi_1,\pi_2^t) = \pi_1[\pi_2 x^t - f_2^*(\pi_2)]\boxed{-\pi_1[\pi_2^t x^t - f_2^*(\pi_2^t)]}, \tag{3.13}$$

$$Q_1(z^t,z) := \mathcal{L}(x^t;\pi_1,\pi_2^t) - \mathcal{L}(x^t;\pi_1^t,\pi_2^t) = \pi_1\mathcal{L}_2(\pi_2^t;x^t) - f_1^*(\pi_1)$$
$$\boxed{-[\pi_1^t\mathcal{L}_2(\pi_2^t;x^t) - f_1^*(\pi_1^t)]}, \tag{3.14}$$

$$Q_0(z^t,z) := \mathcal{L}(x^t;\pi_1^t,\pi_2^t) - \mathcal{L}(x;\pi_1^t,\pi_2^t) = \boxed{\pi_1^t\pi_2^t x^t + u(x^t)} - (\pi_1^t\pi_2^t x + u(x)), \tag{3.15}$$

where Q_2, Q_1, and Q_0 relate to the optimality of π_2^t, π_1^t and x^t, respectively. The conceptual sequential dual (SD) method, originally proposed in [55], performs sequential prox-mappings to generate π_2^t, π_1^t and x^t to reduce Q_2, Q_1, and Q_0, respectively, that is, the boxed terms in (Equation 3.13), (Equation 3.14) and (Equation 3.15). With $(x^0; \pi_1^0, \pi_2^0) \in Z$, the t^{th} iteration of the SD method is given by

$$\begin{aligned}
&\pi_2^t \leftarrow \arg\max_{\pi_2 \in \Pi_2} \pi_2 \tilde{x}^t - f_2^*(\pi_2) - \tau_{2,t} U_2(\pi_2; \pi_2^{t-1}), \text{ where } \tilde{x}^t := x^{t-1} + \theta_t (x^{t-1} - x^{t-2}); \\
&\pi_1^t \leftarrow \arg\max_{\pi_1 \in \Pi_1} \pi_1 \tilde{y}_1^t - f_1^*(\pi_1) - \tau_{1,t} U_1(\pi_1; \pi_1^{t-1}), \\
&\qquad \text{where } \tilde{y}_1^t := \mathcal{L}_2(x^{t-1}; \pi_2^t) + \theta_t \pi_2^{t-1} (x^{t-1} - x^{t-2}); \\
&x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t x + u(x) + \tfrac{\eta_t}{2} \left\| x - x^{t-1} \right\|^2, \ \tilde{y}_0^t := \pi_1^t \pi_2^t.
\end{aligned} \tag{3.16}$$

In the above listing, U_2 and U_1 denote general prox-functions and scalars $\tau_{2,t}$, $\tau_{1,t}$ and η_t represent non-negative stepsizes. Since x^t is yet available, the arguments to the π_2^t and π_1^t prox-mappings are extrapolated points, i.e., $\tilde{x}^t$ and $\tilde{y}_1^t$ to predict x^t and $\mathcal{L}_2(x^t; \pi_2^t)$.

The deterministic SD method has been shown to achieve optimal complexities under various settings [59, 55, 60]. To adapt it to the NSCO setting, we propose to replace the deterministic arguments, $\tilde{y}_1^t$ and $\tilde{y}_0^t$, with some stochastic estimators, $\tilde{y}_1^t(\xi)$ and $\tilde{y}_0^t(\xi)$. This leads to a stochastic version of the SD method, namely, the SSD method. Initialized to $(x^0; \pi_1^0, \pi_2^0) \in Z$ and $x^{-1} = x^0$, its t^{th} iteration is given by

$$\begin{aligned}
&\pi_2^t \leftarrow \arg\max_{\pi_2 \in \Pi_2} \pi_2 \tilde{x}^t - f_2^*(\pi_2) - \tau_{2,t} U_2(\pi_2; \pi_2^{t-1}), \\
&\qquad \text{where where } \tilde{x}^t := x^{t-1} + \theta_t (x^{t-1} - x^{t-2}); \\
&\pi_1^t \leftarrow \arg\max_{\pi_1 \in \Pi_1} \pi_1 \tilde{y}_1^t(\xi) - f_1^*(\pi_1) - \tau_{1,t} U_1(\pi_1; \pi_1^{t-1}), \\
&\qquad \text{where } \tilde{y}_1^t := \mathcal{L}_2(x^{t-1}; \pi_2^t) + \theta_t \pi_2^{t-1} (x^{t-1} - x^{t-2}); \\
&x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t(\xi) x + u(x) + \tfrac{\eta_t}{2} \left\| x - x^{t-1} \right\|^2, \text{where } \tilde{y}_0^t := \pi_1^t \pi_2^t.
\end{aligned} \tag{3.17}$$

Next, we provide the concrete implementation to the above prox-mappings and the construction of the unbiased estimators, i.e., $\tilde{y}_1^t(\xi)$ and $\tilde{y}_0^t(\xi)$ for $\tilde{y}_1^t$ and $\tilde{y}_0^t$. First, let us consider the two-layer smooth problem where f_1 is smooth. A key challenge appears to be that the conjugate functions, f_i^*'s, are not explicitly known. However, if $D_{f_i^*}$, the Bregman distance function generated by f_i^*, is selected as the prox-function U_i, there exists some primal equivalences to the prox-mappings of f_i^*'s such that the steps in (Equation 3.17) simplify to gradient evaluations. Towards that end, an important result is the following relation (see Theorem 23.5 in [58]) between a closed convex and proper function g and its Fenchel conjugate g^*,

$$\pi \in \arg\max_{\bar{\pi} \in \text{dom}(g^*)} \bar{\pi}\underline{y} - g^*(\bar{\pi}) \Longleftrightarrow \pi \in \partial g(\underline{y}) \Longleftrightarrow g^*(\pi) = \pi\underline{y} - g(\underline{y}). \tag{3.18}$$

The first relation in (Equation 3.18) implies the equivalence of the prox-mapping of f_i^* to a gradient evaluation at some averaged point. Specifically, the following lemma is an extension of a similar result in [28, 56] to vector-valued functions and its proof is deferred to the Appendix.

Lemma 3.2 *Given a convex, closed and proper vector function g, if π^{t-1} is associated with some primal point $\underline{y}^{t-1}$, i.e., $\pi^{t-1} = g'(\underline{y}^{t-1}) \in \partial g(\underline{y}^{t-1})$, then the prox-mapping for π^t, i.e., $\pi^t \in \arg\min_{\pi \in dom(g^*)} -\pi\tilde{y}^t + g^*(\pi) + \tau_t D_{g^*}(\pi; \pi^{t-1})$, is equivalent to*

$$\pi^t = g'(\underline{y}^t) \in \partial g(\underline{y}^t) \text{ with } \underline{y}^t := (\tilde{y}^t + \tau_t \underline{y}^{t-1})/(1+\tau_t). \tag{3.19}$$

Thus, initialized to $\pi_1^0 = \nabla f_1(\underline{y}_1^0)$ and $\pi_2^0 = \nabla f_2(\underline{y}_2^0)$, each iteration of the SSD algorithm

in its primal form is given by

$$
\begin{aligned}
&\pi_2^t \leftarrow \nabla f_2(\underline{y}_2^t), \text{ where } \underline{y}_2^t := (\tilde{x}^t + \tau_{2,t}\underline{y}_2^{t-1})/(1+\tau_{2,t}),\ \tilde{x}^t := x^{t-1} + \theta_t(x^{t-1} - x^{t-2});\\
&\tilde{y}_1^t := f_2(\underline{y}_2^t) + \nabla f_2(\underline{y}_2^t)(x^{t-1} - \underline{y}_2^t) + \theta_t \nabla f_2(\underline{y}_2^{t-1})(x^{t-1} - x^{t-2});\\
&\pi_1^t \leftarrow \nabla f_1(\underline{y}_1^t), \text{ where } \underline{y}_1^t := (\tilde{y}_1^t(\xi) + \tau_{1,t}\underline{y}_2^{t-1})/(1+\tau_{1,t});\\
&x^t \leftarrow \arg\min_{x\in X} \tilde{y}_0^t(\xi)x + u(x) + \tfrac{\eta_t}{2}\left\|x - x^{t-1}\right\|^2 \text{ where } \tilde{y}_0^t := \nabla f_1(\underline{y}_1^t)\nabla f_2(\underline{y}_2^t).
\end{aligned}
\tag{3.20}
$$

Notice that the second line in (Equation 3.20) utilizes the fact $f_2(\underline{y}_2^t) + \nabla f_2(\underline{y}_2^t)(x^{t-1} - \underline{y}_2^t) = \mathcal{L}_2(x^{t-1}; \pi_2^t)$. This is valid because $\pi_2^t = \nabla f_2(\underline{y}_2^t)$ and $f_2^*(\pi_2^t) = \nabla f_2(\underline{y}_2^t)\underline{y}_2^t - f_2(\underline{y}_2^t)$ (the second equivalence in (Equation 3.18)).

The construction for $\tilde{y}_1^t(\xi)$ and $\tilde{y}_2^t(\xi)$ are provided in the concrete primal-form SSD method shown in Algorithm 4. We make two observations. First, thanks to its primal representation in Line 2 of (Equation 3.20), an unbiased estimator to $\tilde{y}_1^t$, $\tilde{y}_1^t(\xi)$, can be constructed in Line 4 of Algorithm 4 with stochastic estimators from $\mathcal{SO}_2$, i.e., $f_2(\underline{y}_2^t, \xi_2^1)$, $\nabla f_2(\underline{y}_2^t, \xi_2^1)$, $f_2(\underline{y}_2^{t-1}, \hat{\xi}_2)$. Notice that the estimator for $f_2^*(\pi_2^t)$ (as a part of $\mathcal{L}_2(x^{t-1}; \pi_2^t)$ in line 2 of (Equation 3.17)) is obtained without explicitly knowing either f_2^* nor π_2^t. Second, the dependence between the sequentially generated iterates requires us to make three independent queries to $\mathcal{SO}_2$ (e.g. Line 3 of Algorithm 4). For example, consider $\pi_1^t\pi_2^t$, the argument to the x^t-prox mapping in (Equation 3.20). As shown in Figure Figure 3.1, since the estimator $\nabla f_2(\pi_2^t, \xi_2^1)$ is used to generate π_1^t in Line 4 of Algorithm 4, $\mathbb{E}[\nabla f_2(\underline{y}_2^t, \xi_2^1)|\pi_1^t] \neq \nabla f_2(\underline{y}_2^t)$. Line 5 needs a new estimator $\nabla f_2(\underline{y}_2^t, \xi_2^0)$ independent of π_1^t (conditioned on π_2^t) to ensure

$$
\mathbb{E}[\nabla f_1(\underline{y}_1^t, \xi_1^0)\nabla f_2(\underline{y}_2^t, \xi_2^0)|\pi_1^t, \pi_2^t] = \nabla f_1(\underline{y}_1^t)\nabla f_2(\underline{y}_2^t) = \pi_1^t\pi_2^t.
$$

An additional estimator $f_2(\underline{y}_2^{t-1}, \hat{\xi}_2)$ is also required in the next iteration for a similar reason. To highlight their independence, we use the notation ξ_i^j to emphasize that an independent

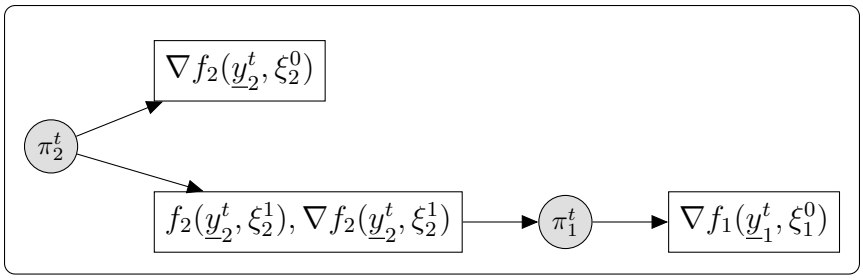


Figure 3.1: Illustration of stochastic dependency: $\nabla f_2(\underline{y}_2^t, \xi_2^0)$ is independent of $\nabla f_1(\underline{y}_1^t, \xi_1^0)$ conditioned on $\pi_2^t = \nabla f_(\underline{y}_2^t)$.

j^{th} estimator is drawn from $\mathcal{SO}_i$ and that it is used as part of the argument for the prox-mapping to reduce Q_j, and use the notation $\hat{\xi}_i$ to emphasize it being used for momentum extrapolation in the next iteration.

Algorithm 4 Stochastic Sequential Dual (SSD) Method for the Smooth Two Layer Problem

Input: $x^0 \in X$.

1: Set $\underline{y}_2^0 = x^{-1} \leftarrow x^0$. Call $\mathcal{SO}_2$ to obtain $\nabla f_2(\underline{y}_2^0, \hat{\xi}_2)$ and set $\underline{y}_1^0 \leftarrow f_2(x^0, \hat{\xi}_2)$.

2: **for** $t = 1, 2, 3...N$ **do**

3: Set $\tilde{x}^t \leftarrow x^{t-1} + \theta_t(x^{t-1} - x^{t-2})$. Set $\underline{y}_2^t \leftarrow (\tau_{2,t}\underline{y}_2^{t-1} + \tilde{x}^t)/(1 + \tau_{2,t})$. Call $\mathcal{SO}_2$ to obtain independent estimates $\{f_2(\underline{y}_2^t, \xi_2^1), \nabla f_2(\underline{y}_2^t, \xi_2^1), \nabla f_2(\underline{y}_2^t, \xi_2^0), \nabla f_2(\underline{y}_2^t, \hat{\xi}_2)\}$.

4: Set $\tilde{y}_1^t(\xi) \leftarrow f_2(\underline{y}_2^t, \xi_2^1) + \nabla f_2(\underline{y}_2^t, \xi_2^1)(x^{t-1} - \underline{y}_2^t) + \theta_t \nabla f_2(\underline{y}_2^{t-1}, \hat{\xi}_2)(x^{t-1} - x^{t-2})$. Set $\underline{y}_1^t \leftarrow (\tau_{1,t}\underline{y}_1^{t-1} + \tilde{y}_1^t(\xi))/(1 + \tau_{1,t})$. Call $\mathcal{SO}_1$ to obtain $\nabla f_1(\underline{y}_1^t, \xi_1^0)$.

5: Set $\tilde{y}_0^t(\xi) \leftarrow \nabla f_1(\underline{y}_1^t, \xi_1^0)\nabla f_2(\underline{y}_2^t, \xi_2^0)$. Compute $x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t(\xi)x + u(x) + \frac{\eta_t}{2}\|x - x^{t-1}\|^2$, .

6: **end for**

7: Return $\bar{x}^N = \sum_{t=1}^N \omega_t x^t / (\sum_{t=1}^N \omega_t)$.

Now we present some modifications required to handle a structured non-smooth f_1 (c.f. (Equation 3.7)), i.e.,

$$f_1(y_1) := \max_{\pi_1 \in \Pi_1} \langle \pi_1, y_1 \rangle - f_1^*(\pi_1), \ \forall y_1 \in \mathbb{R}^{n_1}. \tag{3.21}$$

Recall that our definition of structured non-smoothness implies that the prox-mapping of f_1^*, with $\|\cdot\|^2$ as the prox-function, can be efficiently computed. So the concrete imple-

mentation of (Equation 3.17), the SSD method for the two-layer problem with a structured non-smooth f_1, is the same as Algorithm 4 except for initializing π_1^0 to some point in Π_1 and replacing Line 4 and 5 by

$$\begin{aligned}
&\text{Set } \tilde{y}_1^t(\xi) \leftarrow f_2(\underline{y}_2^t, \xi_2^1) + \nabla f_2(\underline{y}_2^t, \xi_2^1)(x^{t-1} - \underline{y}_2^t) + \theta_t \nabla f_2(\underline{y}_2^{t-1}, \hat{\xi}_2)(x^{t-1} - x^{t-2}).\\
&\text{Set } \pi_1^t \leftarrow \arg\max_{\pi_1 \in \Pi_1} \langle \pi_1, \tilde{y}_1^t(\xi) \rangle - f_1^*(\pi) - \tau_{1,t} \left\| \pi_1 - \pi_1^{t-1} \right\|^2 / 2.\\
&\text{Set } \tilde{y}_0^t(\xi) \leftarrow \pi_1^t \nabla f_2(\underline{y}_2^t, \xi_2^0)\\
&\qquad \text{and compute } x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t(\xi) x + u(x) + \tfrac{\eta_t}{2} \left\| x - x^{t-1} \right\|^2.
\end{aligned} \tag{3.22}$$

3.2.2 Convergence Results

We present in this subsection the convergence guarantees for the proposed SSD method. The proofs are deferred to subsection 3.2.4.

First, we need to specify a few problem parameters. We use the following sets, Y_1 and Y_2, to define the effective domains for f_1 and f_2.

$$Y_2 := X, Y_1 := \text{Conv}\left(\{ \mathcal{L}_2(\pi_2; x) \mid \pi_2 = \nabla f_2(\underline{y}_2),\ \underline{y}_2, x \in Y_2 \} \right). \tag{3.23}$$

Notice that Y_1 is bounded set because X is assumed to be bounded. For the layer function f_2, the definitions of the smoothness constant L_2, the Lipschitz-continuity constant M_2 (over Y_2) and variance constants, σ_2 and σ_{f_2}, are listed below.

$$\begin{aligned}
&\|\nabla f_2(y_2) - \nabla f_2(\bar{y}_2)\| \le L_2 \|y_2 - \bar{y}_2\|, \forall y_2, \bar{y}_2 \in \mathbb{R}^n.\\
&\|\nabla f_2(y_2)\| \le M_2,\ \forall y_2 \in Y_2.\\
&\mathbb{E}[\left\| \nabla f_2(\underline{y}_2) - \nabla f_2(\underline{y}_2, \xi_2) \right\|^2] \le \sigma_2^2,\ \mathbb{E}[\left\| f_2(\underline{y}_2) - f_2(\underline{y}_2, \xi_2) \right\|^2] \le \sigma_{f_2}^2 \le \sigma_2^2 \mathcal{D}_X^2, \forall \underline{y}_2 \in Y_2.
\end{aligned} \tag{3.24}$$

Notice that we assume the variance of $f_2(x, \xi)$ to be bounded also by $\sigma_2^2 \mathcal{D}_X^2$ to simplify

the notation. For the layer function f_1, we require the Lipschitz-smoothness L_1 and the variance constant σ_1 to be defined over $\mathbb{R}^{m_1}$ because the stochastic estimate $\tilde{y}_1^t(\xi)$ could deviate from Y_1. However, the Lipschitz-continuity constant M_1 is defined only over the bounded effective domain Y_1 (c.f. (Equation 3.23)). This ensures that it always remains finite. Specifically, their definitions are given by

$$\begin{aligned} &\|\nabla f_1(y_1) - \nabla f_1(\bar{y}_1)\| \leq L_1 \|y_1 - \bar{y}_1\|, \\ &\mathbb{E}[\left\|\nabla f_1(\underline{y}_1) - \nabla f_1(\underline{y}_1, \xi_1)\right\|^2] \leq \sigma_1^2, \ \forall y_1, \bar{y}_1 \in \mathbb{R}^{n_1}. \\ &\left\|\nabla f_1(\underline{y}_1)\right\| \leq M_1, \ \forall \underline{y}_1 \in Y_1. \end{aligned} \tag{3.25}$$

Additionally, it is also useful to define uniform upper bounds for variances associated with the stochastic arguments to the prox-mappings in Algorithm 4.

$$\tilde{\sigma}_1 := \{\max_{t\geq 0} \mathbb{E}[\left\|\tilde{y}_1^t(\xi) - \mathbb{E}[\tilde{y}_1^t(\xi)]\right\|^2]\}^{1/2}, \ \tilde{\sigma}_x := \{\max_{t\geq 0} \mathbb{E}[\left\|\tilde{y}_0^t(\xi) - \mathbb{E}[\tilde{y}_0^t(\xi)]\right\|^2]\}^{1/2}. \tag{3.26}$$

Now we are ready to state the convergence rate for the non-strongly convex problem.

Theorem 3.1 *Let a smooth two-layer function f be given, with its problem parameters defined in* (Equation 3.24)*,* (Equation 3.25) *and* (Equation 3.26)*. If $\{x^t\}$ is generated by Algorithm 4 with*

$$\omega_t = t, \ \theta_t = (t-1)/t, \ \tau_{1,t} = \tau_{2,t} = (t-1)/2, \tag{3.27}$$

then $\tilde{\sigma}_1^2 \leq 3\sigma_2^2\mathcal{D}_X^2 + 2\sigma_{f_2}^2 \leq 5\sigma_2^2\mathcal{D}_X^2$ and $\tilde{\sigma}_x^2 \leq \sigma_1^2 M_2^2 + 2M_1^2\sigma_2^2 + \sigma_2^2(\sigma_1^2 + 10L_1^2\sigma_2^2\mathcal{D}_X^2)$. If, in addition, η_t is chosen as

$$\eta_t = \max\{\tfrac{2}{t+1}(\bar{M}_1 L_2 + L_1 M_2^2), \tfrac{\sqrt{t}\tilde{\sigma}_x}{D_X}\}, \tag{3.28}$$

for some $\bar{M}_1 \geq \left\|\nabla f_1(f_2(\bar{x}^N))\right\|$ *, the ergodic average solution* $\bar{x}^N$ *satisfies*

$$\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq \frac{\bar{M}_1 L_2 + L_1 M_2^2}{N(N+1)} \left\|x^0 - x^*\right\|^2 + \frac{4L_1\tilde{\sigma}_1^2}{N} + \frac{2\bar{M}_1\tilde{\sigma}_1}{\sqrt{N}} + \frac{4\tilde{\sigma}_x \mathcal{D}_X}{\sqrt{N}}, \forall N \geq 2.$$

We make two remarks regarding the result. First, in the deterministic case, i.e., $\sigma_1 = \sigma_2 = 0$, the SSD method has an oracle complexity of $\mathcal{O}((\bar{M}_1 L_2 + L_1 M_2^2)^{1/2} \|x^0 - x^*\| / \sqrt{\epsilon})$. Since $f_1 \circ f_2$ restricted to X has a smoothness constant of $L_1 M_2^2 + M_1 L_2$ and $\bar{M}_1 \leq M_1$, the oracle complexity of the SSD method is of the same order as the optimal oracle complexity of $\mathcal{O}((M_1 L_2 + L_1 M_2^2)^{1/2} \|x^0 - x^*\| / \sqrt{\epsilon})$ for solving a smooth f, e.g. obtained by using the Nesterov's accelerated gradient method. In practice, if there exists some prior knowledge about $\left\|\nabla f_1(f_2(\bar{x}^N))\right\|$, say the output solution $\bar{x}^N$ is inside some ball around x^*, then we can select $\bar{M}_1$ to be significantly smaller than M_1 such that the SSD method outperforms the Nesterov's accelerated gradient method. This improvement is made possible by exploiting the nested problem structure. Second, the stepsize parameters of prox-mappings for the dual variables in (Equation 3.27) are parameter independent. The specific of choice of η_t in (Equation 3.28) is only required to achieve the desired constant dependence. In practice, any η_t that scales $\Theta(\sqrt{t})$ leads to the order-optimal stochastic oracle complexity of $\mathcal{O}(1/\epsilon^2)$.

The next theorem presents the convergence rate for the strongly convex problem.

Theorem 3.2 *Let a smooth two-layer function* f *(c.f.* (Equation 3.4)*) be given and let the regularization term* $u(x)$ *have a positive strong convexity modulus of* α*. Let problem parameters be defined in* (Equation 3.24)*,* (Equation 3.25) *and* (Equation 3.26)*. If* $\{x^t\}$ *is generated by Algorithm 4 with*

$$\omega_t = t,\ \theta_t = (t-1)/t,\ \tau_{1,t} = \tau_{2,t} = (t-1)/2,$$

we have $\tilde{\sigma}_1^2 \leq 3\sigma_2^2 \mathcal{D}_X^2 + 2\sigma_{f_2}^2 \leq 5\sigma_2^2 \mathcal{D}_X^2$ *and* $\tilde{\sigma}_x^2 \leq \sigma_1^2 M_2^2 + 2M_1^2\sigma_2^2 + \sigma_2^2(\sigma_1^2 + 10L_1^2\sigma_2^2\mathcal{D}_X^2)$.

If, in addition, η_t is chosen as

$$\eta_t := \max\{\tfrac{2}{t+1}(\bar{M}_1 L_2 + L_1 M_2^2), \tfrac{(t-1)\alpha}{2}\},$$

for some $\bar{M}_1 \geq \left\|\nabla f_1(f_2(\bar{x}^N))\right\|$, the ergodic average solution $\bar{x}^N$ and the last iterate x^N satisfy

$$\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq (\tfrac{L_1 M_2^2}{\alpha} + 1)[\tfrac{\log(N+1)\tilde{L}}{N^2}\left\|x^0 - x^*\right\|^2 + \tfrac{4}{N}(L_1\tilde{\sigma}_1^2 + \tfrac{\tilde{\sigma}_x^2}{\alpha})], \tag{3.29}$$

$$\mathbb{E}[\left\|x^N - x^*\right\|^2] \leq \tfrac{\tilde{L}}{\alpha N(N+1)}\left\|x^0 - x^*\right\|^2 + \tfrac{4}{\alpha N}(L_1\tilde{\sigma}_1^2 + \tfrac{\tilde{\sigma}_x^2}{\alpha}), \tag{3.30}$$

where $\tilde{L} := \bar{M}_1 L_2 + L_1 M_2^2$ denotes the overall smoothness constant.

A few remarks are in order. First, a simpler stepsize choice of $\eta_t = (t-1)\alpha/2$ still leads to an order-optimal stochastic oracle complexity of $\mathcal{O}(1/\epsilon)$, improving over the $\mathcal{O}(1/\epsilon^2)$ complexity of the non-strongly problem. Second, by applying a certain restarting technique to Algorithm 4 (see Section 4.2.3 of [28]), the stochastic oracle complexity for finding an ϵ-close x^N, i.e., $\mathbb{E}[\left\|x^N - x^*\right\|^2] \leq \epsilon$, can be improved further to

$$\mathcal{O}[\sqrt{\tilde{L}/\alpha}\log(\left\|x^0 - x^*\right\|^2/\epsilon) + \tfrac{4}{\alpha\epsilon}(L_1\tilde{\sigma}_1^2 + \tfrac{\tilde{\sigma}_x^2}{\alpha})].$$

Notice that the deterministic part, $\mathcal{O}[\sqrt{\tilde{L}/\alpha}\log(\|x^0 - x^*\|^2/\epsilon)]$, is not improvable and the stochastic part has optimal dependence with respect to both α and ϵ. Third, the stochastic oracle complexity for finding an ϵ-optimal solution, i.e., $\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq \epsilon$, is also $\mathcal{O}(1/\epsilon\alpha^2)$. Its dependence on α is worse than the $\mathcal{O}(1/\epsilon\alpha)$ complexity for solving a one-layer stochastic optimization problem. It results from the analysis technique of combining the Q-gap function and the distance $\left\|\bar{x}^N - x^*\right\|^2$ together to derive a function-value bound. Such a worse dependence on α is also observed in [61] where a similar technique is used for analyzing a randomized algorithm. However, it is unclear if the dependence on α is improvable for the NSCO problem.

Now we move on to consider the case with a structured non-smooth f_1. We use $\tilde{M}_1$ to denote its Lipschitz continuity constant, i.e.,

$$\max_{\pi_1 \in \Pi_1} \|\pi_1\| \leq \tilde{M}_1. \tag{3.31}$$

Notice that the above definition requires f_1 to be $\tilde{M}_1$-Lipschitz continuous over $\mathbb{R}^{n_1}$, rather than over the bounded effective domain Y_1 in (Equation 3.25). The convergence guarantee for the non-strongly case is presented in the next theorem.

Theorem 3.3 *Let a two-layer function f (c.f.* (Equation 3.4)*) comprised of a structured non-smooth f_1 and a smooth f_2 be given. Let the problem parameters be defined in* (Equation 3.24)*,* (Equation 3.26) *and* (Equation 3.31)*. If $\{x^t\}$ is generated according to* (Equation 3.22) *with*

$$\omega_t = t,\ \theta_t = (t-1)/t,\ \tau_{2,t} = (t-1)/2,$$

we have $\tilde{\sigma}_1^2 \leq 3\sigma_2^2\mathcal{D}_X^2 + 2\sigma_{f_2}^2 \leq 5\sigma_2^2\mathcal{D}_X^2$ and $\tilde{\sigma}_x^2 \leq M_1^2\sigma_2^2$. If, in addition, the remaining prox-mapping stepsizes are chosen as

$$\eta_t := \max\{\tfrac{2M_1L_2}{t+1} + \tfrac{2M_1M_2}{\mathcal{D}_X}, \tfrac{\sqrt{t}\tilde{\sigma}_x}{\mathcal{D}_X}\}, \tau_{1,t} := \max\{\tfrac{M_2\mathcal{D}_X}{2M_1}, \tfrac{\sqrt{t}\tilde{\sigma}_1}{2M_1}\}, \tag{3.32}$$

the ergodic average solution $\bar{x}^N$ satisfies

$$\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq \tfrac{\tilde{M}_1L_2}{N^2}\left\|x^0 - x^*\right\|^2 + \tfrac{4\tilde{M}_1M_2\mathcal{D}_X}{N} + \tfrac{4\tilde{M}_1\tilde{\sigma}_1}{\sqrt{N}} + \tfrac{2\mathcal{D}_X\tilde{\sigma}_x}{\sqrt{N}}. \tag{3.33}$$

We make three remarks regarding the result. First, in the deterministic case with $\sigma_2 = 0$, the oracle complexity simplifies to $\mathcal{O}((\tilde{M}_1L_2)^{1/2}\|x^0 - x^*\|/\sqrt{\epsilon} + \mathcal{D}_X\tilde{M}_1M_2/\epsilon)$, where the first and the second term can be attributed to the smooth f_2 and the structure non-smooth f_1, respectively. Second, the specific choices of η_t and $\tau_{1,t}$ in (Equation 3.32) is required

only for the desired constant dependence. In the stochastic case, any $\eta_t = \Theta(\sqrt{t})$ and $\tau_{1,t} = \Theta(\sqrt{t})$ would lead to the order-optimal oracle complexity of $\mathcal{O}(1/\epsilon^2)$. Third, the $\mathcal{O}(1/\epsilon^2)$ complexity is not improvable even under the strongly convex setting, which we will discuss in the next subsection.

3.2.3 Lower Complexity Bound

We present a lower complexity result for the strongly convex problem with a structured non-smooth outer-layer function. Specifically, we develop a lower bound on the number of queries to $\mathcal{SO}_2$ required to obtain an (expected) ϵ-optimal solution for a class of first-order methods. For simplicity, we assume X to be a ball centered on 0, $B(0, r)$, and $u(x) := \alpha \|x\|^2 /2$.

We take an approach similar to Nesterov [1] by proposing an abstract computation scheme that updates the reachable affine subspaces. The abstract scheme consists of the following steps. In the beginning, provided with $x^0 \in X$, $y_1^0 \in \mathbb{R}^{n_1}$ and $\pi_1^0 \in \Pi_1$, the affine sub-spaces are initialized to $\mathcal{X}^0 := \text{span}(x^0)$, $\mathcal{Y}_1^0 := \{y_1^0\}$, and $\Pi_1^0 = \text{span}(\pi_1^0)$. In the t^{th} iteration, the updates are given by (the "+" in the definitions of $\mathcal{X}^t$, $\mathcal{Y}_1^t$ and Π_1^t represents the Minkowski sum.)

$$
\begin{aligned}
&\textit{Query } \mathcal{SO}_2 \textit{ to obtain } (f_2(y_2^t, \xi_2^t), f_2'(y_2^t, \xi_2^t)) \textit{ at some } y_2^t \in \mathcal{X}^{t-1}.\\
&\mathcal{Y}_1^t := \mathcal{Y}_1^{t-1} + \textit{span}\{f_2(y_2^t, \xi_2^t) - y_1^0\} + \{f_2'(y_2^j, \xi_2^j)x : j \le t, x \in \mathcal{X}^{t-1}\}.\\
&\Pi_1^t := \Pi_1^{t-1} + \textit{span}(\pi_1^t) \textit{ where } \pi_1^t = \arg\max_{\pi_1 \in \Pi_1} \langle \pi_1, y_1^t \rangle - f_1^*(\pi_1) - \tfrac{\tau_{1,t} \left\| \pi_1 - \pi_1^{t-1} \right\|^2}{2},\\
&\qquad \textit{for some } y_1^t \in \mathcal{Y}_1^t, \tau_{1,t} \ge 0, \pi_1^{t-1} \in \Pi_1^{t-1}.\\
&\mathcal{X}^t := \mathcal{X}^{t-1} + \{f_2'(y_2^j, \xi_2^j)^\top \pi_1^\top : j \le t, \pi_1 \in \Pi_1^t\}.
\end{aligned}
\tag{3.34}
$$

After N iterations, the scheme can output any $x^N \in \mathcal{X}^N$. Notice the superscript t in ξ_2^t above represents the unbiased estimator drawn for the t^{th} iterate y_2^t, rather than the t^{th}

repeated sample used in the last subsection.

The abstract scheme is quite general. As X is a ball centered at zero, $\mathcal{X}^t$ encompasses all proximal mappings from each proximal center in $\mathcal{X}^{t-1}$, using any non-negative step size η_t and any argument within the set $\mathcal{A}^t := \{f_2'(y_2^j, \xi_2^j)^\top \pi_1^\top : j \leq t, \pi_1 \in \Pi_1^t\}$. An example of this is the SCGD algorithm in [48]. Since $\mathcal{Y}_1^t$ in (Equation 3.34) contains every possible convex combination of evaluated function values $\{f_2(y_2^j, \xi_2^j)\}_{j \leq t}$, and the above π_1^t evaluation corresponds to gradient evaluation with $\tau_{1,t} = 0$, the argument set $\mathcal{A}^t$ contains the pseudo-gradient in the SCGD algorithm. Another example is our SSD method. Since it includes all possible momentum-extrapolation terms, $\mathcal{Y}_1^t$ contains $\tilde{y}_1^t(\xi)$ and $\underline{y}_1^t$ from Algorithm 4-(Equation 3.22) so that $\mathcal{A}^t$ contains $\tilde{y}_0^t(\xi)$ from Algorithm 4-(Equation 3.22). Additionally, since each iteration of Algorithm 4-(Equation 3.22) requires three independent queries to $\mathcal{SO}_2$, three iterations of the abstract scheme are sufficient to implement it. We state the lower complexity result for the abstract scheme in the next theorem, and its proof is deferred to the Appendix.

Theorem 3.4 *Let the abstract scheme in* (Equation 3.34) *be initialized to* $x^0 = 0$, $\pi_1^0 = 0$ *and some* y_1^0. *Given problem parameters* $\epsilon > 0$, $\tilde{M}_1 > 0$, $\bar{\alpha} \leq M_1^2/(4\epsilon)$, *and* $\sigma_{f_2} \geq 4\epsilon/M_1$, *there exists a nested two-layer problem* (Equation 3.4) *consisting of a structured non-smooth* f_1 *and a stochastic linear* f_2 *such that* f_1 *is* $\tilde{M}_1$*-Lipschitz continuous (c.f.* (Equation 3.31)*), the variance of* $f_2(x, \xi)$ *is bounded by* σ_{f_2} *(c.f.* (Equation 3.25)*),* $u(x) = \bar{\alpha} \|x\|^2 /2$, *and at least* $N = \Omega(\tilde{M}_1^2 \sigma_{f_2}^2/\epsilon^2)$ *iterations is necessary for finding an* ϵ*-solution* x^N *with* $\mathbb{E}[f(x^N) - f(x^*)] \leq \epsilon$.

The preceding lower bound establishes the $\mathcal{O}(1/\epsilon^2)$ oracle complexity in (Equation 3.33) to be order-optimal even when the setting is strongly convex. This shows that solving the nested composite problem with a structured non-smooth outer-layer function may be inherently more challenging than solving the one-layer problem.

3.2.4 Convergence Proofs

In this subsection, we present proofs to results in subsection 3.2.2. We begin with a convergence bound for the Q-gap function, which will be useful for proving Theorem 3.1, Theorem 3.2 and Theorem 3.3.

Proposition 3.1 *Consider a two-layer problem of the form* (Equation 3.4)*, with the Lipschitz-continuity and smoothness constants of f_2 defined in* (Equation 3.24)*. Suppose U_1 is β-strongly convex, and f_1^* is μ-strongly convex with respect to U_1, i.e.,*

$$U_1(\pi_1;\bar{\pi}_1) \geq \beta \left\|\pi_1 - \bar{\pi}_1\right\|^2/2,\ D_{f_1^*}(\pi_1;\bar{\pi}_1) \geq \mu U_1(\pi_1;\bar{\pi}_i)\ \forall \pi_1, \bar{\pi}_1 \in \Pi_1.$$

Consider solution iterates $\{z^t := (x^t; \pi_1^t, \pi_2^t)\}$ generated by the abstract Algorithm (Equation 3.17) *with $U_2 = D_{f_2^*}$ and $\pi_2^0 = \nabla f_2(x^0)$. Assume the stochastic arguments $\tilde{y}_1^t(\xi)$ and $\tilde{y}_0^t(\xi)$ satisfy the variance bounds in* (Equation 3.26)*. Let $z := (x^*; \pi_1, \pi_2 = \nabla f_2(y_2))$, with $y_2 \in X$, denote a reference point which could potentially depend on $\{z^t\}$. If the following requirements are satisfied with some non-negative weights $\{\omega_t\}$ for some constant $\tilde{M}_2 \geq \|\pi_2^t\|\ \forall t \geq 1$:*

$$\begin{aligned} &\omega_t = \theta_{t+1}\omega_{t+1},\ \omega_t(\tau_{2,t}+1) \geq \omega_{t+1}\tau_{2,t+1}, \\ &\eta_t \geq \tfrac{\theta_{t+1}\|\pi_1\|L_2}{\tau_{2,t+1}} + \tfrac{\theta_{t+1}\tilde{M}_2^2}{\tau_{1,t+1}\beta},\ \eta_N \geq \tfrac{\|\pi_1\|L_2}{\tau_{2,N}+1} + \tfrac{\tilde{M}_2^2}{(\tau_{1,N}+\mu)\beta}, \end{aligned} \tag{3.35}$$

the next Q-gap bound is valid (with $\omega_0 := 0$)

$$\begin{aligned} &\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q(z^t; z) + \omega_N(\eta_N + \alpha)\left\|x^N - x^*\right\|^2/2] \\ \leq &\mathbb{E}[\textstyle\sum_{t=1}^N \{\omega_t\eta_t - \omega_{t-1}(\eta_{t-1}+\alpha)\}\left\|x^{t-1} - x^*\right\|^2/2] \\ &+ \mathbb{E}[\textstyle\sum_{t=1}^N \{\omega_t\tau_{1,t} - \omega_{t-1}(\tau_{1,t-1}+\mu)\}U_1(\pi_1;\pi_1^{t-1})] + \omega_1\tau_{2,1}D_{f_2^*}(\pi_2;\pi_2^0) \\ &+ \textstyle\sum_{t=1}^N \omega_t\tilde{\sigma}_x^2/(\eta_t+\alpha) + \sum_{t=1}^N \omega_t\tilde{\sigma}_1^2/\beta(\tau_{1,t}+\mu) + \mathbb{E}[\sum_{t=1}^N \omega_t\langle\pi_1, \tilde{y}_1^t - \tilde{y}_1^t(\xi)\rangle]. \end{aligned} \tag{3.36}$$

Proof: First, we use the definition of prox-mapping for π_1 in (Equation 3.17) to derive a convergence bound for Q_1 (c.f. (Equation 3.14)). The $\mu-$strong convexity of f_1^* with respect to U_1 implies a three point inequality (e.g. Lemma 3.8 of [28])

$$\begin{aligned} &-(\pi_1^t - \pi_1)\tilde{y}_1^t(\xi) + f_1^*(\pi_1) - f_1^*(\pi_1^t) + (\tau_{1,t} + \mu)U_1(\pi_1; \pi_1^t) \\ &\quad + \tau_{1,t}(U_1(\pi_1^t; \pi_1^{t-1}) - U_1(\pi_1; \pi_1^{t-1})) \leq 0. \end{aligned} \tag{3.37}$$

Let's focus on $-(\pi_1^t - \pi_1)\tilde{y}_1^t(\xi)$. Setting $\delta_1^t := \tilde{y}_1^t - \tilde{y}_1^t(\xi)$, we get

$$\begin{aligned} \mathbb{E}[-(\pi_1^t - \pi_1)\tilde{y}_1^t(\xi)] = \mathbb{E}[&- (\pi_1^t - \pi_1)\mathcal{L}_2(\pi_2^t; x^t) \\ &+ (\pi_1^t - \pi_1)\delta_1^t + (\pi_1^t - \pi_1)(\mathcal{L}_2(\pi_2^t; x^t) - \tilde{y}_1^t)], \end{aligned}$$

whereby

$$\begin{aligned} (\pi_1^t - \pi_1)(\mathcal{L}_2(\pi_2^t; x^t) - \tilde{y}_1^t) &= (\pi_1^t - \pi_1)\pi_2^t(x^t - x^{t-1}) - \theta_t(\pi_1^{t-1} - \pi_1)\pi_2^{t-1}(x^{t-1} - x^{t-2}) \\ &\quad + \theta_t(\pi_1^{t-1} - \pi_1^t)\pi_2^{t-1}(x^{t-1} - x^{t-2}), \\ \mathbb{E}[(\pi_1^t - \pi_1)\delta_1^t] &= \mathbb{E}[\pi_1^t\delta_1^t] - \mathbb{E}[\pi_1\delta_1^t]. \end{aligned}$$

Consider $\mathbb{E}[\pi_1^t\delta_1^t]$ for a fixed t. We can write the π_1^t proximal mapping in (Equation 3.17) equivalently as:

$$\pi_1^t \leftarrow \underset{\pi_1 \in \Pi_1}{\arg\min}\, \pi_1(g + \delta_1^t) + [f_1^*(\pi_1) + \tau_{1,t}U_i(\pi_1; \pi_1^{t-1})],$$

with $g := \tilde{y}_1^t$. Since $\mathbb{E}[\delta_1^t|\tilde{y}_1^t] = 0$, $\mathbb{E}[\|\delta_1^t\|^2] \leq \tilde{\sigma}_1^2$, and that $[f_1^*(\pi_1) + \tau_{1,t}U_i(\pi_1; \pi_1^{t-1})]$ has a strong convexity modulus of $(\mu + \tau_{1,t})\beta$, we have from Lemma 3.7 from the Chapter Appendix that

$$-\mathbb{E}[\pi_1^t\delta_1^t] \leq \tfrac{\tilde{\sigma}_1^2}{\beta(\mu+\tau_{1,t})}.$$

Moreover, Young's inequality and the fact $x^0 = x^{-1}$ lead us to

$$\begin{aligned}
&\mathbb{E}[-\theta_t(\pi_1^{t-1} - \pi_1^t)\pi_2^{t-1}(x^{t-1} - x^{t-2}) - \tau_{1,t}U_1(\pi_1^t; \pi_1^{t-1})] \\
&\qquad \leq \theta_t^2 L_1 \tilde{M}_2^2 \left\|x^{t-1} - x^{t-2}\right\|^2 / \beta\tau_{1,t}, \\
&\mathbb{E}[-(\pi_1^N - \pi_1)\pi_2^N(x^N - x^{N-1}) - (\tau_N + \alpha)U_1(\pi_1^N; \pi_1^{N-1})] \\
&\qquad \leq L_1 \tilde{M}_2^2 \left\|x^N - x^{N-1}\right\|^2 / \beta(\tau_N + \mu), \\
&\mathbb{E}[-\theta_t(\pi_1^0 - \pi_1^1)\pi_2^0(x^0 - x^{-1})] = 0.
\end{aligned}$$

Thus, the ω_t-weighted sum of (Equation 3.37) satisfies

$$\begin{aligned}
\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q_1(z^t; z)] \leq \sum_{t=1}^N [\omega_t\tau_{1,t} - \omega_{t-1}(\tau_{1,t-1} + \mu)]U_1(\pi_1; \pi_1^{t-1}) + \sum_{t=1}^N \tfrac{\omega_t \tilde{\sigma}_1^2}{\beta(\tau_{1,t}+\mu)} \\
+ \mathbb{E}[\textstyle\sum_{t=1}^N \pi_1 \delta_1^t] + \sum_{t=2}^N \tfrac{\omega_{t-1}\theta_t \tilde{M}_2^2}{\beta\tau_{1,t}} \left\|x^{t-1} - x^{t-2}\right\|^2 + \tfrac{\omega_N \tilde{M}_2^2}{\beta(\tau_{1,N}+\mu)} \left\|x^N - x^{N-1}\right\|^2.
\end{aligned} \tag{3.38}$$

Next, we consider the prox-mapping for π_2. The definition of π_2^t in (Equation 3.17), together with f_2^* being 1-strongly convex with respect to $D_{f_2^*}$, implies the following three point inequality

$$\begin{aligned}
&-(\pi_2^t - \pi_2)\tilde{x}^t + f_2^*(\pi_2) - f_2^*(\pi_2^t) + (\tau_{2,t} + 1)D_{f_2^*}(\pi_2; \pi_2^t) + \tau_{2,t}D_{f_2^*}(\pi_2^t; \pi_2^{t-1}) \\
&\qquad - \tau_{2,t}D_{f_2^*}(\pi_2; \pi_2^{t-1}) \leq 0.
\end{aligned}$$

Multiplying its rows with weight π_1 leads to,

$$\begin{aligned}
&- \pi_1[(\pi_2^t - \pi_2)\tilde{x}^t + f_2^*(\pi_2) - f_2^*(\pi_2^t)] + \\
&\qquad (\tau_{2,t} + 1)\pi_1 D_{f_2^*}(\pi_2; \pi_2^t) + \tau_{2,t}\pi_1 D_{f_2^*}(\pi_2^t; \pi_2^{t-1}) - \tau_{2,t}\pi_1 D_{f_2^*}(\pi_2; \pi_2^{t-1}) \leq 0.
\end{aligned} \tag{3.39}$$

Since $\pi_2^t = \nabla f_2(y_2^t)$ for some $y_2^t \; \forall t$ (see Lemma 3.2), Lemma 3.6 in the Chapter Appendix

leads to

$$\|\pi_1\|\, \pi_1 D_{f_2^*}(\pi_2^t; \pi_2^{t-1}) \geq \frac{\left\|\pi_1(\pi_2^t - \pi_2^{t-1})\right\|^2}{2L_2}.$$

We can obtain a Q_2 bound using a similar argument as above

$$\begin{aligned}\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q_2(z^t; z)] \leq & \omega_1 \tau_{2,1} \pi_1 D_{f_2^*}(\pi_2; \pi_2^0) \\ & + \textstyle\sum_{t=2}^N \frac{\omega_{t-1}\theta_t L_2 \|\pi_1\|}{\tau_{2,t}} \left\|x^{t-1} - x^{t-2}\right\|^2 + \frac{\omega_N L_2 \|\pi_1\|}{\tau_{2,N}+1} \left\|x^N - x^{N-1}\right\|^2.\end{aligned} \tag{3.40}$$

Similarly, we can derive from the prox-mapping for x^t (Equation 3.17) that

$$\begin{aligned}&\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q_0(z^t; z) + \frac{\omega_N(\eta_N+\alpha)}{2} \left\|x^N - x^*\right\|^2] + \mathbb{E}[\textstyle\sum_{t=1}^N \frac{\omega_t \eta_t}{2} \left\|x^t - x^{t-1}\right\|^2] \\ &\quad \leq \textstyle\sum_{t=1}^N \omega_t \tilde{\sigma}_x^2/(\eta_t + \alpha) \mathbb{E}[\textstyle\sum_{t=1}^N \{\omega_t \eta_t - \omega_{t-1}(\eta_{t-1} + \alpha)\} \left\|x^{t-1} - x^*\right\|^2].\end{aligned} \tag{3.41}$$

The desired inequality in (Equation 3.36) then follows from adding up (Equation 3.38), (Equation 3.40) and (Equation 3.41). □

We specialize Proposition 3.1 to prove Theorem 3.1.

Proof of Theorem 3.1 Clearly, the proposed Algorithm 4 is a special case of (Equation 3.17) with $U_1 = D_{f_1^*}$, $\pi_1^t = \nabla f_1(\underline{y}_1^t)$ and $\pi_2^t = \nabla f_2(\underline{y}_2^t)$.

We provide bounds for the variance constants in (Equation 3.26). We have $\tilde{\sigma}_1^2 \leq 2\sigma_{f_2}^2 + 3\sigma_2^2 \mathcal{D}_X^2$ because

$$\begin{aligned}&\mathbb{E}[\left\|\tilde{y}_1^t(\xi) - \mathbb{E}[\tilde{y}_1^t(\xi)]\right\|^2] \\ &\quad = \mathbb{E}[\|\{f_2(\underline{y}_2^t, \xi_2^1) - f_2(\underline{y}_2^t)\} + \{\nabla f_2(\underline{y}_2^t, \xi_2^1) - \nabla f_2(\underline{y}_2^t)\}(x^{t-1} - \underline{y}_2^t) \\ &\qquad\quad + \theta_t\{\nabla f_2(\underline{y}_2^t, \hat{\xi}_2) - \nabla f_2(\underline{y}_2^t)\}(x^{t-1} - x^{t-2})\|^2] \\ &\quad \leq 2\sigma_{f_2}^2 + 2\sigma_2^2 \mathcal{D}_X^2 + \sigma_2^2 \mathcal{D}_X^2 \leq 2\sigma_{f_2}^2 + 3\sigma_2^2 \mathcal{D}_X^2.\end{aligned}$$

The variance constant associated with $\tilde{y}_0^t(\xi)$ is more involved. We begin with bounds for $\|\pi_1^t\| = \left\|\nabla f_1(\underline{y}_1^t)\right\|$ and $\|\pi_2^t\| = \left\|\nabla f_2(\underline{y}_2^t)\right\|$. We need some more useful characterizations

for $\underline{y}_1^t$ and $\underline{y}_2^t$. Since $\theta_t = (t-1)/t$, $\tau_{2,t} = (t-1)/2$ and $\underline{y}_1^0 = x^0 \in X$, we get

$$\underline{y}_2^t = \frac{2(\sum_{l=1}^{t-1} lx^l + tx^{t-1})}{t(t+1)} \quad \forall t \geq 1.$$

So $\underline{y}_2^t \in X$ and $\|\pi_2^t\| = \left\|\nabla f_2(\underline{y}_2^t)\right\| \leq M_2$ for all $t \geq 0$. Similarly, setting $\hat{\mathcal{L}}_2(x; \pi_2^t(\xi)) := f_2(\underline{y}_2^t, \xi_2^1) - \nabla f_2(\underline{y}_2^t, \xi_2^1)\underline{y}_2^t + \nabla f_2(\underline{y}_2^t, \hat{\xi}_2)x + [\nabla f_2(\underline{y}_2^t, \xi_2^1) - \nabla f_2(\underline{y}_2^t, \hat{\xi}_2)]x^{t-1}$, we have

$$\underline{y}_1^t = \frac{2[\sum_{l=1}^{t-1} l\hat{\mathcal{L}}_2(x^l; \pi_2^l(\xi)) + t\hat{\mathcal{L}}_2(x^{t-1}; \pi_2^t(\xi)]}{t(t+1)} \quad \forall t \geq 1. \tag{3.42}$$

Define $\underline{y}_{1,\mathbb{E}}^t := 2[\sum_{l=1}^{t-1} l\mathcal{L}_2(x^l; \pi_2^l) + \mathcal{L}_2(x^{t-1}; \pi_2^t)]/[t(t+1)] \in Y_1$. By Jensen's inequality, we have

$$\mathbb{E}[\left\|\underline{y}_1^t - \underline{y}_{1,\mathbb{E}}^t\right\|^2] \leq \max_{l \leq t} \mathbb{E}[\left\|\mathcal{L}_2(x^l; \pi_2^l(\xi)) - \mathcal{L}_2(x^t; \pi_2^t)\right\|^2] \leq 5\sigma_2^2 \mathcal{D}_X^2.$$

Thus the L_1-smoothness of f_1 implies that

$$\mathbb{E}[\left\|\nabla f_1(\underline{y}_1^t)\right\|^2] \leq 2\mathbb{E}[\left\|\nabla f_1(\underline{y}_{1,\mathbb{E}}^t)\right\|^2] + 2L_1^2\mathbb{E}[\left\|\underline{y}_1^t - \underline{y}_{1,\mathbb{E}}^t\right\|^2] \leq 2M_1^2 + 10L_1^2\mathcal{D}_X^2\sigma_2^2.$$

The conditional independence of $\nabla f_1(\underline{y}_1^t, \xi_1^0)$ and $\nabla f_2(\underline{y}_2^t, \xi_2^0)$ implies that

$$\begin{aligned}
\mathbb{E}[\left\|\underline{y}_0^t(\xi) - \underline{y}_0^t\right\|^2] &= \mathbb{E}[\left\|\nabla f_1(\underline{y}_1^t, \xi_1^0)\nabla f_2(\underline{y}_2^t, \xi_2^0) - \nabla f_1(\underline{y}_1^t)\nabla f_2(\underline{y}_2^t)\right\|^2] \\
&\leq \sigma_1 M_2^2 + \sigma_1^2\sigma_2^2 + \sigma_2^2\mathbb{E}[\left\|\nabla f_1(\underline{y}_1^t)\right\|] \\
&\leq \sigma_1^2 M_2^2 + 2M_1^2\sigma_2^2 + \sigma_2^2(\sigma_1^2 + 10L_1^2\sigma_2^2\mathcal{D}_X^2).
\end{aligned}$$

So the upper bound for $\tilde{\sigma}_x^2$ in the theorem statement is valid.

Now we return to derive the function value convergence bound. Clearly, the requirements in Proposition 3.1 is satisfied with $\beta = 1/L_1$, $\mu = 1$, and $\tilde{M}_2 = M_2$, so it follows

from (Equation 3.36) that

$$\begin{aligned}\mathbb{E}[\textstyle\sum_{t=1}^{N}\omega_t Q(z^t;z)] \leq \textstyle\sum_{t=1}^{N}[\omega_t\eta_t-\omega_{t-1}\eta_{t-1}]\left\|x^{t-1}-x^*\right\|^2/2+\sum_{t=1}^{N}\omega_t\tilde{\sigma}_x^2/\eta_t \\ +2NL_1\tilde{\sigma}_1^2+\mathbb{E}\textstyle\sum_{t=1}^{N}\omega_t\langle\pi_1,\tilde{y}_1^t-\tilde{y}_1^t(\xi)\rangle.\end{aligned} \tag{3.43}$$

In particular, the Cauchy-Schwartz inequality implies that

$$\begin{aligned}\mathbb{E}[\textstyle\sum_{t=1}^{N}\omega_t\langle\pi_1,\tilde{y}_1^t-\tilde{y}_1^t(\xi)\rangle] &= \mathbb{E}^{1/2}[\|\pi_1\|^2]\mathbb{E}^{1/2}[\left\|\textstyle\sum_{t=1}^{N}\omega_t\{\tilde{y}_1^t-\tilde{y}_1^t(\xi)\}\right\|^2] \\ &\leq N^{3/2}\tilde{\sigma}_1^2\mathbb{E}^{1/2}[\|\pi_1\|^2]\;\forall N\geq 2.\end{aligned} \tag{3.44}$$

Moreover, setting $\hat{\pi}_1^N=\nabla f_1(f_2(\bar{x}^N))$ and $\hat{\pi}_2^N=\nabla f_2(\bar{x}^N)$, the strong duality and the weak duality relations in Lemma 4.1, and the linearity of $\mathcal{L}(x;\pi_1,\pi_2)$ with respect to x imply

$$\begin{aligned}(\textstyle\sum_{t=1}^{N}\omega_t)(f(\bar{x}^N)-f(x^*)) &= \textstyle\sum_{t=1}^{N}\omega_t\mathcal{L}(\sum_{t=1}^{N}\omega_t x^t/[\sum_{t=1}^{N}\omega_t];\hat{\pi}_1^N,\hat{\pi}_2^N)-\sum_{t=1}^{N}\omega_t f(x^*) \\ &\leq \textstyle\sum_{t=1}^{N}\omega_t[\mathcal{L}(x^t;\hat{\pi}_1^N,\hat{\pi}_2^N)-\mathcal{L}(x^*;\pi_1^t,\pi_2^t)] \\ &\leq \textstyle\sum_{t=1}^{N}\omega_t Q(z^t;(x^*;\hat{\pi}_1^N,\hat{\pi}_2^N)).\end{aligned} \tag{3.45}$$

In view of the preceding two inequalities and that $\|\hat{\pi}_1^t\|\leq\bar{M}_1$, the desired function-value convergence rate can be derived from (Equation 3.43) by setting z to $(x^*;\hat{\pi}_2^t,\hat{\pi}_2^t)$ and dividing the both sides by $(\sum_{t=1}^{N}\omega_t)$. □

Proof of Theorem 3.2 First, the bounds on $\tilde{\sigma}_x$ and $\tilde{\sigma}_1$ follow from the same argument as that of Theorem 3.1.

Now let us develop a convergence bound for $\left\|x^N-x^*\right\|^2$. Let $z^*:=(x^*;\pi_1^*,\pi_2^*)$ denote the saddle point to (Equation 3.8). Because our stepsizes satisfy the requirements in Proposition 3.1 with $\mu=1$ and $\beta=1/L_1$, the consequent bound in (Equation 3.36) holds for $z=z^*$. Taking in account $Q(z^t;z^*)\geq 0\;\forall t$ and that $\mathbb{E}[\sum_{t=1}^{N}\omega_t\langle\pi_1^*,\tilde{y}_1^t-\tilde{y}_1^t(\xi)\rangle]=0$,

we get the

$$\mathbb{E}[\left\|x^N - x^*\right\|^2] \leq \frac{\tilde{L}}{\alpha N(N+1)} \left\|x^0 - x^*\right\|^2 + \frac{4}{\alpha N}(L_1\tilde{\sigma}_1^2 + \frac{\tilde{\sigma}_x^2}{\alpha}), \forall N \geq 1.$$

In particular, Jensen's inequality implies

$$\mathbb{E}[\left\|\bar{x}^N - x^*\right\|^2] \leq \frac{2\tilde{L}\log(N+1)}{\alpha N^2} \left\|x^0 - x^*\right\|^2 + \frac{8}{\alpha N}(L_1\tilde{\sigma}_1^2 + \frac{\tilde{\sigma}_x^2}{\alpha}). \tag{3.46}$$

Next, choosing some $\hat{\pi}_2 \in \partial f_2(\bar{x}^N)$ and $\hat{\pi}_1 = \nabla f_1(f_2(\bar{x}^N))$, and applying Proposition 3.1 again with the reference point $(x^*; \pi_1^*, \hat{\pi}_2)$ for some $\hat{\pi}_1 \in \partial f_2(\bar{x}^N)$, we get

$$\mathbb{E}[\mathcal{L}(\bar{x}^N; \pi_1^*, \hat{\pi}_2) - f(x^*)] \leq \frac{\tilde{L}}{\alpha N(N+1)} \left\|x^0 - x^*\right\|^2 + \frac{4}{\alpha N}(L_1\tilde{\sigma}_1^2 + \frac{\tilde{\sigma}_x^2}{\alpha}). \tag{3.47}$$

Towards characterizing the function value convergence $\mathbb{E}[f(\bar{x}^N) - f(x^*)]$, we need to bound the difference $\mathbb{E}[f(\bar{x}^N) - \mathcal{L}(\bar{x}^N; \pi_1^*, \hat{\pi}_2)]$:

$$\begin{aligned}
\mathbb{E}[f(\bar{x}^N) - \mathcal{L}(\bar{x}^N; \pi_1^*, \hat{\pi}_2)] &= \mathbb{E}[\mathcal{L}(\bar{x}^N; \hat{\pi}_1, \hat{\pi}_2) - \mathcal{L}(\bar{x}^N; \pi_1^*, \hat{\pi}_2)] \\
&= \mathbb{E}[f_1^*(\pi_1^*) - f_1^*(\hat{\pi}_1) - \langle \pi_1^* - \hat{\pi}_1, \mathcal{L}_2(\bar{x}^N; \hat{\pi}_2)\rangle] \\
&= \mathbb{E}[D_{f_1^*}(\pi_1^*; \hat{\pi}_1)] = \mathbb{E}[D_{f_1}(f_2(\bar{x}^N); f_2(x^*))] \\
&\leq \tfrac{L_1}{2}\mathbb{E}[\left\|f_2(\bar{x}^N) - f_2(x^*)\right\|^2] \leq \tfrac{L_1 M_2^2}{2}\mathbb{E}[\left\|\bar{x}^N - x^*\right\|^2].
\end{aligned} \tag{3.48}$$

Here, the fourth equality utilizes the fact that $\hat{\pi}_1 = \nabla f_1(f_2(\bar{x}^N)) \Leftrightarrow \mathcal{L}_2(\hat{\pi}_2; \bar{x}^N) = f_2(\bar{x}^N) \in \partial f_1^*(\hat{\pi}_1)$. The fifth equality relates the dual Bregman distance function $D_{f_i^*}$ generated by the conjugate function f_1^* to the primal one generated by f_1. Then the second last inequality follows from the L_1-smoothness of f_1, which provides upper bounds for the D_{f_1}.

Substituting (Equation 3.46) into (Equation 3.48) and then combining it with (Equation 3.47), we can conclude the desired functional value convergence bound in

(Equation 3.29). □

Proof of Theorem 3.3 Clearly, the proposed Algorithm in (Equation 3.22) is a special case of (Equation 3.17) with $U_1(\cdot) = \|\cdot\|^2$ and $\pi_2^t = \nabla f_2(\underline{y}_2^t)$. The bound on $\tilde{\sigma}_1$ can be derived similarly to that of Theorem 3.1. For $\tilde{\sigma}_x$, we have

$$\mathbb{E}[\left\|\underline{y}_0^t(\xi) - \underline{y}_0^t\right\|^2] = \mathbb{E}[\left\|\pi_1^t\{\nabla f_2(\underline{y}_2^t, \xi_2^0) - \nabla f_2(\underline{y}_2^t)\}\right\|^2] \leq M_1^2\sigma_2^2.$$

It is easy to check the requirements in Proposition 3.1 are satisfied with $\beta = 1$ and $\mu = 0$. Then function value convergence can also be deduced using an argument similar to that of Theorem 3.1. □

3.3 General Nonsmooth Two-layer Problem

We study the general nonsmooth two-layer problem (Equation 3.4) where both f_1 and f_2 are stochastic and nonsmooth. As suggested in the introduction, the nonsmooth outer-layer function f_1 poses a critical challenge to the SSD method in that it is difficult to implement the prox-mapping for π_1 (see Line 2 in (Equation 3.17)) with a strongly convex prox-function. Although using $D_{f_1^*}$ as the prox-function can lead to efficient computation, it fails to provide the necessary stabilization that is crucial for the convergence guarantee. This is because $D_{f_1^*}$ may not be strongly convex. Conversely, using $\|\cdot\|^2$ as the prox-function can provide the desired stabilization, but the prox-mapping with f_1^* may not be easily computable. We resolve the challenge in this section by proposing a novel tri-conjugate reformulation and developing a non-smooth stochastic sequential dual (nSSD) method. Specifically, subsection 3.3.1 introduces the nSSD method, subsection 3.3.2 presents a lower complexity result when the problem is strongly convex, and subsection 3.3.3 concludes the section with the detailed convergence analysis.

3.3.1 The nSSD Method and Convergence Guarantee

First, we introduce a nested linearization reformulation for (Equation 3.4) to motivate the nSSD method. Towards that end, we need another reformulation for the non-smooth outer layer function. Given a convex function $g : \mathbb{R}^m \to \mathbb{R}$, a general linear approximation to it is specified by the evaluation point v and the approximation vector π

$$\mathcal{L}_g(y, v; \pi) := \langle \pi, y - v \rangle + g(v), \quad \pi \in \mathbb{R}^m, v \in \mathbb{R}^m. \tag{3.49}$$

Such a linear approximation has two desirable properties (Equation 3.49) $\forall y \in \mathbb{R}^n$.

- Given any $\bar{v} \in \mathbb{R}^n$, there always exists a $\hat{\pi} \in \mathbb{R}^n$, say $\hat{\pi} \in \partial g(y)$, such that $\mathcal{L}_g(y, \bar{v}; \hat{\pi}) \geq g(y)$.
- Given any $\bar{\pi} \in \mathbb{R}^n$, there always exists a $\hat{v} \in \mathbb{R}^n$ such that $\mathcal{L}_g(y, \hat{v}; \bar{\pi}) \leq g(y)$. Particularly, when $\bar{\pi}$ is some subgradient of g, i.e., $\bar{\pi} \in \text{dom}(g^*)$, we have $\mathcal{L}_g(y, \hat{v}; \bar{\pi}) = \langle \bar{\pi}, y \rangle - g^*(\bar{\pi}) \leq g(y)$ if $\bar{\pi} \in \partial g(\hat{v})$.

Taken together, the two properties imply that the $\min - \max$ saddle-point value of $\mathcal{L}_g(y, v; \pi)$ (Equation 3.49) is equivalent [②] to $g(y)$:

$$g(y) = \max_{\pi \in \mathbb{R}^n} \min_{v \in \mathbb{R}^n} \mathcal{L}_g(y, v; \pi) \quad \forall y \in \mathbb{R}^n. \tag{3.50}$$

Since $\mathcal{L}_g$ in (Equation 3.49) also provides a certain kind of decoupling between g and the possibly stochastic argument y, the reformulation could be advantageous for designing convergent algorithms for the non-smooth nested problem in (Equation 3.4).

We provide two instructive interpretations to (Equation 3.50). First, if we regard π as the Lagrange multiplier to the constraint $v = y$, (Equation 3.49) and (Equation 3.50) correspond to the Lagrangian reformulation to the linearly constrained optimization problem,

[②] We can verify the fact by the first-order optimality condition for the saddle point $(\hat{v}; \hat{\pi})$: $\hat{v} = y$ and $\hat{\pi} \in \partial g(\hat{v})$. In particular, $\hat{\pi} \in \partial g(\hat{v})$ implies $\mathcal{L}_g(y, \hat{v}; \hat{pi}) \leq g(y)$, and $\hat{\pi} \in \partial g(\hat{v}) = \partial g(y)$ implies $\mathcal{L}_g(y, \hat{v}; \hat{\pi}) \geq g(y)$. Thus we have $g(y) = \mathcal{L}_g(y; \hat{v}; \hat{\pi})$.

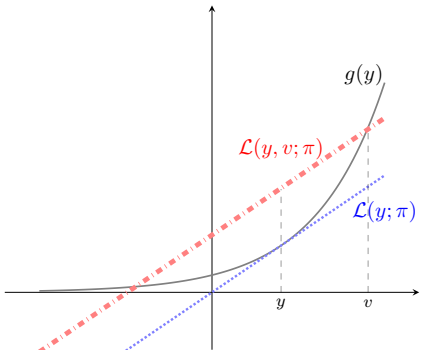


Figure 3.2: A comparison of the bi-conjugate reformulation, $\mathcal{L}_g(y;\pi)$, and the tri-conjugate reformulation, $\mathcal{L}_g(y,v;\pi)$. For a fixed π, $\mathcal{L}_g(y,v;\pi) \geq \mathcal{L}_g(y;\pi)\ \forall y \in Y$ and the equality holds if and only if $\pi \in \partial g(v)$.

$g(y) = \min_{v \in V}\{g(v) \text{ s.t. } v = y\}$. We know from nonlinear optimization literature, say [28], that choosing some bounded dual domain Π for π, instead of the unbounded $\mathbb{R}^n$, can offer a simpler convergence analysis for solving the problem. For example, choosing Π to be a bounded ball that includes an optimal Lagrangian multiplier π^*. In what follows, we are also going to select more restricted domains for v and π to streamline our analysis. Second, if we regard v as the dual variable to π, (Equation 3.49) and (Equation 3.50) represent an additional conjugation to the bi-conjugate, giving rise to the name tri-conjugate reformulation. In this case, we have

$$g(y) = \max_{\pi \in \Pi} \langle \pi, y \rangle \quad \underbrace{- \quad g^*(\pi).}_{:= \min_{v \in V} -\langle \pi, v \rangle + g(v)} \quad .$$

A comparison between the tri-conjugate and the bi-conjugate reformulations is depicted in Figure Figure 3.2. Because of their close relationship, we use the common notation $\mathcal{L}_g$ for both, but emphasize the tri-conjugate reformulation by the auxiliary primal variable v. The key advantage of $\mathcal{L}_g(y,v;\pi)$ for us is the implementability of the prox-mappings for both v and π with $\|\cdot\|^2$ as the prox-function, which is crucial for the nSSD method.

Returning to the two-layer problem in (Equation 3.4), the nested linearization is given

by

$$\min_{(x,v_1)\in X\times V_1} \max_{(\pi_1,\pi_2)\in\tilde{\Pi}_1\times\Pi_2} \{\mathcal{L}(x,v_1;\pi_1,\pi_2) := \mathcal{L}_1(x,v_1;\pi_1,\pi_2) + u(x)\}, \tag{3.51}$$

$$\text{where } \mathcal{L}_1(x,v_1;\pi_1,\pi_2) := \pi_1[\mathcal{L}_2(x;\pi_2) - v_1] + f_1(v_1), \text{ and } \mathcal{L}_2(x;\pi_2) := \pi_2 x - f_2^*(\pi_2).$$

Here, the reformulation for the inner function f_2 utilizes the bi-conjugate with $\Pi_2 = \text{dom}(f^*)$. The reformulation for the outer layer function f_1 utilizes the tri-conjugate in (Equation 3.49) and (Equation 3.50) because of its stochastic argument. To streamline the convergence analysis, we restrict the domain for v_1 and π_1 to some compact convex sets V_1 and $\tilde{\Pi}_1$, respectively. We require both sets to be relatively simple such that Euclidean projections onto them can be efficiently computed. Additionally, we also require them to be large enough to contain some important points and to have finite radii D_{V_1} and $\hat{M}_1$:

$$\begin{aligned}
&\max_{v_1,\bar{v}_1} \|v_1 - \bar{v}_1\| \le D_{V_1} \text{ and } f_2(x^*) \in V_1.\\
&\max_{\pi_1\in\tilde{\Pi}_1} \|\pi_1\| \le \hat{M}_1,\ \{s \in \partial f_1(y_1), y_1 \in \mathbb{R}^{n_1}\} \subset \tilde{\Pi}_1, \text{ and } \tilde{\Pi}_i \subset \mathbb{R}^{n_1}_+ \text{ if } f_2 \text{ is non-affine.}
\end{aligned} \tag{3.52}$$

For example, if $R_1 \ge \max_{x\in X} \|f_2(x)\|$, V_1 can be chosen to be an Euclidean ball, $\{v_1 \in \mathbb{R}^{n_1} |\ \|v_1\| \le R_1\}$. If f_1 is M_1-Lipschitz continuous, $\tilde{\Pi}_1$ can be chosen to be $\{\pi_1 \in \mathbb{R}^{n_1} |\ \|\pi_1\| \le M_1\}$. Furthermore, when Assumption 2 requires f_1 to be monotone, we restrict $\tilde{\Pi}_1$ to the positive orthant, i.e., $\tilde{\Pi}_1 = \{\pi_1 \in \mathbb{R}^{n_1} |\ \|\pi_1\| \le M_1\} \cap \mathbb{R}^{n_1}_+$.

We present a few properties of the nested linearization reformulation (Equation 3.51) and a Q-gap function, which will lead to the nSSD method. The next Lemma relates (Equation 3.51) to the original problem in (Equation 3.4). It is a counterpart to Lemma 4.1.

Lemma 3.3 *The following relations between $\mathcal{L}$ in* (Equation 3.51) *and f in* (Equation 3.4) *are valid.*

a) If $v_1^* := f_2(x^*)$, then $\mathcal{L}(x^*, v_1^*; \pi_1, \pi_2) \le f(x^*)\ \forall(\pi_1 \times \pi_2) \in \tilde{\Pi}_1 \times \Pi_2$.

b) Given a pair $(x, v_1) \in X \times V_1$, $\mathcal{L}(x, v_1; \hat{\pi}_1, \hat{\pi}_2) \geq f(x)$ *if* $\hat{\pi}_1 \in \partial f_1(f_2(x))$ *and* $\hat{\pi}_2 \in \partial f_2(x)$.

c) Thus for any $z := (x, v_1; \pi_1, \pi_2) \in Z$, $f(x) - f(x^*) \leq \mathcal{L}(x, v_1; \hat{\pi}_1, \hat{\pi}_2) - \mathcal{L}(x^*, v_1^*; \pi_1, \pi_2)$ *if* $\hat{\pi}_1 \in \partial f_1(f_2(x))$, $\hat{\pi}_2 \in \partial f_2(x)$, *and* $v_1^* = f_2(x^*)$.

Proof: For Part a), we have

$$\mathcal{L}(x^*, v_1^*; \pi_1, \pi_2) = \pi_1[\mathcal{L}_2(x^*; \pi_2) - v_1^*] + f_1(v_1^*) + u(x^*) \leq f_1(f_2(x^*)) + u(x^*) = f(x^*).$$

Here, if f_2 is affine, $\mathcal{L}_2(x^*; \pi_2) = f_2(x^*) = v_1^*$ such that $\pi_1[\mathcal{L}_2(x^*; \pi_2) - v_1^*] = 0$. Otherwise, π_1 is non-negative, and $\mathcal{L}_2(x^*; \pi_2) \leq f_2(x^*) = v_1^*$ such that $\pi_1[\mathcal{L}_2(x^*; \pi_2) - v_1^*] \leq 0$.

Part b) holds because the nested linearization reformulation is always large than the nested Lagrangian reformulation, i.e.,

$$\begin{aligned}
\mathcal{L}_1(x, v_1; \hat{\pi}_1, \hat{\pi}_2) &= \hat{\pi}_1[\mathcal{L}_2(x; \hat{\pi}_2) - v_1] + f_1(v_1) = \hat{\pi}_1 \mathcal{L}_2(x; \hat{\pi}_2) - [\hat{\pi}_1 v_1 - f_1(v_1)] \\
&\geq \hat{\pi}_1 \mathcal{L}_2(x; \hat{\pi}_2) - \max_{v_1 \in \mathbb{R}^{n_1}} [\hat{\pi}_1 v_1 - f_1(v_1)] \\
&= \hat{\pi}_1 \mathcal{L}_2(x; \hat{\pi}_2) - f_1^*(\hat{\pi}_1) = \mathcal{L}_1(x; \hat{\pi}_1, \hat{\pi}_2) = f_1(f_2(x)),
\end{aligned}$$

where $\mathcal{L}_1(x; \hat{\pi}_1, \hat{\pi}_2)$ is defined in (Equation 3.8) and the last inequality follows from Lemma 4.1.b). Part c) is a direct consequence of Part a) and Part b). □

To simplify notation, we will use $Z := X \times V_1 \times \tilde{\Pi}_1 \times \Pi_2$ and $z := (x, v_1; \pi_1, \pi_2)$ for the rest of the section. Lemma 3.3.c) suggests a Q-gap function for algorithm design. Given a point $z^t := (x^t, v_1^t; \pi_1^t, \pi_2^t) \in Z$, the Q-gap function with respect to a reference point $z \in Z$ is given by

$$Q(z^t, z) := \mathcal{L}(x^t, v_1^1; \pi_1, \pi_2) - \mathcal{L}(x, v_1; \pi_1^t, \pi_2^t). \tag{3.53}$$

It can be decomposed into component Q-gap functions given by

$$Q(z^t, z) = Q_2(z^t, z) + Q_1(z^t, z) + Q_1^v(z^t, z) + Q_0(z^t, z), \tag{3.54}$$

$$\begin{aligned} Q_2(z^t, z) &:= \mathcal{L}(x^t, v_1^t; \pi_1, \pi_2) - \mathcal{L}(x^t, v_1^t; \pi_1, \pi_2^t) \\ &= \pi_1[\pi_2 x^t - f_2^*(\pi_2)] \boxed{-\pi_1[\pi_2^t x^t - f_2^*(\pi_2^t)]}, \end{aligned} \tag{3.55}$$

$$\begin{aligned} Q_1(z^t, z) &:= \mathcal{L}(x^t, v_1^t; \pi_1, \pi_2^t) - \mathcal{L}(x^t, v_1^t; \pi_1^t, \pi_2^t) \\ &= \pi_1[\mathcal{L}_2(\pi_2^t; x^t) - v_1^t] \boxed{-\pi_1^t[\mathcal{L}_2(\pi_2^t; x^t) - v_1^t]}, \end{aligned} \tag{3.56}$$

$$Q_1^v(z^t, z) := \mathcal{L}(x^t, v_1^t; \pi_1^t, \pi_2^t) - \mathcal{L}(x^t, v_1; \pi_1^t, \pi_2^t) = \boxed{f_1(v_1^t) - \pi_1^t v_1^t} - [f_1(v_1) - \pi_1^t v_1], \tag{3.57}$$

$$Q_0(z^t, z) := \mathcal{L}(x^t, v_1; \pi_1^t, \pi_2^t) - \mathcal{L}(x, v_1; \pi_1^t, \pi_2^t) = \boxed{\pi_1^t \pi_2^t x^t + u(x^t)} - [\pi_1^t \pi_2^t x + u(x)], \tag{3.58}$$

where Q_2, Q_1, Q_1^v, and Q_0 relate to the optimality of π_2^t, π_1^t, v_1^1 and x^t, respectively.

We now present the nSSD method, beginning with the simpler setting where both f_1 and f_2 are deterministic. For this setting, we propose a non-smooth sequential dual (nSD) method. Each nSD iteration employs prox-mappings to sequentially minimize Q_2, Q_1, Q_1^v, and Q_0:

$$\begin{aligned} &\pi_2^t \leftarrow \arg\max_{\pi_2 \in \Pi_2} \pi_2 x^{t-1} - f_2^*(\pi_2); \\ &\pi_1^t \leftarrow \arg\max_{\pi_1 \in \tilde{\Pi}_1} \pi_1[\tilde{y}_1^t - v_1^{t-1}] - \tau_{1,t} \left\|\pi_1 - \pi_1^{t-1}\right\|^2 / 2, \text{ where } \tilde{y}_1^t := \mathcal{L}_2(x^{t-1}; \pi_2^t); \\ &v_1^t \leftarrow \arg\min_{v_1 \in V_1} \ \langle f_1'(v_1^{t-1}) - \pi_1^t, v_1 \rangle + \gamma_{1,t} \left\|v_1 - v_1^{t-1}\right\|^2 / 2; \\ &x^t \leftarrow \arg\min_{x \in X} \quad \tilde{y}_0^t x + u(x) + \eta_t \left\|x - x^{t-1}\right\|^2 / 2, \text{ where } \tilde{y}_0^t := \pi_1^t \pi_2^t. \end{aligned} \tag{3.59}$$

There are two simplifications compared to the SD method (c.f. (Equation 3.16)). First,

rather than momentum-extrapolated prediction terms, values from the last iterate, i.e., x^{t-1} and $\mathcal{L}_2(x^{t-1};\pi_2^t)$ (or $\tilde{x}^t$ and $\tilde{y}_1^t$ in (Equation 3.16) with $\theta_t = 0$), are used as arguments for the π_2^t and π_1^t-prox mappings. Second, the π_2^t-prox mapping is implemented with $\tau_{2,t} = 0$. These simplifications are justified because the $\mathcal{O}(1/\epsilon^2)$ oracle complexity can not be improved by the momentum-extrapolated acceleration even for the simple deterministic nonsmooth problem. Additionally, notice that a linear approximation of f_1, $\langle f_1'(v_1^{t-1}), \cdot \rangle$, is utilized when performing the prox-mapping for v_1^t.

We adapt the deterministic nSD algorithm (Equation 3.59) to the nested stochastic setting to obtain the nSSD method shown in Algorithm 5. First, Line 2 of Algorithm 5 queries the stochastic oracle of f_2 to obtain unbiased estimators at x^{t-1}. This is because the π_2^t prox-mapping in (Equation 3.59) corresponds to the gradient evaluation, $\pi_2^t = f_2'(x^{t-1})$, (see Lemma 3.2 with $\tau_{2,t} = 0$). Next, except for replacing the argument $\tilde{y}_1^t$ with a stochastic estimator $\tilde{y}_1^t(\xi)$, Line 3 of Algorithm 5 is the same as the π_1^t prox-mapping in (Equation 3.59). In particular, the unbiased estimator $\tilde{y}_1^t(\xi)$ is obtained from the function value estimator $f_2(x^{t-1}, \xi_2^1)$ because

$$\mathcal{L}_2(x^{t-1};\pi_2^t) := f_2^*(\pi_2^t) + \pi_2^t x^{t-1} = [f_2(x^{t-1}) - f_2'(x^{t-1})x^{t-1}] + f_2'(x^{t-1})x^{t-1} = f_2(x^{t-1}).$$

Next, Line 4 of Algorithm 5 is the same as the v_1^t prox-mapping in (Equation 3.59) except for using the stochastic subgradient estimator $f_1'(v_1^{t-1}, \xi_1)$ in place of the deterministic subgradient $f_1'(v_1^{t-1})$. Lastly, except for replacing the argument $\tilde{y}_0^t$ with an unbiased stochastic estimator $\tilde{y}_0^t(\xi)$, Line 5 of Algorithm 5 is the same as the x^t prox-mapping in (Equation 3.59). Here, similar to the SSD Algorithm 4, an independently sampled $f_2'(x^{t-1}, \xi_2^0)$, which is conditionally independent of π_1^t, is used to construct an unbiased estimator for $\pi_1^t \pi_2^t$.

We present the convergence properties of the proposed nSSD method. First, we need to specify a few problem parameters. The Lipschitz-continuity constants, M_1 and M_2, and

Algorithm 5 Stochastic Sequential Dual-primal (nSSD) Method for Non-Smooth Two Layer Problem

Input: $x^0 \in X$, $v_1^0 \in V_1$, $\pi_1^0 \in \tilde{\Pi}_1$.

1: **for** $t = 1, 2, 3...N$ **do**

2: Call $\mathcal{SO}_2$ to obtain independent estimates $\{f_2(x^{t-1}, \xi_2^1), f_2'(x^{t-1}, \xi_2^0)\}$.

3: Compute $\pi_1^t \leftarrow \arg\max_{\pi_1 \in \tilde{\Pi}_1} \pi_1[\tilde{y}_1^t(\xi) - v_1^{t-1}] - \tau_{1,t} \left\|\pi_1 - \pi_1^{t-1}\right\|^2 /2$ where $\tilde{y}_1^t(\xi) := f_2(x^{t-1}, \xi_2^1)$.

4: Compute $v_1^t \leftarrow \arg\min_{v_1 \in V_1} \langle f_1'(v_1^{t-1}, \xi_1) - \pi_1^t, v_1\rangle + \gamma_{1,t} \left\|v_1 - v_1^{t-1}\right\|^2 /2$.

5: Set $\tilde{y}_0^t(\xi) \leftarrow \pi_1^t f_2'(x^{t-1}, \xi_2^0)$. Compute $x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t(\xi)x + u(x) + \eta_t \left\|x - x^{t-1}\right\|^2 /2$.

6: **end for**

7: Return $\bar{x}^N = \sum_{t=1}^N \omega_t x^t / (\sum_{t=1}^N \omega_t)$.

the variance constants, σ_1, σ_{f_2} and σ_2, associated with the layer functions are defined as

$$M_1 := \max_{y_1 \in \mathbb{R}^{n_1}} \left\|f_1'(y_1)\right\|, \; M_2 := \max_{x \in X} \left\|f_2'(x)\right\|,$$

$$\mathbb{E}[\left\|f_1'(y_1, \xi) - f_1'(y_1)\right\|^2] \leq \sigma_1^2 \; \forall y_1 \in \mathbb{R}^{n_1},$$

$$\mathbb{E}[\left\|f_2'(x, \xi) - f_2'(x)\right\|^2] \leq \sigma_2^2 \text{ and } \mathbb{E}[\left\|f_2(x, \xi) - f_2(x)\right\|^2] \leq \sigma_{f_2}^2 \leq \mathcal{D}_X^2 \sigma_2^2 \; \forall x \in X. \tag{3.60}$$

The aggregate variance constants of the stochastic arguments in Algorithm 5 are defined as follows:

$$\tilde{\sigma}_1 := \{\max_{t \geq 0} \mathbb{E}[\left\|\tilde{y}_1^t(\xi) - \mathbb{E}[\tilde{y}_1^t(\xi)]\right\|^2]\}^{1/2}, \; \tilde{\sigma}_x := \{\max_{t \geq 0} \mathbb{E}[\left\|\tilde{y}_0^t(\xi) - \mathbb{E}[\tilde{y}_0^t(\xi)]\right\|^2]\}^{1/2}. \tag{3.61}$$

With these parameters defined, we can now state the convergence result. Its proof is deferred to subsection 3.3.3.

Theorem 3.5 *Consider a two-layer problem (c.f.* (Equation 3.4)*) with stochastic non-smooth functions f_1 and f_2. Let their problem parameters be defined in* (Equation 3.60)*, and the radius constants of the tri-conjugate reformulation of f_1 be defined in* (Equation 3.52)*. If the solution iterates $\{x^t\}$ are generated by Algorithm 5, the variance*

constants in (Equation 3.61) *for the aggregate stochastic estimators in the algorithm satisfy* $\tilde{\sigma}_1 \leq \sigma_{f_2} \leq \sigma_2 \mathcal{D}_X$ *and* $\tilde{\sigma}_x \leq \hat{M}_1 \sigma_2$. *Moreover, if the stepsizes are given by*

$$\begin{aligned} &\omega_t = 1,\ \eta_t = 2\max\{4\hat{M}_1 M_2, \tilde{\sigma}_x\}\sqrt{t}/\mathcal{D}_X,\ \tau_{1,t} = \sqrt{t}\tilde{\sigma}_1/\hat{M}_1, \\ &\gamma_{1,t} = 2\max\{4\hat{M}_1, \sigma_1\}\sqrt{t}/D_{V_1}, \end{aligned} \tag{3.62}$$

the ergodic average solution $\bar{x}^N$ *satisfies*

$$\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq \tfrac{1}{\sqrt{N}}\{8\hat{M}_1 M_2 \mathcal{D}_X + \tilde{\sigma}_x \mathcal{D}_X + 5\hat{M}_1\tilde{\sigma}_1 + 8\hat{M}_1 D_{V_1} + \sigma_1 D_{V_1}\}. \tag{3.63}$$

Three remarks are in order regarding the result. First, when estimating $f_2(x^*)$ is difficult and the total iteration number N is known beforehand, we can set V_1 to encompass the entire $\mathbb{R}^{n_1}$ rather than being confined to a bounded set. In this case, we choose $V_1 = \mathbb{R}^{n_1}$ and $\gamma_{1,t} = 2\max\{4\hat{M}_1, \sigma_1\}\sqrt{N}/(M_2\mathcal{D}_X)$, and keep η_t and $\tau_{1,t}$ to be the same as those in (Equation 3.62). The convergence bound in (Equation 3.63) would still be valid, and it can be simplified further to

$$\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq \mathcal{O}\{(\hat{M}_1 + \sigma_1)(M_2 + \sigma_2)\mathcal{D}_X/\sqrt{N}\}.$$

Second, the parameter-free stepsize choices with $\eta_t = \Theta(\sqrt{t})$, $\gamma_{1,t} = \Theta(\sqrt{t})$ and $\tau_{1,t} = \Theta(\sqrt{t})$ can lead to a stochastic oracle complexity of $\mathcal{O}(1/\epsilon^2)$, which is order-optimal. Third, the bound in (Equation 3.63) is order-optimal because the term $\mathcal{O}(\hat{M}_1\sigma_{f_2}/\sqrt{N})$ is not improvable even for strongly convex problems. This fact will be illustrated in the next Subsection.

3.3.2 Lower Complexity Bound

We develop a lower stochastic oracle complexity bound for the strongly convex NSCO problem with a (general) nonsmooth outer-layer function. At first glance, such a develop-

ment might appear unnecessary because Theorem 3.4 already establishes the $O(1/\epsilon^2)$ lower bound for the easier problem with a structured non-smooth outer-layer function. However, upon closer examination, we see Theorem 3.4 is not applicable because the abstract computation scheme in (Equation 3.34) does not cover the nSSD Algorithm 5. Specifically, the π_1^t generated by (Equation 3.34) contains only sub-gradient information. In contrast, for Algorithm 5, π_1^t incorporates both residue information, $v_1 - f_2(x)$ and subgradient information (see Line 3 and 4 in Algorithm 5). This arises because π_1 serves the dual purpose of acting as the multiplier for the constrained problem and as the conjugate dual variable in the tri-conjugate reformulation.

This motivates us to propose a more general abstract scheme to show the $\mathcal{O}(1/\epsilon^2)$ oracle-complexity for nSSD to be unimprovable. For simplicity, we assume $\tilde{\Pi}_1 = \mathbb{R}_+^{n_1} \cap \{\pi_1 \in \mathbb{R}^{n_1} | \, \|\pi_1\| \leq M_1\}$, and V_1 and X being rotationally invariant, i.e., zero-centered balls. The abstract scheme consists of the following steps. Initially, the scheme is provided with inputs $x^0 \in X$, $v_0^1 \in V_1$, and $\pi_1^0 \in \tilde{\Pi}_1$. It then initializes the affine subspaces to $\mathcal{X}^0 := \text{span}(x^0)$, $V_1^0 := v_1^0$, and $\mathcal{M}_1^t = \text{span}(\pi_1^0)$. In the t^{th} iteration, the updates are given by (the summations below represent the Minkowski sum and $[\cdot]_{\tilde{\Pi}_1}^+$ denotes the projection onto $\tilde{\Pi}_1$)

$$
\begin{aligned}
&\textit{Query } \mathcal{SO}_2 \textit{ to obtain } (f_2(y_2^t, \xi_2^t), f_2'(y_2^t, \xi_2^t)) \textit{at some } y_2^t \in \mathcal{X}^{t-1};\\
&\mathcal{M}_1^t := \textit{span}([\tilde{\mathcal{M}}_\infty]_{\tilde{\Pi}_1}^+) + \tilde{\mathcal{M}}_\infty \textit{ where}\\
&\qquad \tilde{\mathcal{M}}_\infty := \textit{span}\{f_2(y_2^t, \xi_2^t) + \sum_{j=1}^{t} f_2'(y_2^j, \xi_2^j) x^j - v_1^0 : x^j \in \mathcal{X}^{t-1}\}\\
&\qquad\qquad\qquad + \textit{span}\{f_1'(\pi_1 + v_1^0) : \pi_1 \in \mathcal{M}_1^{t-1}\} + \mathcal{M}_1^{t-1};\\
&\mathcal{X}^t := \{f_2'(y_2^j, \xi_2^j)^\top \pi_1^\top : j \leq t, \pi_1 \in \Pi_1^t\} + \mathcal{X}^{t-1}.
\end{aligned}
\tag{3.64}
$$

After N iterations, the scheme outputs some $x^N \in \mathcal{X}^N$.

We make three remarks for the abstract scheme in (Equation 3.64). First, the $\text{span}([\tilde{\mathcal{M}}_\infty]_{\tilde{\Pi}_1}^+)$ operation, employed for the $\mathcal{M}_1^t$ computation, is unusual among lower

complexity models (see [1]). However, it is necessary in this case because $\tilde{\Pi}_1 \subset \mathbb{R}^{n_1}_+$ is not rotationally invariant. Second, the nSSD method can be implemented by the abstract scheme. This can shown by observing that π_1^t and v_1^t in Algorithm 5 satisfy $\pi_1^t \in \mathcal{M}_1^t$ and $v_1^t \in v_1^0 + \mathcal{M}_1^t$, respectively. Third, the abstract scheme still accommodates the SCGD-type algorithms [48], where x is updated with pseudo-gradients of the form $\{f_1'(y_1)f_2'(x^j, \xi^j)\}$. This can be seen by noticing $\mathcal{M}_1^t + v_0^1$ contains all the convex combinations of $\{f_2(x^j, \xi_2)\}_{j=1}^t$ and $\mathcal{M}_1^t$ contains $\{f_1'(y_1) : y_1 \in \mathcal{M}_1^{t-1} + v_0^1\}$. Next, we state the lower complexity result for the abstract scheme and its proof is deferred to the Appendix.

Theorem 3.6 *Let the abstract computation scheme in* (Equation 3.64) *be initialized to* $x^0 = 0$, $\pi_1^0 = 0$. *Given problem parameters* $\sigma_{f_2} \geq 0$, $M_1 \geq 0$, $\epsilon > 0$ *and* $\alpha \leq M_1^2/(4\epsilon)$, *there exists a nested two-layer problem* (Equation 3.4) *consisting of a general nonsmooth* f_1 *and a stochastic linear* f_2 *such that* f_1 *is* M_1*-Lipschitz continuous, the variance of* $f_2(x, \xi)$ *is bounded by* σ_{f_2} *(c.f.* (Equation 3.60)*), and* $u(x) = \alpha \|x\|^2 /2$. *Moreover, for some* v_1^0, *the abstract scheme in* (Equation 3.64) *requires at least* $N \geq \Omega(\tilde{M}_1^2 \sigma_{f_2}^2/\epsilon^2)$ *to find an* ϵ*-optimal solution, i.e.,* $f(x^N) - f(x^*) \leq \epsilon$.

3.3.3 Convergence Proofs

We provide detailed convergence proof for Theorem 3.5. Let us begin with a general Q-gap convergence bound.

Proposition 3.2 *Consider a two-layer problem (c.f.* (Equation 3.4)*) with stochastic nonsmooth functions* f_1 *and* f_2. *Let its problem parameters be defined in* (Equation 3.60), (Equation 3.52), (Equation 3.61). *Suppose iterates* $\{z^t := (x^t, v_1^t; \pi_1^t, \pi_2^t)\}$ *are generated by Algorithm 5 and the stepsize choice satisfy the following conditions:*

$$\omega_t \eta_t \geq \omega_{t-1}\eta_{t-1},\ \omega_t \tau_{1,t} \geq \omega_{t-1}\tau_{1,t-1},\ \omega_t \gamma_{1,t} \geq \omega_{t-1}\gamma_{1,t-1}, \forall t \geq 2. \tag{3.65}$$

Then for any reference point $z := (x^*, v_1^*; \hat{\pi}_1, \hat{\pi}_2)$*, where* $v_1^* := f_2(x^*)$ *and* $(\hat{\pi}_1, \hat{\pi}_2) \in \tilde{\Pi}_1 \times \Pi_2$ *could potentially depend on* $\{z^t\}$*, we have*

$$\begin{aligned}\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q(z^t; z)] \leq & (8\hat{M}_1^2 M_2^2 + \tilde{\sigma}_x^2) \textstyle\sum_{t=1}^N \frac{\omega_t}{\eta_t} + \frac{\omega_N \eta_N \mathcal{D}_X^2}{2} \\ & + \tilde{\sigma}_1^2 \textstyle\sum_{t=1}^N \frac{\omega_t}{\tau_{1,t}} + 2\omega_N \tau_{1,N} \hat{M}_1^2 + \hat{M}_1 \tilde{\sigma}_1 \sqrt{\textstyle\sum_{t=1}^N \omega_t^2} \\ & + (8\hat{M}_1^2 + \sigma_1^2) \textstyle\sum_{t=1}^N \frac{\omega_t}{\gamma_{1,t}} + \frac{\omega_N \gamma_{1,N} D_{V_1}^2}{2}.\end{aligned} \tag{3.66}$$

Proof: We begin by developing a bound for Q_1 and Q_1^v. Line 3 of Algorithm 5 implies

$$(\pi_1^t - \hat{\pi}_1)[v_1^{t-1} - f_2(x^{t-1}, \xi_2^1)] + \tfrac{\tau_{1,t}}{2}(\left\|\pi_1^t - \hat{\pi}_1\right\|^2 + \left\|\pi_1^t - \pi_1^{t-1}\right\|^2 - \left\|\pi_1^{t-1} - \pi_1\right\|^2) \leq 0.$$

Denoting $\delta_1^t := f_2(x^{t-1}, \xi_2^1) - f_2(x^{t-1})$, we get

$$(\pi_1^t - \hat{\pi}_1)v_1^{t-1} = (\pi_1^t - \hat{\pi}_1)v_1^t - (\pi_1^t - \hat{\pi}_1)[v_1^t - v_1^{t-1}] \geq (\pi_1^t - \hat{\pi}_1)v_1^t - 2\hat{M}_1 \left\|v_1^t - v_1^{t-1}\right\|,$$

and

$$\begin{aligned}&\mathbb{E}[(\pi_1^t - \hat{\pi}_1) f_2(x^{t-1}, \xi_2^1)] = \mathbb{E}[(\pi_1^t - \hat{\pi}_1)\delta_1^t] + \mathbb{E}[(\pi_1^t - \hat{\pi}_1)\mathcal{L}_2(x^{t-1}; \pi_2^t)] \\ =&\mathbb{E}[\pi_1^t \delta_1^t] - \mathbb{E}[\hat{\pi}_1 \delta_1^t] + \mathbb{E}[(\pi_1^t - \hat{\pi}_1)\{\mathcal{L}_2(x^{t-1}; \pi_2^t) - \mathcal{L}_2(x^t; \pi_2^t)\}] + (\pi_1^t - \hat{\pi}_1)\mathcal{L}_2(x^t; \pi_2^t)] \\ =&-\mathbb{E}[\hat{\pi}_1 \delta_1^t] + \mathbb{E}[\pi_1^t \delta_1^t] + \mathbb{E}[(\pi_1^t - \hat{\pi}_1)\pi_2^t(x^{t-1} - x^t)] + \mathbb{E}[(\pi_1^t - \hat{\pi}_1)\mathcal{L}_2(x^t; \pi_2^t)] \\ \leq&-\mathbb{E}[\hat{\pi}_1 \delta_1^t] + \tilde{\sigma}_1^2/\tau_{1,t} + 2\hat{M}_1 M_2 \mathbb{E}[\left\|x^{t-1} - x^t\right\|] + \mathbb{E}[(\pi_1^t - \hat{\pi}_1)\mathcal{L}_2(x^t; \pi_2^t)].\end{aligned}$$

In the last inequality, the bound for $\mathbb{E}[\pi_1^t \delta_1^t]$ follows from Lemma 3.7 ($\mathbb{E}[\delta_1^t | \pi_1^{t-1}] = 0$), and the bound for $\mathbb{E}[(\pi_1^t - \hat{\pi}_1)\pi_2^t(x^{t-1} - x^t)]$ follows from $\max\{\|\pi_1^t, \hat{\pi}_1^t\|\} \leq \hat{M}_1$. Thus we get

$$\begin{aligned}&\mathbb{E}[Q_1^v(z^t; z) + \tfrac{\tau_{1,t}}{2}(\left\|\pi_1^t - \hat{\pi}_1\right\|^2 - \left\|\pi_1^{t-1} - \pi_1\right\|^2)] \\ &\quad \leq -\mathbb{E}[\hat{\pi}_1 \delta_1^t] + \tilde{\sigma}_1^2/\tau_{1,t} + 2\hat{M}_1 M_2 \mathbb{E}[\left\|x^{t-1} - x^t\right\|] + 2\hat{M}_1 \left\|v_1^t - v_1^{t-1}\right\|.\end{aligned} \tag{3.67}$$

Next, Line 4 of Algorithm 5 implies that

$$\langle v_1^t - v_1^*, f_1'(v_1^{t-1}, \xi_1) - \pi_1^t \rangle + \tfrac{\gamma_{1,t}}{2}(\left\|v_1^t - v_1^*\right\|^2 + \left\|v_1^t - v_1^{t-1}\right\|^2 - \left\|v_1^{t-1} - v_1^*\right\|^2) \leq 0,$$

whereby

$$\begin{aligned}
&\mathbb{E}[\langle v_1^t - v_1^*, f_1'(v_1^{t-1}, \xi_1)\rangle] \\
&= \mathbb{E}[\langle v_1^t - v_1^*, f_1'(v_1^{t-1}, \xi_1) - f_1'(v_1^{t-1})\rangle] + \mathbb{E}[\langle v_1^{t-1} - v_1^*, f_1'(v_1^{t-1})\rangle + \langle v_1^t - v_1^{t-1}, f_1'(v_1^t)\rangle] \\
&\quad + \mathbb{E}[\langle v_1^t - v_1^{t-1}, f_1'(v_1^{t-1}) - f_1'(v_1^t)\rangle] \\
&\geq \mathbb{E}[f_1(v_1^t) - f_1(v_1^*)] - \sigma_1^2/\gamma_{1,t} - \mathbb{E}[2\left\|v_1^t - v_1^{t-1}\right\| \hat{M}_1].
\end{aligned}$$

In the last inequality, we utilized the convexity of f_1 and the bound for $\mathbb{E}[\langle v_1^t, f_1'(v_1^{t-1}, \xi_1) - f_1'(v_1^{t-1})\rangle]$ follows from Lemma 3.7. Thus we get

$$\begin{aligned}
&\mathbb{E}[Q_1(z^t; z)] + \mathbb{E}[\tfrac{\gamma_{1,t}}{2}(\left\|v_1^t - v_1^*\right\|^2 + \left\|v_1^t - v_1^{t-1}\right\|^2 - \left\|v_1^{t-1} - v_1^*\right\|^2)] \\
&\quad \leq \sigma_1^2/\gamma_{1,t} + \mathbb{E}[2\left\|v_1^t - v_1^{t-1}\right\| \hat{M}_1].
\end{aligned} \tag{3.68}$$

Combine (Equation 3.67) and (Equation 3.68) and apply Young's inequality, $\mathbb{E}[-\gamma_{1,t}\left\|v_1^t - v_1^{t-1}\right\|^2/2 + 4\hat{M}_1\left\|v_1^t - v_1^{t-1}\right\|] \leq 8\hat{M}_1^2/\gamma_{1,t}$. We get

$$\begin{aligned}
&\mathbb{E}[Q_1^v(z^t; z) + Q_1(z^t; z) + \tfrac{\gamma_{1,t}}{2}(\left\|v_1^t - v_1^*\right\|^2 - \left\|v_1^{t-1} - v_1^*\right\|^2)] \\
&\qquad + \mathbb{E}[\tfrac{\tau_{1,t}}{2}(\left\|\pi_1^t - \hat{\pi}_1\right\|^2 - \left\|\pi_1^{t-1} - \pi_1\right\|^2)] \\
&\quad \leq -\mathbb{E}[\hat{\pi}_1\delta_1^t] + 8\hat{M}_1^2/\gamma_{1,t} + \tilde{\sigma}_1^2/\tau_{1,t} + \sigma_1^2/\gamma_{1,t} + 2\hat{M}_1 M_2 \mathbb{E}[\left\|x^{t-1} - x^t\right\|].
\end{aligned}$$

Then we take an ω_t-weighted sum of the above inequality. Using $\mathbb{E}[\|v_1^t - v_1^*\|^2] \leq D_{V_1}^2$, the stepsize requirements in (Equation 3.65) lead to $\mathbb{E}[\sum_{t=1}^N \omega_t\gamma_{1,t}(\left\|v_1^t - v_1^*\right\|^2 - \left\|v_1^{t-1} - v_1^*\right\|^2]/2 \leq \omega_N\gamma_{N,t}D_{V_1}^2/2$. The Cauchy-Schwartz inequality (see (Equation 3.44))

allows us to simplify $\mathbb{E}[\sum_{t=1}^{N} \omega_t \hat{\pi}_1 \delta_1^t]$. As a result, we get

$$\begin{aligned}
&\mathbb{E}[\textstyle\sum_{t=1}^{N} \omega_t \{Q_1^v(z^t; z) + Q_1(z^t; z)\}] \\
&\quad \leq \omega_N(\gamma_{N,t} D_{V_1}^2/2 + 2\tau_{N,t}\hat{M}_1^2) + \textstyle\sum_{t=1}^{N} \omega_t [8\hat{M}_1^2/\gamma_{1,t} + \tilde{\sigma}_1^2/\tau_{1,t} + \sigma_1^2/\gamma_{1,t}] \\
&\quad + 2\hat{M}_1 M_2 \mathbb{E}[\textstyle\sum_{t=1}^{N} \omega_t \left\|x^{t-1} - x^t\right\|] + \hat{M}_1 \sigma_1 \sqrt{\textstyle\sum_{t=1}^{N} \omega_t^2}.
\end{aligned} \tag{3.69}$$

Define $\pi_2^t = f_2'(x^{t-1})$. We get from Line 2 of Algorithm 5 that $(\pi_2^t - \hat{\pi}_2)x^{t-1} - (f_2^*(\pi_2^t) - f_2^*(\hat{\pi}_2)) \leq 0$, so that

$$Q_2(z^t; z) \leq \hat{\pi}_1(\pi_2^t - \hat{\pi}_2)(x^t - x^{t-1}) \leq \hat{M}_1 M_2 \left\|x^{t-1} - x^t\right\|.$$

Summing up the inequality with weight ω_t and taking expectation, we obtain

$$\mathbb{E}[\textstyle\sum_{t=1}^{N} \omega_t Q_2(z^t; z)] \leq 2\hat{M}_1 M_2 \mathbb{E}[\textstyle\sum_{t=1}^{N} \omega_t \left\|x^{t-1} - x^t\right\|]. \tag{3.70}$$

Define $\delta_x^t = \pi_1^t f_2(x^{t-1}, \xi_2^0) - \pi_1^t \pi_2^t$. We get from Line 5 of Algorithm 5 (see Lemma 3.6 in [28]) that

$$\langle x^t - x^*, \pi_1^t \pi_2^t + \delta_x^t \rangle + \tfrac{\eta_t}{2}(\left\|x^t - x^*\right\|^2 + \left\|x^t - x^{t-1}\right\|^2) \leq \tfrac{\eta_t}{2} \left\|x^{t-1} - x^*\right\|^2.$$

Since $\mathbb{E}[\delta_x^t | x^{t-1}] = 0$ and $\mathbb{E}[\delta_x^t | x^*] = 0$, we get $\mathbb{E}[\langle \delta_x^t, x^t \rangle] \leq \tilde{\sigma}_x^2/\eta_t$ from Lemma 3.7 and $\mathbb{E}[\langle \delta_x^t, x^* \rangle] = 0$. Then the stepsize condition in (Equation 3.65) leads to

$$\mathbb{E}[\textstyle\sum_{t=1}^{N} \omega_t Q_0(z^t; z)] \leq \sum_{t=1}^{N} \omega_t \tilde{\sigma}_x^2/\eta_t + \omega_N \eta_N \mathcal{D}_X^2/2 - \mathbb{E}[\textstyle\sum_{t=1}^{N} \omega_t \eta_t \left\|x^{t-1} - x^t\right\|^2]. \tag{3.71}$$

Therefore, the desired Q-gap convergence bound in (Equation 3.66) can be deduced from adding up (Equation 3.69), (Equation 3.70) and (Equation 3.71), and applying Young's inequality.

□

Proof of Theorem 3.5 The bounds on $\tilde{\sigma}_1$ and $\tilde{\sigma}_x$ can be derived directly from the definition of $\tilde{y}_1^t(\xi)$ and $\tilde{y}_0^t(\xi)$ in Algorithm 5. Applying Lemma 3.3 with $\hat{\pi}_1^N \in \partial f_1(f_2(\bar{x}^N))$ and $\hat{\pi}_2^N \in \partial f_2(\bar{x}^N)$ (see (Equation 3.45)), we get

$$\begin{aligned}
&\textstyle\sum_{t=1}^N \omega_t (f(\bar{x}^N) - f(x^*)) \\
&\qquad \leq \textstyle\sum_{t=1}^N \omega_t \mathcal{L}(\frac{\sum_{t=1}^N \omega_t x^t}{\sum_{t=1}^N \omega_t}, \frac{\sum_{t=1}^N \omega_t v_1^t}{\sum_{t=1}^N \omega_t}; \hat{\pi}_1^N, \hat{\pi}_2^N) - \textstyle\sum_{t=1}^N \omega_t \mathcal{L}(x^*, v_1^*; \pi_1^t, \pi_2^t) \\
&\qquad \leq \textstyle\sum_{t=1}^N \omega_t \mathcal{L}(x^t, v_1^t; \hat{\pi}_1^N, \hat{\pi}_2^N) - \textstyle\sum_{t=1}^N \omega_t \mathcal{L}(x^*, v_1^*; \pi_1^t, \pi_2^t) \\
&\qquad = \textstyle\sum_{t=1}^N \omega_t Q(z^t; (x^*, v_1^*; \hat{\pi}_1^N, \hat{\pi}_2^N)).
\end{aligned}$$

Here, the second inequality follows from the joint convexity of $\mathcal{L}$ (Equation 3.51) with respect to (x, v_1). The desired function gap convergence bound in (Equation 3.63) can be derived from applying Proposition 3.2 and dividing both sides of the resulting inequality by $(\sum_{t=1}^N \omega_t)$. □

3.4 Multi-layer Problem

In this section, we extend the SSD and the nSSD methods proposed in the last two sections to the multi-layer NSCO problem:

$$\min_{x \in X} \{f(x) := f_1 \circ f_2 \circ \ldots \circ f_k(x) + u(x)\}. \tag{3.72}$$

We impose a multi-layer compositional convexity assumption similar to Assumption 2 throughout this section.

Assumption 3 *A nested function $f_1 \circ f_2 \circ \ldots \circ f_k(x)$ in* (Equation 3.72) *is said to satisfy the compositional convexity assumption if*

- *Every layer function $f_i : \mathbb{R}^{n_i} \to \mathbb{R}^{n_{i-1}}$ is proper closed and convex.*
- *If f_i is not affine, $\{\pi_1 \pi_2 \ldots \pi_{i-1} : \pi_j \in \partial f_j(y_j), y_j \in \mathbb{R}^{n_j}, j \leq i-1\} \subset \mathbb{R}_+^{n_{i-1}}$.*

The second assumption is an extension of the two-layer monotonicity assumption in Assumption 2. For a non-affine f_i, rather than requiring every outer-layer function to be component-wise non-negative, our assumption is slightly weaker, requiring only the product of all possible subgradients of outer-layer functions to be component-wise non-negative. This weaker assumption can be useful for handling the semi-deviation risk measure.

Our development consists of three parts. In subsection 3.4.1, we propose the SSD method for the smooth multi-layer problem. In subsection 3.4.2, we propose the nSSD method, which is designed to handle the general multi-layer problem arising from arbitrary compositions of different types of layer functions. Lastly, we provide a sketch of the convergence analysis in subsection 3.4.3.

3.4.1 Smooth Multi-layer Problem

In this subsection, we study the multi-layer problem composed only of smooth layer functions. Our development follows the same structure as that of subsection 3.2.1. We begin by introducing a nested Lagrangian reformulation, which gives rise to a Q-gap function. The Q-gap function in turn motivates a conceptual algorithm. By implementing the conceptual algorithm using the stochastic oracles, we derive the multilayer SSD method. Lastly, we offer convergence guarantees for the SSD method.

For notation simplicity, we will use $i:j$ to denote the indices $i, i+1, ..., j$, and use $i:$ as a shorthand for $i:k$. For example, $\pi_{i:j} := (\pi_i, \pi_{i+1}, \ldots, \pi_j)$ and $\Pi_{i:j} := \Pi_i \times \Pi_{i+1} \times \ldots \times \Pi_j$. The nested Lagrangian reformulation to (Equation 3.72) is given by

$$\min_{x \in X} \max_{\pi_{1:} \in \Pi_{1:}} \{\mathcal{L}(x; \pi_{1:}) := \mathcal{L}_1(x; \pi_{1:}) + u(x)\}, \tag{3.73}$$

$$\text{where } \mathcal{L}_i(x; \pi_{i:}) := \begin{cases} x & \text{if } i = k+1, \\ \pi_i \mathcal{L}_{i+1}(x; \pi_{i+1:}) - f_i^*(\pi_i) & \text{if } 1 \le i \le k. \end{cases}$$

Here $\Pi_i := \text{dom}(f_i^*)$ denotes the domain of π_i. Let $z := (x; \pi_{1:})$ and $Z := X \times \Pi_{1:}$ denote the collections of decision variables and of their domains. We can relate the reformulated problem (Equation 3.73) to the original problem (Equation 3.72) through the following duality result.

Lemma 3.4 *Consider f and $\mathcal{L}$ defined in* (Equation 3.72) *and* (Equation 3.73)*, respectively. For any $x \in X$, the following relations are valid.*

a) Weak Duality: $f(x) \geq \mathcal{L}(x; \pi_{1:}) \; \forall \pi_{1:} \in \Pi_{1:}$.

b) Strong Duality: $f(x) = \mathcal{L}(x; \hat{\pi}_{1:})$ if $\hat{\pi}_i \in \partial f_i(f_{i+1:}(x)) \; \forall i \in [k]$.

c) There exists some $(\pi_1^*, \pi_2^*, \ldots, \pi_k^*)$ such that $z^* := (x^*; \pi_{1:}^*) \in Z$ is a saddle point, i.e.,

$$\mathcal{L}(x^*; \pi_{1:}) \leq \mathcal{L}(x^*; \pi_{1:}^*) \leq \mathcal{L}(x; \pi_{1:}^*) \; \forall (x, \pi_{1:}) \in Z.$$

d) The optimality gap of x is upper bounded by $f(x) - f(x^*) \leq \max_{\bar{\pi}_{1:} \in \Pi_{1:}} \mathcal{L}(x; \bar{\pi}_{1:}) - \mathcal{L}(x^*; \pi_{1:}), \forall \pi_{1:} \in \Pi_{1:}$.

Proof: The proof is similar to that of Lemma 4.1. Part b) follows from the same argument. For Part a), let $\pi_{1:} \in \Pi_{1:}$ be given and let $\hat{\pi}_i \in \partial f_i(f_{i+1:}(x)) \; \forall i$ such that Part b) is valid, then we have

$$\begin{aligned} f(x) - \mathcal{L}(x; \pi_{1:}) &= \textstyle\sum_{j=1}^{k} [\mathcal{L}(x; \pi_{1:j-1}, \hat{\pi}_j, \hat{\pi}_{j+1:}) - \mathcal{L}(x; \pi_{1:j-1}, \pi_j, \hat{\pi}_{j+1:})] \\ &= \textstyle\sum_{j=1}^{k} \underbrace{\pi_{1:j-1}[(\hat{\pi}_j f_{j+1:}(x) - f_j^*(\hat{\pi}_j)) - (\pi_j f_{j+1:}(x) - f_j^*(\pi_j))]}_{A_j}. \end{aligned}$$

Here, the definition of $\hat{\pi}_i$ implies that $\hat{\pi}_j \in \arg\max_{\pi_j \in \Pi_j} \pi_j f_{j+1:}(x) - f_j^*(\pi_j)$. If f_j is affine, then we have $A_j = 0$ because Π_j is a singleton set and $\hat{\pi}_j = \pi_j$. Otherwise, Assumption 3 implies the non-negativity of $\pi_{1:j-1}$, so we have $A_j \geq 0$. Therefore, we get $f(x) \geq \mathcal{L}(x; \pi_{1:})$. Next, the derivations of Part c) and d) are similar to those of Lemma 4.1. $\square$ Part d) of the preceding lemma motivates us to consider a multi-layer Q-gap

function as an alternative optimality criterion. Specifically, for a point $z^t := (x^t; \pi^t_{1:})$, the Q-gap function, defined with respect to a reference point $z \in Z$, is given by

$$Q(z^t, z) := \mathcal{L}(x^t; \pi_{1:}) - \mathcal{L}(x; \pi^t_{1:}). \tag{3.74}$$

The Q function admits the following decomposition

$$Q(z^t, z) = Q_0(z^t, z) + \textstyle\sum_{i=1}^k Q_i(z^t, z), \tag{3.75}$$

where

$$\begin{aligned} Q_0(z^t, z) &:= \mathcal{L}(x^t; \pi^t_{1:}) - \mathcal{L}(x; \pi^t_{1:}) \\ &= \boxed{\pi^t_{1:} x^t + u(x^t)} - \pi^t_{1:} x - u(x), \end{aligned}$$

$$\begin{aligned} Q_i(z^t, z) &:= \mathcal{L}(x^t; \pi_{1:i-1}, \pi_i, \pi^t_{i+1:}) - \mathcal{L}(x^t; \pi_{1:i-1}, \pi^t_i, \pi^t_{i+1:}) \\ &= \pi_{1:i-1} \left(\pi_i \mathcal{L}_{i+1}(x^t; \pi^t_{i+1:}) - f_i^*(\pi_i) \boxed{-[\pi^t_i \mathcal{L}_{i+1}(x^t; \pi^t_{i+1:}) - f_i^*(\pi^t_i)]} \right). \end{aligned}$$

To reduce the Q-gap function, we can sequentially update π_i to reduce Q_i, and then update x to reduce Q_0 in an iterative manner. This idea leads us to a conceptual SSD method. Specifically, initialized to $x^0 \in X$ and $\pi^0_i = \nabla f_i(\underline{y}^0_i)\ \forall i \in [k]$, the t^{th} iteration of the iterative method is given by

$$\begin{aligned} &\text{for } i = k, k-1, \ldots, 1 \\ &\quad \pi^t_i \leftarrow \underset{\pi_i \in \Pi_i}{\arg\max}\, \pi_i \tilde{y}^t_i(\xi) - f_i^*(\pi_i) - \tau_{i,t} U_i(\pi_i; \pi^{t-1}_i), \\ &\qquad\qquad \text{where } \tilde{y}^t_i := \mathcal{L}_{i+1}(x^{t-1}; \pi^t_{i+1:}) + \theta_t \pi^t_{i+1:}(x^{t-1} - x^{t-2}). \\ &x^t \leftarrow \arg\min_{x \in X} \tilde{y}^t_0(\xi) x + \tfrac{\eta_t}{2} \left\| x - x^{t-1} \right\|^2, \text{ where } \tilde{y}^t_0 := \pi^t_{1:}. \end{aligned} \tag{3.76}$$

Here U_i is some Bregman distance function, $\tau_{i,t}$ and η_t are stepsize parameters, and $\tilde{y}^t_i(\xi)$'s and $\tilde{y}^t_0(\xi)$ are some unbiased estimators to $\tilde{y}^t_i$'s and $\tilde{y}^t_0$, respectively.

We provide a concrete implementation of (Equation 3.76), which would lead to the SSD Algorithm 6. First, we select the prox function U_i in (Equation 3.76) to be the conjugate Bregman distance function, $D_{f_i^*}$. Similar to Algorithm 4, such a choice simplifies the π_i^t prox mapping in (Equation 3.76) to gradient evaluation at some averaged $\underline{y}_i^t$, i.e., $\pi_i^t = \nabla f_i(\underline{y}_i^t)$, in Line 5 and 6 of Algorithm 6. Now we construct the unbiased estimators $\{\tilde{y}_i^t(\xi)\}$ in (Equation 3.76). The following conventions are used:

$$\mathcal{L}_{k+1}(x;\pi_{k+1:}) = x,\ \nabla f_{k+1:}(\underline{y}_{k+1:}^t,\xi) = I,\ \text{and}\ \nabla f_{i:}(\underline{y}_{i:}^t,\xi_{i:}^j) := \prod_{l=i}^{k} \nabla f_l(\underline{y}_l^t,\xi_i^j).$$

At the i^{th} layer, the unbiased estimators for $\mathcal{L}_i(x^{t-1};\pi_{i:}^t)$ and $\prod_{l=i}^k \nabla f_l(\underline{y}_l^t)$ are constructed using i independent estimators in Line 6. The necessity of the repeated independent calls to $\mathcal{SO}_i$ is explained in Figure Figure 3.3. After all the π_i^t prox-mappings, $\nabla f_{1:}(\underline{y}_{1:}^t,\xi_{1:}^0)$ is used as the stochastic argument, $\tilde{y}_0^t(\xi)$, for the x^t prox mapping in Line 8 of Algorithm 6.

Algorithm 6 Stochastic Sequential Dual (SSD) Method for the Smooth Multi-layer Problem

Input: $x^0 \in X$.

1: Set $\underline{y}_k^0 := x^0$ and $x^{-1} := x^0$.
2: Call $\mathcal{SO}_i$ to obtain $\nabla f_i(\underline{y}_i^0,\hat{\xi}_i)$ and $\underline{y}_{i-1}^0 = f_i(\underline{y}_i^0,\hat{\xi}_i)$ for $i = k, k-1,\ldots,1$.
3: **for** $t = 1,2,3...N$ **do**
4: **for** $i = k, k-1,\ldots,1$ **do**
5: Set $\tilde{y}_i^t(\xi) \leftarrow \mathcal{L}_{i+1}[x^{t-1};\pi_{i+1:}^t(\xi_{i+1:}^i)] + \theta_t \nabla f_{i+1:}(\underline{y}_{i+1:}^{t-1},\hat{\xi}_{i+1:})(x^{t-1}-x^{t-2})$.
 Set $\underline{y}_i^t \leftarrow (\tau_{i,t}\underline{y}_i^{t-1} + \tilde{y}_i^t(\xi))/(1+\tau_{i,t})$.
6: Call $\mathcal{SO}_i$ to obtain independent estimates $\{f_i(\underline{y}_i^t,\xi_i^j), \nabla f_i(\underline{y}_i^t,\xi_i^j)\}_{j=0}^{i-1}$ and $\nabla f_i(\underline{y}_i^t,\hat{\xi}_i)$.
 Set $\mathcal{L}_i[x^{t-1};\pi_{i:}^t(\xi_{i:}^j)] \leftarrow f_i(\underline{y}_i^t,\xi_i^j) + \nabla f_i(\underline{y}_i^t,\xi_i^j)\{\mathcal{L}_{i+1}[x^{t-1};\pi_{i+1:}^t(\xi_{i+1:}^j)] - \underline{y}_i^t\}, \forall j = 1,\ldots,i-1$.
 Set $\nabla f_{i:}(\underline{y}_{i:}^t,\hat{\xi}_{i:}) \leftarrow \nabla f_i(\underline{y}_i^t;\hat{\xi}_i)\nabla f_{i+1:}(\underline{y}_{i+1:}^t,\hat{\xi}_{i+1:}),\ \forall j = 1\ldots i-1$.
 Set $\nabla f_{i:}(\underline{y}_{i:}^t,\xi_{i:}^0) \leftarrow \nabla f_i(\underline{y}_i^t;\xi_i^0)\nabla f_{i+1:}(\underline{y}_{i+1:}^t,\xi_{i+1:}^0)$.
7: **end for**
8: Set $x^t \leftarrow \arg\min_{x\in X} \tilde{y}_0^t(\xi)x + u(x) + \frac{\eta_t}{2}\|x-x^{t-1}\|^2$ where $\tilde{y}_0^t(\xi) := \nabla f_{1:}(\underline{y}_{1:}^t,\xi_{1:}^0)$.
9: **end for**
10: Return $\bar{x}^N := \sum_{t=1}^N \omega_t x^{t+1} / \sum_{t=1}^N \omega_t$.

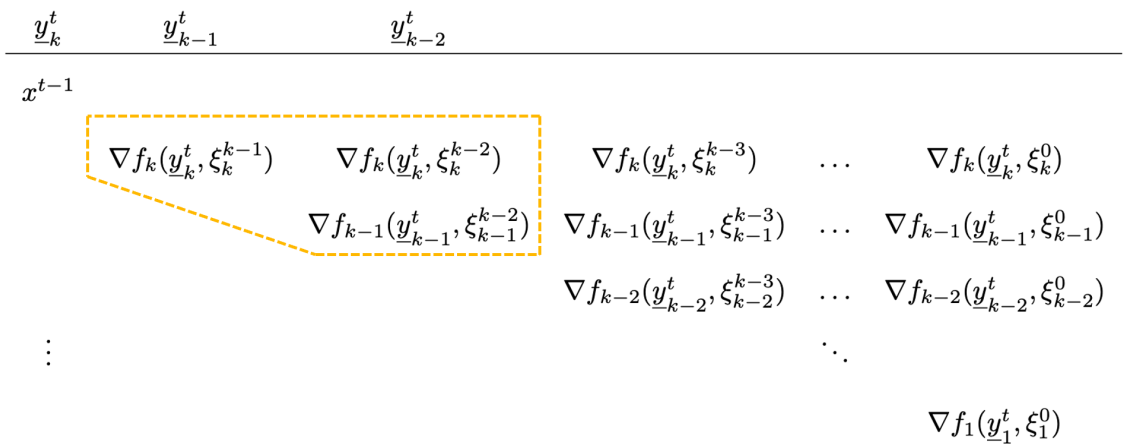


Figure 3.3: Necessity of Resampling in the SSD method: in order to obtain an unbiased estimator for $\mathcal{L}_i(x^{t-1}; \pi_{i:}^t) := \mathcal{L}_i(x^{t-1}; \nabla f_i(\underline{y}_i^t), \dots, \nabla f_k(\underline{y}_k^t))$, we need conditionally independent estimators to $\nabla f_i(\underline{y}_i^t), \nabla f_{i+1}(\underline{y}_{i+1}^t), \dots, \nabla f_k(\underline{y}_k^t)$. However, when generating the evaluation point $\underline{y}_l^t$ for any $l > i$, Algorithm 6 utilizes estimators $\{\nabla f_j(\underline{y}_j^t, \xi_j^l)\}_{j>l}$, causing $\underline{y}_l^t$ to be conditionally correlated with $\{\nabla f_j(\underline{y}_j^t, \xi_j^l)\}_{j>l}$. In the above figure, this means that the stochastic estimators in the l^{th} column are correlated with $\underline{y}_l^t$ in the header row. Consequently, the entire triangle of old estimators $\{\nabla f_l(\underline{y}_l^t, \xi_l^j)\}_{l>i, j>i}$ (highlighted above) are conditionally correlated with the required evaluation points $\{\underline{y}_l^t\}_{l>i}$. Therefore, $(k-i+1)$ new independent estimators need to be redrawn from $\mathcal{SO}_i$'s, i.e., $\nabla f_i(\underline{y}_i^t, \xi_i^{i-1}), \nabla f_{i+1}(\underline{y}_{i+1}^t, \xi_{i+1}^{i-1}), \dots,$ and $\nabla f_k(\underline{y}_k^t, \xi_k^{i-2})$, for constructing an unbiased $\mathcal{L}_i(x^{t-1}; \pi_{i:}^t(\xi_{i:}^{i-1}))$.

We now provide the convergence result for Algorithm 6. First, we need to define a few problem parameters. We require each layer function f_i to be L_i-smooth and M_i-continuous for some $M_i \geq 1$[③], i.e.,

$$\|\nabla f_i(y_i)\| \leq M_i \text{ and } \|\nabla f_i(y_i) - \nabla f_i(\bar{y}_i)\| \leq L_i \|y_i - \bar{y}_i\| \ \ \forall y_i, \bar{y}_i \in \mathbb{R}^{n_i}. \tag{3.77}$$

For notation compactness, we will use the shorthand $M_{i:j} := \prod_{l=i}^{j} M_l$ throughout this section. The aggregate variance constants for Algorithm 6 are defined as

$$\tilde{\sigma}_i = \{\max_{t\geq 0} \mathbb{E}[\left\|\tilde{y}_i^t(\xi) - \mathbb{E}[\tilde{y}_i^t(\xi)]\right\|^2]\}^{1/2}, \ \tilde{\sigma}_x = \{\max_{t\geq 0} \mathbb{E}[\left\|\tilde{y}_0^t(\xi) - \mathbb{E}[\tilde{y}_0^t(\xi)]\right\|^2]\}^{1/2}. \tag{3.78}$$

Similar to Sections section 3.2 and section 3.3, the above uniform aggregate variance con-

[③]We assume $M_i \geq 1$ to avoid trivialities.

stants are finite because they can be decomposed into variance bounds associated with individual layer functions, i.e., $\mathcal{SO}_i$'s. However, for the sake of simplicity, we do not present the decomposition in this section. In the multi-layer nested setting, the decomposition can be quite complicated, but its knowledge is unimportant in practice since the SSD method can be implemented with parameter-free stepsizes to achieve the order-optimal oracle complexity.

We are now prepared to state the convergence result for Algorithm 6. The following theorem suggests appropriate stepsize choices and provides the convergence rates under both non-strongly convex and strongly convex settings. The proof of this theorem is deferred to subsection 3.4.3.

Theorem 3.7 *Consider a smooth multi-layer function f (c.f.* (Equation 3.72)*) with its problem parameters defined in* (Equation 3.77) *and* (Equation 3.78)*. Suppose iterates $\{x^t\}$ are generated by Algorithm 6 with*

$$\omega_t = t,\ \theta_t = (t-1)/t,\ \tau_{i,t} = (t-1)/2, \forall i \in [k]. \tag{3.79}$$

Let $\tilde{L} := \sum_{i=1}^{k} M_{1:i-1} L_i M_{i+1:}^2$ denote the aggregate smoothness constant of f. The stepsize η_t can be selected based on the strong convexity modulus α of $u(x)$, resulting in the following convergence guarantees:

a) If $u(x)$ is non-strongly convex, selecting $\eta_t = \max\{2\tilde{L}/(t+1), \tilde{\sigma}_x\sqrt{t}/\mathcal{D}_X\}$ leads to

$$\begin{aligned}\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq & \tfrac{\tilde{L}}{N(N+1)} \left\|x^0 - x^*\right\|^2 + \tfrac{4}{N}\textstyle\sum_{i=1}^{k-1} M_{1:i-1} L_i \tilde{\sigma}_i^2 \\ & + \tfrac{4}{\sqrt{N}}(\tilde{\sigma}_x \mathcal{D}_X + \textstyle\sum_{i=1}^{k-1} M_{1:i}\tilde{\sigma}_i).\end{aligned} \tag{3.80}$$

b) If $u(x)$ is strongly convex with $\alpha > 0$, selecting $\eta_t = \max\{2\tilde{L}/(t+1), \alpha(t-1)/2\}$

leads to

$$\begin{aligned}\mathbb{E}[f(\bar{x}^N) - f(x^*)] \leq [\tfrac{\sum_{i=1}^{k-1} M_{1:i-1} L_i M_{i+1:}^2}{\alpha} + 1][\tfrac{\log(N+1)\tilde{L}}{N^2} \left\|x^0 - x^*\right\|^2 \\ + \tfrac{4}{N}(\textstyle\sum_{i=1}^{k-1} M_{1:i-1} L_i \tilde{\sigma}_i^2 + \tfrac{\tilde{\sigma}_x^2}{\alpha})], \\ \mathbb{E}[\left\|x^N - x^*\right\|^2] \leq \tfrac{\tilde{L}}{\alpha N(N+1)} \left\|x^0 - x^*\right\|^2 + \tfrac{4}{\alpha N}(\textstyle\sum_{i=1}^{k-1} M_{1:i-1} L_i \tilde{\sigma}_i^2 + \tfrac{\tilde{\sigma}_x^2}{\alpha}).\end{aligned} \tag{3.81}$$

A few observations are worth noting concerning the result. In the deterministic scenario where $\tilde{\sigma}_i = 0 \; \forall i$, Algorithm 6 attains the optimal oracle complexity under the non-strongly convex setting, while a restarted version of it (see Section 4.2.3 of [28]) achieves optimal oracle complexity under the strongly convex setting. In the stochastic scenario, Algorithm 6 achieves order-optimal stochastic oracle complexity of $\mathcal{O}(1/\epsilon^2)$ ($\mathcal{O}(1/\epsilon)$) with the parameter-free step sizes in (Equation 3.79) and any $\eta_t = \Theta(\sqrt{t})$ ($\eta_t = (t-1)\alpha/2$) under the non-strongly (strongly) convex setting.

3.4.2 The nSSD Method for the General Nested Composition Problem

In this subsection, we introduce a multi-layer nSSD method to address the general nested composite problem. For the k-layer problem described in (Equation 3.72), we consider an arbitrary composition of layer functions, meaning that the layer indices $[k]$ can be partitioned into subsets $\mathfrak{S}$, $\mathfrak{P}$, and $\mathfrak{N}$, and f_i is smooth, structured non-smooth, or general non-smooth if $i \in \mathfrak{S}$, $i \in \mathfrak{P}$, or $i \in \mathfrak{N}$, respectively. Our development follows the same structure as the previous subsection. We propose a nested linearization reformulation, develop a Q-gap function, design the nSSD method, and present the convergence result.

First, we introduce the multi-layer nested linearization reformulation. We need to generalize (Equation 3.73) because the general problem could involve non-smooth outer-layer functions. Let $\mathfrak{N}_o$ denote the set of such layer indices, i.e., $\mathfrak{N}_o := \mathfrak{N} \cap [k-1]$. The triconjugate reformulations (Equation 3.49) for these layer functions require additional auxiliary primal variables. For compactness, we use the shorthand notation $v_{i:} := \{v_l\}_{l \in \mathfrak{N}_o \cap i:k}$ to denote these variables, and use $V_{i:} := \prod_{l \in \mathfrak{N}_o \cap i:} V_l$ to denote domains (see (Equation 3.49)).

The multi-layer nested linearization reformulation is defined as:

$$\min_{x\in X}\min_{v_{1:}\in V_{1:}}\max_{\pi_{1:}\in\Pi_{1:}}\{\mathcal{L}(x;\pi_{1:}) := \mathcal{L}_1(x;\pi_{1:}) + u(x)\},$$
$$\text{where } \mathcal{L}_i(x;\pi_{i:}) := \begin{cases} x & \text{if } i = k+1, \\ \pi_i[\mathcal{L}_{i+1}(x;\pi_{i+1:}) - v_i] + f_i(v_i) & \text{if } i \in \mathfrak{N}_o, \\ \pi_i\mathcal{L}_{i+1}(x;\pi_{i+1:}) - f_i^*(\pi_i) & \text{otherwise.} \end{cases} \tag{3.82}$$

Here, when $i \notin \mathfrak{N}_o$, we assume $\Pi_i := \text{dom}(f_i^*)$. When $i \in \mathfrak{N}_o$, we assume (V_i, Π_i) satisfy $f_{i:}(x^*) \in V_i$, $\text{dom}(f_i^*) \subset \Pi_i$ and the monotonicity requirement in Assumption 3, i.e., $\{\pi_{1:l-1} : \pi_j \in \Pi_j, j \leq l-1\} \subset \mathbb{R}_+^{n_{l-1}}$ if f_l is not affine.

Let $z := (x, v_{1:}; \pi_{1:})$ and $Z := X \times V_{1:} \times \Pi_{1:}$. The next lemma relates the above reformulation (Equation 3.82) to the original problem in (Equation 3.72).

Lemma 3.5 *The following relations between $\mathcal{L}$ in* (Equation 3.82) *and f in* (Equation 3.72) *are valid.*

a) If $v_i^* = f_{i:}(x^*)\ \forall i \in \mathfrak{N}_o$, then $\mathcal{L}(x^*, v_{1:}^*; \pi_{1:}) \leq f(x^*)\ \forall \pi_{1:} \in \Pi_{1:}$.

b) Given a pair $(x, v_{1:}) \in X \times V_{1:}$, $\mathcal{L}(x, v_1; \hat{\pi}_{1:}) \geq f(x)$ if $\hat{\pi}_i \in \partial f_i(f_{i+1:}(x))\ \forall i$.

c) For any $z := (x, v_{1:}; \pi_{1:}) \in Z$, $f(x) - f(x^*) \leq \mathcal{L}(x, v_{1:}; \hat{\pi}_{1:}) - \mathcal{L}(x^*, v_{1:}^*; \pi_{1:})$ if $\hat{\pi}_i \in \partial f_i(f_{i+1:}(x))\ \forall i$.

Proof: The proof is a straightforward extension to that of Lemma 3.3. We need to show recursively that $\mathcal{L}_i(x^*, v_{i:}^*; \pi_{i:}) \leq f_i(f_{i+1:}(x^*))\ \forall i$ for Part a), and that $f_{i:}(x) \leq \mathcal{L}_i(x, v_{i:}; \hat{\pi}_{i:})\ \forall i$ for Part b). □

Part c) of the preceding lemma motivates us to use the following Q-gap function, defined with respect to some reference point $\bar{z} \in Z$, as an alternative optimality criterion for algorithm design

$$Q(z; \bar{z}) := \mathcal{L}(x, v_{1:}; \bar{\pi}_{1:}) - \mathcal{L}(\bar{x}, \bar{v}_{1:}; \pi_{1:}).$$

The Q-gap function admits the following useful decomposition

$$Q(z;\bar{z}) = Q_0(z;\bar{z}) + \textstyle\sum_{i=1}^{k} Q_i(z;\bar{z}) + \sum_{i\in\mathfrak{N}_o} Q_i^v(z;\bar{z}), \tag{3.83}$$

where

$$\begin{aligned}
Q_0(z^t,z) &:= \mathcal{L}(x^t, v_{1:}; \pi_{1:}^t) - \mathcal{L}(x, v_{1:}; \pi_{1:}^t) = \boxed{\pi_{1:}^t x^t + u(x^t)} - \pi_{1:}^t x - u(x),\\
Q_i^v(z^t,z) &:= \mathcal{L}(x^t, v_{1:i-1}^t, v_i^t, v_{i+1:}; \pi_{1:}^t) - \mathcal{L}(x, v_{1:i-1}^t, v_i, v_{i+1:}; \pi_{1:}^t)\\
&:= \boxed{\pi_{1:i-1}^t[f_i(v_i^t) - \pi_i^t v_i^t]} - \pi_{1:i-1}^t[f_i(v_i) - \pi_i^t v_i],\\
Q_i(z^t,z) &:= \mathcal{L}(x^t, v_{1:}^t; \pi_{1:i-1}, \pi_i, \pi_{i+1:}^t) - \mathcal{L}(x^t, v_{1:}^t; \pi_{1:i-1}, \pi_i^t, \pi_{i+1:}^t)\\
&= \begin{cases}
\pi_{1:i-1}\{\pi_i[\mathcal{L}_{i+1}(x^t, v_{i+1:}^t; \pi_{i+1:}^t) - v_i^t] \boxed{-\pi_i^t[\mathcal{L}_{i+1}(x^t, v_{i+1:}^t; \pi_{i+1:}^t) - v_i^t]}\} \\
\qquad \text{if } i \in \mathfrak{N}_o,\\
\pi_{1:i-1}\{\pi_i \mathcal{L}_{i+1}(x^t, v_{i+1:}^t; \pi_{i+1:}^t) - f_i^*(\pi_i) \boxed{-[\pi_i^t \mathcal{L}_{i+1}(x^t, v_{i+1:}^t; \pi_{i+1:}^t) - f_i^*(\pi_i^t)]}\} \\
\qquad \text{otherwise.}
\end{cases}
\end{aligned}$$

To reduce the Q-gap function, we can sequentially update π_i's to reduce their respective Q_i's, and then sequentially update primal variables v_i's and x to reduce Q_i^v's and Q_0. This idea leads us to a conceptual iterative nSSD method. Specifically, initialized to some $z^0 := (x^0, v_{1:}^0; \pi_{1:}^0) \in Z$, the t^{th} iteration of this iterative method is given by:

For $i = k, k-1, \ldots, 1$:

If $i \in \mathfrak{S} \cup \mathfrak{P}$,

$$\pi_i^t \leftarrow \arg\max_{\pi_i \in \Pi_i} \pi_i \tilde{y}_i^t(\xi) - f_i^*(\pi_i) - \tau_{i,t} U_i(\pi_i; \pi_i^{t-1}),$$

$$\text{where } \tilde{y}_i^t := \mathcal{L}_{i+1}(x^{t-1}; \pi_{i+1:}^t) + \theta_t \pi_{i+1:}^t (x^{t-1} - x^{t-2}).$$

Else if $i = k$ and f_k is general non-smooth,

$$\pi_k^t \leftarrow \arg\max_{\pi_k \in \Pi_k} \pi_k x^{t-1} - f_k^*(\pi_k).$$

Otherwise, i.e.,$i \in \mathfrak{N}_o$,

$$\pi_i^t \leftarrow \arg\max_{\pi_i \in \tilde{\Pi}_i} \pi_i [\tilde{y}_i^t(\xi) - v_1^{t-1}] - \tau_{i,t} \left\| \pi_i - \pi_i^{t-1} \right\|^2 / 2,$$

$$\text{where } \tilde{y}_i^t := \mathcal{L}_{i+1}(x^{t-1}; \pi_{i+1:}^t);$$

For $i \in \mathfrak{N}_o$,

$$v_i^t \leftarrow \arg\min_{v_i \in V_i} \tilde{w}_i^t(\xi) v_i + \gamma_{1,t} \left\| v_i - v_i^{t-1} \right\|^2 / 2, \text{ where } \tilde{w}_i^t = \pi_{1:i-1}^t (f_i'(v_{t-1}^t) - \pi_i^t)$$

$$x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t(\xi) x + \tfrac{\eta_t}{2} \left\| x - x^{t-1} \right\|^2, \text{ where } \tilde{y}_0^t := \pi_{1:}^t. \tag{3.84}$$

We describe the implementation detail for (Equation 3.84) under the nested stochastic setting, which leads to the concrete multi-layer nSSD method shown in Algorithm 7. The prox-mappings in Line 5 to Line 18 are straightforward implementations of the $\{\pi_i^t\}$, $\{v_i^t\}$, and x^t updates shown in (Equation 3.84). For smooth layer functions, we utilize the Bregman distance generated by the conjugate function, $D_{f_i^*}$, as the prox-function U_i, and the π_i^t computation is expressed equivalently as some gradient evaluation in Algorithm 7 (refer to Lemma 3.2). Now we only need to specify the construction of the unbiased stochastic estimators for the arguments in these prox-mappings. To ensure unbiasedness, independently sampled estimators are necessary, as explained in detail in Figure Figure 3.3. The nested composite estimators are constructed incrementally, where at most $i+1$ queries to $\mathcal{SO}_i$ are

required in each iteration. This construction is given by:

$$
\begin{aligned}
&\mathcal{L}_i(x^{t-1}, v_{i:}^{t-1}(\xi_{i:}^j); \pi_{i:}^t(\xi_{i:}^j)) \\
&:= \begin{cases}
x^{t-1} & \text{if } i = k+1 \\
\nabla f_i(\underline{y}_i^t, \xi_i^j)\{\mathcal{L}_{i+1}[x^{t-1}, v_{i+1:}^{t-1}(\xi_{i+1:}^j); \pi_{i+1:}^t(\xi_{i+1:}^j)] - \underline{y}_i^t\} + f_i(\underline{y}_i^t, \xi_i^j) & \text{else if } i \in \mathfrak{S} \\
\pi_i^t \mathcal{L}_{i+1}[x^{t-1}, v_{i+1:}^{t-1}(\xi_{i+1:}^j); \pi_{i+1:}^t(\xi_{i+1:}^j)] - f_i^*(\pi_i^t) & \text{else if } i \in \mathfrak{P} \\
\pi_i^t\{\mathcal{L}_{i+1}[x^{t-1}, v_{i+1:}^{t-1}(\xi_{i+1:}^j); \pi_{i+1:}^t(\xi_{i+1:}^j)] - v_i^{t-1}\} + f_i(v_i^{t-1}, \xi_i^j) & \text{else if } i \in \mathfrak{N}_o \\
f_k(x^{t-1}, \xi_k^j) & \text{otherwise}
\end{cases} \\
&\pi_{i:}^t(\xi_{i:}^j) := \begin{cases}
I & \text{if } i \geq k+1 \\
\nabla f_i(\underline{y}_i^t, \xi_i^j)\pi_{i+1:}^t(\xi_{i+1:}^j) & \text{else if } i \in \mathfrak{S} \\
\pi_i^t \pi_{i+1:}^t(\xi_{i+1:}^j) & \text{else if } i \in \mathfrak{N}_o \cup \mathfrak{P} \\
f_k'(x^{t-1}, \xi_k^j) & \text{otherwise, i.e., } i = k \text{ and } k \in \mathfrak{N}.
\end{cases}
\end{aligned}
\tag{3.85}
$$

Here $\{\hat{\xi}_i\} \cup \{\xi_i^j\}_{0 \leq j < i}$ represent independent samples drawn from $\mathcal{SO}_i$. Since it simplifies to Algorithm 5 if $k = 2$ and $\{1, 2\} \subset \mathfrak{N}$, and to Algorithm 6 if $[k] \subset \mathfrak{S}$, Algorithm 7 can be regarded as a generalization to preceding algorithms.

We present the convergence result for the proposed nSSD method. First, we need to define a few problem parameters that characterize the Lipschitz-continuity and Lipschitz-

Algorithm 7 Stochastic Sequential Dual-primal (nSSD) Method for the General Multi-layer Problem

Input: $x^0 \in X$, $\pi_i^0 \in \Pi_i \; \forall i \in \mathfrak{P}$, $(v_i^0, \pi_i^0) \in V_i \times \Pi_i \; \forall i \in \mathfrak{N}_o$.

Set $\underline{y}_k^0 := x^0$ and $\underline{y}_i^0 = f_{i+1}(\underline{y}_{i+1}^0, \xi_{i+1}) \forall i \leq k-1$. Set $\pi_i^0 = \nabla f_i(\underline{y}_i^0) \; \forall i \in \mathfrak{S}$ and $\pi_k^0 = f_k'(x^0)$ if $k \in \mathfrak{N}$.
for $t = 1, 2, 3...N$ **do**
  for $i = k, k-1, \ldots, 1$ **do**
    Set $y_i^t(\xi) := \mathcal{L}_{i+1}(x^{t-1}, v_{i+1:}^{t-1}(\xi_{i+1:}^i); \pi_{i+1:}^t(\xi_{i+1:}^i))$ and $\tilde{y}_i^t(\xi) := \mathcal{L}_{i+1}(x^{t-1}, v_{i+1:}^{t-1}(\xi_{i+1:}^i); \pi_{i+1:}^t(\xi_{i+1:}^i)) + \pi_{i+1:}^{t-1}(\hat{\xi}_{i+1:})(x^{t-1} - x^{t-2})$ (c.f. (Equation 3.85)).
    if f_i is smooth **then**
      set $\pi_i^t \leftarrow \nabla f_i(\underline{y}_i^t)$ where $\underline{y}_i^t := (\tau_{i,t}\underline{y}_i^{t-1} + \tilde{y}_i^t(\xi))/(1+\tau_{i,t})$.
    else if f_i is structured non-smooth **then**
      set $\pi_{i,j}^t \leftarrow \arg\max_{\pi_{i,j} \in \Pi_{i,j}} \pi_{i,j}[\tilde{y}_i^t(\xi)] - \tau_{i,t} \left\|\pi_{i,j} - \pi_{i,j}^{t-1}\right\|^2 /2 \; \forall j \in [n_{i-1}]$.
    else if $i = k$ and f_i is general non-smooth **then**
      set $\pi_k^t \leftarrow f_k'(x^{t-1})$.
    else $i \in \mathfrak{N}_o$,i.e., f_i is general non-smooth
      Set $\pi_{i,j}^t \leftarrow \arg\max_{\pi_{i,j} \in \Pi_{i,j}} \pi_{i,j}[y_i^t(\xi) - v_i^{t-1}] - \tau_{i,t} \left\|\pi_{i,j} - \pi_{i,j}^{t-1}\right\|^2 /2 \; \forall j \in [n_{i-1}]$.
    end if
  end for
  for $i \in \mathfrak{N}_o$ **do**
    Set $v_i^t \leftarrow \arg\min_{v_i \in V_i} \tilde{w}_i^t(\xi) v_i + \gamma_{i,t} \left\|v_i - v_i^{t-1}\right\|^2 /2$ where $\tilde{w}_i^t(\xi) = \pi_{1:i-1}^t(\xi_{1:i-1}^0)[f_i'(v_i^{t-1}, \hat{\xi}_i) - \pi_i^t]$.
  end for
  Set $x^t \leftarrow \arg\min_{x \in X} \tilde{y}_0^t(\xi) x + u(x) + \frac{\eta_t}{2} \left\|x - x^t\right\|^2$ where $\tilde{y}_0^t(\xi) := \pi_{1:}^t(\xi_{1:}^0)$.
end for
Return $\bar{x}^N := \sum_{t=1}^N \omega_t x^{t+1} / \sum_{t=1}^N \omega_t$.

smoothness of the layer functions:

$$\|\nabla f_i(y_i) - \nabla f_i(\bar{y}_i)\| \leq L_i \|y_i - \bar{y}_i\| \text{ and } \|\nabla f_i(y_i)\| \leq M_i \; \forall y_i, \bar{y}_i \in \mathbb{R}^{n_i} \; \forall i \in \mathfrak{S}.$$

$$\|\pi_i\| \leq M_i \; \forall \pi_i \in \Pi_i \text{ and } \|v_i - \bar{v}_i\| \leq M_{i+1:} \mathcal{D}_X \; \forall v_i, \bar{v}_i \in V_i, \forall i \in \mathfrak{N}_o.$$

$$\|\pi_i\| \leq M_i \; \forall \pi_i \in \Pi_i \; \forall i \in \mathfrak{P}, \text{ and } \|f_k'(x)\| \leq M_k \; \forall x \in X \text{ if } k \in \mathfrak{N}.$$

(3.86)

We also need the aggregate variance bounds for the unbiased estimators in Algorithm 7:

$$\begin{aligned}
&\tilde{\sigma}_i = \max\{\{\max_{t\geq 0} \mathbb{E}[\left\|\tilde{y}_i^t(\xi) - \mathbb{E}[\tilde{y}_i^t(\xi)]\right\|^2]\}^{1/2}, \{\max_{t\geq 0} \mathbb{E}[\left\|y_i^t(\xi) - \mathbb{E}[y_i^t(\xi)]\right\|^2]\}^{1/2}\},\\
&\tilde{\sigma}_x = \{\max_{t\geq 0} \mathbb{E}[\left\|\tilde{y}_0^t(\xi) - \mathbb{E}[\tilde{y}_0^t(\xi)]\right\|^2]]\}^{1/2},\\
&\tilde{\sigma}_{v,i} = \{\max_{t\geq 0} \mathbb{E}[\left\|\pi_{1:i-1}^t(\xi_{1:i-1}^0)[f_i'(v_i^{t-1}, \hat{\xi}_i)] - \mathbb{E}[\pi_{1:i-1}^t(\xi_{1:i-1}^0)[f_i'(v_i^{t-1}, \hat{\xi}_i)]\right\|^2]\}^{1/2}.
\end{aligned} \tag{3.87}$$

Now we are ready to state the convergence result.

Theorem 3.8 *Consider a general multi-layer function f* (Equation 3.72)*, where $\mathfrak{S}$, $\mathfrak{P}$ and $\mathfrak{N}$ denoting the indices of smooth, structured nonsmooth and general nonsmooth layer functions, respectively. Suppose the problem parameters are defined in* (Equation 3.86) *and* (Equation 3.87)*. Let $\mathfrak{N}_{1:i}$ denote the layer indices of general non-smooth functions outer to f_{i+1}, i.e., $\mathfrak{N}_{1:i} := \{1, 2, ...i\} \cap \mathfrak{N}$. Consider $\{x^t\}$ generated by Algorithm 7 with stepsizes chosen according to:*

$$\begin{aligned}
&\omega_t := t,\ \theta_t := (t-1)/t,\\
&\tau_{i,t} := (t-1)/2,\ \eta_i^t := 2M_{1:i-1}L_iM_{i+1:}^2/(t+1), \forall i \in \mathfrak{S}\\
&\tau_{i,t} := \max\{\tilde{\sigma}_i\sqrt{t},\ M_{i+1:}\mathcal{D}_X\}/M_i,\ \eta_i^t = M_{1:}/\mathcal{D}_X,\ \forall i \in \mathfrak{P}\\
&\tau_{i,t} := \tilde{\sigma}_i\sqrt{t}/M_i,\ \gamma_{i,t} := \max\{(|\mathfrak{N}_{1:i}|+1)M_{1:i}, \tilde{\sigma}_i\}\sqrt{t}/(M_{i+1:}D_X),\ \forall i \in \mathfrak{N}_{1:k-1}\\
&\eta_i^t := M_{1:}\sqrt{t}/\mathcal{D}_X\ \forall i \in \mathfrak{N}.
\end{aligned} \tag{3.88}$$

Under these conditions, we obtain the following convergence guarantees.

a) If $u(x)$ is non-strongly convex, selecting $\eta_t := \max\{\sum_{i=1}^k \eta_i^t, \tilde{\sigma}_x\sqrt{t}/\mathcal{D}_X\}$ leads to

$$\begin{aligned}
&\mathbb{E}[f(\bar{x}^N) - f(x^*)]\\
&\leq \mathcal{O}\{\textstyle\sum_{i\in\mathfrak{S}}[\frac{M_{1:i-1}L_iM_{i+1:}^2\mathcal{D}_X^2}{N^2} + \frac{M_{1:i-1}L_i\tilde{\sigma}_i^2}{N}] + \sum_{i\in\mathfrak{P}}[\frac{M_{1:}\mathcal{D}_X}{N}]\\
&+ \frac{\sum_{i\in\mathfrak{N}_{1:k-1}}[|\mathfrak{N}_{1:i}|M_{1:}D_X + \tilde{\sigma}_{v,i}M_{i+1:}\mathcal{D}_X]}{\sqrt{N}} + \textstyle\sum_{i\in\mathfrak{N}}[\frac{M_{1:}\mathcal{D}_X}{\sqrt{N}}]\\
&+ \textstyle\sum_{i=1}^k[\frac{M_{1:i}\tilde{\sigma}_i}{\sqrt{N}}] + \frac{D_X\tilde{\sigma}_x}{\sqrt{N}}\}.
\end{aligned} \tag{3.89}$$

b) If $u(x)$ is strongly convex, i.e., $\alpha > 0$ and all outer layer functions are smooth, i.e., $\{1, 2, \ldots, k-1\} \subset \mathfrak{S}$. In this case, selecting $\eta_t := \max\{\sum_{i=1}^{k} \eta_i^t, \alpha(t-1)/2\}$ leads to

$$\begin{aligned}
&\mathbb{E}[f(\bar{x}^N) - f(x^*)] \\
&\leq \mathcal{O}\left([\tfrac{\sum_{i=1}^{k-1} M_{1:i-1} L_i M_{i+1:}^2}{\alpha} + 1]\{\tfrac{\log(N+1)}{N^2}[\textstyle\sum_{i\in\mathfrak{S}} M_{1:i-1} L_i M_{i+1:}^2 D_X^2] \right. \\
&\left. + \tfrac{1}{N}[\textstyle\sum_{i=1}^{k-1} M_{1:i-1} L_i \tilde{\sigma}_i^2 + \sum_{i\in\mathfrak{P}}(M_{1:}\mathcal{D}_X) + \sum_{i\in\mathfrak{N}} \tfrac{M_{1:}\mathcal{D}_X}{\alpha} + \tfrac{\tilde{\sigma}_x^2}{\alpha}]\}\right).
\end{aligned} \tag{3.90}$$

We make three remarks regarding the convergence result. First, under the deterministic setting, the terms attributable to the smooth and the structured non-smooth functions in (item 3.89), i.e., $M_{1:i-1} L_i M_{i+1:}^2 \mathcal{D}_X^2 / N^2$ and $M_{1:}\mathcal{D}_X/N$, respectively, are unimprovable. Second, under the stochastic setting, the assumption of all outer layer functions being smooth is necessary for obtaining the improved stochastic oracle complexity of $\mathcal{O}(1/\epsilon)$ in (item 3.90). As discussed in Theorem 3.4 and Theorem 3.6, the $\mathcal{O}(1/\epsilon^2)$ oracle complexity is order optimal for the more general setting. Third, the specific stepsize choices listed above are required only to achieve the desired constant dependence. In practice, the same order-optimal stochastic oracle complexity can be obtained with simpler parameter-independent stepsize choices. For the non-strongly convex case, the same order-optimal stochastic oracle complexity of $\mathcal{O}(1/\epsilon^2)$ can be obtained if

$$\begin{aligned}
&\omega_t = t, \tau_{i,t} := (t-1)/2 \; \forall i \in \mathfrak{S}, \tau_{i,t} := \Theta(\sqrt{t}) \; \forall i \in \mathfrak{N} \cup \mathfrak{P}, \\
&\gamma_{i,t} := \Theta(\sqrt{t}) \; \forall i \in \mathfrak{N}_o, \; \eta_t := \Theta(\sqrt{t}).
\end{aligned} \tag{3.91}$$

For the strongly convex case with all outer layer function being smooth, the order-optimal oracle complexity of $\mathcal{O}(1/\epsilon)$ can be obtained if

$$\omega_t = t, \tau_{i,t} := (t-1)/2 \; \forall i \in \mathfrak{S}, \tau_{i,t} := \Theta(1) \; \forall i \in \mathfrak{P}, \; \eta_t := (t-1)\alpha/2.$$

3.4.3 Convergence Analysis

We sketch in this subsection the convergence analysis of the SSD method applied to the multi-layer smooth problem. It is structured similarly as that of subsection 3.2.4. We begin by presenting a general convergence result of the Q-gap function. We then specialize it to provide the concrete convergence rates in Theorem 3.7.

Proposition 3.3 *Consider a smooth multi-layer problem* (Equation 3.72) *with Lipschitz-continuity and smoothness constants defined in* (Equation 3.77)*, and stochastic arguments* $\tilde{y}_i^t(\xi)$ $0 \leq i \leq k-1$ *that satisfy the aggregate variance bounds in* (Equation 3.78)*. Let* $z^t := (x^t; \pi_{1:}^t)$ *be generated according to* (Equation 3.76) *with* $U_i = D_{f_i^*}$ $\forall i \in [k]$*. Fix a reference point* $z = (x^*; \pi_1 = \nabla f_1(y_1) \dots \pi_k = \nabla f_k(y_k))$ *, where* $y_i \in \mathbb{R}^{n_i}$ *may depend on* $\{z^t\}$*. If the following conditions are satisfied with some non-negative weights* ω_t*'s for all* $t \geq 1$*:*

$$\begin{aligned} &\omega_t = \theta_{t+1}\omega_{t+1},\ \omega_t(\tau_{i,t}+1) \geq \omega_{t+1}\tau_{i,t+1}\ \forall i \in [k], \\ &\eta_t \geq \textstyle\sum_{i=1}^k \frac{\theta_{t+1}M_{1:i-1}L_iM_{i+1:}^2}{\tau_{i,t+1}},\ \eta_N \geq \sum_{i=1}^k \frac{M_{1:i-1}L_iM_{i+1:}^2}{\tau_{i,N}+1}, \end{aligned} \tag{3.92}$$

the following Q*-gap bound is valid for* $\omega_0 = 0$*:*

$$\begin{aligned} &\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q(z^t; z)] + \omega_N(\eta_N + \alpha)\left\|x^N - x^*\right\|^2/2 \\ &\quad \leq \mathbb{E}\{\textstyle\sum_{t=1}^N [\omega_t\eta_t - \omega_{t-1}(\eta_{t-1}+\alpha)]\left\|x^{t-1}-x^*\right\|^2/2\} + \textstyle\sum_{i=1}^k \omega_1\tau_{i,1}D_{f_i^*}(\pi_i; \pi_i^0) \\ &\quad + \textstyle\sum_{t=1}^N \omega_t\tilde{\sigma}_x^2/(\eta_t+\alpha) + \sum_{i=1}^{k-1}\{M_{1:i-1}L_i\tilde{\sigma}_i^2[\sum_{t=1}^N \omega_t/(\tau_{i,t}+1)]\} \\ &\quad + \textstyle\sum_{i=1}^{k-1}\mathbb{E}\{\sum_{t=1}^N \omega_t\pi_{1:i-1}(\pi_i - \pi_{i,\mathbb{E}}^t)[\tilde{y}_i^t - \tilde{y}_i^t(\xi)]\}, \end{aligned} \tag{3.93}$$

where $\pi_{i,\mathbb{E}} = \nabla f_i(y_i)$ *for some* $y_i \in \mathbb{R}^{n_i}$ *and* $\pi_{i,\mathbb{E}}^t$ *is independent of* $\tilde{y}_i^t - \tilde{y}_i^t(\xi)$ *conditioned on* $\tilde{y}_i^t$*.*

Proof: First, let us develop a convergence bound for $Q_i \ \forall i \geq 1$ (c.f. (Equation 3.74)). The π_i^t update in (Equation 3.76) implies a (vector) three-point inequality given by

$$\begin{aligned}(\pi_i - \pi_i^t)\tilde{y}_i^t(\xi) + f_i^*(\pi_i^t) - f_i^*(\pi_i) + (\tau_{i,t}+1)D_{f_i^*}(\pi_i;\pi_i^t) + \tau_{i,t}D_{f_i^*}(\pi_i^t;\pi_i^{t-1}) \\ \leq \tau_{i,t}D_{f_i^*}(\pi_i;\pi_i^{t-1}).\end{aligned}$$

Let $\delta_i^t := \tilde{y}_i^t - \tilde{y}_i^t(\xi)$. Multiplying both sides of the above inequality with the non-negative weight $\pi_p := \pi_{1:i-1}$ leads to

$$\begin{aligned}\pi_p[(\pi_i - \pi_i^t)\tilde{y}_i^t + f_i^*(\pi_i^t) - f_i^*(\pi_i)] + (\tau_{i,t}+1)\pi_p D_{f_i^*}(\pi_i;\pi_i^t) + \tau_{i,t}\pi_p D_{f_i^*}(\pi_i^t;\pi_i^{t-1}) \\ \leq \tau_{i,t}\pi_p D_{f_i^*}(\pi_i;\pi_i^{t-1}) + \pi_p(\pi_i - \pi_i^t)\delta_i^t.\end{aligned} \tag{3.94}$$

Take $\pi_{i,\mathbb{E}}^t \leftarrow \arg\max_{\pi_i \in \Pi_i} \pi_i \tilde{y}_i^t - f_i^*(\pi_i) - \tau_{i,t}U_i(\pi_i;\pi_i^{t-1})$ such that it is conditionally independent of δ_i^t. We now apply the same argument as that of (Equation 3.103) to provide a bound for $\left\|\pi_p(\pi_i^t - \pi_{i,\mathbb{E}}^t)\right\|$. Observe $\pi_i\pi_i^t$ and $\pi_i\pi_{i,\mathbb{E}}^t$ can be regarded as functions of its argument, i.e., $\pi_p\hat{\pi}_i(\tilde{y}_i^t + \delta_i^t)$ and $\pi_p\hat{\pi}_i(\tilde{y}_i^t)$, where

$$\pi_p\hat{\pi}_i(y) = \arg\max_{\pi_i \in \Pi_i} \pi_p\pi_i y - \pi_p f_i^*(\pi_i) - \tau_{i,t}\pi_p D_{f_i^*}(\pi_i;\pi_i^{t-1}).$$

As a function of $\pi_p\pi_i$, the prox term $\pi_p f_i^*(\pi_i) + \tau_{i,t}\pi_p D_{f_i^*}(\pi_i;\pi_i^{t-1})$ have a strong convexity modulus of $(\tau_{i,t}+1)/(L_i\left\|\pi_i\right\|)$, so $\pi_p\hat{\pi}_i(y)$ should be Lipschtiz continuous with respect to its argument with a Lipschitz constant of $(L_i\left\|\pi_i\right\|)/(\tau_{i,t}+1)$. Thus we have $\left\|\pi_p(\pi_i^t - \pi_{i,\mathbb{E}}^t)\right\| \leq \left\|\pi_p\right\| L_i \left\|\delta_i^t\right\| /(\tau_{i,t}+1)$ and

$$\begin{aligned}\mathbb{E}[\pi_p(\pi_i - \pi_i^t)\delta_i^t] &= \mathbb{E}[\pi_p(\pi_i - \pi_{i,\mathbb{E}}^t)\delta_i^t] + \mathbb{E}[\pi_p(\pi_{i,\mathbb{E}}^t - \pi_i^t)\delta_i^t] \\ &\leq \mathbb{E}[\pi_p(\pi_i - \pi_{i,\mathbb{E}}^t)\delta_i^t] + L_i\mathbb{E}[\left\|\pi_p\right\|\left\|\delta_i^t\right\|^2]/(\tau_{i,t}+1) \\ &\leq \mathbb{E}[\pi_p(\pi_i - \pi_{i,\mathbb{E}}^t)\delta_i^t] + M_{1:i-1}L_i\tilde{\sigma}_i^2/(\tau_{i,t}+1).\end{aligned} \tag{3.95}$$

Next consider the left-hand-side of (Equation 3.94), we have

$$\begin{aligned}\pi_p(\pi_i - \pi_i^t)\tilde{y}_i^t =& \pi_p(\pi_i - \pi_i^t)\mathcal{L}_{i+1}(x^t; \pi_{i+1:}^t) \\ &- \underbrace{\{\pi_p(\pi_i - \pi_i^t)\pi_{i+1:}^t(x^t - x^{t-1}) - \theta_t \pi_p(\pi_i - \pi_i^t)\pi_{i+1:}^t(x^{t-1} - x^{t-2})\}}_{A_t}.\end{aligned}$$

Also, an argument similar to that of (Equation 3.38) implies

$$\begin{aligned}&\textstyle\sum_{t=1}^N \omega_t\{A_t - (\tau_{i,t}+1)\pi_p D_{f_i^*}(\pi_i; \pi_i^t) - \tau_{i,t}\pi_p D_{f_i^*}(\pi_i^t; \pi_i^{t-1}) + \tau_{i,t} D_{f_i^*}(\pi_i; \pi_i^{t-1})\} \\ &\quad\le \tau_{i,1} D_{f_i^*}(\pi_i; \pi_i^0) + \textstyle\sum_{t=1}^{N-1} \frac{\omega_t\theta_{t+1}}{2\tau_{i,t+1}}(M_{1:i-1}L_i M_{i+1:}^2)\left\|x^t - x^{t-1}\right\|^2 \\ &\qquad + \tfrac{\omega_N}{2(\tau_{i,N}+1)}(M_{1:i-1}L_i M_{i+1:}^2)\left\|x^N - x^{N-1}\right\|^2.\end{aligned} \tag{3.96}$$

Applying (Equation 3.95) and (Equation 3.96) to an ω_t-weighted sum of (Equation 3.94), we obtain

$$\begin{aligned}&\mathbb{E}[\textstyle\sum_{t=1}^N \omega_t Q_i(z^t; z)] \\ &\le \textstyle\sum_{t=1}^{N-1} \frac{\omega_t\theta_{t+1}}{2\tau_{i,t+1}}(M_{1:i-1}L_i M_{i+1:}^2)\left\|x^t - x^{t-1}\right\|^2 + \frac{\omega_N}{2(\tau_{i,N}+1)}(M_{1:i-1}L_i M_{i+1:}^2)\left\|x^N - x^{N-1}\right\|^2 \\ &\quad + \tau_{i,1} D_{f_i^*}(\pi_i; \pi_i^0) + \mathbb{E}[\textstyle\sum_{t=1}^N \omega_t \pi_p(\pi_i - \pi_{i,\mathbb{E}}^t)\delta_i^t] + \textstyle\sum_{t=1}^N \omega_t M_{1:i-1}L_i\tilde{\sigma}_i^2/(\tau_{i,t}+1).\end{aligned} \tag{3.97}$$

The convergence bound for Q_0 is similar to (Equation 3.41). Therefore, adding up the bounds for Q_k, Q_{k-1}, ..., and Q_0 and noting the stepsize requirement for η_t, we obtain the desired convergence bound for Q in (Equation 3.93).

□

Proof of Theorem 3.7 Clearly, Algorithm 6 is a concrete implementation of (Equation 3.76). With the specific choices of η_t in the theorem statement, the requirements of Proposition 3.3 are satisfied, so the convergence rates in (item 3.80) and (item 3.81) can be derived in a similar fashion as those of Theorem 3.1 and Theorem 3.2. □

Proof to Theorem 3.8 The analysis is a straightforward generalization to those of The-

orem 3.4, Theorem 3.5 and Theorem 3.7. □

3.5 Applications

In this section, we showcase the practical effectiveness of the SSD and nSSD methods by applying them to two specific problems: risk-averse optimization and additive composite optimization. A direct application of the proposed methods would lead to order-optimal oracle complexities. Moreover, the proposed methods can be easily modified to exploit specific problem structures to improve the constant dependence. In Subsection subsection 3.5.1, we show the modified nSSD algorithm can solve the risk-averse problem with almost the same oracle complexity (with respect to all problem parameters) as the simpler risk-neutral problem. In Subsection subsection 3.5.2, we show our general SSD method can achieve the same oracle complexity as the best algorithm specifically designed for additive composite optimization [42].

3.5.1 Risk Averse Optimization

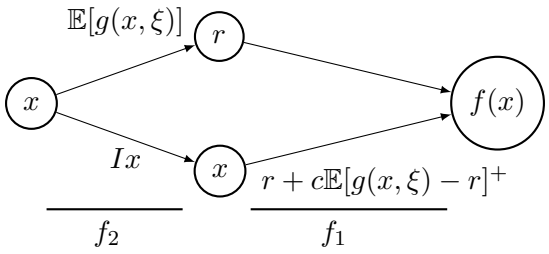


Figure 3.4: Two Layer Formulation for Semi-deviation.

First, we consider a risk-averse two-stage stochastic program given by

$$\min_{x\in X}\{\rho(Z(x)) := \mathbb{E}[g(x,\xi)] + c\mathbb{E}[g(x,\xi) - \mathbb{E}[g(x,\xi)]]^+\}. \tag{3.98}$$

Here, the random variable $Z(x) := g(x,\xi)$ denotes the cost incurred by the decision x under the scenario ξ. We select mean-upper-semideviation of order one [9] as the risk

measure ρ and use the trade-off parameter $c \in [0,1]$ to characterize the degree of the optimizer's risk aversiveness. For generality, we assume $g(x,\xi)$ to be non-smooth. For example, $g(x,\xi)$ in the two-stage linear program, the minimum total cost incurred by the first-stage decision x under the scenario ξ, is piecewise linear.

To apply our nSSD method in Algorithm Algorithm 5, we formulate (Equation 3.98) as a two-layer problem illustrated in Figure Figure 3.4. We assume $g(x,\xi)$ to be Lipschtiz continuous and to have bounded variances:

$$\left\|\mathbb{E}[g'(x,\xi)\mathbb{1}_{\{g(x,\xi)\geq r\}}]\right\| \leq M_g,$$
$$\mathbb{E}[\|g'(x,\xi)-\mathbb{E}[g'(x,\xi)]\|^2] \leq \sigma_G^2,\ \mathbb{E}[g(x,\xi)-\mathbb{E}[g(x,\xi)]]^2 \leq \sigma_g^2\ \forall x \in X\ \forall r \in \mathbb{R}.$$

To handle the non-smooth outer-layer function, we need a tri-conjugate reformulation to f_1 (c.f. (Equation 3.49)). We choose the domains $\tilde{\Pi}_1$ and V_1 to be

$$V_1 := \mathbb{B}^1(\bar{r}; \mathcal{D}_X M_g) \times X,\ \tilde{\Pi}_1 := [0,1] \times \mathbb{B}^n(0; M_g). \tag{3.99}$$

Here, $\mathbb{B}^m(x;l)$ denotes an m-dimensional ball centered at x with a radius l, and $\bar{r}$ denotes any possible value of $E[g(x,\xi)]$ for any $x \in X$. This $(V_1, \tilde{\Pi}_1)$ satisfies all requirements in (Equation 3.52) except for the non-negativity condition, $\tilde{\Pi}_{1,2:} = \mathbb{B}^n(0;M_g) \not\subset \mathbb{R}^n_+$. Nevertheless, because the input to $\pi_{1,2:}$ is a linear function, Ix, the results in Section section 3.3 are still valid. We can directly apply the nSSD method in Algorithm Algorithm 5 to find an ϵ-optimal solution, and the stochastic oracle complexity is $\mathcal{O}\{(M_g^4\mathcal{D}_X^4 + M_g^2\sigma_g^2 + \sigma_{g'}^2 M_g^2 \mathcal{D}_X^2)/\epsilon^2\}$ (see Theorem Theorem 3.5).

To improve the constant dependence, we need to look into the structure of $f_1 : (r,x) \to \mathbb{R}$. As illustrated in Figure Figure 3.4, the mappings for the first coordinate r and for the next n coordinates x are quite different. The domains $(V_1, \tilde{\Pi}_1)$ also have some block structures (see (Equation 3.99)); they are constructed from Cartesian products of two separable blocks, $V_1 = V_{1,1} \times V_{1,2:}$ and $\tilde{\Pi}_i = \tilde{\Pi}_{1,1} \times \tilde{\Pi}_{1,2:}$. As a result, the v_i (π_1) prox mappings should

be decomposed into the prox-mapping for $v_{1,1}$ ($\pi_{1,1}$) and that for $v_{1,2:}$ ($\pi_{1,2:}$). Indeed, in the prox-mapping for v_1, the first coordinate of the argument, $1 - c\mathbb{1}_{\{g(x^{t-1},\xi_1)\geq v_1^{t-1}\}}$, differs greatly from the next n coordinates of the argument, $cg'(x^{t-1},\xi_1)\mathbb{1}_{\{g(x^{t-1},\xi_1)\geq v_1^{t-1}\}}$. In the prox-mapping for π_1, only the first coordinate of the argument, $g(x^{t-1},\xi_2^1)$, is stochastic. These separable structures motivate us to use different stepsizes to update the first block, $v_{1,1}$ and $\pi_{1,1}$, compared to the remaining block, $v_{1,2:}$ and $\pi_{1,2:}$. Specifically, we modify Line 3 and 4 of Algorithm Algorithm 5 to the following:

$$
\begin{aligned}
&\pi_{1,1}^t \leftarrow \arg\max_{\pi_{1,1}\in\tilde{\Pi}_{1,1}} \pi_{1,1}[\tilde{y}_{1,1}^t(\xi) - v_{1,1}^{t-1}] - \tau_{1,t}^r \left\|\pi_{1,1} - \pi_{1,1}^{t-1}\right\|^2/2,\\
&\pi_{1,2:}^t \leftarrow \arg\max_{\pi_{1,2:}\in\tilde{\Pi}_{1,2:}} \pi_{1,2:}[\tilde{y}_{1,2:}^t(\xi) - v_{1,2:}^{t-1}] - \tau_{1,t}^\nabla \left\|\pi_{1,2:} - \pi_{1,2:}^{t-1}\right\|^2/2,\\
&v_{1,1}^t \leftarrow \arg\min_{v_{1,1}\in V_{1,1}} \langle f_{1,1}'(v_1^{t-1},\xi_1) - \pi_{1,1}^t, v_{1,1}\rangle + \gamma_{1,t}^r \left\|v_{1,1} - v_{1,1}^{t-1}\right\|^2/2,\\
&v_{1,2:}^t \leftarrow \arg\min_{v_{1,2:}\in V_{1,2:}} \langle f_{1,2:}'(v_1^{t-1},\xi_1) - \pi_{1,2:}^t, v_{1,2:}\rangle + \gamma_{1,t}^\nabla \left\|v_{1,2:} - v_{1,2:}^{t-1}\right\|^2/2.
\end{aligned}
$$

We set $\tau_{1,t}^\nabla = 0$, $\tau_{1,t}^r = \sigma_g\sqrt{t}$, $\gamma_{1,t}^\nabla = \max\{M_g,\sigma_G\}\sqrt{t}/\mathcal{D}_X$, and $\gamma_{1,t}^r = \sqrt{t}/(\mathcal{D}_X M_g)$, while choosing η_t similarly as Theorem Theorem 3.5. Then the resulting oracle complexity can be improved to

$$\mathcal{O}(\{M_g^2 D_X^2 + \sigma_g^2 + \sigma_G^2\mathcal{D}_X^2\}/\epsilon^2).$$

Comparing it to the $\mathcal{O}(\{M_g^2 D_X^2 + \sigma_G^2\mathcal{D}_X^2\}/\epsilon^2)$ oracle complexity for solving the risk-neutral two-stage program, the only extra cost is $\mathcal{O}(\sigma_g^2/\epsilon^2)$, which arises from the need to estimate the function value. Therefore, the modified nSSD algorithm can solve the more challenging risk-averse problem almost as efficiently as solving the simpler risk-neutral problem up to all problem parameters.

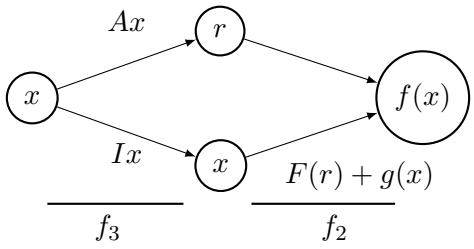


Figure 3.5: Two Layer Formulation for Additive Composite (Equation 3.100).

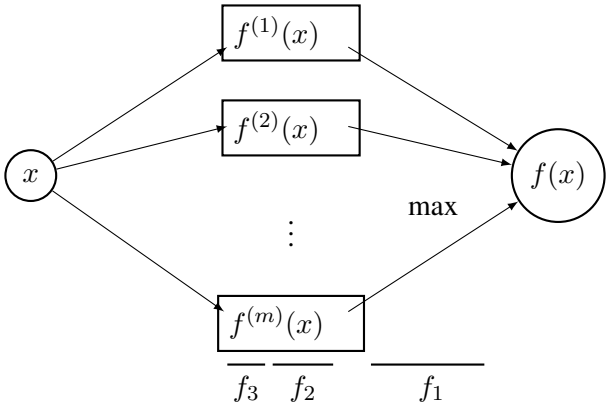


Figure 3.6: Three Layer Formulation for (Equation 3.101).

3.5.2 Stochastic Composite Optimization

Next, we consider a stochastic composite optimization problem that frequently arises in machine learning and data analysis [42]:

$$\min_{x \in X} \{ f(x) := F(Ax) + g(x) \equiv \max_{\pi_F \in \Pi_F} \langle \pi_F, Ax \rangle - F^*(\pi_F) + g(x) \}. \tag{3.100}$$

Here, F represents a structured non-smooth function, for example, a regularity term like the total variation function, and g represents a stochastic smooth function, for example, the data fidelity loss function. Since the dimension of A is usually large, we assume stochastic oracles to return unbiased estimators $A(\xi)$, $A^\top(\xi)$, and $g'(x, \xi)$ for A, $A^\top$ and $g'(x)$, respectively. Furthermore, we assume their variances to be uniformly bounded by σ_A^2, $\sigma_{A^\top}^2$ and σ_G^2, respectively.

The problem (Equation 3.100) can be formulated as a two-layer problem, as shown

Algorithm 8 SSD Algorithm for Composite Optimization

Input: $x_{-1} = x_0 \in X$ and $\pi_F^0 \in \Pi_F$.

1: Set $\underline{y}_g^0 := x_0$ and call $\mathcal{SO}$ to obtain estimate $A(\xi_{3,0}^2)$.
2: **for** $t = 1, 2, 3...N$ **do**
3: Call $\mathcal{SO}$ to obtain estimates $A(\xi_{3,t}^2)$ and $A^\top(\xi_{3,t}^0)$.
4: Let $\tilde{x}^t := x^{t-1} + \theta_t(x^{t-1} - x^{t-2})$.
Let $\underline{y}_g^t := (\tau_{g,t}\underline{y}_g^{t-1} + \tilde{x}^t)/(1+\tau_{g,t})$ and call $\mathcal{SO}$ to obtain $\pi_g^t(\xi_{2,t}^0) := g'(\underline{y}_g^t, \xi_{2,t}^0)$.
5: Let $\tilde{y}_F^t(\xi) := A(\xi_{3,t}^2)x^{t-1} + A(\hat{\xi}_{3,t-1})(x^{t-1} - x^{t-2})$.
Compute $\pi_F^t := \arg\min_{\pi_F \in \Pi_F} -\langle \pi_F, \tilde{y}_F^t(\xi)\rangle + F^*(\pi_F) + \tau_{F,t}\left\|\pi_F - \pi_F^{t-1}\right\|^2/2$
6: Set $x^t := \arg\min_{x\in X}\langle \pi_g^t(\xi_{2,t}^0) + A^\top(\xi_{3,t}^0)\pi_F^t, x\rangle + \eta_t\left\|x - x^t\right\|^2/2$.
7: **end for**
8: Return $\bar{x}^N := \sum_{t=1}^N \omega_t x^{t+1} / \sum_{t=1}^N \omega_t$.

in Figure Figure 3.5. The outer layer f_2 is the additive composite of a structured non-smooth function $F(r)$ and a smooth function $g(x)$. By treating the outer f_2 as a general non-smooth layer function, we can apply the nSSD method in Algorithm Algorithm 5 to achieve an order-optimal oracle complexity of $\mathcal{O}(1/\epsilon^2)$. To further improve the constant dependence, we need to exploit the block structure of f_2 in a similar fashion as Subsection subsection 3.5.1. Since $f_2(x, r) := F(r) + g(x)$, the dual variable π_2 can be decomposed into blocks attributable to F and g respectively:

$$\pi_2 := [\ \pi_F \quad | \quad \pi_g\ ].$$

Accordingly, the conjugate function and the component gap function Q_2 (see (Equation 3.12)) also decompose,

$$\begin{aligned} f_2^*(\pi_2) &:= \max_{x,r}\langle x, \pi_g\rangle + \langle r, \pi_F\rangle - F(r) - g(x) \\ &= [\max_r\langle r, \pi_F\rangle - F(r)] + [\max_x\langle x, \pi_g\rangle - g(x)] = F^*(\pi_F) + g^*(\pi_g). \\ Q_2(z^t, z) &:= (\pi_2 - \pi_2^t)\mathcal{L}_3(x^t; \pi_3^t) - [f_2^*(\pi_2) - f_2^*(\pi_2^t)] \\ &= \underbrace{(\pi_F - \pi_F^t)(Ax^t) - [F^*(\pi_F) - F^*(\pi_F^t)]}_{Q_{2,F}} + \underbrace{(\pi_g - \pi_g^t)x^t - [g^*(\pi_g) - g^*(\pi_g^t)]}_{Q_{2,g}}. \end{aligned}$$

This decomposition motivates the use of separate prox-mappings for π_F and π_g in Lines 4 and 5 of the proposed Algorithm Algorithm 8. If L_g denotes the Lipschitz-smoothness constant of g and M_F is an upper bound for $\|\pi_F\| \ \forall \pi_F \in \Pi_F$, we can establish the following oracle complexity (see Theorems Theorem 3.1 and Theorem 3.3) with appropriately chosen stepsizes:

$$\mathcal{O}\{\frac{\sqrt{L_g}\|x-x^*\|}{\sqrt{\epsilon}} + \frac{\|A\|\mathcal{D}_X M_F}{\epsilon} + \frac{(\sigma_A^2+\sigma_{A^\top}^2)\mathcal{D}_X^2 M_F^2}{\epsilon^2} + \frac{\sigma_G^2\mathcal{D}_X^2}{\epsilon^2}\}.$$

Compared to the specialized accelerated primal-dual (APD) method [42] designed for solving problem (Equation 3.100), Algorithm Algorithm 8 achieves the same oracle complexity. However, our approach is more general. It allows for easy extension to handle more complicated problems where f in (Equation 3.100) is but one sub-component. In particular, if $f^{(i)} := g^{(i)}(x) + F^{(i)}(A^{(i)}x)$, a minimax sub-problem that frequently arises in either constrained optimization or multi-objective optimization is given by

$$\min_{x\in X}\{f(x) := \max\{f^{(1)}(x), f^{(2)}(x), \ldots, f^{(m)}(x)\} \equiv \max_{\pi_1\in\Delta_m^+} \sum_{i=1}^{m} \pi_1^{(i)} f^{(i)}(x)\}^{④}. \quad (3.101)$$

Clearly, (Equation 3.101) admits a three-layer formulation shown in Figure Figure 3.6 where the outermost layer f_1 is structured non-smooth. To derive a convergent SSD method, We only need to add an additional prox-mapping for π_1 before Line 6 in Algorithm Algorithm 8. If the variances for $g^{(i)}(\cdot,\xi_i)$, $\nabla g^{(i)}(\cdot,\xi_i)$, $A^{(i)}(\xi_i)$, ${A^{(i)}}^\top(\xi_i)$ are uniformly bounded by σ_g^2, σ_G^2, σ_A^2 and $\sigma_{A^\top}^2$, and if $\left\|A^{(i)}\right\|$, $M_{F^{(i)}}$ and $\left\|\nabla g^{(i)}(\cdot)\right\|$ are uniformly bounded by $\|A\|$, M_F and M_g, a straightforward application of Theorem Theorem 3.8 implies an oracle complexity of

$$\mathcal{O}\{\frac{\sqrt{L_g}\|x-x^*\|}{\sqrt{\epsilon}} + \frac{\sqrt{m}\|A\|\mathcal{D}_X M_F}{\epsilon} + \frac{m(\sigma_A^2+\sigma_{A^\top}^2)\mathcal{D}_X^2 M_F^2}{\epsilon^2} + \frac{m(\sigma_{g'}^2\mathcal{D}_X^2+\sigma_g^2)}{\epsilon^2} + \frac{\sqrt{m}M_g}{\epsilon}\}.$$

④ $\Delta_+^m := \{\pi_1 \in \mathbb{R}_+^m | \sum_{i=1}^m \pi_1^{(i)} = 1\}$ is the probability simplex.

Additionally, if the entropy Bregman distance function is selected as the prox-function for the π_1-prox mapping, similar to [55], the above complexity can be improved to be nearly independent of the number of sub-components,

$$\mathcal{O}\{\frac{\sqrt{L_g}\|x-x^*\|}{\sqrt{\epsilon}} + \frac{\sqrt{\log(m)}\|A\|\mathcal{D}_X M_F}{\epsilon} + \frac{\log(m)(\sigma_A^2+\sigma_{A^\top}^2)\mathcal{D}_X^2 M_F^2}{\epsilon^2} + \frac{\log(m)(\sigma_g^2\mathcal{D}_X^2+\sigma_g^2)}{\epsilon^2} + \frac{\sqrt{\log(m)}M_g}{\epsilon}\}.$$

3.6 Conclusion

To sum up, this paper characterizes the order of stochastic oracle complexity for the convex NSCO problem under a mild compositional convexity assumption. We propose order-optimal SSD/nSSD methods to solve the general multi-layer problem constructed from an arbitrary composition of smooth, structured non-smooth, and general non-smooth layer functions. Additionally, we develop lower complexity results to illustrate the $O(1/\epsilon^2)$ oracle complexity for the strongly convex and outer-non-smooth problem to be unimprovable, demonstrating an intrinsic difficulty for solving the NSCO problem. To demonstrate the effectiveness of our proposed methods, we apply the SSD/nSSD method to two motivating applications. For one application, the method achieves the same performance as the best algorithm specifically designed for the problem. For the other application, it provides new insights into the oracle complexity of the problem class.

In future work, we will attempt to demonstrate the empirical performance of our proposed methods by conducting numerical studies on real-world datasets, and to establish their almost-sure convergence properties. An interesting research direction is to investigate the possibility of an order-optimal algorithm without the compositional convexity assumption.

3.7 Appendix

3.7.1 Technical lemmas for vector-valued functions

Proof of Lemma 3.2 This is a direct consequence of the conjugate duality relationship [10],

$$\pi_i \in \partial g_i(\underline{y}) \Leftrightarrow \pi_i \in \underset{\bar{\pi}_i \in \text{dom}(g_i^*)}{\arg\max}\, \bar{\pi}_i \underline{y} - g_i^*(\bar{\pi}_i).$$

In particular, since $\underline{y}^{t-1} \in \partial g^*(\pi^{t-1})$, the i^{th} row of π^t satisfies

$$\begin{aligned}
& \pi_i^t \in \underset{\pi_i \in \text{dom}(g_i^*)}{\arg\min}\, -\langle \pi_i, y^t \rangle + g_i^*(\pi_i) + \tau_t D_{g_i^*}(\pi_i; \pi_i^{t-1}), \\
\Longleftrightarrow\ & \pi_i^t \in \underset{\pi_i \in \text{dom}(g_i^*)}{\arg\min}\, -\langle \pi_i, y^t \rangle - \tau_t \langle \pi_i, \underline{y}^{t-1} \rangle + (1+\tau_t) g_i^*(\pi_i), \\
\Longleftrightarrow\ & \pi_i^t \in \underset{\pi_i \in \text{dom}(g_i^*)}{\arg\min}\, -\langle \pi_i, \underbrace{\frac{y^t + \tau_t \underline{y}^{t-1}}{1+\tau_t}}_{\underline{y}^t} \rangle + g_i^*(\pi_i) \ \Longleftrightarrow\ \pi_i^t = g_i'(\underline{y}^t) \in \partial g_i(\underline{y}^t).
\end{aligned}$$

Therefore, we have $\pi^t = g'(\underline{y}^t)$.

□

Lemma 3.6 *Let a closed convex and proper vector-vectored function g be defined on R^n. Let it be L_g-smooth , i.e.,*

$$\|\nabla g(y) - \nabla g(\bar{y})\|^{\text{⑤}} \leq L_g \|y - \bar{y}\|, \forall y, \bar{y} \in \mathbb{R}^n.$$

Let g^* and D_{g^*} denote its (component-wise) conjugate function and (component-wise) conjugate Bregman's distance function. Then given an m-dimensional non-negative weight vector w, we have

$$\|w\|\, w^\top D_{g^*}(\bar{\pi}; \pi) \geq \frac{\left\| w^\top (\bar{\pi} - \pi) \right\|^2}{2L_g} \tag{3.102}$$

if $\bar{\pi} = \nabla g(\bar{y})$ and $\pi = \nabla g(y)$ for some $\bar{y}, y$.

⑤ l_2 operator norm.

Proof: First, if $w = 0$, (Equation 3.102) is clearly true. Now, assume $w \neq 0$. The definition of operator norm implies

$$\left\|u^\top(g'(y) - g'(\bar{y}))\right\| \leq L_g \left\|y - \bar{y}\right\| \ \forall u \text{ with } \|u\| = 1.$$

So the one-dimensional $g_{u_w}(y) := u_w^\top g(y)$ with $u_w := \frac{w}{\|w\|}$ is L_g-Lipschitz smooth and its Fenchel conjugate $g_{u_w}^*$ is $1/L_g$ strongly convex. More specifically, since $g'_{u_w}(y) = u_w^\top g'(y)$, we have

$$g_{u_w}^*(u_w^\top \pi) - g_{u_w}^*(u_w^\top \bar{\pi}) - u_w^\top(\pi - \bar{\pi})\bar{y} \geq \frac{1}{2L_g}\left\|u_w^\top(\pi - \bar{\pi})\right\|^2,$$
$$\text{if } \bar{\pi} = g'(\bar{y}) \text{ , i.e., } u_w^\top \bar{\pi} = g'_{u_w}(\bar{y}).$$

Thus the key to showing (Equation 3.102) is to relate $g_{u_w}^*(u_w^\top \pi)$ to $u_w^\top g^*(\pi)$.

Those two quantities are quite different in general. For $g_{u_w}^*(u_w^\top \bar{\pi}) := \max_y u_w^\top \bar{\pi} y - g_{u_w}(y)$, we can choose only one overall maximizer, y^*, but for $u_w^\top g^*(\bar{\pi}) = \sum_j u_{w,j} \max_{y_j} \bar{\pi}_j y_j - g_j(y_j)$, different maximizers $\bar{y}_j^*$'s can be selected for different $\bar{\pi}_j$'s. So we have $g_{u_w}^*(u_w^\top \pi) \leq u_w^\top g^*(\pi)\ \forall \pi$. However, if $\tilde{\pi}$ is associated with some primal solution $\tilde{y}$, i.e., $\tilde{\pi} = g'(\tilde{y})$, all those $\tilde{y}_j$'s are the same. Let $\tilde{\pi} = g'(\tilde{y})$ such that $\tilde{\pi}_j = g'_j(\tilde{y})$. The conjugate duality implies that $g^*(\tilde{\pi}) = \tilde{\pi}\tilde{y} - g(\tilde{y})$, so

$$g_{u_w}^*(u_w^\top \tilde{\pi}) := \max_{\bar{y}} u_w^\top \tilde{\pi} \bar{y} - g_{u_w}(\bar{y}) \geq u_w^\top(\tilde{\pi}\tilde{y} - g(\tilde{y})) = u_w^\top g^*(\tilde{\pi}).$$

Therefore, $g_{u_w}^*(u_w^\top \tilde{\pi}) = u_w^\top g^*(\tilde{\pi})$ holds if $\tilde{\pi}$ is associated with some $\tilde{y}$.

Now returning to the π and $\bar{\pi}$ in the lemma statement, since $\bar{\pi} = g'(\bar{y})$ and $\pi = g'(y)$,

we have

$$u_w^\top D_{g^*}(\bar{\pi}, \pi) = u_w^\top g^*(\pi) - u_w^\top g^*(\bar{\pi}) - u_w^\top(\pi - \bar{\pi})\bar{y}$$
$$= g^*_{u_w}(u_w\pi) - g^*_{u_w}(u_w\bar{\pi}) - u_w^\top(\pi - \bar{\pi})\bar{y} \geq \frac{1}{2L_g}\left\|u_w^\top(\pi - \bar{\pi})\right\|^2.$$

Then (Equation 3.102) follows from multiplying both sides of the above inequality by $\|w\|^2$. □

The following lemma provides a bound on the error incurred from utilizing a stochastic argument during prox update.

Lemma 3.7 *Let $\Pi \subset \mathbb{R}^m$ be a non-empty closed and convex domain and let function $u(\pi)$ be μ-strongly convex. Let $\hat{\pi}$ be generated via a prox-mapping with the argument $g + \delta$, $\hat{\pi} \leftarrow \arg\min_{\pi\in\Pi}\langle\pi, g + \delta\rangle + u(\pi)$, where δ denote a noise term with $\mathbb{E}[\delta] = 0$ and $\mathbb{E}[\|\delta\|^2] \leq \sigma^2$. Then $|\mathbb{E}[\langle\hat{\pi}, \delta\rangle]| \leq \sigma^2/\mu$.*

Proof: Let $\pi(y) := \arg\min_{\pi\in\Pi}\langle\pi, y\rangle + u(\pi)$. The μ-strong convexity of $u(\pi)$ implies that $\pi(y)$ is an $1/\mu$-Lipschitz continuous function of y (see [31]). Define a auxiliary point $\bar{\pi} := \arg\min_{\pi\in\Pi}\langle\pi, g\rangle + u(\pi)$ which is independent of δ, i.e., $\mathbb{E}[\langle\delta, \bar{\pi}\rangle] = 0$. The $1/\mu$-Lipschitz continuity of $\pi(y)$ implies

$$\|\bar{\pi} - \hat{\pi}\| \leq \|\delta\|/\mu. \tag{3.103}$$

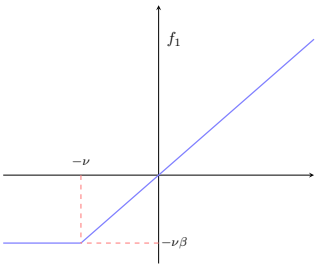
Thus we get

$$|\mathbb{E}\langle\hat{\pi}, \delta\rangle| \leq |\mathbb{E}\langle\hat{\pi} - \bar{\pi}, \delta\rangle| + |\mathbb{E}\langle\bar{\pi}, \delta\rangle|$$
$$\leq \mathbb{E}[\|\bar{\pi} - \hat{\pi}\|\,\|\delta\|] \leq \mathbb{E}[\|\delta\|^2/\mu] \leq \sigma^2/\mu.$$

□

3.7.2 Lower Complexity Bounds

We present in this subsection the detailed analysis for lower complexity results in Subsection subsection 3.2.3 and subsection 3.3.2 . First, we state an one-dimensional hard problem which will be useful for proving both results. Parameterized by $(\alpha, \sigma, \nu, \beta)$ with $\sigma \geq \nu$, the problem is given by



$$
\begin{aligned}
& f(x) := f_1(\mathbb{E}[f_2(x, \xi_2)]) + \alpha \|x\|^2 / 2, \\
& X := [-2\nu, 2\nu], \text{ where} \\
& f_1(y_1) := \beta \max\{y_1, -\nu\}, \\
& f_2(x, \xi_2) := x + \xi_2, \\
& \text{with iid r.v. } \xi_2 := \begin{cases} -\nu & \text{w.p. } 1-q \\ \nu(1-q)/q & \text{w.p. } q, \end{cases} \\
& \text{and } q := \nu^2/\sigma^2.
\end{aligned} \tag{3.104}
$$

Observe that f_1 is β-Lipschitz continuous and the variance of $f_2(x, \xi_2)$ satisfies

$$\mathbb{E} \|f_2(x, \xi_2) - \mathbb{E}[f_2(x, \xi)]\|^2 = \mathbb{E}[\|\xi_2\|^2] \leq \sigma^2.$$

The key to our construction is to show the reachable subspace of x being restricted to 0, i.e., $\mathcal{X}^t = \{0\}$, if a certain condition is met for all generated stochastic estimators. The next technical lemma characterizes the probability and the optimality gap of that scenario.

Lemma 3.8 *The following results are valid for* (Equation 3.104).

a) $f(0) - f(x^*) \geq \min\{\beta\nu, \beta^2/\alpha\}/2$.

b) If ξ_2^j *denotes the* j^{th} *query to* $\mathcal{SO}_2$ *and* $N < \sigma^2/(4\nu^2)$*, then* $\mathbb{P}\{f_2(0, \xi_2^l) = -\nu \ \forall l \leq N\} > 1/2$.

Proof: Part a) can be derived from the first order optimality condition. Since $f(x) = \beta \max\{x, -\nu\} + \alpha \|x\|^2/2$, the optimal solution and the optimal objective value are

$$x^* = \begin{cases} -\beta/\alpha & \text{if } \alpha > \beta/\nu \\ -\nu & \text{if } \alpha \in \frac{\beta}{\nu}[0,1], \end{cases} \Rightarrow f(x^*) \leq \begin{cases} -\beta^2/(2\alpha) & \text{if } \alpha > \beta/\nu \\ -\beta\nu/2 & \text{if } \alpha \in \frac{\beta}{\nu}[0,1]. \end{cases}$$

So the inequality in part a) represents an uniform lower bound on $f(0) - f(x^*)$. Part b) follows from the algebraic fact that $(1-p)^t > 3/4 - tp$ if $tp < 1/4$:

$$\mathbb{P}\{f_2(0, \xi_2^l) = -\nu \; \forall l \leq N\} = (1-q)^N > 3/4 - Nq > 1/2.$$

□

Now we are ready to prove the lower bound results.

Proof for Theorem Theorem 3.4: Consider applying the abstract scheme in (Equation 3.34) to the problem in (Equation 3.104). A structured non-smooth formulation to f_1 is given by

$$f_1(y_1) = \max_{\pi_1 \in [0,\beta]} \pi_1 y_1 - \nu(\beta - \pi_1),$$

where $\Pi_1 = [0, \beta]$ and $f_1^*(\pi_1) = \nu(\beta - \pi_1)$. Choosing $y_1^0 = -\nu$, we have $\mathcal{X}^0 = \Pi_1^0 = \{0\}$ and $\mathcal{Y}_1^0 = \{-\nu\}$. Now assume $\mathcal{X}^{t-1} = \Pi_1^{t-1} = \{0\}$ and $\mathcal{Y}_1^{t-1} = \{-\nu\}$, then $f_2(0, \xi_2^t) = -\nu$ (c.f. (Equation 3.34)) implies $\mathcal{Y}_1^t = \{-\nu\}$ and $\Pi_1^t = \{0\}$ since

$$\begin{aligned} \pi_1^t &= \arg\max_{\pi_1 \in \Pi_1} \langle \pi_1, y_1^t \rangle - f_1^*(\pi_1) - \tau_{1,t} \left\| \pi_1 - \pi_1^{t-1} \right\|^2 /2 \\ &= \arg\max_{\pi_1 \in [0,\beta]} \pi_1 (y_1^t + \nu) - \beta\nu - \tau_{1,t} \left\| \pi_1 - \pi_1^{t-1} \right\|^2 /2 \\ &= \arg\min_{\pi_1 \in [0,\beta]} \tau_{1,t} \|\pi_1\|^2 /2 = 0. \end{aligned}$$

It then follows $\mathcal{X}^t = \{0\}$. Such an argument can be applied recursively to show $\mathcal{X}^N = \{0\}$

if the event $B^N := \{f_2(0, \xi_2^l) = -\nu \; \forall l \leq N\}$ occurs.

Now selecting $\beta = M_1$, $\nu = 4\epsilon/M_1$, $\alpha = \bar{\alpha}$ and $\sigma = \sigma_{f_2}$, the hard problem in (Equation 3.104) satisfies the hard problem requirements in the theorem statement. Moreover, with $N < \sigma_{f_2}^2 M_1^2/(4\epsilon^2)$, Lemma 3.8 implies $\mathbb{P}(B^N) > 1/2$ such that

$$\mathbb{E}[f(x^N) - f(x^*)] \geq \mathbb{P}(B^N)\mathbb{E}[f(0) - f(x^*)] > \min\{\beta\nu, \beta^2/\alpha\}/4 = \epsilon. \tag{3.105}$$

Thus it takes at least $\Omega(M_1^2\sigma_{f_2}^2/\epsilon^2)$ $\mathcal{SO}_2$ queries to obtain an ϵ-optimal solution.

Proof for Theorem Theorem 3.6 The analysis is similar to that of Theorem Theorem 3.4. With $v_1^0 = -\nu$, we need to show $\mathcal{M}_1^N = \mathcal{X}^N = \{0\}$ if $f_2(0, \xi_2^j) = -\nu \; \forall j \leq N$. □We remark that the lower complexity bound of $\Omega(M_1^2\sigma_{f_2}^2/\epsilon^2)$ is applicable beyond the first-order schemes like (Equation 3.34) and (Equation 3.64). In fact, it is not hard to use the hard instance in (Equation 3.104) to show that at least $\Omega(M_1^2\sigma_{f_2}^2/\epsilon^2)$ samples are required by any SAA-type method to find an ϵ-optimal solution for the strongly convex NSCO problem with either a structured non-smooth or a general non-smooth outer-layer function.

CHAPTER 4

RISK AVERSE OPTIMIZATION OVER A DISTRIBUTED NETWORK

4.1 Background and Our Contribution

Consider the following risk-averse optimization problem over a star-shape (worker-server) communication network [62]:

$$\min_{x\in X}\{f(x) := \max_{p\in P} \textstyle\sum_{i=1}^{m} p_i f_i(x) - \rho^*(p) + u(x)\}, \tag{4.1}$$

where $P \subseteq \Delta_+^m := \{p \in \mathbb{R}^m | \sum_{i=1}^m p_i = 1, p_i \geq 0\}$ and $X \subseteq \mathbb{R}^n$ and $\Pi_i \subseteq \mathbb{R}^{m_i}$ are closed and convex, and functions $f_i(x)$, $u(x)$, and $\rho^*(p)$ are proper closed and convex. We assume the scenario (or local) cost function f_i to be only available to worker node i and focus on the situation where f_i's are either all smooth or all structured non-smooth. We use the following generic representation for both types of f_i's:

$$f_i(x) = \max_{\pi_i\in\Pi_i} \langle A_i x, \pi_i\rangle - f_i^*(\pi_i),$$

where Π_i is a closed convex set and f_i^* is a proper, closed and convex function. Specifically, if f_i is smooth, A_i is the identity matrix, $I \in \mathbb{R}^{n\times n}$, f_i^* is the Fenchel conjugate to f_i, and $\Pi_i = \mathrm{dom}(f_i^*)$[①]. If f_i is structured non-smooth [31], then $A_i \in \mathbb{R}^{m_i\times n}$ is a linear operator, Π_i is bounded, and the f_i^*-prox mapping can be solved efficiently [28]. This type of structured non-smooth function has found a wide range of applications, including total variation regularization in image processing [63], low-rank tensor [64, 65], overlapped group lasso [66, 67], and graph regularization [68, 67]. Additionally, we assume the (strongly) convex regularization term $u(x)$ and the risk measure (ρ^*, P) are available to the server node.

[①] $\mathrm{dom}(\mathrm{f_i^*}) := \{\pi_\mathrm{i} \in \mathbb{R}^\mathrm{n} : \mathrm{f_i^*}(\pi_\mathrm{i}) < \infty\}$.

If the ambiguity set P consists of only a fixed probability vector $\bar{p}$, say the empirical distribution, (Equation 4.1) is called risk-neutral, and it can be written as a finite-sum problem (see Chapter 5 of [28]):

$$\min_{x \in X} \sum_{i=1}^{m} \bar{p}_i f_i(x) + u(x). \tag{4.2}$$

However, if the costs among workers are imbalanced (different importance, limited availability of data, etc.), taking an average over the costs across workers might be meaningless or operationally wrong. In such cases, non-trivial ρ^* and P in (Equation 4.1) generalizes risk-neutral optimization to risk-averse optimization and distributionally robust optimization (DRO). Specifically, if $\boldsymbol{f} := (f_1, \ldots, f_m)$ denotes the scenario costs and ρ is a convex risk measure, it can be formulated as (Equation 4.1) using Fenchel conjugates (see Definition 6.4 and Theorem 6.5 of [9]):

$$\rho(\boldsymbol{f}) := \arg\max_{p \in P} \langle p, \boldsymbol{f} \rangle - \rho^*(p). \tag{4.3}$$

For example, if we denote the (reference) probability mass function by $\bar{p}$, some widely used risk measures and their conjugates are given as follows.

- Mean semideviation of order r:

  $$\rho(\boldsymbol{f}) = \textstyle\sum_{i=1}^{m} \bar{p}_i f_i + c(\sum_{i=1}^{m} \bar{p}_i [f_i - \mathbb{E}\boldsymbol{f}]_+^r)^{1/r} = \max_{p \in P} \langle p, \boldsymbol{f} \rangle,$$

  where the ambiguity set $P := \{p \in \Delta_+^m : \exists \zeta_i \geq 0 \text{ s.t. } p_i = \bar{p}_i(1 + \zeta_i - \langle \zeta, \bar{p} \rangle), \|\zeta\|_s \leq c\}$, $c \in [0, 1]$ and $\|\cdot\|_s$ is the conjugate norm to $\|\cdot\|_r$, i.e., $1/s + 1/r = 1$.

- Entropic risk:

  $$\rho(\boldsymbol{f}) = \tau^{-1} \log \textstyle\sum_{i=1}^{m} \bar{p}_i \exp(\tau f_i) = \max_{p \in \Delta_m^+} \langle p, \boldsymbol{f} \rangle - \tau^{-1} \sum_{i=1}^{m} p_i \log(p_i / \bar{p}_i).$$

- Distributionally robust objective: $\rho(\boldsymbol{f}) := \sup_{p \in P} \langle \boldsymbol{f}, p \rangle$ for some uncertainty set P.

The incorporation of all the above risk measures makes our problem (Equation 4.1) more challenging than the finite-sum problem (Equation 4.2). We note that (Equation 4.1) also covers a popular risk measure CV@R with $\rho(\boldsymbol{f}) = \max_{p \in \Delta_+^m, p_i \in [0, \bar{p}_i/\alpha]} \langle p, \boldsymbol{f} \rangle$, where the parameter $\alpha > 0$ captures the degree of risk aversion. The risk measure admits a finite-sum reformulation, $\rho(\boldsymbol{f}) = \inf_t \sum_{i=1}^m \bar{p}_i \{[f_i - t]_+/\alpha + t\}$, but the function $\tilde{f}_i(x,t) := [f_i(x) - t]_+/\alpha + t$ is nonsmooth with a very large Lipschitz-continuity constant, even if the original f_i is smooth. In contrast, our conjugate formulation avoids the situation.

As alluded to earlier, we assume the communication network to have a star-topology where a computationally powerful central server node is connected directly to many worker nodes. During a communication round, all the worker nodes send their local information to the server, and the server node broadcasts processed information to all worker nodes. This type of distributed optimization framework is very popular in machine learning, such as federated learning [69], where the data are held privately in each worker (device) and the central server learns a global model by communicating with the workers. Since communication in a network tends to be slower than computation inside a single node by orders of magnitude, and less communication implies better protection of privacy, one of the main goals of this paper is to study the system's communication complexity, i.e., the number of communication rounds required to find a quality solution $\bar{x} \in X$ s.t. $f(\bar{x}) - f(x^*) \leq \epsilon$, where x^* denotes an optimal solution of (Equation 4.1).

Risk-averse optimization problems of form (Equation 4.1) have a wide range of applications in portfolio selection [70], renewable energy [71], power security [72], telecommunication [73] and climate change planning [74]. As a concrete example, consider the massive multiple-input multiple-output (MIMO) system in the 5G communication network consisting of multiple active antennas and terminal devices [73, 75]. The multiple active antennas at the base station should be configured to ensure stable connections for all the

terminal devices in its service area, rather than a high connection speed when averaged over all devices. Such an objective can be formulated as (Equation 4.1) with f_i being the negative data speed at the ith terminal device and (P, ρ^*) being the conjugate to the mean-semideviation risk measure. To gather information for the downlink and uplink channels, the base station needs to communicate with terminal devices. So in a highly mobile environment, finding a quality antenna connection quickly, i.e., with a only few rounds of communication, is crucial. A second example is motivated by climate change. The state government may wish to invest in infrastructure to prepare for it. Each scenario cost function f_i may denote the long-term economic cost estimated by a certain climate model and a certain impact model [74]. To avoid downside risk, (P, ρ^*) could be chosen as the conjugate to some risk measures mentioned above, say the entropic risk measure. Because these models involve large amounts of data and costly simulation runs, we might need to store f_i's on separate computing nodes and use a communication network to find the optimal policy. In this case, a small number of communication rounds is crucial for efficiency.

Our formulation is also applicable to the computationally demanding distributionally robust optimization (DRO). DRO provides a powerful framework for learning from limited data [76] and data-driven decision-making [77, 78]. Under the assumption of finite scenario support $\Xi = [\xi_1, \ldots, \xi_m]$, we could use $f_i(x) := f(x, \xi_i)$ to denote the cost under scenario ξ_i and choose ρ to be the risk measure induced by the corresponding probability uncertainty set [9]. When implemented on a distributed communication network with the evaluation of $f(x, \xi_i)$'s performed in parallel on different machines, a small number of communication rounds is essential for fast computation.

Additionally, the risk-averse formulation in (Equation 4.1) could also be useful for federated learning between organizations, i.e., the cross-silo federated learning [69]. Cross-silo federated learning has found applications in finance risk prediction in reinsurance [79], drug discovery [80], electronic health record mining [81] and smart manufacturing [82]. If the workers represent demographically partitioned organizations or geographically parti-

tioned data centers, we could choose ρ to be the mean-semideviation risk measure to ensure that the trained model offers consistent performance across different populations. Risk measures may also provide incentives for competing organizations to cooperate. For example, consider the operations of competing airlines. When $f_i(x)$ is the expected relative operation cost of i^{th} airline, choosing $\rho(\boldsymbol{f}(x)) := \max_{i\in[m]} f_i(x)$ ensures the new policy x benefits every participant. In both cases, a smaller number of communication rounds implies better protection of privacy.

Despite the importance of problem (Equation 4.1), however, the study of its communication complexity and the development of efficient algorithms are rather limited. Since (Equation 4.1) can be viewed as a trilinear saddle point problem, we can potentially apply some recently developed first-order algorithms (e.g. [55, 83]) for solving it. However, these methods are designed without special consideration for communication burden. The most related algorithm is perhaps the sequential dual (SD) method, which was first proposed in [55] for the structured non-smooth problem and later extended in [83] to the smooth problem. The method is single-loop, so a direct implementation on a communication network requires one communication round in each iteration, leading to communication complexities of $\mathcal{O}(\sqrt{L_f}D_{X^0}/\sqrt{\epsilon} + D_P D_\Pi M_A D_{X^0}/\epsilon)$ and $\mathcal{O}(D_\Pi D_{X^0}/\epsilon + D_P D_\Pi M_A D_{X^0}/\epsilon)$ for the smooth and the structured non-smooth problems, respectively. Here L_f, D_Π, M_A, D_P, and D_{X^0} correspond to the overall smoothness constant, the dual radius, the operator norm of A_i, the radius of P, and the distance to the optimal solution (see Tables 1.1 and 1.2, and Section 3 for their precise definitions). On the other hand, for the risk-neutral problem (Equation 4.2) with $P := \{\bar{p}\}$, direct distributed implementations of the Nesterov accelerated gradient method [33] and the primal-dual algorithm [84] can achieve communication complexities of $\mathcal{O}(\sqrt{L_f}D_{X^0}/\sqrt{\epsilon})$ and $\mathcal{O}(D_\Pi M_A D_{X^0}/\epsilon)$ for the smooth and the structured non-smooth problems, respectively, which were shown to be tight (see, e.g., [85]). Clearly, there exists a significant gap in communication complexities, especially for smooth problems where the $\mathcal{O}(1/\epsilon)$ communication complexity for the risk-averse setting

is much larger than the $\mathcal{O}(1/\sqrt{\epsilon})$ complexity for the risk-neutral setting. Therefore we pose the following research question:

Can we solve the risk-averse problem over a star-shape network with the same communication complexity as the finite-sum problem?

This paper intends to provide a positive answer to this question in three steps.

First, we propose a conceptual distributed risk-averse optimization (DRAO) method. It is inspired by works of Nesterov (Section 2.3.1 of [1]) and Lan [27] on composite optimization of the form $\min_x \rho(\boldsymbol{f}(x))$ for a smooth vector function $\boldsymbol{f}$. While Nesterov [1] considers the problem with $\rho(\boldsymbol{f}(x)) = \max_{i=1,\ldots,m} f_i(x)$, Lan [27] generalizes ρ to any monotone convex function. They can achieve an $\mathcal{O}(1/\sqrt{\epsilon})$ first-order (FO) oracle complexity of $\boldsymbol{f}$ by incorporating the following inner-linearization prox-mapping into the accelerated gradient descent (AGD) method or into the accelerated prox-level (APL) method:

$$x^t \leftarrow \arg\min_{x\in X} \rho\left(f_1(\underline{x}^t) + \langle \nabla f_1(\underline{x}^t), x - \underline{x}^t\rangle, \ldots, f_m(\underline{x}^t) + \langle \nabla f_m(\underline{x}^t), x - \underline{x}^t\rangle\right) + \tfrac{\eta}{2}\left\|x - x^{t-1}\right\|^2. \tag{4.4}$$

Such an update is a simplified version of (Equation 4.1) with $f_i(\underline{x}^t) + \langle \nabla f_i(\underline{x}^t), x - \underline{x}^t\rangle$ denoting some (iterative) linearization of f_i at $\underline{x}^t$ and $\frac{\eta}{2}\|x - x^{t-1}\|^2$ being the proximal term. Similarly, we modify the SD method by combining the p and x-prox updates into a single (x, p)-prox update given by

$$x^t \leftarrow \arg\min_{x\in X} \max_{p\in P} \textstyle\sum_{i=1}^m p_i[\langle x, A_i\pi_i^t\rangle - f_i^*(\pi_i^t)] - \rho^*(p) + u(x) + \frac{\eta}{2}\|x - x^{t-1}\|^2, \tag{4.5}$$

where $\langle x, A_i\pi_i^t\rangle - f_i^*(\pi_i^t)$ also represents some (iterative) linearization of f_i specified by the

dual variable π_i^t. In fact, rewriting ρ in its primal form (Equation 4.3) shows (Equation 4.5) to be equivalent to

$$x^t \leftarrow \underset{x\in X}{\arg\min}\rho\left(\langle x, A_1\pi_1^t\rangle - f_1^*(\pi_1^t), \ldots, \langle x, A_m\pi_m^t\rangle - f_m^*(\pi_m^t)\right) + u(x) + \frac{\eta}{2}\left\|x - x^{t-1}\right\|^2,$$

which matches (Equation 4.4) if π_i^t is selected to be $\nabla f_i(\underline{x}^t)$ for smooth f_i's. Such a modification of the SD method leads to the DRAO method. As shown in Table Table 4.1, it achieves the optimal FO oracle complexities for $\boldsymbol{f}$ (or Π_i-projection complexities) for both the smooth and the structured non-smooth problems. Since (ρ^*, P) is available to the server, (Equation 4.5) can be performed entirely on the server, so the communication complexities are the same (shown in Table Table 4.1). However, this approach requires ρ to be simple so that (Equation 4.5) can be efficiently solved. This assumption might be too strong in practice. For example, if m is large, the (x, p)-prox update in (Equation 4.5) with either the mean-semideviation risk measure ρ or the Kantorovich ambiguity set P is known to be computationally challenging.

Table 4.1: Communication Complexity and FO Oracle Complexity of $\boldsymbol{f}$ for DRAO and DRAO-S[②]

	Convex ($\alpha = 0$)	strongly convex ($\alpha > 0$)
Smooth	$\mathcal{O}(\sqrt{L_f}\|x^0 - x^*\|/\sqrt{\epsilon})$	$\mathcal{O}(\sqrt{L_f/\alpha}\log(1/\epsilon))$
Structured Non-smooth	$\mathcal{O}(M_A D_\Pi \left\|x^0 - x^*\right\|/\epsilon)$	$\mathcal{O}(M_A D_\Pi/\sqrt{\epsilon\alpha})$

Table 4.2: P-projection and X-projection Complexity of DRAO-S [③]

	convex ($\alpha = 0$)	strongly convex ($\alpha > 0$)
Smooth	$\mathcal{O}(D_P\tilde{M}\|x^0 - x^*\|/\epsilon)$	$\mathcal{O}((L_f/\alpha)^{1/4}\tilde{M}D_P/\alpha\sqrt{\epsilon})$[④]
Structured Non-smooth	$\mathcal{O}(D_P\bar{M}_{A\Pi}\|x^0 - x^*\|/\epsilon)$	$\mathcal{O}(D_P\bar{M}_{A\Pi}/\sqrt{\epsilon\alpha})$

[②] $M_A = \max_{i\in[m]}\|A_i\|$, $D_\Pi = \max_{i\in[m]}\max_{\pi_i,\bar{\pi}_i\in\Pi_i}\|\pi_i - \bar{\pi}_i\|$.

[③] D_P denotes P's radius. $\tilde{M}$ denotes the operator norm of $\|\nabla f_1(x), \ldots, \nabla f_m(x)\|$ over some bounded ball around x^* and $\bar{M}_{A\Pi}$ denotes the operator norm of $\|A_1\pi_1, \ldots, A_m\pi_m\|$ over the whole feasible region Π.

[④] Number of P-projections required to generate an ϵ-close solution, i.e., $\left\|x^N - x^*\right\|^2 \le \epsilon$.

Second, we overcome the restrictive assumption of ρ being simple by developing a saddle point sliding (SPS) subroutine. It replaces (Equation 4.5) in the DRAO method by performing only a finite number of P-projections and X-projections to solve the saddle point subproblem inexactly. The new method, called distributed risk-averse optimization with sliding (DRAO-S), maintains the same communication complexities as DRAO while improving on its computation efficiency. Since each inner iteration of the sliding subroutine requires one P-projection and one X-projection, the total numbers of these projections are optimal in most cases [5]. As shown in Table Table 4.2, they match the lower bounds [86] for solving a single (x, p) bi-linear saddle point problem, i.e., (Equation 4.5) with a fixed π^t and $\eta_t = 0$. Such a result is similar to that of the gradient sliding (GS) method [87] for solving an additive composite problem,

$$\min_{x \in X} f(x) + g(x). \tag{4.6}$$

The GS method can achieve both optimal f-oracle and optimal g-oracle complexities. However, our nested composite problem appears to be more challenging. This is because for a fixed x, the optimal dual variables p and π in (Equation 4.1) are dependent, while the optimal dual variables π_f and π_g (associated with the saddle point reformulation of (Equation 4.6) through bi-conjugation [10]) are independent. In fact, (Equation 4.6) can always be rewritten as a nested composite problem (see the discussion in Example 3 of [27]). Additionally, the SPS subroutine in the DRAO-S method is initialized differently from the usual sliding subroutines in [87] and [88]. Such a modification simplifies both the outer loop algorithm and the convergence analysis. This simplification could motivate the application of the sliding technique to a wider range of problems. Furthermore, an interesting feature of the DRAO-S method is that its inner loop, the SPS subroutine, can adjust dynamically to the varying levels of difficulty, characterized by $\|\pi^t\|$, of the saddle point subproblem (Equation 4.5). This allows us to remove the assumption of the smooth f_i's

[5] Except for the strongly convex smooth problem which is worse off by a factor of $(L_f/\alpha)^{1/4}$.

being Lipschitz continuous, which is required by the SD method in [83], but may not hold if the domain X is unbounded.

Third, we show that the communication complexities of both DRAO and DRAO-S are not improvable by constructing lower complexity bounds. Previous developments are restricted to a trivial P and the smooth problem [89]. We propose a more general computation model which includes both the f_i-gradient oracle and the f_i^*-prox mapping oracle, and introduce a different set of problem parameters appropriate for the risk-averse problem. They allow us to develop, for a non-trivial P and for both the smooth and the structured non-smooth problems, new lower complexity bounds matching the upper communication complexity bounds possessed by DRAO and DRAO-S.

The rest of the paper is organized as follows. Preliminary Section 2 reviews a gap function in [55] which will guide the algorithm design. Section 3 and Section 4 propose and analyze the DRAO and DRAO-S methods, respectively. Section 5 provides lower communication complexity bounds and Section 6 provides some encouraging numerical results. Finally, some concluding remarks are made in Section 7.

4.1.1 Notation & Assumptions

The following assumptions and notations will be used throughout the paper.

- The set of optimal solutions to (Equation 4.1), X^*, is nonempty. x^* denotes an arbitrary optimal solution, and f_* denotes the optimal objective, $f(x^*)$. R_0 represents an estimate of the distance from the initial point to x^*, i.e., $R_0 \geq \|x^0 - x^*\|$.

- D_P denotes the radius of P, i.e., $D_P := \max_{p,\bar{p}\in P} \sqrt{2U(p,\bar{p})}$ where U is the chosen Bregman distance function [28].

- $\boldsymbol{f} : \mathbb{R}^n \to \mathbb{R}^m$ denotes a vector of scenario cost functions, $[f_1; ...; f_m]$, and $\nabla \boldsymbol{f}(x) : \mathbb{R}^n \to \mathbb{R}^{m\times n}$ denotes the Jacobian matrix function.

- We refer to the following computation as either a prox mapping or a projection:

$$\hat{w} \leftarrow \arg\min_{w \in W} \langle g, w \rangle + h(w) + \tau V(w; \bar{w}), \tag{4.7}$$

where the vector g represents some "descent direction" (the gradient for example), and $h(w)$ denotes a simple convex function [28]. V denotes the Bregman distance function, $\bar{w}$ is a prox center, and τ is a stepsize parameter. Together they ensure the output $\hat{w}$ is close to $\bar{w}$. In particular, we call it an x, a π_i or a p-prox mapping (an X, a Π_i or a P-projection) if $W = X$ and $h \equiv 0$, $W = \Pi_i$ and $h = f_i^*$, or $W = P$ and $h = \rho^*$, respectively. Sometimes, the term prox update also is used to emphasize that the prox mapping is performed to update $w^t = \hat{w}$ from $\bar{w} = w^{t-1}$.

4.2 Preliminary: Q-gap function

We introduce a gap function [55] which will guide our algorithmic development throughout the paper. For notation convenience, we denote $\pi \equiv (\pi_1, \ldots, \pi_m)$ and $\Pi \equiv \Pi_1 \times \Pi_2 \times \ldots \times \Pi_m$ so that (Equation 4.1) can be written as

$$\min_{x \in X} \max_{p \in P} \max_{\pi \in \Pi} \{\mathcal{L}(x; p, \pi) := \textstyle\sum_{i=1}^m p_i \left(\langle A_i x, \pi_i \rangle - f_i^*(\pi_i)\right) - \rho^*(p) + u(x)\}. \tag{4.8}$$

The following duality relation between the reformulation and the original problem (Equation 4.1) is valid (see Proposition 2.1 of [55]).

Lemma 4.1 *Let f and $\mathcal{L}$ be defined in* (Equation 4.1) *and* (Equation 5.1)*, then the following statements hold for all $x \in X$.*

a) Weak Duality: $f(x) \geq \mathcal{L}(x, p, \pi)$ for all $p \in P, \pi \in \Pi$.

b) Strong Duality: $f(x) = \mathcal{L}(x, \hat{p}, \hat{\pi})$ for any $\hat{\pi}_i \in \arg\max_{\pi_i \in \Pi_i} \langle \pi_i, A_i x \rangle - f_i^*(\pi_i)$, $i = 1, \ldots, m$, and any $\hat{p} \in \arg\max_{p \in P} \sum_{i=1}^m p_i f_i(x) - \rho^*(p)$.

We measure the quality of a feasible solution $z = (x, p, \pi)$ by a gap function Q associated with some feasible reference point $\hat{z} := (\hat{x}; \hat{p}, \hat{\pi})$:

$$Q(z; \hat{z}) := \mathcal{L}(x; \hat{p}, \hat{\pi}) - \mathcal{L}(\hat{x}; p, \pi). \tag{4.9}$$

The Q function provides a bound on the function optimality gap from above.

Lemma 4.2 *Let Q be defined in* (Equation 4.9)*, then*

$$f(x) - f(x^*) \leq \max_{\hat{p} \in P, \hat{\pi} \in \Pi} Q((x; p, \pi); (x^*; \hat{p}, \hat{\pi})). \tag{4.10}$$

Moreover, the optimal solution x^* of (Equation 4.1)*, together with some $\pi_i^* \in \arg\max_{\pi_i \in \Pi_i} \langle \pi_i, A_i x^* \rangle - f_i^*(\pi_i)$, $i = 1, \ldots, m$, and $p^* \in \arg\max_{p \in P} \sum_{i=1}^m p_i f_i(x^*) - \rho^*(p)$ forms a saddle point $z^* := (x^*; p^*, \pi^*)$ of* (Equation 5.1)*, i.e.,*

$$Q(z; z^*) \geq 0, \ \forall z \equiv (x; p, \pi) \in X \times P \times \Pi. \tag{4.11}$$

Proof: Let $\hat{p}$ and $\hat{\pi}$ by defined in Lemma 4.1.b). By Lemma 4.1, we have $f(x) - f(x^*) \leq \mathcal{L}(x, \hat{p}, \hat{\pi}) - \mathcal{L}(x^*, p, \pi) = Q((x; p, \pi); (x^*; \hat{p}, \hat{\pi}))$, from which (Equation 4.10) follows immediately. Next, the first-order optimality condition of (Equation 4.1) implies that there exist some $\pi_i^* \in \arg\max_{\pi_i \in \Pi_i} \langle \pi_i, A_i x^* \rangle - f_i^*(\pi_i)$, $g^* \in \partial u(x^*)$ and some $p^* \in \arg\max_{p \in P} \sum_{i=1}^m p_i f_i(x^*) - \rho^*(p)$ such that $\langle \sum_{i=1}^m p_i^* A_i^\top \pi_i^*, x - x^* \rangle + u(x) - u(x^*) \geq \langle \sum_{i=1}^m p_i^* A_i^\top \pi_i^* + g^*, x - x^* \rangle \geq 0$ for any $x \in X$. This observation together with the definition of $\mathcal{L}$ in (Equation 5.1) then imply that

$$\mathcal{L}(x; p^*, \pi^*) \geq \mathcal{L}(x^*; p^*, \pi^*), \forall x \in X.$$

Moreover, due to our choice of (p^*, π^*), Lemma 4.1 also implies that

$$f(x^*) = \mathcal{L}(x^*; p^*, \pi^*) \geq \mathcal{L}(x^*; p, \pi), \forall (p, \pi) \in P \times \Pi.$$

(Equation 4.11) then follows from combining the preceding two inequalities. □

In view of Lemma 4.2, we can use Q to guide our search for an ϵ-optimal solution. In particular, we decompose Q into three sub-gap functions given by

$$Q(\bar{z}; \hat{z}) = Q_x(\bar{z}; \hat{z}) + Q_p(\bar{z}; \hat{z}) + Q_\pi(\bar{z}; \hat{z})$$

with

$$\begin{aligned}
Q_\pi(\bar{z}; \hat{z}) &:= \mathcal{L}(\bar{x}; \hat{p}, \hat{\pi}) - \mathcal{L}(\bar{x}; \hat{p}, \bar{\pi}) = \textstyle\sum_{i=1}^m \hat{p}_i \left[\langle A_i \bar{x}, \hat{\pi}_i - \bar{\pi}_i \rangle - f_i^*(\hat{\pi}_i) + f_i^*(\bar{\pi}_i) \right]. \\
Q_p(\bar{z}; \hat{z}) &:= \mathcal{L}(\bar{x}; \hat{p}, \bar{\pi}) - \mathcal{L}(\bar{x}; \bar{p}, \bar{\pi}) = \textstyle\sum_{i=1}^m (\hat{p}_i - \bar{p}_i)[\langle A_i \bar{x}, \bar{\pi}_i \rangle - f_i^*(\bar{\pi}_i)] \\
&\quad - (\rho^*(\hat{p}) - \rho^*(\bar{p})). \\
Q_x(\bar{z}; \hat{z}) &:= \mathcal{L}(\bar{x}; \bar{p}, \bar{\pi}) - \mathcal{L}(\hat{x}; \bar{p}, \bar{\pi}) = \langle \textstyle\sum_{i=1}^m \bar{p}_i A_i^\top \bar{\pi}_i, \bar{x} - \hat{x} \rangle + u(\bar{x}) - u(\hat{x}).
\end{aligned} \tag{4.12}$$

4.3 Upper Bounds for Communication Complexity

We propose the distributed risk-averse optimization (DRAO) method to provide upper bounds on communication complexity. The algorithm and its convergence properties are presented in Subsection subsection 4.3.1 and the convergence analysis is presented in Subsection subsection 4.3.2.

4.3.1 The DRAO method

The DRAO method is designed for solving the min-max-max trilinear saddle point problem in (Equation 5.1). It is inspired by two algorithms for optimizing nested composite prob-

lems. First, the sequential dual (SD) algorithm, proposed in [55, 83], performs sequential proximal updates to the dual variables π and p before updating the primal variable x. The DRAO method is built on similar sequential proximal updates for π, p, and x. Second, the accelerated prox-level (APL) algorithm, proposed in [27], can reduce the number of outer iterations further by solving a more complicated proximal sub-problem (Equation 4.4). The DRAO method exploits this property by combining the separate p and x proximal updates into a single (x, p) prox update step in the server node to save communication.

Algorithm Algorithm 9 describes a generic DRAO method which will be later specialized for solving the smooth and the structured nonsmooth problems. As shown in Algorithm Algorithm 9, the server first sends an extrapolated point $\tilde{x}^t$ to the workers for them to perform dual proximal updates in Line 3. The only goal is to reduce the sub-gap function Q_{π_i} (c.f. (Equation 4.12)). Here we intentionally leave the prox-function V_i in an abstract form because its selection and the resulting implementation will depend on the smoothness properties of f_i. Next, the server collects the newly generated $A_i\pi_i^t$ in Line 4 to solve the (x, p) prox update problem in Line 5 to reduce both Q_x and Q_p.

Algorithm 9 A Generic **D**istributed **R**isk **A**verse **O**ptimization (DRAO) Method

Input: $x^0 = x^{-1} \in X$, $\pi_i^0 \in \Pi_i$ in every node, stepsizes $\{\theta_t\}$, $\{\eta_t\}$, $\{\tau_t\}$, and weights $\{\omega_t\}$.

1: **for** $t = 1, 2, 3...N$ **do**

2: Server computes $\tilde{x}^t \leftarrow x^{t-1} + \theta_t(x^{t-1} - x^{t-2})$. Broadcast it to all workers.

3: Every worker computes $\pi_i^t \leftarrow \arg\max_{\pi_i \in \Pi_i} \langle A_i \tilde{x}^t, \pi_i \rangle - f_i^*(\pi_i) - \tau_t V_i(\pi_i; \pi_i^{t-1})$, and evaluates $v_i^t \leftarrow A_i^\top \pi_i^t$ and $f_i^*(\pi_i^t)$.

4: All workers send their $(v_i^t, f_i^*(\pi_i^t))$ to the server.

5: Server updates

$x^t \leftarrow \arg\min_{x \in X} \max_{p \in P} \sum_{i=1}^m p_i(\langle x, v_i^t \rangle - f_i^*(\pi_i^t)) - \rho^*(p) + u(x) + \frac{\eta_t}{2} \|x - x^{t-1}\|^2$.

6: **end for**

7: **return** $\bar{x}^N := \sum_{t=1}^N \omega_t x^t / (\sum_{t=1}^N \omega_t)$.

In the generic DRAO algorithm, we assume subproblems in Lines 3 and 5 to be solved exactly by the workers and server, respectively. Line 3 reduces to local gradient evaluations in the smooth case, while requiring a prox mapping for the structured nonsmooth case. Line 5 requires us to solve a structured bilinear saddle point problem. We will discuss in detail how to solve these problems approximately in the next section while focusing on the communication complexity now.

First, we consider the smooth problem where all A_i's are identity matrices and all f_i's are smooth such that $\|\nabla f_i(x_1) - \nabla f_i(x_2)\| \leq L_i \|x_1 - x_2\|, \forall x_1, x_2 \in \mathbb{R}^n$. Since the Fenchel conjugate to a smooth convex function is strongly convex [90], a natural choice of the prox-function V_i in DRAO would be the Bregman distance function generated by f_i^* given by

$$W_{f_i^*}(\pi_i; \bar{\pi}_i) := f_i^*(\pi_i) - f_i^*(\bar{\pi}_i) - \langle (f_i^*)'(\bar{\pi}_i), \pi_i - \bar{\pi}_i \rangle. \tag{4.13}$$

It has been shown in [56, 28, 83] that the π_i proximal update in Line 3 of Algorithm Algorithm 9 is equivalent to a gradient evaluation. Specifically, with $\underline{x}^0 = x^0$ and $\pi_i^0 = \nabla f_i(\underline{x}^0)$, Line 3 reduces to the following steps:

$$\underline{x}^t \leftarrow (\tilde{x}^t + \tau_t \underline{x}^{t-1})/(1+\tau_t), \tag{4.14}$$

$$\pi_i^t \leftarrow \nabla f_i(\underline{x}^t), \tag{4.15}$$

$$f_i^*(\pi_i^t) \leftarrow \langle \underline{x}^t, \pi_i^t \rangle - f_i(\underline{x}^t). \tag{4.16}$$

Plugging $f_i^*(\pi_i^t)$ defined in (Equation 4.16) into Line 5 of Algorithm Algorithm 9, we can completely remove the information about the conjugate function f_i^*. Therefore, DRAO is a purely primal algorithm for the smooth problem.

To discuss the convergence properties of DRAO, we need to properly define some Lipschitz smoothness constants. For a given $p \in P$, let us denote $f_p(x) := \sum_{i=1}^m p_i f_i(x)$. Clearly, f_p is a smooth convex function with Lipschitz continuous gradients, i.e., $\|\nabla f_p(x_1) - \nabla f_p(x_2)\| \leq L_p \|x_1 - x_2\|, \forall x_1, x_2 \in X$. Moreover, $L_p \leq \sum_{i=1}^m p_i L_i$. We define an aggregate smoothness constant L_f to characterize the overall smoothness property of the risk-averse problem (Equation 5.1):

$$L_f = \max_{p \in P} L_p. \tag{4.17}$$

Observe that in the risk neutral case with $P = \{(1/m, \ldots, 1/m)\}$, L_f is the global smoothness constant of f [85], which is upper bounded by $\frac{1}{m}\sum_{i=1}^m L_i$. In the robust case when $P = \Delta_m^+$, $L_f = \max_i L_i$. Theorem Theorem 4.1 and Theorem 4.2 below show the convergence rates of the DRAO method applied to the aforementioned smooth problems, for a non-strongly convex $u(x)$ and a strongly convex $u(x)$ respectively. Their proofs are given in Section subsection 4.3.2.

Theorem 4.1 *Let L_f be defined in* (Equation 4.17)*. If $\{x^t\}_{t=1}^N$ are generated by the DRAO method applied to a smooth problem with*

$$\omega_t = t,\ \theta_t = (t-1)/t, \tau_t = (t-1)/2, \eta_t = 2L_f/t.$$

Then for a reference point $\hat{z} := (\hat{x}; \hat{p}, \hat{\pi})$ in which $\hat{\pi}_i = \nabla f_i(\bar{x})$ for some $\bar{x} \in X$, we have

$$\textstyle\sum_{t=1}^N \omega_t Q(z^t; \hat{z}) + L_f \left\|x^N - \hat{x}\right\|^2 \leq L_f \left\|x^0 - \hat{x}\right\|^2. \tag{4.18}$$

In particular, the ergodic solution $\bar{x}^N$ satisfies

$$f(\bar{x}^N) - f(x^*) \leq 2L_f \left\|x^0 - x^*\right\|^2 / N(N+1), \forall N \geq 1. \tag{4.19}$$

Theorem 4.2 *Let L_f be defined in* (Equation 4.17)*. Assume, in addition, that $u(x)$ is α-strongly convex for some $\alpha > 0$. Let $\kappa := L_f/\alpha$ denote the condition number. If $\{x^t\}_{t=1}^N$ are generated by the DRAO method applied to smooth problems with*

$$\theta_t = \theta := \tfrac{\sqrt{4\kappa+1}-1}{\sqrt{4\kappa+1}+1},\ \omega_t = (\tfrac{1}{\theta})^{t-1},\ \tau_t = \tau := \tfrac{\sqrt{4\kappa+1}-1}{2},\ \eta_t = \eta := \tfrac{\alpha(\sqrt{4\kappa+1}-1)}{2}. \tag{4.20}$$

Then for a reference point $\hat{z} := (\hat{x}; \hat{p}, \hat{\pi})$ in which $\hat{\pi}_i = \nabla f_i(\bar{x})$ for some $\bar{x} \in X$, we have

$$\textstyle\sum_{t=1}^N \omega_t Q(z^t; \hat{z}) + \ \tfrac{\alpha(\sqrt{4\kappa+1}-1)}{4\theta^N} \left\|x^N - \hat{x}\right\|^2 \leq \tfrac{(\sqrt{4\kappa+1}-1)}{4}(\alpha \left\|x^0 - \hat{x}\right\|^2 + L_f \left\|x^0 - \bar{x}\right\|^2). \tag{4.21}$$

In particular, the last iterate x^N converges geometrically:

$$\left\|x^N - x^*\right\|^2 \leq \theta^N(1+\kappa) \left\|x^0 - x^*\right\|^2, \forall N \geq 1. \tag{4.22}$$

We make two remarks regarding the above convergence results. First, selecting the saddle point z^* defined in Lemma 4.2 as $\hat{z}$, Theorem Theorem 4.1 (c.f. (Equation 4.18))

and Theorem 4.2 (c.f. (Equation 4.21)) imply that all generated iterates, $\{x^t\}_{t\geq 1}$, are inside some ball around x^*:

$$\begin{aligned} \left\|x^t - x^*\right\| &\leq \left\|x^0 - x^*\right\| && \text{if } \alpha = 0, \\ \left\|x^t - x^*\right\| &\leq (1 + L_f/\alpha) \left\|x^0 - x^*\right\| && \text{if } \alpha > 0. \end{aligned}$$

This shows that the search space for x^t is essentially bounded. Such a property will become useful when we solve the saddle point subproblem in Line 5 of DRAO approximately in the next section. Second, Theorem Theorem 4.1 and Theorem 4.2 imply, respectively, $\mathcal{O}(\sqrt{L_f}\left\|x^0 - x^*\right\|/\sqrt{\epsilon})$ and $\mathcal{O}(\sqrt{L_f/\alpha}\log(1/\epsilon))$ communication complexities to find ϵ-optimal solutions. It is interesting to note that with L_f defined in (Equation 4.17), these results are valid even if P is larger than the probability simplex, i.e., $\Delta_+^m \subsetneq P \subset \mathbb{R}_+^m$, which could be useful if the risk measure ρ is not positive homogeneous. We will show later in Section 5 that these communication complexity bounds are not improvable in general.

Next, let us consider the structured non-smooth problem. Because f_i^* may not be strongly convex, the Bregman distance function $W_{f_i^*}$ (c.f. (Equation 4.13)) is no longer suitable for π_i prox update. Instead, we choose $V_i(\pi_i; \bar{\pi}_i) := \frac{1}{2}\left\|\pi_i - \bar{\pi}_i\right\|^2$, so that the π_i proximal update is given by:

$$\pi_i^t \leftarrow \arg\max_{\pi_i \in \Pi_i} \langle A_i \tilde{x}^t, \pi_i \rangle - f_i^*(\pi_i) - \tfrac{\tau_t}{2}\left\|\pi_i - \pi_i^{t-1}\right\|^2. \tag{4.23}$$

Theorem Theorem 4.3 below states the convergence properties of Algorithm Algorithm 9 applied to the structured nonsmooth problem and its proof is provided in Section subsection 4.3.2. We need to define the maximum linear operator norm M_A and the maximum dual radius D_Π as

$$M_A := \max_{i\in[m]} \left\|A_i\right\|_{2,2}, \; D_\Pi := \max_{i\in[m]} \max_{\pi_i, \bar{\pi}_i \in \Pi_i} \left\|\pi_i - \bar{\pi}_i\right\|. \tag{4.24}$$

Note that $M_A D_\Pi$ provides an estimate of the Lipschitz continuity constant of $\sum_i p_i f_i(x)$.

Theorem 4.3 *Let a structured non-smooth risk-averse problem* (Equation 4.1) *be given. Let* M_A *and* D_Π *be defined above in* (Equation 4.24) *and let* $R_0 \geq \|x^0 - x^*\|$.
a) If $\alpha = 0$ *and the stepsizes satisfy*

$$\omega_t = 1,\ \theta_t = 1, \eta_t = M_A D_\Pi / R_0, \tau_t = M_A R_0 / D_\Pi,$$

the following convergence rate holds for the solution $\bar{x}^N$ *returned by the DRAO algorithm*

$$f(\bar{x}^N) - f(x^*) \leq M_A D_\Pi R_0 / N. \tag{4.25}$$

b) If $\alpha > 0$ *and the stepsizes satisfy*

$$\omega_t = t,\ \theta_t = (t-1)/t, \eta_t = t\alpha/3, \tau_t = 3M_A^2/t\alpha,$$

the following convergence rate holds for the solution $\bar{x}^N$ *returned by the DRAO algorithm*

$$f(\bar{x}^N) - f(x^*) \leq \left(\alpha \left\|x^0 - x^*\right\|^2 / 3 + 3M_A^2 D_\Pi^2 / \alpha\right) / N^2. \tag{4.26}$$

The preceding theorem gives us $\mathcal{O}(R_0 D_\Pi M_A / \epsilon)$ and $\mathcal{O}(M_A D_\Pi / \sqrt{\epsilon\alpha})$[6] communication complexities for solving the structured nonsmooth problem under the non-strongly convex and the strongly convex settings, respectively. These complexity bounds are worse than those of the smooth problem by an order of magnitude. It is interesting to note that the smoothness properties of the scenario cost functions have a significant impact on communication complexity, even under the assumption that the workers are equipped with the capability to solve the π_i proximal update in (Equation 4.23).

[6] We assume the strong convexity modulus α to be small such that the $M_A^2 D_\Pi^2 / \alpha$ term dominates in (Equation 4.26).

4.3.2 Convergence analysis

Our main goal in this subsection is to establish the convergence rates associated with the DRAO method stated in Theorems Theorem 4.1, Theorem 4.2, and Theorem 4.3.

We will first show some general convergence properties about the generic DRAO method in Algorithm Algorithm 9. Since this result holds regardless of the strong convexity of f_i^* (i.e., $\mu = 0$ is allowed in (Equation 4.27)) and the strong convexity of u (i.e., $\alpha = 0$ is allowed), it will be applied to both smooth and nonsmooth problems under either convex or strongly convex settings.

Proposition 4.1 *Let $\{z^t \equiv (x^t; p^t, \pi^t)\}_{t=1}^N$ be generated by Algorithm Algorithm 9 for some $p^t \in \arg\max_{p\in P} \sum_{i=1}^m p_i(\langle x^t, \pi_i^t\rangle - f_i^*(\pi_i^t)) - \rho^*(p)$. Fix a reference point $\hat{z} := (\hat{x}; \hat{p}, \hat{\pi}) \in X \times P \times \Pi$ (c.f.* (Equation 4.9)*). Assume μ is a non-negative constant satisfying*

$$f_i^*(\pi_i) - f_i^*(\bar{\pi}_i) - \langle g_i'(\bar{\pi}_i), \pi_i - \bar{\pi}_i\rangle \geq \mu V_i(\pi_i; \bar{\pi}_i),\ \forall \pi_i, \bar{\pi}_i \in \Pi_i, \forall i \in [m]. \tag{4.27}$$

If there exists a positive constant q satisfying

$$\begin{aligned} &\textstyle\sum_{i=1}^m p_i V_i(\pi_i^t; \pi_i^{t-1}) \geq \frac{1}{2q} \left\|\sum_{i=1}^m p_i A_i^\top(\pi_i^t - \pi_i^{t-1})\right\|^2, \forall t \geq 2, \forall p \in P, \\ &\textstyle\sum_{i=1}^m p_i V_i(\hat{\pi}_i; \pi_i^N) \geq \frac{1}{2q} \left\|\sum_{i=1}^m p_i A_i^\top(\hat{\pi}_i - \pi_i^N)\right\|^2, \forall p \in P, \end{aligned} \tag{4.28}$$

and the stepsizes satisfy the following conditions for all $t \geq 2$:

$$\begin{aligned} &\omega_{t-1} = \omega_t \theta_t, \\ &\eta_{t-1}\tau_t \geq \theta_t q,\ (\tau_N + \mu)\eta_N \geq q, \\ &\omega_t \eta_t \leq \omega_{t-1}(\eta_{t-1} + \alpha),\ \omega_t \tau_t \leq \omega_{t-1}(\tau_{t-1} + \mu), \end{aligned} \tag{4.29}$$

then the next bound is valid for all $\hat{z} := (\hat{x}; \hat{p}, \hat{\pi}) \in X \times P \times \Pi$ *and* $N \geq 1$*:*

$$\sum_{t=1}^{N} \omega_t Q(z^t; \hat{z}) + \frac{\omega_N(\eta_N+\alpha)}{2} \left\|x^N - \hat{x}\right\|^2 \leq \frac{\omega_1\eta_1}{2} \left\|x^0 - \hat{x}\right\|^2 + \omega_1\tau_1 \sum_{i=1}^{m} \hat{p}_i V_i(\hat{\pi}_i; \pi_i^0). \tag{4.30}$$

Proof: Let Q_π, Q_p and Q_x be defined in (Equation 4.12). We begin by analyzing the convergence of Q_π. It follows from the definition of $\tilde{x}^t$ that

$$\begin{aligned}\langle \hat{\pi}_i - \pi_i^t, A_i(\tilde{x}^t - x^t)\rangle = &- \langle \hat{\pi}_i - \pi_i^t, A_i(x^t - x^{t-1})\rangle + \theta_t \langle \hat{\pi}_i - \pi_i^{t-1}, A_i(x^{t-1} - x^{t-2})\rangle \\ &+ \theta_t \langle \pi_i^{t-1} - \pi_i^t, A_i(x^{t-1} - x^{t-2})\rangle.\end{aligned}$$

The optimality condition for the dual update in Line 3 of Algorithm Algorithm 9 (see Lemma 3.1 of [28]) implies

$$\begin{aligned}&\langle \hat{\pi}_i - \pi_i^t, A_i x^t\rangle + f_i^*(\pi_i^t) - f_i^*(\hat{\pi}_i) + \langle \hat{\pi}_i - \pi_i^t, A_i(\tilde{x}^t - x^t)\rangle \\ &\qquad \leq \tau_t V_i(\hat{\pi}_i; \pi_i^{t-1}) - (\tau_t + \mu) V_i(\hat{\pi}_i; \pi_i^t) - \tau_t V_i(\pi_i^t; \pi_i^{t-1}).\end{aligned}$$

So, combining the above two relations, taking the ω_t weighted sum of the resulting inequalities and using the conditions that $\omega_{t-1} = \omega_t\theta_t$ and $\omega_t\tau_t \leq \omega_{t-1}(\tau_{t-1} + \mu)$, we obtain

$$\begin{aligned}&\sum_{t=1}^{N} \omega_t(\langle \hat{\pi}_i - \pi_i^t, A_i x^t\rangle + f_i^*(\pi_i^t) - f_i^*(\hat{\pi}_i)) \\ &\quad \leq - (\omega_N(\tau_N + \mu) V_i(\hat{\pi}_i; \pi_i^N) - \omega_N \langle \hat{\pi}_i - \pi_i^N, A_i(x^N - x^{N-1})\rangle) \\ &\qquad - \sum_{t=2}^{N} [\omega_t\tau_t V_i(\pi_i^t; \pi_i^{t-1}) + \omega_{t-1}\langle \pi_i^{t-1} - \pi_i^t, A_i(x^{t-1} - x^{t-2})\rangle] \\ &\qquad + \omega_1\tau_1 V_i(\hat{\pi}_i; \pi_i^0).\end{aligned}$$

A $\hat{p}_i$-weighted sum of the above inequality leads to the desired Q_π convergence bound given by

$$\begin{aligned}
&\textstyle\sum_{t=1}^{N} \omega_t Q_\pi(z^t; \hat{z}) \\
&\quad \leq -(\omega_N(\tau_N + \mu) \textstyle\sum_{i=1}^{m} \hat{p}_i V_i(\hat{\pi}_i; \pi_i^N) - \omega_N \langle \textstyle\sum_{i=1}^{m} \hat{p}_i A_i^\top (\hat{\pi}_i - \pi_i^N), x^N - x^{N-1} \rangle) \\
&\qquad - \textstyle\sum_{t=2}^{N} [\omega_t \tau_t \textstyle\sum_{i=1}^{m} \hat{p}_i V_i(\pi_i^t; \pi_i^{t-1}) + \omega_{t-1} \langle \textstyle\sum_{i=1}^{m} \hat{p}_i A_i^\top (\pi_i^{t-1} - \pi_i^t), x^{t-1} - x^{t-2} \rangle] \\
&\qquad + \omega_1 \tau_1 \textstyle\sum_{i=1}^{m} \hat{p}_i V_i(\hat{\pi}_i; \pi_i^0).
\end{aligned} \tag{4.31}$$

Next, we consider x^t and p^t generated by Line 5 in Algorithm Algorithm 9. Let $F(x; \pi^t) := \max_{p \in P} \sum_{i=1}^{m} p_i [\langle x, v_i^t \rangle - f_i^*(\pi_i^t)] - \rho^*(p) + u(x) + \frac{\eta}{2} \left\| x - x^{t-1} \right\|^2$. Since $x^t \in \arg\min_{x \in X} F(x; \pi^t)$, the first-order necessity condition implies the existence of some maximizer p^t and some subgradient $u'(x^t) \in \partial u(x^t)$ such that

$$\begin{aligned}
&\textstyle\sum_{i=1}^{m} p_i^t v_i^t + \eta(x^t - x^{t-1}) + u'(x^t) \in -\partial \delta_X(x^t) \\
&\qquad\qquad \Rightarrow \langle \textstyle\sum_{i=1}^{m} p_i^t A_i^\top \pi_i^t, x^t - \hat{x} \rangle + \langle \eta_t(x^t - x^{t-1}) + u'(x^t), x^t - \hat{x} \rangle \leq 0.
\end{aligned}$$

Since α-strong convexity of u implies that $u(x^t) + \alpha \|x^t - \hat{x}\|^2/2 - u(\hat{x}) \leq \langle u'(x^t), x^t - \hat{x} \rangle$, and $\frac{\eta_t}{2} \left\| x^t - \hat{x} \right\|^2 + \frac{\eta_t}{2} \left\| x^t - x^{t-1} \right\|^2 - \frac{\eta_t}{2} \left\| x^{t-1} - \hat{x} \right\|^2 = \langle \eta_t(x^t - x^{t-1}), x^t - \hat{x} \rangle$, we get

$$Q_x(z^t; \hat{z}) + \tfrac{\eta_t + \alpha}{2} \left\| x^t - \hat{x} \right\|^2 \leq \tfrac{\eta_t}{2} \left\| x^{t-1} - \hat{x} \right\|^2 - \tfrac{\eta_t}{2} \left\| x^t - x^{t-1} \right\|^2. \tag{4.32}$$

Additionally, being a maximizer, p^t satisfies

$$\begin{aligned}
&p^t \in \arg\max_{p \in P} \textstyle\sum_{i=1}^{m} p_i (\langle x^t, A_i \pi_i^t \rangle - f_i^*(\pi_i^t)) - \rho^*(p) \\
&\quad \Rightarrow \textstyle\sum_{i=1}^{m} (\hat{p}_i - p^t)(\langle x^t, A_i \pi_i^t \rangle - f_i^*(\pi_i^t)) + \rho^*(p^t) - \rho^*(\hat{p}) \leq 0 \Rightarrow Q_p(z^t; \hat{z}) \leq 0,
\end{aligned} \tag{4.33}$$

So, combining (Equation 4.33) and (Equation 4.32), taking a ω_t-weighted sum of the resulting inequality and using $\omega_t\eta_t \leq \omega_{t-1}(\eta_{t-1}+\alpha)$, we obtain

$$\begin{aligned}\textstyle\sum_{t=1}^{N} \omega_t(Q_x(z^t;\hat{z}) + Q_p(z^t;\hat{z})) + \frac{\omega_N(\eta_N+\alpha)}{2}\left\|x^N-\hat{x}\right\|^2 \\ \leq \textstyle\frac{\omega_1\eta_1}{2}\left\|x^0-\hat{x}\right\|^2 - \sum_{t=1}^{N}\frac{\omega_t\eta_t}{2}\left\|x^t-x^{t-1}\right\|^2.\end{aligned} \tag{4.34}$$

Then utilizing (Equation 4.28) and the Young's inequality, (Equation 4.30) follows immediately by adding (Equation 4.34) to (Equation 4.31). □

We now apply the result in Proposition 4.1 to the smooth problem. Observe that the gradient evaluation point $\underline{x}^t$ is common for all workers. This allows us to easily characterize the strong convexity modulus of the aggregate prox-penalty function $\sum_{i=1}^m \hat{p}_i W_{f_i^*}(\cdot;\cdot)$ (c.f. (Equation 4.28)) in the next lemma.

Lemma 4.3 *Let $\hat{p} \in P$ be given. If $\pi_i = \nabla f_i(x)$ and $\bar{\pi}_i = \nabla f_i(\bar{x})$ for some x and $\bar{x}$, then*

$$\textstyle\sum_{i=1}^m \hat{p}_i W_{f_i^*}(\pi_i;\bar{\pi}_i) \geq \frac{1}{2L_f}\left\|\sum_{i=1}^m \hat{p}_i A_i^\top(\pi_i-\bar{\pi}_i)\right\|^2. \tag{4.35}$$

Proof: Let $f_{\hat{p}}(x) := (\sum_{i=1}^m \hat{p}_i f_i(x))$. Then by the definition of L_f in (Equation 4.17), $f_{\hat{p}}$ is L_f-smooth and its conjugate $(f_{\hat{p}})^*$ is $1/L_f$ strongly convex. Next, we relate $W_{(f_{\hat{p}})^*}$ to $\sum_{i=1}^m \hat{p}_i W_{f_i^*}$ to calculate its strong convexity modulus. Since $\sum_{i=1}^m \hat{p}_i\pi_i = \nabla f_{\hat{p}}(x)$ and $\sum_{i=1}^m \hat{p}_i\bar{\pi}_i = \nabla f_{\hat{p}}(\bar{x})$, we have by Fenchel duality that

$$\begin{aligned}\textstyle\sum_{i=1}^m \hat{p}_i\langle \pi_i-\bar{\pi}_i, \nabla f_i^*(\bar{\pi}_i)\rangle &= \textstyle\langle\sum_{i=1}^m \hat{p}_i(\pi_i-\bar{\pi}_i), \bar{x}\rangle \\ &= \textstyle\langle\sum_{i=1}^m \hat{p}_i(\pi_i-\bar{\pi}_i), \nabla(f_{\hat{p}})^*(\sum_{i=1}^m \hat{p}_i\bar{\pi}_i)\rangle, \\ \textstyle\sum_{i=1}^m \hat{p}_i f_i^*(\pi_i) &= \textstyle\sum_{i=1}^m \hat{p}_i(\langle\pi_i,x\rangle - f_i(x)) \\ &= \textstyle\langle\sum_{i=1}^m \hat{p}_i\pi_i, x\rangle - (\sum_{i=1}^m \hat{p}_i f_i)(x) = (f_{\hat{p}})^*(\sum_{i=1}^m \hat{p}_i\pi_i),\end{aligned}$$

and, similarly, $\sum_{i=1}^m \hat{p}_i f_i^*(\bar{\pi}_i) = (f_{\hat{p}})^*(\sum_{i=1}^m \hat{p}_i \bar{\pi}_i)$. Thus

$$\begin{aligned}
&\textstyle\sum_{i=1}^m \hat{p}_i W_{f_i^*}(\pi_i; \bar{\pi}_i) \\
&\quad = (f_{\hat{p}})^*(\textstyle\sum_{i=1}^m \hat{p}_i \pi_i) - (f_{\hat{p}})^*(\sum_{i=1}^m \hat{p}_i \bar{\pi}_i) - \langle \sum_{i=1}^m \hat{p}_i(\pi_i - \bar{\pi}_i), \nabla (f_{\hat{p}})^*(\sum_{i=1}^m \hat{p}_i \bar{\pi}_i)\rangle \\
&\quad = W_{(f_{\hat{p}})^*}(\textstyle\sum_{i=1}^m \hat{p}_i \pi_i; \sum_{i=1}^m \hat{p}_i \bar{\pi}_i) \geq \frac{1}{2L_f} \left\|\sum_{i=1}^m \hat{p}_i(\pi_i - \bar{\pi}_i)\right\|^2 \\
&\quad = \frac{1}{2L_f} \left\|\sum_{i=1}^m \hat{p}_i A_i^\top (\pi_i - \bar{\pi}_i)\right\|^2,
\end{aligned}$$

where the last inequality follows from $A_i^\top = I$ in smooth problems. □

We are now ready to prove Theorems Theorem 4.1 and Theorem 4.2.

Proof of Theorem Theorem 4.1: We apply Proposition 4.1 to obtain the convergence result in (Equation 4.19). Since f_i^* is 1-strongly convex with respect to $W_{f_i^*}$, $\mu = 1$ satisfies condition (Equation 4.27). Since $\pi_i^t = \nabla f_i(\underline{x}^t)$ and $\hat{\pi}_i = \nabla f_i(\bar{x})$ for some $\bar{x}$, $q = L_f$ satisfies condition (Equation 4.28) (c.f. Lemma 4.3). Moreover, since stepsizes proposed in Theorem Theorem 4.1 verifies (Equation 4.29), all the requirements in Proposition 4.1 are met. Thus Proposition 4.1 leads to (Equation 4.18), i.e.,

$$\textstyle\sum_{t=1}^N \omega_t \mathcal{L}(x^t; \hat{p}, \hat{\pi}) - \sum_{t=1}^N \omega_t \mathcal{L}(x^*; p^t, \pi^t) + L_f \left\|x^N - x^*\right\|^2 \leq L_f \left\|x^0 - x^*\right\|^2.$$

In particular, with $\hat{\pi}_i^N = \nabla f_i(\bar{x}^N)$ and $\hat{p}^N \in \arg\max_{p \in P} \sum_{i=1}^m p_i f_i(\bar{x}^N)$ such that $f(\bar{x}^N) = \mathcal{L}(\bar{x}^N; \hat{p}^N, \hat{\pi}^N)$ (see Lemma 4.1), we have

$$\textstyle\sum_{t=1}^N \omega_t \mathcal{L}(x^t; \hat{p}^N, \hat{\pi}^N) - \sum_{t=1}^N \omega_t \mathcal{L}(x^*; p^t, \pi^t) \leq L_f \left\|x^N - x^*\right\|^2.$$

Because $\mathcal{L}(\cdot; \hat{p}^N, \hat{\pi}^N)$ is convex with respect to x, the first term satisfies

$$\textstyle\sum_{t=1}^N \omega_t \mathcal{L}(x^t; \hat{p}^N, \hat{\pi}^N) \geq \frac{N(N+1)}{2} \mathcal{L}(\bar{x}^N; \hat{p}^N, \hat{\pi}^N) = \frac{N(N+1)}{2} f(\bar{x}^N).$$

Due to the weak duality in Lemma 4.1, the second term is upper bounded by

$$\sum_{t=1}^{N} \omega_t \mathcal{L}(x^*; p^t, \pi^t) \leq \sum_{t=1}^{N} \omega_t f(x^*) = \tfrac{N(N+1)}{2} f(x^*).$$

Then the desired inequality in (Equation 4.19) follows immediately.

□

Proof of Theorem Theorem 4.2: Similar to the preceding proof, the proposed stepsizes (c.f. (Equation 4.20)), together with $\mu = 1$ and $q = L_f$, verify the requirements in Proposition 4.1, thus

$$\begin{aligned}\sum_{t=1}^{N} \omega_t(\mathcal{L}(x^t; \hat{p}, \hat{\pi}) - \mathcal{L}(x^*; p^t, \pi^t)) + \omega_N(\eta + \alpha) \left\|x^N - x^*\right\|^2 \\ \leq \omega_1(\eta \left\|x^0 - x^*\right\|^2 + \tau \textstyle\sum_{i=1}^{m} \hat{p}_i W_{f_i^*}(\hat{\pi}_i; \pi_i^0)).\end{aligned}$$

Using the relation of conjugate Bregman distance functions and the identity $\nabla(\sum_{i=1}^{m} \hat{p}_i f_i)(\bar{x}) = \sum_{i=1}^{m} \hat{p}_i \nabla f_i(\bar{x})$, the last term can be upper bounded by

$$\begin{aligned}&\textstyle\sum_{i=1}^{m} \hat{p}_i W_{f_i^*}(\hat{\pi}_i; \pi_i^0) = \sum_{i=1}^{m} \hat{p}_i (W_{f_i}(x^0; \bar{x})) \\ &\quad = (\textstyle\sum_{i=1}^{m} \hat{p}_i f_i)(x^0) - (\sum_{i=1}^{m} \hat{p}_i f_i)(\bar{x}) - \langle \nabla(\sum_{i=1}^{m} \hat{p}_i f_i)(\bar{x}), x^0 - \bar{x} \rangle \\ &\quad \leq \tfrac{L_f}{2} \left\|x^0 - \bar{x}\right\|^2.\end{aligned}$$

Thus the Q convergence bound in (Equation 4.21) follows immediately from

$$\textstyle\sum_{t=1}^{N} \omega_t Q(z^t; \hat{z}) + \frac{\alpha(\sqrt{4\kappa+1}-1)}{4\theta^N} \left\|x^N - \hat{x}\right\|^2 \leq \frac{(\sqrt{4\kappa+1}-1)}{4}(\alpha \left\|x^0 - \hat{x}\right\|^2 + L_f \left\|x^0 - \bar{x}\right\|^2).$$

Additionally, setting the preceding $\hat{z}$ to the saddle point z^* defined in Lemma (4.2) (c.f. (Equation 4.11)) such that $Q(z^t; z^*) \geq 0 \; \forall t$ and dividing both sides by $\frac{\alpha(\sqrt{4\kappa+1}-1)}{4\theta^N}$, the geometric convergence of x^N to x^* in (Equation 4.22) can be deduced. □Now we move on to present convergence proofs for structured non-smooth problems.

Proof of Theorem Theorem 4.3 First, we consider part a) with a non-strongly convex $u(x)$. The result is also a consequence of Proposition 4.1. Since $V_i(\pi_i; \bar{\pi}_i) := \frac{1}{2}\|\pi_i - \bar{\pi}_i\|^2 \geq \frac{1}{2M_A^2}\left\|A_i^\top(\pi_i - \bar{\pi}_i)\right\|^2$, the Jensen's inequality implies the condition (Equation 4.28) is satisfied with $q = M_A^2$. Since f_i^* is convex, the condition (Equation 4.27) is satisfied with $\mu = 0$. Additionally, since the chosen stepsizes in Theorem Theorem 4.3 satisfy the condition (Equation 4.29), all the requirements for Proposition 4.1 are met. Thus for any feasible $\hat{z} := (\hat{x}; \hat{p}, \hat{\pi})$, we have

$$\textstyle\sum_{t=1}^N \omega_t Q(z^t; \hat{z}) + \frac{\omega_N \eta_N}{2}\left\|x^N - \hat{x}\right\|^2 \leq \frac{\omega_1\eta_1}{2}\|x^0 - \hat{x}\|^2 + \frac{\omega_1\tau_1}{2}\sum_{i=1}^m \hat{p}_i\|\hat{\pi}_i - \pi_i^0\|^2.$$

Let $\hat{\pi}_i^N \in \arg\max_{\pi_i \in \Pi_i}\langle \pi_i, A_i\bar{x}^N\rangle - f_i^*(\pi_i)$ and $\hat{p}^N \in \arg\max_{p\in P}\sum_{i=1}^m p_i f_i(\bar{x}^N)$ such that $f(\bar{x}^N) = \mathcal{L}(\bar{x}^N; \hat{p}_i^N, \hat{\pi}_i^N)$ (c.f. Lemma 4.1). Setting $\hat{z}$ to $\hat{z}^N := (x^*; \hat{p}^N, \hat{\pi}^N)$ leads to

$$\textstyle\sum_{t=1}^N \omega_t Q(z^t; \hat{z}^N) \leq \frac{\eta_1}{2}R_0^2 + \frac{\tau_1}{2}D_\Pi^2 = R_0 M_A D_\Pi.$$

Then the resulting convergence bound in (Equation 4.25) can be deduced from the fact $(\sum_{t=1}^N \omega_t)f(\bar{x}^N) - f(x^*) \leq \sum_{t=1}^N \omega_t Q(z^t; \hat{z}^N)$.

As for part b), the derivation is the same except for the different stepsize choice to take advantage of the α-strong convexity of $u(x)$. □

4.4 The DRAO-S method

The practical application of the DRAO method is limited by the exact computation to the following saddle point problem in Line 5 of Algorithm Algorithm 9:

$$x^t \leftarrow \arg\min_{x\in X}\max_{p\in P}\textstyle\sum_{i=1}^m p_i[\langle x, v_i^t\rangle - f_i^*(\pi_i^t)] - \rho^*(p) + u(x) + \frac{\eta_t}{2}\|x - x^{t-1}\|^2. \quad (4.36)$$

We relax it by assuming only the ability to efficiently compute the p-prox mapping defined in (Equation 5.6). In the DRO setting, the efficient implementations for risk measures

induced by several probability uncertainty sets are described in [55]. For the entropic risk measure, if we select prox-function to be $U(p;\bar{p}) := \sum_{i=1}^{m} p_i \log(p_i/\bar{p}_i)$, the computation amounts to a softmax evaluation. For the mean semi-deviation risk of order two, the computation can be implemented as a quadratically constrained quadratic program (QCQP).

In this section, we design a novel saddle point sliding (SPS) subroutine to solve (Equation 4.36) inexactly in the DRAO method and call the resulting method *distributed risk-averse optimization with sliding (DRAO-S)*. We show the DRAO-S method maintains the same order of communication complexity as the DRAO method. Moreover, the total number of P-projections required by the DRAO-S method will be mostly optimal, in the sense that it is equivalent to the optimal one required for solving problem (Equation 4.1) with linear local cost functions f_i's.

4.4.1 The Algorithm and Convergence Results

The SPS subroutine for solving (Equation 4.36) inexactly is presented in Algorithm Algorithm 10. It is closely related to the classic primal dual (PD) algorithm (see [84, 28]) for solving a structured bilinear saddle point problem given by

$$\min_{y \in X} \max_{p \in P} \langle v^t y, p \rangle - \rho^*(p),$$

where the matrix v^t is obtained from stacking $(v_1^t)^\top, (v_2^t)^\top, \ldots, (v_m^t)^\top$ from Line 3 of Algorithm Algorithm 9 vertically. In iteration s, the subroutine computes an extrapolated prediction of $\sum_{i=1}^{m} p_i^s v_i^t$ in Line 2, a y-prox update in Line 3, and then a p-prox update in Line 4. The p-prox update utilizes a general Bregman distance function U to allow a suitable choice to take advantage of the geometry of P. At the end of S_t iterations, the SPS subroutine returns a weighted ergodic average of $\{y^s\}$ as an approximate solution to (Equation 4.36).

Algorithm 10 Saddle **P**oint **S**liding (SPS) Subroutine

Input: Initial points $x^{t-1}, y^0 \in X$, $p^0, p^{-1} \in P$, and gradients $\{v_i^t\}, \{v_i^{t-1}\}$. Non-negative stepsizes η_t, $\{\delta_s\}$, $\{\gamma_s\}$ and $\{\beta_s\}$, averaging weights $\{q_s\}$, and iteration number S_t.

1: **for** $s = 1, 2, 3...S_t$ **do**

2: $\tilde{v}^s \leftarrow \begin{cases} \sum_{i=1}^m p_i^0 v_i^t + \delta_1 \sum_{i=1}^m (p_i^0 - p_i^{-1}) v_i^{t-1} & \text{if } s = 1, \\ \sum_{i=1}^m p_i^{s-1} v_i^t + \delta_s \sum_{i=1}^m (p_i^{s-1} - p_i^{s-2}) v_i^t & \text{if } s \geq 2. \end{cases}$

3: $y^s \leftarrow \arg\min_{y \in X} \langle y, \tilde{v}^s \rangle + u(y) + \frac{\beta_s}{2} \|y - y^{s-1}\|^2 + \frac{\eta_t}{2} \|y - x^{t-1}\|^2$.

4: $p^s \leftarrow \arg\max_{p \in P} \sum_{i=1}^m p_i(\langle v_i^t, y^s \rangle - f_i^*(\pi_i^t)) - \rho^*(p) - \gamma_s U(p; p^{s-1})$.

5: **end for**

6: **return** $x^t := \sum_{s=1}^{S_t} q_s y^s / (\sum_{s=1}^{S_t} q_s)$, $y^t := y^{S_t}$, $\bar{p}^t := \sum_{s=1}^{S_t} q_s p^s / (\sum_{s=1}^{S_t} q_s)$, $\tilde{p}^t := p^{S_t}$ and $\tilde{\tilde{p}}^t := p^{S_t - 1}$.

The DRAO-S method is obtained by making the following two modifications to DRAO. First, we require additional initial points $(y^0, \tilde{p}^0, \tilde{\tilde{p}}^0) \in X \times P \times P$. For simplicity, we set $y^0 = x^0$ and $\tilde{p}^0 = \tilde{\tilde{p}}^0$. Second, we replace the definition of x^t in Line 5 of DRAO with the output solution of the SPS subroutine according to:

$$(x^t, y^t, \bar{p}^t, \tilde{p}^t, \tilde{\tilde{p}}^t) = SPS(x^{t-1}, y^{t-1}, \tilde{p}^{t-1}, \tilde{\tilde{p}}^{t-1}, \{v_i^t\}, \{v_i^{t-1}\} \mid \eta_t, \{\delta_s^t\}, \{\gamma_s^t\}, \{\beta_s^t\}, \{q_s^t\}, S_t) \; \forall t \geq 1. \tag{4.37}$$

At the beginning, i.e., $t = 1$, we set $v_i^0 = v_i^1 \; \forall i \in [m]$. Due to its similarity to the DRAO method, we call the outer loop of the DRAO-S method (Algorithm Algorithm 9 with modification (Equation 4.37)) the outer DRAO loop and say a phase of the DRAO-S method happens if t is increased by 1. Accordingly, we call the inner loop of the DRAO-S method (Algorithm Algorithm 10) the inner SPS loop and say an (inner) iteration happens if s is incremented by 1. Intuitively, if the inner iteration limits S_t were large enough, x^t obtained from the inner SPS loop would be a good approximate solution of (Equation 4.36). How-

ever, the P-projection complexity, i.e., the total number of inner iterations given by $\sum_t S_t$, might be too large. In addition, notice that the computation burdens of the server subproblem and the worker subproblem during each communication round are still different: the server requires several rounds of P and X-projections for the SPS subroutine, while the worker needs just one π_i-prox mapping. However, since the server is often more powerful, e.g. the cloud-edge system, this asymmetry may have a limited impact on the overall computation performance of the system.

We note here that the DRAO-S method is related to the Primal Dual Sliding (PDS) method in [88], where the sliding subroutine is also a primal dual type algorithm. However, since we are dealing with a nested trilinear saddle point problem, rather than the sum of two bilinear saddle point problems, the DRAO-S method differs from the PDS method in two important ways. First, the linear operator v^t for the saddle point subproblem (c.f. (Equation 4.36)) changes in every phase. To coordinate consecutive inner SPS loops, we construct a special momentum term from both the current v^t and the previous v^{t-1} when transitioning into a new phase, i.e., Line 2 of Algorithm Algorithm 10. This construction is inspired by the novel momentum term in the SD method [55]. Second, as opposed to the single initialization in both the PDS method and the gradient sliding method [87], the y prox update in Line 3 of Algorithm Algorithm 10 utilizes two distinct initialization points, the ergodic average x^{t-1} and the last iterate obtained in the last phase, y^0. Even though both are approximate solutions to (Equation 4.36), y^s is used only in the inner SPS loop while the ergodic average x^t is used in both the outer loop and the inner loop. As will be discussed in the next subsection, the additional initialization point appears to significantly simplify the convergence analysis and the selection of stepsizes in comparison to [87, 88].

Now, let us consider the smooth problem. The stepsizes associated with the inner SPS loop need to adapt dynamically in two aspects. First, the inner iteration limit S_t needs to be an increasing function of t to maintain the same communication complexity as the DRAO method. Second, as a primal dual type algorithm, the inner SPS loop stepsizes,

γ_s^t, δ_s^t and β_s^t, need to satisfy a certain condition related to the operator norm of v^t, i.e., $\gamma_{s-1}^t \beta_s^t \geq \delta_s^t \|v^t\|^2$, to ensure convergence. Specifically, if the Bregman distance function U in Line 4 of Algorithm Algorithm 10 is 1-strongly convex with respect to $\|\cdot\|_U$, the operator norm of interest is given by

$$M_t := \left\|v^t\right\|_{2,U^*} := \max_{\|p\|_U \leq 1, \|y\| \leq 1} \sum_{i=1}^m p_i (v_i^t)^\top y. \tag{4.38}$$

Since a uniform bound on M_t may not exist if X is unbounded under the smooth setting, we choose γ_s^t and β_s^t to adjust dynamically to M_t in each phase. In particular, the next theorem presents the stepsize choice and the convergence result under the non-strongly convex setting.

Theorem 4.4 *Let a smooth risk-averse problem (c.f.* (Equation 4.1)*) be given. Let M_t and L_f be defined in* (Equation 4.38) *and* (Equation 4.17)*, and let R_0 and D_P be defined in Subsection subsection 4.1.1. If $\{x^t\}_{t=1}^N$ are generated by the DRAO-S method* (Equation 4.37) *with the following stepsizes:*

$$\begin{aligned}
&\omega_t = t,\ \theta_t = (t-1)/(t), \tau_t = (t-1)/2, \eta_t = 2L_f/t, \\
&\Delta > 0,\ S_t = \lceil t\Delta M_t \rceil,\ \bar{M}_t = \tfrac{S_t}{t\Delta},\ \beta_s^t = \beta^t = \tfrac{D_P \bar{M}_t}{R_0},\ \gamma_s^t = \gamma^t = \tfrac{R_0 \bar{M}_t}{D_P}, \\
&q_s^t = 1,\ \delta_1^t = \begin{cases} \bar{M}_t / \bar{M}_{t-1} & \text{if } t \geq 2 \\ 1 & \text{if } t = 1 \end{cases}, \text{ and } \delta_s^t = 1\ \forall\ s \geq 2,
\end{aligned} \tag{4.39}$$

then the solution $\bar{x}^N$ returned by the outer DRAO loop satisfies

$$f(\bar{x}^N) - f(x^*) \leq \tfrac{2L_f R_0^2}{N(N+1)} + \tfrac{2D_P R_0}{N(N+1)\Delta}, \forall N \geq 1, \tag{4.40}$$

and there exists an uniform upper bound $\tilde{M}$ for M_t, i.e., $\tilde{M} \geq M_t,\ \forall t \geq 1$. In addition, if

$\Delta = D_P/(L_f R_0)$, *the convergence bound can be simplified to*

$$f(\bar{x}^N) - f(x^*) \leq \frac{4L_f R_0^2}{N(N+1)}, \forall N \geq 1. \tag{4.41}$$

A few remarks are in place regarding the above result. First, the stepsizes in the outer DRAO loop are exactly the same as that of Theorem Theorem 4.1. Since each phase requires only two rounds of communication, the DRAO-S method has a communication complexity of $\mathcal{O}(\sqrt{L_f}R_0/\sqrt{\epsilon})$. Second, $\bar{M}_t$ defined in (Equation 4.39) is the smallest upper bound of M_t needed to make the inner iteration limit S_t an integer. The factor Δ in (Equation 4.39) represents the conversion factor between the P-projection complexity and the communication complexity (Π projection complexity). A communication complexity of the order $\mathcal{O}(1/\sqrt{\epsilon})$ can be maintained for any $\Delta > 0$ and the specific choice in (Equation 4.39) is needed only for optimal constant dependence. Third, since the number of phases needed to find an ϵ-optimal solution is bounded by $N_\epsilon := 2\sqrt{L_f}R_0/\sqrt{\epsilon}$, the total number of P-projections is given by

$$\textstyle\sum_{t=1}^{N_\epsilon} \lceil t\Delta M_t \rceil \leq \Delta \tilde{M} N_\epsilon^2 + N_\epsilon = \mathcal{O}(D_P \tilde{M} R_0/\epsilon).$$

Fourth, both the inner SPS loop stepsizes, β_s^t, γ_s^t and δ_s^t, and the inner iteration limit S_t adjust dynamically to the varying operator norm M_t characterizing the difficulty of the saddle point problem (Equation 4.36) of each phase. Specifically, when the saddle point problem is easy, i.e., M_t is small, γ_s^t, β_s^t, and S_t become small so that a small number of inner iterations is performed, and vice versa. Thus, when most M_t's are significantly smaller than the upper bound $\tilde{M}$, the total number of P-projections can be much smaller than $\mathcal{O}(D_P \tilde{M} R_0/\epsilon)$. Such a saving is possible because S_t can compensate for the changing β_s^t and γ_s^t such that the the effective proximal penalty parameter, $\omega_t \gamma_s^t/S_t$ and $\omega_t \beta_s^t/S_t$, remains constant across phases. In contrast, it is difficult for single loop primal dual type algorithms, such as the SD method [83], to adjust dynamically to the varying operator norm

of v^t in each iteration.

The following theorem presents the stepsize choice and the convergence result under the strongly convex setting.

Theorem 4.5 *Let a smooth problem f (c.f.* (Equation 4.1)*) with $\alpha > 0$ be given. Let R_0 and D_P be defined in Subsection subsection 4.1.1. Let the smoothness constant L_f be defined in* (Equation 4.17) *such that $\kappa := L_f/\alpha$ denotes the condition number. If $\{x^t\}_{t=1}^N$ are generated by the DRAO-S method* (Equation 4.37) *with the following stepsize:*

$$\begin{aligned}
&\theta_t = \theta := \tfrac{\sqrt{8\kappa+1}-1}{\sqrt{8\kappa+1}+1},\ \omega_t = (\tfrac{1}{\theta})^{t-1},\ \tau_t = \tau := \tfrac{\sqrt{8\kappa+1}-1}{2},\ \eta_t = \eta := \tfrac{\alpha(\sqrt{8\kappa+1}-1)}{4},\\
&\beta_s^t = \tfrac{\alpha(s-1)}{4},\ \gamma_s^t = \tfrac{2S_t(S_t+1)}{\omega_t \alpha s \Delta},\ \delta_s^t = (s-1)/s,\\
&q_s^t = s,\ S_t = \lceil (2\omega_t\Delta)^{1/2} M_t \rceil,\ \Delta > 0,
\end{aligned} \tag{4.42}$$

the last iterate x^N converges geometrically:

$$\left\|x^N - x^*\right\|^2 \leq \theta^N \left((1+2\kappa) \left\|x^0 - x^*\right\|^2 + \tfrac{4D_P^2}{\alpha\eta\Delta} \right), \forall N \geq 1, \tag{4.43}$$

and there exist an $\tilde{M}$ such that $M_t \leq \tilde{M}\ \forall t \geq 1$. If $\Delta = 2D_P^2/\eta L_f R_0^2$ and $\kappa \geq 1$, the total number of P-projections required to find an ϵ-close solution, i.e., $\left\|x^N - x^*\right\| \leq \epsilon$, is bounded by $\mathcal{O}(\frac{\kappa^{1/4}\tilde{M}D_P}{\alpha\sqrt{\epsilon}} + \sqrt{\kappa}\log(\frac{1}{\epsilon}))$.

Note that the strong convexity modulus α is split into two to accelerate both the outer DRAO loop and the inner SPS loop. For the outer DRAO loop, the proposed stepsize is the same as that of Theorem Theorem 4.2 if $\alpha/2$ is viewed as the strong convexity modulus. This allows the DRAO-S method to maintain the same order of communication complexity, i.e., $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$. For the inner SPS loop, the stepsizes are similar to that of the accelerated primal-dual method (see [28]) with a strong convexity modulus of $\alpha/2$. Moreover, similar to Theorem Theorem 4.4, the stepsize γ_s^t and the inner iteration limit S_t adjust dynamically to the varying operator norm $\bar{M}_t$ in each phase. It is also worth noting

that the constant dependence of the P-projection complexity on κ in Theorem Theorem 4.5 is larger than the optimal by a factor of $\kappa^{1/4}$. This complexity can be further improved to $\mathcal{O}(\tilde{M}D_P/\alpha\sqrt{\epsilon} + \sqrt{\kappa}\log(1/\epsilon))$ if some stepsize choice similar to Theorem 6 of [60] is utilized.

Next, let us move on to the structured non-smooth problem. Since Π is assumed be bounded, the following uniform upper bound of M_t (c.f. (Equation 4.38)) is useful for convergence analysis,

$$\tilde{M}_{A\Pi} = \max_{\pi\in\Pi}\{\left\|[A_1^\top\pi_1^t;\ldots;A_m^\top\pi_m^t]\right\|_{2,U^*} := \max_{\pi\in\Pi}\max_{\|y\|_2\leq 1,\|p\|_U\leq 1}\textstyle\sum_{i=1}^m p_i\langle A_i^\top\pi_i, y\rangle\}. \quad (4.44)$$

Specifically, the stepsize choices and the convergence properties of the DRAO-S method, applied to both non-strongly and strongly convex settings, are presented in the next theorem.

Theorem 4.6 *Let a structured non-smooth problem* f (Equation 4.1) *be given. Suppose* D_Π, M_A, M_t, *and* $\tilde{M}_{A\Pi}$ *are defined in* (Equation 4.24), (Equation 4.38) *and* (Equation 4.44)*, and suppose* D_P *and* R_0 *are defined in Subsection subsection 4.1.1.*
a) If $\alpha = 0$ *and the stepsizes are given by*

$$\begin{aligned}
&\omega_t = 1,\ \theta_t = 1, \eta_t = M_A D_\Pi/2R_0, \tau_t = M_A R_0/2D_\Pi,\\
&\Delta > 0, S_t = \lceil M_t\Delta\rceil,\ \bar{M}_t := \tfrac{S_t}{\Delta},\ \beta_s^t = \beta = \tfrac{D_P\bar{M}_t}{R_0},\ \gamma_s^t = \gamma = \tfrac{R_0\bar{M}_t}{D_P},\\
&q_s^t = 1,\ \delta_1^t = \begin{cases} \bar{M}_t/\bar{M}_{t-1} & \text{if } t\geq 2\\ 1 & \text{if } t = 1\end{cases}, \text{ and } \delta_s^t = 1\ \forall\ s\geq 2,
\end{aligned} \quad (4.45)$$

the solution $\bar{x}^N$ *returned by the DRAO-S method satisfies*

$$f(\bar{x}^N) - f(x^*) \leq \tfrac{M_A D_\Pi R_0}{N} + \tfrac{D_P R_0}{\Delta N}, \forall N\geq 1. \quad (4.46)$$

In particular, if $\Delta = \frac{D_P}{M_A D_\Pi}$*, the convergence bound can be simplified to*

$$f(\bar{x}^N) - f(x^*) \leq \frac{2M_A D_\Pi R_0}{N} \ \forall N \geq 1. \tag{4.47}$$

b) If $\alpha > 0$ *and the stepsizes are given by*

$$\begin{aligned} &\omega_t = t,\ \theta_t = (t-1)/t,\ \eta_t = t\alpha/6,\ \tau_t = 6/t\alpha, \\ &\Delta > 0\ , S_t = \lceil \Delta \tilde{M}_{A\Pi}^2 \rceil,\ \gamma_s^t = \gamma^t := 4\tilde{M}_{A\Pi}^2/\alpha t, \\ &\beta_s^t = \begin{cases} \frac{\alpha}{4}(t-1) & \text{if } s = 1 \\ \frac{\alpha}{4} t & \forall\ s \geq 2 \end{cases},\ \delta_s^t = \begin{cases} \frac{t-1}{t} & \text{if } s = 1 \\ 1 & \forall\ s \geq 2 \end{cases},\ q_s^t = 1, \end{aligned} \tag{4.48}$$

the solution $\bar{x}^N$ *returned by the DRAO-S method satisfies*

$$f(\bar{x}^N) - f(x^*) \leq \frac{1}{N(N+1)}\left(\frac{\alpha}{6} R_0^2 + \frac{6M_A^2 D_\Pi^2}{\alpha} + \frac{4\tilde{M}_{A\Pi}^2 D_P^2}{\alpha\Delta}\right). \tag{4.49}$$

In particular, if $\Delta = D_P^2/(M_A^2 D_\Pi^2 + \alpha^2 R_0^2/6)$*, the convergence bound can be simplified to*

$$f(\bar{x}^N) - f(x^*) \leq \frac{1}{N(N+1)}\left(\alpha R_0^2 + \frac{10M_A^2 D_\Pi^2}{\alpha}\right). \tag{4.50}$$

A few remarks are in place. First, observe the above inner iteration limit S_t adjust dynamically to the operator norm M_t for the non-strongly convex case (Equation 4.45), but not for the strongly convex case (Equation 4.48). This shortcoming is an artifact of the order of prox updates in Algorithm Algorithm 10. If a p-prox update, utilizing a y-momentum prediction, is performed before the y-prox update, S_t can be chosen to be $\lceil \Delta M_t^2 \rceil$ to achieve the same effect. Moreover, since two communication rounds is required for each phase, the preceding result implies an $\mathcal{O}(1/\epsilon)$ ($\mathcal{O}(1/\sqrt{\epsilon})$) communication complexity when $\alpha = 0$ (resp. $\alpha > 0$). Since the inner iteration limit S_t is bounded, it also implies an $\mathcal{O}(1/\epsilon)$ (resp. $\mathcal{O}(1/\sqrt{\epsilon})$) P-projection complexity. In particular, with the specific choices of Δ shown

above, the DRAO-S method can achieve the optimal constant dependence on problem parameters, that is, $\mathcal{O}(R_0 M_A D_\Pi/\epsilon)$ communication and $\mathcal{O}(\tilde{M}_{A\Pi} D_P R_0/\epsilon)$ P-projection complexities when $\alpha = 0$, and $\mathcal{O}(M_A D_\Pi/\sqrt{\epsilon\alpha})$ communication and $\mathcal{O}(\tilde{M}_{A\Pi} D_P/\sqrt{\epsilon\alpha})$ P-projection complexities when $\alpha > 0$.

4.4.2 Convergence Analysis

Our goal in this subsection is to establish the convergence rates of the DRAO-S method stated in Theorem Theorem 4.4, Theorem 4.5 and Theorem 4.6.

First, we present a recursive bound to characterize the convergence property of each inner SPS loop under both the non-strongly convex ($\alpha = 0$) and the strongly convex ($\alpha > 0$) settings.

Proposition 4.2 *Fix a $t \geq 1$. Let $M_t := \|v^t\|_{2,U^*}$ and $M_{t-1} := \|v^{t-1}\|_{2,U^*}$. If the SPS stepsizes in Algorithm Algorithm 10 satisfy:*

$$\begin{aligned} &\delta_s = q_s/q_{s-1},\ \beta_s\gamma_{s-1} \geq \delta_s M_t^2, \forall s \geq 2, \\ &q_{s-1}(\beta_{s-1} + \alpha/2) \geq q_s\beta_s,\ q_{s-1}\gamma_{s-1} \geq q_s\gamma_s, \forall s \geq 2, \end{aligned} \tag{4.51}$$

the generated iterates, $\{(y^s, p^s)\}$ and $(x^t, \bar{p}^t)$, satisfy the following relation for all $x \in X$, $p \in P$ and $S \geq 1$:

$$\begin{aligned} &(\textstyle\sum_{s=1}^S q_s)[\mathcal{L}(x^t; p, \pi^t) - \mathcal{L}(x; \bar{p}^t, \pi^t)] + \delta_1 q_1 \langle y^0 - x, \textstyle\sum_{i=1}^m v_i^{t-1}(p_i^0 - p_i^{-1})\rangle \\ &\quad + \tfrac{(\sum_{s=1}^S q_s)}{2}[\eta_t \left\|x^t - x^{t-1}\right\|^2 + (\eta_t + \alpha/2)\left\|x^t - x\right\|^2 - \eta_t \left\|x^{t-1} - x\right\|^2] \\ &\quad - q_S\langle y^S - x, \textstyle\sum_{i=1}^m v_i^t(p_i^S - p_i^{S-1})\rangle \\ &\leq q_1\gamma_1 U(p; p^0) - q_S\gamma_S[U(p; p^S) + \tfrac{1}{2}\left\|p^S - p^{S-1}\right\|_U^2] + \tfrac{q_1\delta_1^2 M_{t-1}^2}{2\beta_1}\left\|p^0 - p^{-1}\right\|_U^2 \\ &\quad - \tfrac{1}{2}[q_S(\beta_S + \alpha/2)\left\|y^S - x\right\|^2 - q_1\beta_1\left\|y^0 - x\right\|^2]. \end{aligned} \tag{4.52}$$

Proof: Fix points $x \in X$ and $p \in P$. First, consider the convergence of y^s. Since $u(y) + \eta_t \left\| y - x^{t-1} \right\|^2 /2$ has a strong convexity modulus of $\alpha + \eta_t$, the y-proximal update in Line 3 of Algorithm Algorithm 10 leads a three-point inequality (see Lemma 3.1 of [28]):

$$\begin{aligned}\langle y^s - x, \tilde{v}^s \rangle + \tfrac{1}{2}[(\beta_s + \alpha + \eta_t) \left\| x - y^s \right\|^2 + \beta_s \left\| y^s - y^{s-1} \right\|^2 - \beta_s \left\| y^{s-1} - x \right\|^2] \\ + u(y^s) - u(x) + \tfrac{\eta_t}{2}(\left\| y^s - x^{t-1} \right\|^2 - \left\| x - x^{t-1} \right\|^2) \leq 0.\end{aligned}$$

Equivalently, we have

$$\begin{aligned}&\langle y^s - x, \tilde{v}^s \rangle + \tfrac{1}{2}[(\beta_s + \alpha/2) \left\| y - y^s \right\|^2 + \beta_s \left\| y^s - y^{s-1} \right\|^2 - \beta_s \left\| y^{s-1} - x \right\|^2] \\ &\quad + u(y^s) - u(x) + \tfrac{1}{2}[\eta_t \left\| y^s - x^{t-1} \right\|^2 + (\eta_t + \alpha/2) \left\| y^s - x \right\|^2 - \eta_t \left\| x - x^{t-1} \right\|^2] \leq 0.\end{aligned} \tag{4.53}$$

In particular, the definition of $\tilde{v}^s$ in Line 2 of Algorithm Algorithm 10 implies

$$\begin{aligned}&\langle y^s - x, \tilde{v}^s \rangle = \langle y^s - x, \textstyle\sum_{i=1}^m p^s v_i^t \rangle - \langle y^s - x, \textstyle\sum_{i=1}^m (p_i^s - p_i^{s-1}) v_i^t \rangle \\ &\quad + \delta_s \langle y^{s-1} - x, \textstyle\sum_{i=1}^m (p_i^{s-1} - p_i^{s-2}) v_i^t \rangle + \delta_s \langle y^s - y^{s-1}, \textstyle\sum_{i=1}^m (p_i^{s-1} - p_i^{s-2}) v_i^t \rangle, \forall s \geq 2, \\ &\langle y^1 - x, \tilde{v}^1 \rangle = \langle y^1 - x, \textstyle\sum_{i=1}^m p^1 v_i^t \rangle - \langle y^1 - x, \textstyle\sum_{i=1}^m (p_i^1 - p_i^0) v_i^t \rangle \\ &\quad + \delta_1 \langle y^0 - x, \textstyle\sum_{i=1}^m (p_i^0 - p_i^{-1}) v_i^{t-1} \rangle + \delta_1 \langle y^1 - y^0, \textstyle\sum_{i=1}^m (p_i^0 - p_i^{-1}) v_i^{t-1} \rangle.\end{aligned}$$

So, substituting them into (Equation 5.60), summing up the resulting inequality with weight q_s, noting the step-sizes conditions in (Equation 4.51), and utilizing Young's inequality, we

get

$$\begin{aligned}
&\textstyle\sum_{s=1}^{S_t} q_s \left(\mathcal{L}(y^s; p^s, \pi^t) - \mathcal{L}(x; p^s, \pi^t) + \frac{1}{2}[\eta_t \left\| y^s - x^{t-1} \right\|^2 + (\eta_t + \alpha/2) \left\| y^s - x \right\|^2 \right) \\
&\quad - \textstyle\sum_{s=1}^{S_t} q_s \eta_t \left\| x - x^{t-1} \right\|^2] + q_1 \delta_1 \langle y^0 - x, \sum_{i=1}^m (p_i^0 - p_i^{-1}) v_i^{t-1} \rangle \\
&\quad - q_S \langle y^S - x, \textstyle\sum_{i=1}^m (p_i^S - p_i^{S-1}) v_i^t \rangle \\
&\leq \textstyle\sum_{s=2}^{S} \frac{q_{s-1}\gamma_{s-1}}{2} \left\| p^{s-1} - p^{s-2} \right\|_U^2 + \frac{q_1 \delta_1^2 M_{t-1}^2}{2\beta_1} \left\| p^0 - p^{-1} \right\|_U^2 \\
&\quad - \tfrac{1}{2}[q_S(\beta_S + \alpha/2) \left\| y^S - x \right\|^2 - q_1 \beta_1 \left\| y^0 - x \right\|^2].
\end{aligned} \tag{4.54}$$

Next, consider the convergence of p^s. The p-proximal update in Line 4 of Algorithm Algorithm 10 implies

$$\mathcal{L}(y^s; p, \pi^t) - \mathcal{L}(y^s; p^s, \pi^t) + \gamma_s [U(p; p^s) + U(p^s; p^{s-1}) - U(p; p^{s-1})] \leq 0.$$

Observing the strong convexity of U with respect to $\|\cdot\|_U$ and the stepsize conditions in (Equation 4.51), the q_s weighted sum satisfies

$$\begin{aligned}
&\textstyle\sum_{s=1}^{S_t} q_s [\mathcal{L}(y^s; p, \pi^t) - \mathcal{L}(y^s; p^s, \pi^t)] + \gamma_S q_S U(p; p^S) + \sum_{s=1}^{S_t} \frac{q_s \gamma_s}{2} \left\| p^s - p^{s-1} \right\|_U^2 \\
&\leq q_1 \gamma_1 U(p; p^0).
\end{aligned}$$

Then, combining it with the y convergence bound in (Equation 5.61), we get

$$\begin{aligned}
&\textstyle\sum_{s=1}^{S_t} q_s \left(\mathcal{L}(y^s; p, \pi^t) - \mathcal{L}(x; p^s, \pi^t) + \frac{1}{2}[\eta_t \left\| y^s - x^{t-1} \right\|^2 + (\eta_t + \alpha/2) \left\| y^s - x \right\|^2 \right) \\
&\quad - \textstyle\sum_{s=1}^{S_t} q_s \eta_t \left\| x^{t-1} - x \right\|^2] + q_1 \delta_1 \langle y^0 - x, \sum_{i=1}^m v_i^{t-1} (p_i^0 - p_i^{-1}) \rangle \\
&\quad - q_S \langle y^S - x, \textstyle\sum_{i=1}^m v_i^t (p_i^S - p_i^{S-1}) \rangle \\
&\leq q_1 \gamma_1 U(p; p^0) - q_S \gamma_S [U(p; p^S) + \tfrac{1}{2} \left\| p^S - p^{S-1} \right\|_U^2] + \frac{q_1 \delta_1^2 M_{t-1}^2}{2\beta_1} \left\| p^0 - p^{-1} \right\|_U^2 \\
&\quad - \tfrac{1}{2}[q_S(\beta_S + \alpha/2) \left\| y^S - x \right\|^2 - q_1 \beta_1 \left\| y^0 - x \right\|^2].
\end{aligned}$$

Moreover, since $\mathcal{L}(y^s; p, \pi^t)$, $\|y^s - x^{t-1}\|^2$ and $\|y^s - x\|^2$ are convex with respect to y^s and $\mathcal{L}(x; p^s, \pi^t)$ is linear with respect to p^s, the desired convergence bound (Equation 4.52) can be derived using the Jensen's inequality. □

In the above proposition, an ω_t-weighted sum of the terms related to the outer DRAO loop, x^t and $\bar{p}^t$, in (Equation 4.52) is the same[7] as the Q_x and Q_p convergence bound in the proof of Proposition 4.1, (c.f. (Equation 4.34)). So a convergence bound of Q in the DRAO-S method can be deduced by plugging it into the proof of Proposition 4.1, i.e, the analysis for the Q convergence in the DRAO method.

Proposition 4.3 *Let $z^t := \{x^t, \bar{p}^t, \pi^t\}$ be generated by the DRAO-S method with the outer DRAO loop stepsize satisfying* (Equation 4.27)*,* (Equation 4.28) *and* (Equation 4.29)*, and the inner SPS loop stepsize satisfying* (Equation 4.51)*. Let $\tilde{\omega}^t := \omega_t / \sum_{s=1}^{S_t} q_s^t$ denote the effective summation weight for the inner SPS loops and let M_t be defined in* (Equation 4.38)*. Suppose the following inter-phase stepsize requirements for inner SPS loop hold for $t \geq 2$:*

$$\begin{aligned}
&\tilde{\omega}^t q_1^t (\delta_1^t)^2 M_{t-1}^2 \leq \tilde{\omega}^{t-1} \beta_1^t q_{S_{t-1}}^{t-1} \gamma_{S_{t-1}}^{t-1},\ M_N^2 \leq \gamma_{S_N}^N (\beta_{S_N}^N + \alpha/2), \\
&\tilde{\omega}^t q_1^t \beta_1^t \leq \tilde{\omega}^{t-1} q_{S_{t-1}}^{t-1} (\beta_{S_{t-1}}^{t-1} + \alpha/2),\ \tilde{\omega}^t q_1^t \gamma_1^t \leq \tilde{\omega}^{t-1} q_{S_{t-1}}^{t-1} \gamma_{S_{t-1}}^{t-1}, \\
&\tilde{\omega}^t \delta_1^t q_1^t = \tilde{\omega}^{t-1} q_{S_{t-1}}^{t-1}.
\end{aligned} \tag{4.55}$$

Then the following Q-convergence bound holds for any reference point $z := (x^*, p, \pi)$ and for all $N \geq 1$

$$\begin{aligned}
&\textstyle\sum_{t=1}^N \omega_t Q\left(z^t, z\right) + \omega_N (\eta_N + \alpha/2) \left\|x^N - x^*\right\|^2 / 2 \\
&\qquad \leq \omega_1 \tau_1 \textstyle\sum_{i=1}^m p_i W_{f_i^*}(\pi_i; \pi_i^0) + (\omega_1 \eta_1 + \tilde{\omega}^1 q_1^1 \beta_1^1) \left\|x^0 - x^*\right\|^2 / 2 + \tilde{\omega}^1 q_1^1 \gamma_1^1 D_P^2 / 2.
\end{aligned} \tag{4.56}$$

[7] Except that $\alpha/2$, instead of α, is regarded as the strong convexity modulus for the outer DRAO loop.

The above bound also holds if the last condition in (Equation 4.55) *is replaced by*

$$\gamma_{S_t}^t(\beta_{S_t}^t + \alpha/2) \geq M_t^2,\ \beta_1^t = 0,\ \textit{and}\ \delta_1^t = 0\ \forall t \geq 1. \tag{4.57}$$

Proof: As pointed out above the proposition, dividing both sides of (Equation 4.52) by $\sum_{s=1}^{S_t} q_s^t$, taking its ω_t-weighted sum, and noting the initialization points in (Equation 4.37) and the telescope cancellation resulting from the stepsize requirements (Equation 4.55), we get a convergence bound of Q_x and Q_p given by

$$\begin{aligned}&\textstyle\sum_{t=1}^N \omega_t[Q_x(z^t; z) + Q_p(z^t; z)] + \sum_{t=1}^N \frac{\omega_t}{2}\eta_t \left\|x^t - x^{t-1}\right\|^2 + \omega_N(\eta_N + \alpha/2)\left\|x^N - x^*\right\|^2 \\ &\qquad \leq \tilde{\omega}^1 q_1^1(\gamma_1^1 U(p; \tilde{p}^0) + \beta_1^1 \left\|y^0 - x^*\right\|^2/2) + \tfrac{\omega_1\eta_1}{2}\left\|x^0 - x^*\right\|^2.\end{aligned} \tag{4.58}$$

The preceding bound is almost the same as its counterpart in Proposition 4.1, i.e., (Equation 4.34). Moreover, since the generation of π^t in the outer DRAO loop of the DRAO-S method is also the same as that of the DRAO method, the Q_π convergence bound in Proposition 4.1 (c.f. (Equation 4.31)) is also valid. The desired Q convergence bound in (Equation 4.56) then follows from combining (Equation 4.31) with (Equation 5.62), and noting $U(p; \tilde{p}^0) \leq D_P^2/2$ and $y^0 = x^0$.

In addition, if the alternative stepsize requirement (Equation 4.57) is satisfied, (Equation 4.52) can be simplified further to

$$\begin{aligned}&(\textstyle\sum_{s=1}^{S_t} q_s)(\mathcal{L}(x^t; p, \pi^t) - \mathcal{L}(x; \bar{p}^t, \pi^t) + \tfrac{1}{2}\eta_t \left\|x^t - x^{t-1}\right\|^2 + (\eta_t + \alpha/2)\left\|x^t - x\right\|^2) \\ &\qquad \leq q_1\gamma_1 U(p; p^0) - q_{S_t}\gamma_{S_t} U(p; p^{S_t}) + \tfrac{(\sum_{s=1}^{S_t} q_s)}{2}\eta_t \left\|x^{t-1} - x\right\|^2.\end{aligned}$$

Then a similar argument would lead to the Q convergence bound in (Equation 4.56) as well. □

The next convergence proofs of the DRAO-S method for the smooth problem, i.e., Theorem Theorem 4.4 and Theorem Theorem 4.5, are direct applications of Proposition

4.3.

Proof of Theorem Theorem 4.4 It is easy to verify that the stepsize choice in (Equation 4.39) satisfies the requirements in Proposition 4.3, thus the following convergence bound is valid for any reference point $z := (x^*; p, \pi)$ and for all $N \geq 1$,

$$\sum_{t=1}^{N} \omega_t Q\left(z^t, z\right) + L_f \left\|x^N - x^*\right\|^2 \leq L_f R_0^2 + D_P R_0/\Delta. \tag{4.59}$$

Let $\hat{\pi}_i^N = \nabla f_i(\bar{x}^N)$ and $\hat{p}^N \in \arg\max_{p \in P} \sum_{i=1}^m p_i f_i(\bar{x}^N)$ such that $f(\bar{x}^N) = \mathcal{L}(\bar{x}^N; \hat{p}^N, \hat{\pi}^N)$ (see Lemma 4.1), then the desired convergence result in (Equation 4.40) can be deduced by choosing the reference point to be $(x^*; \hat{p}^N, \hat{\pi}^N)$. Furthermore, the result in (Equation 4.41) can be deduced by substituting in the specific choice of Δ.

Next, we show the boundedness of M_t. Since $\pi^t = \nabla \boldsymbol{f}(\underline{x}^t)$ (c.f. (Equation 4.15)), $\boldsymbol{f}$ is smooth and $\underline{x}^t$ is a convex combination of x^0 and $\{\tilde{x}^t\}$, the boundedness of $\|\pi^t\|_{2,U^*}$ follows from the boundedness of $\tilde{x}^t$. Setting the reference point to the saddle point (x^*, p^*, π^*) (c.f. Lemma 4.2), we get from (Equation 4.59)

$$L_f \left\|x^N - x^*\right\|^2 \leq 2 L_f R_0^2 \, \forall N \geq 2.$$

This shows that x^t's are restricted to be a bounded ball around x^*. Since $\theta_t \leq 1$, the extrapolated sequence $\tilde{x}^t$'s are also restricted to a bounded ball around x^*, implying the boundedness of $\{M_t\}_{t=1}^{\infty}$. □

Proof of Theorem Theorem 4.5 We can verify that the stepsize choice in (Equation 4.42) satisfies the alternative requirements in Proposition 4.3 (c.f. (Equation 4.57)). Setting the reference point z to the saddle point (x^*, p^*, π^*) (c.f. Lemma 4.2), we get from (Equation 4.56) the desired geometric convergence of x^N (Equation 4.43), i.e.,

$$\left\|x^N - x^*\right\|^2 \leq \theta^N [(1 + 2\kappa) \left\|x^0 - x^*\right\|^2 + \tfrac{4D_P^2}{\eta \Delta \alpha}], \forall N \geq 1. \tag{4.60}$$

Observe that the above convergence bound also implies the boundedness of $\{x^t\}$. Thus the existence of an uniform bound for M_t follows from an argument similar to that of the proof of Theorem Theorem 4.4.

Next, we establish an upper bound on the total of inner iterations when κ is large and $\Delta := 2D_P^2/\eta L_f \left\|x^0 - x^*\right\|^2$. The specific choice of Δ allows us to simplify (Equation 4.60) further to

$$\left\|x^N - x^*\right\|^2 \leq \theta^N \left(5\kappa R_0^2\right).$$

Let N_ϵ denotes the least number of phases required to satisfy $\left\|x^{N_\epsilon} - x^*\right\|^2 \leq \epsilon$. Clearly, $N_\epsilon = \mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$. Specifically, since $\kappa \geq 1$ implies $1/\theta \leq 2$, we have $(1/\theta)^{N_\epsilon} \leq 10\kappa R_0/\epsilon$.

For the total inner iteration number, a bound for S_t are provided by the stepsizes requirement (Equation 4.42). Since $M_t \leq \tilde{M}$ and $S_t \leq 1 + \sqrt{2\omega_t\Delta}\tilde{M}$, the total number is upper bounded by

$$\begin{aligned}\textstyle\sum_{t=1}^{N_\epsilon} S_t &\leq N_\epsilon + \textstyle\sum_{t=1}^{N_\epsilon}(\sqrt{\frac{1}{\theta}})^{t-1}\sqrt{\Delta}\tilde{M} = N_\epsilon + \frac{(1/\theta)^{N_\epsilon/2}-1}{\sqrt{1/\theta}-1}\sqrt{\Delta}\tilde{M} \\ &\leq N_\epsilon + \frac{1}{\sqrt{1/\theta}-1}\frac{1}{\sqrt{\sqrt{8\kappa+1}-1}}\frac{\tilde{M}D_P}{\sqrt{L_f}R_0}\frac{16\sqrt{L_f}R_0}{\alpha\sqrt{\epsilon}} \leq N_\epsilon + \frac{64\kappa^{1/4}\tilde{M}D_P}{\alpha\sqrt{\epsilon}}.\end{aligned}$$

The second last inequality follows from the algebraic fact $\sqrt{1+l} - 1 \geq l/4$ for $l \leq 1$, and

$$\begin{aligned}\frac{1}{\sqrt{1/\theta}-1}\frac{1}{\sqrt{\sqrt{8\kappa+1}-1}} &= \frac{1}{\sqrt{1+2/(\sqrt{8\kappa+1}-1)}-1}\frac{1}{\sqrt{\sqrt{8\kappa+1}-1}} \leq 2(\sqrt{8\kappa+1}-1)\frac{1}{\sqrt{\sqrt{8\kappa+1}-1}} \\ &\leq 2\sqrt{\sqrt{8\kappa+1}-1} \leq 4\kappa^{1/4}.\end{aligned}$$

Thus the number of inner iterations, and hence the P projection complexity, are upper bounded by $\mathcal{O}(\frac{\kappa^{1/4}\tilde{M}D_P}{\alpha\sqrt{\epsilon}} + \sqrt{\kappa}\log(\frac{1}{\epsilon}))$. □

Proof of Theorem Theorem 4.6 The proof is similar to that of Theorem Theorem 4.3. Let us first consider the non-strongly convex case. Since $\bar{M}_t \geq M_t, \forall t$, the stepsize choice in (Equation 4.45) satisfies all the requirements in Proposition 4.2 (c.f. (Equation 4.55)). So

substituting the stepsize choice into (Equation 4.56), we obtain the following convergence bound of the Q gap function for any reference point $z := (x^*, p, \pi)$:

$$\begin{aligned} &\textstyle\sum_{t=1}^{N} Q(z^t; z) + \frac{\eta}{2} \left\|x^N - x\right\|^2 + \frac{\gamma}{\Delta} U(p; p^N) \\ &\qquad \leq (\tfrac{\eta}{2} + \tfrac{\beta}{2\Delta}) \left\|x^0 - x\right\|^2 + \tfrac{\gamma}{\Delta} U(p; p^0) + \tau \textstyle\sum_{i=1}^{m} p_i V_i(\pi_i; \pi_i^0). \end{aligned} \tag{4.61}$$

The desired function value convergence bound (Equation 4.46) follows immediately by selecting the reference point to be $(x^*, \hat{p}^N, \hat{\pi}^N)$, where $\hat{p}^N \in \arg\max_{p \in P} \sum_{i=1}^{m} p_i f_i(\bar{x}^N)$ and $\hat{\pi}_i^N = \nabla f_i(\bar{x}^N)$.

The convergence bound (Equation 4.49) for the strongly case also follows from substituting the stepsize choice in (Equation 4.48) into (Equation 4.56). □

4.5 Lower Communication Complexities

In this section, we establish theoretical lower bounds for distributed risk-averse optimization to show the communication complexities of both DRAO and DRAO-S are not improvable. Towards that end, we propose a distributed prox mapping (DPM) computing environment consisting of the following requirements and propose uniform lower bounds for all algorithms satisfying the requirement.

- **Local memory**: the server node has a finite local memory $\mathcal{M}_s$ and each worker node has a finite primal memory and a finite dual memory, $\mathcal{M}_i$ and $\mathcal{M}_i^\pi$, respectively. In the beginning, the local memories contain only the trivial vector 0, i.e.,

  $$\mathcal{M}_{i,0} = \mathcal{M}_{s,0} := \{0\},\ \mathcal{M}_{i,0}^\pi := \{0\}\ \forall i \in [m].$$

  In one communication round, these local memories can be updated by both local computation and server-worker communication:

  $$\mathcal{M}_{s,t+1} := \mathcal{M}_{s,t}^{\text{cp}} \cup \mathcal{M}_{s,t}^{\text{comm}},\ \mathcal{M}_{i,t+1} := \mathcal{M}_{i,t}^{\text{cp}} \cup \mathcal{M}_{i,t}^{\text{comm}},\ \mathcal{M}_{i,t+1}^\pi := \mathcal{M}_{i,t}^{\pi,\text{cp}},\ \forall i \in [m],$$

where $\mathcal{M}_{s,t}^{\text{cp}}$ and $\mathcal{M}_{i,t}^{\pi,\text{cp}}$ represent results from the local computation, and $\mathcal{M}_{s,t}^{\text{comm}}$ and $\mathcal{M}_{i,t}^{\text{comm}}$ denote the vector(s) communicated to the server and the i^{th} worker node, respectively.

- **Server-worker communication:** in one communication round, each worker can send one vector from its local primal memory to the server:

$$\mathcal{M}_{s,t}^{\text{comm}} := \{y_i \in \text{span}(\mathcal{M}_{i,t-1}), i \in [m]\},$$

and the server can share one vector from its memory with the worker:

$$\mathcal{M}_{i,t}^{\text{comm}} \in \text{span}(\mathcal{M}_{s,t-1}).$$

- **Local computations**: between communication rounds, each worker can query its dual prox mapping oracle and the A_i-multiplication oracle[8] for $L \geq 0$ times.

$$\begin{aligned}
&\mathcal{M}_{i,t}^{\text{cp}} := \mathcal{M}_{i,t}^{\text{cp},L},\ \mathcal{M}_{i,t}^{\pi,\text{cp}} := \mathcal{M}_{i,t}^{\pi,L} \text{ where } \mathcal{M}_{i,t}^{\text{cp},0} := \mathcal{M}_{i,t-1}, \mathcal{M}_{i,t}^{\pi,0} := \mathcal{M}_{i,t-1}^{\pi}.\\
&\textit{For } l = 1, 2, 3, \ldots, L:\\
&\quad \mathcal{M}_{i,t}^{\text{cp},l} = \mathcal{M}_{i,t}^{\text{cp},l-1} \cup \{A_i^\top \pi_i^{t,l},\ A_i^\top \bar{\pi}_i\},\ \mathcal{M}_{i,t}^{\pi,l} := \mathcal{M}_{i,t}^{\pi,l-1} \cup \{\pi_i^{t,l}, A_i\bar{x}\},\\
&\qquad\qquad \text{where } \bar{x} \in \text{span}(\mathcal{M}_{i,t}^{\text{cp},l-1}),\\
&\quad \bar{\pi}_i \in \text{span}(\mathcal{M}_{i,t}^{\pi,l-1}),\ \pi_i^{t,l} \in \underset{\pi_i \in \Pi_i}{\arg\max} \langle A_i\bar{x}, \pi_i\rangle - f_i^*(\pi_i) - \tfrac{\tau}{2}\left\|\pi_i - \bar{\pi}_i\right\|^2,\\
&\qquad\qquad \text{for some } \tau \geq 0.
\end{aligned} \tag{4.62}$$

[8] The oracle returns matrix vector multiplication result of the form $A_i x$ and $A_i^\top \pi$.

The server node can query its $u(x)$ prox mapping oracle for $L \geq 0$ times.

$\mathcal{M}_{s,t}^{\text{cp}} := \mathcal{M}_{s,t}^{\text{cp},L}$ where $\mathcal{M}_{s,t}^{\text{cp},0} := \mathcal{M}_{s,t-1}$.

For $l = 1, 2, 3, \ldots, L$:

$\mathcal{M}_{s,t}^{\text{cp},l} := \mathcal{M}_{s,t}^{\text{cp},l-1} \cup \{x_s^l\}$, where $x_s^l := \arg\min_{x \in X} u(x) + \frac{\eta}{2} \|x - \bar{x}\|^2, \bar{x} \in \text{span}(\mathcal{M}_{s,t}^{\text{cp},l-1})$.

- **Output solution:** the output solution x^t comes from local primal memories,

$$x^t \in \text{span}((\cup_{i \in [m]} \mathcal{M}_{i,t}) \cup \mathcal{M}_{s,t}), t \geq 1.$$

The only hard requirement for the DPM environment is that only one vector can be sent and received by each worker during one communication round. Indeed, the computations supported by the DPM environment are quite strong in several aspects. First, it allows gradient evaluation of f_i since it is equivalent to the π_i-prox mapping (c.f. (Equation 4.62)) with $\tau = 0$, i.e.,

$$\bar{\pi}_i = \nabla f_i(\bar{x}) \Leftrightarrow \bar{\pi}_i \in \arg\max_{\pi_i \in \Pi_i} \langle \pi_i, \bar{x} \rangle - f_i^*(\pi_i).$$

Second, it allows a possibly large number of local computation steps to be performed between communications. This assumption of generous computing resource at each node helps us to focus on the communication bottleneck. Third, it allows the freedom to make an arbitrary selection from the span of the local memory for communication, computation, and outputting solutions. For example, it might appear that the DRAO method violates the requirement because of the (x, p)-prox mapping in Line 5 of Algorithm Algorithm 9. However, if we let $(x^t, \hat{p}^t)$ be an optimal pair of saddle point solutions in the (x, p)-prox mapping step (c.f. (Equation 4.36)), the output x^t can be written alternatively as

$$x^t \leftarrow \arg\max_{x \in X} \eta_t \|x - \underline{x}\|^2 / 2 + u(x),$$

where $\underline{x} := x^{t-1} - \sum_{i=1}^m \hat{p}_i^t v_i^t / \eta_t$ and $\underline{x} \in \text{span}(\mathcal{M}_{s,t})$. So the (x, p)-prox mapping ac-

tually satisfies the above local computation requirement. Moreover, the computation and communication of $f_i^*(\pi_i^t)$'s are unnecessary because they are only used for generating $\hat{p}^t$. Since all other steps are directly supported, the DRAO method can be implemented on the DPM environment. Indeed, our setup implies that the desired p can be obtained from any oracle when selecting x (from the span of local memory of the server). This renders all communication and computation related to p unnecessary. So the DRAO-S method, and, more generally, any distributed algorithm consisting of the x-prox mapping, the π-prox mapping, and some p update can be implemented on the DPM environment. For simplicity, we will call an algorithm satisfying the DPM requirement a DPM algorithm for the rest of this section.

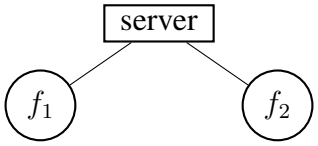


Figure 4.1: Network topology of hard instances.

Now we present some hard instances, inspired by [2, 1, 85], for all DPM algorithms. We first describe a network topology and a general result which will be used in all our constructions. As shown in Figure Figure 4.1, the problem has only two workers, node 1 and node 2. Let $\mathcal{K}_i$ denote the subspace with non-zero entries only in the first i coordinates, $\mathcal{K}_i := \{x \in \mathbb{R}^n : x_j = 0 \ \forall j > i\}$. We will construct f_1 and f_2 such that the iterate x^t generated in t communication rounds will be restricted to a certain $\mathcal{K}_i$. Towards that end, we call a hard problem *odd-even preserving* if the memories generated by any DPM

algorithm satisfies

$$\begin{aligned}
&\mathcal{M}_{1,0} \cup \mathcal{M}_{2,0} \cup \mathcal{M}_{s,0} \subset \mathcal{K}_2, \\
&\mathcal{M}_{1,t-1} \subset \mathcal{K}_i \Rightarrow \begin{cases} \mathcal{M}_{1,t}^{\text{cp}} \subset \mathcal{K}_i & i \geq 2 \text{ even} \\ \mathcal{M}_{1,t}^{\text{cp}} \subset \mathcal{K}_{i+1} & i \geq 2 \text{ odd} \end{cases}, \\
&\mathcal{M}_{2,t-1} \subset \mathcal{K}_i \Rightarrow \begin{cases} \mathcal{M}_{2,t}^{\text{cp}} \subset \mathcal{K}_{i+1} & i \geq 2 \text{ even} \\ \mathcal{M}_{2,t}^{\text{cp}} \subset \mathcal{K}_i & i \geq 2 \text{ odd} \end{cases}, \\
&\mathcal{M}_{s,t-1} \subset \mathcal{K}_i \Rightarrow \mathcal{M}_{s,t}^{\text{cp}} \subset \mathcal{K}_i.
\end{aligned} \tag{4.63}$$

This property stipulates that the progresses on the reachable subspace $\mathcal{K}_i$ are possible only on node 1 or 2 depending on if i is odd or even, so that a large number of communication rounds between node 1 and 2 are necessary for a non-trivial solution. The next lemma formalizes such limited progress by a DPM algorithm.

Lemma 4.4 *If the odd-even preserving property* (Equation 4.63) *holds, the output solution* x^t *generated by a DPM algorithm after* t *communication rounds satisfy* $x^t \subset \mathcal{K}_{\lceil t/2 \rceil + 2}$.

Proof: Let $\mathcal{M}_t := \text{span}(\mathcal{M}_{1,t} \cup \mathcal{M}_{2,t} \cup \mathcal{M}_{s,t})$, and let $t(i) := \min\{t \geq 0 : \exists y \in \mathcal{M}_t, j \geq i \; s.t \; y_j \neq 0\}$ denote the first time a vector with a non-zero $j \geq i$th index is generated. We develop a lower bound for $t(i)$.

Consider an even $i > 2$. By the definition of $t(i)$, $\mathcal{M}_{t(i)-1} \subset \mathcal{K}_{i-1}$. The odd-even preserving property then implies $\mathcal{M}_{2,t(i)}^{\text{cp}} \subset \mathcal{K}_{i-1}$ and $\mathcal{M}_{s,t(i)}^{\text{cp}} \subset \mathcal{K}_{i-1}$, so $\mathcal{M}_{1,t(i)} \subset \mathcal{K}_i$, $\mathcal{M}_{s,t(i)} \subset \mathcal{K}_{i-1}$, and $\mathcal{M}_{2,t(i)} \subset \mathcal{K}_{i-1}$ after one communication round. Next, the odd-even preserving property again implies $\mathcal{M}_{1,t(i)+1}^{\text{cp}} \subset \mathcal{K}_i$, $\mathcal{M}_{2,t(i)+1}^{\text{cp}} \subset \mathcal{K}_{i-1}$ and $\mathcal{M}_{s,t(i)+1}^{\text{cp}} \subset \mathcal{K}_{i-1}$, so $\mathcal{M}_{1,t(i)+1} \subset \mathcal{K}_i$, $\mathcal{M}_{s,t(i)+1} \subset \mathcal{K}_i$, and $\mathcal{M}_{2,t(i)} \subset \mathcal{K}_{i-1}$ after another communication round. Therefore, we have $t(i+1) > t(i)+1$, i.e., $t(i+1) \geq t(i)+2$. The same recursive bound can also be obtained for an odd $i \geq 2$. In view of $t(2) \geq 0$, the largest non-zero index i in $\mathcal{M}_t$ satisfies $t \geq t(i) \geq t(2) + 2i - 4 \geq 2i - 4$, thus $i \leq \lceil t/2 \rceil + 2$. $\square$

We are now ready to provide lower bounds under different problem settings. The next two results establish tight lower communication bounds for the smooth problem with a non-strongly convex $u(x)$ and a strongly convex $u(x)$, respectively.

Theorem 4.7 *Let $L_f > 0$, $R_0 \geq 1$ and $\epsilon > 0$ be given. For a sufficiently large problem dimension, i.e., $n > 2\lceil \sqrt{L_f} R_0 / 8\sqrt{\epsilon} \rceil$, there exists a smooth hard problem of form* (Equation 4.1) *with an aggregate smoothness constant L_f (c.f.* (Equation 4.17)*), $\|x^0 - x^*\| \leq R_0$ such that any DPM algorithm takes at least $\Omega(\sqrt{L_f} R_0 / \sqrt{\epsilon})$ communication rounds to find an ϵ-optimal solution.*

Proof: Consider the following hard problem parameterized by $\beta \geq 0, \gamma \geq 0$ and $k \geq 4$,

$$\begin{aligned} &f(x) := \max_{p \in \Delta_2^+} p_1 f_1(x) + p_2 f_2(x) + u(x) \text{ with } X = \mathbb{R}^{2k+1},\ u(x) = 0, \\ &f_1(x) := \tfrac{\beta}{2}[2\textstyle\sum_{i=1}^{k}(x_{2i-1} - x_{2i})^2 + x_1^2 + x_{2k+1}^2 - 2\gamma x_1], \\ &f_2(x) := \tfrac{\beta}{2}[2\textstyle\sum_{i=1}^{k}(x_{2i} - x_{2i+1})^2 + x_1^2 + x_{2k+1}^2 - 2\gamma x_1]. \end{aligned} \tag{4.64}$$

Its aggregate smoothness constant $\bar{L}_f$ (c.f. (Equation 4.17)) satisfies $\bar{L}_f \leq 6\beta$, and its optimal solution (x^*, p^*) satisfies

$$\begin{aligned} p^* = [\tfrac{1}{2}, \tfrac{1}{2}],\ x_i^* = \gamma(1 - \tfrac{i}{2k+2})\ \forall i \leq 2k+1, \\ \text{s.t. } \left\|x^0 - x^*\right\| \leq \gamma\sqrt{k+1} \text{ and } f_* = -\tfrac{\beta\gamma^2}{2}[1 - \tfrac{1}{2k+2}]. \end{aligned}$$

Their optimality can be verified with the first order conditions:

$$0 = \nabla(\tfrac{1}{2}f_1 + \tfrac{1}{2}f_2)(x^*) \text{ and } [1/2, 1/2] \in \arg\max_{p \in \Delta_2^+} p_1 f_1(x^*) + p_2 f_2(x^*).$$

The even-odd preserving property holds for (Equation 4.64). To see this, consider the worker node f_1. Let an even $i \geq 2$ be given and assume $\mathcal{M}_{1,t-1} \subset \mathcal{K}_i$, i.e., $\mathcal{M}_{1,t}^{\text{cp},0} \subset \mathcal{K}_i$. Because $A_i = I$, the update rule in (Equation 4.62) imply that $\mathcal{M}_{1,t}^{\pi,0} \subset \mathcal{K}_i$. We show $\mathcal{M}_{1,t}^{\text{cp},l} \cup \mathcal{M}_{1,t}^{\pi,l} \subset \mathcal{K}_i$ for all $l \geq 0$ by induction. Clearly, the statement holds for $l = 0$.

If $\mathcal{M}_{1,t}^{\text{cp},l-1} \cup \mathcal{M}_{1,t}^{\pi,l-1} \subset \mathcal{K}_i$, $\bar{x}$ and $\bar{\pi}_i$ chosen in (Equation 4.62) must be in $\mathcal{K}_i$. As for the π_i-prox mapping, if $\tau = 0$, $\pi_1^{t,l} := \nabla f_1(\bar{x}) \subset \mathcal{K}_i$. If $\tau > 0$, Lemma 4.5 in the appendix allows us to write $\pi_1^{t,l}$ as

$$\pi_1^{t,l} = \bar{\pi}_1 + \tfrac{1}{\tau}(\bar{x} - \underline{y}), \text{ where } \underline{y} \leftarrow \arg\min_y f_1(y) + \tfrac{1}{2\tau} \left\|\bar{x} + \tau\bar{\pi}_1 - y\right\|^2.$$

In particular, $\underline{y} \in \mathcal{K}_i$ because

$$\underline{y} = \arg\min_{x \in \mathbb{R}^{2k+1}} \tfrac{\beta}{2}[2\textstyle\sum_{j=1}^{i/2}(x_{2j-1} - x_{2j})^2 + x_1^2 - 2\gamma x_1] + \tfrac{1}{2\tau}\sum_{j=1}^{i} \left\|\bar{x}_j + \tau\bar{\pi}_{1,j} - x_j\right\|^2 + \tfrac{\beta}{2}x_{2k+1}^2 + \tfrac{1}{2\tau}\textstyle\sum_{j=i+1}^{2k+1} \left\|x_j\right\|^2.$$

So $\pi_1^{t,l} \in \mathcal{K}_i$ also holds for $\tau > 0$. Thus the principle of induction implies that $\mathcal{M}_{1,t}^{\text{cp},L} \cup \mathcal{M}_{1,t}^{\pi,L} \subset \mathcal{K}_i, \forall L \geq 0$, i.e., $\mathcal{M}_{1,t}^{\text{cp}} \subset \mathcal{K}_i$. In addition, when $i \geq 2$ is odd and $\mathcal{M}_{1,t-1} \subset \mathcal{K}_i$, we have $\mathcal{M}_{1,t-1} \subset \mathcal{K}_i \subset \mathcal{K}_{i+1}$. Since $i + 1$ is even, the preceding result implies that $\mathcal{M}_{1,t}^{\text{cp}} \subset \mathcal{K}_{i+1}$. A similar result can also be derived for the worker f_2 for both even and odd $i \geq 2$. Therefore, problem (Equation 4.64) satisfies the even-odd preserving property.

Applying Lemma 4.4, the output solution x^k from any DPM algorithm in k communication rounds must satisfy $x^k \in \mathcal{K}_k$. In particular, let $\bar{f} := (f_1 + f_2)/2$ denote a lower bound for f. Then $f(x^k) \geq \min_{x \in \mathcal{K}_k} f(x) \geq \min_{x \in \mathcal{K}_k} \bar{f}(x) = -\frac{\beta\gamma^2}{2}[1 - \frac{1}{k+1}]$.

Now we set the parameters in (Equation 4.64) to obtain the desired lower bound. If $\epsilon \geq L_f R_0^2/4096$, $\Omega(\sqrt{L_f}R_0/\sqrt{\epsilon}) = \Omega(1)$, so the lower bound clearly hold. Otherwise, we set $\beta := L_f/6$, $\gamma := R_0/\sqrt{k+1}$ and $k := \lceil \sqrt{L_f}R_0/8\sqrt{\epsilon} \rceil$ such that (Equation 4.64) is L_f-smooth (c.f (Equation 4.17)) with $\|x^0 - x^*\| \leq R_0$ and $k \geq 4$. A solution x^k generated by any DPM algorithm in k communication rounds satisfy $f(x^k) - f_* \geq \frac{\gamma^2\beta}{4k+4} \geq \epsilon$. Thus they imply the desired $\Omega(\sqrt{L_f}R_0/\sqrt{\epsilon})$ lower communication complexity bound when the problem dimension is $2\lceil \sqrt{L_f}R_0/8\sqrt{\epsilon} \rceil + 1$. $\square$ We remark here that the above risk-averse lower bound is the same as the risk-neutral lower bound of $\Omega(\sqrt{L_{f,\bar{p}}}R_0/\sqrt{\epsilon})$, developed in [85], if P is a singleton set of the empirical distribution, $P = \{\bar{p} := (1/m, ..., 1/m)\}$, and $L_{f,\bar{p}}$ denotes the aggregate smoothness constant (c.f. (Equation 4.17)) associated with

$\bar{p}$. But, other than the intuition that the risk-averse problem should be harder than the risk-neutral problem, the latter bound offers limited insights. Our risk-averse lower bound can be larger than the risk-neutral lower bound because the aggregate smoothness constant L_f (c.f. (Equation 4.17)) defined over a non-trivial P can be significantly larger than $L_{f,\bar{p}}$. For example, consider an expanded version of (Equation 4.64) constructed by adding $(m-2)$ additional workers with constant local cost functions, $f_i(x) \equiv C$ for some $C < f_*$, and by setting P to the m-dimensional simplex Δ_m^+. The same argument as above will lead to the same lower bound of $\Omega(\sqrt{L_f}R_0/\sqrt{\epsilon})$ for the expanded problem. However, because the smoothness constants of $\{f_i\}_{i=3}^m$ are zero, we have $L_{f,\bar{p}} \leq 2L_f/m << L_f$.

Theorem 4.8 *Let $L_f > 8\alpha > 0$ and $\epsilon > 0$ be given. There exists an infinite-dimensional smooth problem of form* (Equation 4.1) *with an aggregate smoothness constant L_f (c.f.* (Equation 4.17)*) and a strong convexity modulus α such that any DPM algorithm requires at least $\Omega(\sqrt{L_f/\alpha}\log(1/\epsilon))$[9] communication rounds to find an ϵ-close solution, i.e., x such that $\|x - x^*\|^2 \leq \epsilon$.*

Proof: Again we prove the result by construction. Consider the following infinite dimensional problem parameterized by $\beta > 2\alpha$

$f(x) := \max_{p\in\Delta_2^+} p_1 f_1(x) + p_2 f_2(x) + u(x)$ where $X := \mathbb{R}^\infty$

$u(x) := \frac{\alpha}{2}\|x\|^2, f_1(x) := \frac{\beta-\alpha}{4}[x^\top A_1 x - 2x_1],\ f_2(x) := \frac{\beta-\alpha}{4}[x^\top A_2 x - 2x_1],$

$$A_1 := \begin{bmatrix} \frac{3+5\gamma}{4+4\gamma} & -1 & & & & & & \\ -1 & 1 & & & & & & \\ & & 1 & -1 & & & & \\ & & -1 & 1 & & & & \\ & & & & 1 & -1 & & \\ & & & & -1 & 1 & & \\ & & & & & & \ddots & \end{bmatrix}, A_2 := \begin{bmatrix} \frac{5+3\gamma}{4+4\gamma} & & & & & & & \\ & 1 & -1 & & & & & \\ & -1 & 1 & & & & & \\ & & & 1 & -1 & & & \\ & & & -1 & 1 & & & \\ & & & & & \ddots & & \\ & & & & & & \ddots & \end{bmatrix}, \gamma = \frac{\alpha}{\beta}. \tag{4.65}$$

[9] We ignore the problem parameter R_0 inside the log.

Clearly, the aggregate smoothness constant (c.f. (Equation 4.17)) of the problem is bounded by $\beta - \alpha$ and its strong convexity modulus is α. The optimal solutions are given by $p^* = (1/2, 1/2)$ and x^*, with $x_i^* = (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^i \ \forall i \geq 1$, since they satisfy the first order optimality conditions:

$$\nabla(\tfrac{1}{2}f_1 + \tfrac{1}{2}f_2 + u)(x^*) = 0 \text{ and } (\tfrac{1}{2}, \tfrac{1}{2}) \in \arg\max_{p \in \Delta_2^+} p_1 f_1(x^*) + p_2 f_2(x^*).$$

Moreover, similar to Theorem Theorem 4.7, the alternating block diagonal structure of A_1 and A_2 implies the even-odd preserving property, so x^k generated by any DPM algorithm in $k \geq 4$ communications rounds satisfy $x^k \subset \mathcal{K}_k$, i.e.,

$$\begin{aligned} \left\|x^k - x^*\right\|^2 &\geq \textstyle\sum_{i=k+1}^{\infty} (x_i^*)^2 = (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^{2k} (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^2 / (1 - (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^2) = (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^{2k} R_0^2 \\ &= (1 - \tfrac{2\sqrt{\gamma}}{1+\sqrt{\gamma}})^{2k} R_0^2 \geq (1 - 2\sqrt{\gamma})^{2k} R_0^2 = (1 - 2\sqrt{\gamma})^{2k} R_0^2 \\ &\geq (1 - 2\sqrt{\alpha/(\beta - \alpha)})^{2k} R_0^2, \end{aligned} \tag{4.66}$$

where $R_0^2 := (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^2 / (1 - (\frac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^2)$. Thus it takes at least $\Omega(\sqrt{(\beta - \alpha)/\alpha} \log(1/\epsilon))$ communication rounds to obtain an x with $\|x - x^*\|^2 \leq \epsilon$.

Now we select the parameter β to derive the lower complexity bound. If $\epsilon \geq (1 - 2\sqrt{1/\kappa})^8 R_0^2$ such that $\Omega(\sqrt{L_f/\alpha} \log(1/\epsilon)) = \Omega(1)$, the desired lower bound clearly holds. Otherwise, we can set $\beta := L_f + \alpha$ such that the hard problem (Equation 4.65) is L_f-smooth, and the desired lower communication bound of $\Omega(\sqrt{L_f/\alpha} \log(1/\epsilon))$ follows from (Equation 4.66). □ We remark here that a finite dimensional hard problem can also be obtained by modifying (Equation 4.65) according to [56]. Next, we move on to consider the structured non-smooth problem.

Theorem 4.9 *Let $M_A > 0$, $D_\Pi > 0$ $R_0 \geq 1$ and $\epsilon > 0$ be given. When the problem dimension n is sufficiently large (specified below), there exists a structured non-smooth problem f of form* (Equation 4.1) *with $M_A \geq \max_{i \in [m]} \|A_i\|_{2,2}$, $D_\Pi \geq \max_{i \in [m]} \max_{\pi_i, \bar{\pi}_i \in \Pi_i} \|\pi_i - \bar{\pi}_i\|$ (c.f.* (Equation 4.24)*) and $R_0 \geq \|x^0 - x^*\|$ such that the*

following communication lower bounds hold.

a) When $u(x)$ is convex and $n > 2\lceil D_\Pi M_A R_0/96\epsilon \rceil$, any DPM algorithm requires at least $\Omega(M_A D_\Pi R_0/\epsilon)$ communication rounds to find an ϵ-optimal solution.

b) When $u(x)$ is $\alpha > 0$ strongly convex and $n > 2\lceil D_\Pi M_A/48\sqrt{\alpha\epsilon} \rceil$, any DPM algorithm requires at least $\Omega(M_A D_\Pi/\sqrt{\epsilon\alpha})$ communication rounds to find an ϵ-optimal solution.

Proof: We consider the following hard problem parameterized by $k \geq 4$, α, γ_A and γ_π:

$$\begin{aligned}
&f(x) := \max\nolimits_{p\in\Delta_2^+} p_1 f_1(x) + p_2 f_2(x) \text{ with } X = \mathbb{R}^{2k+1},\ u(x) = \tfrac{\alpha}{2}\left\|x\right\|^2, \\
&f_1(x) := \gamma_A\gamma_\pi[2\textstyle\sum_{i=1}^k |x_{2i-1} - x_{2i}| - (\frac{3}{2} + \frac{1}{k})x_1], \\
&f_2(x) := \gamma_A\gamma_\pi[2\textstyle\sum_{i=1}^k |x_{2i} - x_{2i+1}| - (\frac{1}{2} + \frac{1}{k})x_1].
\end{aligned} \tag{4.67}$$

In particular, the scenario cost functions f_1 and f_2 are specified in the structured maximization form (c.f. (Equation 4.1)), $f_i(x) := \max_{\pi_i\in\Pi_i}\langle A_i x, \pi_i\rangle - f_i^*(\pi)$ with

$$\begin{aligned}
A_1 &:= \gamma_A \begin{bmatrix} -(\frac{3}{2}+\frac{1}{k}) & & & & & & & \\ 1 & & -1 & & & & & \\ & & & 1 & -1 & & & \\ & & & & & \ddots & & \\ & & & & & & 1 & -1 & 0 \end{bmatrix}, \\
\Pi_1 &:= \gamma_\pi(\{1\}\times[-2,2]^k) \subset \mathbb{R}^{k+1},\ f_1^*(\pi_1) \equiv 0, \\
A_2 &:= \gamma_A \begin{bmatrix} -(\frac{1}{2}+\frac{1}{k}) & & & & & & \\ & & 1 & -1 & & & \\ & & & & 1 & -1 & \\ & & & & & \ddots & \\ & & & & & & 1 & -1 \end{bmatrix}, \\
\Pi_2 &:= \gamma_\pi(\{1\}\times[-2,2]^k) \subset \mathbb{R}^{k+1},\ f_2^*(\pi_2) \equiv 0.
\end{aligned} \tag{4.68}$$

Clearly, $\max_{i\in[2]}\left\|A_i\right\|_{2,2} \leq 2\gamma_A$ and $\max_{i\in[m]}\max_{\pi_i,\bar{\pi}_i\in\Pi_i}\left\|\pi_i - \bar{\pi}_i\right\| \leq 5\sqrt{k}\gamma_\pi$. Furthermore, $p^* := [1/2, 1/2]$ and $x^* := \frac{\gamma_\pi\gamma_A}{2k\alpha}[2, 1, 1, \ldots, 1]$ form the optimal solution since they

satisfy the first order optimality conditions given by:

$$0 \in \partial(\tfrac{1}{2}f_1 + \tfrac{1}{2}f_2 + u)(x^*) \text{ and } (\tfrac{1}{2}, \tfrac{1}{2}) \in \arg\max_{p \in \Delta_2^+} p_1 f_1(x^*) + p_2 f_2(x^*).$$

So $f_* = f(x^*) = -(\frac{\gamma_\pi^2 \gamma_A^2}{2k^2\alpha} + \frac{\gamma_A^2 \gamma_\pi^2}{4k\alpha})$, $\|x^* - x^0\|^2 = \frac{\gamma_A^2 \gamma_\pi^2}{4k^2\alpha^2}(2k+4) \leq \frac{\gamma_A^2 \gamma_\pi^2}{k\alpha^2}$.

Now, let us verify (Equation 4.67) satisfies the even-odd preserving property. Consider the worker f_1. Let an even $i \geq 2$ be given and assume $\mathcal{M}_{1,t-1} \subset \mathcal{K}_i$, i.e., $\mathcal{M}_{1,t}^{\text{cp},0} \subset \mathcal{K}_i$. Let $\mathcal{S}_j$ denote a (dual) subspace with non-zero entries only in the first j coordinates, $\mathcal{S}_j := \{\pi \in \mathbb{R}^{k+1} : \pi_l = 0 \; \forall l > j\}$. Because of the block structure of A_1, if the $i+1^{\text{th}}$ coordinate of π_1 is non-zero for any $i \geq 2$, the $2i-1^{\text{th}}$ and the $2i^{\text{th}}$ coordinates of $A_1\pi_1$ must be non-zero. So we have $\mathcal{M}_{1,t}^{\pi,0} \subset \mathcal{S}_{i/2+1}$; otherwise the update rule (Equation 4.62) in the DPM environment would lead to $\mathcal{M}_{1,t-1} \not\subset \mathcal{K}_i$.

Next, we show $\mathcal{M}_{1,t}^{\text{cp},l} \subset \mathcal{K}_i$ and $\mathcal{M}_{1,t}^{\pi,l} \subset \mathcal{S}_{i/2+1}$ for all $l \geq 0$ by induction. Clearly the statement holds for $l = 0$. Moreover, if $\mathcal{M}_{1,t}^{\text{cp},l-1} \subset \mathcal{K}_i$ and $\mathcal{M}_{1,t}^{\pi,l-1} \subset \mathcal{S}_{i/2+1}$, $A_1\bar{x}$ and $\bar{\pi}_1$ must be in $\mathcal{S}_{i/2+1}$, so the dual proximal in (Equation 4.62) can be written as

$$\pi_1^{t,l} = \arg\max_{\pi_1 \in \Pi_1} \textstyle\sum_{j=1}^{i/2+1} (A_1\bar{x})_j \pi_{1,j} - \tau/2(\pi_{1,j} - \bar{\pi}_{1,j})^2 - \tau/2 \sum_{j=i/2+2}^{k+1} (\pi_{1,j})^2.$$

This leads us to $\pi_1^{t,l} \in \mathcal{S}_{i/2+1}$ and $A_1^\top \pi_1^{t,l} \in \mathcal{K}_i$, i.e., $\mathcal{M}_{1,t}^{\text{cp},l} \subset \mathcal{K}_i$ and $\mathcal{M}_{1,t}^{\pi,l} \subset \mathcal{S}_{i/2+1}$. Then the principle of induction implies that the statement holds for all $l \geq 0$, i.e., $\mathcal{M}_{1,t}^{\text{cp}} \subset \mathcal{K}_i$. In addition, when $i \geq 2$ is odd and $\mathcal{M}_{1,t-1} \subset \mathcal{K}_i$, we have $\mathcal{M}_{1,t-1} \subset \mathcal{K}_{i+1}$. Since $i+1$ is even, the preceding result implies that $\mathcal{M}_{1,t}^{\text{cp}} \subset \mathcal{K}_{i+1}$. The property (Equation 4.63) for the worker f_2 for both even and odd i's can also be deduced in a similar way. Therefore we have shown that the even-odd preserving property holds for the hard problem (Equation 4.67).

Next, applying Lemma 4.4, the solution x^k returned by any DPO algorithm in $k \geq 4$ communication rounds must satisfy $x^k \in \mathcal{K}_k$. We provide a lower bound of f on $\mathcal{K}_k$. Let $\bar{f} := \frac{1}{2}(f_1 + f_2) + u(x)$ denote a uniform lower bound for f given by

$$\bar{f}(x) := \gamma_\pi \gamma_A [\textstyle\sum_{i=1}^{2k} |x_i - x_{i+1}| - (1 + \frac{1}{k})x_1] + \frac{\alpha}{2} \|x\|^2.$$

In order to find the minimum of $\bar{f}$ on $\mathcal{K}_k$, observe that arranging $\{x_i\}_{i=1}^k$ in a decreasing order decreases $\bar{f}$. Moreover, if $x_k < 0$, setting all negative coordinates to zero decreases $\bar{f}$, so we can focus on $x_1 \geq x_2 ... \geq x_k \geq x_{k+1} = ... = x_{2k+1} = 0$.

$$\min_{x \in \mathcal{K}_k} \bar{f}(x) = \min_{x \in \mathcal{K}_k} -\frac{\gamma_\pi \gamma_A}{k} x_1 + \frac{\alpha}{2} x_1^2 + \frac{\alpha}{2} \sum_{i=2}^{2k+1} x_i^2 \geq -\frac{\gamma_\pi \gamma_A}{2k^2 \alpha}.$$

Thus, $f(x^k) - f_* \geq \min_{x \in \mathcal{K}_k} \bar{f}(x) - f_* \geq \gamma_A^2 \gamma_\pi^2 / 4k\alpha$.

Finally, we choose appropriate problem parameters to establish the lower bounds. If $\epsilon \geq D_\Pi M_A R_0 / 400\epsilon$, $\Omega(M_A D_\Pi R_0/\epsilon) = \Omega(1)$, so the lower bound in a) clearly holds. Otherwise, setting $k := \lceil D_\Pi M_A R_0 / 96\epsilon \rceil$, $n := 2k+1$, $\gamma_\pi := D_\Pi / 5\sqrt{k}$, $\gamma_A := M_A/2$, and $\alpha := \gamma_A \gamma_\pi / R_0 \sqrt{k}$, the parameters of (Equation 4.67) satisfy $\|x^0 - x^*\| \leq R_0$, and $\max_{i \in [2]} \|A_i\|_{2,2} \leq M_A$, $\max_{i \in [m]} \max_{\pi_i, \bar{\pi}_i \in \Pi_i} \|\pi_i - \bar{\pi}_i\| \leq D_\Pi$, $4 \leq k$. Since the minimum optimality gap attainable in $k = \Omega(M_A D_\Pi R_0/\epsilon)$ communication rounds is lower bounded by ϵ, the result in a) follows.

Now consider $u(x)$ being α-strongly convex for a fixed $\alpha > 0$. If $\epsilon \geq D_\Pi^2 M_A^2 / 40000\alpha$, $\Omega(M_A D_\Pi / \sqrt{\alpha\epsilon}) = \Omega(1)$, so the lower bound in b) clearly holds. Otherwise, setting $k := \lceil D_\Pi M_A / 48\sqrt{\alpha\epsilon} \rceil$, $n := 2k+1$, $\gamma_\pi := D_\Pi / 5\sqrt{k}$, and $\gamma_A := M_A/2$, the parameters of (Equation 4.67) satisfy $\max_{i \in [2]} \|A_i\|_{2,2} \leq M_A$, $\max_{i \in [m]} \max_{\pi_i, \bar{\pi}_i \in \Pi_i} \|\pi_i - \bar{\pi}_i\| \leq D_\Pi$, $4 \leq k$, and $\|x^0 - x^*\| \leq R_0$. Since the minimum optimality gap attainable in $k = \Omega(M_A D_\Pi / \sqrt{\alpha\epsilon})$ communication rounds is lower bounded by ϵ, the result in b) follows. □

4.6 Numerical Experiments

In this section, we present a few numerical experiments to verify the theoretical convergence properties of the proposed DRAO-S method.

4.6.1 Implementation Details

The numerical experiments are implemented in MATLAB 2021b and are tested on an Alienware Desktop with a 4.20 GHz Intel Core i7 processor and 16 GB of 2400MHz DDR4 memory. The stepsize of the DRAO-S method are chosen according to Theorem Theorem 4.4, Theorem 4.5 and Theorem 4.6. The implementation details of the proximal mappings are deferred to the Appendix. Parameter tuning is used to achieve better empirical performance. The DRAO-S method is first tested on a few trial stepsizes, each running for only 20 phases. Next, the one achieving the lowest objective value during the trials is selected to run till the desired accuracy, subject to a termination limit of 5000 phases. The trial stepsizes are calculated according to (Equation 4.39), (Equation 4.42), (Equation 4.45) and (Equation 4.48). The trial stepsizes are calculated from conservative estimates of $D_P \geq \|p^0 - p^*\|$ and $R_0 \geq \|x^0 - x^*\|$, and from a few scaled estimates of L_f, M_t, M_A and $\tilde{M}_{A\Pi}$. Specifically, for the smooth linear regression problem (Equation 4.69), the parameters L_f and M_t used for the calculations in (Equation 4.39) and (Equation 4.42) are given by (refer to Subsection subsection 4.6.2 for the definition of H_i)

Parameter	Choices	Conservative Estimate
L_f	$\{\widehat{L_f}, 0.3\widehat{L_f}\}$	$\widehat{L_f} := \max_{i\in[m]} \left\|H_i^\top H_i\right\|$
M_t	$\{\widehat{M_t}, 0.3\widehat{M_t}\}$	$\widehat{M_t} := \|[\nabla f_1(\underline{x}^t); \ldots; \nabla f_m(\underline{x}^t)]\|$

.

So there are four sets of trial stepsizes. For the structured non-smooth two-stage stochastic program, the parameters $\tilde{M}_{A\Pi}$ and M_A used for the calculations in (Equation 4.45) and (Equation 4.48) are given by (refer to Subsection subsection 4.6.2 for the definition of T_i and e_i)

Parameter	Choices	Conservative Estimate
M_A	$\{\widehat{M_A}, 0.3\widehat{M_A}, 0.1\widehat{M_A}\}$	$\widehat{M_A} := \max_{i\in[m]} \|T_i\|$
$\tilde{M}_{A\Pi}$	$\{\widehat{M_{A\Pi}}, 0.3\widehat{M_{A\Pi}}, 0.1\widehat{M_{A\Pi}}\}$	$\widehat{M_{A\Pi}} := \left\|[T_1^\top e_1; \ldots; T_m^\top e_m]\right\|$

.

So there are nine sets of trial stepsizes.

4.6.2 Risk Averse Linear Regression Problem

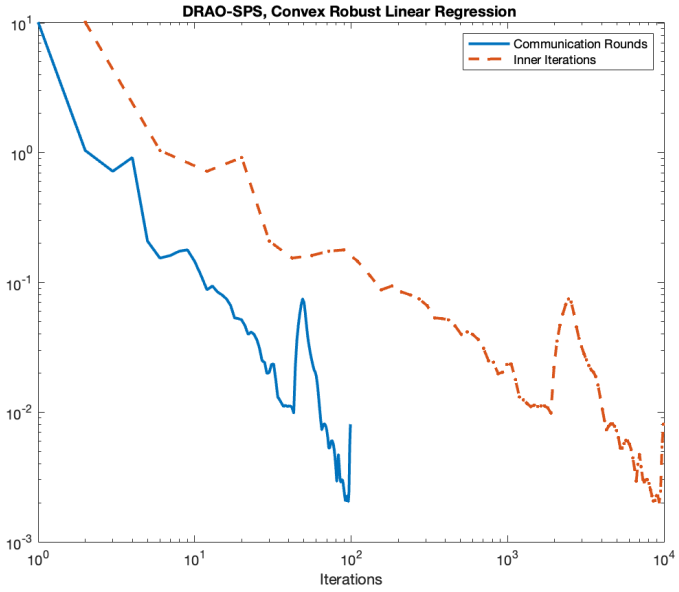


Figure 4.2: Convergence of DRAO-S for a Randomly Generated Robust Linear Regression Problem with $\alpha = 0$

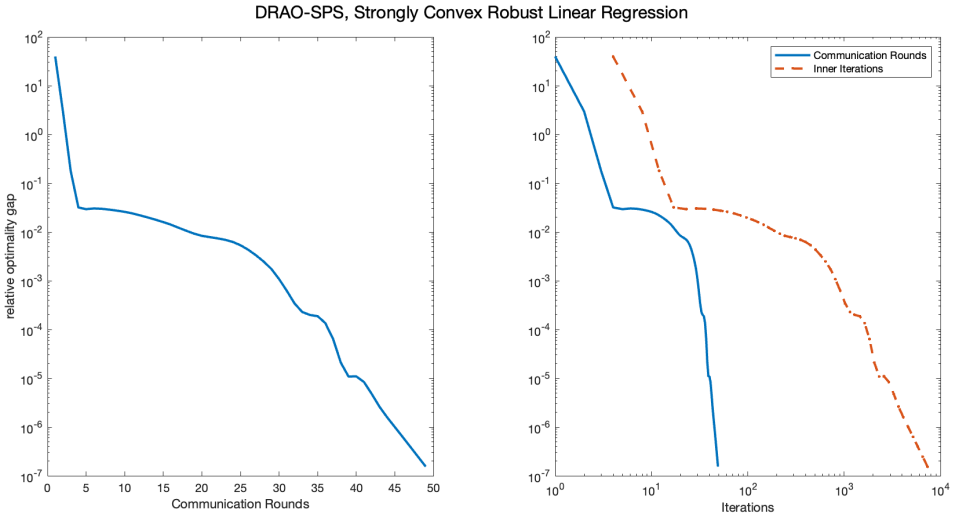


Figure 4.3: Convergence of DRAO-S for a Randomly Generated Robust Linear Regression Problem $\alpha > 0$

Table 4.3: Communications Rounds and P-Projections Required by DRAO-S for Linear Regression under a CV@R Risk

#Scenarios	Opt. Gap	10% Risk		5% Risk		1.25% Risk	
Non-strongly Convex $\alpha = 0$							
		#Comm	#P-proj	#Comm	#P-proj	#Comm	#P-proj
20	10%	6	16	8	50	8	50
	1%	31	529	38	1236	38	1236
	0.1%	76	2858	85	6282	85	6282
50	10%	3	4	6	16	7	40
	1%	16	126	20	199	34	1162
	0.1%	63	1935	67	2206	68	4245
200	10%	3	4	3	4	5	14
	1%	6	19	12	72	23	260
	0.1%	30	516	51	1336	65	2113
Strongly Convex Condition Number $\kappa = 10$							
20	1e-3	32	665	39	1254	39	1254
	1e-4	43	1926	44	2139	44	2139
	1e-5	48	2980	49	3603	49	3603
50	1e-3	19	159	29	485	39	1306
	1e-4	36	854	38	1132	44	2051
	1e-5	41	1538	43	2061	49	3754
200	1e-3	16	54	14	69	29	454
	1e-4	28	205	32	437	42	1777
	1e-5	40	660	44	1357	46	2934

For the smooth case, the following risk-averse linear regression problem of the form (Equation 4.1) is considered:

$$f(x) := \text{CV@R}_\delta(f_1(x), \ldots, f_m(x)) + \tfrac{\alpha}{2}\|x\|^2 \text{ with } f_i(x) := \tfrac{1}{2}\|H_i x - b_i\|^2, X := \mathbb{R}^n. \tag{4.69}$$

Here f_i denotes the loss function associated with the i^{th} dataset. Such a problem is motivated by the need for a single robust model under fairness or risk considerations. For example, the state education department might wish to build a model to help teachers to identify students who need extra help. (H_i, b_i) could represent the data collected in the i^{th} county and the CV@R risk measure could be used to ensure fairness among counties.

In our experiments, we set $n = 40$, and generate matrices $H_i \in \mathbb{R}^{40\times 200}$, and $b_i \in \mathbb{R}^{40}$ randomly. We generate an estimate of f_* by running the bundle level method [91] to an extremely high degree of accuracy. We record the average number of communication rounds and P-projections steps needed, over five randomly generated instances, to achieve the desired relative optimality gap, i.e., $(f(x^t) - f_*)/f_* \leq \epsilon$ under different settings. In particular, the DRAO-S method is tested on problems with different levels of risk and different numbers of computing nodes to understand how the communication and the P-projection complexities vary with D_P and m in practice. The results are presented in Table Table 4.3. For the number of computing nodes m, both the number of P-projections and the number of communication rounds scale well with it. In fact, they seem to decrease slightly when m increases. For D_P, recall that a lower risk level corresponds to a larger ambiguity set P and hence a larger radius D_P (c.f. Subsection subsection 4.1.1). Both the number of P-projections and the number of communication rounds increase with D_P, but the number of communication rounds seems to have a weaker dependence on it. Additionally, typical convergence curves of the DRAO-S method are plotted in Figure 4.2 and Figure 4.3, and they seem to verify the theoretical convergence guarantees. When $\alpha = 0$, Table Table 4.3 and the convergence curve in Figure Figure 4.2 illustrate a communication complexity and a

P-projection complexity on the order of $\mathcal{O}(1/\sqrt{\epsilon})$ and $\mathcal{O}(1/\epsilon)$, respectively. When $\alpha > 0$, the convergence curves in Figure Figure 4.3 and Table Table 4.3 illustrate a communication complexity and a P-projection complexity on the order of $\mathcal{O}(\log(1/\epsilon))$ and $\mathcal{O}(1/\sqrt{\epsilon})$, respectively. Thus, the DRAO-S method can find highly accurate solutions within a small number of communication rounds.

4.6.3 Risk Averse Two-Stage Stochastic Programming

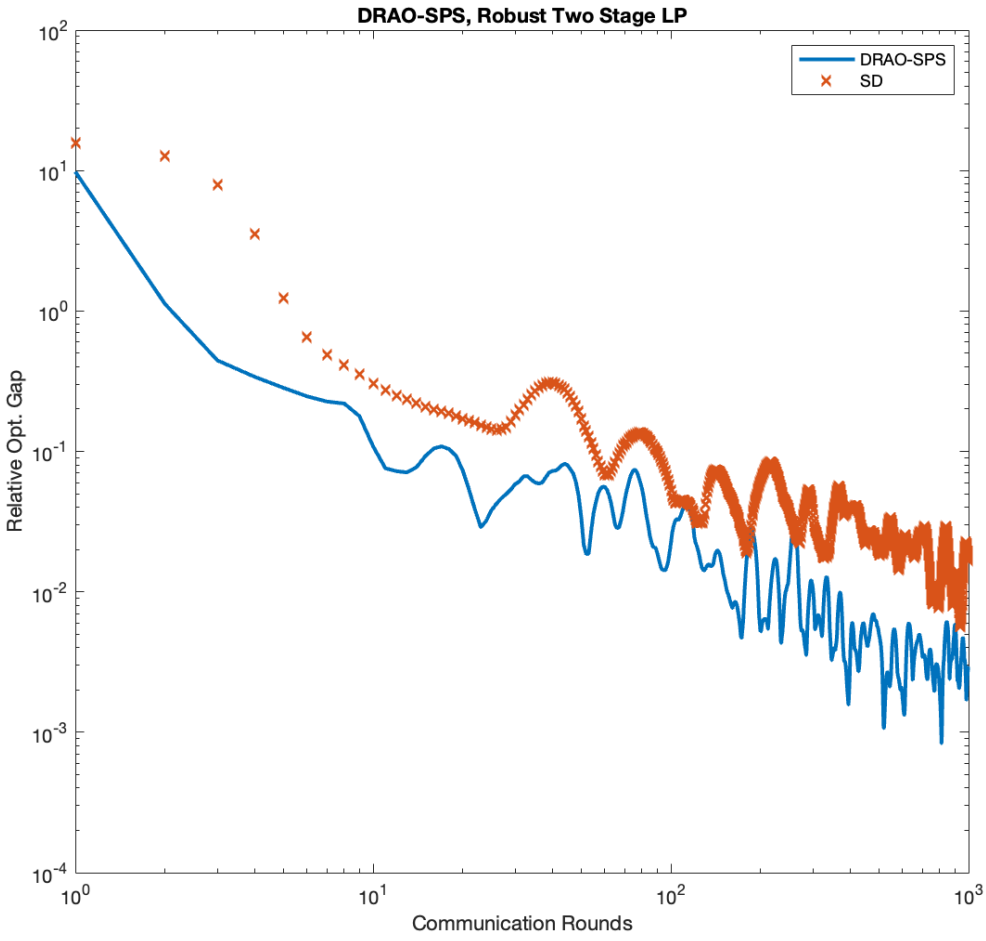


Figure 4.4: Convergence of DRAO-S for a Randomly Two-Stage Linear Problem: $\alpha = 0$, 6 Inner Iterations

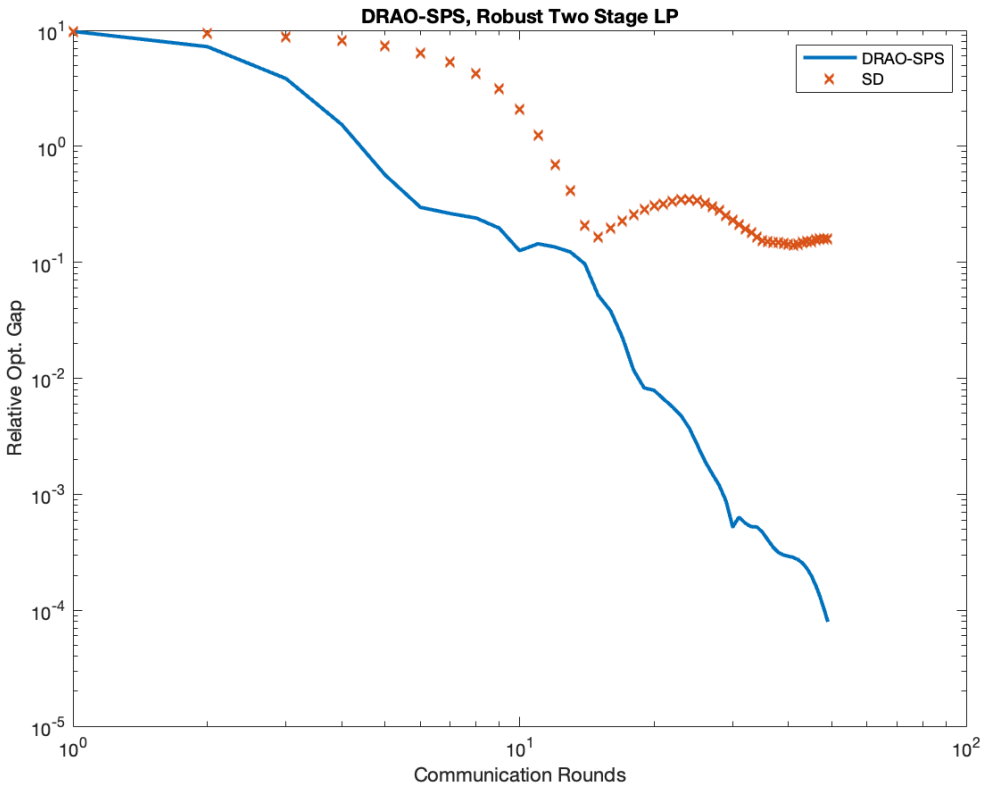


Figure 4.5: Convergence of DRAO-S for a Randomly Two-Stage Linear Problem: $\alpha > 0$, 36 Inner Iterations

For the structured non-smooth case, we compare the DRAO-S method with the SD method [55] using the same risk-averse two-stage stochastic linear programming problem from [55]:

$$\begin{aligned} \min_{x\in\mathbb{R}^n} \quad & c^\intercal x + \text{CV@R}_\delta(g_1(x),\ldots,g_m(x)) + \tfrac{\alpha}{2}\|x\|^2, \\ s.t. \quad & 0 \le x_j \le U \;\forall j \in [n], \\ & g_i(x) := \min\nolimits_{y_i\in\mathbb{R}^l_+}\{y_i^\intercal e_i,\ s.t.\ \ Ry_i \ge d_i - T_i x\}. \end{aligned} \tag{4.70}$$

The problem models the capacity expansion decision of an electricity company. Being the sole provider of electricity, the company has to meet all demand profiles $\{d_i\}$ using a combination of installed capacity, with an availability factor of T_i, and electricity purchased from outside the grid, at a unit cost of e_i. Being risk averse, the company intends to find a

Table 4.4: Communication Rounds Required by Two-Stage Stochastic Program under a CV@R Risk

#Scenarios	Opt. Gap	10% Risk		5% Risk		1.25% Risk	
Non-strongly Convex $\alpha = 0$							
		SD	DRAO-S	SD	DRAO-S	SD	DRAO-S
20	10%	250	**59**	386	**103**	386	**103**
	1%	2328	**420**	2834	**597**	2834	**597**
50	10%	357	**83**	520	**79**	733	**94**
	1%	2699	**511**	2971	**590**	4292	**543**
200	10%	74	**12**	183	**16**	447	**34**
	1%	1614	**187**	3354	**275**	NA	**293**
Strongly Convex $\alpha = 1$							
20	10%	43	**13**	41	**14**	41	**14**
	1%	181	**24**	129	**25**	129	**25**
	0.1%	494	**41**	321	**47**	321	**47**
50	10%	26	**13**	35	**14**	65	**15**
	1%	98	**21**	131	**23**	183	**24**
	0.1%	226	**37**	258	**41**	335	**43**
200	10%	11	**10**	20	**12**	44	**14**
	1%	75	**18**	125	**20**	208	**24**
	1%	320	**25**	508	**31**	438	**41**

NA : Algorithm has not reached specified accuracy after 5000 communication rounds.

decision that keeps the total cost low for roughly $(1 - \delta)$ of all possible scenarios.

In our experiments, we set $n = 40$ and $l = 20$, generate $T_i \in \mathbb{R}^{20\times 40}$, $e_i \in \mathbb{R}^{20}, d_i \in \mathbb{R}^{20}$ and $c \in \mathbb{R}^{40}$ randomly, and choose $R := I_{20,20}$ to be the simple complete recourse matrix. We record the average number of communication rounds required to achieve the desired relative optimality gaps for both methods in Table Table 4.4. Clearly, DRAO-S enjoys significant savings compared to the SD method. The number of communications rounds required by DRAO-S is also less sensitive to the risk level and D_P. Moreover, typical convergence curves are plotted in Figure Figure 4.4 and Figure 4.5. They seem to verify the theoretical communication complexities of DRAO-S on the orders of $\mathcal{O}(1/\epsilon)$ and $\mathcal{O}(1/\sqrt{\epsilon})$, respectively, for the non-strongly convex and the strongly convex problems.

4.6.4 Risk Measure induced by the χ^2 Ambiguity Set

Next, we test these algorithms on a more complicated quadratically constrained set P. Given a radius parameter r, the modified χ^2 probability uncertainty set respect to the empirical probability $[1/m, \ldots, 1/m]$ is given by

$$P_r = \{p \in \mathbb{R}^m_+ : \textstyle\sum_{i=1}^m p_i = 1, \|p - [1/m, \ldots, 1/m]\|^2 \leq r\}.$$

Inspired by the χ^2 test, P_r is useful for distributionally robust optimization (DRO) [92]. We conduct our experiments with the induced risk-measure $\rho(g) = \max_{p \in P_r} \langle p, g \rangle$ on both the linear regression problem (Equation 4.69) and the two-stage stochastic program (Equation 4.70). The average number of communication rounds required to reach the desired sub-optimalities for various levels of r are recorded in Table Table 4.5 and Table 4.6. Since a larger r implies a larger P, the results are consistent with our findings under the CV@R setting.

4.7 Conclusion

This paper introduces the problem of distributed risk-averse optimization. A conceptual DRAO method and a more practical DRAO-S method are proposed. Both of them are able to solve the risk-averse problem with the same communication complexities as those for solving the risk-neutral problem. The optimality of their communication complexities is established with matching lower bounds. And preliminary numerical experiments seem to indicate promising empirical performance for DRAO-S.

In future work, we will attempt to extend our proposed methods to the more general cross-device federated learning setting [69] where f_i's are accessible only via a stochastic first-order oracle and the communication network is unreliable. We will also attempt to study the extension to more complicated risk measures for which p-prox mappings are prohibitively expensive and only gradient evaluations are possible.

Table 4.5: Communications Rounds and P-Projections Required by DRAO-S for Linear Regression under a modified χ^2 Risk Measure

#Scenarios	Opt. Gap	$r = 0.05$		$r = 0.1$		$r = 0.2$	
Non-strongly Convex $\alpha = 0$							
		#Comm	#P-proj	#Comm	#P-proj	#Comm	#P-proj
20	10%	3	4	3	4	3	4
	1%	14	93	17	134	20	188
	0.1%	28	274	40	944	73	2710
50	10%	3	4	3	4	3	4
	1%	8	33	13	83	19	162
	0.1%	21	216	30	524	69	2369
200	10%	3	4	3	4	3	4
	1%	7	22	14	95	19	164
	0.1%	21	207	40	872	70	2426
Strongly Convex Condition Number $\kappa = 10$							
20	1e-3	6	10	9	28	26	235
	1e-4	32	216	33	343	35	605
	1e-5	35	312	36	462	38	754
50	1e-3	6	10	15	67	26	244
	1e-4	29	164	32	323	35	566
	1e-5	34	266	35	409	37	690
200	1e-3	20	68	18	80	25	220
	1e-4	28	163	34	356	36	676
	1e-5	34	280	36	450	39	882

Table 4.6: Communication Rounds Required by Two-Stage Stochastic Program under a modified χ^2 Risk Measure

#Scenarios	Opt. Gap	$r = 0.05$		$r = 0.1$		$r = 0.2$	
Non-strongly Convex $\alpha = 0$							
		SD	DRAO-S	SD	DRAO-S	SD	DRAO-S
20	10%	88	**41**	135	**49**	190	**54**
	1%	703	**292**	1006	**330**	2032	**343**
50	10%	240	**46**	319	**57**	388	**75**
	1%	1543	**313**	2146	**377**	2838	**409**
200	10%	194	**16**	273	**32**	332	**43**
	1%	1747	**270**	2818	**315**	3191	**335**
Strongly Convex $\alpha = 1$							
20	10%	10	**10**	14	**11**	21	**12**
	1%	41	**17**	68	**19**	83	**21**
	0.1%	188	**28**	301	**37**	338	**39**
50	10%	**9**	10	15	**12**	46	**14**
	1%	52	**18**	96	**20**	199	**23**
	0.1%	231	**29**	348	**36**	605	**42**
200	10%	14	**11**	21	**13**	33	**14**
	1%	100	**18**	158	**21**	183	**24**
	1%	556	**30**	771	**36**	671	**37**

4.8 Appendix

Lemma 4.5 *Let $f_i : \mathbb{R}^n \to \mathbb{R}$ be a proper convex closed function and f_i^* be its Fenchel conjugate. The following computations are equivalent for all $\bar{y} \in X, \bar{\pi}_i \in \mathbb{R}^n, \tau > 0$:*

$$\pi_i^t \leftarrow \arg\max_{\pi_i} \langle \bar{y}, \pi_i \rangle - f_i^*(\pi_i) - \tfrac{\tau}{2} \left\| \pi_i - \bar{\pi}_i \right\|^2, \tag{4.71}$$

$$\pi_i^t \leftarrow \bar{\pi}_i + \tfrac{1}{\tau}(\bar{y} - \underline{y}), \text{ where } \underline{y} \leftarrow \arg\min_y f_i(y) + \tfrac{1}{2\tau} \left\| \bar{y} + \tau \bar{\pi}_i - y \right\|^2. \tag{4.72}$$

Proof: Let us fix an $i \in [m]$ and let $\bar{u} := \bar{y} + \tau \bar{\pi}_i$. Let π_i^t be generated according to (Equation 4.72). Consider a Moreau envelop of f_i given by $g(u) = (f_i \square \frac{1}{2\tau} \| \cdot - y \|^2)(u) := \inf_y f_i(y) + \frac{1}{2\tau} \|u - y\|^2$. Since f_i is convex, g is convex and smooth over $\mathbb{R}^n$, thus $\partial g(\bar{u})$ is non-empty and unique.

Next, define $\bar{g}(u) := f_i(\underline{y}) + \frac{1}{2\tau}\|u - \underline{y}\|^2$ such that $g(u) \leq \bar{g}(u)$. Since $\underline{y} := \inf_y f_i(y) + \frac{1}{2\tau}\|\bar{u} - y\|^2$ in (Equation 4.72) implies $g(\bar{u}) = \bar{g}(\bar{u})$, the subgradient of g at $\bar{u}$ must be a subgradient of $\bar{g}$, a dominating function, at $\bar{u}$, i.e.,

$$\partial g(\bar{u}) \subset \partial \bar{g}(\bar{u}) = \{\pi_i^t := \tfrac{1}{\tau}(\bar{u} - \underline{y})\}.$$

Therefore $\pi_i^t = \nabla g(\bar{u})$. Using the infimal convolution identity (c.f. Theorem 4.16 in [10]) $(g)^*(\pi_i) = (f_i \square \frac{1}{2\tau}\|\cdot\|^2)^*(\pi_i) = f_i^*(\pi_i) + \frac{\tau}{2}\|\pi_i\|^2 \ \forall \pi_i$, the equivalence between maximization and sub-gradient evaluation, and the fact $\bar{u} := \bar{y} + \tau\bar{\pi}_i$, we get

$$\begin{aligned}
\pi_i^t \in \partial g(\bar{u}) = \partial(f_i \square \tfrac{1}{2\tau}\|\cdot\|^2)(\bar{u}) &\Leftrightarrow \pi_i^t \in \arg\max_{\pi_i} \langle \bar{u}, \pi_i \rangle - (f_i \square \tfrac{1}{2\tau}\|\cdot\|^2)^*(\pi_i) \\
&\Leftrightarrow \pi_i^t \in \arg\max_{\pi_i \in \Pi_i} \langle \bar{y} + \tau\bar{\pi}_i, \pi_i \rangle - f_i^*(\pi_i) - \tfrac{\tau}{2}\|\pi_i\|^2 \\
&\Leftrightarrow \pi_i^t \in \arg\max_{\pi_i \in \Pi_i} \langle \bar{y}, \pi_i \rangle - f_i^*(\pi_i) - \tfrac{\tau}{2}\|\pi_i - \bar{\pi}_i\|^2 .
\end{aligned}$$

□

4.8.1 Efficient Implementations for Proximal Mappings

Since X is either a box or $\mathbb{R}^n$, the x-prox mappings are implemented with closed-form solutions. The π-prox mappings also admit closed-form solutions. For the linear regression problem in (Equation 4.69), the equivalent primal gradient computation amounts to a matrix-vector multiplication. For the two-stage stochastic program in (Equation 4.70), since the simple complete recourse is assumed [55], Π_i is a box and the projection onto it can implemented by component-wise thresholding.

The p-proximal update are implemented with binary searches and some basic matrix operations. When ρ is a δ-CV@R risk measure, P can expressed as the intersection of an equality constraint and a box constraint [9]. By dualizing the coupling equality constraint,

we arrive at an equivalent two-level optimization formulation for the p^t-prox mapping.

$$\begin{aligned} p^t = \arg\max_p \langle p, g\rangle - \tfrac{1}{2}\left\|p - p^{t-1}\right\|^2 \\ s.t.\ 0 \le p_i \le 1/(m\delta) \\ \textstyle\sum_{i=1}^m p_i = 1 \\ \Leftrightarrow \quad p^t = \min_{\lambda\in\mathbb{R}} \arg\max_p \langle p, g\rangle - \tfrac{1}{2}\left\|p - p^{t-1}\right\|^2 + \lambda(\textstyle\sum_{i=1}^m p_i - 1) \\ s.t.\ 0 \le p_i \le 1/(m\delta). \end{aligned} \tag{4.73}$$

For a fixed λ, the inner solution $p(\lambda)$ can be computed via a component-wise vector thresholding and the optimal λ^t is characterized by the root condition $\sum_{i=1}^m p_i(\lambda^t) - 1 = 0$. Since $p(\lambda)$ is a monotonically non-decreasing function of λ, an accurate approximation to λ^t and hence p^t can be found by a binary search on λ. Next, when ρ is the risk measure induced by the χ^2 ambiguity set, we can dualize the χ^2 constraint to express the p^t-prox mapping equivalently as follows.

$$\begin{aligned} p^t = \arg\max_{p\ge 0} \langle p, g\rangle - \tfrac{1}{2}\left\|p - p^{t-1}\right\|^2 \\ s.t.\ \|p - [1/m, \ldots, 1/m]\|^2 \le r \\ \textstyle\sum_{i=1}^m p_i = 1. \\ \Leftrightarrow \quad p^t = \min_{u\in\mathbb{R}_+} \arg\max_{p\ge 0} \langle p, g\rangle - \tfrac{1}{2}\left\|p - p^{t-1}\right\|^2 + u(\|p - [1/m, \ldots, 1/m]\|^2 - r) \\ s.t.\ \textstyle\sum_{i=1}^m p_i = 1. \end{aligned}$$

For a fixed u, the inner solution $p(u)$ above can be solved similarly to (Equation 4.73). A sufficient optimality condition for u^t is the KKT condition, i.e. either $u^t = 0$, or $u^t > 0$ and $\|p(u^t) - [1/m, \ldots, 1/m]\|^2 - r = 0$. So an accurate u^t and hence p^t can be found by a binary search on u.

CHAPTER 5

SMOOTH FUNCTION CONSTRAINED OPTIMIZATION

text of this chapter goes here

5.1 Background and Our Contribution

Consider convex smooth function-constrained optimization of the form

$$\begin{aligned} &\min_{x\in X} \{F(x) := f(x) + u(x)\} \\ &s.t.\ g(x) \leq 0, \end{aligned} \tag{5.1}$$

where both $f : \mathbb{R}^n \to \mathbb{R}$ and $g := [g_1, g_2, \ldots, g_m]^\top : \mathbb{R}^n \to \mathbb{R}^m$ are convex and differentiable with Lipschitz continuous gradients, and the domain X is convex and closed. The regularization function $u(x)$ is assumed to be both simple [93] and α-strongly convex for some $\alpha \geq 0$. Problem of this type finds a wide range of applications in, for example, the Neyman-Pearson classification problem [94], the fairness-constrained classification [95], and the risk-constrained portfolio optimization [96]. Since the dimensions n and m are large in many applications, we focus on first-order methods to find an approximation solution. Specifically, (f, g) is assumed to be accessible only via a black-box first-order oracle, which returns $(f(x), g(x); \nabla f(x), \nabla g(x))$ when queried at some $x \in \mathbb{R}^n$, and the goal is to find an $(\epsilon; \epsilon/c)$-optimal solution (or, in short, an ϵ-solution):

$$F(x^N) - F(x^*) \leq \epsilon \text{ and } \left\|[g(x^N)]_+\right\| \leq \epsilon/c, \tag{5.2}$$

where the scaling constant $c \geq 1$ represents the modeler's preference for constraint violation relative to sub-optimality.

Table 5.1: Ideal Complexity for Solving (Equation 5.1)

Case	Oracle Complexity	Computation Complexity
Non-strongly Convex $\alpha = 0$	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\epsilon)$
Strongly Convex $\alpha > 0$	$\mathcal{O}(\log(1/\epsilon))$	$\mathcal{O}(1/\sqrt{\epsilon})$

This paper intends to develop fast methods and to understand the requisite computation cost. To ensure practical efficiency, special attention is paid to simple methods involving only oracle evaluations, projections onto X, and basic vector operations like matrix-vector multiplication. In particular, we consider consider two kinds of cost involved in solving (Equation 5.1): a) the oracle complexity, i.e., the total number of queries to the first-order oracle, and b) the computation complexity, i.e., the total number of matrix-vector multiplications (each costing at most $O(mn)$ FLOPs). During implementation, these complexities translate to different computation burdens, so the dominating cost depends on the context. For instance, if the constraint g is complicated, say a large finite-sum function in the fairness-constrained problem, the gradient evaluation could be the bottleneck. On the other hand, if both n and m are large and g is an affine function, the matrix-vector multiplication might be the bottleneck. Thus the ideal optimization method should excel in both directions.

Since convex optimization has been studied extensively, e.g. [1], we can conjecture the best possible complexities for (Equation 5.1) by reducing it to simpler cases for which the optimal results are available. Recall from [2, 33], the number of oracle evaluations required to find an ϵ-optimal solution to a smooth problem without function constraints is $\Theta(1/\sqrt{\epsilon})$ in general, and $\Theta(\sqrt{\kappa}\log(1/\epsilon))$ if the problem is also strongly convex, where κ denotes the condition number. One way to reduce (Equation 5.1) is to consider the Lagrange dual formulation:

$$\min_{x\in X}\max_{\lambda\in\mathbb{R}^m_+}\{\mathcal{L}(x;\lambda) := f(x) + u(x) + \lambda^\top(g(x))\}. \tag{5.3}$$

If the multiplier λ is fixed to the optimal dual multiplier λ^*, the optimal Lagrangian function $\mathcal{L}(x;\lambda^*)$ is smooth and does not have any function constraint. Since some minimizer

to $\mathcal{L}(x;\lambda^*)$ is the solution to (Equation 5.1), in the ideal case, optimizing (Equation 5.1) should have the same oracle complexity as optimizing $\mathcal{L}(x;\lambda^*)$, i.e., $\mathcal{O}(1/\sqrt{\epsilon})$ for the non-strongly case ($\alpha = 0$) and $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$ for the strongly convex case ($\alpha > 0$), where κ is the condition number associated with $\mathcal{L}(x;\lambda^*)$. On the other hand, inspired by [1] and [97], Xu and Ouyang constructed novel large-scale linearly constrained quadratic programs in [86] to show the tight computation complexity for the linearly constrained smooth problem is $\Theta(1/\epsilon)$ if $\alpha = 0$, and $\Theta(1/\sqrt{\epsilon})$ if $\alpha > 0$. Since the nonlinear constraint function includes the affine function as a special case, the computation complexity for (Equation 5.1) should be at least as expensive as the affine case. These conjectured complexities, summarized in Table Table 5.1, lead naturally to the research question:

Can we solve function-constrained problems with the same oracle complexities as those without function constraints, while maintaining the same computation complexities as solving linearly constrained problems?

However, despite much research effort on the subject from different directions, the question remains open. Broadly speaking, the current results can be grouped according to reformulations (see the summary in Table Table 5.2). The methods based on the Lagrangian formulation (Equation 5.3) are usually simple to implement and have a low per-iteration cost. For instance, the ConEx method in [98] and the APD method in [99] are single-loop algorithms with optimal computation complexities, but their oracle complexities are worse than the ideal ones by an order of magnitude. The current best method based on the augmented Lagrangian reformulation [100], a three-loop algorithm constructed from the inexact augmented Lagrangian method, the ellipsoid method, and the accelerated gradient descent (AGD) method, is closer to the ideal oracle complexities. However, the proposed method in [100] scales poorly with the number of constraints. In fact, the method is advantageous to the APD method only when $m \leq 5$ in the numerical experiments [100].

Another line of research [101, 102], called the level-set method, reformulates (Equation 5.1) to a root finding problem associated with a certain mini-max problem. These

Table 5.2: Complexities for Smooth Constrained Optimization (Equation 5.1)

	Strongly Convex $\alpha > 0$		Convex $\alpha = 0$		Strong Oracle
Method	Oracle	Computation	Oracle	Computation	Assumption
Level Set[1][102]			$\mathcal{O}(\frac{1}{\sqrt{\epsilon}}\log(\frac{1}{\epsilon}))$		QCQP Oracle
Level Set[1][101]			$\mathcal{O}(1/\epsilon)$	$\mathcal{O}(1/\epsilon)$	
iALM [100]	$\mathcal{O}(\frac{m}{\sqrt{\epsilon}}\log^3(\frac{1}{\epsilon}))$	$\mathcal{O}(\frac{m}{\sqrt{\epsilon}}\log^3(\frac{1}{\epsilon}))$	$\mathcal{O}(m\log^3(\frac{1}{\epsilon}))$	$\mathcal{O}(m\log^3(\frac{1}{\epsilon}))$	$m = \mathcal{O}(1)$
APD [99]	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\epsilon)$	$\mathcal{O}(1/\epsilon)$	
ConEx [98]	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\epsilon)$	$\mathcal{O}(1/\epsilon)$	
ACGD [*]	$\mathcal{O}(\log(1/\epsilon))$		$\mathcal{O}(1/\sqrt{\epsilon})$		QP Oracle
ACGD-S [*]	$\mathcal{O}(\log(1/\epsilon))$	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\sqrt{\epsilon})$	$\mathcal{O}(1/\epsilon)$	

[1] If, in addition to f, all constraint functions, $g_1, \ldots, g_m$, are α-strongly convex, the oracle complexity is $\mathcal{O}(\log(1/\epsilon))$ for the level-set method in [102], and is $\mathcal{O}(1/\sqrt{\epsilon})$ for [101].

methods typically require all constraint functions to share the same strong convexity modulus as the objective function. Such an assumption can be quite restrictive because it is violated when there exists one affine constraint. Assuming the uniform strong-convexity condition holds, the method proposed by Nesterov in Section 2.3.5 of [102] can achieve an $\mathcal{O}(\log(1/\epsilon))$ oracle complexity when $\alpha > 0$. However, the method might be too computationally demanding to implement for the large-scale setting because it requires the exact solution to a quadratic program (QP) in each iteration, and the exact solution to a quadratic-constrained quadratic program (QCQP) from time to time. Lin et al. relaxes the expensive computation oracle assumption in [101], but the oracle complexity also becomes worse by an order of magnitude. To sum up, there exist two major deficiencies: a) current methods fail to match the ideal oracle complexity, and b) methods that are close to the ideal oracle complexity are impractical in the large-scale setting.

In this paper, we provide an affirmative answer to the research question by proposing efficient algorithms to achieve the ideal complexities. An important observation for our development is that the ideal oracle complexity and the ideal computation complexity, shown in Table Table 5.1, have different orders of magnitudes. This explains why the single loop algorithms which carry out $\mathcal{O}(1)$ matrix-vector operations per oracle evaluation, for example, the APD method [99] and the ConEx method [98], can only achieve the ideal computation complexity, but not the ideal oracle complexity. This also explains why the

more complicated algorithms with better oracle complexities, for example, the level set method in [102] and the iALM method in [100], require stronger computation oracles than basic matrix-vector operations. Following this observation, we first ask if the ideal oracle complexity in Table Table 5.1 is attainable with any strong computation oracle. The question leads us to the oracle-efficient Accelerated Constrained Gradient Descent (ACGD).

The ACGD method is based on the Lagrangian formulation (Equation 5.3). The method can be motivated by how Nesterov adapts the AGD method to solve a minimax problem of the form $\min_x \max_i\{f_1(x), ..., f_m(x)\}$ in Section 2.3 of [102]. Since the Lagrangian function is also a minimax problem, it also satisfies the following max-type smoothness condition (see Lemma 2.3.1 in [102]):

$$\max_{\lambda\in\mathbb{R}^m_+} \mathcal{L}(x,\lambda) - \max_{\lambda\in\mathbb{R}^m_+}\{l_f(x;\bar{x}) + \textstyle\sum_{i=1}^m \lambda_i l_g(x;\bar{x})\} \leq \frac{L(\mathbb{R}^m_+)}{2}\left\|x-\bar{x}\right\|^2, \tag{5.4}$$

where l_f and l_g denote the linearization of f and g at $\bar{x}$, and the aggregate smoothness constant $L(\mathbb{R}^m_+)$ denotes the maximum, over $\lambda \in \mathbb{R}^m_+$, of the Lipschitz smoothness constants of $\mathcal{L}(x;\lambda)$ for a fixed λ. To obtain the desired oracle complexity, Nesterov proposes to modify the AGD method by replacing the descent step with a max-type descent step (see (2.3.12) in [102] or the DRAO method in [59]), which in our case becomes

$$x^t \leftarrow \arg\min_{x\in X}\max_{\lambda\in\mathbb{R}^m_+}\ l_f(x;\underline{x}^t) + \sum_{i=1}^m \lambda_i(l_{g_i}(x;\underline{x}^t)) + u(x) + \tfrac{\eta_t}{2}\left\|x-x^{t-1}\right\|^2, \tag{5.5}$$

where the linearization center $\underline{x}^t$ is a certain convex combination of $\{x^0,\ldots,x^{t-1}\}$. Since the maximization of λ is over $\mathbb{R}^m_+$, (Equation 5.5) is equivalent to the following quadratic program,

$$x^t \leftarrow \min_{x\in X}\{l_f(x;\underline{x}^t)\ + u(x) + \tfrac{\eta_t}{2}\left\|x-x^{t-1}\right\|^2 \text{ s.t. } l_{g_i}(x;\underline{x}^t) \leq 0\ \forall i \in [m]\}.$$

We call it the constrained descent step and the so-modified AGD method the Accelerated

Constrained Gradient Descent method. Just like the AGD method, the stepsize parameter η_t should be selected to be proportional to the aggregate smoothness constant $L(\mathbb{R}^m_+)$. Note, however, that one difficulty with this approach is $L(\mathbb{R}^m_+) = \infty$, so η_t is infinite and the convergence rate is $\mathcal{O}(\infty/N^2)$, i.e., the method for may not even converge. By taking a primal-dual perspective, we show η_t only needs to be proportional to the smoothness constant of $\mathcal{L}(x, \lambda^*)$, denoted by $L(\lambda^*)$. As a result, η_t remains finite and the oracle complexity of the proposed ACGD method almost matches the optimal oracle complexity for optimizing $\mathcal{L}(x; \lambda^*)$.

We also investigate the optimality of the proposed ACGD method by constructing new lower oracle complexity bounds. In the strongly convex case with $\alpha > 0$, the lower bound shows the oracle complexity bound of the ACGD method to be unimprovable for all problem parameters. In the non-strongly case with $\alpha = 0$, the lower bound shows the oracle complexity bound of the ACGD method to be tight up to a factor of $\mathcal{O}(\sqrt{\|\lambda^*\|/c}+1)$, i.e., the ACGD method has tight oracle complexity with respect to all problem parameters if $c \geq \|\lambda^*\|$. If λ^* is interpreted as the shadow price of constraint violation around x^*, such a choice of c ensures that the increase in objective value incurred by moving an ϵ/c-feasible solution to feasibility is roughly $\mathcal{O}(\epsilon)$.

To enhance its efficiency for the large-scale setting with many constraints, we use the sliding technique [87, 59] to extend the ACGD method to the ACGD with Sliding (ACGD-S) method. Given a linearization center $\underline{x}^t$, instead of solving (Equation 5.5) to optimality in each iteration, the ACGD-S method solves the bilinear saddle point inexactly by performing only a finite number of λ-prox mappings and x-prox mappings (see (Equation 5.7)), the most expensive operation during which is matrix-vector multiplication. Particularly, after the t^{th} oracle evaluation, i.e., $\{\nabla g(\underline{x}^t), \nabla f(\underline{x}^t); f(\underline{x}^t), g(\underline{x}^t)\}$, the x-prox mapping and λ-prox mapping are repeated only $\mathcal{O}(t)$ times if $\alpha = 0$, and $\mathcal{O}(\sqrt{\theta^t})$ times for some $\theta > 1$ if $\alpha > 0$. The repetitive updating of prox-mappings and judicious computation of gradients allow the ACGD-S to achieve the optimal computation complexity (matching

the lower bound for linearly constrained problem [86]), while maintaining the same oracle complexity as the ACGD method. Therefore the ACGD-S method provides an almost complete characterization of both the computation and the oracle complexity for solving a smooth function-constrained problem. Moreover, the intricate step-size choice to achieve both the optimal $\mathcal{O}(\log(1/\epsilon))$ oracle evaluations (outer loops) and the optimal $\mathcal{O}(1/\sqrt{\epsilon})$ matrix-vector multiplications (inner iterations) appears to be new to the sliding technique [87, 88, 59], so it is of independent interest.

One potential concern for implementing ACGD and ACGD-S is that their stepsize selection depends on the Lipschitz smoothness constant of $\mathcal{L}(x;\lambda^*)$ (see (Equation 5.3)), which could be hard to estimate in practice. To address this issue, we proposed a binary guess-and-check scheme to search for the elusive problem parameter. Then, given the strong convexity parameter α, under the assumption X is bounded and its radius D_X is known, both the ACGD and ACGD-S methods can be implemented without the knowledge of any other problem parameters to find an $(\epsilon;\epsilon/c)$-solution with the same oracle and computation complexities as those listed in Table Table 5.1.

The rest of the is organized as follows. Section section 5.2 proposes the ACGD method, and Section section 5.3 develops matching lower bounds. Section section 5.4 extends it to a computationally efficient ACGD-S method, and some concluding remarks are made in Section section 5.5.

5.1.1 Notations & Assumptions

The following assumptions and notations will be used throughout the paper.

- The set of optimal solutions to (Equation 5.1), X^*, is nonempty, and x^* is an arbitrary optimal solution. F_* denotes the optimal objective, $F(x^*)$.
- The Fenchel conjugate (see [10]) of a convex function g(x) is defined as $g^*(\pi) := \max_{x\in\mathbb{R}^n}\langle x,\pi\rangle - g(x)$.

- U_h denotes the Bregman distance function (see [10]) generated by a convex function h, i.e., $U_h(\pi;\bar{\pi}) := h(\pi) - h(\bar{\pi}) - \langle h'(\bar{\pi}), \pi - \bar{\pi}\rangle$ where $h'(\bar{\pi})$ is some fixed sub-gradient in $\partial h(\bar{\pi})$. If g is vector-valued, i.e., $g(\nu) := [g_1(\nu), g_2(\nu), \ldots, g_m(\nu)]$, U_g is vector-valued with its i^{th} component being the Bregman distance function generated by g_i, namely, U_{g_i}.

- We refer to the following computation as either a prox-mapping or a projection:

$$\hat{w} \leftarrow \arg\min_{w \in W} \langle y, w\rangle + h(w) + \tau U(w;\bar{w}), \tag{5.6}$$

  where the vector y represents some "descent direction" (the gradient for example), and $h(w)$ is a simple convex function [28]. U is the Bregman distance function, $\bar{w}$ is a prox center, and τ is a stepsize parameter. Together they ensure the output $\hat{w}$ is close to $\bar{w}$. In particular, the following will be referred to as the X-projection:

$$\hat{x} \leftarrow \arg\min_{x \in X} \langle y, x\rangle + u(x) + \tfrac{\tau}{2}\left\|x - \bar{x}\right\|^2. \tag{5.7}$$

5.2 The Accelerated Constrained Gradient Descent Method

We present in this section the ACGD method. Specifically, Subsection subsection 5.2.1 introduces a novel primal-dual perspective to motivate the ACGD method, and Subsection subsection 5.2.2 presents the convergence results. Subsection subsection 5.2.3 proposes the guess-and-check scheme to look for the problem parameter $L(\lambda^*)$, and Subsection subsection 5.2.4 contains the detailed proofs of the convergence results.

5.2.1 The ACGD method

This subsection presents the primal-dual perspective to motivate the ACGD method. Such a perspective is important for understanding the finite stepsize parameter η_t discussed in Section section 5.1. Specifically, we first introduce a novel nested Lagrangian function

which reformulates (Equation 5.1) as a $\min - \max - \max$ trilinear saddle point problem. To search for the saddle point, we propose a primal-dual type method similar to [59, 55]. The ACGD method then simply follows from rewriting the proposed method in the primal form.

First, we need to assume the existence of a KKT point to the Lagrangian function defined in (Equation 5.3) throughout this paper:

Assumption 4 *There exists a* $\lambda^* \in \mathbb{R}^m_+$ *and* $x^* \in X$ *such that* $\nabla f(x^*) + \sum_{i=1}^m \lambda_i^* \nabla g_i(x^*) + u' \in N_X(x^*)$ *for some* $u' \in \partial u(x^*)$, $g(x^*) \leq 0$, *and* $\lambda_i^* g_i(x^*) = 0 \ \forall i$, *where* N_X *denotes the normal cone to* X.

Note x^* and λ^* could be interpreted as an arbitrary element in X^* and $\Lambda^* := \arg\max_{\lambda \in \mathbb{R}^m_+} \min_{x \in X} \mathcal{L}(x; \lambda)$ for the rest of the paper, because any $(\bar{x}, \bar{\lambda})$ with $\bar{x} \in X^*$ and $\bar{\lambda} \in \Lambda^*$ also constitutes a KKT point (see Proposition 3.4.1 in [103]).

To motivate the nested Lagrangian function, consider the simplified case where the optimal dual multiplier λ^* is known. Fix λ to λ^*, an optimal solution x^* can be found by solving the following simplified problem under certain regularity conditions:

$$\min_{x \in X} f(x) + \langle \lambda^*, g(x) \rangle + \ u(x). \tag{5.8}$$

One useful framework for designing an optimal algorithm for the constrained problem in (Equation 5.8) is to consider a bilinear saddle-point reformulation

$$\min_{x \in X} \max_{\nu \in V, \pi \in \Pi} \{ \langle x, \pi \rangle - f^*(\pi) + u(x) + \langle \lambda^*, \nu x - g^*(\nu) \rangle \}, \tag{5.9}$$

where f^* is the Fenchel conjugate function to f and Π is its domain, namely, $\{\pi \in \mathbb{R}^n : f^*(\pi) < \infty\}$, and $g^* := [g_1^*, \ldots, g_m^*]$ is the vector-valued Fenchel conjugate to g and V is its domain, namely, $\{v \in \mathbb{R}^{m \times n} : g_i^*(v_i) < \infty \ \forall i \in [m]\}$. Moreover, since the dual

variables π and ν are associated with a common primal variable x, it is sometimes helpful consider the following joint domain $[V, \Pi]$:

$$[V, \Pi] = \{(\nabla g(x), \nabla f(x)) : x \in \mathbb{R}^n\}. \tag{5.10}$$

However, λ^* is unknown in practice. So we propose to consider the following nested Lagrangian reformulation which combines (Equation 5.3) and (Equation 5.9):

$$\min_{x \in X} \max_{\lambda \in \mathbb{R}^m_+, \pi \in \Pi, \nu \in V} \{\mathcal{L}(x; \lambda, \pi, \nu) := \langle x, \pi \rangle - f^*(\pi) + u(x) + \langle \lambda, \nu x - g^*(\nu) \rangle\}, \tag{5.11}$$

where f^*, g^*, Π and V are defined in the same way as (Equation 5.9). Notice a common notation $\mathcal{L}$ is used for the nested Lagrangian and the ordinary Lagrangian, but the exact meaning should be clear from the context. Let Z denote the joint domain, $Z := X \times \mathbb{R}^m_+ \times \Pi \times V$ and $z^* := (x^*; \lambda^*, \nu^* := \nabla g(x^*), \pi^* := \nabla f(x^*))$. It is not hard to see that z^* is saddle point to (Equation 5.11), and a useful criterion to measure the optimality of an iterate $z^t = (x^t; \lambda^t, \nu^t, p_i^t) \in Z$ is to compare it to some reference point $z \in Z$ in the following gap function:

$$Q(z^t; z) := \mathcal{L}(x^t; \lambda, \nu, \pi) - \mathcal{L}(x; \lambda^t, \nu^t, p_i^t). \tag{5.12}$$

Indeed, the saddle point z^* satisfies $Q(z^*; z) \leq 0 \; \forall z \in Z$.

A crucial observation for our development is that for the convergence to an ϵ-solution, it is sufficient to consider only the reference λ's inside a certain bounded set rather than the positive orthant. The next lemma shows the Q function still provides upper bounds for both the feasibility violation and the optimality gap.

Lemma 5.1 *Let $z^t = (x^t; \lambda^t, \nu^t, p_i^t) \in Z$ be given and let Λ_r denote a certain set of reference λ's,*

$$\Lambda_r = \{0\} \cup \{\lambda^* + \lambda : \lambda \in B_r(0) \cap \mathbb{R}^m_+\}. \tag{5.13}$$

If $\max_{\lambda\in\Lambda_r,(\nu,\pi)\in[V,\Pi]} Q(z^t;(x^*;\lambda,\nu,\pi)) \leq \epsilon$, *we have* $F(x^t) - F(x^*) \leq \epsilon$ *and* $\|[g(x^t)]_+\| \leq \epsilon/r$.

Proof: Consider Part a). Fixing $\hat{p}_i = \nabla f(x^t)$ and $\hat{\nu} = \nabla g(x^t)$, the given condition implies that $Q(z^t;(x^*;\lambda,\hat{\nu},\hat{p}_i)) \leq \epsilon \; \forall \lambda \in \Lambda_r$. It then follows from the conjugate duality relationship (see Section 4.1 and 4.2 of [10]) that

$$f(x^t) + \langle \lambda, g(x^t)\rangle + u(x^t) - [f(x^*) + \langle \lambda^t, g(x^*)\rangle + u(x^*)] \leq \epsilon \; \forall \lambda \in \Lambda_r.$$

Since x^* is feasible, i.e., $g(x^*) \leq 0$, we have $\langle \lambda^t, g(x^*)\rangle \leq 0$. Then taking $\lambda = 0$ leads to

$$f(x^t) + u(x^t) - f(x^*) - u(x^*) \leq f(x^t) + u(x^t) \; - [f(x^*) + \langle \lambda^t, g(x^*)\rangle + u(x^*)] \leq \epsilon.$$

Next since $(x^*;\lambda^*)$ is a saddle point to (Equation 5.3), we have $0 \leq F(x^t) + \langle \lambda^*, g(x^t)\rangle - [F(x^*) + \langle \lambda^t, g(x^*)\rangle]$. Setting $\hat{\lambda} = \lambda^* + r[g(x^t)]_+ / \|[g(x^t)]_+\| \in \Lambda_r$, we get

$$\begin{aligned} r\left\|[g(x^t)]_+\right\| &\leq F(x^t) + \; \langle \lambda^* + r[g(x^t)]_+ / \left\|[g(x^t)]_+\right\|, g(x^t)\rangle - [F(x^*) + \langle \lambda^t, g(x^*)\rangle] \\ &\leq Q(z^t;(x^*;\hat{\lambda},\hat{\nu},\hat{p}_i)) \leq \epsilon. \end{aligned}$$

□ We remark that a lower bound to $F(x^t) - F_*$ can also be derived in similar fashion, see [104, 105].

Now, let us move on to consider minimizing Q to find a saddle point to (Equation 5.11). An essential feature of the nested Lagrangian function (Equation 5.11) is the trilinear term $\langle \lambda, \nu x - g^*(\nu)\rangle$. The problem cannot be simplified to a $\min - \max$ saddle point problem by combining the ν and λ into a single dual block, because their joint maximization is difficult to compute. Similar problems have been studied by Lan and Zhang in [59, 55, 83] and the key to handle the $\min - \max - \max$ trilinear structure is to decompose the Q function into the sub-gap functions and optimize each sub-gap function sequentially. Specifically,

the following decomposition of (Equation 5.12) into sub-gap functions associated with x, λ, ν and π is useful.

$$Q(z^t; z) = Q_x(z^t; z) + Q_\lambda(z^t; z) + Q_\pi(z^t; z) + Q_\nu(z^t; z) \tag{5.14}$$

where

$$Q_\pi(z^t; z) := \mathcal{L}(x^t; \lambda, \nu, \pi) - \mathcal{L}(x^t; \lambda, \nu, p_i^t) = \langle x^t, \pi\rangle - f^*(\pi)\boxed{-[\langle x^t, p_i^t\rangle - f^*(p_i^t)]}, \tag{5.15a}$$

$$Q_\nu(z^t; z) := \mathcal{L}(x^t; \lambda, \nu, p_i^t) - \mathcal{L}(x^t; \lambda, \nu^t, p_i^t) = \textstyle\sum_{i=1}^m \lambda_i(\langle \nu_i, x^t\rangle - g_i^*(\nu_i)] \\ \boxed{-\textstyle\sum_{i=1}^m \lambda_i[\langle \nu_i^t, x^t\rangle - g_i^*(\nu_i^t]}), \tag{5.15b}$$

$$Q_\lambda(z^t; z) := \mathcal{L}(x^t; \lambda, \nu^t, p_i^t) - \mathcal{L}(x^t; \lambda^t, \nu^t, p_i^t) = \langle \lambda, \nu^t x^t - g^*(\nu^t)\rangle \\ \boxed{-\langle \lambda^t, \nu^t x^t - g^*(\nu^t)\rangle}, \tag{5.15c}$$

$$Q_x(z^t; z) := \mathcal{L}(x^t; \lambda^t, \nu^t, p_i^t) - \mathcal{L}(x; \lambda^t, \nu^t, p_i^t) = \boxed{\langle p_i^t + \textstyle\sum_{i=1}^m \lambda_i^t \nu_i^t, x^t\rangle + u(x^t)} \\ - \langle p_i^t + \textstyle\sum_{i=1}^m \lambda_i^t \nu_i^t, x\rangle - u(x). \tag{5.15d}$$

Similar to the DRAO method in [59], we propose to reduce the boxed terms associated with Q_ν and Q_π, and Q_x and Q_λ using the following sequence of prox-mappings in each iteration:

$$\begin{aligned}
\tilde{x}^t &\leftarrow x^{t-1} + \theta_t(x^{t-1} - x^{t-2});\\
p_i^t &\leftarrow \underset{\pi\ \in \Pi}{\arg\max} \langle \pi, \tilde{x}^t\rangle - f^*(\pi) - \tau_t U_{f^*}(\pi; p_i^{t-1});\\
\nu_i^t &\leftarrow \underset{\nu_i \in V_i}{\arg\max} \langle \nu_i, \tilde{x}^t\rangle - g_i^*(\nu_i) - \tau_t U_{g_i^*}(\nu_i; \nu_i^{t-1})\ \forall i \in [m];\\
(x^t, \lambda^t) &\leftarrow \arg\min_{x\in X} \max_{\lambda \in \mathbb{R}_+^m} \langle p_i^t, x\rangle + \langle \lambda, \nu^t x - g^*(\nu^t)\rangle + u(x) + \eta_t \left\|x - x^{t-1}\right\|^2/2.
\end{aligned} \tag{5.16}$$

In the above listing, θ_t, τ_t and η_t are non-negative stepsize parameters, and $U_{g_i^*}$ and U_{f^*} are Bregman distance functions generated by g_i^* and f^* respectively. Particularly, with the

yet available x^t replaced by some proxy $\tilde{x}^t$, the above p_i^t update corresponds to the minimization of the variable p_i^t in (Equation 5.15a) subject to a prox term $\tau_t U_{f^*}(\pi; p_i^{t-1})$. Similarly, the ν^t update corresponds to the minimization of the variable ν^t in (Equation 5.15b) subject to a prox term $\tau_t \sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^{t-1})$. Since the summation weights $\{\lambda_i\}$ are non-negative and the maximizations are separable, the ν^t update is written equivalently as individual ν_i^t updates in (Equation 5.16). Finally, since both the maximization of λ^t in (Equation 5.15c) and the minimization of x^t in (Equation 5.15d) do not require any oracle information related to either f or g, we evaluate them simultaneously subject to a prox term $\eta_t \left\|x - x^{t-1}\right\|^2 /2$. This leads to the joint (x^t, λ^t) update in (Equation 5.16). We remark $\tilde{x}^t$, the proxy for x^t, is chosen as the momentum extrapolation from x^{t-1}. Such a choice helps us to achieve acceleration (e.g. see Section 3.4 in [28] for the connection).

The implementable version of (Equation 5.16) is shown in Algorithm Algorithm 11. It employs two additional simplifications. First, we initialize the dual variables to some gradients, i.e., $p_i^0 = \nabla f(x^0)$ and $\nu^0 = \nabla g(x^0)$. With U_{g^*} and U_{f^*} selected as prox-functions, we can show recursively that p_i^t and ν^t in (Equation 5.16) are same as the gradients at some averaged point (see Lemma 2 in [83]). Thus the ν_i^t and p_i^t computation in (Equation 5.16) simplifies to gradient evaluations in Line 4 of Algorithm Algorithm 11. Second, the (x^t, λ^t)-saddle point problem in (Equation 5.16) is formulated as a linearly constrained quadratic program in Line 5 of Algorithm Algorithm 11. Since $\nu^t = \nabla g(\underline{x}^t)$ in Algorithm Algorithm 11 implies the relation $g_i(\underline{x}^t) + g_i^*(\nu_i^t) = \langle \underline{x}^t, \nu_i^t \rangle$ (see Theorem 4.20 in [10]), we have $g^*(\nu^t) = \nu^t \underline{x}^t - g(\underline{x}^t)$. Interestingly, other than the additional linear constraint associated with g in the descent step (Line 5), Algorithm Algorithm 11 is the same as Nesterov's AGD method [3]. Therefore we name it the Accelerated *Constrained* Gradient Descent method.

Algorithm 11 **A**ccelerated **C**onstrained **G**radient **D**escent Method

Input: $x^{-1} = \underline{x}^0 = x^0 \in X,$ stepsizes $\{\theta_t\}$, $\{\eta_t\}$, $\{\tau_t\}$, and weights $\{\omega_t\}$.

1: Set $p_i^0 = \nabla f(x^0)$, $\nu^t = \nabla g(x^0)$.

2: **for** $t = 1, 2, 3...N$ **do**

3: Set $\underline{x}^t \leftarrow (\tau_t \underline{x}^{t-1} + \tilde{x}^t)/(1+\tau_t)$ where $\tilde{x}^t = x^{t-1} + \theta_t(x^{t-1} - x^{t-2})$.

4: Set $p_i^t \leftarrow \nabla f(\underline{x}^t)$ and $\nu^t \leftarrow \nabla g(\underline{x}^t)$.

5: Solve $x^t \leftarrow \arg\min_{x \in X}\{\langle p_i^t, x\rangle + u(x) + \eta_t \left\|x - x^{t-1}\right\|^2/2 \; s.t. \; \nu^t(x - \underline{x}^t) + g(\underline{x}^t) \leq 0\}$.

6: **end for**

7: **return** $\bar{x}^N := \sum_{t=1}^N \omega_t x^t / (\sum_{t=1}^N \omega_t)$.

5.2.2 The Convergence Results

Next, we present convergence results for the ACGD method. The detailed analysis is deferred to Subsection subsection 5.2.4. The next proposition states some conditions required for the Q function in (Equation 5.12) to converge. Since these conditions depend on the set of reference multiplier λ's under consideration (see Lemma 5.1), it is useful to define an aggregate Lipschitz smoothness constant as a function of Λ:

$$L(\Lambda) := \max_{\lambda \in \Lambda} L_\lambda, \text{ where } L_\lambda \left\|\bar{x} - \hat{x}\right\| \geq \left\|\nabla_x \mathcal{L}(\bar{x}; \lambda) - \nabla_x \mathcal{L}(\hat{x}; \lambda)\right\| \forall \bar{x}, \hat{x} \in \mathbb{R}^n. \quad (5.17)$$

Proposition 5.1 *Let a set of reference multipliers $\Lambda \subset \mathbb{R}^m_+$ be given and let the aggregate smoothness constant $L(\Lambda)$ be defined in* (Equation 5.17). *Consider the iterates $z^t := (x^t; \lambda^t, \nu^t, p_i^t)$ generated by Algorithm 11, where λ^t is defined in* (Equation 5.16). *Suppose the following conditions are satisfied by the stepsizes together with some non-*

negative weights $\omega_t \geq 0$ *for all* $t \geq 2$*:*

$$\omega_t \eta_t \leq \omega_{t-1}(\eta_{t-1} + \alpha), \tag{5.18}$$

$$\omega_t \tau_t \leq \omega_{t-1}(\tau_{t-1} + 1), \tag{5.19}$$

$$\eta_N(\tau_N + 1) \geq L(\Lambda), \eta_{t-1}\tau_t \geq \theta_t L(\Lambda) \textit{ with } \theta_t := \omega_{t-1}/\omega_t. \tag{5.20}$$

Consider the ergodic iterate $\bar{z}^N = (\bar{x}^N; \bar{\lambda}^N, \bar{\nu}^N, \bar{\pi}^N)$ *specified according to*

$$\bar{x}^N := \sum_{t=1}^{N} \omega_t x^t / (\textstyle\sum_{t=1}^N \omega_t),\ \bar{\lambda}^N := \textstyle\sum_{t=1}^N \omega_t \lambda^t / (\sum_{t=1}^N \omega_t),\ \bar{\pi}^N := \textstyle\sum_{t=1}^N \omega_t p_i^t / (\sum_{t=1}^N \omega_t),$$
$$\bar{\nu}_i^N := \begin{cases} \sum_{t=1}^N \omega_t \lambda_i^t \nu_i^t / (\sum_{t=1}^N \omega_t \lambda_i^t) & \textit{o.w.} \\ \nabla g_i(x^0) & \textit{if } \lambda_i^t = 0\ \forall t. \end{cases} \tag{5.21}$$

Then the following convergence bound is valid for any reference point $z = (x; \lambda,\ \nu, \pi) \in X \times \Lambda \times [V, \Pi]$ *with* $[V, \Pi]$ *defined in* (Equation 5.10)*:*

$$\begin{aligned} &(\textstyle\sum_{t=1}^N \omega_t) Q(\bar{z}^N, z) + \omega_N(\eta_N + \alpha) \left\| x^N - x \right\|^2 / 2 \\ &\qquad \leq \omega_1 \eta_1 \left\| x^0 - x \right\|^2 / 2 + \omega_1 \tau_1 (U_f^*(\pi; p_i^0) + \langle \lambda, U_g^*(\nu; \nu^0) \rangle). \end{aligned} \tag{5.22}$$

Proposition 5.1, together with Lemma 5.1, show that it is possible to select a finite stepsize η_t related to $L(\Lambda_r)$ in place of the infinite $L(\mathbb{R}_+^m)$. To provide convergence guarantees for both the feasibility violation and the optimality gap associated with the function-constrained problem in (Equation 5.1), Lemma 5.1 states that we should only consider the Q gap function defined with respect to Λ_r, some small neighborhood of reference λ's around λ^*. Proposition 5.1 shows such an Q function converges when the stepsize choice satisfies certain conditions related to $L(\Lambda_r)$, rather than $L(\mathbb{R}_+^m)$. Thus a finite η_t proportional to $L(\Lambda_r)$ is sufficient for our purpose, and the next theorem states the result more

precisely.

Theorem 5.1 *Let a smooth constrained optimization problem* (Equation 5.1) *be given and let* $L(\Lambda_r)$ *be defined in* (Equation 5.13) *and* (Equation 5.17)*. Denote its condition number by* $\kappa_r = L(\Lambda_r)/\alpha$ *(We set* $\kappa_r = \infty$ *if* $\alpha = 0$*). Suppose the solution iterates* $\{x^t\}$ *are generated by Algorithm Algorithm 11 with the following stepsizes for* $t \geq 1$

$$\tau_t = \min\{\tfrac{t-1}{2}, \sqrt{\kappa_r}\},\ \eta_t = \tfrac{L(\Lambda_r)}{\tau_{t+1}},\ \theta_t = \tfrac{\tau_t}{\tau_{t-1}+1},\ \omega_t = \begin{cases} \omega_{t-1}/\theta_t & \text{if } t \geq 2, \\ 1 & \text{if } t = 1. \end{cases} \tag{5.23}$$

Then the ergodic average $\bar{x}^N$ *solution satisfies*

$$\max\{F(\bar{x}^N) - F_*, r\left\|[g(\bar{x}^N)]_+\right\|\} \leq \frac{2L(\Lambda_r)}{N(N+1)}\left\|x^0 - x^*\right\|^2. \tag{5.24}$$

Moreover, in the strongly convex case with $\alpha > 0$*,* $\bar{x}^N$ *also satisfies*

$$\max\{F(\bar{x}^N) - F_*, r\left\|[g(\bar{x}^N)]_+\right\|\} \leq \frac{\sqrt{L(\Lambda_r)\alpha}\left\|x^0 - x^*\right\|^2}{(1+1/\sqrt{\kappa_r})^{N-4} - 1}, \tag{5.25}$$

$$\left\|\bar{x}^N - x^*\right\|^2 \leq \frac{2\sqrt{\kappa_r}\left\|x^0 - x^*\right\|^2}{(1+1/\sqrt{\kappa_r})^{N-4} - 1}. \tag{5.26}$$

As a consequence of the preceding theorem, we can derive upper bounds on the number of iterations required of the ACGD method to find an $(\epsilon; \epsilon/c)$-solution (see (Equation 5.2)). The next corollary focuses on the non-strongly convex case, i.e., $\alpha = 0$.

Corollary 5.1 *Suppose* $\{x^t\}$ *are generated by Algorithm Algorithm 11 using the stepsizes choice in* (Equation 5.23)*, with* $L(\Lambda_r) = L(\Lambda_c)$ *(see* (Equation 5.17)*). The ergodic average solution* $\bar{x}^N$ *is an* $(\epsilon, \epsilon/c)$*-solution if*

$$N \geq \sqrt{\tfrac{2L(\Lambda_c)}{\epsilon}}\left\|x^0 - x^*\right\|.$$

For the strongly convex problem with $\alpha > 0$, it is advantageous to choose a small r to

ensure a small aggregate smoothness constant $L(\Lambda_r)$ and hence a small condition number κ_r. We set it to $L(\Lambda_1)$ in the next corollary.

Corollary 5.2 *Suppose* $\{x^t\}$ *are generated by Algorithm Algorithm 11 using the stepsizes choice in* (Equation 5.23)*, with* $L(\Lambda_r) = L(\Lambda_1)$ *(see* (Equation 5.17)*). Then the ergodic average solution* $\bar{x}^N$ *is an* $(\epsilon, \epsilon/c)$*-solution if*

$$N \geq \min\{\sqrt{\tfrac{2\max\{c,1\}L(\Lambda_1)}{\epsilon}}\left\|x^0 - x^*\right\|, [\sqrt{\tfrac{L(\Lambda_1)}{\alpha}}+1]\log[\tfrac{\max\{c,1\}\sqrt{L(\Lambda_1)\alpha}\left\|x^0-x^*\right\|^2}{\epsilon}+1]+4\}.$$

Note that for a small λ^*, i.e., $\|\lambda^*\|$ being significantly less than 1, it may be worthwhile to choose $L(\Lambda_r) = L(\Lambda_{\|\lambda^*\|})$ to further reduce the condition number from $\kappa_1 = L(\Lambda_1)/\alpha$ to $\kappa_{\|\lambda^*\|} = L(\Lambda_{\|\lambda^*\|})/\alpha$.

Since each iteration of the ACGD method requires only one gradient evaluation, the preceding two corollaries establish the desired $\mathcal{O}(1/\sqrt{\epsilon})$ and $\mathcal{O}(\log(1/\epsilon))$ oracle complexities for the non-strongly convex and the strongly convex problems, respectively.

5.2.3 The Binary Search for $L(\Lambda_r)$

A crucial limitation to the ACGD method is that the desired aggregate smoothness constants, $L(\Lambda_c)$ and $L(\Lambda_1)$, may be unavailable for the stepsize calculation in (Equation 5.23) during implementation. To address this issue, we propose a binary search outer-loop method to look for the suitable constant in an online fashion. Starting with a guess $\tilde{L}$, we run the ACGD method for $N(\tilde{L})$ iterations and check if the ergodic average solution $\bar{x}^{N(\tilde{L})}$ satisfies a certain termination criterion. The outer loop would be stopped if the termination criterion is met, otherwise the process would be repeated with the guess doubled, i.e., $2\tilde{L}$. To ensure the soundness of the search routine, we need to choose the iteration limit function $N(\tilde{L})$ to guarantee the satisfaction of the termination criterion when $\tilde{L}$ were larger than the desired aggregate smoothness constant. Additionally, the termination criterion should also be both easily computable and valid, i.e., a test solution $\bar{x}^N$ which satisfies the

criterion should be $(\epsilon; \epsilon/c)$-optimal. Toward these ends, we need the following additional assumptions.

Assumption 5 *a) X is bounded and an estimate D_X of its radius, i.e., $D_X \geq \max_{x,y\in X} \|x - y\|$, is known.*

b) The dual multiplier to the quadratic program in Line 5 of Algorithm Algorithm 11, λ^t, is available.

Given iterates $\{(x^t, \underline{x}^t), \lambda^t, \ \nu^t, p_i^t\}_{t=1}^N$ generated by the ACGD method, our termination criterion is the two conditions:

$$\left\|[g(\bar{x}^N)]_+\right\| \leq c\epsilon \text{ and } F(\bar{x}^N) - \underline{F}^N \leq \epsilon, \tag{5.27}$$

where $\bar{x}^N$ is the ergodic average solution and $\underline{f}^N$ is given by

$$\begin{aligned} \underline{F}^N := \underset{x\in X}{\arg\min} \ & \frac{1}{\sum_{t=1}^N \omega_t} \sum_{t=1}^N \omega_t [\langle p_i^t, x - \underline{x}^t\rangle + f(\underline{x}^t) + u(x))] \\ s.t. \ & \frac{1}{\sum_{t=1}^N \omega_t \lambda_i^t} \sum_{t=1}^N \omega_t \lambda_i^t [\langle \nu_i^t, x - \underline{x}^t\rangle + g_i(\underline{x}^t)] \leq 0 \ \forall i \in [m]. \end{aligned} \tag{5.28}$$

Notice Assumption 5.a) ensures $\underline{F}^N$ to be finite, and the dual solutions in Assumption 5.b) are used for constructing the linearized constraints above. Since its objective and constraints are lower linear approximations to f and $g_1, g_2, \ldots, g_m$ respectively, (Equation 5.28) is a relaxation to (Equation 5.1), and the minimum value $\underline{F}^N$ is a lower bound to F_*. Therefore the satisfaction of (Equation 5.27) guarantees the solution $\bar{x}^N$ to be $(\epsilon, \epsilon/c)$-optimal.

Algorithm 12 Binary Search for the Aggregate Smoothness Constant L.

Input: $\bar{x} \in X$, $\tilde{L} > 0$, and an iteration limit function $N(L)$.

while true **do**
  Set stepsizes to (Equation 5.23) with $L(\Lambda_r)$ replaced by $\tilde{L}$.
  Run the ACGD method for $N(\tilde{L})$ iterations starting from $\bar{x}$.
  Set $\bar{x} \leftarrow \bar{x}^{N(\tilde{L})}$, where $\bar{x}^{N(\tilde{L})}$ is the ergodic average solution.
  if $\bar{x}$ satisfies the termination criterion (Equation 5.27) **then**
    break
  else
    Set $\tilde{L} \leftarrow 2\tilde{L}$.
  end if
end while
return $\bar{x}$.

Putting the pieces together, the binary search scheme is listed in Algorithm Algorithm 12. When it terminates, the output solution is $(\epsilon, \epsilon/c)$-optimal. Now we suggest some concrete iteration limit functions, $N(L)$, and derive upper bounds on the total numbers of iterations, across all ACGD runs, required for termination. Their proof is deferred to Subsection subsection 5.2.4.

Theorem 5.2 *Consider a smooth constrained optimization problem of form* (Equation 5.1). *Suppose its aggregate smoothness constants,* $L(\Lambda_c)$ *and* $L(\Lambda_1)$*, are defined according to* (Equation 5.17) *and* (Equation 5.13)*, and Assumption 5 is valid. Let* $\tilde{L}$ *be initialized to some* $\tilde{L}_0$*, then Algorithm Algorithm 12 terminate finitely for the following choices of the iteration limit function* $N(L)$.

a) In the non-strongly convex case with $\alpha = 0$*, if* $N(L) = \lceil \sqrt{\frac{2L}{\epsilon}} D_X \rceil$*, the total number of ACGD iterations required for termination is upper bounded by*

$$\max\{7\sqrt{\tfrac{L(\Lambda_c)}{\alpha}} D_X + \lceil \log_2(\tfrac{L(\Lambda_c)}{\tilde{L}_0}) \rceil, \sqrt{\tfrac{2\tilde{L}_0}{\epsilon}} D_X + 1\}. \tag{5.29}$$

b) In the strongly convex case with $\alpha > 0$, if

$$N(L) = \lceil (\sqrt{\frac{L}{\alpha}} + 1) \log(\frac{\max\{c,1\}\sqrt{L\alpha}D_X^2}{\epsilon} + 1) \rceil + 4,$$

the total number of ACGD iterations required for termination is upper bounded by

$$\begin{aligned} \max \Bigg\{ & 9\sqrt{\frac{L(\Lambda_1)}{\alpha}} \log(\frac{\max\{c,1\}\sqrt{2L(\Lambda_1)\alpha}D_X^2}{\epsilon} + 1) + 5\lceil \log_2 \frac{L(\Lambda_1)}{\tilde{L}_0} \rceil, \\ & \lceil (\sqrt{\frac{\tilde{L}_0}{\alpha}} + 1) \log(\frac{\max\{c,1\}\sqrt{\tilde{L}_0\alpha}D_X^2}{\epsilon} + 1) \rceil + 4 \Bigg\}. \end{aligned} \tag{5.30}$$

We make two remarks regarding the result. First, the proposed $N(L)$'s depend only on the target accuracy $(\epsilon; \epsilon/c)$ and some easy-to-estimate problem parameters, that is, D_X in the non-strongly convex case, and D_X and α in the strongly convex case. So Algorithm Algorithm 12 is easy to implement. Second, the total numbers of ACGD iterations required by Algorithm Algorithm 12, (item 5.29) and (item 5.30), are nearly independent of the initial guess $\tilde{L}_0$ for all $\tilde{L}_0 \leq L(\Lambda_0)$. The FO oracle complexities of Algorithm Algorithm 12 have the same orders as those in Corollary 5.1 and 5.2, so the total cost of the multiple ACGD runs in Algorithm Algorithm 12 is on the same order as a single ACGD run with some carefully selected stepsizes.

5.2.4 Convergence Analysis

We present the detailed proofs to convergence results of Algorithm Algorithm 11 and Algorithm Algorithm 12.

Proof of Proposition 5.1 Let Q_x, Q_λ, Q_ν, and Q_π be defined in (Equation 5.14). It is useful to view the updates in Algorithm Algorithm 11 from the perspective of prox-mappings in (Equation 5.16). First, let's consider Q_ν. We have from the definition of $\tilde{x}^t$

that

$$\langle \nu_i - \nu_i^t, (\tilde{x}^t - x^t) \rangle = - \langle \nu_i - \nu_i^t, (x^t - x^{t-1}) \rangle + \theta_t \langle \nu_i - \nu_i^{t-1}, (x^{t-1} - x^{t-2}) \rangle + \theta_t \langle \nu_i^{t-1} - \nu_i^t, (x^{t-1} - x^{t-2}) \rangle.$$

Since g_i^* has a strong convexity modulus of 1 with respect to $U_{g_i^*}$, the definition of ν_i^t prox-mapping in (Equation 5.16) implies a three-point inequality (see Lemma 3.5 in [28]):

$$\langle \nu_i - \nu_i^t, x^t \rangle + g_i^*(\nu_i^t) - g_i^*(\nu_i) + \langle \nu_i - \nu_i^t, \tilde{x}^t - x^t \rangle \leq \tau_t U_{g_i^*}(\nu_i; \nu_i^{t-1}) - (\tau_t + 1) U_{g_i^*}(\nu_i; \nu_i^t) - \tau_t U_{g_i^*}(\nu_i^t; \nu_i^{t-1}).$$

So, combining the above two relations, taking the ω_t weighted sum of the resulting inequalities and using the conditions $\omega_{t-1} = \omega_t \theta_t$ and $\omega_t \tau_t \leq \omega_{t-1}(\tau_{t-1} + \mu)$, we obtain

$$\begin{aligned} &\sum_{t=1}^{N} \omega_t (\langle \nu_i - \nu_i^t, x^t \rangle + g_i^*(\nu_i^t) - g_i^*(\nu_i)) \\ &\quad \leq - (\omega_N(\tau_N + 1) U_{g_i^*}(\nu_i; \nu_i^N) - \omega_N \langle \nu_i - \nu_i^N, (x^N - x^{N-1}) \rangle) \\ &\qquad - \textstyle\sum_{t=2}^{N} [\omega_t \tau_t U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + \omega_{t-1} \langle \nu_i^{t-1} - \nu_i^t, (x^{t-1} - x^{t-2}) \rangle] \\ &\qquad + \omega_1 \tau_1 U_{g_i^*}(\nu_i; \nu_i^0). \end{aligned}$$

A λ_i-weighted sum of the above inequality leads to the desired Q_ν convergence bound given by

$$\begin{aligned} &\sum_{t=1}^{N} \omega_t Q_\nu(z^t; z) \\ &\quad \leq - (\omega_N(\tau_N + 1) \textstyle\sum_{i=1}^{m} \lambda_i U_{g_i^*}(\nu_i; \nu_i^N) - \omega_N \langle \sum_{i=1}^{m} \lambda_i (\nu_i - \nu_i^N), x^N - x^{N-1} \rangle) \\ &\qquad - \textstyle\sum_{t=2}^{N} [\omega_t \tau_t \sum_{i=1}^{m} \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + \omega_{t-1} \langle \sum_{i=1}^{m} \lambda_i (\nu_i^{t-1} - \nu_i^t), x^{t-1} - x^{t-2} \rangle] \\ &\qquad + \omega_1 \tau_1 \textstyle\sum_{i=1}^{m} \lambda_i U_{g_i^*}(\nu_i; \nu_i^0). \end{aligned}$$

A Q_π bound can be derived similarly. Taken together, we get

$$\begin{aligned}
&\sum_{t=1}^{N} \omega_t [Q_\nu(z^t; z) + Q_\pi(z^t; z)] \\
&\leq -(\omega_N(\tau_N + 1)[\textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^t) + U_{f^*}(\pi; p_i^t)] \\
&- \omega_N \langle \pi - p_i^N + \textstyle\sum_{i=1}^m \lambda_i (\nu_i - \nu_i^N), x^N - x^{N-1} \rangle) \\
&\quad - \textstyle\sum_{t=2}^N \{\omega_t \tau_t [\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + U_{f^*}(p_i^t; p^{t-1})] \\
&+ \omega_{t-1} \langle p_i^{t-1} - p_i^t + \textstyle\sum_{i=1}^m \lambda_i (\nu_i^{t-1} - \nu_i^t), x^{t-1} - x^{t-2} \rangle \} \\
&\quad + \omega_1 \tau_1 [\textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^0) + U_{f^*}(\pi; p_i^0)].
\end{aligned} \tag{5.31}$$

Since $(\nu^t, p_i^t) = (\nabla g(\underline{x}^t), \nabla f(\underline{x}^t))\ \forall t$, applying Lemma 5.2 with $\tilde{\lambda} = [\lambda; 1]$ and $\tilde{g} = [g; f]$ implies that

$$\textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + U_{f^*}(p_i^t; p^{t-1}) \geq \frac{1}{2L(\Lambda)} \left\| p_i^t - p^{t-1} + \sum_{i=1}^m \lambda_i (\nu_i^t - \nu_i^{t-1}) \right\|^2.$$

Similarly, with $(\nu, \pi) \in [V, \Pi]$ (see (Equation 5.10)), we get

$$\textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu; \nu_i^N) + U_{f^*}(\pi; p_i^N) \geq \frac{1}{2L(\Lambda)} \left\| \pi - p_i^N + \sum_{i=1}^m \lambda_i (\nu_i - \nu_i^N) \right\|^2.$$

Thus applying the Young's inequality to (Equation 5.31) leads to

$$\begin{aligned}
\sum_{t=1}^{N} \omega_t [Q_\nu(z^t; z) + Q_\pi(z^t; z)] \leq & \tfrac{\omega_N L(\Lambda)}{2(\tau_N + 1)} \left\| x^N - x^{N-1} \right\|^2 + \textstyle\sum_{t=1}^{N-1} \frac{\omega_t \theta_t L(\Lambda)}{2\tau_t} \left\| x^t - x^{t-1} \right\|^2 \\
& + \omega_1 \tau_1 [\textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^0) + U_{f^*}(\pi; p_i^0)].
\end{aligned} \tag{5.32}$$

Now let us move onto Q_x and Q_λ. Fix λ^t. The x^t-prox mapping implies a three-point

inequality (Lemma 3.5 in [28]):

$$\begin{aligned}&\langle p_i^t + \textstyle\sum_{i=1}^m \lambda_i^t \nu_i^t, x^t - x\rangle + u(x^t) - u(x) + \frac{\eta_t+\alpha}{2}\left\|x^t - \hat{x}\right\|^2 \\ &\leq \frac{\eta_t}{2}\left\|x^{t-1} - \hat{x}\right\|^2 - \frac{\eta_t}{2}\left\|x^t - x^{t-1}\right\|^2.\end{aligned}$$

Fix x^t. The optimality of λ^t implies that:

$$\langle \lambda - \lambda^t, \nu^t x^t - g^*(\nu^t)\rangle \leq 0.$$

So we have

$$Q_x(z^t; z) + Q_\lambda(z^t; z) + \frac{\eta_t+\alpha}{2}\left\|x^t - x\right\|^2 \leq \frac{\eta_t}{2}\left\|x^{t-1} - x\right\|^2 - \frac{\eta_t}{2}\left\|x^t - x^{t-1}\right\|^2.$$

Summing across iterations with weight ω_t leads to

$$\begin{aligned}&\textstyle\sum_{t=1}^N \omega_t[Q_x(z^t; z) + Q_\lambda(z^t; z)] + \frac{\omega_N(\eta_N+\alpha)}{2}\left\|x^t - x\right\|^2 \\ &\leq \frac{\omega_1\eta_1}{2}\left\|x^0 - x\right\|^2 - \sum_{t=1}^N \frac{\omega_t\eta_t}{2}\left\|x^t - x^{t-1}\right\|^2.\end{aligned} \tag{5.33}$$

Utilizing the stepsize assumption (Equation 5.20), we can add it to (Equation 5.32) to obtain a convergence bound for the Q function

$$\begin{aligned}&\textstyle\sum_{t=1}^N \omega_t Q(z^t; z) + \frac{\omega_N(\eta_N+\alpha)}{2}\left\|x^t - x\right\|^2 \\ &\leq \frac{\omega_1\eta_1}{2}\left\|x^0 - x\right\|^2 + \omega_1\tau_1[\textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^0) + U_{f^*}(\pi; p_i^0)].\end{aligned} \tag{5.34}$$

Moreover, the Jensen's inequality implies that

$$(\textstyle\sum_{t=1}^N \omega_t)\mathcal{L}(\bar{x}^N; \lambda, \nu, \pi) \leq \sum_{t=1}^N \omega_t \mathcal{L}(x^t; \lambda, \nu, \pi), \tag{5.35}$$

$$\sum_{t=1}^{N} \omega_t \mathcal{L}(x; \lambda^t, \nu^t, p_i^t)$$
$$\geq (\sum_{t=1}^{N} \omega_t)[\langle x, \bar{\pi}^N \rangle - f^*(\bar{\pi}^N) + u(x)] + \sum_{i=1}^{m} (\sum_{t=1}^{N} \omega_t \lambda_i)[\langle \bar{\nu}_i^t, x \rangle - g_i^*(\bar{\nu}_i^N)]$$
$$\geq \sum_{t=1}^{N} \omega_t (\mathcal{L}(x; \bar{\lambda}^N, \bar{\nu}^N, \bar{\pi}^N)).$$

Thus, we get $(\sum_{t=1}^{N} \omega_t) Q(\bar{z}^N; z) \leq \sum_{t=1}^{N} \omega_t Q(z^t; z)$, and the desired inequality in (Equation 5.22) follows from (Equation 5.34). □

Next, the proof of Theorem Theorem 5.1 is a direct application of Proposition 5.1. The analysis is complicated by the switch from a diminishing stepsize to a constant stepsize in (Equation 5.23).

Proof of Theorem Theorem 5.1 We apply Proposition 5.1 to obtain the results. First, we verify that the requirements in (Equation 5.20) and (Equation 5.19) are satisfied by the stepsize choice in (Equation 5.23). Since $\theta_t = \tau_t/(\tau_{t-1} + 1)$ and $\eta_t = L(\Lambda_r)/\tau_{t+1}$, all requirements other than (Equation 5.18) hold automatically. Now let $T = \lceil 2\sqrt{\kappa_r} \rceil + 1$ denote the first iteration at which we switch to $\tau_t = \sqrt{\kappa_r}$. Since the other iterations and the case with $\alpha = 0$ are straightforward to check, we focus on iteration $T - 1$ and T, and assume $\alpha > 0$. For $t = T$, we have

$$\omega_T \eta_T = \omega_{T-1} \eta_T \frac{\tau_{T-1}+1}{\tau_T} = \omega_{T-1} \frac{L(\Lambda_r)}{\sqrt{\kappa_r}} \frac{\lceil 2\sqrt{\kappa_r} \rceil + 1}{2\sqrt{\kappa_r}}$$
$$= \omega_{T-1} \alpha \frac{2\sqrt{\kappa_r}+2}{2} \leq \omega_{T-1}(\alpha\sqrt{\kappa_r} + \alpha) \leq \omega_{T-1}(\eta_{T-1} + \alpha).$$

For $t = T - 1$, we have

$$\omega_{T-1} \eta_{T-1} = \omega_{T-2} \eta_{T-1} \frac{\tau_{T-2}+1}{\tau_{T-1}} = \omega_{T-2} \frac{2L(\Lambda_r)}{2\sqrt{\kappa_r}} \frac{\lceil 2\sqrt{\kappa_r} \rceil - 1}{\lceil 2\sqrt{\kappa_r} \rceil - 2}$$
$$\leq \omega_{T-2} \frac{2L(\Lambda_r)}{\lceil 2\sqrt{\kappa_r} \rceil - 2} = \omega_{T-2} \eta_{T-2} \leq \omega_{T-2}(\eta_{T-2} + \alpha).$$

Thus the requirements in Proposition 5.1 are satisfied, and we have

$$Q(\bar{z}^N, z) \leq L(\Lambda_r) \left\|x^0 - x\right\|^2 / (\sum_{t=1}^N \omega_t) \; \forall z = (x; \lambda, \nu, \pi) \in X \times \Lambda_r \times [V, \Pi]. \quad (5.36)$$

We provide a lower bound to $(\sum_{t=1}^N \omega_t)$. It is useful to show

$$\omega_t \geq \max\{t, (1 + 1/\sqrt{\kappa_r})^{t-5}\}.$$

Since $\omega_t = 1/(\prod_{t=2}^N \theta_t)$, the fact $\omega_t \geq t \; \forall t \geq 1$ is straightforward. Regarding the second lower bound, let us first consider $t \leq T = \lceil 2\sqrt{\kappa_r} \rceil + 1$. The algebraic fact in Lemma 5.3 implies that $\omega_{t+2} \geq t \geq (1 + 1/\sqrt{\kappa_r})^{t-3} \forall t \leq 2\sqrt{\kappa_r}$, so $\omega_t \geq (1 + 1/\sqrt{\kappa_r})^{t-5} \forall t \leq T = \lceil 2\sqrt{\kappa_r} \rceil + 1$. For $t \geq T + 1$, the relation is also valid since $1/\theta_t = (1 + 1/\sqrt{\kappa_r})$. Therefore we get $\omega_t \geq \max\{t, (1 + 1/\sqrt{\kappa_r})^{t-5}\} \forall t \geq 1$. Using these two lower bounds, it is easy to derive

$$\sum_{t=1}^N \omega_t \geq \max\{N(N+1)/2, \sqrt{\kappa_r}[(1 + 1/\sqrt{\kappa_r})^{N-4} - 1]\}. \quad (5.37)$$

Substituting the preceding lower bound into (Equation 5.36) and applying Lemma 5.1 lead us to the convergence results in (Equation 5.24) and (Equation 5.25).

Now we deduce the convergence bound for $\|x^t - x^*\|$. Choosing the reference point z to be $\hat{z} = (x^*; \lambda^*, \nabla g(\bar{x}^N), \nabla f(\bar{x}^N))$ leads to

$$Q(\bar{z}^N; \hat{z}) \leq L(\Lambda_r) \left\|x^0 - x^*\right\|^2 / (\sum_{t=1}^N \omega_t). \quad (5.38)$$

Moreover, since x^* minimizes the optimal Lagrangian, we get

$\langle \nabla f(x^*) + (\lambda^*)^\top \nabla g(x^*) + u'(x^*), \bar{x}^N - x^* \rangle \geq 0$ for all $u'(x^*) \in \partial u(x^*)$. So we have

$$\begin{aligned} Q(\bar{z}^N; \hat{z}) &\geq f(\bar{x}^N) + (\lambda^*)^\top g(\bar{x}^N) + u(\bar{x}^N) - [f(x^*) + (\lambda^*)^\top g(x^*) + u(x^*)] \\ &\quad - \langle \nabla f(x^*) + (\lambda^*)^\top \nabla g(x^*) + u'(x^*), \bar{x}^N - x^* \rangle \\ &= f(\bar{x}^N) - f(x^*) - \langle \nabla f(x^*), \bar{x}^N - x^* \rangle + \textstyle\sum_{i=1}^m \lambda_i^* [g_i(\bar{x}^N) - g_i(x^*)] \\ &\quad - \textstyle\sum_{i=1}^m \lambda_i^* [\langle \nabla g_i(x^*), \bar{x}^N - x^* \rangle] + [u(\bar{x}^N) - u(x^*) - \langle u'(x^*), \bar{x}^N - x^* \rangle] \\ &\geq \alpha \left\| \bar{x}^N - x^* \right\|^2 / 2, \end{aligned} \tag{5.39}$$

where the last inequality follows from the fact that the regularization function $u(x)$ is α-strongly convex. Combining the above two relations, we obtain the desired inequality in (Equation 5.26):

$$\left\| \bar{x}^N - x^* \right\|^2 \leq 2\kappa_r \left\| x^0 - x \right\|^2 / (\textstyle\sum_{t=1}^N \omega_t) \leq \frac{2\sqrt{\kappa_r} \left\| x^0 - x^* \right\|^2}{(1+1/\sqrt{\kappa_r})^{N-4} - 1}.$$

□

Next, we move on to provide proofs to the binary search outer-loop in Algorithm Algorithm 12.

Proof of Theorem Theorem 5.2: First, let's consider the non-strongly problem. We show the soundness of the Algorithm Algorithm 12, that is, the output ergodic average solution from the ACGD method satisfies termination criterion (Equation 5.27) if the estimate $\tilde{L}$ is larger than the desired aggregate smoothness constant $L(\Lambda_c)$. Let $\tilde{N} := N(\tilde{L}) = \lceil D_X \sqrt{2\tilde{L}/\epsilon} \rceil$ such that $\bar{x}^{\tilde{N}}$ denotes the output solution. Since $D_X \geq \|x^0 - x^*\|$, $\|[g(\bar{x}^{\tilde{N}})]_+\| \leq \epsilon/c$ is a direct consequence of Corollary 5.1. Now consider the objective criterion. Since $\tilde{L} \geq L(\Lambda_c)$, it follows from (Equation 5.34) that for all $z = (x; \lambda, \nu, \pi) \in X \times L(\Lambda_c) \times [V, \Pi]$

$$\textstyle\sum_{t=1}^N \omega_t Q(z^t; z) \leq \omega_1 \eta_1 \left\| x^0 - x \right\|^2 / 2 \leq \omega_1 \eta_1 D_X^2 / 2.$$

Substituting the stepsize choice in (Equation 5.23) then leads to

$$\frac{\sum_{t=1}^{\tilde{N}} \omega_t Q(z^t, z)}{\sum_{t=1}^{\tilde{N}} \omega_t} \leq \frac{\tilde{L} D_X^2}{\sum_{t=1}^{\tilde{N}} \omega_t} \leq \epsilon, \forall z = (x; \lambda, \nu, \pi) \in X \times L(\Lambda_c) \times [V, \Pi].$$

In particular, fixing $\hat{p}_i = \nabla f(\bar{x}^{\tilde{N}})$ and $\hat{\nu} = \nabla g(\bar{x}^{\tilde{N}})$, we have

$$F(\bar{x}^{\tilde{N}}) \leq (\sum_{t=1}^{\tilde{N}} \omega_t \mathcal{L}(x^t; 0, \hat{\nu}, \hat{p}_i))(\sum_{t=1}^{\tilde{N}} \omega_t).$$

Moreover, since

$$\begin{aligned}\sum_{t=1}^{\tilde{N}} \frac{\omega_t}{\sum_{t=1}^{\tilde{N}} \omega_t} \mathcal{L}(x; \lambda^t, \nu^t, p_i^t) =& \frac{1}{\sum_{t=1}^{\tilde{N}} \omega_t} \sum_{t=1}^{\tilde{N}} \omega_t [\langle p_i^t, x - \underline{x}^t \rangle + f(\underline{x}^t) + u(x^t)] \\ &+ \sum_{i=1}^{m} [\frac{\sum_{t=1}^{\tilde{N}} \omega_t \lambda_i^t}{\sum_{t=1}^{\tilde{N}} \omega_t}] \frac{\sum_{t=1}^{\tilde{N}} \omega_t \lambda_i^t [\langle \nu_i^t, x - \underline{x}^t \rangle + g_i(\underline{x}^t)]}{\sum_{t=1}^{\tilde{N}} \omega_t \lambda_i^t},\end{aligned}$$

the function is a Lagrangian relaxation to the linearly constrained problem in (Equation 5.28), so

$$\min_{x \in X} \frac{\omega_t}{\sum_{t=1}^{\tilde{N}} \omega_t} \mathcal{L}(x; \lambda^t, \nu^t, p_i^t) \leq \underline{f}^{\tilde{N}}.$$

Putting these facts together, we get

$$F(\bar{x}^{\tilde{N}}) - \underline{F}^{\tilde{N}} \leq \max_{z \in X \times L(\Lambda_c) \times [V, \Pi]} \frac{\sum_{t=1}^{\tilde{N}} \omega_t Q(z^t, z)}{\sum_{t=1}^{\tilde{N}} \omega_t} \leq \epsilon.$$

Thus $\bar{x}^{\tilde{N}}$ must satisfy the termination criterion in (Equation 5.27) if $\tilde{L} \geq L(\Lambda_c)$, and the total number of ACGD iterations follows from a straightforward algebraic calculation.

Next, the finite termination of Algorithm Algorithm 12 for the strongly convex case can be deduced similarly. With the given choice of $N(L)$, we need to show the termination criterion (Equation 5.27) is satisfied for the ergodic solution $\bar{x}^{\tilde{N}}$ if the estimate $\tilde{L}$ were larger than $L(\Lambda_1)$. $\square$

5.3 Lower Oracle Complexity Bound

In this section, we present the lower oracle complexity bounds, that is, the minimum number of queries to the FO oracle required to find an $(\epsilon; \epsilon/c)$-optimal solution. These results illustrate the optimality of the ACGD method under a certain optimality regime. We assume, for the sake of simplicity, that $u(x) = \alpha \|x\|^2 /2$ (see (Equation 5.1)) and that X is radially invariant, for example, the $\mathbb{R}^n$.

Similar to Nesterov's lower complexity computation model in [1], we consider the class of all first-order methods, $\mathcal{F}$, verifying a linear-span update requirement. Given a (finite) memory of reachable points $\mathcal{M}_{t-1}$ after the $t-1^{\text{th}}$ query to FO, the updated memory $\mathcal{M}_t$ after evaluating the FO oracle at some $y \in \text{span}(\mathcal{M}_{t-1})$ needs to satisfy

$$\mathcal{M}_t \subset \{x + \eta_t \nabla f(y) + \textstyle\sum_{i=1}^m \tau_i \nabla g_i(y) : x \in \text{span}(\mathcal{M}_{t-1}), \eta_t, \tau_{i,t} \in \mathbb{R}\}. \tag{5.40}$$

The freedom to choose arbitrary elements from the linear span allows $\mathcal{F}$ to cover many first-order algorithms. For example, the ACGD method is a special case of $\mathcal{F}$ because the generated points in the ACGD method, $\mathcal{M}_t = \{x^0, x^1, \ldots, x^t\}$, satisfy the requirement (Equation 5.40). Specifically, when the memory $\mathcal{M}_{t-1} = \{x^0, x^1, \ldots, x^{t-1}\}$, the evaluation point $\underline{x}^t$ in Line 3 is inside span$(\mathcal{M}_{t-1})$. Moreover, the x^t-update in Line 5 of Algorithm Algorithm 11 can be expressed as

$$\begin{aligned}
x^t \leftarrow & \arg\min_{x \in X} \{\langle \nabla f(\underline{x}^t), x\rangle + u(x) + \eta_t \left\|x - x^{t-1}\right\|^2 /2 \\
& \qquad s.t.\ \nabla g(\underline{x}^t)(x - \underline{x}^t) + g(\underline{x}^t) \le 0\} \\
\overset{(a)}{\Leftrightarrow} x^t \leftarrow & \arg\min_{x \in X} \{\langle \nabla f(\underline{x}^t) + \textstyle\sum_{i=1}^m \lambda_i^t \nabla g_i(\underline{x}^t), x\rangle + \alpha \|x\|^2 /2 + \eta_t \left\|x - x^{t-1}\right\|^2 /2 \} \\
\Leftrightarrow x^t \leftarrow & \arg\min_{x \in X} \left\| x - \tfrac{1}{\eta_t + \alpha}[x^{t-1} - \tfrac{1}{\eta_t}(\nabla f(\underline{x}^t) + \textstyle\sum_{i=1}^m \lambda_i^t \nabla g_i(\underline{x}^t))] \right\|^2 \\
\overset{(b)}{\Leftrightarrow} x^t = & \ \tfrac{1}{\gamma(\eta_t + \alpha)}[x^{t-1} - \tfrac{1}{\eta_t}(\nabla f(\underline{x}^t) + \textstyle\sum_{i=1}^m \lambda_i^t \nabla g_i(\underline{x}^t))] \ \text{ for some } \gamma > 0,
\end{aligned}$$

where the multiplier λ^t in (a) is the optimal dual solution to the quadratic program, and (b) follows from X being radially invariant. Therefore, x^t is a member of the right-hand side of (Equation 5.40), $\mathcal{M}_t := \mathcal{M}_{t-1} \cup \{x^t\}$ satisfies the update requirement (Equation 5.40), and the ACGD method is a member of $\mathcal{F}$. In fact, similar arguments can be used to show that $\mathcal{F}$ covers both the primal methods [101, 1], and the primal-dual methods [98, 99, 100] in the literature.

Since the dependence on parameters of the smooth objective function is well-established, e.g. [2, 1], we will investigate the dependence of the lower complexity bounds on the parameters of the constraint function. Toward that end, we consider an affine f in the objective, i.e. $L_f = 0$, and study the dependence of the lower complexity bound on the norm of the optimal Lagrange multiplier $\|\lambda^*\|$ and the Lipschitz smoothness constant of the vector-valued constraint function $\bar{L}_g$, i.e., $\|\nabla g(x) - \nabla g(y)\| \leq \bar{L}_g \|x - y\| \, \forall x, y \in \mathbb{R}^n$. These parameters are used more often in the literature, but we will relate them to the aggregate smoothness constant $L(\Lambda_r)$ (Equation 5.17) for the ACGD method in the forthcoming discussion.

5.3.1 Strongly Convex Case

First, we study the strongly convex problem with $\alpha > 0$. Since linear convergence is expected, different optimality criteria have little impact on the lower complexity bound, that is, they only have different constants inside the "$\log$". So, without the loss of generality, we choose to focus on the convergence of $\|x^t - x^*\|$ in the next theorem.

Theorem 5.3 *Let problem parameters $\bar{L}_g > 0$, $l \geq 1$, and $\bar{L}_g l \geq \alpha > 0$ be given. There exists an infinite dimensional hard problem of the form* (Equation 5.1) *with $\|\lambda^*\| = l$, g being $\bar{L}_g$-Lipschitz smooth, and f being affine, i.e., $L_f = 0$, such that every first-order method in $\mathcal{F}$ requires at least $\Omega(\sqrt{\bar{L}_g l/\alpha} \log(1/\epsilon))$ queries to the FO oracle to find an x^N with $\left\|x^N - x^*\right\|^2 \leq \epsilon$ for all $\epsilon > 0$.*

Proof: Let $\gamma = \alpha/(\bar{L}_g l)$, $\Delta = (1-\sqrt{\gamma})/(1+\sqrt{\gamma})$, $\beta = \bar{L}_g$ and $\bar{x} = [\Delta, \Delta^2, \ldots, \Delta^i, \ldots] \in \mathbb{R}^\infty$. Consider the following hard problem:

$$\begin{aligned} &\min_{x\in\mathbb{R}^\infty} -\tfrac{l\beta-\alpha}{4}x_1 + \tfrac{\alpha}{2}\|x\|^2 \\ &s.t.\ g(x) := h(x) - h(\bar{x}) \le 0 \text{ with } h(x) = \tfrac{\beta-\alpha/l}{8}[x_1^2 + \textstyle\sum_{i=1}^m [\infty](x_i - x_{i+1})^2]. \end{aligned} \tag{5.41}$$

Clearly, the objective has a strong convexity modulus of α and the constraint function $g(x)$ has a smoothness constant of $\bar{L}_g$. It is straightforward to verify that $\lambda^* = l$ and $x^* = \bar{x}$ satisfies the KKT condition:

$$g(x^*) = 0 \text{ and } \left(\tfrac{\lambda^*}{l}\tfrac{l\beta-\alpha}{4} \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \ldots \\ & & & & \ddots \end{bmatrix} + \alpha \begin{bmatrix} 1 \\ 1 \\ 1 \\ \vdots \end{bmatrix} \right) \begin{bmatrix} x_1^* \\ x_2^* \\ x_3^* \\ \vdots \end{bmatrix} = \tfrac{l\beta-\alpha}{4} \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}.$$

Starting from $\mathcal{M}_0 = \{0\}$, at iteration t, the solution x_t generated by any first-order method satisfying (Equation 5.40) have non-zeros in only the first t coordinates. Thus we have

$$\begin{aligned} \left\|x^t - x^*\right\|^2 &\ge \textstyle\sum_{i=t+1}^\infty (x_i^*)^2 = (\tfrac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^{2t} \sum_{i=1}^m [\infty](\tfrac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^{2i} = (\tfrac{1-\sqrt{\gamma}}{1+\sqrt{\gamma}})^{2t} \left\|x^0 - x^*\right\|^2 \\ &\ge (1-\sqrt{\gamma})^{2t} \left\|x^0 - x^*\right\|^2, \end{aligned}$$

where the last inequality follows from $1 \ge \gamma \ge 0$. Therefore we require at least $t = \Omega(\sqrt{\bar{L}_g l/\alpha}\log(1/\epsilon))$ iterations to find x^t with $\|x^t - x^*\|^2 \le \epsilon$. □

Since, in this section, f is assumed to be affine such that $L_f = 0$, we have $L(\Lambda_1) \le (1+\|\lambda^*\|)\bar{L}_g = \mathcal{O}(\|\lambda^*\|\,\bar{L}_g)$. The preceding theorem then shows the $\mathcal{O}(\sqrt{L(\Lambda_1)/\alpha}\log(1/\epsilon))$ upper complexity bound in Corollary 5.2 to be unimprovable when $\|\lambda^*\| \ge 1$, that is, the ACGD method has a tight oracle complexity.

5.3.2 Non-strongly Convex Case

Now we move on to consider the non-strongly convex problem with $\alpha = 0$. The next theorem states the lower oracle complexity bound to find an $(\epsilon; \epsilon/c)$-optimal solution.

Theorem 5.4 *Let problem parameters $\bar{L}_g > 0$, $R_0 \geq 1$, $l > 0$, $c \geq 1$ and $\epsilon > 0$ be given. For a large enough problem dimension, $n > 2\lceil R_0\sqrt{\bar{L}_g c/\epsilon}\rceil$, there exists a hard problem of form* (Equation 5.1) *with $\|\lambda^*\| = l$, $\|x^0 - x^*\| \leq R_0$, g being $\bar{L}_g$-Lipschitz smooth, and f being affine, i.e., $L_f = 0$, such that every first order method in $\mathcal{F}$ requires at least $\Omega(\sqrt{\bar{L}_g c}R_0/\sqrt{\epsilon})$ queries to the first order oracle to find an $(\epsilon; \epsilon/c)$-optimal solution.*

Proof: Consider the following function-constrained problem parameterized by $\gamma > 0$, $\beta > 0$, and $k \in \mathbb{N}_+$:

$$\begin{aligned}
&\min_{x\in\mathbb{R}^{2k+1}} -2l\gamma\beta x_1 \\
&\quad s.t.\ g_1(x) := \beta[x_1^2 + \textstyle\sum_{i=1}^m [2k](x_i - x_{i+1})^2 + x_{2k+1}^2] - (\frac{2k+1}{2k+2})\gamma^2\beta \leq 0 \\
&\qquad g_2(x) := \beta[-2x_1\gamma + x_1^2 + \textstyle\sum_{i=1}^m [2k](x_i - x_{i+1})^2 + x_{2k+1}^2] + (\frac{2k+1}{2k+2})\gamma^2\beta \leq 0,
\end{aligned} \tag{5.42}$$

where l is the given parameter in the theorem statement. Without loss of generality, we take $x^0 = 0$ and $\mathcal{M}_0 = \{0\}$. Let $\mathcal{K}_i$ denote the subspace with non-zeros in only the first i^{th} coordinates, i.e., $\{x \in \mathbb{R}^{2k+1} : x_j = 0 \ \ \forall j > i\}$. Given a first-order method satisfying (Equation 5.40), it is easy to show inductively that $\mathcal{M}_t \subset \mathcal{K}_t \ \forall t \in [2k+1]$. This is because $\mathcal{G}_i = \{\nabla f(\bar{x}), \nabla g_1(\bar{x}), \nabla g_2(\bar{x}) : \bar{x} \in \mathcal{K}_i\}$ are non-zero only in the first $i+1$ coordinates, i.e., $\mathcal{G}_i \subset \mathcal{K}_{i+1}$. Thus in k iterations, we have the following lower bound on feasibility violation:

$$\left\|[g(x^k)]_+\right\| \geq \min_{x\in\mathcal{K}_k} g_2(x) \geq (\tfrac{1}{2k+2})\beta\gamma^2. \tag{5.43}$$

Now we calculate the problem parameters associated with (Equation 5.42). It is straightforward to verify via the KKT condition that the optimal solution and the optimal

dual multiplier are respectively:

$$\lambda^* = [l, 0],\ x_i^* = \gamma[1 - \tfrac{i}{2k+2}]\ \forall i \in [2k+1].$$

So $\|x^0 - x^*\| \leq \gamma\sqrt{k+1}$, $\|\lambda^*\| = l$, and the constraint function g have a smoothness constant of 12β. By selecting $k = \lfloor \frac{1}{5}\sqrt{\frac{\bar{L}_g c}{\epsilon}}R_0 \rfloor - 1$, $\beta = \bar{L}_g/12$, $\gamma = R_0/\sqrt{k+1}$, the problem satisfies the requirements in the theorem statement. Moreover, (Equation 5.43) implies that in $k = \lfloor \frac{1}{5}\sqrt{\frac{\bar{L}_g c}{\epsilon}}R_0 \rfloor - 1 = \Omega(\sqrt{\frac{\bar{L}_g c}{\epsilon}}R_0)$ iterations, the feasibility violation are lower bounded by

$$\left\|[g(x^k)]_+\right\| \geq (\tfrac{1}{2k+2})\beta\gamma^2 = (\tfrac{1}{24})(12\beta)[\gamma^2(k+1)][\tfrac{1}{(k+1)^2}] \geq (\tfrac{1}{24})(\bar{L}_g)R_0^2\tfrac{25\epsilon}{\bar{L}_g c R_0^2} \geq \tfrac{\epsilon}{c}.$$

This shows that $\Omega(\sqrt{\frac{\bar{L}_g c}{\epsilon}}R_0)$ iterations are necessary for finding an $(\epsilon, \epsilon/c)$-optimal solution. Since the choice among $\mathcal{F}$ is arbitrary, the lower complexity bound is valid for all first-order methods in $\mathcal{F}$. □

Comparing the above lower bound of $\Omega\{\sqrt{\frac{\bar{L}_g c}{\epsilon}}\,\|x^* - x^0\|\}$ to the upper bound of $\mathcal{O}\{\sqrt{\frac{L(\Lambda_c)}{\epsilon}}\,\|x^* - x^0\|\}$ from Corollary 5.2, we see that the dependences of the oracle complexity of the ACGD method on ϵ and $\|x^* - x^0\|$ are not improvable. Only the dependence on the smoothness constant $\bar{L}_g$ might be sub-optimal. Specifically, the (big-O) factor of sub-optimality can be characterized by the following function of c:

$$H(c) := \mathcal{O}(\sqrt{\tfrac{L(\Lambda_c)}{\bar{L}_g c}}).$$

Since, in this section, $L_f = 0$ such that $L(\Lambda_c) \leq \bar{L}_g(\|\lambda^*\| + c)$ (see (Equation 5.17)), the big-O relationship between $L(\Lambda_r)$ of the upper complexity bound and the $c\bar{L}_g$ of the lower complexity bound is shown in Figure Figure 5.1.

Clearly $H(c)$ has two distinct regions. When $c \in [1, \|\lambda^*\|]$, we have $1 \leq H(c) \leq$

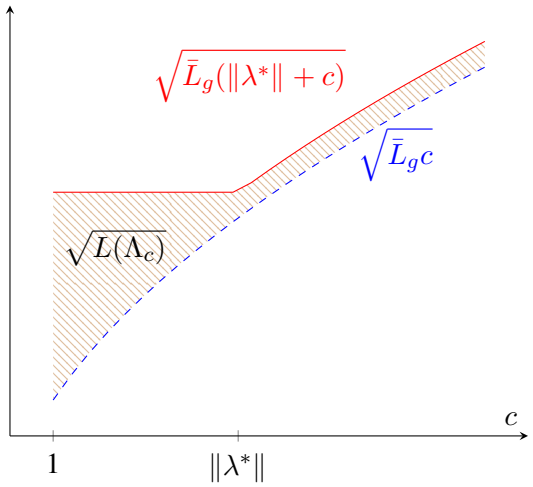


Figure 5.1: (Big-O) Dependence of Complexities on c in Function Constrained Optimization

$\sqrt{\|\lambda^*\|/c}$, i.e., the oracle complexity of the ACGD method can be sub-optimal up to a factor of $\sqrt{\|\lambda^*\|/c}$. In practice, the factor $H(c)$ can be smaller. In fact, when the constraint functions have imbalanced Lipschitz-smoothness constants, $H(c)$ could be 1 so that the ACGD method is optimal. For instance, consider a slightly modified version of (Equation 5.42):

$$\min_{x\in\mathbb{R}^{2k+1}} \{-2l\gamma\beta x_1 \text{ s.t. } g_1(x)\le 0, lg_2(x)\le 0\},$$

where the second constraint g_2 is multiplied by l. Since the constraint $lg_2(x)\le 0$ is not active, the above modified problem still has the same x^* and $\lambda^*=[l,0]$ as (Equation 5.42). Since the smoothness constants $\bar{L}_g$ is increased from $\mathcal{O}(\beta)$ to $\mathcal{O}(l\beta)$, $c\bar{L}_g=\mathcal{O}(lc\beta)$ and $L(\Lambda_c)=\mathcal{O}[(1+c)l\beta]$ are now of the same order such that $H(c)=1$.

When $c\ge\|\lambda^*\|$, we have $H(c)=1$ such that the ACGD method is always optimal. Since λ^* presents the shadow price of the constraints close to x^*, the cost of changing an (ϵ/c)-feasible solution, i.e. $g_i(x^t)\le\tilde{\epsilon}_i$, to feasibility is roughly

$$\max_{\|\tilde{\epsilon}\|\le\epsilon/c} \textstyle\sum_{i=1}^m \lambda^*\tilde{\epsilon}_i = \|\lambda^*\|\,\epsilon/c.$$

This shows that a scaling constant $c \geq \|\lambda^*\|$ should be selected to ensure that the (ϵ/c)-feasibility requirement is comparable to the ϵ-optimality requirement. In this case, the proposed ACGD method is optimal.

5.4 The ACGD-S method

We extend the ACGD method to the ACGD with sliding (ACGD-S) method to handle the large-scale problem where both the problem dimension n and the number of constraints m are large. This section follows the same structure as Section section 5.2. We first discuss the computation bottleneck in the large-scale setting. Then Subsection subsection 5.4.1 introduces the ACGD-S method and presents the convergence results, Subsection subsection 5.4.2 proposes a guess-and-check search scheme to look for some elusive problem parameters important for stepsizes selection, and Subsection subsection 5.4.3 contains the detailed proofs to the convergence results.

Despite its optimal oracle complexities, the ACGD method may be lacking in computation efficiency for the large-scale problem. The bottleneck to Algorithm Algorithm 11 lies in Line 5:

$$\begin{aligned} x^t \leftarrow \arg\min_{x \in X} & \langle p_i^t, x \rangle + u(x) + \eta_t \left\| x - x^{t-1} \right\|^2 /2 \\ s.t.\ & \nu_i^t (x - \underline{x}^t) + g_i(\underline{x}^t) \leq 0 \ \ \forall i \in [m]. \end{aligned} \tag{5.44}$$

It amounts to a large-scale quadratic program (QP) if X is linearly constrained, say a box, and a large-scale quadratic constrained quadratic program (QCQP) if X is a Euclidean ball.

In this section, we address the bottleneck by replacing the large-scale QP with a sequence of basic matrix-vector operations, each requiring at most $\mathcal{O}(mn)$ FLOPs. The proposed ACGD-S method requires only a similar number of matrix-vector operations as solving a single linear constrained problem, i.e., $g(x)$ is affine, and maintains the same optimal oracle complexity as the ACGD method. Towards that end, we need to assume the

projection onto X is easy, i.e., the following operation can be computed in $\mathcal{O}(n)$ FLOPs for any $\pi, \bar{x} \in \mathbb{R}^n$ and $\eta_t \geq 0$.

$$x^t \leftarrow \arg\min_{x \in X} \langle \pi, x \rangle + u(x) + \eta_t \left\| x - \bar{x} \right\|^2 / 2. \tag{5.45}$$

For instance, if $u(x) = \alpha \left\| x \right\|^2 / 2$, the computation simplifies to component-wise thresholding if X is a box,and to vector scaling if X is a Euclidean ball. If X is more challenging, we can model the complicated part using the function constraints.

Algorithm 13 The ACGD-S Method

Input: $x^{-1} = \underline{x}^0 = y_0^{(1)} = x^0 \in X$, stepsizes $\{\theta_t\}$, $\{\eta_t\}$, $\{\tau_t\}$, and weights $\{\omega_t\}$.

1: Set $p_i^0 = \nabla f(x^0)$, $\nu^0 = \nabla g(x^0)$, $\lambda_{-1}^{(1)} = \lambda_0^{(1)} = 0$.
2: **for** $t = 1, 2, 3...N$ **do**
3: Set $\underline{x}^t \leftarrow (\tau_t \underline{x}^{t-1} + \tilde{x}^t)/(1 + \tau_t)$ where $\tilde{x}^t = x^{t-1} + \theta_t(x^{t-1} - x^{t-2})$.
4: Set $p_i^t \leftarrow \nabla f(\underline{x}^t)$ and $\nu^t \leftarrow \nabla g(\underline{x}^t)$.
5: Calculate inner loop iteration limit $\{S_t\}$ stepsizes $\{\beta_s^{(t)}\}$ and $\{\gamma_s^{(t)}\}$, and weights $\{\delta_s^{(t)}\}$.
6: **for** $s = 1, 2, ..., S_t$ **do**
7: Set $\tilde{h}^{(t),s} = \begin{cases} (\nu^t)^\top \lambda_0^{(t)} + \rho_1^{(t)} (\nu^{t-1})^\top (\lambda_0^{(t)} - \lambda_{-1}^{(t)}) & \text{if } s = 1 \\ (\nu^t)^\top \lambda_{s-1}^{(t)} + \rho_s^{(t)} (\nu^t)^\top (\lambda_{s-1}^{(t)} - \lambda_{s-2}^{(t)}) & \text{o.w.} \end{cases}$
8: Set $y_s^{(t)} \leftarrow \arg\min_{y \in X} \langle \tilde{h}^{(t),s} + p_i^t, y \rangle + u(y) + \eta_t \left\| y - x^{t-1} \right\|^2 / 2 + \beta_s^{(t)} \left\| y - y_{s-1}^{(t)} \right\|^2 / 2.$
9: Set $\lambda_s^{(t)} \leftarrow \arg\max_{\lambda \in \mathbb{R}_+^m} \langle \lambda, \nu^t (y_s^{(t)} - \underline{x}^t) + g(\underline{x}^t) \rangle + \gamma_s^{(t)} \left\| \lambda - \lambda_{s-1}^{(t)} \right\|^2 / 2.$
10: **end for**
11: Set $\lambda_0^{(t+1)} = \lambda_{S_t}^{(t)}$, $\lambda_{-1}^{(t+1)} = \lambda_{S_t - 1}^{(t)}$, $y_0^{(t+1)} = y_{S_t}^{(t)}$
12: Set $x^t = \sum_{s=1}^{S_t} \delta_s^{(t)} y_s^{(t)} / (\sum_{s=1}^{S_t} \delta_s^{(t)})$ and $\tilde{\lambda}^t = \sum_{s=1}^{S_t} \delta_s^{(t)} \lambda_s^{(t)} / (\sum_{s=1}^{S_t} \delta_s^{(t)})$.
13: **end for**
14: **return** $\bar{x}^N := \sum_{t=1}^{N} \omega_t x^t / (\sum_{t=1}^{N} \omega_t)$.

5.4.1 The ACGD-S Method and its Convergence Results

The ACGD-S method, listed in Algorithm Algorithm 13, consisted of two loops. For clarity, we will call an outer iteration a phase, and an inner iteration an iteration for the rest of rest section. In phase t, the outer loop updates happen in Lines 3, 4, and 12; they are iden-

tical to the ACGD method except that the exact solution x^t to the QP in (Equation 5.44) is replaced by some average of inner iterates. The others steps, Lines 5-11, constitute the sliding subroutine. Its goal is to solve the Lagrangian reformulation to (Equation 5.44), or the (x^t, λ^t) saddle point problem in (Equation 5.16), inexactly:

$$(x^t, \lambda^t) \leftarrow \arg\min_{y \in X} \arg\max_{\lambda \in \mathbb{R}^m_+} \langle p_i^t, y \rangle + \langle \lambda, \nu^t y - \underline{x}^t \rangle + u(y) + \eta_t \left\| y - x^{t-1} \right\|^2 / 2. \quad (5.46)$$

To avoid confusion, we use the dummy variable y to emphasize it being used only in the inner loop. Specifically, Line 5 calculates the stepsize parameters and iteration number S_t. Lines 7-9 carry out primal-dual type updates for S_t iterations. Line 7 computes a momentum extrapolation term $\tilde{h}^{(t),s}$ as a proxy for $(\nu^t)^\top \lambda_s^{(t)}$. In Line 8, with the variable $(\nu^t)^\top \lambda$ being fixed to $\tilde{h}^{(t),s}$, $y_s^{(t)}$ is generated by minimizing the variable y in (Equation 5.46) subject to a prox-function $\beta_s^{(t)} \left\| y - y_{s-1}^{(t)} \right\|^2$. Then in Line 9, with the variable y fixed to $y_s^{(t)}$, $\lambda_s^{(t)}$ is generated by maximizing (Equation 5.46) with subject to a prox-function $\gamma_s^{(t)} \left\| \lambda - \lambda_{s-1}^{(t)} \right\|^2$. After that, Line 11 prepares the initialization points for the inner loop in the next phase.

We highlight three features that are essential for achieving the desired computation efficiency. First, rather than being pre-specified, the inner loop step-size parameters and iteration limit S_t are calculated in an online fashion in Line 5. This allows the method to adjust dynamically to the varying difficulty of the saddle point problems (Equation 5.46) from different phases. Second, the last operator ν^{t-1}, rather than ν^t, is used for calculating the momentum extrapolation term at the first iteration $s = 1$ in Line 7. This is characteristic of the sequential dual type algorithms [59, 83, 55] for solving the trilinear saddle point problem in (Equation 5.11). Third, two primal iterates, x^t and $y_{S_t}^{(t)}$, are stored after each inner loop to kick-start the next one. This is common to sliding-type algorithms [87, 88, 59].

Now we suggest certain stepsize choices to obtain concrete convergence rates for Al-

gorithm Algorithm 13. The non-strongly convex case and the strongly convex case are presented in separate theorems.

Theorem 5.5 *Consider a non-strongly problem of form* (Equation 5.1) *with* $\alpha = 0$. *Let the aggregate smoothness constant* $L(\Lambda_c)$ *and the reference multiplier set* Λ_c *be defined in* (Equation 5.17) *and* (Equation 5.13) *respectively. Suppose Algorithm Algorithm 13 is performed with the following stepsizes. The outer-loop stepsizes are*

$$\tau_t = \tfrac{t-1}{2}, \eta_t = \tfrac{L(\Lambda_c)}{\tau_{t+1}}, \omega_t = t, \theta_{t+1} = \omega_{t+1}/\omega_t \ \ \forall t \geq 1. \tag{5.47}$$

With $M_t = \|\nu^t\|$, $d(\Lambda_r) = \|\lambda^*\| + r$, *and some* $\Delta > 0$, *the inner loop parameters in the phase* t *are calculated according to*

$$\begin{aligned} S_t = \lceil M_t \Delta t \rceil, \tilde{M}_t = \tfrac{S_t}{\Delta t}, \ \rho_s^{(t)} = \begin{cases} \tilde{M}_t / \tilde{M}_{t-1} & \text{if } s = 1 \\ 1 & \text{if } s \geq 2, \end{cases} \\ \beta_s^{(t)} = \tfrac{\tilde{M}_t d(\Lambda_c)}{\|x^0 - x^*\|}, \ \gamma_s^{(t)} = \tfrac{\tilde{M}_t^2}{\beta_s^{(t)}}, \ \delta_s^{(t)} = 1 \ \forall s \geq 1, \end{aligned} \tag{5.48}$$

Then $M_t \leq \bar{M} \ \forall t$, *where* $\bar{M}$ *is an upper bound of* $\|\nabla g(x)\|$ *for all* x *in some bounded neighbourhood around* x^*. *Moreover, the ergodic average solution* $\bar{x}^N$ *satisfies*

$$\max\{F(\bar{x}^N) - F_*, c \left\|[g(\bar{x}^N)]_+\right\|\} \leq \tfrac{1}{N(N+1)}(\tfrac{2d(\Lambda_c)\left\|x^0 - x^*\right\|}{\Delta} + L(\Lambda_c)\left\|x^0 - x^*\right\|^2). \tag{5.49}$$

Corollary 5.3 *Under the setting of Theorem Theorem 5.5, if* $\Delta = \frac{d(\Lambda_c)}{\|x^0 - x^*\| L(\Lambda_c)}$, *the total numbers of operations required by the ACGD-S method to find an* $(\epsilon; \epsilon/c)$*-optimal solution are bounded by:*

- $N_\epsilon = \lceil \sqrt{\frac{3L(\Lambda_c)}{\epsilon}} \|x^0 - x^*\| \rceil$ *FO-oracle evaluations.*
- $C_\epsilon = \mathcal{O}\{(\sqrt{\frac{L(\Lambda_c)}{\epsilon}} \|x^0 - x^*\| + \frac{d(\Lambda_c)\bar{M}\left\|x^0 - x^*\right\|}{\epsilon})\}$ *matrix-vector multiplications.*

Three remarks are in order regarding the above results. First, the oracle complexity of the ACGD-S method matches that of the ACGD method, while its computation complexity, measured by the number of matrix-vector multiplications, matches the lower bound for solving a single linearly constrained problem [86]. Second, the stepsize choices in (Equation 5.47) and (Equation 5.48) only require an upper bound to $L(\Lambda_c)$. The misspecification of $\|x^0 - x^*\|$, $d(\Lambda_c)$ or Δ would still lead to an $\mathcal{O}(1/\sqrt{\epsilon})$ oracle complexity and an $\mathcal{O}(1/\epsilon)$ computation complexity. Third, the iteration limit function S_t in (Equation 5.47) adapts to the varying difficulty of the saddle point sub-problem (Equation 5.46) from different phases. Specifically, S_t scales in proportion both to $\|\nu^t\|$, which characterizes the hardness of the saddle point sub-problem in (Equation 5.46), and to t, which captures the degree of accuracy required by the outer loop.

Theorem 5.6 *Consider a strongly convex problem of form* (Equation 5.1) *with* $\alpha > 0$. *Let the aggregate smoothness constant* $L(\Lambda_r)$ *and the reference multiplier set* Λ_r *be defined in* (Equation 5.17) *and* (Equation 5.13) *respectively, and let* $\kappa_r := L(\Lambda_r)/\alpha$ *be the condition number. Suppose Algorithm Algorithm 13 is performed with the following stepsizes. The outer-loop stepsizes are*

$$\tau_t = \min\{\tfrac{t-1}{2}, \sqrt{2\kappa_r}\}, \eta_t = \tfrac{L(\Lambda_c)}{\tau_{t+1}}, \text{ and } \theta_t = \tfrac{\tau_t}{\tau_{t-1}+1} \forall t \geq 1. \quad \omega_t = \begin{cases} \omega_{t-1}/\theta_t & \text{if } t \geq 2, \\ 1 & \text{if } t = 1. \end{cases} \tag{5.50}$$

For the inner loops, let $M_t = \|\nu^t\|$, *and let a parameter* $\Delta > 0$ *be given. At the beginning, the iteration limit is set to* $S_1 = \min\{S \in \mathbb{N}_+ : \sum_{j=1}^{S} s \geq \omega_1 M_1^2 \Delta\}$, *and the step-sizes for all* $s \in [S_1]$ *are set to*

$$\delta_s^{(1)} = \tfrac{s}{\Gamma_1}, \ \beta_s^{(1)} = \tfrac{\alpha}{4}(s-1), \ \gamma_s^{(1)} = \tfrac{4}{\alpha}\tfrac{M_1^2\Gamma_1}{\delta_s^{(1)}}, \text{ and } W_2 = \tfrac{\delta_{S_1}^{(1)}}{M_1}, \tag{5.51}$$

where Γ_1 *is the non-negative root to* $\sum_{s=1}^{S_t}[S_1]s = \Gamma_1^2(\omega_1 M_1^2 \Delta)$. *Then for phase* $t \geq 2$, *the*

iteration limit S_t and the parameter $\Gamma_t \geq 0$ are specified to satisfy

$$\begin{aligned} S_t = &\min\{S \in \mathbb{N}_+ : \textstyle\sum_{s=1}^{S_t}[S_t]W_t M_t + (s-1) \geq \omega_t M_t^2 \Delta\}, \\ &\textstyle\sum_{s=1}^{S_t} \Gamma_t (W_t M_t) + (s-1) = \Gamma_t^2 \omega_t M_t^2 \Delta, \end{aligned} \tag{5.52}$$

and the stepsizes are chosen according to

$$\delta_s^{(t)} = W_t M_t + \tfrac{1}{\Gamma_t}(s-1),\ \gamma_s^{(t)} = \tfrac{4}{\alpha}\tfrac{M_t^2 \Gamma_t}{\delta_s^{(t)}},\ \beta_s^{(t)} = \begin{cases} \frac{\alpha}{4}(W_t \Gamma_t M_t) & \text{if } s = 1, \\ \frac{\alpha}{4}[W_t \Gamma_t M_t + (s-2)] & \text{otherwise,} \end{cases}$$

$$\text{and } W_{t+1} = \tfrac{\delta_{S_t}^{(t)}}{M_t}. \tag{5.53}$$

Then we have $M_t \leq \bar{M}\ \forall t$, where $\bar{M}$ is an upper bound for $\|\nabla g(x)\|$ for x in some bounded neighbourhood around x^*, and the ergodic average solution $\bar{x}^N$ satisfies

$$\begin{aligned} &\max\{F(\bar{x}^N) - F_*, r\left\|[g(\bar{x}^N)]_+\right\|\} \leq \tfrac{1}{\mathcal{W}_N}(\tfrac{2}{\alpha\Delta}[d(\Lambda_r)]^2 + L(\Lambda_r)\left\|x^0 - x^*\right\|^2), \\ &\left\|\bar{x}^N - x^*\right\|^2 \leq \tfrac{2}{\alpha \mathcal{W}_N}(\tfrac{2}{\alpha\Delta}[d(\Lambda_r)]^2 + L(\Lambda_r)\left\|x^0 - x^*\right\|^2), \end{aligned} \tag{5.54}$$

where $d(\Lambda_r) = \|\lambda^*\| + r$ and the denominator satisfies $\mathcal{W}_N \geq \max\{N(N+1)/2, \sqrt{2\kappa_r}[(1+1/\sqrt{2\kappa_r})^{N-4} - 1]\}$.

Corollary 5.4 *Under the setting of Theorem Theorem 5.6, if $\Delta = \frac{[d(\Lambda_r)]^2}{L(\Lambda_r)\|x^0 - x^*\|^2 \alpha}$, the numbers of FO-oracle evaluations N_ϵ and of matrix-vector multiplications C_ϵ required by the ACGD-S method to find an $(\epsilon; \epsilon/c)$-optimal solution are bounded by:*

- $N_\epsilon \leq \min\{\sqrt{\frac{6L(\Lambda_r)\max\{c/r,1\}}{\epsilon}}\left\|x^0 - x^*\right\|^2, [\sqrt{\frac{2L(\Lambda_r)}{\alpha}} + 1]\log(\frac{3\max\{c/r,1\}\sqrt{L(\Lambda_r)\alpha}\left\|x^0 - x^*\right\|^2}{\epsilon} + 1) + 4\} + 1\}$.
- $C_\epsilon = \mathcal{O}\{\sqrt{\frac{\max\{c/r,1\}[d(\Lambda_r)]^2 \bar{M}^2}{\alpha\epsilon}} + N_\epsilon\}$.

Corollary 5.5 *Under the setting of Theorem Theorem 5.6, if the goal is to find an ϵ-close solution, i.e., $\left\|\bar{x}^N - x^*\right\|^2 \leq \epsilon$, we can choose $c = r = 1$ and $\Delta = \frac{[d(\Lambda_1)]^2}{L(\Lambda_1)\|x^0 - x^*\|^2\alpha}$ such that the numbers of required operations can be bounded by*

- $N_\epsilon = \mathcal{O}\{\sqrt{\frac{2L(\Lambda_1)}{\alpha}}\log(\frac{\sqrt{\kappa_1}\left\|x^0 - x^*\right\|^2}{\epsilon})\}$
- $C_\epsilon = \mathcal{O}\{\frac{d(\Lambda_1)\bar{M}}{\alpha\sqrt{\epsilon}} + \sqrt{\frac{2L(\Lambda_1)}{\alpha}}\log(\frac{\sqrt{\kappa_1}\left\|x^0 - x^*\right\|^2}{\epsilon})\}$.

Again, we make a few remarks regarding the results. First, to find an ϵ-close solution, Corollary (5.5) implies that the ACGD-S method has the same oracle complexity as the ACGD method, and has the same computation complexity as that of the lower computation complexity bound for solving a single strongly-convex linearly constrained problem [86]. Second, the iteration limit function S_t is again adaptive to the varying difficulty of the saddle-point subproblem (Equation 5.46) from different phases. Third, the rather complicated inner-loop stepsize choice in (Equation 5.51) and (Equation 5.53) is the first among sliding algorithms, e.g. [87, 8, 88, 59], to achieve both the optimal inner loop complexity of $O(1/\sqrt{\epsilon})$ and the optimal outer loop complexity of $\mathcal{O}(\sqrt{\kappa}\log(1/\epsilon))$ without restarting. It is unclear if the same effect is achievable with simpler stepsize choices. Notwithstanding that, the stepsize is easy to implement in practice because only conservative estimates of α and $L(\Lambda_r)$ are required to obtain the $\mathcal{O}(\sqrt{\kappa_r}\log(1/\epsilon))$ oracle complexity and the $\mathcal{O}(1/\sqrt{\epsilon})$ computation complexity.

5.4.2 The Binary Search for $L(\Lambda_r)$ and $d(\Lambda_r)$

To enhance its implementability, we propose a binary search scheme to find (upper) approximations to these hard-to-estimate problem parameters required by the ACGD-S method. Since both quantities are proportional to Λ_r, it is convenient to look for a common upper bound:

$$\tilde{H} \geq \max\{L(\Lambda_r), d(\Lambda_r)\}.$$

The search scheme has the same logic as Algorithm Algorithm 12: starting with a guess $\tilde{H}$, we first run the ACGD-S method for $N(\tilde{H})$ phases, with the stepsizes specified according to $\tilde{H}$, to generate an ergodic solution $\bar{x}^{N(\tilde{H})}$, then we check it against a certain termination criterion and update our guess $\tilde{H}$ accordingly. The following assumption is required for constructing an easy-to-verify termination criterion.

Assumption 6 *The feasible region X is bounded and we know an estimate D_X of its radius, i.e., $D_X \geq \max_{x,y\in X} \|x - y\|$.*

Compared to Assumption 5, we no longer require access to the dual solution λ^t (Line 5 of Algorithm Algorithm 11). This is because the sliding subroutine (Line 12 of Algorithm Algorithm 13) provides a feasible dual iterate $\tilde{\lambda}^t$ as a by-product. So the new termination criterion, termed (Equation 5.27)-(Equation 5.55), requires only the following modification to (Equation 5.27):

$$\boxed{\text{Change } \lambda_i^t \text{ in (Equation 5.28) to } \tilde{\lambda}_i^t.} \tag{5.55}$$

Since $\tilde{\lambda}^t$ from Line 12 of Algorithm Algorithm 13 is dual feasible, the minimum value $\underline{F}^N$ to the modified linear program in (Equation 5.28) (with $\tilde{\lambda}^t$ in place of λ^t) still bounds F_* from below. The satisfaction of new termination criterion (Equation 5.27)-(Equation 5.55) then implies that $\bar{x}^{N(\tilde{H})}$ is $(\epsilon; \epsilon/c)$-optimal. Accordingly, we modify Algorithm Algorithm 12 by switching to the ACGD-S method and to the termination criterion (Equation 5.27)-(Equation 5.55) to arrive at Algorithm Algorithm 14.

Algorithm 14 Binary Search Outer Loop to ACGD-S.

Input: $\bar{x} \in X$, $\tilde{H} > 0$, and a phase-limit function $N(H)$.

while true **do**
  With D_X, $\tilde{H}$, and $\tilde{H}$ in place of $\|x^0 - x^*\|$, $L(\Lambda_r)$, and $d(\Lambda_r)$, set step-sizes to (Equation 5.48) and (Equation 5.47) if $\alpha = 0$, and to (Equation 5.50), (Equation 5.51) and (Equation 5.53) if $\alpha > 0$.
  Run the ACGD-S method for $N(\tilde{H})$ phases starting from $\bar{x}$.
  Set $\bar{x} \leftarrow \bar{x}^{N(\tilde{H})}$, where $\bar{x}^{N(\tilde{H})}$ is the ergodic average solution.
  if $\bar{x}$ satisfies the condition (Equation 5.27)-(Equation 5.55) **then**
    break
  else
    Set $\tilde{H} \leftarrow 2\tilde{H}$.
  end if
end while
return $\bar{x}$.

Clearly, the above algorithm terminates with an $(\epsilon, \epsilon/c)$-optimal solution. We only need to specify the phase-limit function $N(\tilde{H})$ to make it implementable. The next theorem provides some choices of $N(\tilde{H})$ and the corresponding complexities. Its derivation is similar to Theorem Theorem 5.2

Theorem 5.7 *Consider a problem of form* (Equation 5.1). *Let Assumption 6 hold and* $\bar{M} := \max_{x \in X} \|\nabla g(x)\|$. *Suppose the reference set* Λ_r *is defined according to* (Equation 5.13). *Let* $L(\Lambda_r)$ *denote the aggregate smoothness constant* (Equation 5.17) *and* $d(\Lambda_r) = \|\lambda^*\| + r$. *If* $\tilde{H}$ *is initialized to some* $\tilde{H}_0$, *Algorithm Algorithm 14 then terminates finitely with an* $(\epsilon; \epsilon/c)$*-optimal solution under the following choices of phase limit function* $N(\tilde{H})$.

a) In the non-strongly convex case with $\alpha = 0$, *if* $N(\tilde{H}) = \lceil \sqrt{\frac{3\tilde{H}}{\epsilon}} D_X \rceil$, *the total numbers of oracle evaluations (phases) and matrix-vector multiplications required for*

termination are bounded respectively by

$$\mathcal{O}(\sqrt{\tfrac{\max\{L(\Lambda_c),d(\Lambda_c),\tilde{H}_0\}}{\epsilon}}D_X), \text{ and } \mathcal{O}(\tfrac{\max\{L(\Lambda_c),d(\Lambda_c),\tilde{H}_0\}D_X\bar{M}}{\epsilon}).$$

b) In the strongly convex case with $\alpha > 0$, *if* $N(\tilde{H}) = \lceil(\sqrt{\frac{2\tilde{H}}{\alpha}} + 1)\log(\frac{3\max\{c/r,1\}\sqrt{\tilde{H}\alpha}D_X^2}{\epsilon}+1)\rceil + 4$, *the total numbers of oracle evaluations (phases) and matrix-vector multiplications required for termination are bounded respectively by*

$$\mathcal{O}(\sqrt{\tfrac{\max\{\tilde{H}_0,L(\Lambda_1),d(\Lambda_1)\}}{\alpha}}\log\tfrac{c\max\{\tilde{H}_0,L(\Lambda_1),d(\Lambda_1)\}}{\epsilon}), \text{ and } \mathcal{O}(\tfrac{\sqrt{c}\max\{\tilde{H}_0,L(\Lambda_1),d(\Lambda_1)\}\bar{M}D_X}{\sqrt{\alpha\epsilon}}).$$

We remark that both the oracle complexity and the computation complexity across the multiple ACGD-S runs in Algorithm Algorithm 14 have the optimal order for all $\alpha \geq 0$. If the Lipschitz smoothness constants L_f and L_g for f and g are known, we can search instead for $d(\Lambda_r) = \|\lambda^*\| + r$ and utilize $L_f + d(\Lambda_r)L_g$ as an upper bound to $L(\Lambda_r)$. The resulting search scheme would have the same complexities, with matching constant dependences, as those of Corollary 5.3 and 5.4.

5.4.3 The Convergence Analysis

We first prove a generic result for the Q-function (Equation 5.14) useful for both the non-strongly convex and the strongly convex cases.

Proposition 5.2 *Consider an α-strongly convex problem of form* (Equation 5.1). *Let a set of reference multipliers $\Lambda \in \mathbb{R}^m_+$ be given and let the aggregate smoothness constant $L(\Lambda)$ be defined in* (Equation 5.17). *Let iterates $z^t := \{x^t; \tilde{\lambda}^t, \nu^t, p_i^t\}$ be generated by Algorithm Algorithm 13. Suppose the following stepsize requirements are met. For all $t \geq 1$, the*

outer-loop stepsize requirements are

$$\begin{aligned}
&\omega_t \eta_t \leq \omega_{t-1}(\eta_{t-1} + \alpha/2), \\
&\omega_t \tau_t \leq \omega_{t-1}(\tau_{t-1} + 1), \\
&\eta_{t-1}\tau_t \geq \theta_t L(\Lambda) \text{ with } \theta_t := \omega_{t-1}/\omega_t, \\
&\eta_N(\tau_N + 1) \geq L(\Lambda).
\end{aligned} \tag{5.56}$$

For all $t \geq 1, s \geq 1$, the intra-phase stepsize requirements are

$$\begin{aligned}
&\delta_s^{(t)}(\beta_s^{(t)} + \alpha/2) \geq \delta_{s+1}^{(t)}\beta_{s+1}^{(t)}, \\
&\delta_s^{(t)}\gamma_s^{(t)} \geq \delta_{s+1}^{(t)}\gamma_{s+1}^{(t)}, \\
&\gamma_s^{(t)}\beta_{s+1}^{(t)} \geq \rho_{s+1}^{(t)} \left\|\nu^t\right\|^2, \ \rho_{s+1}^{(t)} = \delta_s^{(t)}/\delta_{s+1}^{(t)}, \\
&\gamma_{S_N}^{(N)}(\beta_{S_N}^{(N)} + \alpha/2) \geq \left\|\nu^N\right\|^2.
\end{aligned} \tag{5.57}$$

For all $t \geq 1$, the inter-phase requirements are

$$\begin{aligned}
&\tilde{w}_{S_t}^{(t)}(\beta_{S_t}^{(t)} + \alpha/2) \geq \tilde{w}_1^{(t+1)}\beta_1^{(t+1)}, \\
&\tilde{w}_{S_t}^{(t)}\gamma_{S_t}^{(t)} \geq \tilde{w}_1^{(t+1)}\gamma_1^{(t+1)}, \\
&\gamma_{S_t}^{(t)}\beta_1^{(t+1)} \geq \rho_1^{(t+1)} \left\|\nu^{t-1}\right\|^2, \ \rho_1^{(t+1)} = \tilde{w}_{S_t}^{(t)}/\tilde{w}_1^{(t+1)},
\end{aligned} \tag{5.58}$$

where $\tilde{w}_s^{(t)} := \omega_t \delta_s^{(t)}/(\sum_{s=1}^{S_t} \delta_s^{(t)})$ denotes the aggregate weights. Then for any reference point $z = (x; \lambda, \nu, \pi) \in X \times \Lambda \times [V, \Pi]$ with $[V, \Pi]$ being defined in (Equation 5.10)*, we have*

$$\begin{aligned}
\textstyle\sum_{t=1}^N \omega_t Q(z^t; z) + \frac{\omega_N}{2}(\eta_N + \frac{\alpha}{2}) \left\|x^N - x\right\|^2 \leq \frac{\tilde{w}_1^{(1)}\beta_1^{(1)} + \omega_1\eta_1}{2} \left\|x^0 - x\right\|^2 \\
+ \frac{\tilde{w}_1^{(1)}\gamma_1^{(1)}}{2} \left\|\lambda_0^{(1)} - \lambda\right\|^2 + \omega_1\tau_1[U_{f^*}(\pi; p_i^0) + \lambda^\top U_{g^*}(\nu; \nu^0)].
\end{aligned} \tag{5.59}$$

Proof: We first establish a convergence bound the inner loop within a phase. Fix $t \geq 1$. Consider the convergence of $y_s^{(t)}$. Since $u(y) + \eta_t \left\|y - x^{t-1}\right\|^2/2$ has a strong convexity

modulus of $\alpha + \eta_t$, the y-prox mapping in Line 8 of Algorithm Algorithm 13 leads to a three point inequality (see Lemma 3.1 of [28]):

$$\begin{aligned}&\langle y_s^{(t)} - x, \tilde{h}^{(t),s}\rangle + u(y_s^{(t)}) - u(x) + \tfrac{\eta_t}{2}(\left\|y_s^{(t)} - x^{t-1}\right\|^2 - \left\|x - x^{t-1}\right\|^2)\\ &\quad \tfrac{1}{2}[(\beta_s^{(t)} + \alpha + \eta_t)\left\|x - y_s^{(t)}\right\|^2 + \beta_s^{(t)}\left\|y_s^{(t)} - y_{s-1}^{(t)}\right\|^2 - \beta_s^{(t)}\left\|y_{s-1}^{(t)} - x\right\|^2] \leq 0.\end{aligned}$$

Equivalently, we have

$$\begin{aligned}&\langle y_s^{(t)} - x, \tilde{h}^{(t),s}\rangle + \tfrac{1}{2}[(\beta_s^{(t)} + \alpha/2)\left\|x - y_s^{(t)}\right\|^2 + \beta_s^{(t)}\left\|y_s^{(t)} - y_{s-1}^{(t)}\right\|^2 - \beta_s^{(t)}\left\|y_{s-1}^{(t)} - x\right\|^2]\\ &\quad + u(y_s^{(t)}) - u(x) + \tfrac{\eta_t\left\|y_s^{(t)} - x^{t-1}\right\|^2 + (\eta_t + \alpha/2)\left\|y_s^{(t)} - x\right\|^2 - \eta_t\left\|x - x^{t-1}\right\|^2}{2} \leq 0. \qquad (5.60)\end{aligned}$$

In particular, the definition of $\tilde{h}^{(t),s}$ in Line 7 of Algorithm Algorithm 13 implies

$$\begin{aligned}\langle y_s^{(t)} - x, \tilde{h}^{(t),s}\rangle =&\langle y_s^{(t)} - x, \textstyle\sum_{i=1}^m \lambda_{s,i}^{(t)}\nu_i^t\rangle - \langle y_s^{(t)} - x, \textstyle\sum_{i=1}^m(\lambda_{s,i}^{(t)} - \lambda_{s-1,i}^{(t)})\nu_i^t\rangle\\ &+ \rho_s^{(t)}\langle y_s^{(t)} - y_{s-1}^{(t)}, \textstyle\sum_{i=1}^m(\lambda_{s-1,i}^{(t)} - \lambda_{s-2,i}^{(t)})\nu_i^t\rangle\\ &+ \rho_s^{(t)}\langle y_{s-1}^{(t)} - x, \textstyle\sum_{i=1}^m(\lambda_{s-1,i}^{(t)} - \lambda_{s-2,i}^{(t)})\nu_i^t\rangle,\ \forall s \geq 2,\end{aligned}$$

and

$$\begin{aligned}\langle y_1^{(t)} - x, \tilde{h}^{(t),1}\rangle =&\langle y_1^{(t)} - x, \textstyle\sum_{i=1}^m \lambda_{1,i}^{(t)}\nu_i^t\rangle - \langle y_1^{(t)} - x, \textstyle\sum_{i=1}^m(\lambda_{1,i}^{(t)} - \lambda_{0,i}^{(t)})\nu_i^t\rangle\\ &+ \rho_1^{(t)}\langle y_1^{(t)} - y_0^{(t)}, \textstyle\sum_{i=1}^m(\lambda_{0,i}^{(t)} - \lambda_{-1,i}^{(t)})\nu_i^{t-1}\rangle\\ &+ \rho_1^{(t)}\langle y_0^{(t)} - x, \textstyle\sum_{i=1}^m(\lambda_{0,i}^{(t)} - \lambda_{-1,i}^{(t)})\nu_i^{t-1}\rangle.\end{aligned}$$

So, substituting them into (Equation 5.60), summing up the resulting inequality with weight $\delta_s^{(t)}$, noting the stepsizes conditions in (Equation 5.57), and utilizing Young's inequality, we

get

$$\begin{aligned}
&\sum_{s=1}^{S_t}\delta_s^{(t)}\left(\mathcal{L}(y_s^{(t)};\lambda_s^{(t)},\nu^t,p_i^t)-\mathcal{L}(x;\lambda_s^{(t)},\nu^t,p_i^t)\right)\\
&\quad+\sum_{s=1}^{S_t}\delta_s^{(t)}\tfrac{1}{2}[\eta_t\left\|y_s^{(t)}-x^{t-1}\right\|^2+(\eta_t+\tfrac{\alpha}{2})\left\|y_s^{(t)}-x\right\|^2-\eta_t\left\|x-x^{t-1}\right\|^2]\\
&\quad+\delta_1^{(t)}\rho_1^{(t)}\langle y_0^{(t)}-x,\textstyle\sum_{i=1}^m(\lambda_{0,i}^{(t)}-\lambda_{-1,i}^{(t)})\nu_i^{t-1}\rangle-\delta_{S_t}^{(t)}\langle y_{S_t}^{(t)}-x,\textstyle\sum_{i=1}^m(\lambda_{S_t,i}^{(t)}-\lambda_{S_t-1,i}^{(t)})\nu_i^t\rangle\\
&\leq\textstyle\sum_{s=2}^{S_t}\frac{\delta_{s-1}^{(t)}\gamma_{s-1}^{(t)}}{2}\left\|\lambda_{s-1}^{(t)}-\lambda_{s-2}^{(t)}\right\|^2+\frac{\delta_1^{(t)}\rho_1^{(t)}\left\|\nu^{t-1}\right\|^2}{2\beta_1^{(t)}}\left\|\lambda_0^{(t)}-\lambda_{-1}^{(t)}\right\|^2\\
&\quad-\tfrac{1}{2}[\delta_{S_t}^{(t)}(\beta_{S_t}^{(t)}+\alpha/2)\left\|y_{S_t}^{(t)}-x\right\|^2-\delta_1^{(t)}\beta_1^{(t)}\left\|y_0^{(t)}-x\right\|^2].
\end{aligned}\tag{5.61}$$

Next, consider the convergence of $\lambda_s^{(t)}$. The λ-proximal mapping in Line 9 of Algorithm Algorithm 13 implies

$$\mathcal{L}(y_s^{(t)};\lambda,\nu^t,\pi^t)-\mathcal{L}(y_s^{(t)};\lambda_s^{(t)},\nu^t,\pi^t)+\frac{\gamma_s^{(t)}[\left\|\lambda-\lambda_s^{(t)}\right\|^2+\left\|\lambda_s^{(t)}-\lambda_{s-1}^{(t)}\right\|^2-\left\|\lambda-\lambda_{s-1}^{(t)}\right\|^2]}{2}\leq 0.$$

Due to the stepsize conditions in (Equation 5.57), the $\delta_s^{(t)}$ weighted sum satisfies

$$\begin{aligned}
&\textstyle\sum_{s=1}^{S_t}\delta_s^{(t)}[\mathcal{L}(y_s^{(t)};\lambda,\nu^t,\pi^t)-\mathcal{L}(y_s^{(t)};\lambda_s^{(t)},\nu^t,\pi^t)]+\frac{\gamma_{S_t}^{(t)}\delta_{S_t}^{(t)}}{2}\left\|\lambda-\lambda_{S_t}^{(t)}\right\|^2\\
&\quad+\textstyle\sum_{s=1}^{S_t}\frac{\delta_s^{(t)}\gamma_s^{(t)}}{2}\left\|\lambda_s^{(t)}-\lambda_{s-1}^{(t)}\right\|^2\leq\delta_1^{(t)}\gamma_1^{(t)}\left\|\lambda-\lambda_0^{(t)}\right\|^2.
\end{aligned}$$

Then, combining it with the y convergence bound in (Equation 5.61), we get

$$\begin{aligned}
&\sum_{s=1}^{S_t}\delta_s^{(t)}\left(\mathcal{L}(y_s^{(t)};\lambda,\nu^t,\pi^t)-\mathcal{L}(x;\lambda_s^{(t)},\nu^t,\pi^t)+\right)\\
&\quad+\sum_{s=1}^{S_t}\delta_s^{(t)}\tfrac{1}{2}[\eta_t\left\|y_s^{(t)}-x^{t-1}\right\|^2+(\eta_t+\tfrac{\alpha}{2})\left\|y_s^{(t)}-x\right\|^2-\eta_t\left\|x-x^{t-1}\right\|^2]\\
&\quad+\delta_1^{(t)}\rho_1^{(t)}\langle y_0^{(t)}-x,\textstyle\sum_{i=1}^m\nu_i^{t-1}(\lambda_i^0-\lambda_{-1,i}^{(t)})\rangle-\delta_{S_t}^{(t)}\langle y_{S_t}^{(t)}-x,\textstyle\sum_{i=1}^m\nu_i^t(\lambda_{S_t,i}^{(t)}-\lambda_{S_t-1,i}^{(t)})\rangle\\
&\leq\tfrac{\delta_1^{(t)}\gamma_1^{(t)}}{2}\left\|\lambda-\lambda_0^{(t)}\right\|^2-\tfrac{\delta_{S_t}^{(t)}\gamma_{S_t}^{(t)}}{2}[\left\|\lambda-\lambda_{S_t}^{(t)}\right\|^2+\left\|\lambda_{S_t}^{(t)}-\lambda_{S_t-1}^{(t)}\right\|^2]\\
&\quad+\tfrac{\delta_1^{(t)}\rho_1^{(t)}\left\|\nu^{t-1}\right\|^2}{2\beta_1^{(t)}}\left\|\lambda^0-\lambda_{-1}^{(t)}\right\|^2-\tfrac{1}{2}[\delta_{S_t}^{(t)}(\beta_{S_t}^{(t)}+\alpha/2)\left\|y_{S_t}^{(t)}-x\right\|^2-\delta_1^{(t)}\beta_1^{(t)}\left\|y_0^{(t)}-x\right\|^2].
\end{aligned}$$

Moreover, since $\mathcal{L}(y_s^{(t)};\lambda,\nu^t,\pi^t)$, $\left\|y_s^{(t)}-x^{t-1}\right\|^2$ and $\left\|y_s^{(t)}-x\right\|^2$ are convex with respect to $y_s^{(t)}$ and $\mathcal{L}(x;\lambda_s^{(t)},\nu^t,\pi^t)$ is linear with respect to $\lambda_s^{(t)}$, multiplying both sides by $\omega_t/(\sum_{s=1}^{S_t}\delta_s^{(t)})$ and applying the Jensen's inequality leads to

$$\begin{aligned}
&\omega_t\left(\mathcal{L}(x^t;\lambda,\nu^t,\pi^t)-\mathcal{L}(x;\tilde{\lambda}^t,\nu^t,\pi^t)+\tfrac{\eta_t\left\|x^t-x^{t-1}\right\|^2+(\eta_t+\alpha/2)\left\|x^t-x\right\|^2-\eta_t\left\|x^{t-1}-x\right\|^2}{2}\right)\\
&+\tilde{w}_1^{(t)}\rho_1^{(t)}\langle y_0^{(t)}-x,\textstyle\sum_{i=1}^m\nu_i^{t-1}(\lambda_i^0-\lambda_{-1,i}^{(t)})\rangle-\tilde{w}_{S_t}^{(t)}\langle y_{S_t}^{(t)}-x,\textstyle\sum_{i=1}^m\nu_i^t(\lambda_{S_t,i}^{(t)}-\lambda_{S_t-1,i}^{(t)})\rangle\\
&\leq\tfrac{\tilde{w}_1^{(t)}\gamma_1^{(t)}}{2}\left\|\lambda-\lambda_0^{(t)}\right\|^2-\tfrac{\tilde{w}_{S_t}^{(t)}\gamma_{S_t}^{(t)}}{2}[\left\|\lambda-\lambda_{S_t}^{(t)}\right\|^2+\left\|\lambda_{S_t}^{(t)}-\lambda_{S_t-1}^{(t)}\right\|^2]\\
&+\tfrac{\tilde{w}_s^{(t)}\rho_1^{(t)}\left\|\nu^{t-1}\right\|^2}{2\beta_1^{(t)}}\left\|\lambda^0-\lambda_{-1}^{(t)}\right\|^2-\tfrac{1}{2}[\tilde{w}_{S_t}^{(t)}(\beta_{S_t}^{(t)}+\alpha/2)\left\|y_{S_t}^{(t)}-x\right\|^2-\tilde{w}_1^{(t)}\beta_1^{(t)}\left\|y_0^{(t)}-x\right\|^2],
\end{aligned}$$

where $\tilde{w}_s^{(t)}=\omega_t\delta_s^{(t)}/(\sum_{s=1}^{S_t}\delta_s^{(t)})$ represents the aggregate weight for the inner iterates.

Next, we consider the inner loops from different phases. The inter-phase stepsize condition in (Equation 5.58) implies the sum of preceding inequality across t satisfies

$$\begin{aligned}
&\sum_{t=1}^N\omega_t[Q_x(z^t;z)+Q_\lambda(z^t;z)]+\sum_{t=1}^N\tfrac{\omega_t}{2}\eta_t\left\|x^t-x^{t-1}\right\|^2+\omega_N(\eta_N+\alpha/2)\left\|x^N-x\right\|^2\\
&\qquad\leq\tfrac{\tilde{w}_1^{(1)}}{2}(\gamma_1^{(1)}\left\|\lambda_1^{(0)}-\lambda\right\|^2+\beta_1^{(1)}\left\|y_0^{(1)}-x\right\|^2)+\tfrac{\omega_1\eta_1}{2}\left\|x^0-x\right\|^2.
\end{aligned}\tag{5.62}$$

Notice (Equation 5.62) is almost identical to the Q_x and Q_λ inequality in (Equation 5.33).

Thus a similar argument to Proposition 5.1 and the outer-loop stepsize requirements in (Equation 5.56) leads to the desired convergence result in Equation 5.59. □

Now we leverage the preceding proposition to prove the convergence of the ACGD-S method under the non-strongly convex setting.

Proof to Theorem Theorem 5.5 and Corollary 5.3: It is straightforward to verify that the outer loop stepsize in (Equation 5.47) satisfies the condition (Equation 5.56), and the adaptive inner loop stepsize in (Equation 5.48) satisfies both the intra-phase condition (Equation 5.57) and the inter-phase condition (Equation 5.58). So it follows from (Equation 5.59) that

$$\sum_{t=1}^{N} \omega_t Q(z^t; z) + \frac{\omega_N \eta_N}{2} \left\|x^N - x\right\|^2 \leq \frac{\tilde{w}_1^{(1)} \beta_1^{(1)} + \omega_1 \eta_1}{2} \left\|x^0 - x\right\|^2 + \frac{\tilde{w}_1^{(1)} \gamma_1^{(1)}}{2} \left\|\lambda_0^{(1)} - \lambda\right\|^2. \tag{5.63}$$

Now consider setting the reference point to $z^* = (x^*; \lambda^*, \nu^* = \nabla g(x^*), \pi^* = \nabla f(x^*))$ such that $Q(z^t, z^*) \geq 0 \; \forall t$. The preceding inequality implies that

$$\left\|x^N - x^*\right\|^2 \leq \frac{1}{L(\Lambda_c)} [(\tilde{w}_1^{(1)} \beta_1^{(1)} + \omega_1 \eta_1) \left\|x^0 - x\right\|^2 + \tilde{w}_1^{(1)} \gamma_1^{(1)} \left\|\lambda_0^{(1)} - \lambda\right\|^2] \; \forall N \geq 1.$$

So $\underline{x}^t$, being the convex combination of $\{x^t\}$s, remains in a bounded ball around x^*, and $\nu^t = \nabla g(\underline{x}^t)$ is bounded for all $t \geq 1$.

Next, setting the ergodic average solution as $\bar{z}^N := (\bar{x}^N; \sum_{t=1}^{N} \omega_t \tilde{\lambda}^t / (\sum_{t=1}^{N} \omega_t), \bar{\nu}^N, \bar{\pi}^N)$ with

$$\bar{\pi}^N := \sum_{t=1}^{N} \omega_t p_i^t / (\sum_{t=1}^{N} \omega_t), \bar{\nu}_i^N := \begin{cases} \sum_{t=1}^{N} \omega_t \lambda_i^t \nu_i^t / (\sum_{t=1}^{N} \omega_t \lambda_i^t) & \text{o.w.} \\ \nabla g_i(x^0) & \text{if } \lambda_i^t = 0 \; \forall t, \end{cases} \tag{5.64}$$

a similar application of the Jensen's inequality as (Equation 5.35) and the stepsize choice

in (Equation 5.47) and (Equation 5.48) lead to, for all $\lambda \in \Lambda_c, (\nu, \pi) \in [V, \Pi]$

$$Q(\bar{z}^N; (x^*; \lambda, \nu, \pi)) \leq \frac{1}{N(N+1)}\left(\frac{2d(\Lambda_c)\left\|x^0 - x^*\right\|}{\Delta} + L(\Lambda_c)\left\|x^0 - x^*\right\|^2\right). \tag{5.65}$$

The convergence in both the optimality gap and the feasibility violation in (Equation 5.49) then follows from Lemma 5.1.

Consider now the given choice of Δ in Corollary 5.3. It follows from (Equation 5.49) that at most $N_\epsilon = \lceil \sqrt{\frac{3L(\Lambda_c)}{\epsilon}} \left\|x^0 - x^*\right\| \rceil$ phases are required to find an $(\epsilon; \epsilon/c)$-optimal solution. Since each phase requires one gradient evaluation for f and g, the oracle complexity of the ACGD-S method is N_ϵ. Moreover, since each inner iteration requires less than three matrix-vector multiplication, the total number of matrix-vector multiplication across N_ϵ phases can bounded as

$$C_\epsilon = 3\sum_{t=1}^{N_\epsilon} S_t \leq N_\epsilon + N_\epsilon^2 \bar{M}\Delta = \mathcal{O}\{\sqrt{\frac{L(\Lambda_c)}{\epsilon}}\left\|x^0 - x^*\right\| + \frac{d(\Lambda_c)\bar{M}\left\|x^0 - x^*\right\|}{\epsilon}\}.$$

□

The next proof considers the strongly convex case.

Proof to Theorem Theorem 5.6, Corollary 5.4 and Corollary 5.5: It is straightforward to check that the outer-loop stepsize in (Equation 5.50) satisfies the condition (Equation 5.56), and the adaptive inner-loop stepsize in (Equation 5.51) and (Equation 5.53) satisfy the intra-phase condition (Equation 5.57). Now we verify the inter-phase condition in (Equation 5.58). Consider a fixed $t \geq 2$, we have

$$\begin{aligned}
\tilde{w}_1^{(t)} &= \frac{\omega_t \delta_1^{(t)}}{\sum_{s=1}^{S_t} \delta_s^{(t)}} = \frac{\omega_t \delta_1^{(t)}}{\omega_t M_t^2 \Gamma_t \Delta} = \frac{\delta_1^{(t)}}{M_t^2 \Gamma_t \Delta} = \frac{W_t}{M_t \Gamma_t \Delta}, \\
\tilde{w}_{S_{t-1}}^{(t-1)} &= \frac{\omega_{t-1}\delta_{S_{t-1}}^{(t-1)}}{\sum_{s=1}^{S_t}[S_{t-1}]\delta_s^{(t-1)}} = \frac{\omega_{t-1}\delta_{S_{t-1}}^{(t-1)}}{\omega_{t-1} M_{t-1}^2 \Gamma_{t-1} \Delta} = \frac{\delta_{S_{t-1}}^{(t-1)}}{M_{t-1}^2 \Gamma_{t-1} \Delta} = \frac{W_t}{M_{t-1}\Gamma_{t-1}\Delta}.
\end{aligned}$$

Thus

$$\tilde{w}_{S_{t-1}}^{(t-1)}(\beta_{S_t-1}^{(t-1)}+\alpha/2)=\tfrac{W_t}{M_{t-1}\Gamma_{t-1}\Delta}\tfrac{\alpha}{4}(W_{t-1}\Gamma_{t-1}M_{t-1}+S_t)\geq\tfrac{W_t}{M_{t-1}\Gamma_{t-1}\Delta}\tfrac{\alpha}{4}(\delta_{S_{t-1}}^{(t-1)}\Gamma_{t-1})$$
$$=\tfrac{W_t}{M_{t-1}\Gamma_{t-1}\Delta}\tfrac{\alpha}{4}(M_{t-1}W_t\Gamma_{t-1})=\tfrac{\alpha}{4}\tfrac{W_t^2}{\Delta}=\tfrac{W_t}{M_t\Gamma_t\Delta}[\tfrac{\alpha}{4}M_t\Gamma_tW_t]=\tilde{w}_1^{(t)}\beta_1^{(t)}.$$

$$\tilde{w}_{S_{t-1}}^{(t-1)}\gamma_{S_{t-1}}^{(t-1)}=\tfrac{\delta_{S_{t-1}}^{(t-1)}}{M_{t-1}^2\Gamma_{t-1}\Delta}(\tfrac{4}{\alpha})\tfrac{M_{t-1}^2\Gamma_{t-1}}{\delta_{S_{t-1}}^{(t-1)}}=\tfrac{4}{\alpha\Delta}=\tfrac{\delta_1^{(t)}}{M_t^2\Gamma_t\Delta}\tfrac{M_t^2\Gamma_t}{\delta_1^{(t)}}=\tilde{w}_1^{(t)}\gamma_1^{(t)}.$$

$$\beta_1^{(t)}\gamma_{S_{t-1}}^{(t-1)}=(\tfrac{\alpha}{4}M_t\Gamma_tW_t)(\tfrac{4}{\alpha})\tfrac{M_{t-1}^2\Gamma_{t-1}}{M_{t-1}W_t}=M_tM_{t-1}\Gamma_t\Gamma_{t-1}$$
$$\overset{(a)}{\geq}\tfrac{M_t\Gamma_t}{M_{t-1}\Gamma_{t-1}}M_{t-1}^2=\tfrac{\tilde{w}_{S_{t-1}}^{(t-1)}}{\tilde{w}_1^{(t)}}M_{t-1}^2=\rho_1^{(t)}M_{t-1}^2,$$

where the inequality in (a) holds because we have $\Gamma_{t-1}\geq 1$ as a consequence of its definition. Thus all the requirements in Proposition 5.2 are satisfied. We get from (Equation 5.59) that

$$\textstyle\sum_{t=1}^N\omega_tQ(z^t;z)+\tfrac{\omega_N\eta_N}{2}\left\|x^N-x\right\|^2\leq\tfrac{\tilde{w}_1^{(1)}\beta_1^{(1)}+\omega_1\eta_1}{2}\left\|x^0-x\right\|^2+\tfrac{\tilde{w}_1^{(1)}\gamma_1^{(1)}}{2}\left\|\lambda_0^{(1)}-\lambda\right\|^2. \tag{5.66}$$

Similar arguments as that of Theorem Theorem 5.5 imply the boundedness of M_t, and that the ergodic average solution $\bar{z}^N$ defined according to (Equation 5.21) satisfies

$$Q(\bar{z}^N;(x^*;\lambda,\nu,\pi))\leq\tfrac{1}{\sum_{t=1}^N\omega_t}(\tfrac{2d(\Lambda_r)\left\|x^0-x^*\right\|}{\Delta}+L(\Lambda_r)\left\|x^0-x^*\right\|^2),$$
$$\forall\lambda\in\Lambda_r,(\nu,\pi)\in[V,\Pi]. \tag{5.67}$$

Since $\sum_{t=1}^N\omega_t\geq\max\{N(N+1)/2,\sqrt{2\kappa_r}[(1+1/\sqrt{2\kappa_r})^4-1]\}$ (see (Equation 5.37)), we get the optimality gap and feasibility violation convergence bound in (Equation 5.54). Moreover, since $\frac{\alpha}{2}\left\|\bar{x}^N-x^*\right\|^2\leq Q(\bar{z}^N;(x^*;\lambda^*,\nabla g(\bar{x}^N),\nabla f(\bar{x}^N)))$

(see (Equation 5.39)), the convergence of $\bar{x}^N$ to x^* in (Equation 5.54) also follows from (Equation 5.67).

Next we show Corollary 5.4. Fix a $r \geq 1$, and set $\Delta = \frac{[d(\Lambda_r)]^2}{L(\Lambda_r)\|x^0 - x^*\|^2 \alpha}$. For any

$$N \geq \min\{\sqrt{\tfrac{6L(\Lambda_r)\max\{c/r,1\}}{\epsilon}}\left\|x^0 - x^*\right\|^2, \\ [\sqrt{\tfrac{2L(\Lambda_r)}{\alpha}} + 1]\log(\tfrac{3\max\{c/r,1\}\sqrt{L(\Lambda_r)\alpha}\left\|x^0 - x^*\right\|^2}{\epsilon} + 1) + 4\},$$

we get

$$Q(\bar{z}^N; (x^*; \lambda, \nu, \pi)) \leq \min\{1, \tfrac{r}{c}\}\epsilon, \forall \lambda \in \Lambda_r, (\nu, \pi) \in [V, \Pi], \\ \Rightarrow \max\{F(\bar{x}^N) - F_*, r\left\|[g(\bar{x}^N)]_+\right\|\} \leq \min\{1, \tfrac{r}{c}\}\epsilon,$$

so that $\bar{x}^N$ is an $(\epsilon; \epsilon/c)$ solution. Therefore, the least number of phases N_ϵ required for such a solution admits the upper bound in the corollary statement.

Now we consider the corresponding number of matrix-vector multiplications C_ϵ in the N_ϵ phases, i.e., $C_\epsilon = \sum_{t=1}^{N_\epsilon} S_t$. We can deduce from the preceding argument that

$$\sum_{t=1}^{N_\epsilon} \omega_t \leq \frac{6\max\{1, \frac{c}{r}\}L(\Lambda_r)\left\|x^0 - x^*\right\|^2}{\epsilon}.$$

Calculating the sum of S_t directly is challenging, so we consider an easier quantity $R_t := [S_t - 3]_+$. An useful algebraic relation is

$$\textstyle\sum_{i=1}^{t-1} R_i \leq \bar{M} W_t. \tag{5.68}$$

The result can be deduced by induction. For $t = 2$, the relation clearly holds since $W_2\bar{M} \geq W_2 M_1 = S_1 \geq R_1$. Assuming (Equation 5.68) is valid up to $t \geq 2$, we have

$$W_{t+1} = \tfrac{W_t M_t + \frac{1}{\Gamma_t}(S_t - 1)}{M_t} \overset{(a)}{\geq} W_t + \tfrac{[S_t - 3]_+}{M_t} \overset{(b)}{\geq} \tfrac{1}{\bar{M}}\{\textstyle\sum_{i=1}^{t-1} R_i + [S_t - 3]_+\} = \tfrac{1}{\bar{M}}\textstyle\sum_{i=1}^{t-1}[R_i],$$

where (a) follows from the algebraic fact $\frac{S_t}{\Gamma_t} \geq S_t - 2$ (see Lemma 5.4) and that $S_t - 1 \geq 0$ and (b) follows from the induction hypothesis. Thus the principle of mathematical induction implies that (Equation 5.68) is valid. Consequently, for $R_\epsilon = \sum_{t=1}^{N_\epsilon} R_t$, we have

$$\begin{aligned}
R_\epsilon^2/2 &\leq \textstyle\sum_{s=1}^{S_t}[R_\epsilon]s = \sum_{t=1}^{N_\epsilon}\sum_{s=1}^{S_t}[R_t][(\sum_{j=1}^{t-1} R_j) + s] \leq \sum_{t=1}^{N_\epsilon}\sum_{s=1}^{S_t}[R_t][\bar{M}W_t + s] \\
&= \sum_{t=1}^{N_\epsilon}\textstyle\sum_{s=1}^{S_t}[S_t - 3][\bar{M}W_t + s] = \sum_{t=1}^{N_\epsilon}\sum_{t=2}^{S_t-2}[\bar{M}W_t + (s-1)] \\
&\leq \sum_{t=1}^{N_\epsilon}\textstyle\sum_{t=1}^{S_t-1}[\bar{M}W_t + (s-1)] \leq \bar{M}\sum_{t=1}^{N_\epsilon}\sum_{t=1}^{S_t-1}[W_t + \frac{(s-1)}{M_t}] \\
&\overset{(a)}{\leq} \bar{M}\sum_{t=1}^{N_\epsilon} M_t\omega_t\Delta \leq \sum_{t=1}^{N_\epsilon}\bar{M}^2\omega_t\Delta \leq \frac{6\max\{1,\frac{c}{r}\}L(\Lambda_r)\left\|x^0-x^*\right\|^2}{\epsilon}\frac{[d(\Lambda_r)]^2}{L(\Lambda_r)\|x^0-x^*\|^2\alpha}\bar{M}^2 \\
&\leq \frac{6\max\{1,\frac{c}{r}\}\bar{M}^2[d(\Lambda_r)]^2}{\epsilon\alpha},
\end{aligned}$$

where (a) follows from the fact that $\sum_{s=1}^{S_t}[S_t - 1]W_tM_t + (s-1) < \omega_t M_t^2\Delta$ (see (Equation 5.52)). Therefore we get $R_\epsilon \leq \sqrt{\frac{12}{\alpha\epsilon}}\bar{M}d(\Lambda_r)$. Since $\sum_{t=1}^{N_\epsilon} S_t \leq R_\epsilon + 3N_\epsilon$, the big-O bound on C_ϵ follows immediately.

The complexity bounds in Corollary 5.5 can be derived similarly.

□

5.5 Conclusion

To sum up, this paper proposes two efficient methods for large-scale function-constrained optimization. The simple ACGD method has the optimal oracle complexity, but it requires access to a QP solver. The more complicated ACGD-S method has both the optimal oracle complexity and the optimal computation complexity. Lower complexity bounds are provided to illustrate the oracle complexity of both ACGD and ACGD-S to be unimprovable for a general case of first-order methods. Together they provide a complete characterization of the difficulty of solving a smooth function-constrained optimization problem from both the oracle complexity and the computation complexity perspective.

5.6 Appendix

Lemma 5.2 *Let $\Lambda \subset \mathbb{R}^m_+$ and a convex vector-valued function $g : \mathbb{R}^n \to \mathbb{R}^m$ be given. If $\sum_{i=1}^m \lambda_i g_i$ is L-smooth for all $\lambda \in \Lambda$, i.e., $\|\sum_{i=1}^m \lambda_i \nabla g_i(x) - g_i(\bar{x})\| \leq L \|x - \bar{x}\| \; \forall x, \bar{x} \in R^n, \forall \lambda \in \Lambda$, the Bregman distance function generated by its (vector-valued) conjugate function U_g^* satisfies*

$$\langle \lambda, U_g^*(\nu, \bar{\nu}) \rangle \geq \|\sum_{i=1}^m \lambda_i(\nu_i - \bar{\nu}_i)\|^2 / (2L) \; \forall \nu, \bar{\nu} \in \{\nabla g(x) : x \in \mathbb{R}^n\}.$$

Proof: Let $\lambda \in \Lambda$, $\nu = \nabla g(x)$, and $\bar{\nu} = \nabla g(\bar{x})$ be given. Consider the function $\hat{g} := \sum_{i=1}^m \lambda_i g_i$. Clearly, $\hat{g}$ is L-smooth such that the Bregman distance function generated by its conjugate satisfies $U_{\hat{g}^*}(v, \bar{v}) \geq \|v - \bar{v}\|^2 / (2L)$. Since $\nabla \hat{g}(x) = \lambda^\top \nabla g(x)$, we have

$$\begin{aligned}
\lambda^\top U_{g^*}(\nu; \bar{\nu}) &\overset{(a)}{=} \textstyle\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \bar{\nu}_i) = \sum_{i=1}^m \lambda_i U_{g_i}(\bar{x}; x) \\
&= \textstyle\sum_{i=1}^m \lambda_i [g_i(\bar{x}) - g_i(x) - \langle \nabla g_i(x), \bar{x} - x \rangle] \\
&= \hat{g}(\bar{x}) - \hat{g}(x) - \langle \nabla \hat{g}(x), \bar{x} - x \rangle \\
&\overset{(b)}{=} U_{\hat{g}}(\bar{x}; x) = U_{\hat{g}^*}(\lambda^\top \nu; \lambda^\top \bar{\nu}) \geq \|\textstyle\sum_{i=1}^m \lambda_i(\nu_i - \bar{\nu}_i)\|^2 / (2L),
\end{aligned}$$

where (a) and (b) follows from the algebraic identity between Bregman distance functions generated by Fenchel conjugates (h, h^*), $U_h(y; \bar{y}) = U_{h^*}(\nabla h(\bar{y}); \nabla h(y))$. □ The next two lemmas provide some basic algebraic identifies useful for deriving complexity bounds.

Lemma 5.3 *Given an $x > 0$, the following algebraic relation is valid:*

$$h(y) := (1 + 1/x)^{y-3} \leq y \; \forall \; 2x \geq y \geq 1. \tag{5.69}$$

Proof: First, we show the relation for $y = 2x$,

i.e. $h(2x) = (1 + 1/x)^{2x-3} \leq 2x \; \forall x > 0$. Let's consider two cases. If $x \geq 4$, we have

$$(1 + 1/x)^{2x-3} \leq [(1 + 1/x)^x]^2 \leq \exp(2) \leq 8 \leq 2x.$$

If $0 < x < 4$, we have

$$[(1 + 1/x)^{x-1.5}]^2 \leq [1/(1 - \tfrac{x-1.5}{x})]^2 = (x/1.5)^2 = (x/2.25) \times x \leq 2x.$$

Thus $h(2x) \leq 2x$. Since $h(1) \leq 1$, the relation in (Equation 5.69) follows from the convexity of h with respect to y. □

Lemma 5.4 *Given non-negative parameters Δ_t, H, and $h \in \{0, 1\}$, suppose $S_t = \min\{S \in \mathbb{N}_+ : \sum_{s=1}^{S_t}[S]H + (s - h) \geq \Delta_t\}$, and Γ is the non-negative root of $\sum_{s=1}^{S_t}[S_t][\Gamma H + (s - h)] = \Gamma^2\Delta_t$, then S_t satisfies*

$$S_t/\Gamma \geq S_t - 2.$$

Proof: Suppose for the sake of contradiction that $S_t/\Gamma < S_t - 2$. On the one hand, the definition of S_t implies that

$$\Delta_t = H(\tfrac{S_t}{\Gamma}) + \textstyle\sum_{s=1}^{S_t} \frac{(s-h)}{\Gamma^2} = H(\tfrac{S_t}{\Gamma}) + \frac{1}{2}\frac{(S_t-h+1)}{\Gamma}\frac{S_t}{\Gamma} \geq H(\tfrac{S_t}{\Gamma}) + \tfrac{1}{2}(\tfrac{S_t}{\Gamma})^2.$$

On the other hand, the choice of S_t implies that $\sum_{s=1}^{S_t}[S_t - 1][H + (s - h)] < \Delta_t$, thus

$$\begin{aligned} &H(S_t - 1) + \tfrac{1}{2}(S_t - 1)(S_t - h - 1) < \Delta_t \\ \Rightarrow\; &H(\tfrac{S_t}{\Gamma}) + \tfrac{1}{2}(\tfrac{S_t}{\Gamma})^2 < \Delta_t. \end{aligned}$$

These two relations leads to the desired contradiction. □

REFERENCES

[1] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.

[2] A. S. Nemirovsky and D. B. Yudin, *Problem complexity and method efficiency in optimization.* John Wiley UK/USA, 1983.

[3] Y. E. Nesterov, "A method of solving a convex programming problem with convergence rate o\bigl(k^2\bigr)," in *Doklady Akademii Nauk*, Russian Academy of Sciences, vol. 269, 1983, pp. 543–547.

[4] Y. Nesterov, "Gradient methods for minimizing composite objective function. core discussion papers 2007076, université catholique de louvain," *Center for Operations Research and Econometrics (CORE)*, vol. 5, no. 5.3, 2007.

[5] P. Tseng, "On accelerated proximal gradient methods for convex-concave optimization, manuscript," *University of Washington, USA*, 2008.

[6] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[7] G. Lan, "Gradient sliding for composite optimization," *Mathematical Programming*, vol. 159(1-2), pp. 201–235, 2016.

[8] G. Lan and Y. Ouyang, "Accelerated gradient sliding for structured convex optimization," *Computational Optimization and Applications*, 2020, under revision.

[9] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2014.

[10] A. Beck, *First-order methods in optimization*. SIAM, 2017.

[11] L. Pardo, *Statistical inference based on divergence measures*. Chapman and Hall/CRC, 2018.

[12] G. C. Pflug and A. Pichler, "Approximations for probability distributions and stochastic optimization problems," in *Stochastic Optimization Methods in Finance and Energy*, Springer, 2011, pp. 343–387.

[13] C. Zhao and Y. Guan, "Data-driven risk-averse two-stage stochastic program with ζ-structure probability metrics," *Available on Optimization Online*, 2015.

[14] D. Bertsimas, V. Gupta, and N. Kallus, "Data-driven robust optimization," *Mathematical Programming*, vol. 167, no. 2, pp. 235–292, 2018.

[15] C. Zhao and Y. Guan, "Data-driven risk-averse stochastic optimization with Wasserstein metric," *Operations Research Letters*, vol. 46, no. 2, pp. 262–267, 2018.

[16] Y. Chen, H. Sun, and H. Xu, "Decomposition and discrete approximation methods for solving two-stage distributionally robust optimization problems," *Computational Optimization and Applications*, vol. 78, no. 1, pp. 205–238, 2021.

[17] A. Pichler and H. Xu, "Quantitative stability analysis for minimax distributionally robust risk optimization," *Mathematical Programming*, pp. 1–31, 2017.

[18] P. M. Esfahani and D. Kuhn, "Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations," *Mathematical Programming*, vol. 171, no. 1-2, pp. 115–166, 2018.

[19] R. Gao and A. J. Kleywegt, "Distributionally robust stochastic optimization with Wasserstein distance," *arXiv preprint arXiv:1604.02199*, 2016.

[20] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.

[21] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2009.

[22] G. Lan, "Complexity of stochastic dual dynamic programming," *Mathematical Programming*, pp. 1–38, 2020.

[23] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov, "New variants of bundle methods," *Mathematical Programming*, vol. 69, no. 1-3, pp. 111–147, 1995.

[24] A. Ben-Tal and A. S. Nemirovskii, *Lectures on modern convex optimization: analyse, algorithms, and engineering applications*. SIAM, 2001.

[25] Y. Liu, X. Yuan, S. Zeng, and J. Zhang, "Primal–dual hybrid gradient method for distributionally robust optimization problems," *Operations Research Letters*, vol. 45, no. 6, pp. 625–630, 2017.

[26] A. Chambolle and T. Pock, "On the ergodic convergence rates of a first-order primal–dual algorithm," *Mathematical Programming*, vol. 159, no. 1-2, pp. 253–287, 2016.

[27] G. Lan, "Bundle-level type methods uniformly optimal for smooth and nonsmooth convex optimization," *Mathematical Programming*, vol. 149, no. 1-2, pp. 1–45, 2015.

[28] G. Lan, *Lectures on Optimization Methods for Machine Learning*. Springer-Nature, 2020.

[29] A. Nemirovsky, "Information-based complexity of linear operator equations," *Journal of Complexity*, vol. 8, pp. 153–175, 2 1992.

[30] Y. Ouyang and Y. Xu, "Lower complexity bounds of first-order methods for convex-concave bilinear saddle-point problems," *Mathematical Programming*, pp. 1–35, 2019.

[31] Y. Nesterov, "Smooth minimization of non-smooth functions," *Mathematical programming*, vol. 103, no. 1, pp. 127–152, 2005.

[32] A. Ben-Tal and A. Nemirovski, "Non-euclidean restricted memory level method for large-scale convex optimization," *Mathematical Programming*, vol. 102, no. 3, pp. 407–456, 2005.

[33] Y. Nesterov, "A method for unconstrained convex minimization problem with the rate of convergence o (1/kˆ 2)," in *Doklady AN USSR*, vol. 269, 1983, pp. 543–547.

[34] S. Sen, R. D. Doverspike, and S. Cosares, "Network planning with random demand," *Telecommunication Systems*, vol. 3, no. 1, pp. 11–30, 1994.

[35] A. Shapiro and S. Ahmed, "On a class of minimax stochastic programs," *SIAM Journal on Optimization*, vol. 14, no. 4, pp. 1237–1249, 2004.

[36] D. Bertsimas and J. N. Tsitsiklis, *Introduction to linear optimization*. Athena Scientific Belmont, MA, 1997, vol. 6.

[37] J. Linderoth, A. Shapiro, and S. Wright, "The empirical behavior of sampling methods for stochastic programming," *Annals of Operations Research*, vol. 142, pp. 215–241, 2006.

[38] G. Lan, "Efficient methods for stochastic composite optimization," Georgia Institute of Technology, Manuscript, 2008.

[39] G. Lan, "An optimal method for stochastic composite optimization," *Mathematical Programming*, pp. 365–397, 2012.

[40] S. Ghadimi and G. Lan, "Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization i: A generic algorithmic framework," *SIAM Journal on Optimization*, vol. 22, no. 4, pp. 1469–1492, 2012.

[41] S. Ghadimi and G. Lan, "Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization, II: Shrinking procedures and optimal algorithms," *SIAM Journal on Optimization*, vol. 23, pp. 2061–2089, 2013.

[42] Y. Chen, G. Lan, and Y. Ouyang, "Optimal primal-dual methods for a class of saddle point problems," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 1779–1814, 2014.

[43] A. S. Lewis and S. J. Wright, "A proximal method for composite minimization," *Mathematical Programming*, vol. 158, no. 1-2, pp. 501–546, 2016.

[44] Y. M. Ermoliev, "A general stochastic programming problem. journal of cybernetics," *Journal of Cybernetics*, vol. 1, no. 4, pp. 106–112, 1971.

[45] Y. M. Ermoliev, *Methods of Stochastic Programming*. Nauka, Moscow, 1976.

[46] M. Wang, J. Liu, and E. X. Fang, "Accelerating stochastic composition optimization," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 3721–3743, 2017.

[47] A. Ruszczynski, "A stochastic subgradient method for nonsmooth nonconvex multilevel composition optimization," *SIAM Journal on Control and Optimization*, vol. 59, no. 3, pp. 2301–2320, 2021.

[48] M. Wang, E. X. Fang, and H. Liu, "Stochastic compositional gradient descent: Algorithms for minimizing compositions of expected-value functions," *Mathematical Programming*, vol. 161, no. 1-2, pp. 419–449, 2017.

[49] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[50] S. Ghadimi, A. Ruszczynski, and M. Wang, "A single timescale stochastic approximation method for nested stochastic optimization," *SIAM Journal on Optimization*, vol. 30, no. 1, pp. 960–979, 2020.

[51] K. Balasubramanian, S. Ghadimi, and A. Nguyen, "Stochastic multilevel composition optimization algorithms with level-independent convergence rates," *SIAM Journal on Optimization*, vol. 32, no. 2, pp. 519–544, 2022.

[52] J. Zhang and L. Xiao, "Multi-level composite stochastic optimization via nested variance reduction," *arXiv preprint arXiv:1908.11468*, 2019.

[53] S. Yang, M. Wang, and E. X. Fang, "Multilevel stochastic gradient methods for nested composition optimization," *SIAM Journal on Optimization*, vol. 29, no. 1, pp. 616–659, 2019.

[54] T. Chen, Y. Sun, and W. Yin, "Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization," *arXiv preprint arXiv:2008.10847*, 2020.

[55] Z. Zhang, S. Ahmed, and G. Lan, "Efficient algorithms for distributionally robust stochastic optimization with discrete scenario support," *arXiv preprint arXiv:1909.11216*, 2019.

[56] G. Lan and Y. Zhou, "An optimal randomized incremental gradient method," *Mathematical Programming*, vol. 171, pp. 167–215, 2018.

[57] Z. Allen-Zhu, "How to make the gradients small stochastically: Even faster convex and nonconvex sgd," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[58] R. T. Rockafellar, *Convex analysis*. Princeton university press, 1970.

[59] G. Lan and Z. Zhang, "Optimal methods for risk averse distributed optimization," *arXiv preprint arXiv:2203.05117*, 2022.

[60] Z. Zhang and G. Lan, "Solving convex smooth function constrained optimization is as almost easy as unconstrained optimization," *arXiv preprint arXiv:2210.05807*, 2022.

[61] G. Lan and Y. Zhou, "Random gradient extrapolation for distributed and stochastic optimization," *SIAM Journal on Optimization*, vol. 28, no. 4, pp. 2753–2782, 2018.

[62] A. Froehlich, *Definition: Star network*, https : / / www . techtarget . com / searchnetworking/definition/star-network, Retrieved: July. 2022.

[63] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: nonlinear phenomena*, vol. 60, no. 1-4, pp. 259–268, 1992.

[64] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM review*, vol. 51, no. 3, pp. 455–500, 2009.

[65] R. Tomioka, T. Suzuki, K. Hayashi, and H. Kashima, "Statistical performance of convex tensor decomposition," *Advances in neural information processing systems*, vol. 24, 2011.

[66] J. Mairal, R. Jenatton, G. Obozinski, and F. Bach, "Convex and network flow optimization for structured sparsity.," *Journal of Machine Learning Research*, vol. 12, no. 9, 2011.

[67] R. Tibshirani, M. Saunders, S. Rosset, J. Zhu, and K. Knight, "Sparsity and smoothness via the fused lasso," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 67, no. 1, pp. 91–108, 2005.

[68] L. Jacob, G. Obozinski, and J.-P. Vert, "Group lasso with overlap and graph lasso," in *Proceedings of the 26th annual international conference on machine learning*, 2009, pp. 433–440.

[69] P. Kairouz *et al.*, "Advances and open problems in federated learning," *Foundations and Trends® in Machine Learning*, vol. 14, no. 1–2, pp. 1–210, 2021.

[70] H. Markowitz, "Portfolio selection," *The Journal of Finance*, vol. 7, no. 1, pp. 77–91, 1952.

[71] G. Martínez and L. Anderson, "A risk-averse optimization model for unit commitment problems," in *2015 48th Hawaii International Conference on System Sciences*, IEEE, 2015, pp. 2577–2585.

[72] P. Javanbakht and S. Mohagheghi, "A risk-averse security-constrained optimal power flow for a power grid subject to hurricanes," *Electric Power Systems Research*, vol. 116, pp. 408–418, 2014.

[73] E. G. Larsson, O. Edfors, F. Tufvesson, and T. L. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE communications magazine*, vol. 52, no. 2, pp. 186–195, 2014.

[74] R. S. Tol, "The economic effects of climate change," *Journal of economic perspectives*, vol. 23, no. 2, pp. 29–51, 2009.

[75] S. Parkvall, E. Dahlman, A. Furuskar, and M. Frenne, "Nr: The new 5G radio access technology," *IEEE Communications Standards Magazine*, vol. 1, no. 4, pp. 24–30, 2017.

[76] D. Kuhn, P. M. Esfahani, V. A. Nguyen, and S. Shafieezadeh-Abadeh, "Wasserstein distributionally robust optimization: Theory and applications in machine learning," in *Operations research & management science in the age of analytics*, Informs, 2019, pp. 130–166.

[77] D. Bertsimas and A. Thiele, "Robust and data-driven optimization: Modern decision making under uncertainty," in *Models, methods, and applications for innovative decision making*, INFORMS, 2006, pp. 95–122.

[78] Z. Wang, P. W. Glynn, and Y. Ye, "Likelihood robust optimization for data-driven problems," *Computational Management Science*, vol. 13, no. 2, pp. 241–261, 2016.

[79] WeBank, *WeBank and Swiss Re signed cooperation MoU*, http:https://www.fedai.org/news/webank-and-swiss-re-signed-cooperation-mou/, Retrieved: July. 2022.

[80] E. CORDIS, *Machine learning ledger orchestration for drug discovery*, https://featurecloud.eu/about/our-vision/, Retrieved: July. 2022.

[81] FeatureCloud, *Featurecloud: Our vision, 2022*, https://cordis.europa.eu/project/id/831472, Retrieved: July. 2022.

[82] musketeer, *Musketeer: About, 2022*, https://musketeer.eu/project/, Retrieved: July. 2022.

[83] Z. Zhang and G. Lan, "Optimal algorithms for convex nested stochastic composite optimization," *arXiv preprint arXiv:2011.10076*, 2020.

[84] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *Journal of mathematical imaging and vision*, vol. 40, no. 1, pp. 120–145, 2011.

[85] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié, "Optimal convergence rates for convex distributed optimization in networks," *Journal of Machine Learning Research*, vol. 20, no. 159, pp. 1–31, 2019.

[86] Y. Ouyang and Y. Xu, "Lower complexity bounds of first-order methods for convex-concave bilinear saddle-point problems," *Mathematical Programming*, vol. 185, no. 1, pp. 1–35, 2021.

[87] G. Lan, "Gradient sliding for composite optimization," *Mathematical Programming*, vol. 159, no. 1, pp. 201–235, 2016.

[88] G. Lan, Y. Ouyang, and Y. Zhou, "Graph topology invariant gradient and sampling complexity for decentralized and stochastic optimization," *arXiv preprint arXiv:2101.00143*, 2021.

[89] K. Scaman, F. Bach, S. Bubeck, Y. T. Lee, and L. Massoulié, "Optimal algorithms for smooth and strongly convex distributed optimization in networks," in *international conference on machine learning*, PMLR, 2017, pp. 3027–3036.

[90] J.-B. Hiriart-Urruty and C. Lemaréchal, *Convex analysis and minimization algorithms II*. Springer science & business media, 1993, vol. 306.

[91] C. Lemaréchal, A. Nemirovskii, and Y. Nesterov, "New variants of bundle methods," *Mathematical programming*, vol. 69, no. 1, pp. 111–147, 1995.

[92] Y. Chen, H. Sun, and H. Xu, "Decomposition and discrete approximation methods for solving two-stage distributionally robust optimization problems," *Computational Optimization and Applications*, vol. 78, no. 1, pp. 205–238, 2021.

[93] A. Nemirovski, "Lectures on modern convex optimization," in *Society for Industrial and Applied Mathematics (SIAM*, Citeseer, 2001.

[94] P. Rigollet and X. Tong, "Neyman-pearson classification, convexity and stochastic constraints," *Journal of Machine Learning Research*, 2011.

[95] M. B. Zafar, I. Valera, M. G. Rogriguez, and K. P. Gummadi, "Fairness constraints: Mechanisms for fair classification," in *Artificial intelligence and statistics*, PMLR, 2017, pp. 962–970.

[96] R. Gandy, "Portfolio optimization with risk constraints," Ph.D. dissertation, Universität Ulm, 2005.

[97] A. S. Nemirovsky, "On optimality of krylov's information when solving linear operator equations," *Journal of Complexity*, vol. 7, no. 2, pp. 121–130, 1991.

[98] D. Boob, Q. Deng, and G. Lan, "Stochastic first-order methods for convex and nonconvex functional constrained optimization," *Mathematical Programming*, pp. 1–65, 2022.

[99] E. Y. Hamedani and N. S. Aybat, "A primal-dual algorithm with line search for general convex-concave saddle point problems," *SIAM Journal on Optimization*, vol. 31, no. 2, pp. 1299–1329, 2021.

[100] Y. Xu, "First-order methods for problems with o (1) functional constraints can have almost the same convergence rate as for unconstrained problems," *arXiv preprint arXiv:2010.02282*, 2020.

[101] Q. Lin, S. Nadarajah, and N. Soheili, "A level-set method for convex optimization with a feasible solution path," *SIAM Journal on Optimization*, vol. 28, no. 4, pp. 3290–3311, 2018.

[102] Y. Nesterov, *Lectures on convex optimization*. Springer, 2018, vol. 137.

[103] D. Bertsekas, *Convex optimization theory*. Athena Scientific, 2009, vol. 1.

[104] G. Lan and R. D. Monteiro, "Iteration-complexity of first-order penalty methods for convex programming," *Mathematical Programming*, vol. 138, no. 1, pp. 115–139, 2013.

[105] S. Yang, X. Li, and G. Lan, "Data-driven minimax optimization with expectation constraints," *arXiv preprint arXiv:2202.07868*, 2022.