Stochastic Augmented Lagrangian Framework with Second-Order Convergence Guarantees for Nonconvex Expectation-Constrained Optimization

Raghu Bollapragada* Yash Kumar*

September 23, 2026

Abstract

In this paper, we propose and analyze an augmented Lagrangian framework for solving stochastic nonconvex optimization problems with expectation-based equality constraints over a closed and convex constraint set. The framework generates a sequence of nonconvex primal subproblems, which are solved inexactly using stochastic second-order methods. We establish iteration complexity results for obtaining approximate second-order stationary points, both in expectation and with prescribed probability, under corresponding conditions on the accuracy of the primal subproblem solutions. We further establish sample complexity results for the framework with general stochastic second-order subproblem solvers under reasonable assumptions on their theoretical guarantees. Moreover, we incorporate three existing stochastic second-order solvers and derive the corresponding deterministic and probabilistic sample complexity results for obtaining approximate second-order stationary points. To the best of our knowledge, such sample complexity guarantees for obtaining approximate second-order stationary points in nonconvex expectation-constrained optimization have not been established previously. Finally, we demonstrate the empirical performance of the proposed framework on two nonconvex machine learning problems.

1 Introduction

In this paper, we consider expectation-constrained stochastic optimization problems of the form

$$\begin{aligned} \min_{x\in\mathcal{X}} \quad & f(x) := \mathbb{E}_\xi[F(x,\xi)] \\ \text{s.t.} \quad & c(x) := \mathbb{E}_\zeta[C(x,\zeta)] = 0, \end{aligned} \tag{1.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ and $c : \mathbb{R}^d \to \mathbb{R}^m$ are smooth, twice continuously differentiable and possibly nonconvex; ξ and ζ are random variables with associated probability spaces $(\mathcal{S}_\xi, \mathcal{G}_\xi, \mathcal{P}_\xi)$ and $(\mathcal{S}_\zeta, \mathcal{G}_\zeta, \mathcal{P}_\zeta)$ respectively; The stochastic functions $F : \mathbb{R}^d \times \mathcal{S}_\xi \to \mathbb{R}$ and $C : \mathbb{R}^d \times \mathcal{S}_\zeta \to \mathbb{R}^m$ are the stochastic objective and constraint functions; and $\mathbb{E}_\xi[\cdot]$ and $\mathbb{E}_\zeta[\cdot]$ denote expectations with respect to $\mathcal{P}_\xi$ and $\mathcal{P}_\zeta$, respectively. We assume that the constraint set $\mathcal{X} \subseteq \mathbb{R}^d$ is closed and convex and that only stochastic estimates of the objective function, constraint function, and their derivatives are available. Although (1.1) contains only equality constraints, the formulation can accommodate inequality constraints. In particular, inequality constraints can be reformulated as equality constraints by introducing slack variables and incorporating their nonnegativity constraints into the set $\mathcal{X}$. Problems of the form (1.1) arise in a range of applications, including fairness-constrained machine learning [21, 50], physics-informed neural networks [8, 31], robotics [33, 34], and energy systems [20, 63].

*Operations Research and Industrial Engineering Graduate Program, The University of Texas at Austin. Emails: {raghu.bollapragada, yashkumar1803}@utexas.edu

Deterministic methods that use exact evaluations of functions and their derivatives to solve (1.1) have been extensively studied; see, e.g., [49]. Nevertheless, nonconvex constrained optimization remains challenging because the problem landscape may contain multiple local minimizers/maximizers, and saddle points. Unlike in convex optimization, an approximate first-order stationary point of a nonconvex problem need not correspond to an approximate local minimizer. Therefore, algorithms require second-order (curvature) information to distinguish between different stationary points.

These challenges are further exacerbated in the stochastic setting, where only stochastic estimates of the objective, constraint functions, and their derivatives are available, and errors in these estimators can adversely affect the optimization process. A few stochastic frameworks have been developed for expectation-constrained problems, including proximal-point methods [14, 44], subgradient-switching methods [32, 30], sequential quadratic programming methods [60, 23], and augmented Lagrangian or penalty-based methods [1, 39, 42, 64, 71]. However, in nonconvex settings, these methods primarily establish convergence to only approximate first-order stationary points.

We propose and analyze stochastic algorithms for obtaining approximate second-order stationary points of (1.1). We develop an inexact stochastic augmented Lagrangian framework that solves a sequence of nonconvex subproblems over the convex set $\mathcal{X}$, with penalty parameters and Lagrange multipliers updated between subproblem solves. Each subproblem is solved inexactly to satisfy either expectation-based or probability-based second-order inexactness conditions. We establish theoretical guarantees and evaluate the empirical performance of the framework using three stochastic second-order subproblem solvers. The proposed framework requires only stochastic estimates of the objective and constraint functions, their gradients, and Hessian vector products, thereby enabling the use of curvature information without explicitly forming or storing Hessian matrices.

1.1 Notation and Preliminary Definitions

We first introduce notation and preliminary definitions used throughout the paper. We denote the set of positive integers by $\mathbb{N} := \{1, 2, 3, \dots, \}$, the set of non-negative integers by $\mathbb{N}_0 := \{0, 1, 2, \dots, \}$, the set of real numbers by $\mathbb{R}$, and the set of non-negative real numbers by $\mathbb{R}_+$. The sets of d-dimensional real vectors and $d \times m$ real matrices are denoted by $\mathbb{R}^d$ and $\mathbb{R}^{d\times m}$, respectively. For a vector a, $a^{(j)}$ denotes its jth component. For each realization of ζ, $C(\cdot, \zeta) : \mathbb{R}^d \to \mathbb{R}^m$, with $C^{(j)}(\cdot, \zeta)$ denoting its jth component, we use $\nabla C^{(j)}(x, \zeta) \in \mathbb{R}^d$ to denote the gradient of $C^{(j)}(\cdot, \zeta)$ at x, and $\nabla^2 C^{(j)}(x, \zeta) \in \mathbb{R}^{d\times d}$ to denote the Hessian of $C^{(j)}(\cdot, \zeta)$ at x. Likewise, $c^{(j)}$ denotes the jth component of c, $\nabla c^{(j)}(x) \in \mathbb{R}^d$ denotes the gradient of $c^{(j)}(\cdot)$ at x, and $\nabla^2 c^{(j)}(x) \in \mathbb{R}^{d\times d}$ denotes its Hessian, for each $j \in \{1, \dots, m\}$. For a matrix A, $\text{Null}(A) := \{x \mid Ax = 0\}$ denotes its null space. For a real symmetric matrix A, $\sigma_{\min}(A)$ denotes its smallest eigenvalue. Unless indicated otherwise in the subscripts, $\|\cdot\|$ denotes the ℓ_2 vector norm for vectors and the vector-induced ℓ_2 norm for matrices. We use $\text{proj}_{\mathcal{X}}(x)$ and $\text{dist}(x, \mathcal{X})$ to denote the projection of x onto $\mathcal{X}$ and the distance from x to $\mathcal{X}$, respectively. That is,

$$\text{proj}_{\mathcal{X}}(x) := \arg\min_{x'\in\mathcal{X}} \|x - x'\|, \quad \text{dist}(x, \mathcal{X}) := \min_{x'\in\mathcal{X}} \|x - x'\|. \tag{1.2}$$

Furthermore, $T_{\mathcal{X}}(x)$ and $N_{\mathcal{X}}(x)$ denote the tangent and normal cones to $\mathcal{X}$ at x, respectively. The critical tangent cone is defined by

$$T^{(c)}_{\mathcal{X}}(x) := T_{\mathcal{X}}(x) \cap \text{Null}(\nabla c(x)). \tag{1.3}$$

For an event E, $\mathbb{P}(E)$ denotes its probability and $\mathbf{1}_A$ denotes its indicator variable. In particular, $\mathbf{1}_A = 1$ when E occurs and $\mathbf{1}_A = 0$ otherwise. The notation $Var_\zeta(C(x, \zeta))$ represents the trace of the covariance matrix of stochastic vector function $C(x, \zeta)$ with respect to random variable ζ. Finally, we use the standard asymptotic notation $\mathcal{O}(\cdot)$, $\Omega(\cdot)$, and $\Theta(\cdot)$. In particular, $h(n) \in \mathcal{O}(g(n))$ means that $h(n)$ is bounded above by a positive constant multiple of $g(n)$ for all sufficiently large n, while $h(n) \in \Omega(g(n))$ gives the corresponding lower bound. The notation $h(n) \in \Theta(g(n))$ means that both bounds hold, implying that both functions share same asymptotic growth rates. Additionaly, we write $h(n) \in \tilde{\mathcal{O}}(g(n))$ if $h(n) \in \mathcal{O}(g(n) \log^r g(n))$ for some constant $r \geq 0$. That is, $\tilde{\mathcal{O}}$ suppresses polylogarithmic factors.

With this notation, we say that a point $x \in \mathcal{X}$ is a second-order stationary point of (1.1) if there exists $\lambda \in \mathbb{R}^m$ such that

$$\|\mathrm{proj}_{T_{\mathcal{X}}(x)}(-\nabla f(x) - \nabla c(x)^\top \lambda)\| = 0, \qquad \|c(x)\| = 0, \text{ and}$$
$$\min_{u \in T_{\mathcal{X}}^{(c)}(x), \|u\|=1} u^\top \big(\nabla^2 f(x) + \textstyle\sum_{j=1}^m \lambda^{(j)} \nabla^2 c^{(j)}(x)\big) u \geq 0.$$

More generally, for $\epsilon^{(g)} > 0$ and $\epsilon^{(H)} > 0$, we say that $x \in \mathcal{X}$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate second-order stationary point if there exists $\lambda \in \mathbb{R}^m$ such that

$$\begin{aligned}
&\text{(i)} && \|\mathrm{proj}_{T_{\mathcal{X}}(x)}(-\nabla f(x) - \nabla c(x)^\top \lambda)\| \leq \epsilon^{(g)}, && (1^{\text{st}}\text{-order stationarity condition})\\
&\text{(ii)} && \|c(x)\| \leq \epsilon^{(g)}, && (\text{feasibility condition})\\
&\text{(iii)} && \min_{u \in T_{\mathcal{X}}^{(c)}(x), \|u\|=1} u^\top \big(\nabla^2 f(x) + \textstyle\sum_{j=1}^m \lambda^{(j)} \nabla^2 c^{(j)}(x)\big) u \geq -\epsilon^{(H)}, && (2^{\text{nd}}\text{-order stationarity condition})
\end{aligned} \tag{1.4}$$

for some $\lambda \in \mathbb{R}^m$, $\epsilon^{(g)} > 0$, and $\epsilon^{(H)} > 0$; see e.g., [46, 56, 58].

Remark 1.1. *The first-order stationarity condition in* (1.4)(i) *is equivalent to to the distance-based condition*

$$\mathrm{dist}(\nabla f(x) + \nabla c(x)^\top \lambda, -N_{\mathcal{X}}(x)) \leq \epsilon^{(g)},$$

which is the form commonly used in literature; see, e.g., [1, 39, 58]. However, we use the projection-based form in (1.4)(i) *because it simplified our analysis. To our knowledge, theoretical analyses based on first-order stationarity conditions in this projection form are limited. This condition makes explicit use of projections onto the tangent cone $T_{\mathcal{X}}(x)$. For a detailed discussion of the properties of the projection operator $\mathrm{proj}_{T_{\mathcal{X}}}(\cdot)$, see Remark A.1.*

In the stochastic settings, we may not be able to obtain approximate solutions that satisfy the deterministic stationarity conditions in (1.4). Instead, we consider random iterates that satisfy these conditions either in expectation or with a prescribed probability. To this end, we define two types of approximate solutions as follows:

Definition 1.1. *A (possibly random) iterate $\tilde{x} \in \mathcal{X}$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate expected second-order stationary point of* (1.1)*, if there exists $\tilde{\lambda} \in \mathbb{R}^m$ such that,*

$$\begin{aligned}
\mathbb{E}[\|\mathrm{proj}_{T_{\mathcal{X}}(\tilde{x})}(-\nabla f(\tilde{x}) - \nabla c(\tilde{x})^\top \tilde{\lambda})\|] &\leq \epsilon^{(g)},\\
\mathbb{E}[\|c(\tilde{x})\|] &\leq \epsilon^{(g)},\\
\mathbb{E}\left[\min_{u \in T_{\mathcal{X}}^{(c)}(\tilde{x}), \|u\|=1} u^\top (\nabla^2 f(\tilde{x}) + \textstyle\sum_{j=1}^m \tilde{\lambda}^{(j)} \nabla^2 c^{(j)}(\tilde{x})) u\right] &\geq -\epsilon^{(H)}.
\end{aligned} \tag{1.5}$$

Definition 1.2. *A (possibly random) iterate $\tilde{x} \in \mathcal{X}$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point of* (1.1)*, if there exists $\tilde{\lambda} \in \mathbb{R}^m$ such that,*

$$\mathbb{P}\left(\begin{array}{c}
\{\|\mathrm{proj}_{T_{\mathcal{X}}(\tilde{x})}(-\nabla f(\tilde{x}) - \nabla c(\tilde{x})^\top \tilde{\lambda})\| \leq \epsilon^{(g)}\} \cap\\
\{\|c(\tilde{x})\| \leq \epsilon^{(g)}\} \cap\\
\{\min_{u \in T_{\mathcal{X}}^{(c)}(\tilde{x}), \|u\|=1} u^\top (\nabla^2 f(\tilde{x}) + \textstyle\sum_{j=1}^m \tilde{\lambda}^{(j)} \nabla^2 c^{(j)}(\tilde{x})) u \geq -\epsilon^{(H)}\}
\end{array}\right) \geq p. \tag{1.6}$$

We utilize algorithms based on stochastic function, gradient and Hessian vector product evaluations to obtain these approximate solutions. To quantify the computational effort required by these algorithms, we define the sample complexity of an algorithm as follows:

Definition 1.3. *The sample complexity of an algorithm, denoted by $\mathcal{W}$, is the total number of stochastic objective gradient evaluations $\nabla F(x,\xi)$, stochastic objective Hessian vector product evaluations $\nabla^2 F(x,\xi)v$, stochastic constraint function evaluations $C^{(j)}(x,\zeta)$, stochastic constraint gradient evaluations $\nabla C^{(j)}(x,\zeta)$, and stochastic constraint Hessian vector product evaluations $\nabla^2 C^{(j)}(x,\zeta)v$, for $j \in 1,\ldots,m$ and an arbitrary vector v, required to obtain either an $(\epsilon^{(g)},\epsilon^{(H)})$-accurate second-order stationary point of (1.1).*

1.2 Contributions

The primary contributions of this paper are three-fold.

1. We propose and analyze a novel algorithmic framework, named MESCAL (Method for Expectation-constraints with Second-order Convergence via Augmented Lagrangian), based on the inexact augmented Lagrangian framework for solving nonconvex expectation-constrained optimization problems of the form (1.1). The framework involves solving a sequence of nonconvex subproblems over the convex constraint set $\mathcal{X}$. We develop inexactness conditions in both expectation and probability for solving these subproblems inexactly, which can be satisfied by general second-order stochastic solvers. We establish iteration complexity bounds for MESCAL to obtain an $(\epsilon^{(g)},\epsilon^{(H)})$-accurate expected or p-probabilistic second-order stationary point within $\mathcal{O}\left(\log\left((\epsilon^{(g)})^{-1}\right)\right)$ iterations (see Theorem 2.1).

2. We establish sample complexity results for MESCAL when general second-order solvers are used to solve the subproblems inexactly under reasonable assumptions on their theoretical properties, motivated by the lower bounds established in [6], both in expectation and in probability (see Theorem 3.8). Furthermore, when $\mathcal{X} = \mathbb{R}^d$, we use three second-order solvers, SG-HV-NC [6, Algorithm 4], Natasha2 [3], and SPIDER [24], within MESCAL to establish deterministic and probabilistic sample complexity bounds for obtaining an $(\epsilon^{(g)},\epsilon^{(H)})$-accurate second-order stationary point in probability. Table 1 contextualizes our results against prior work. To the best of our knowledge, such second-order stationarity guarantees for expectation-constrained problems of the form (1.1) have not previously appeared in the literature.

3. We demonstrate the empirical performance of the proposed algorithmic framework on two relevant machine learning tasks: (i) a fairness-constrained optimization problem and (ii) a nonconvex Neyman–Pearson classification problem. We compare the performance of the three second-order solvers when used within the proposed MESCAL framework on these problems.

Table 1: Summary of sample complexity $\mathcal{W}$ results in the relevant literature under different problem settings. In all the works mentioned here, the objective considered is nonconvex, stochastic and the constraint set $\mathcal{X} = \mathbb{R}^d$. There are two types of constraints referred to here: 1) $\mathbb{E}$ refers to expectation constraints, 2) None means no constraints. There are two types of iterates referred to here: 1) $\mathbb{E}$ refers to random iterates satisfying stationarity conditions in (1.5), 2) $\mathbb{P}$ refers to random iterates satisfying stationarity conditions in (1.6). Here, first-order sample complexity refers to work bounds in achieving first-order stationarity conditions and feasibility conditions, and second-order sample complexity refers to work bounds in achieving second-order stationarity conditions and feasibility conditions, if constraints present. The sample complexity bounds denoted in our results are deterministic. We further prove sample complexity bounds in probability that improve upon these results by a factor of $\mathcal{O}((\epsilon^{(g)})^{-1})$; see Section 4. (Note: *For marked paper, $\epsilon^{(g)} = \epsilon, \epsilon^{(H)} = \sqrt{L_2\epsilon}$, where L_2 is Lipschitz constant corresponding to Hessian of the objective function.)

Reference	**Constraints**	**Iterate Type**	**1st-Order Sample Complexity**	**2nd-Order Sample Complexity**
ConEx [14]	$\mathbb{E}$	$\mathbb{E}$	$\mathcal{O}\left((\epsilon^{(g)})^{-6}\right)$	-
Stoch-iALM [39]	$\mathbb{E}$	$\mathbb{E}$	$\mathcal{O}\left((\epsilon^{(g)})^{-5}\right)$	-
Stoch-SQP [60]	$\mathbb{E}$	$\mathbb{E}$	$\mathcal{O}\left((\epsilon^{(g)})^{-4}\right)$	-
SCR [61]	None	$\mathbb{P}$	-	$\mathcal{O}\left(\epsilon^{-3.5}\right)^*$
Natasha2 [3]	None	$\mathbb{P}$	-	$\tilde{\mathcal{O}}\big((\epsilon^{(g)})^{-3}(\epsilon^{(H)})^{-1} + (\epsilon^{(H)})^{-5}\big)$
SPIDER [24]	None	$\mathbb{P}$	-	$\tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-3} + (\epsilon^{(H)})^{-5}\right)$
SG-HV-NC [6]	None	$\mathbb{P}$	-	$\tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-3} + (\epsilon^{(H)})^{-5}\right)$
MESCAL-SG-HV-NC (Theorem 4.2) **MESCAL-SPIDER (Theorem 4.6)**	$\mathbb{E}$	$\mathbb{P}$	-	$\boldsymbol{\tilde{\mathcal{O}}\big((\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}\big)}$
MESCAL-Natasha2 (Theorem 4.4)	$\mathbb{E}$	$\mathbb{P}$	-	$\boldsymbol{\tilde{\mathcal{O}}\big((\epsilon^{(g)})^{-7}(\epsilon^{(H)})^{-1} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}\big)}$

1.3 Literature Review

Deterministic augmented Lagrangian (AL) methods date back to the independent works of Hestenes [28] and Powell [51] in 1969. The framework was subsequently developed and analyzed in, among others, the works of Rockafellar [54] in 1973 and Bertsekas [10] in 1975. Our framework for expectation-constrained optimization is based in part on a stochastic version of the deterministic framework established by Bertsekas in [10]. Since then, the framework has been employed extensively in the optimization literature, and several methods have established complexity results under deterministic settings [36, 41, 67]. The literature most closely related to our work can be broadly divided into two categories: methods for expectation-constrained optimization and second-order methods for unconstrained nonconvex optimization.

Expectation-Constrained Optimization. The development of algorithms for expectation-constrained optimization remains an active area of research [14, 23, 30, 32, 35, 44, 45, 69, 70], with several works using

AL and penalty-based methods [1, 39, 42, 64, 71]. For convex expectation-constrained problems, existing methods include primal subgradient switching [32], primal subgradient descent with dual ascent [68], and linearized proximal methods of multipliers [71], with sample complexity bounds of $\mathcal{O}((\epsilon^{(g)})^{-2})$ under their respective assumptions.

Existing work for nonconvex expectation-constrained optimization is more extensive but focuses primarily on first-order methods. Boob et al. [14] employ a proximal-point method to transform the nonconvex problem into a sequence of convex subproblems. Under stronger assumptions, including strong feasibility, they establish an $\mathcal{O}((\epsilon^{(g)})^{-6})$ sample complexity bound. Ma et al. [44] obtain the same bound using a related proximal-point approach based on the online stochastic subgradient routine of Yu et al. [70]. Huang et al. [30] also establish an $\mathcal{O}((\epsilon^{(g)})^{-6})$ bound using a switching-subgradient method related to that of Lan et al. [32]. Beyond AL and penalty-based approaches, stochastic sequential quadratic programming methods have recently been considered for nonconvex expectation-constrained problems [23, 60]; in particular, Shen et al. [60] establish an $\mathcal{O}((\epsilon^{(g)})^{-4})$ sample complexity bound under linear independence constraint qualification (LICQ).

Within AL and penalty-based first-order methods, Li et al. [39] provided an early result for nonconvex expectation-constrained optimization, establishing an $\mathcal{O}((\epsilon^{(g)})^{-5})$ sample complexity bound. Their method employs a proximal variant of the STORM estimator [19] to solve the primal subproblems. Subsequently, Alacaoglu and Wright [1] developed a single-loop penalty-based method using the STORM estimator and established the same sample complexity bound. Some of the conditions underlying our framework are motivated by [1, 39]. For weakly convex problems, Liu et al. [42] developed a single-loop method based on an exact penalty formulation. Xiao et al. [64] developed an AL-based framework which embeds generalized subgradient methods for a single-step primal variable update.

Unconstrained Second-order Methods. Second-order methods are typically used to obtain approximate second-order stationary points. To the best of our knowledge, no existing work establishes second-order sample complexity guarantees for nonconvex expectation-constrained optimization. However, second-order methods have been extensively studied for unconstrained nonconvex optimization. In the deterministic setting, Cartis et al. [16] established a lower bound of $\mathcal{O}(\max((\epsilon^{(g)})^{-2}, (\epsilon^{(H)})^{-3}))$ evaluations for finding an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate second-order stationary point under reasonable assumptions, which is attained by cubic regularization methods [15]. Newton-type methods [66] require $\mathcal{O}(\max((\epsilon^{(g)})^{-2}(\epsilon^{(H)})^{-1}, (\epsilon^{(H)})^{-3}))$ evaluations to obtain an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate second-order stationary point.

Second-order methods have also been developed for stochastic optimization problems [3, 6, 24, 61], with their sample complexity bounds summarized in Table 1. These are typically stochastic extensions of existing deterministic second-order methods. In [61], a stochastic cubic regularized Newton method is proposed with a sample complexity bound of $\mathcal{O}(\epsilon^{-3.5})$ for obtaining an $(\epsilon, \sqrt{L\epsilon})$-accurate point, where L is a Lipschitz constant. The negative curvature-based approaches in [3, 6, 24] achieve a similar sample complexity bound of $\tilde{\mathcal{O}}(\epsilon^{-3.5})$ under the same accuracy requirements. However, when the first-order and second-order tolerance parameters are of the same order, i.e., $\mathcal{O}(\epsilon_g) = \mathcal{O}(\epsilon_H) = \mathcal{O}(\epsilon)$, the sample complexity bound for the stochastic cubic regularized Newton method becomes $\mathcal{O}(\epsilon^{-7})$, whereas the negative curvature-based approaches achieve a bound of $\tilde{\mathcal{O}}(\epsilon^{-5})$.

In this work, we consider [3, 6, 24] as subproblem solvers within the MESCAL framework due to their favorable sample complexity bounds. For ease of comparison, Table 1 suppresses the dependence of these bounds on Lipschitz constants, variance bounds, and the initial optimality gap. However, these quantities play an important role in the sample complexity analysis of MESCAL; see the theoretical results in Section 3 and Section 4.

1.4 Paper Organization

The paper is organized as follows. In Section 2, we present the MESCAL framework and establish iteration complexity guarantees for obtaining an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate expected or p-probabilistic second-order stationary point. In Section 3, we establish sample complexity results for MESCAL with a general solver to

solve the subproblems arising within MESCAL. In Section 4, under the restriction $\mathcal{X} = \mathbb{R}^d$, we adapt three existing stochastic second-order solvers for use within MESCAL and establish their corresponding sample complexity results. In Section 5, we present numerical results on two nonconvex optimization problems using MESCAL with these solvers. Finally, we provide concluding remarks in Section 6.

2 Algorithmic Framework

In this section, we propose an algorithmic framework for solving (1.1). The proposed framework, named MESCAL (Method for Expectation-constraints with Second-order Convergence via Augmented Lagrangian), is based on augmented Lagrangian methods. These methods rely on the property of *augmentability*, which states that if a point $x^* \in \mathcal{X}$ is a local solution of (1.1), then there exist some $\lambda^* \in \mathbb{R}^m$ and a scalar $\alpha > 0$ such that x^* is a local minimizer of the augmented Lagrangian problem $\min_{x\in\mathcal{X}} \mathcal{L}_\alpha(x, \lambda^*) := f(x) + (\lambda^*)^\top c(x) + \frac{\alpha}{2} c(x)^2$ [29]. A typical augmented Lagrangian method iteratively updates estimates of the primal-dual solution (x^*, λ^*) through the following two steps:

Step 1: Solve the primal subproblem by finding

$$x_k \in \arg\min_{x\in\mathcal{X}} \mathcal{L}_{\alpha_k}(x, \lambda_k) := f(x) + \lambda_k^\top c(x) + \tfrac{\alpha_k}{2}\|c(x)\|^2, \tag{2.1}$$

where $\alpha_k > 0$ is the penalty parameter and λ_k is the dual variable at iteration k. The function $\mathcal{L}_{\alpha_k}$ is referred to as the augmented Lagrangian function.

Step 2: Update the dual variable according to

$$\lambda_{k+1} = \lambda_k + \alpha_k c(x_k).$$

However, implementing these steps in the nonconvex stochastic optimization setting considered in this paper poses significant challenges. Due to nonconvexity, finding an exact global solution to the primal subproblem is generally intractable. Moreover, even finding an exact local minimizer of the primal subproblem is challenging, as iterative nonlinear optimization algorithms typically require access to function and derivative evaluations of the objective function f and constraint function c, both of which involve expectations of stochastic functions. While inexact augmented Lagrangian methods, which solve the primal subproblems only approximately up to a near local minimizer, are well established in the literature [36, 67], they typically impose deterministic inexactness conditions. Such conditions are generally impractical in stochastic settings. To overcome this limitation, we propose solving the primal subproblems inexactly while imposing inexactness conditions in expectation or in probability, rather than deterministically.

Furthermore, the dual update step requires evaluating the exact constraint function $c(x_k)$ and may result in unbounded dual variables in nonconvex settings. We address this challenge by employing subsampled approximations of $c(x_k)$ and modifying the dual update using an upper-bounding sequence $\{\gamma_k\}$. For notational simplicity, unless otherwise specified, ∇ and ∇^2 applied to the augmented Lagrangian $\mathcal{L}_\alpha(x, \lambda)$, and its estimators in forthcoming sections, denote differentiation with respect to the primal variable x. In particular, $\nabla\mathcal{L}_\alpha(x, \lambda) := \nabla_x\mathcal{L}_\alpha(x, \lambda)$ and $\nabla^2\mathcal{L}_\alpha(x, \lambda) := \nabla^2_{xx}\mathcal{L}_\alpha(x, \lambda)$. We also introduce notation for expectations and probabilities that will be used throughout the paper. Unless otherwise specified, let

$$\mathbb{E}[\cdot] := \mathbb{E}[\cdot \mid x_{-1}, \lambda_0], \qquad \mathbb{P}(\cdot) := \mathbb{P}(\cdot \mid x_{-1}, \lambda_0), \tag{2.2}$$

where the conditioning is with respect to the initial primal dual iterates and all randomness generated by the algorithm thereafter. At the beginning of iteration k, let $\mathcal{F}_k$ denote the σ-algebra generated by all the randomness available up to the point at which the primal dual iterates (x_{k-1}, λ_k) are obtained. We then use

$$\mathbb{E}[\cdot \mid \mathcal{F}_k] \qquad \text{and} \qquad \mathbb{P}(\cdot \mid \mathcal{F}_k), \tag{2.3}$$

to denote the conditional expectation and probability given the history of the algorithm up to the beginning of iteration k.

The step-by-step overview of the MESCAL framework is presented in Algorithm 1.

Algorithm 1: Method for Expectation-constraints with Second-order Convergence via Augmented Lagrangian (MESCAL)

Input: Initial iterate $x_{-1} \in \mathcal{X}$; initial dual variable $\lambda_0 \in \mathbb{R}^m$; initial penalty parameter $\alpha_0 > 0$; penalty parameter increase factor $\beta > 1$; primal subproblem solver $\mathcal{M}$ satisfying either Option I (expectation conditions) or Option II (probabilistic conditions); primal subproblem error tolerance sequences $\epsilon_k^{(g)}, \epsilon_k^{(H)} > 0$; success probability sequences $\{p_k\}$ for Option II, constraint sample size sequence $\{|S_k^{(d)}|\}$; and dual upper bound sequence $\{\gamma_k\}$

1 **for** $k = 0, 1, \ldots$ **do**

2 Update penalty parameter: $\alpha_k = \alpha_0 \, \beta^k$

3 Starting from x_{k-1}, obtain primal iterate x_k by applying solver $\mathcal{M}$ to solve subproblem (2.1) satisfying either Option I or Option II below:

4 **Option I (expectation conditions):**

$$\begin{aligned} &\mathbb{E}[\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k, \lambda_k))\| \mid \mathcal{F}_k] \leq \epsilon_k^{(g)}, \\ &\mathbb{E}\left[\min_{u \in T_{\mathcal{X}}(x_k), \|u\|=1} u^\top \nabla^2 \mathcal{L}_{\alpha_k}(x_k, \lambda_k) u \mid \mathcal{F}_k\right] \geq -\epsilon_k^{(H)}. \end{aligned} \tag{2.4}$$

5 **Option II (probabilistic conditions):**

$$\mathbb{P}\left(\begin{array}{c} \left\{\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k, \lambda_k))\| \leq \epsilon_k^{(g)}\right\} \cap \\ \left\{ \min_{u \in T_{\mathcal{X}}(x_k), \|u\|=1} u^\top \left(\nabla^2 \mathcal{L}_{\alpha_k}(x_k, \lambda_k)\right) u \geq -\epsilon_k^{(H)} \right\} \end{array} \middle| \mathcal{F}_k \right) \geq p_k. \tag{2.5}$$

6 Draw $|S_k^{(d)}|$ i.i.d. samples $\{\zeta_i\}_{i=1}^{S_k^{(d)}}$ and compute stochastic constraint estimate

$$\overline{C}_{S_k^{(d)}}(x_k) = \tfrac{1}{|S_k^{(d)}|} \textstyle\sum_{i=1}^{|S_k^{(d)}|} C(x_k, \zeta_i)$$

7 Update dual variable:

$$\lambda_{k+1} = \begin{cases} \lambda_k & \text{if } \|\overline{C}_{S_k^{(d)}}(x_k)\| = 0 \\ \lambda_k + \min\left\{\alpha_k, \frac{\gamma_k}{\left\|\overline{C}_{S_k^{(d)}}(x_k)\right\|}\right\} \overline{C}_{S_k^{(d)}}(x_k) & \text{otherwise} \end{cases} \tag{2.6}$$

Remark 2.1. *We make a few remarks about the MESCAL framework as presented in Algorithm 1:*

- *The framework follows the structure of a typical augmented Lagrangian method: the primal subproblem is solved inexactly in Line 3, followed by the dual variable update in Line 7. Although augmented Lagrangian methods traditionally offer stronger practical and theoretical advantages over quadratic penalty methods by incorporating dual updates to improve feasibility and conditioning of the subproblem, these benefits are not fully reflected in existing theoretical analysis of nonconvex problems. In particular, nonasymptotic guarantees for augmented Lagrangian variants with a fixed penalty parameter and dual step size [65] exhibit worse iteration complexity than those relying on increasing penalty parameters and decaying dual step sizes [58], as surveyed by Alacaoglu and Wright [1]. Accordingly, we increase the penalty parameter α_k at each iteration in Line 2. Consequently, we modify the dual update to ensure that the sequence of dual variables remains uniformly bounded by using an upper-bounding sequence γ_k.*

- *We impose second-order inexactness conditions for the primal subproblem at each iteration, either in expectation as in (2.4) or in probability as in (2.5). These conditions ensure that each primal subproblem is solved to sufficient accuracy for the overall framework to produce an approximate second-order stationary point of (1.1). They also enable us to establish iteration complexity results for obtaining such*

an iterate; see Theorem 2.1. We note that these inexactness conditions can be satisfied by a (stochastic) second-order solver $\mathcal{M}$. For example, under $\mathcal{X} = \mathbb{R}^d$, the method in [9] uses a condition analogous to (2.4) to define second-order stationarity in expectation. Similarly, the methods in [3, 6, 9, 24] use conditions analogous to (2.5) to define second-order stationarity in probability.

- *In Section 4, we discuss three well-known second-order solvers with established theoretical complexity guarantees for solving primal subproblems of the form (2.1) under $\mathcal{X} = \mathbb{R}^d$ while satisfying the inexactness conditions given in Algorithm 1. Furthermore, for each of these solvers, we establish theoretical sample complexity results for obtaining approximate second-order stationary points. We note that we were unable to identify second-order solvers with established sample complexity guarantees for the case $\mathcal{X} \subset \mathbb{R}^d$. Consequently, we restrict our sample complexity results to the case $\mathcal{X} = \mathbb{R}^d$.*

Next, we state the assumptions on the objective and constraint functions required to establish the iteration complexity results for the proposed framework.

Assumption 2.1. *The functions $f : \mathcal{X} \to \mathbb{R}$, and $c : \mathcal{X} \to \mathbb{R}^m$ are twice continuously differentiable, where $\mathcal{X} \subseteq \mathbb{R}^d$ is a closed and convex set. Moreover, there exist positive constants κ_f, $\kappa_{\nabla f}$, κ_c, $\kappa_{\nabla c}$ such that*

$$|f(x)| \leq \kappa_f, \quad \|\nabla f(x)\| \leq \kappa_{\nabla f}, \quad \|c(x)\| \leq \kappa_c, \quad \|\nabla c(x)\| \leq \kappa_{\nabla c}, \quad \|\nabla^2 c(x)\| \leq \kappa_{\nabla^2 c}, \quad \forall x \in \mathcal{X}.$$

Assumption 2.2. *There exists a positive constant C_{reg} such that*

$$C_{reg}\|c(x)\| \leq \|\text{proj}_{T_{\mathcal{X}}(x)}(-\nabla c(x)^\top c(x))\|, \quad \forall x \in \mathcal{X}.$$

Remark 2.2. *The boundedness conditions imposed in Assumption 2.1 are standard assumptions in nonconvex constrained optimization; see, for example, [1, 38, 41, 58]. Assumption 2.2 is a regularity condition on the constraints, and similar assumptions are commonly employed in the literature; see [1, 38, 41, 43, 58, 64]. In the case $\mathcal{X} = \mathbb{R}^d$, Assumption 2.2 implies that $\|\nabla c(x)^T c(x)\| \geq C_{reg}\|c(x)\|$, which ensures that $\nabla c(x)^\top c(x)$ cannot be arbitrarily small relative to $c(x)$. Such regularity conditions, including Slater's condition, LICQ, and MFCQ (Mangasarian-Fromovitz constraint qualification), are useful in establishing the convergence of iterative optimization algorithms to local solutions in nonconvex optimization settings. We emphasize that the constants appearing in these assumptions need not be known to establish our theoretical convergence guarantees or to implement the algorithm in practice; it is only required that such constants exist. We also note that an equivalent condition to Assumption 2.2, which is commonly used in the literature, is*

$$C_{reg}\|c(x)\| \leq \text{dist}(-\nabla c(x)^\top c(x), N_{\mathcal{X}}(x)) \quad \forall x \in \mathcal{X}.$$

However, we use the projection-based condition in Assumption 2.2 because it is more convenient for our analysis.

We now present the main theoretical result of this section on the iteration complexity of Algorithm 1.

Theorem 2.1 (Iteration complexity)**.** *Suppose Assumptions 2.1 and 2.2 hold. Let $\{x_k\}$ be the sequence of iterates generated by Algorithm 1 with initial iterate $x_{-1} \in \mathcal{X}$ and initial dual iterate $\lambda_0 \in \mathbb{R}^m$. Suppose that the dual bound parameters $\gamma_k > 0$ be a bounded sequence satisfying $\sum_{k=0}^{\infty} \gamma_k = C_\gamma < \infty$, and that the dual sample sizes satisfy $|S_k^{(d)}| \geq 1$ for all $k \in \mathbb{N}_0$. If the error bounds are chosen such that $\epsilon_k^{(g)} \leq \epsilon^{(g)} < 1$ and $\epsilon_k^{(H)} \leq \epsilon^{(H)} < 1$ for any $k \in \mathbb{N}_0$, then, for any $k = K \geq K^*$, where $\Lambda := \|\lambda_0\| + C_\gamma$ and*

$$K^* := \left\lceil \log_\beta \left(\frac{1+\kappa_{\nabla f}+\kappa_{\nabla c}\Lambda}{C_{reg}\alpha_0 \epsilon^{(g)}} \right) \right\rceil = \mathcal{O}(\log(\tfrac{1}{\epsilon^{(g)}})), \tag{2.7}$$

x_K satisfies the following:

*(i) If **Option I** is chosen in Algorithm 1, then x_K is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate expected second-order stationary point satisfying (1.5), with $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$.*

(ii) If ***Option II*** *is chosen in Algorithm 1 with $p_k = p$ for all $k \in \mathbb{N}_0$, then x_K is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point satisfying (1.6), with $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$.*

Proof. We first show that the sequence $\{\lambda_k\}$ is uniformly bounded. From (2.6), for any $k \in \mathbb{N}_0$, if $\left\|\overline{C}_{S_k^{(d)}}(x_k)\right\| = 0$, then $\|\lambda_{k+1}\| = \|\lambda_k\|$. If $\left\|\overline{C}_{S_k^{(d)}}(x_k)\right\| \neq 0$, then

$$\|\lambda_{k+1}\| \leq \|\lambda_k\| + \left\| \min\left\{\alpha_k, \tfrac{\gamma_k}{\|\overline{C}_{S_k^{(d)}}(x_k)\|}\right\} \overline{C}_{S_k^{(d)}}(x_k) \right\| \leq \|\lambda_k\| + \gamma_k.$$

Therefore, $\|\lambda_{k+1}\| \leq \|\lambda_k\| + \gamma_k$ for all $k \in \mathbb{N}_0$. Summing this inequality from $k = 0$ to $k = t-1$ for any $t \geq 1$ yields

$$\|\lambda_t\| \leq \|\lambda_0\| + \sum_{k=0}^{t-1} \gamma_k \leq \Lambda < \infty. \tag{2.8}$$

Therefore, we have $\|\lambda_k\| \leq \Lambda$ for any $k \in \mathbb{N}_0$. Using this bound, the feasibility error at any $k \in \mathbb{N}_0$ can be bounded as

$$\begin{aligned}
\|c(x_k)\| &\leq \tfrac{1}{C_{\mathrm{reg}}} \|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla c(x_k)^\top c(x_k))\| \\
&= \tfrac{1}{C_{\mathrm{reg}}} \left\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}\left(\tfrac{-\nabla \mathcal{L}_{\alpha_k}(x_k,\lambda_k) + \nabla f(x_k) + \nabla c(x_k)^\top \lambda_k}{\alpha_k}\right)\right\| \\
&= \tfrac{1}{C_{\mathrm{reg}}\alpha_k} \|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}\left(-\nabla \mathcal{L}_{\alpha_k}(x_k,\lambda_k) + \nabla f(x_k) + \nabla c(x_k)^\top \lambda_k\right)\| \\
&\leq \tfrac{1}{C_{\mathrm{reg}}\alpha_k} \left(\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k,\lambda_k))\| + \|\nabla f(x_k) + \nabla c(x_k)^\top \lambda_k\|\right) \\
&\leq \tfrac{1}{C_{\mathrm{reg}}\alpha_k} \left(\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k,\lambda_k))\| + \|\nabla f(x_k)\| + \|\nabla c(x_k)\|\|\lambda_k\|\right) \\
&\leq \tfrac{\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k,\lambda_k))\| + \kappa_{\nabla f} + \kappa_{\nabla c}\Lambda}{C_{\mathrm{reg}}\alpha_k},
\end{aligned} \tag{2.9}$$

where the first inequality follows from Assumption 2.2, the second equality follows from the positive homogeneity property of the projection onto the tangent cone, i.e., $\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(av) = a\,\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(v)$ for any $a > 0$ and $v \in \mathbb{R}^d$, the third inequality follows from Lemma A.1, and the last inequality follows from Assumption 2.1 and $\|\lambda_k\| \leq \Lambda$.

Let $\tilde{\lambda}_k := \lambda_k + \alpha_k c(x_k)$ for all $k \in \mathbb{N}_0$. Then, we have

$$\begin{aligned}
\|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla f(x_k) - \nabla c(x_k)^\top \tilde{\lambda}_k)\| &= \|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla f(x_k) - \nabla c(x_k)^\top (\lambda_k + \alpha_k c(x_k)))\| \\
&= \|\mathrm{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k,\lambda_k))\|.
\end{aligned} \tag{2.10}$$

For the second-order condition, consider

$$\begin{aligned}
&\min_{u \in T_{\mathcal{X}}^{(c)}(x_k),\, \|u\|=1} u^\top \left(\nabla^2 f(x_k) + \textstyle\sum_{j=1}^m \tilde{\lambda}_k^{(j)} \nabla^2 c^{(j)}(x_k)\right) u \\
&= \min_{u \in T_{\mathcal{X}}^{(c)}(x_k),\, \|u\|=1} u^\top \left(\nabla^2 f(x_k) + \textstyle\sum_{j=1}^m (\lambda_k^{(j)} + \alpha_k c^{(j)}(x_k)) \nabla^2 c^{(j)}(x_k)\right) u + \alpha_k u^\top \nabla c(x_k)^\top \nabla c(x_k) u \\
&= \min_{u \in T_{\mathcal{X}}^{(c)}(x_k),\, \|u\|=1} u^\top (\nabla^2 \mathcal{L}_{\alpha_k}(x_k,\lambda_k)) u \\
&\geq \min_{u \in T_{\mathcal{X}}(x_k),\, \|u\|=1} u^\top (\nabla^2 \mathcal{L}_{\alpha_k}(x_k,\lambda_k)) u,
\end{aligned} \tag{2.11}$$

where the first equality follows because $u \in T_{\mathcal{X}}^{(c)}(x_k)$ implies $u \in \mathrm{Null}(\nabla c(x_k))$, and hence $\nabla c(x_k) u = 0$, and the inequality is due to the fact that $T_{\mathcal{X}}^{(c)}(x_k) \subseteq T_{\mathcal{X}}(x_k)$.

Case (i): If **Option I** is chosen in Algorithm 1, then, taking expectations in (2.10),and using the tower property of conditional expectations and the first inequality in (2.4), we have, for all $k \in \mathbb{N}_0$,

$$\mathbb{E}[\|\text{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla f(x_k) - \nabla c(x_k)^\top \tilde{\lambda}_k)\|] = \mathbb{E}[\mathbb{E}[\|\text{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k, \lambda_k))\| \mid \mathcal{F}_k]] \leq \epsilon_k^{(g)}. \tag{2.12}$$

Using (2.12), taking expectations in (2.9), and choosing $k = K \geq K^*$, we obtain

$$\mathbb{E}[\|c(x_K)\|] \leq \frac{\epsilon_K^{(g)} + \kappa_{\nabla f} + \kappa_{\nabla c}\Lambda}{C_{\text{reg}}\alpha_K} \leq \frac{1 + \kappa_{\nabla f} + \kappa_{\nabla c}\Lambda}{C_{\text{reg}}\alpha_K} \leq \epsilon^{(g)}, \tag{2.13}$$

where the last inequality is due to (2.7). Next, taking expectations in (2.11), and using the tower property of conditional expectations and the second inequality in (2.4), we have, for all $k \in \mathbb{N}_0$,

$$\begin{aligned} &\mathbb{E}\left[\min_{u \in T_{\mathcal{X}}^{(c)}(x_k),\, \|u\|=1} u^\top \left(\nabla^2 f(x_k) + \textstyle\sum_{j=1}^m \tilde{\lambda}_k^{(j)} \nabla^2 c^{(j)}(x_k)\right) u\right] \\ &\quad \geq \mathbb{E}\left[\mathbb{E}\left[\min_{u \in T_{\mathcal{X}}(x_k),\, \|u\|=1} u^\top (\nabla^2 \mathcal{L}_{\alpha_k}(x_k, \lambda_k)) u \mid \mathcal{F}_k\right]\right] \geq -\epsilon_k^{(H)}. \end{aligned} \tag{2.14}$$

Therefore, using $\epsilon_k^{(g)} \leq \epsilon^{(g)}$, $\epsilon_k^{(H)} \leq \epsilon^{(H)}$, (2.13), and setting $k = K$ in (2.12) and (2.14), we conclude that $(x_K, \tilde{\lambda}_K)$ satisfies (1.5).

Case (ii): If **Option II** is chosen in Algorithm 1, then, define the following events:

$$\begin{aligned} P_{1,k} &:= \{\|\text{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla f(x_k) - \nabla c(x_k)^\top \tilde{\lambda}_k)\| \leq \epsilon_k^{(g)}\}, \\ P_{2,k} &:= \left\{\min_{u \in T_{\mathcal{X}}^{(c)}(x_k),\, \|u\|=1} u^\top \left(\nabla^2 f(x_k) + \textstyle\sum_{j=1}^m \tilde{\lambda}_k^{(j)} \nabla^2 c^{(j)}(x_k)\right) u \geq -\epsilon_k^{(H)}\right\}, \\ P_{3,k} &:= \left\{\|c(x_k)\| \leq \epsilon_k^{(g)}\right\}, \\ E_{1,k} &:= \{\|\text{proj}_{T_{\mathcal{X}}(x_k)}(-\nabla \mathcal{L}_{\alpha_k}(x_k, \lambda_k))\| \leq \epsilon_k^{(g)}\}, \\ E_{2,k} &:= \left\{\min_{u \in T_{\mathcal{X}}(x_k),\, \|u\|=1} u^\top (\nabla^2 \mathcal{L}_{\alpha_k}(x_k, \lambda_k)) u \geq -\epsilon_k^{(H)}\right\}. \end{aligned}$$

From (2.10), we have $P_{1,k} = E_{1,k}$, while (2.11) implies $E_{2,k} \subseteq P_{2,k}$. Moreover, by (2.9), we have $E_{1,K} \subseteq P_{3,K}$ for any $K \geq K^*$. Therefore,

$$\mathbb{P}\left(P_{1,K} \cap P_{2,K} \cap P_{3,K}\right) = \mathbb{P}\left(E_{1,K} \cap P_{2,K} \cap P_{3,K}\right) \geq \mathbb{P}\left(E_{1,K} \cap E_{2,K}\right).$$

Consider the indicator variable $\mathbf{1}_{E_{1,K} \cap E_{2,K}}$ for event $E_{1,K} \cap E_{2,K}$. By the tower property of conditional expectations and (2.5),

$$\mathbb{E}[\mathbf{1}_{E_{1,K} \cap E_{2,K}}] = \mathbb{E}[\mathbb{E}[\mathbf{1}_{E_{1,K} \cap E_{2,K}} \mid \mathcal{F}_K]] \geq p_K.$$

Therefore, using $p_K = p$, $\epsilon_k^{(g)} \leq \epsilon^{(g)}$, and $\epsilon_k^{(H)} \leq \epsilon^{(H)}$, we conclude that x_K satisfies (1.6), with $\tilde{\lambda} := \tilde{\lambda}_K$.

□

Remark 2.3. *We make the following remarks about Theorem 2.1.*

- *The dual upper-bounding sequence $\{\gamma_k\}$ is introduced to ensure that the dual variables remain uniformly bounded. With this bound, our analysis follows a structure similar to that of penalty-based methods. In particular, due to the nonconvexity of the problem, our analysis does not exploit the potential benefits of the dual updates in establishing the iteration complexity result. In practice, however, dual updates can still be beneficial. Accordingly, in our numerical experiments in Section 5, we set $\gamma_k = \infty$ for all $k \in \mathbb{N}_0$, thereby removing the explicit upper bound on the dual updates. We observe that this choice provides good empirical performance while the resulting dual variables remain bounded in our experiments.*

- *The iteration complexity result in Theorem 2.1 holds for any dual sample size $|S_k^{(d)}| \geq 1$, and the effect of this sample size is not reflected in the complexity bound. This is a consequence of the dual upper-bounding sequence $\{\gamma_k\}$, which allows the analysis to accommodate arbitrary dual sample sizes. In practice, however, sufficiently large sample sizes can provide more accurate constraint estimates and, consequently, more accurate dual updates. Inspired by [39] and the fact that we can maintain near-accurate dual updates without compromising sample complexity bounds, we choose $S_k^{(d)} = \Theta((\epsilon^{(g)})^{-2})$, which, under the standard variance assumption, gives $\mathbb{E}[\|\overline{C}_{S_k^{(d)}}(x_k) - c(x_k)\|^2] = \mathcal{O}(1/|S_k^{(d)}|) = \mathcal{O}((\epsilon^{(g)})^2)$. Thus, the sampled constraint function provides a sufficiently accurate approximation of $c(x_k)$ in expectation. This choice of dual sample size does not alter the overall sample complexity results established in Section 3 or Section 4.*

- *Although Theorem 2.1 requires $\mathcal{O}(\log(1/\epsilon^{(g)}))$ outer iterations, the same approximate second-order stationarity guarantees can be obtained using a single outer iteration by choosing the initial penalty parameter as $\alpha_0 = \mathcal{O}(\frac{1}{\epsilon^{(g)}})$. Nevertheless, using single iteration solves can yield worse sample complexity bounds in expectation and in probability, similar to the deterministic sample complexity bounds observed in Section 4 (see Lemma 3.6 and Remark 3.3). Moreover, the use of multiple outer iterations can provide both theoretical and practical benefits in convex settings [12, 13]. Finally, although the theorem is stated with $p_k = p$ for all $k \in \mathbb{N}_0$, this sequence may instead be chosen to decrease gradually, provided that $p_K \leq p$ for $K \geq K^*$; the same conclusions then follow [12, 13].*

3 Sample Complexity

In this section, we establish sample complexity, as defined in Definition 1.3, required to obtain an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate expected or p-probabilistic second-order stationary point of (1.1) for the proposed MESCAL framework. We consider a general solver $\mathcal{M}$ for solving the primal subproblem inexactly at each iteration $k \in \mathbb{N}_0$ in Algorithm 1 to achieve iterates x_k satisfying (2.4) or (2.5) under reasonable assumptions on the theoretical properties of the solver $\mathcal{M}$ (see Assumption 3.4). In Section 4, we discuss three specific existing stochastic second-order solvers that can be employed when $\mathcal{X} = \mathbb{R}^d$ and **Option II** is chosen in Algorithm 1, as we are not aware of existing solvers with theoretical guarantees for the more general settings considered here.

We first introduce additional notation to denote the stochastic estimates of the gradient and Hessian of the augmented Lagrangian function used by the primal solver $\mathcal{M}$, as their exact counterparts may be unavailable or expensive to compute. To simplify notation throughout the analysis, we define a joint random vector $\theta = (\xi, \zeta^{(1)}, \zeta^{(2)})$, where $\xi \sim \mathcal{P}_\xi$ and $\zeta^{(1)}, \zeta^{(2)} \overset{\text{i.i.d.}}{\sim} \mathcal{P}_\zeta$. We define the probability space on which θ is defined as $(\mathcal{S}_\theta, \mathcal{G}_\theta, \mathcal{P}_\theta) := (\mathcal{S}_\xi \times \mathcal{S}_\zeta \times \mathcal{S}_\zeta, \mathcal{G}_\xi \otimes \mathcal{G}_\zeta \otimes \mathcal{G}_\zeta, \mathcal{P}_\xi \otimes \mathcal{P}_\zeta \otimes \mathcal{P}_\zeta)$. Next, we define the following stochastic gradient and Hessian estimators of the augmented Lagrangian function:

$$\nabla \bar{\mathcal{L}}_\alpha(x, \lambda; \theta) := \nabla F(x, \xi) + \nabla C(x, \zeta^{(1)})^\top (\lambda + \alpha C(x, \zeta^{(2)})), \tag{3.1}$$

$$\nabla^2 \bar{\mathcal{L}}_\alpha(x, \lambda; \theta) := \nabla^2 F(x, \xi) + \sum_{j=1}^{m} (\lambda^{(j)} + \alpha C^{(j)}(x, \zeta^{(2)})) \nabla^2 C^{(j)}(x, \zeta^{(1)}) + \alpha \nabla C(x, \zeta^{(1)})^\top \nabla C(x, \zeta^{(2)}) \tag{3.2}$$

By the independence of $\zeta^{(1)}$ and $\zeta^{(2)}$, the estimators in (3.1) and (3.2) are unbiased. That is, $\mathbb{E}[\nabla \bar{\mathcal{L}}_\alpha(x, \lambda; \theta)] = \nabla \mathcal{L}_\alpha(x, \lambda)$ and $\mathbb{E}[\nabla^2 \bar{\mathcal{L}}_\alpha(x, \lambda; \theta)] = \nabla^2 \mathcal{L}_\alpha(x, \lambda)$.

Next, we state the assumptions used to establish sample complexity for the MESCAL framework.

Assumption 3.1. *The stochastic objective function gradient $\nabla F(x, \xi)$ is $L_{\nabla f}$-Lipschitz continuous $\mathcal{P}_\xi$-almost surely on $\mathcal{X}$, and each stochastic constraint gradient $\nabla C^{(j)}(x, \zeta)$ is $L_{\nabla c}$-Lipschitz continuous $\mathcal{P}_\zeta$-almost surely on $\mathcal{X}$. That is,*

$$\|\nabla F(x, \xi) - \nabla F(y, \xi)\| \leq L_{\nabla f} \|x - y\|, \quad \forall x, y \in \mathcal{X}, \quad \mathcal{P}_\xi\text{-a.s.}, \quad \text{and}$$
$$\|\nabla C^{(j)}(x, \zeta) - \nabla C^{(j)}(y, \zeta)\| \leq L_{\nabla c} \|x - y\|, \quad \forall j \in \{1, 2, \cdots, m\}, \quad \forall x, y \in \mathcal{X}, \quad \mathcal{P}_\zeta\text{-a.s.}$$

Assumption 3.2. *The stochastic objective function Hessian $\nabla^2 F(x,\xi)$ is $L_{\nabla^2 f}$-Lipschitz continuous $\mathcal{P}_\xi$-almost surely on $\mathcal{X}$, and each stochastic constraint Hessian $\nabla^2 C^{(j)}(x,\zeta)$ is $L_{\nabla^2 c}$-Lipschitz continuous $\mathcal{P}_\zeta$-almost surely on $\mathcal{X}$. That is,*

$$\|\nabla^2 F(x,\xi) - \nabla^2 F(y,\xi)\| \leq L_{\nabla^2 f}\|x-y\|, \quad \forall x,y \in \mathcal{X}, \quad \mathcal{P}_\xi\text{-a.s.}, \quad \text{and}$$
$$\|\nabla^2 C^{(j)}(x,\zeta) - \nabla^2 C^{(j)}(y,\zeta)\| \leq L_{\nabla^2 c}\|x-y\|, \quad \forall j \in \{1,2,\cdots,m\}, \quad \forall x,y \in \mathcal{X}, \quad \mathcal{P}_\zeta\text{-a.s.}$$

Assumption 3.3. *The stochastic objective function gradient and Hessian satisfy uniform error bounds $\mathcal{P}_\xi$-almost surely. That is, there exist constants $\sigma_{\nabla f}, \sigma_{\nabla^2 f} \in [0,\infty)$ such that*

$$\|\nabla F(x,\xi) - \nabla f(x)\| \leq \sigma_{\nabla f}, \quad \text{and} \quad \|\nabla^2 F(x,\xi) - \nabla^2 f(x)\| \leq \sigma_{\nabla^2 f} \quad \forall x \in \mathcal{X}, \quad \mathcal{P}_\xi\text{-a.s.}$$

In addition, the stochastic constraint function, constraint Jacobian and constraint Hessian satisfy uniform error bounds $\mathcal{P}_\zeta$-almost surely. That is, there exist constants $\sigma_c, \sigma_{\nabla c}, \sigma_{\nabla^2 c} \in [0,\infty)$ such that

$$\|C(x,\zeta) - c(x)\| \leq \sigma_c, \|\nabla C(x,\zeta) - \nabla c(x)\| \leq \sigma_{\nabla c}, \text{ and } \|\nabla^2 C(x,\zeta) - \nabla^2 c(x)\| \leq \sigma_{\nabla^2 c} \quad \forall x \in \mathcal{X}, \mathcal{P}_\zeta\text{-a.s.}$$

The following lemma provides useful results as a consequence of these assumptions.

Lemma 3.1. *Suppose Assumptions 2.1 and 3.3 hold. Then, the stochastic objective function gradient and stochastic constraint function, gradient, and Hessian are bounded. That is,*

$$\|\nabla F(x,\xi)\| \leq \kappa_{\nabla f} + \sigma_{\nabla f}, \quad \forall x \in \mathcal{X}, \quad \mathcal{P}_\xi\text{-a.s.},$$
$$\|C(x,\zeta)\| \leq \kappa_c + \sigma_c, \|\nabla C(x,\zeta)\| \leq \kappa_{\nabla c} + \sigma_{\nabla c}, \text{ and } \|\nabla^2 C(x,\zeta)\| \leq \kappa_{\nabla^2 c} + \sigma_{\nabla^2 c}, \quad \forall x \in \mathcal{X}, \quad \mathcal{P}_\zeta\text{-a.s.}$$

Furthermore, each stochastic constraint function $C^{(j)}$ is Lipschitz continuous $\mathcal{P}_\zeta$-almost surely on $\mathcal{X}$. That is,

$$|C^{(j)}(x,\zeta) - C^{(j)}(y,\zeta)| \leq L_c\|x-y\|, \qquad \forall j \in \{1,2,\cdots,m\}, \quad \forall x,y \in \mathcal{X}, \quad \mathcal{P}_\zeta\text{-a.s.},$$

where $L_c := \kappa_{\nabla c} + \sigma_{\nabla c} < \infty$.

The proof of this lemma is provided in Section A.1.

Remark 3.1. *Assumptions 3.1, 3.2, and 3.3 are used to establish that the stochastic augmented Lagrangian gradient and Hessian estimators are Lipschitz continuous and that their errors relative to the corresponding exact quantities are uniformly bounded, as shown in the following technical lemmas. We note that these are relatively strong properties and may not be needed by all solvers that can be employed to solve the primal subproblem. However, solvers that employ variance reduction techniques typically require these properties to hold for the stochastic estimators themselves [7], rather than only for the corresponding expected quantities. Moreover, the almost sure error bounds in Assumption 3.3, or the almost sure boundedness of the stochastic quantities established in Lemma 3.1, are commonly utilized in constrained stochastic optimization; see, for example, [17, 70]. To provide a unified presentation across different potential solvers, including those considered in Section 4, which operate under varying degrees of strictness in their requirements on the augmented Lagrangian properties, we adopt these assumptions. For some solvers, it may suffice to assume Lipschitz continuity of the true augmented Lagrangian gradients and Hessians and boundedness of the errors in expectation. In such cases, the assumptions above can be equivalently imposed on the corresponding expected quantities, with the error bounds required only in expectation.*

Next, we provide technical lemmas establishing the properties of the stochastic augmented Lagrangian gradient and Hessian estimators.

Lemma 3.2. *Suppose Assumptions 2.1, 3.1, and 3.3 hold. At every iteration $k \in \mathbb{N}_0$ of Algorithm 1, suppose that the dual variable satisfies $\|\lambda_k\| \leq \Lambda < \infty$. Let*

$$L_1^{(g)} := L_{\nabla f} + \sqrt{m} L_{\nabla c}\Lambda, \text{ and } L_2^{(g)} := m(L_c(\sigma_{\nabla c} + \kappa_{\nabla c}) + L_{\nabla c}(\sigma_c + \kappa_c)).$$

Then, at each iteration $k \in \mathbb{N}_0$, the function $\nabla \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L^{(g)}_{\mathcal{L},k}$-Lipschitz continuous on $\mathcal{X}$ $\mathcal{P}_\theta$-almost surely. That is,

$$\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L^{(g)}_{\mathcal{L},k}\|x - y\|, \quad \forall k \in \mathbb{N}_0, \quad \forall x, y \in \mathcal{X}, \quad \mathcal{P}_\theta\text{-a.s..},$$

where $L^{(g)}_{\mathcal{L},k} \leq L^{(g)}_1 + \alpha_k L^{(g)}_2$.

Proof. Consider the difference between stochastic gradient estimators of augmented Lagrangian function defined in (3.1) at $x, y \in \mathcal{X}$,

$$\begin{aligned}
&\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \\
&\qquad = \Big\| (\nabla F(x, \xi) - \nabla F(y, \xi)) + \textstyle\sum_{j=1}^m \lambda_k^{(j)} (\nabla C^{(j)}(x, \zeta^{(1)}) - \nabla C^{(j)}(y, \zeta^{(1)})) \\
&\qquad\quad + \alpha_k (\nabla C(x, \zeta^{(1)})^\top C(x, \zeta^{(2)}) - \nabla C(y, \zeta^{(1)})^\top C(y, \zeta^{(2)})) \Big\| \\
&\qquad \leq \|\nabla F(x, \xi) - \nabla F(y, \xi)\| + \textstyle\sum_{j=1}^m |\lambda_k^{(j)}| \|\nabla C^{(j)}(x, \zeta^{(1)}) - \nabla C^{(j)}(y, \zeta^{(1)})\| \\
&\qquad\quad + \alpha_k \|\nabla C(x, \zeta^{(1)})^\top C(x, \zeta^{(2)}) - \nabla C(y, \zeta^{(1)})^\top C(y, \zeta^{(2)})\| \\
&\qquad \leq (L_{\nabla f} + \sqrt{m} L_{\nabla c} \Lambda) \|x - y\| + \alpha_k \|\nabla C(x, \zeta^{(1)})^\top C(x, \zeta^{(2)}) - \nabla C(y, \zeta^{(1)})^\top C(y, \zeta^{(2)})\|,
\end{aligned} \tag{3.3}$$

where the last inequality follows from Assumption 3.1 and $\|\lambda_k\|_1 \leq \sqrt{m}\|\lambda_k\| \leq \sqrt{m}\Lambda$. Now, consider,

$$\begin{aligned}
&\|(\nabla C(x, \zeta^{(1)})^\top C(x, \zeta^{(2)}) - \nabla C(y, \zeta^{(1)})^\top C(y, \zeta^{(2)}))\| \\
&\qquad = \|\nabla C(x, \zeta^{(1)})^\top (C(x, \zeta^{(2)}) - C(y, \zeta^{(2)})) + (\nabla C(x, \zeta^{(1)}) - \nabla C(y, \zeta^{(1)}))^\top C(y, \zeta^{(2)})\| \\
&\qquad \leq \textstyle\sum_{j=1}^m \left\|\nabla C^{(j)}(x, \zeta^{(1)})\right\| \left\|C^{(j)}(x, \zeta^{(2)}) - C^{(j)}(y, \zeta^{(2)})\right\| \\
&\qquad\quad + \textstyle\sum_{j=1}^m \left\|\nabla C^{(j)}(x, \zeta^{(1)}) - \nabla C^{(j)}(y, \zeta^{(1)})\right\| \left\|C^{(j)}(y, \zeta^{(2)})\right\| \\
&\qquad \leq m(L_c(\sigma_{\nabla c} + \kappa_{\nabla c}) + L_{\nabla c}(\sigma_c + \kappa_c)) \|x - y\|,
\end{aligned} \tag{3.4}$$

where the last inequality follows from Lemma 3.1. Substituting (3.4) in (3.3) completes the proof. □

Lemma 3.3. *Suppose Assumptions 2.1, 3.1, 3.2, and 3.3 hold. At every iteration $k \in \mathbb{N}_0$ of Algorithm 1, suppose that the dual variable satisfies $\|\lambda_k\| \leq \Lambda < \infty$. Let*

$$\begin{aligned}
L^{(H)}_1 &:= L_{\nabla^2 f} + \sqrt{m} L_{\nabla^2 c} \Lambda, \text{ and} \\
L^{(H)}_2 &:= m\left(2L_{\nabla c}(\sigma_{\nabla c} + \kappa_{\nabla c}) + L_c(\kappa_{\nabla^2 c} + \sigma_{\nabla^2 c}) + L_{\nabla^2 c}(\sigma_c + \kappa_c)\right).
\end{aligned}$$

Then, at each iteration $k \in \mathbb{N}_0$, the function $\nabla^2 \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L^{(H)}_{\mathcal{L},k}$-Lipschitz continuous on $\mathcal{X}$ $\mathcal{P}_\theta$-almost surely. That is,

$$\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L^{(H)}_{\mathcal{L},k}\|x - y\|, \quad \forall k \in \mathbb{N}_0, \quad \forall x, y \in \mathcal{X}, \quad \mathcal{P}_\theta\text{-a.s.},$$

where $L^{(H)}_{\mathcal{L},k} \leq L^{(H)}_1 + \alpha_k L^{(H)}_2$.

Proof. Consider the difference between stochastic Hessian estimators of augmented Lagrangian function

defined in (3.2) at $x, y \in \mathcal{X}$,

$$
\begin{aligned}
&\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \\
&\qquad = \Big\| (\nabla^2 F(x, \xi) - \nabla^2 F(y, \xi)) + \sum_{j=1}^{m} \lambda_k^{(j)} (\nabla^2 C^{(j)}(x, \zeta^{(1)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)})) \\
&\qquad\quad + \alpha_k \sum_{j=1}^{m} (\nabla C^{(j)}(x, \zeta^{(1)}) \nabla C^{(j)}(x, \zeta^{(2)})^\top - \nabla C^{(j)}(y, \zeta^{(1)}) \nabla C^{(j)}(y, \zeta^{(2)})^\top) \\
&\qquad\quad + \alpha_k \sum_{j=1}^{m} (\nabla^2 C^{(j)}(x, \zeta^{(1)}) C^{(j)}(x, \zeta^{(2)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)}) C^{(j)}(y, \zeta^{(2)})) \Big\| \\
&\qquad \le \underbrace{\|\nabla^2 F(x, \xi) - \nabla^2 F(y, \xi)\| + \sum_{j=1}^{m} |\lambda_k^{(j)}| \|(\nabla^2 C^{(j)}(x, \zeta^{(1)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)}))\|}_{T_1} \\
&\qquad\quad + \alpha_k \sum_{j=1}^{m} \underbrace{\|(\nabla C^{(j)}(x, \zeta^{(1)}) \nabla C^{(j)}(x, \zeta^{(2)})^\top - \nabla C^{(j)}(y, \zeta^{(1)}) \nabla C^{(j)}(y, \zeta^{(2)})^\top)\|}_{T_2} \\
&\qquad\quad + \alpha_k \sum_{j=1}^{m} \underbrace{\|(\nabla^2 C^{(j)}(x, \zeta^{(1)}) C^{(j)}(x, \zeta^{(2)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)}) C^{(j)}(y, \zeta^{(2)}))\|}_{T_3}. \qquad (3.5)
\end{aligned}
$$

We now bound each of the norm terms T_1, T_2, and T_3 defined above separately. First, for T_1, by Assumption 3.2 and $\|\lambda_k\|_1 \le \sqrt{m}\|\lambda_k\| \le \sqrt{m}\Lambda$, we have,

$$
T_1 \le L_{\nabla^2 f}\|x - y\| + \sum_{j=1}^{m} |\lambda_k^{(j)}| L_{\nabla^2 c}\|x - y\| \le (L_{\nabla^2 f} + \sqrt{m} L_{\nabla^2 c}\Lambda)\|x - y\|. \qquad (3.6)
$$

Next, we bound T_2 for any $j \in \{1, \ldots, m\}$ as follows:

$$
\begin{aligned}
&\|\nabla C^{(j)}(x, \zeta^{(1)}) \nabla C^{(j)}(x, \zeta^{(2)})^\top - \nabla C^{(j)}(y, \zeta^{(1)}) \nabla C^{(j)}(y, \zeta^{(2)})^\top\| \\
&\qquad = \|\nabla C^{(j)}(x, \zeta^{(1)}) \nabla C^{(j)}(x, \zeta^{(2)})^\top - \nabla C^{(j)}(x, \zeta^{(1)}) \nabla C^{(j)}(y, \zeta^{(2)})^\top \\
&\qquad\quad + \nabla C^{(j)}(x, \zeta^{(1)}) \nabla C^{(j)}(y, \zeta^{(2)})^\top - \nabla C^{(j)}(y, \zeta^{(1)}) \nabla C^{(j)}(y, \zeta^{(2)})^\top\| \\
&\qquad \le \|\nabla C^{(j)}(x, \zeta^{(1)})(\nabla C^{(j)}(x, \zeta^{(2)}) - \nabla C^{(j)}(y, \zeta^{(2)}))^\top\| \\
&\qquad\quad + \|(\nabla C^{(j)}(x, \zeta^{(1)}) - \nabla C^{(j)}(y, \zeta^{(1)})) \nabla C^{(j)}(y, \zeta^{(2)})^\top\| \\
&\qquad \le \|\nabla C^{(j)}(x, \zeta^{(1)})\| \|\nabla C^{(j)}(x, \zeta^{(2)}) - \nabla C^{(j)}(y, \zeta^{(2)})\| \\
&\qquad\quad + \|\nabla C^{(j)}(x, \zeta^{(1)}) - \nabla C^{(j)}(y, \zeta^{(1)})\| \|\nabla C^{(j)}(y, \zeta^{(2)})\| \\
&\qquad \le 2(\kappa_{\nabla c} + \sigma_{\nabla c}) L_{\nabla c}\|x - y\|, \qquad (3.7)
\end{aligned}
$$

where the final inequality follows from Assumption 3.1 and Lemma 3.1.

Finally, we bound T_3 for any $j \in \{1, \ldots, m\}$ as follows:

$$\begin{aligned}
&\|\nabla^2 C^{(j)}(x, \zeta^{(1)}) C^{(j)}(x, \zeta^{(2)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)}) C^{(j)}(y, \zeta^{(2)})\| \\
&\quad = \Big\| \nabla^2 C^{(j)}(x, \zeta^{(1)}) C^{(j)}(x, \zeta^{(2)}) - \nabla^2 C^{(j)}(x, \zeta^{(1)}) C^{(j)}(y, \zeta^{(2)}) \\
&\qquad + \nabla^2 C^{(j)}(x, \zeta^{(1)}) C^{(j)}(y, \zeta^{(2)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)}) C^{(j)}(y, \zeta^{(2)}) \Big\| \\
&\quad \le \|\nabla^2 C^{(j)}(x, \zeta^{(1)})(C^{(j)}(x, \zeta^{(2)}) - C^{(j)}(y, \zeta^{(2)}))\| \\
&\qquad + \|(\nabla^2 C^{(j)}(x, \zeta^{(1)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)})) C^{(j)}(y, \zeta^{(2)})\| \\
&\quad \le \|\nabla^2 C^{(j)}(x, \zeta^{(1)})\| |C^{(j)}(x, \zeta^{(2)}) - C^{(j)}(y, \zeta^{(2)})| \\
&\qquad + \|\nabla^2 C^{(j)}(x, \zeta^{(1)}) - \nabla^2 C^{(j)}(y, \zeta^{(1)})\| |C^{(j)}(y, \zeta^{(2)})| \\
&\quad \le (L_c\,(\sigma_{\nabla^2 c} + \kappa_{\nabla^2 c}) + L_{\nabla^2 c}\,(\sigma_c + \kappa_c))\|x - y\|, \qquad (3.8)
\end{aligned}$$

where the final inequality follows from Assumption 3.2 and Lemma 3.1. Substituting (3.6), (3.7), and (3.8) in (3.5) completes the proof.

□

Lemma 3.4. *Suppose Assumptions 2.1 and 3.3 hold. At every iteration $k \in \mathbb{N}_0$ of Algorithm 1, suppose that the dual variable satisfies $\|\lambda_k\| \le \Lambda < \infty$. Let*

$$\sigma_1^{(g)} := \sigma_{\nabla f} + \sigma_{\nabla c}\Lambda, \quad \text{and} \quad \sigma_2^{(g)} := \sigma_{\nabla c}\sigma_c + \sigma_{\nabla c}\kappa_c + \kappa_{\nabla c}\sigma_c.$$

Then, at each iteration $k \in \mathbb{N}_0$, the stochastic gradient error of $\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ is uniformly bounded by $\sigma_{\mathcal{L},k}^{(g)}$ on $\mathcal{X}$ $\mathcal{P}_\theta$-almost surely. That is,

$$\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\| \le \sigma_{\mathcal{L},k}^{(g)}, \quad \forall k \in \mathbb{N}_0, \quad \forall x \in \mathcal{X}, \quad \mathcal{P}_\theta\text{-a.s.},$$

where $\sigma_{\mathcal{L},k}^{(g)} \le \sigma_1^{(g)} + \alpha_k \sigma_2^{(g)}$.

Proof. For each iteration $k \in \mathbb{N}_0$, using Assumption 2.1, Assumption 3.3, Lemma 3.1, and $\|\lambda_k\| \le \Lambda$, we get,

$$\begin{aligned}
&\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\| \\
&\quad = \|\nabla F(x, \xi) - f(x) + (\nabla C(x, \zeta^{(1)}) - \nabla c(x))^\top \lambda_k + \alpha_k (\nabla C(x, \zeta^{(1)})^\top C(x, \zeta^{(2)}) - \nabla c(x)^\top c(x))\| \\
&\quad \le \|\nabla F(x, \xi) - f(x)\| + \|\nabla C(x, \zeta^{(1)}) - \nabla c(x)\| \|\lambda_k\| + \alpha_k \|\nabla C(x, \zeta^{(1)})^\top C(x, \zeta^{(2)}) - \nabla c(x)^\top c(x)\| \\
&\quad \le \sigma_{\nabla f} + \sigma_{\nabla c}\Lambda + \alpha_k \|(\nabla C(x, \zeta^{(1)}) - \nabla c(x))^\top C(x, \zeta^{(2)}) + \nabla c(x)^\top (C(x, \zeta^{(2)}) - c(x))\| \\
&\quad \le \sigma_{\nabla f} + \sigma_{\nabla c}\Lambda + \alpha_k (\|\nabla C(x, \zeta^{(1)}) - \nabla c(x)\| \|C(x, \zeta^{(2)})\| + \|\nabla c(x)\| \|C(x, \zeta^{(2)}) - c(x)\|) \\
&\quad \le \sigma_{\nabla f} + \sigma_{\nabla c}\Lambda + \alpha_k (\sigma_{\nabla c}(\sigma_c + \kappa_c) + \kappa_{\nabla c}\sigma_c).
\end{aligned}$$

□

Finally, we state an existing result on the error bounds of stochastic Hessian estimators, provided that the stochastic gradient estimators are Lipschitz continuous almost surely.

Lemma 3.5 (Page 4, [61]; [62])**.** *Suppose Assumptions 2.1, 3.1, and 3.3 hold. At every iteration $k \in \mathbb{N}_0$ of Algorithm 1, suppose that the dual variable satisfies $\|\lambda_k\| \le \Lambda < \infty$. Let*

$$\begin{aligned}
&L_1^{(g)} := L_{\nabla f} + m L_{\nabla c}\Lambda,\ L_2^{(g)} := m(L_{\nabla c}(\sigma_c + \kappa_c) + L_c(\sigma_{\nabla c} + \kappa_{\nabla c})), \\
&\sigma_1^{(H)} := 2L_1^{(g)}, \quad \text{and} \quad \sigma_2^{(H)} := 2L_2^{(g)}.
\end{aligned}$$

Then, at each iteration $k \in \mathbb{N}_0$, the stochastic Hessian error of $\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ is uniformly bounded by $\sigma_{\mathcal{L},k}^{(H)}$ on $\mathcal{X}$ $\mathcal{P}_\theta$-almost surely. That is,

$$\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \mathcal{L}_{\alpha_k}(x, \lambda_k)\| \leq \sigma_{\mathcal{L},k}^{(H)}, \quad \forall k \in \mathbb{N}_0, \quad \forall x \in \mathcal{X}, \quad \mathcal{P}_\theta\text{-a.s.},$$

where $\sigma_{\mathcal{L},k}^{(H)} \leq \sigma_1^{(H)} + \alpha_k \sigma_2^{(H)}$.

Remark 3.2. *We note that each of the terms $L_{\mathcal{L},k}^{(g)}$, $L_{\mathcal{L},k}^{(H)}$, $\sigma_{\mathcal{L},k}^{(g)}$, and $\sigma_{\mathcal{L},k}^{(H)}$ defined in Lemmas 3.2 to 3.5 is bounded above by an expression of the form $c_1 + \alpha_k c_2$ for all $k \in \mathbb{N}_0$, where c_1 and c_2 are constants independent of k. Since α_k is an increasing sequence, these upper bounds are also increasing in k. Consequently, the quantities governing the sample complexity of the primal solver $\mathcal{M}$ can be bounded in terms of the penalty parameter α_k.*

We now establish bounds on the initial optimality gap of the primal subproblems at each iteration k.

Lemma 3.6. *Suppose Assumption 2.1 holds. At every iteration $k \in \mathbb{N}_0$ of Algorithm 1, suppose that the dual variable satisfies $\|\lambda_k\| \leq \Lambda < \infty$ and $\epsilon_k^{(g)} < 1$. For all $k \in \mathbb{N}_0$, let $x_k^* \in \arg\min_{x \in \mathcal{X}} \mathcal{L}_{\alpha_k}(x, \lambda_k)$, and define*

$$\Delta_k := \mathcal{L}_{\alpha_k}(x_{k-1}, \lambda_k) - \mathcal{L}_{\alpha_k}(x_k^*, \lambda_k).$$

Further, define,

$$\Delta_k^* := 2\kappa_f + \frac{\Lambda^2}{\alpha_0} + \alpha_k \kappa_c^2, \quad \text{and} \quad \Delta^* := 2\kappa_f + \frac{\Lambda^2}{\alpha_0} + \frac{\kappa_c \beta (1 + \kappa_{\nabla c} + \Lambda \kappa_{\nabla f})}{C_{reg}}.$$

Then, for all $k \in \mathbb{N}_0$,

$$\Delta_k \leq \Delta_k^* = \mathcal{O}(\alpha_k). \tag{3.9}$$

Further suppose Assumption 2.2 holds. Then, for Algorithm 1

*(i) under **Option I**, for all $k \geq 1$,*

$$\mathbb{E}[\Delta_k] \leq \Delta^*. \tag{3.10}$$

*(ii) under **Option II**, for all $k \geq 1$,*

$$\mathbb{P}(\Delta_k \leq \Delta^* \mid \mathcal{F}_{k-1}) \geq p_{k-1}. \tag{3.11}$$

Furthermore, for any $K \geq 1$,

$$\mathbb{P}(\cap_{k=1}^K \{\Delta_k \leq \Delta^*\}) \geq \prod_{k=0}^{K-1} p_k. \tag{3.12}$$

Proof. Using Young's inequality, Assumption 2.1, $\|\lambda_k\| \leq \Lambda$, and $\alpha_k \geq \alpha_0$ for all $k \in \mathbb{N}_0$, we have,

$$\begin{aligned}
&\mathcal{L}_{\alpha_k}(x_{k-1}, \lambda_k) - \mathcal{L}_{\alpha_k}(x_k^*, \lambda_k) \\
&\quad = f(x_{k-1}) + \lambda_k^\top c(x_{k-1}) + \tfrac{\alpha_k}{2}\|c(x_{k-1})\|^2 - f(x_k^*) - \lambda_k^\top c(x_k^*) - \tfrac{\alpha_k}{2}\|c(x_k^*)\|^2 \\
&\quad \leq f(x_{k-1}) + \lambda_k^\top c(x_{k-1}) + \tfrac{\alpha_k}{2}\|c(x_{k-1})\|^2 - f(x_k^*) + \tfrac{1}{2\alpha_k}\|\lambda_k\|^2 \\
&\quad \leq 2\kappa_f + \|\lambda_k\|\|c(x_{k-1})\| + \tfrac{\alpha_k}{2}\|c(x_{k-1})\|^2 + \tfrac{\|\lambda_k\|^2}{2\alpha_k} \\
&\quad \leq 2\kappa_f + \Lambda\|c(x_{k-1})\| + \tfrac{\alpha_k}{2}\|c(x_{k-1})\|^2 + \tfrac{\Lambda^2}{2\alpha_k} \\
&\quad \leq 2\kappa_f + \alpha_k\|c(x_{k-1})\|^2 + \tfrac{\Lambda^2}{\alpha_0} \qquad (3.13) \\
&\quad \leq 2\kappa_f + \alpha_k\kappa_c^2 + \tfrac{\Lambda^2}{\alpha_0}. \qquad (3.14)
\end{aligned}$$

Therefore, $\Delta_k \leq \Delta_k^* = \mathcal{O}(\alpha_k)$ for all $k \in \mathbb{N}_o$, the equality is due to the fact that α_k keeps growing with k, whereas $\kappa_f, \kappa_c, \alpha_0$, and Λ are constant with respect to k.

Case (i): If **Option I** is chosen in Algorithm 1, then, taking expectation in (3.13), and using Assumption 2.1, (2.9), (2.12) and $\epsilon_k^{(g)} < 1$, it holds for all $k \geq 1$ that

$$\begin{aligned}\mathbb{E}[\mathcal{L}_{\alpha_k}(x_{k-1}, \lambda_k) - \mathcal{L}_{\alpha_k}(x_k^*, \lambda_k)] &\leq 2\kappa_f + \tfrac{\Lambda^2}{\alpha_0} + \alpha_k \mathbb{E}[\|c(x_{k-1})\|^2] \\ &\leq 2\kappa_f + \tfrac{\Lambda^2}{\alpha_0} + \kappa_c \beta \alpha_{k-1} \mathbb{E}[\|c(x_{k-1})\|] \\ &\leq 2\kappa_f + \tfrac{\Lambda^2}{\alpha_0} + \tfrac{\kappa_c \beta (1 + \kappa_{\nabla c} + \Lambda \kappa_{\nabla f})}{C_{\text{reg}}}.\end{aligned}$$

Case (ii): Similarly, if **Option II** is chosen in Algorithm 1, then, using (3.13), Assumption 2.1, (2.9), and (2.5), we have, for all $k \geq 1$ that

$$\begin{aligned}&\mathbb{P}(\mathcal{L}_{\alpha_k}(x_{k-1}, \lambda_k) - \mathcal{L}_{\alpha_k}(x_k^*, \lambda_k) \leq \Delta^* \mid \mathcal{F}_{k-1}) \\ &\quad \geq \mathbb{P}(2\kappa_f + \alpha_k \|c(x_{k-1})\|^2 + \tfrac{\Lambda^2}{\alpha_0} \leq \Delta^* \mid \mathcal{F}_{k-1}) \\ &\quad \geq \mathbb{P}(\|\text{proj}_{T_{\mathcal{X}}(x_{k-1})}(-\nabla \mathcal{L}_{\alpha_{k-1}}(x_{k-1}, \lambda_{k-1}))\| \leq \epsilon_{k-1}^{(g)} \mid \mathcal{F}_{k-1}) \\ &\quad \geq p_{k-1} \qquad (3.15)\end{aligned}$$

For $k \in \{1, \ldots, K\}$, define the events

$$D_k := \{\Delta_k \leq \Delta^*\}, \quad D := \cap_{k=1}^K D_k.$$

Using (3.15) and indicator function, we have

$$\mathbb{P}(D_k \mid \mathcal{F}_{k-1}) = \mathbb{E}[\mathbf{1}_{D_k} \mid \mathcal{F}_{k-1}] \geq p_{k-1}, \quad \forall k = \{1, \cdots, K\}. \qquad (3.16)$$

Considering $\mathbb{P}(D)$ and using (3.16), we obtain

$$\begin{aligned}\mathbb{P}(D) &= \mathbb{E}[\textstyle\prod_{k=1}^K \mathbf{1}_{D_k}] \qquad (3.17) \\ &= \mathbb{E}[\mathbf{1}_{D_K} \textstyle\prod_{k=1}^{K-1} \mathbf{1}_{D_k}] \\ &= \mathbb{E}[\mathbb{E}[\mathbf{1}_{D_K} \textstyle\prod_{k=1}^{K-1} \mathbf{1}_{D_k} \mid \mathcal{F}_{K-1}]] \\ &= \mathbb{E}[\mathbb{E}[\mathbf{1}_{D_K} \mid \mathcal{F}_{K-1}] \textstyle\prod_{k=1}^{K-1} \mathbf{1}_{D_k}] \\ &\geq p_{K-1} \mathbb{E}[\textstyle\prod_{k=1}^{K-1} \mathbf{1}_{D_k}]. \qquad (3.18)\end{aligned}$$

where the fourth equality follows because $\prod_{k=1}^{K-1} \mathbf{1}_{D_k}$ is determined by x_{K-2} and λ_{K-1}. Applying the same argument recursively yields

$$\mathbb{P}(D) \geq \prod_{k=0}^{K-1} p_k.$$

□

Remark 3.3. *We established three bounds on the initial optimality gap Δ_k of the primal subproblem at each iteration in Lemma 3.6. These bounds are useful for establishing the sample complexity results. Among the three bounds, the deterministic bound in (3.9) provides the strongest type of guarantee, as it holds deterministically; however, the corresponding bound Δ_k^* increases with k and is therefore less refined than the bounds obtained in expectation and in probability. We use the deterministic bound to establish deterministic sample complexity results for the different primal solvers considered in Section 4. In contrast, the expectation bound in (3.10) and the probability bound in (3.11) provide weaker types of guarantees, but yield the improved uniform bound $\Delta_k \leq \Delta^*$. These bounds are therefore used to establish stronger sample complexity*

results in expectation and in probability, respectively. Such expectation-based initial optimality gap bounds have also been utilized in the literature for analyzing stochastic subproblem solvers; see, e.g., [39, Lemma 5]. The improvement in our expectation and probability bounds follows from exploiting the fact that $\|c(x_{k-1})\|$ decreases as the iterations progress, either in expectation or in probability, respectively. Nevertheless, these bounds remain weaker than those established in the strongly convex setting in [12], where the optimality gap is shown to decrease as the iterations progress due to the strong convexity of the primal subproblems considered therein.

Based on the technical Lemmas 3.2 to 3.6, we introduce the following definition for ease of presentation.

Definition 3.1. *Suppose Assumptions 2.1, 2.2, 3.1, 3.2, and 3.3 hold. Then, for each iteration $k \in \mathbb{N}_0$ of Algorithm 1, we define the following quantities:*

- *Lipschitz constant $L_{\mathcal{L},k}^{(g)}$ for $\nabla \bar{\mathcal{L}}_{\alpha_k}(\cdot, \lambda_k; \theta)$ as per Lemma 3.2.*
- *Lipschitz constant $L_{\mathcal{L},k}^{(H)}$ for $\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(\cdot, \lambda_k; \theta)$ as per Lemma 3.3.*
- *Variance bound $\sigma_{\mathcal{L},k}^{(g)}$ for $\nabla \bar{\mathcal{L}}_{\alpha_k}(\cdot, \lambda_k; \theta)$ as per Lemma 3.4.*
- *Variance bound $\sigma_{\mathcal{L},k}^{(H)}$ for $\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(\cdot, \lambda_k; \theta)$ as per Lemma 3.5.*
- *Initial optimality gap Δ_k as per Lemma 3.6.*

We are now ready to provide the sample complexity results for the MESCAL framework. We first define the work complexity of a primal solver $\mathcal{M}$.

Definition 3.2. *We define the work complexity of a primal solver $\mathcal{M}$ employed at iteration $k \in \mathbb{N}_0$ in Algorithm 1, denoted by $\mathcal{W}_{\mathcal{M},k}^{(p)}$, as the total number of stochastic augmented Lagrangian gradient evaluations $\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ and stochastic augmented Lagrangian Hessian vector product evaluations $\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)v$ for arbitrary vectors v.*

We next state a technical lemma establishing a bound on the sample complexity of Algorithm 1 in terms of the work complexity of the primal solver $\mathcal{M}$. We set the dual sample sizes to $|S_k^{(d)}| = \Theta((\epsilon^{(g)})^{-2})$ for all $k \in \mathbb{N}_0$, as described in Remark 2.3.

Lemma 3.7. *Suppose Assumptions 2.1 and 2.2 hold. Let $\{x_k\}$ be the sequence of iterates generated by Algorithm 1 with initial iterate $x_{-1} \in \mathcal{X}$ and initial dual iterate $\lambda_0 \in \mathbb{R}^m$. Suppose that the dual bound parameters $\gamma_k > 0$ be a bounded sequence satisfying $\sum_{k=0}^{\infty} \gamma_k = C_\gamma < \infty$, and that the dual sample sizes satisfy $|S_k^{(d)}| = \Theta((\epsilon^{(g)})^{-2})$ for all $k \in \mathbb{N}_0$. If the error bounds are chosen such that $\epsilon_k^{(g)} = \epsilon^{(g)} < 1$ and $\epsilon_k^{(H)} = \epsilon^{(H)} < 1$ for any $k \in \mathbb{N}_0$, then, for any $k = K \geq K^*$, where K^* is defined in (2.7), the sample complexity $\mathcal{W}$, as defined in Definition 1.3, satisfies*

$$\mathcal{W} = \tilde{\mathcal{O}}\left(\left(\sum_{k=0}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)}\right) + (\epsilon^{(g)})^{-2}\right), \tag{3.19}$$

where $\mathcal{W}_{\mathcal{M},k}^{(p)}$ is the work complexity of solver $\mathcal{M}$ defined in Definition 3.2.

Proof. Let $\mathcal{W}_{\mathcal{M}}^{(p)}$ denote the total number of stochastic objective gradient, objective Hessian vector product, constraint function, constraint gradient, and constraint Hessian vector product evaluations used by the solver $\mathcal{M}$ over iterations $k = 0, \ldots, K$. Let $\mathcal{W}^{(d)}$ denote the total number of stochastic constraint function evaluations used in the dual updates over the same iterations. Then,

$$\mathcal{W} = \mathcal{W}_{\mathcal{M}}^{(p)} + \mathcal{W}^{(d)}. \tag{3.20}$$

We first consider the work associated with the dual updates. At each iteration k, the dual update uses $m|S_k^{(d)}|$ stochastic constraint function evaluations. Thus,

$$\mathcal{W}^{(d)} = \sum_{k=0}^{K} m|S_k^{(d)}| = \sum_{k=0}^{K} \Theta\left((\epsilon^{(g)})^{-2}\right) = \mathcal{O}\left((K+1)(\epsilon^{(g)})^{-2}\right) = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-2}\right), \tag{3.21}$$

where the last equality is due to $K^* = \mathcal{O}(\log(1/\epsilon^{(g)}))$.

Next, we consider the work associated with the primal subproblems. From (3.1) and (3.2), each stochastic augmented Lagrangian gradient evaluation requires a constant number of objective gradient, constraint function, and constraint gradient evaluations, while each stochastic augmented Lagrangian Hessian vector product evaluation requires a constant number of objective and constraint Hessian vector product evaluations, constraint function evaluations, and constraint gradient evaluations. Since the number of constraint components is m, which is fixed, the total number of underlying stochastic oracle evaluations required at each outer iteration is of the same order as $\mathcal{W}_{\mathcal{M},k}^{(p)}$. Therefore,

$$\mathcal{W}_{\mathcal{M}}^{(p)} = \mathcal{O}\left(\sum_{k=0}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)}\right). \tag{3.22}$$

Substituting (3.21) and (3.22) in (3.20) completes the proof. □

We make the following assumption on the theoretical sample complexity guarantees of a general solver $\mathcal{M}$.

Assumption 3.4. *Suppose that Assumptions 2.1, 2.2, 3.1, 3.2, and 3.3 hold. For each $k \in \mathbb{N}_0$, define*

$$\mathcal{B}_{\mathcal{M},k} = \min\left\{\frac{\Delta_k \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(g)})^3}, \frac{\Delta_k \sigma_{\mathcal{L},k}^{(g)} (L_{\mathcal{L},k}^{(H)})^{0.5}}{(\epsilon_k^{(g)})^{3.5}}, \frac{\Delta_k \sigma_{\mathcal{L},k}^{(g)} L_{\mathcal{L},k}^{(g)}}{(\epsilon_k^{(g)})^4}\right\} + \frac{\Delta_k (\sigma_{\mathcal{L},k}^{(H)})^2 (L_{\mathcal{L},k}^{(H)})^2}{(\epsilon_k^{(H)})^5}, \tag{3.23}$$

where $\{\Delta_k\}$, $\{L_{\mathcal{L},k}^{(g)}\}$, $\{L_{\mathcal{L},k}^{(H)}\}$, $\{\sigma_{\mathcal{L},k}^{(g)}\}$, $\{\sigma_{\mathcal{L},k}^{(H)}\}$ are defined in Definition 3.1. We assume that one of the following holds:

*(i) Under **Option I**, $\mathcal{M}$ returns an iterate satisfying (2.4) with*

$$\mathcal{W}_{\mathcal{M},k}^{(p)} = \tilde{\mathcal{O}}(\mathcal{B}_{\mathcal{M},k}).$$

*(ii) Under **Option II**, $\mathcal{M}$ returns an iterate satisfying (2.5) for any $r \geq 1$, with*

$$\mathcal{W}_{\mathcal{M},k}^{(p)} = \mathcal{O}\left(\mathcal{B}_{\mathcal{M},k} \log^r\left(\tfrac{1}{1-p_k}\right)\right).$$

Remark 3.4. *We make the following remarks regarding Assumption 3.4.*

- *We are unaware of a stochastic second-order solver $\mathcal{M}$ with established work complexity guarantees for the general convex set $\mathcal{X}$ considered in this paper. We therefore formulate work complexity bounds guided by the lower bound result in [6, Eq. (13)]. In [6], Arjevani et al. established lower bounds for computing an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate stationary point satisfying probabilistic conditions (2.5), for $p \geq 0.5$ and $\mathcal{X} = \mathbb{R}^d$. These lower bounds are consistent with the upper bounds available for SG-HV-NC [6, Algorithm 4], Natasha2 [3], and SPIDER [24]. The corresponding MESCAL complexity results for these methods are established in Section 4, under the restriction $\mathcal{X} = \mathbb{R}^d$.*

- *The lower bound in [6, Eq. (13)] does not characterize the dependence on the desired success probability p. We therefore include a polylogarithmic dependence on the failure probability $1-p$, so that the work complexity increases as p tends to one. Such dependence is common in high-probability stochastic optimization guarantees; see, for example, [18, 26, 55, 57]. Furthermore, the dependence on initial optimality gap, Lipschitz constants, and variance bounds is essential within the MESCAL framework for accurate calculations of sample complexity bounds, as these terms may contribute to increasing the complexity bound. Accordingly, these constants are included explicitly in the work complexity bounds.*

- *Although the lower bound in [6, Eq. (13)] is stated for $\mathcal{X} = \mathbb{R}^d$, several proximal methods attain work complexities comparable to those of their unconstrained counterparts; see, e.g., [11, 37, 40]. In addition, high-probability guarantees often differ from their expectation-based counterparts only by logarithmic factors under suitable assumptions; see [18, 57].*
- *The method proposed and analyzed in [9] computes an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate second-order stationary point under $\mathcal{X} = \mathbb{R}^d$ satisfying (2.4), but its work complexity is not established therein. We therefore also analyze MESCAL under **Option I**, assuming that the solver $\mathcal{M}$ satisfies the expected work complexity bound in Assumption 3.4. The resulting sample complexity guarantees consequently apply to any solver satisfying these conditions.*

Theorem 3.8. *Suppose Assumptions 2.1, 2.2, 3.1, 3.2, 3.3, and 3.4 hold. Let $\{x_k\}$ be the sequence of iterates generated by Algorithm 1 with initial iterate $x_{-1} \in \mathcal{X}$ and initial dual iterate $\lambda_0 \in \mathbb{R}^m$. Suppose that the dual bound parameters $\gamma_k > 0$ be a bounded sequence satisfying $\sum_{k=0}^{\infty} \gamma_k = C_\gamma < \infty$, and that the dual sample sizes satisfy $|S_k^{(d)}| = \Theta((\epsilon^{(g)})^{-2})$ for all $k \in \mathbb{N}_0$. If the error bounds are chosen such that $\epsilon_k^{(g)} = \epsilon^{(g)} < 1$ and $\epsilon_k^{(H)} = \epsilon^{(H)} < 1$ for any $k \in \mathbb{N}_0$, then, for any $k = K \geq K^*$, where K^* is defined in (2.7), the sample complexity $\mathcal{W}$, as defined in Definition 1.3, satisfies the following guarantees:*

*(i) under **(Option I)**, when Assumption 3.4(i) holds, the expected sample complexity to obtain an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate expected second-order stationary point $\tilde{x} := x_K$ satisfying Definition 1.1 for solving (1.1), with $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$, is*

$$\mathbb{E}[\mathcal{W}] = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}).$$

*(ii) under **(Option II)** with $K = K^*$, when Assumption 3.4(ii) holds and $p_k = p^{1/K}$, the sample complexity to obtain an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point $\tilde{x} := x_K$ satisfying Definition 1.2 for solving (1.1), with $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$, is*

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}),$$

with probability at least p.

Proof. By Lemmas 3.2-3.5, we have,

$$\begin{aligned} L_{\mathcal{L},k}^{(g)} &\leq L_1^{(g)} + \alpha_k L_2^{(g)} = \mathcal{O}(\alpha_k), \quad L_{\mathcal{L},k}^{(H)} \leq L_1^{(H)} + \alpha_k L_2^{(H)} = \mathcal{O}(\alpha_k), \\ \sigma_{\mathcal{L},k}^{(g)} &\leq \sigma_1^{(g)} + \alpha_k \sigma_2^{(g)} = \mathcal{O}(\alpha_k), \quad \sigma_{\mathcal{L},k}^{(H)} \leq \sigma_1^{(H)} + \alpha_k \sigma_2^{(H)} = \mathcal{O}(\alpha_k), \end{aligned} \tag{3.24}$$

where the equalities are due to the fact that α_k keeps growing with k, whereas $L_1^{(g)}$, $L_2^{(g)}$, $L_1^{(H)}$, $L_2^{(H)}$, $\sigma_1^{(g)}$, $\sigma_2^{(g)}$, $\sigma_1^{(H)}$, and $\sigma_2^{(H)}$ are constant with respect to k. Let

$$\mathscr{C}_{\mathcal{M},k} = \min\left\{ \frac{\sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(g)})^3}, \frac{\sigma_{\mathcal{L},k}^{(g)} (L_{\mathcal{L},k}^{(H)})^{0.5}}{(\epsilon_k^{(g)})^{3.5}}, \frac{\sigma_{\mathcal{L},k}^{(g)} L_{\mathcal{L},k}^{(g)}}{(\epsilon_k^{(g)})^4} \right\} + \frac{(\sigma_{\mathcal{L},k}^{(H)})^2 (L_{\mathcal{L},k}^{(H)})^2}{(\epsilon_k^{(H)})^5}. \tag{3.25}$$

Therefore, from (3.23), we have $\mathcal{B}_{\mathcal{M},k} = \Delta_k \mathscr{C}_{\mathcal{M},k}$. Considering $\mathscr{C}_{\mathcal{M},k}$ and using (3.24), $\epsilon_k^{(g)} = \epsilon^{(g)}$, and $\epsilon_k^{(H)} = \epsilon^{(H)}$, we obtain

$$\begin{aligned} \mathscr{C}_{\mathcal{M},k} &= \min\left\{ \frac{\sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(g)})^3}, \frac{\sigma_{\mathcal{L},k}^{(g)} (L_{\mathcal{L},k}^{(H)})^{0.5}}{(\epsilon_k^{(g)})^{3.5}}, \frac{\sigma_{\mathcal{L},k}^{(g)} L_{\mathcal{L},k}^{(g)}}{(\epsilon_k^{(g)})^4} \right\} + \frac{(\sigma_{\mathcal{L},k}^{(H)})^2 (L_{\mathcal{L},k}^{(H)})^2}{(\epsilon_k^{(H)})^5} \\ &= \mathcal{O}\left(\min\left\{ \frac{(\alpha_k)^2}{(\epsilon^{(g)})^3}, \frac{(\alpha_k)^{1.5}}{(\epsilon^{(g)})^{3.5}}, \frac{(\alpha_k)^2}{(\epsilon^{(g)})^4} \right\} + \frac{(\alpha_k)^4}{(\epsilon^{(H)})^5} \right). \end{aligned} \tag{3.26}$$

Case (i): Under **Option I**, from Assumption 3.4(i), we establish that the iterate x_k satisfies Condition (2.4) in Algorithm 1. Therefore, by Theorem 2.1, it holds that $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate expected second-order stationary point as defined in Definition 1.1.

At iteration $k = 0$, the work complexity of solver $\mathcal{M}$ is given by

$$\mathcal{W}_{\mathcal{M},0}^{(p)} = \tilde{\mathcal{O}}(\Delta_0^* \mathscr{C}_{\mathcal{M},0}) = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-3} + (\epsilon^{(H)})^{-5}\right), \tag{3.27}$$

where Δ_0^* is defined in Lemma 3.6.

Using (3.10) from Lemma 3.6(i), in expectation, the total work complexity over iterations $1, 2, \cdots, K$ for solver $\mathcal{M}$ is

$$\begin{aligned}
\sum_{k=1}^{K} \mathbb{E}[\mathcal{W}_{\mathcal{M},k}^{(p)}] &= \tilde{\mathcal{O}}\left(\sum_{k=1}^{K} \mathbb{E}[\Delta_k \mathscr{C}_{\mathcal{M},k}]\right) = \tilde{\mathcal{O}}\left(\sum_{k=1}^{K} \mathbb{E}[\Delta_k] \mathscr{C}_{\mathcal{M},k}\right) \\
&= \tilde{\mathcal{O}}\left(\sum_{k=1}^{K} \min\left\{\tfrac{(\alpha_k)^2}{(\epsilon^{(g)})^3}, \tfrac{(\alpha_k)^{1.5}}{(\epsilon^{(g)})^{3.5}}, \tfrac{(\alpha_k)^2}{(\epsilon^{(g)})^4}\right\} + \tfrac{(\alpha_k)^4}{(\epsilon^{(H)})^5}\right) \\
&= \tilde{\mathcal{O}}\left(\min\left\{\tfrac{\beta^{2K}}{(\epsilon^{(g)})^3}, \tfrac{\beta^{1.5K}}{(\epsilon^{(g)})^{3.5}}, \tfrac{\beta^{2K}}{(\epsilon^{(g)})^4}\right\} + \tfrac{\beta^{4K}}{(\epsilon^{(H)})^5}\right) \\
&= \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}\right),
\end{aligned} \tag{3.28}$$

where the last equality is due to the fact that $\beta^K = \mathcal{O}((\epsilon^{(g)})^{-1})$. Combining (3.27) and (3.28), and substituting in (3.19) yields the result.

Case (ii): Under **Option II**, from Assumption 3.4(ii), we establish that the iterate x_k satisfies Condition (2.5) in Algorithm 1. Therefore, by Theorem 2.1, it holds that $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point as defined in Definition 1.2, since $p \leq p^{1/K}$.

At iteration $k = 0$, the work complexity of solver $\mathcal{M}$ is given by

$$\mathcal{W}_{\mathcal{M},0}^{(p)} = \mathcal{O}\left(\Delta_0^* \mathscr{C}_{\mathcal{M},0} \log^r\left(\tfrac{1}{1-p^{1/K}}\right)\right) = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-3} + (\epsilon^{(H)})^{-5}\right), \tag{3.29}$$

where Δ_0^* is defined in Lemma 3.6 and the final equality follows from Lemma A.2.

Let $D := \cap_{k=1}^{K} \{\Delta_k \leq \Delta^*\}$. From (3.12), we have

$$\mathbb{P}(D) \geq \prod_{k=0}^{K-1} p_k = \left(p^{1/K}\right)^K = p. \tag{3.30}$$

Now, define the event

$$E_{\mathcal{W}} := \left\{\sum_{k=1}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)} = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}\right)\right\}.$$

Suppose that event D holds. Then, by Assumption 3.4(ii), (3.25), and Lemma A.2, we obtain

$$\sum_{k=1}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)} = \sum_{k=1}^{K} \tilde{\mathcal{O}}\left(\Delta_k \mathscr{C}_{\mathcal{M},k} \log^r\left(\frac{1}{1-p^{1/K}}\right)\right) = \sum_{k=1}^{K} \tilde{\mathcal{O}}\left(\Delta^* \mathscr{C}_{\mathcal{M},k}\right) = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}\right),$$

where the final inequality follows from (3.28). Consequently, $D \subseteq E_{\mathcal{W}}$. Therefore, by (3.30),

$$\mathbb{P}\left(\sum_{k=1}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)} \leq \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}\right)\right) \geq \mathbb{P}(D) \geq p. \tag{3.31}$$

Since the bound in (3.29) holds almost surely, combining it with (3.31) and (3.19) yields

$$\mathbb{P}\left(\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})\right) \geq p. \tag{3.32}$$

□

4 Existing Primal Solvers

In this section, we consider existing solvers for solving the primal subproblem at each outer iteration $k \in \mathbb{N}_0$ of Algorithm 1 and producing iterates x_k that satisfy the inexactness conditions specified therein. For the general setting considered in this paper, we are unaware of stochastic second-order solvers with theoretical complexity guarantees for obtaining iterates that satisfy expectation conditions (2.4) or probabilistic conditions (2.5). However, theoretical complexity results are available for several stochastic second-order solvers under more specialized settings. We consider three such solvers: SG-HV-NC [6, Algorithm 4], Natasha2 [3], and SPIDER [24]. We describe their implementation within the MESCAL framework and establish the corresponding sample complexity results. For these three solvers, the existing theoretical results are established under two restrictions: (i) the subproblem is unconstrained, i.e., $\mathcal{X} = \mathbb{R}^d$, and (ii) the solver obtains an $(\epsilon_k^{(g)}, \epsilon_k^{(H)})$-accurate iterate satisfying the probabilistic conditions in (2.5). Therefore, our sample complexity results for these solvers establish the complexity of obtaining an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point $\tilde{x}$ satisfying (1.6) under the restriction $\mathcal{X} = \mathbb{R}^d$. However, we note that the theoretical analysis framework established in Section 2 and Section 3 ensures that any new second-order solvers, that can handle general closed and convex constraint sets $\mathcal{X}$ and obtain iterates satisfying (2.4), can be analyzed within the MESCAL framework to establish the corresponding sample complexity results.

The methods considered share a common framework in that they employ two distinct types of steps to update the primal iterate toward an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate point. The first is a gradient-based first-order step that leverages variance-reduction techniques to control the error in the gradient estimator. The second is a negative-curvature-based second-order step involving movement along directions of negative curvature, i.e., directions along which the Hessian exhibits sufficiently negative curvature. Although all three methods employ randomized negative-curvature steps, they differ in their negative-curvature detection procedures, the rule used to select between a first-order and a second-order step, and the variance-reduction technique used to construct the stochastic gradient estimators. In the following subsections, we describe these algorithms and establish the corresponding sample complexity results for their use within the MESCAL framework.

4.1 Primal Solver: SG-HV-NC

The SG-HV-NC (stochastic gradient method with variance reduction using Hessian vector products and negative curvature steps) [6, Algorithm 4] is a second-order solver that alternates between stochastic gradient (SG) steps and negative-curvature steps. We implement this solver as the primal solver $\mathcal{M}$ under **Option II** of Algorithm 1, with the step-by-step implementation provided in Algorithm 2. At each inner iteration $t \in \mathbb{N}$, a Bernoulli random variable determines whether an SG step or a negative-curvature step is performed. The SG steps use the variance-reduced gradient estimator HVP-RVR-GE (Hessian vector product-based recursive variance reduced gradient estimator), described in Algorithm 3, while the negative-curvature steps use Oja's algorithm, as described in [3], to efficiently identify a direction of negative curvature. For it's implementation in Algorithm 2 of Algorithm 2, we refer to [6, Lemma 16]. Further details and theoretical analysis of SG-HV-NC and Oja's algorithm can be found in [3, 4, 6, 59].

We next state the input-output guarantee of the Oja's algorithm that we use in our analysis. In particular, given a point x and a stochastic function class h, Oja either returns a certificate that the Hessian at x has no sufficiently negative curvature or returns a unit-norm direction of sufficiently negative curvature with some probability $1 - p_f$.

Definition 4.1. *The Oja's algorithm takes as input a point $x \in \mathbb{R}^d$; a stochastic function class h satisfying 1) a finite initial optimality gap, 2) Lipschitz continuous stochastic gradient, 3) Lipschitz continuous Hessian, 4) stochastic gradient with a uniform error bound in expectation, and 5) stochastic Hessian with a uniform error bound almost surely; a precision parameter $\epsilon^{(H)} > 0$; and a failure probability $p_f \in (0, 1)$. It outputs $u \in \mathbb{R}^d \cup \{\perp\}$ such that, with probability at least $1 - p_f$, either*

1. *$u = \perp$, and $\sigma_{\min}(\nabla^2 h(x)) \geq -2\epsilon^{(H)}$;*
2. *$u \neq \perp$, $\|u\| = 1$ and $u^\top \nabla^2 h(x) u \leq -\epsilon^{(H)}$.*

Algorithm 2: SG-HV-NC

Input (for all $k \in \mathbb{N}_0$): function class $\mathcal{L}_{\alpha_k}$; iterate x_{k-1}; dual variable λ_k; optimality gap Δ_k; smoothness constants $L_{\mathcal{L},k}^{(g)}$, $L_{\mathcal{L},k}^{(H)}$; variance constants $\sigma_{\mathcal{L},k}^{(g)}$, $\sigma_{\mathcal{L},k}^{(H)}$; primal subproblem tolerances $\epsilon_k^{(g)}$, $\epsilon_k^{(H)} > 0$

set $\eta_k = \min\left\{\frac{\epsilon_k^{(H)}}{L_{\mathcal{L},k}^{(H)}}, \frac{1}{2\sqrt{(L_{\mathcal{L},k}^{(g)})^2+(\sigma_{\mathcal{L},k}^{(H)})^2+\epsilon_k^{(g)}L_{\mathcal{L},k}^{(H)}}}\right\}$, $T_k = \left\lceil \frac{20\Delta_k(L_{\mathcal{L},k}^{(H)})^2}{(\epsilon_k^{(H)})^3} + \frac{2\Delta_k}{\eta_k(\epsilon_k^{(g)})^2}\right\rceil$, $q_k = \frac{(\epsilon_k^{(H)})^3}{(\epsilon_k^{(H)})^3+10\Delta_k(L_{\mathcal{L},k}^{(H)})^2\eta_k(\epsilon_k^{(g)})^2}$, $\delta_k = \frac{\epsilon_k^{(H)}}{40^2 L_{\mathcal{L},k}^{(H)}}$

set $b_k^{(g)} = \min\left\{1, \frac{\eta_k\sqrt{(\sigma_{\mathcal{L},k}^{(H)})^2+\epsilon_k^{(g)}L_{\mathcal{L},k}^{(H)}}}{\sigma_{\mathcal{L},k}^{(g)}}\right\}$ and $b_k^{(H)} = \min\left\{1, \frac{\epsilon_k^{(H)}\sqrt{(\sigma_{\mathcal{L},k}^{(H)})^2+\epsilon_k^{(g)}L_{\mathcal{L},k}^{(H)}}}{\sigma_{\mathcal{L},k}^{(g)}L_{\mathcal{L},k}^{(H)}}\right\}$

initialize $x_{k,0} = x_{k-1}$, $x_{k,1} = x_{k-1}$

initialize $g_{k,1} = \text{HVP-RVR-GE}(x_{k,1}, x_{k,0}, \emptyset, b_k^{(g)}, \epsilon_k^{(g)}, L_{\mathcal{L},k}^{(H)}, \sigma_{\mathcal{L},k}^{(g)}, \sigma_{\mathcal{L},k}^{(H)}, \lambda_k, \mathcal{L}_{\alpha_k})$

for $t = 1$ **to** T_k **do**

  sample $Q_{1,k,t} \sim \text{Bernoulli}(q_k)$

  if $Q_{1,k,t} = 1$ **then**

    set $x_{k,t+1} = x_{k,t} - \eta_k g_{k,t}$ `// first-order step`

    set $g_{k,t+1} = \text{HVP-RVR-GE}(x_{k,t+1}, x_{k,t}, g_{k,t}, b_k^{(g)}, \epsilon_k^{(g)}, L_{\mathcal{L},k}^{(H)}, \sigma_{\mathcal{L},k}^{(g)}, \sigma_{\mathcal{L},k}^{(H)}, \lambda_k, \mathcal{L}_{\alpha_k})$

  else

    set $u_{k,t} = \text{Oja}(x_{k,t}, \mathcal{L}_{\alpha_k}(\cdot, \lambda_k), 2\epsilon_k^{(H)}, \delta_k)$ `// negative curvature search`

    if $u_{k,t} = \perp$ **then**

      set $x_{k,t+1} = x_{k,t}$

      set $g_{k,t+1} = g_{k,t}$

    else

      sample $r_{k,t} \sim \text{Uniform}(\{-1, 1\})$

      set $x_{k,t+1} = x_{k,t} + \frac{\epsilon_k^{(H)} r_{k,t}}{L_{\mathcal{L},k}^{(H)}} u_{k,t}$ `// second-order step`

      set $g_{k,t+1} = \text{HVP-RVR-GE}(x_{k,t+1}, x_{k,t}, g_{k,t}, b_k^{(H)}, \epsilon_k^{(g)}, L_{\mathcal{L},k}^{(H)}, \sigma_{\mathcal{L},k}^{(g)}, \sigma_{\mathcal{L},k}^{(H)}, \lambda_k, \mathcal{L}_{\alpha_k})$

sample $x_k \sim \text{Uniform}(\{x_{k,t}\}_{t=1}^{T_k})$

return x_k

Algorithm 3: HVP-RVR-GE

Input: Iterates $\hat{x}_1, \hat{x}_0$; gradient estimators g_0; estimator selection probability b; error tolerance ϵ; smoothness constant $L^{(H)}$; variance constants $\sigma^{(g)}$, $\sigma^{(H)}$; dual variable λ; function class $\mathcal{L}_\alpha$

set $J = \left\lceil \frac{5((\sigma^{(H)})^2+L^{(H)}\epsilon)}{b\epsilon^2}\|\hat{x}_1 - \hat{x}_0\|^2\right\rceil$, $B = \left\lceil \frac{5(\sigma^{(g)})^2}{\epsilon^2}\right\rceil$

sample $Q_2 \sim \text{Bernoulli}(b)$

if $Q_2 = 1$ **or** $g_0 = \emptyset$ **then**

  sample $\theta_i \sim \mathcal{P}_\theta, \quad \forall i \in \{1, 2, \cdots, B\}$

  set $g_1 = \frac{\sum_{\theta_i=1}^B \nabla\bar{\mathcal{L}}_\alpha(\hat{x}_1, \lambda; \theta)}{B}$

else

  set $\overline{x}_j = \frac{j}{J}\hat{x}_1 + \left(1 - \frac{j}{J}\right)\hat{x}_0, \quad \forall j \in \{0, 1, \cdots, J\}$

  sample $\theta_j \sim \mathcal{P}_\theta, \quad \forall j \in \{0, 1, \cdots, J\}$

  set $g_1 = g_0 + \sum_{j=1}^J \nabla^2\bar{\mathcal{L}}_\alpha(\overline{x}_{j-1}, \lambda; \theta_j)(\overline{x}_j - \overline{x}_{j-1})$

return g_1

Remark 4.1. *Second-order stochastic optimization algorithms typically involve several parameters. For*

Algorithm 2, we briefly describe the roles of the main parameters:

- η_k*: The step size used for the first-order steps.*
- T_k*: The prescribed number of inner iterations performed by the SG-HV-NC solver.*
- q_k*: The probability parameter of the Bernoulli random variable* $Q_{1,k,t}$*, which determines whether a first-order step* ($Q_{1,k,t} = 1$) *or a negative-cuvature step* ($Q_{1,k,t} = 0$) *is performed at inner iteration* t*.*
- δ_k*: The failure probability parameter for Oja's algorithm which controls the probability with which the algorithm correctly identifies a direction of negative curvature or certifies the absence of sufficiently negative curvature.*
- $b_k^{(g)}$*: The probability parameter of Bernoulli random variable* Q_2 *in Algorithm 3 when HVP-RVR-GE is called during a first-order step. It determines whether a subsampled gradient estimator or a recursive variance-reduced gradient estimator is used.*
- $b_k^{(H)}$*: The probability parameter of Bernoulli random variable* Q_2 *in Algorithm 3 when HVP-RVR-GE is called during a second-order step. It determines whether a subsampled gradient estimator or a recursive variance-reduced gradient estimator is used.*

The values of these parameters are specified in Algorithm 2 to ensure that the resulting iterate x_k *is an* $(\epsilon_k^{(g)}, \epsilon_k^{(H)})$*-accurate iterate satisfying the probabilistic conditions in (2.5) at each outer iteration* $k \in \mathbb{N}$*.*

We now state the work complexity at each iteration $k \in \mathbb{N}_0$, as defined in Definition 3.2, of the SG-HV-NC algorithm, denoted by $\mathcal{W}_{SG,k}^{(p)}$, when used as the primal solver in Algorithm 1.

Lemma 4.1 (Work complexity of SG-HV-NC)**.** *Suppose* $\mathcal{X} = \mathbb{R}^d$ *and Assumptions 2.1, 2.2, 3.1, 3.2, and 3.3 hold. Let Algorithm 2 be the solver* $\mathcal{M}$ *under* ***Option II****, with terms* $L_{\mathcal{L},k}^{(g)}$, $L_{\mathcal{L},k}^{(H)}$, $\sigma_{\mathcal{L},k}^{(g)}$, $\sigma_{\mathcal{L},k}^{(H)}$, *and* Δ_k *defined as per Definition 3.1. If the primal subproblem tolerance inputs satisfy* $\epsilon_k^{(g)} \leq \min\{1, \sigma_{\mathcal{L},k}^{(g)}, (\Delta_k L_{\mathcal{L},k}^{(g)})^{0.5}\}$*, and* $\epsilon_k^{(H)} \leq \min\{1, \sigma_{\mathcal{L},k}^{(H)}, L_{\mathcal{L},k}^{(g)}, (\epsilon_k^{(g)} L_{\mathcal{L},k}^{(H)})^{0.5}\}$*, then, at every iteration* $k \in \mathbb{N}_0$*, Algorithm 2 returns an iterate* x_k *satisfying the probabilistic conditions (2.5) with probability at least* $p_k = 5/8$*, with work complexity*

$$\mathcal{W}_{SG,k}^{(p)} = \tilde{\mathcal{O}}\left(\frac{\Delta_k \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(g)})^3} + \frac{\Delta_k L_{\mathcal{L},k}^{(H)} \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(H)})^2 (\epsilon_k^{(g)})^2} + \frac{\Delta_k (L_{\mathcal{L},k}^{(H)})^2 (\sigma_{\mathcal{L},k}^{(H)} + L_{\mathcal{L},k}^{(g)})^2}{(\epsilon_k^{(H)})^5} + \frac{\Delta_k L_{\mathcal{L},k}^{(g)}}{(\epsilon_k^{(g)})^2} \right).$$

The proof of this lemma is provided in Section A.2. We now provide the sample complexity of the MESCAL framework with SG-HV-NC as the primal solver $\mathcal{M}$.

Theorem 4.2 (Sample Complexity of MESCAL-SG-HV-NC)**.** *Suppose* $\mathcal{X} = \mathbb{R}^d$*, and Assumptions 2.1, 2.2, 3.1, 3.2 and 3.3 hold. Let* $\{x_k\}$ *be the sequence of iterates generated by Algorithm 1 with initial iterate* $x_{-1} \in \mathcal{X}$ *and initial dual iterate* $\lambda_0 \in \mathbb{R}^m$*. Suppose that the dual bound parameters* $\gamma_k > 0$ *be a bounded sequence satisfying* $\sum_{k=0}^{\infty} \gamma_k = C_\gamma < \infty$*, and that the dual sample sizes satisfy* $|S_k^{(d)}| = \Theta((\epsilon^{(g)})^{-2})$ *for all* $k \in \mathbb{N}_0$*. Let Algorithm 2 be the solver* $\mathcal{M}$ *under* ***Option II****. Suppose the primal subproblem tolerances* $\epsilon_k^{(g)}$*,* $\epsilon_k^{(H)}$ *satisfy* $\epsilon_k^{(g)} = \epsilon^{(g)} \leq \min\{1, \sigma_{\mathcal{L},k}^{(g)}, (\Delta_k L_{\mathcal{L},k}^{(g)})^{0.5}\}$ *and* $\epsilon_k^{(H)} = \epsilon^{(H)} \leq \min\{1, \sigma_{\mathcal{L},k}^{(H)}, L_{\mathcal{L},k}^{(g)}, (\epsilon_k^{(g)} L_{\mathcal{L},k}^{(H)})^{0.5}\}$ *for all* $k \in \mathbb{N}_0$*. Then, for any* $k = K \geq K^*$*, where* K^* *is defined in (2.7), the iterate* $\tilde{x} := x_K$ *is an* $(\epsilon^{(g)}, \epsilon^{(H)})$*-accurate p-probabilistic second-order stationary point satisfying (1.6) as per Definition 1.2, with* $p = 5/8$ *and* $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$*. Furthermore, the sample complexity of the algorithm, as defined in Definition 1.3, to obtain* $\tilde{x} := x_K$*, is*

(i) ***(Deterministic Sample Complexity):***

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-6} (\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-5} (\epsilon^{(H)})^{-5}).$$

(ii) ***(Probabilistic Sample Complexity):***

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})$$

with probability at least p^K.

Proof. From Lemma 4.1, we establish that the iterate x_k satisfies (2.5) in Algorithm 1. Therefore, by Theorem 2.1, it holds that $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point as defined in Definition 1.2, with $p = 5/8$ and $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$.

At iteration $k = 0$, using Lemma 4.1, (3.24), (3.9) in Lemma 3.6, and setting $\epsilon_0^{(g)} = \epsilon^{(g)}$, $\epsilon_0^{(H)} = \epsilon^{(H)}$, the work complexity of solver $\mathcal{M}$ is given by

$$\begin{aligned}\mathcal{W}_{SG,0}^{(p)} &= \tilde{\mathcal{O}}\left(\frac{\Delta_0^* \sigma_{\mathcal{L},0}^{(g)} \sigma_{\mathcal{L},0}^{(H)}}{(\epsilon^{(g)})^3} + \frac{\Delta_0^* L_{\mathcal{L},0}^{(H)} \sigma_{\mathcal{L},0}^{(g)} \sigma_{\mathcal{L},0}^{(H)}}{(\epsilon^{(H)})^2(\epsilon^{(g)})^2} + \frac{\Delta_0^* (L_{\mathcal{L},0}^{(H)})^2 (\sigma_{\mathcal{L},0}^{(H)} + L_{\mathcal{L},0}^{(g)})^2}{(\epsilon^{(H)})^5} + \frac{\Delta_0^* L_{\mathcal{L},0}^{(g)}}{(\epsilon^{(g)})^2}\right) \\ &= \tilde{\mathcal{O}}((\epsilon^{(g)})^{-3} + (\epsilon^{(g)})^{-2}(\epsilon^{(H)})^{-2} + (\epsilon^{(H)})^{-5}). \end{aligned} \tag{4.1}$$

Case (i): Using Lemma 4.1, (3.24), (3.9) in Lemma 3.6, and setting $\epsilon_k^{(g)} = \epsilon^{(g)}$, $\epsilon_k^{(H)} = \epsilon^{(H)}$, the total work complexity of solver $\mathcal{M}$ over iterations $k = 1, 2, \cdots, K$ is given by

$$\begin{aligned}\sum_{k=1}^{K} \mathcal{W}_{SG,k}^{(p)} &= \sum_{k=1}^{K} \tilde{\mathcal{O}}\left(\frac{\Delta_k \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(g)})^3} + \frac{\Delta_k L_{\mathcal{L},k}^{(H)} \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(H)})^2(\epsilon_k^{(g)})^2} + \frac{\Delta_k (L_{\mathcal{L},k}^{(H)})^2 (\sigma_{\mathcal{L},k}^{(H)} + L_{\mathcal{L},k}^{(g)})^2}{(\epsilon_k^{(H)})^5} + \frac{\Delta_k L_{\mathcal{L},k}^{(g)}}{(\epsilon_k^{(g)})^2}\right) \\ &= \sum_{k=1}^{K} \tilde{\mathcal{O}}\left(\frac{\beta^{3k}}{(\epsilon^{(g)})^3} + \frac{\beta^{4k}}{(\epsilon^{(H)})^2(\epsilon^{(g)})^2} + \frac{\beta^{5k}}{(\epsilon^{(H)})^5} + \frac{\beta^{2k}}{(\epsilon^{(g)})^2}\right) \\ &= \tilde{\mathcal{O}}\left(\frac{\beta^{3K}}{(\epsilon^{(g)})^3} + \frac{\beta^{4K}}{(\epsilon^{(H)})^2(\epsilon^{(g)})^2} + \frac{\beta^{5K}}{(\epsilon^{(H)})^5} + \frac{\beta^{2K}}{(\epsilon^{(g)})^2}\right) \\ &= \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}\right), \end{aligned} \tag{4.2}$$

where the last equality is due to the fact that $\beta^K = \mathcal{O}((\epsilon^{(g)})^{-1})$. Combining (4.1), (4.2), and (3.19) yields the result.

Case (ii): Let $D := \cap_{k=1}^{K} \{\Delta_k \leq \Delta^*\}$. From (3.12), we have $\mathbb{P}(D) \geq \prod_{k=0}^{K-1} p_k = p^K$. Suppose that event D holds, following the same approach as in the proof of Case (ii) in Theorem 3.8, we have

$$\sum_{k=1}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)} = \tilde{\mathcal{O}}\left(\frac{\beta^{2K}}{(\epsilon^{(g)})^3} + \frac{\beta^{3K}}{(\epsilon^{(H)})^2(\epsilon^{(g)})^2} + \frac{\beta^{4K}}{(\epsilon^{(H)})^5} + \frac{\beta^{K}}{(\epsilon^{(g)})^2}\right) = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5}\right).$$

Therefore,

$$\mathbb{P}\left(\sum_{k=1}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})\right) \geq p^K. \tag{4.3}$$

Since the bound in (4.1) holds almost surely, combining it with (4.3) and (3.19) yields the result. □

Remark 4.2. *The deterministic sample complexity result in Theorem 4.2, which uses the deterministic bound on* Δ_k *in (3.9), is worse by a factor of* $(\epsilon^{(g)})^{-1}$ *than the probabilistic bound, which instead uses the constant upper bound on* Δ_k *that holds probabilistically given in (3.11). A limitation of the latter result is that the available theoretical analysis of SG-HV-NC guarantees success only with the fixed probability* $p_k = 5/8$ *at each outer iteration* k*. This differs from Theorem 3.8, where the per-iteration success probabilities can be selected as* $p_k = p^{1/K}$ *to obtain an overall success probability of at least* p*. An extension of the SG-HV-NC theoretical analysis to an arbitrary prescribed success probability could therefore yield a higher-probability counterpart of the present result. Finally, in the nonconvex setting, ignoring the* $\epsilon^{(H)}$ *factors, the resulting probabilistic sample complexity matches the best-known first-order complexity bounds for stochastic augmented-Lagrangian methods; see, for example, [39].*

4.2 Primal Solver: Natasha2

Natasha2 [3] is a second-order method that alternates between negative-curvature steps and proximal variance-reduced gradient steps to obtain second-order stationary points of nonconvex unconstrained optimization problems. The step-by-step implementation of this algorithm as solver $\mathcal{M}$ under **Option II** in Algorithm 1, is provided in Algorithm 4. This is primarily a stochastic solver, i.e., it typically employs single-sample realizations of θ rather than a mini-batch within the variance-reduced gradient estimator step (see Line 11 in Algorithm 5). Natasha2 utilizes two algorithmic subroutines. The first is Oja's algorithm, for which a brief description is provided in Section 4.1. The second is the Natasha1.5 subroutine, described in Algorithm 5, which is intended to obtain an ϵ-accurate first-order stationary point. Once these standard iterate updates are completed, Natasha2 utilizes an SGD variant [2], denoted SGD3SC, on a strongly convex approximation of the objective function to update the iterates. This additional step is necessary for the theoretical convergence but is not important in practice; its usage is established in [3, Theorem 5.7].

We next state the input-output guarantee of SGD3SC that we use in our analysis. Specifically, given a point x and a stochastic function class h, SGD3SC generates an iterate with a bounded gradient norm.

Definition 4.2. *The SGD3SC algorithm takes as input a stochastic function class h satisfying 1) strong convexity, 2) finite initial optimality gap, 3) Lipschitz continuous stochastic gradient, 4) Lipschitz continuous Hessian, 5) stochastic gradient with a uniform error bound in expectation. After T iterations, the algorithm outputs $\bar{x} \in \mathbb{R}^d$ such that $\mathbb{E}[\|\nabla h(\bar{x})\|] = \mathcal{O}(1/\sqrt{T})$.*

Algorithm 4: Natasha2

Input (for all $k \in \mathbb{N}_0$): function class $\mathcal{L}_{\alpha_k}$; iterate x_{k-1}; dual variable λ_k; optimality gap Δ_k; smoothness constants $L^{(g)}_{\mathcal{L},k}$, $L^{(H)}_{\mathcal{L},k}$; variance constant $\sigma^{(g)}_{\mathcal{L},k}$; primal subproblem tolerances $\epsilon^{(g)}_k$, $\epsilon^{(H)}_k > 0$; solver-specific positive constants c_L, c_ι, c_B, c_q, c_η, c_N, $c_{\tilde{L}}$, c_T

initialize $x_{k,0} = x_{k-1}$

if $L^{(H)}_{\mathcal{L},k} \geq \frac{L^{(g)}_{\mathcal{L},k}\epsilon^{(H)}_k}{3(\sigma^{(g)}_{\mathcal{L},k})^{2/3}(\epsilon^{(g)}_k)^{1/3}}$ **then**

  set $\tilde{L}_k = \max\left(L^{(g)}_{\mathcal{L},k}, \frac{c_L L^{(H)}_{\mathcal{L},k}\sqrt[3]{(\sigma^{(g)}_{\mathcal{L},k})^2\epsilon^{(g)}_k}}{\epsilon^{(H)}_k}\right)$, $\iota_k = \tilde{L}_k$

else

  set $\tilde{L}_k = L^{(g)}_{\mathcal{L},k}$, $\iota_k = \min\left(L^{(g)}_{\mathcal{L},k}, \max\left\{\frac{\epsilon^{(H)}_k}{3}, \frac{c_\iota(\sigma^{(g)}_{\mathcal{L},k})^2(\epsilon^{(g)}_k)(L^{(H)}_{\mathcal{L},k})^3}{(L^{(g)}_{\mathcal{L},k})^2(\epsilon^{(H)}_k)^3}, \frac{c_\iota(\epsilon^{(g)}_k)L^{(g)}_{\mathcal{L},k}}{\sigma^{(g)}_{\mathcal{L},k}}\right\}\right)$

set $B_k = \left\lceil \frac{c_B(\sigma^{(g)}_{\mathcal{L},k})^2}{(\epsilon^{(g)}_k)^2} \right\rceil$, $q_k = \left\lceil c_q\left(\frac{B_k\iota_k^2}{\tilde{L}_k^2}\right)^{1/3} \right\rceil$, $\eta_k = \frac{c_\eta\iota_k}{q_k^2\tilde{L}_k^2}$, $N_{k,1} = \left\lceil \frac{c_N\iota_k\Delta_k}{q_k(\epsilon^{(g)}_k)^2} \right\rceil$, $t_N = 0$

for $t = 0$ **to** ∞ **do**

  set $v = \mathrm{Oja}\left(x_{k,t}, \mathcal{L}_{\alpha_k}(\cdot, \lambda_k), \frac{\epsilon^{(H)}_k}{3}, \frac{1}{12}\right)$ `// negative curvature search`

  if $v \neq \perp$ **then**

    sample $r_{k,t} \sim \mathrm{Uniform}(\{-1, 1\})$

    set $x_{k,t+1} = x_{k,t} + \frac{r_{k,t}\epsilon^{(H)}_k}{3L^{(H)}_{\mathcal{L},k}}v$ `// second-order step`

  else

    update $t_N \leftarrow t_N + 1$

    let $V_{k,t}(y) := \mathcal{L}_{\alpha_k}(y, \lambda_k) + L^{(g)}_{\mathcal{L},k}\Big(\max\left\{0, \|y - x_{k,t}\| - \frac{\epsilon^{(H)}_k}{3L^{(H)}_{\mathcal{L},k}}\right\}\Big)^2$

    set $(\hat{x}_{k,t}, x_{k,t+1}) = \mathrm{Natasha1.5}(V_{k,t}, x_{k,t}, B_k, q_k, 1, \eta_k, L^{(g)}_{\mathcal{L},k})$ `// first-order step`

    store the vector pair $(x_{k,t}, \hat{x}_{k,t})$

    go to Algorithm 4 if $t_N = N_{k,1}$

sample $(x, \hat{x}) \sim \mathrm{Uniform}(\{(x_{k,t}, \hat{x}_{k,t})\}_{t=0}^{N_{k,1}})$

let $W(y) := \mathcal{L}_{\alpha_k}(y, \lambda_k) + L^{(g)}_{\mathcal{L},k}\Big(\max\left\{0, \|y - x\| - \frac{\epsilon^{(H)}_k}{3L^{(H)}_{\mathcal{L},k}}\right\}\Big)^2 + \iota_k\|y - \hat{x}\|^2$

set $y^{\mathrm{out}} = \mathrm{SGD3SC}\left(W, \hat{x}, \iota_k, c_{\tilde{L}}\tilde{L}_k, \frac{c_T(\sigma^{(g)}_{\mathcal{L},k})^2}{(\epsilon^{(g)}_k)^2}\log^3\left(\frac{\tilde{L}_k}{\iota_k}\right)\right)$

return y^{out}

Algorithm 5: Natasha1.5

Input: function $V(\cdot)$, starting vector $y^{\emptyset}$, epoch length B, sub-epoch count q, epoch count $T' \geq 1$, learning rate $\eta > 0$, Lipschitz constant $L^{(g)}$

initialize $\hat{y}_{1,1} = y^{\emptyset}$
set $B' = \lceil B/q \rceil$
for $t' = 1$ **to** T' **do**
  sample $\theta_j \sim \mathcal{P}_\theta$ for $j \in \{0, 1, \cdots, B\}$
  set $\tilde{y} = \hat{y}_{(t',1)}$, $\mu = \frac{1}{B}\sum_{\theta_j=0}^{B} \nabla V(\tilde{y}, \theta)$
  for $t_q = 1$ **to** q **do**
    initialize $y_0 = \hat{y}_{(t',t_q)}$
    store the vector $\hat{y}_{(t',t_q)}$
    for $b' = 0$ *to* $B' - 1$ **do**
      sample $\theta \sim \mathcal{P}_\theta$
      set $g = \nabla V(y_{b'}, \theta) - \nabla V(\tilde{y}, \theta) + \mu + 2L^{(g)}(y_{b'} - \hat{y}_{(t',t_q)})$
      $y_{b'+1} = y_{b'} - \eta g$
    sample $\hat{y}_{(t',t_q+1)} \sim \text{Uniform}(\{y_0, y_1, \cdots, y_{B'}\})$
sample $\hat{x} \sim \text{Uniform}(\{\hat{y}_{(t',t_q)} \mid 1 \leq t' \leq T', 1 \leq t_q \leq q\})$; set $x^+ = \hat{y}_{(T',q)}$
return $(\hat{x}, x^+)$

Remark 4.3. *We briefly describe the roles of the main parameters in Algorithm 4:*

- B_k*: The epoch mini-batch length for the Natasha1.5 subroutine.*
- q_k*: The sub-epoch count in the Natasha1.5 subroutine.*
- η_k*: The step size used to update the iterate in the Natasha1.5 subroutine.*
- ι_k*: The strong convexity parameter utilized for the SGD3SC subroutine.*
- $N_{k,1}$ *: The number of first-order steps.*
- $V_{k,t}$*: The approximation of the augmented Lagrangian function.*

These parameters are specified in Algorithm 4 to ensure that the resulting iterate x_k is an $(\epsilon_k^{(g)}, \epsilon_k^{(H)})$-accurate iterate satisfying the probabilistic conditions in (2.5) at each outer iteration $k \in \mathbb{N}$.

We now state the work complexity at each iteration $k \in \mathbb{N}_0$, as defined in Definition 3.2, of the Natasha2 algorithm, denoted by $\mathcal{W}_{Nat2,k}^{(p)}$, when used as the primal solver in Algorithm 1.

Lemma 4.3 (Work complexity of Natasha2)**.** *Suppose $\mathcal{X} = \mathbb{R}^d$, and Assumptions 2.1, 2.2, 3.1, 3.2 and 3.3 hold. Let Algorithm 4 be the solver $\mathcal{M}$ under* ***Option II****, with terms $L_{\mathcal{L},k}^{(g)}$, $L_{\mathcal{L},k}^{(H)}$, $\sigma_{\mathcal{L},k}^{(g)}$, and Δ_k defined as per Definition 3.1. If the primal subproblem tolerance inputs satisfy $\epsilon_k^{(g)} = \mathcal{O}(\min\{1, \sigma_{\mathcal{L},k}^{(g)}\})$, and $\epsilon_k^{(H)} = \mathcal{O}\Big(\min\Big\{1, 3L_{\mathcal{L},k}^{(g)}, \Big(\frac{(\sigma_{\mathcal{L},k}^{(g)})^2 \epsilon_k^{(g)} (L_{\mathcal{L},k}^{(H)})^3}{L_{\mathcal{L},k}^{(g)}}\Big)^{0.25}\Big\}\Big)$, then at every iteration $k \in \mathbb{N}_0$, Algorithm 4 returns an iterate x_k satisfying the probabilistic conditions (2.5) with probability at least $p_k = 2/3$, with work complexity*

$$\mathcal{W}_{Nat2,k}^{(p)} = \tilde{\mathcal{O}}\left(\frac{(\sigma_{\mathcal{L},k}^{(g)})^4}{(\epsilon_k^{(g)})^2} + \frac{\Delta_k (L_{\mathcal{L},k}^{(H)})^2 (L_{\mathcal{L},k}^{(g)})^2}{(\epsilon_k^{(H)})^5} + \frac{\Delta_k L_{\mathcal{L},k}^{(H)} (L_{\mathcal{L},k}^{(g)})^2}{\epsilon_k^{(g)} (\epsilon_k^{(H)})^3} + \frac{\Delta_k L_{\mathcal{L},k}^{(H)} (\sigma_{\mathcal{L},k}^{(g)})^2}{(\epsilon_k^{(g)})^3 \epsilon_k^{(H)}} + \frac{\Delta_k (L_{\mathcal{L},k}^{(g)})^3}{\epsilon_k^{(g)} (\epsilon_k^{(H)})^2 \sigma_{\mathcal{L},k}^{(g)}}\right).$$

The proof of this lemma is provided in Section A.3. We now establish the sample complexity of MESCAL-Natasha2, i.e., the MESCAL framework utilizing Natasha2 as the solver $\mathcal{M}$.

Theorem 4.4 (Sample Complexity of MESCAL-Natasha2). *Suppose $\mathcal{X} = \mathbb{R}^d$, and Assumptions 2.1, 2.2, 3.1, 3.2 and 3.3 hold. Let $\{x_k\}$ be the sequence of iterates generated by Algorithm 1 with initial iterate $x_{-1} \in \mathcal{X}$ and initial dual iterate $\lambda_0 \in \mathbb{R}^m$. Suppose that the dual bound parameters $\gamma_k > 0$ be a bounded sequence satisfying $\sum_{k=0}^{\infty} \gamma_k = C_\gamma < \infty$, and that the dual sample sizes satisfy $|S_k^{(d)}| = \Theta((\epsilon^{(g)})^{-2})$ for all $k \in \mathbb{N}_0$. Let Algorithm 4 be the solver $\mathcal{M}$ under **Option II**. Suppose the primal subproblem tolerances $\epsilon_k^{(g)}$, $\epsilon_k^{(H)}$ satisfy $\epsilon_k^{(g)} = \epsilon^{(g)} = \mathcal{O}(\min\{1, \sigma_{\mathcal{L},k}^{(g)}\})$ and $\epsilon_k^{(H)} = \epsilon^{(H)} = \mathcal{O}\Big(\min\Big\{1, 3L_{\mathcal{L},k}^{(g)}, \Big(\frac{(\sigma_{\mathcal{L},k}^{(g)})^2 \epsilon_k^{(g)} (L_{\mathcal{L},k}^{(H)})^3}{L_{\mathcal{L},k}^{(g)}}\Big)^{0.25}\Big\}\Big)$ for all $k \in \mathbb{N}_0$. Then, for any $k = K \geq K^*$, where K^* is defined in (2.7), the iterate $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point satisfying (1.6) as per Definition 1.2 with $p = 2/3$ and $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$. Furthermore, the sample complexity of the algorithm as defined in Definition 1.3, to obtain $\tilde{x} := x_K$, is*

(i) ***(Deterministic Sample Complexity):***

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-7}(\epsilon^{(H)})^{-1} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}).$$

(ii) ***(Probabilistic Sample Complexity):***

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-1} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})$$

with probability at least p^K.

Proof. From Lemma 4.3, we establish that the iterate x_k satisfies (2.5) in Algorithm 1. Therefore, by Theorem 2.1, it holds that the iterate $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point as defined in Definition 1.2, with $p = 2/3$ and where $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$.

Following the same approach as in (3.29) and (4.1), we have

$$\mathcal{W}_{Nat2,0}^{(p)} = \tilde{\mathcal{O}}\left((\epsilon^{(H)})^{-5} + (\epsilon^{(g)})^{-1}(\epsilon^{(H)})^{-3} + (\epsilon^{(g)})^{-3}(\epsilon^{(H)})^{-1}\right). \tag{4.4}$$

Case (i): By the same reasoning as in (4.2), we have

$$\sum_{k=1}^{K} \mathcal{W}_{Nat2,k}^{(p)} = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-7}(\epsilon^{(H)})^{-1} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}\right). \tag{4.5}$$

Substituting (4.4) and (4.5) in (3.19) completes the proof.

Case (ii): For probabilistic bounds, by following the same approach as that for (4.3), we have

$$\mathbb{P}\left(\sum_{k=1}^{K} \mathcal{W}_{Nat2,k}^{(p)} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-1} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})\right) \geq p^K. \tag{4.6}$$

We use (4.4) and (4.6) to complete the proof, as in (3.32).

□

Remark 4.4. *The deterministic sample complexity results for MESCAL-Natasha2 and MESCAL-SG-HV-NC differ in one term. In particular, the bound for MESCAL-Natasha2 contains the term $(\epsilon^{(g)})^{-7}(\epsilon^{(H)})^{-1}$, whereas the corresponding term for MESCAL-SG-HV-NC is $(\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-2}$. The same distinction arises in the p^K-probabilistic sample complexity results. The bound for MESCAL-Natasha2 contains $(\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-1}$, while that for MESCAL-SG-HV-NC contains $(\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-2}$.*

4.3 Primal Solver: SPIDER

SPIDER–SFO+ (stochastic path-integrated differential estimator–stochastic first-order+) [24, Algorithm 2], concisely denoted as SPIDER, is another second-order algorithm. The step-by-step implementation of this algorithm as the primal solver $\mathcal{M}$ under **Option II** in Algorithm 1, is provided in Algorithm 6. It combines SARAH-based [48] stochastic gradient tracking with a negative-curvature search procedure to efficiently escape saddle points. Similar to Natasha2, SPIDER first identifies directions of negative curvature using the Neon2 algorithm, a variant of Oja's algorithm. When a direction of negative curvature is detected, rather than taking a single large step, SPIDER performs multiple mini-steps along that direction, allowing it to continuously update the gradient estimator without recomputing a highly accurate gradient. If no negative curvature is detected, it similarly performs multiple first-order updates while continuously updating the gradient estimator. The algorithm terminates when the norm of the gradient estimator falls below the prescribed threshold.

We next state the input-output guarantee of the Neon2 algorithm, which is similar to that of Oja's algorithm. For information on the Neon2 procedure, as utilized in Algorithm 6 of Algorithm 6, refer to [24, Theorem 9]. A more detailed analysis of Neon2 is provided in [5].

Definition 4.3. *The Neon2 algorithm takes as input a point $x \in \mathbb{R}^d$, stochastic function class h that has 1) a finite initial optimality gap, 2) Lipschitz continuous stochastic gradient, 3) Lipschitz continuous stochastic Hessian, 4) stochastic gradient with uniform error bound almost surely, a precision parameter $\epsilon^{(H)} > 0$ and a failure probability $p_f \in (0,1)$, and outputs $w \in \mathbb{R}^d \cup \{\perp\}$ such that with probability at least $1 - p_f$, either*

1. $w = \perp$, $\sigma_{\min}(\nabla^2 h(x)) \geq -\epsilon^{(H)}$.
2. $w \neq \perp$, $\|w\| = 1$, *and* $w^\top \nabla^2 h(x) w \leq \frac{\epsilon^{(H)}}{2}$.

Algorithm 6: SPIDER

Input (for all $k \in \mathbb{N}_0$): function class $\mathcal{L}_{\alpha_k}$; initial iterate x_{k-1}; primal subproblem error tolerances $\bar{\epsilon}_k^{(g)}, \epsilon_k^{(H)} > 0$; optimality gap Δ_k; smoothness constants $L_{\mathcal{L},k}^{(g)}$, $L_{\mathcal{L},k}^{(H)}$; variance constant $\sigma_{\mathcal{L},k}^{(g)}$; primal parameter $\bar{\epsilon}_k^{(g)}$, primal subproblem tolerance $\epsilon_k^{(H)} > 0$; solver-specific constant c_{n_0}

set $c_{n_0} \in [1, 20\sigma_{\mathcal{L},k}^{(g)}/\bar{\epsilon}_k^{(g)}]$, $B_k^{(p)} = \frac{20\sigma_{\mathcal{L},k}^{(g)}}{\bar{\epsilon}_k^{(g)} c_{n_0}}$, $B_k^{\text{reset}} = \frac{200(\sigma_{\mathcal{L},k}^{(g)})^2}{(\bar{\epsilon}_k^{(g)})^2}$, $q_k = \frac{10\sigma_{\mathcal{L},k}^{(g)} c_{n_0}}{\bar{\epsilon}_k^{(g)}}$,
$\eta_k = \min\left(\frac{\bar{\epsilon}_k^{(g)}}{10L_{\mathcal{L},k}^{(g)} c_{n_0}\|g_t\|}, \frac{1}{L_{\mathcal{L},k}^{(g)} c_{n_0}}\right)$, $J_k = 4\left\lfloor \max\left(\frac{81L_{\mathcal{L},k}^{(H)}\Delta_k}{(\epsilon_k^{(H)})^3}, \frac{120\Delta_k L_{\mathcal{L},k}^{(H)}}{\epsilon_k^{(H)}\bar{\epsilon}_k^{(g)}}\right)\right\rfloor$, $\mathcal{T}_k = \left\lceil \frac{10\epsilon_k^{(H)} L_{\mathcal{L},k}^{(g)} c_{n_0}}{3L_{\mathcal{L},k}^{(H)}\bar{\epsilon}_k^{(g)}} \right\rceil$

set primal subproblem tolerance $\epsilon_k^{(g)} = \bar{\epsilon}_k^{(g)} \log(64(\mathcal{T}_k J_k + 1))$

set $t = 0$; initialize $x_{k,0} = x_{k-1}$

for $j = 0$ **to** J_k **do**

  set $w_1 = \text{Neon2}\left(\mathcal{L}_{\alpha_k}(\cdot, \lambda_k), x_{k,t}, 2\epsilon_k^{(H)}, \frac{1}{16J_k}\right)$ `// negative curvature search`

  if $w_1 \neq \perp$ **then**

    sample $b_j \sim \text{Uniform}(\{-1, 1\})$

  while $t < (j+1)\mathcal{T}_k$ **do**

    if $t \bmod q_k = 0$ **then**

      sample $\theta_i \sim \mathcal{P}_\theta, \quad \forall i \in \{1, 2, \cdots, B_k^{\text{reset}}\}$

      set $g_t = \frac{\sum_{\theta_i} \nabla\bar{\mathcal{L}}_{\alpha_k}(x_{k,t}, \lambda_k; \theta_i)}{B_k^{\text{reset}}}$

    else

      sample $\theta_i \sim \mathcal{P}_\theta, \quad \forall i \in \{1, 2, \cdots, B_k^{(p)}\}$

      set $g_t = \frac{\sum_{\theta_i} \nabla\bar{\mathcal{L}}_{\alpha_k}(x_{k,t}, \lambda_k; \theta_i)}{B_k^{(p)}} - \frac{\sum_{\theta_i} \nabla\bar{\mathcal{L}}_{\alpha_k}(x_{k,t-1}, \lambda_k; \theta_i)}{B_k^{(p)}} + g_{t-1}$

    if $w_1 \neq \perp$ **then**

      set $x_{k,t+1} = x_{k,t} - b_j \eta_k w_1$ `// second-order step`

    else

      if $\|g_t\| \leq 2\epsilon_k^{(g)}$ **then**

        return $x_{k,t}$

      set $x_{k,t+1} = x_{k,t} - \eta_k(g_t/\|g_t\|)$ `// first-order step`

    update $t \leftarrow t + 1$

return $x_{k,t}$

Remark 4.5. *We briefly describe the roles of the main parameters in Algorithm 6:*

- $B_k^{(p)}$*: The smaller sample size employed in gradient estimator.*
- B_k^{reset}*: The larger sample size employed in computing occasional highly accurate gradient approximation used in gradient estimator.*
- q_k*: The number of iterations before the highly accurate gradient computation is performed.*
- η_k*: The step size for both first-order and second-order steps.*
- J_k*: The total number of times the negative curvature search step is performed.*
- $\mathcal{T}_k$*: The number of first-order or second-order steps performed in between successive negative curvature search steps.*
- $\bar{\epsilon}_k^{(g)}$*: The primal parameter on which the primal subproblem tolerance $\epsilon^{(g)}$ is dependent on, with $\epsilon_k^{(g)} = \tilde{\mathcal{O}}(\bar{\epsilon}_k^{(g)})$.*

Lemma 4.5 (Work complexity of SPIDER). *Suppose $\mathcal{X} = \mathbb{R}^d$, and Assumptions 2.1, 2.2, 3.1, 3.2 and 3.3 hold. Let Algorithm 6 be the solver $\mathcal{M}$ under **Option II**, with terms $L_{\mathcal{L},k}^{(g)}$, $L_{\mathcal{L},k}^{(H)}$, $\sigma_{\mathcal{L},k}^{(g)}$, and Δ_k defined as per Definition 3.1. Suppose for $\bar{\epsilon}_k^{(g)} \leq 1$, $\epsilon_k^{(H)} \leq 1$, for every iteration $k \in \mathbb{N}_0$ such that primal subproblem tolerances $\epsilon_k^{(g)} = \bar{\epsilon}_k^{(g)} \log(64(\mathcal{T}_k J_k + 1)) \leq 1$ and $\epsilon_k^{(H)} \leq 1$, where $\mathcal{T}_k$, J_k are defined in Algorithm 6. Then, Algorithm 6 returns an iterate x_k satisfying the probabilistic conditions (2.5) with probability at least $p_k = 1/2$, with work complexity*

$$\mathcal{W}_{SPDR,k}^{(p)} = \tilde{\mathcal{O}}\left(\frac{\Delta_k L_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(g)}}{(\bar{\epsilon}_k^{(g)})^3} + \frac{\Delta_k L_{\mathcal{L},k}^{(g)} L_{\mathcal{L},k}^{(H)} \sigma_{\mathcal{L},k}^{(g)}}{(\bar{\epsilon}_k^{(g)})^2 (\epsilon_k^{(H)})^2} + \frac{\Delta_k (L_{\mathcal{L},k}^{(g)})^2 (L_{\mathcal{L},k}^{(H)})^2}{(\epsilon_k^{(H)})^5} + \frac{\Delta_k (L_{\mathcal{L},k}^{(g)})^2 L_{\mathcal{L},k}^{(H)}}{\bar{\epsilon}_k^{(g)} (\epsilon_k^{(H)})^3} + \frac{(\sigma_{\mathcal{L},k}^{(g)})^2}{(\bar{\epsilon}_k^{g})^2} + \frac{L_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(g)} \epsilon_k^{(H)}}{(\bar{\epsilon}_k^{g})^2 L_{\mathcal{L},k}^{(H)}} \right).$$

The proof of this lemma is provided in Section A.4. We now establish the sample complexity of MESCAL-SPIDER, i.e., the MESCAL framework utilizing SPIDER as the solver $\mathcal{M}$.

Theorem 4.6 (Sample Complexity of MESCAL-SPIDER). *Suppose $\mathcal{X} = \mathbb{R}^d$, and Assumptions 2.1, 2.2, 3.1, 3.2 and 3.3 hold. Let $\{x_k\}$ be the sequence of iterates generated by Algorithm 1 with initial iterate $x_{-1} \in \mathcal{X}$ and initial dual iterate $\lambda_0 \in \mathbb{R}^m$. Suppose that the dual bound parameters $\gamma_k > 0$ be a bounded sequence satisfying $\sum_{k=0}^{\infty} \gamma_k = C_\gamma < \infty$, and that the dual sample sizes satisfy $|S_k^{(d)}| = \Theta((\epsilon^{(g)})^{-2})$ for all $k \in \mathbb{N}_0$. Let Algorithm 6 be the solver $\mathcal{M}$ under **Option II**. Suppose for $\bar{\epsilon}_k^{(g)} = \bar{\epsilon}^{(g)} \leq 1$, $\epsilon_k^{(H)} \leq 1$, such that primal subproblem tolerance inputs $\epsilon_k^{(g)} = \bar{\epsilon}_k^{(g)} \log(64(\mathcal{T}_k J_k + 1))$, $\epsilon^{(g)} = \epsilon_K^{(g)} \leq 1$ and $\epsilon^{(H)} = \epsilon_k^{(H)} \leq 1$, with number of iterations $k = K = K^*$, where K^* as per Theorem 2.1 and $\mathcal{T}_k$, J_k defined in Algorithm 6. Then, $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point satisfying (1.6) as per Definition 1.2 with $p = 1/2$, and $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$. Furthermore, if there exists a constant $q > 0$ such that $\log(1/\epsilon^{(g)}) \geq \log^q(1/\epsilon^{(H)})$, then the sample complexity of the algorithm, as defined in Definition 1.3, to obtain $\tilde{x} := x_K$ is*

1. ***(Deterministic Sample Complexity):***

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}).$$

2. ***(Probabilistic Sample Complexity):***

$$\mathcal{W} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})$$

with probability at least p^K.

Proof. From Lemma 4.5, we establish that the iterate x_k satisfies (2.5). As $\epsilon_k^{(g)} \leq \epsilon_K^{(g)} = \epsilon^{(g)}$ for all $k \leq K$ by definition; it holds from Theorem 2.1 that the iterate $\tilde{x} := x_K$ is an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point, according to Definition 1.2, with $p = 1/2$, and where $\tilde{\lambda} := \lambda_K + \alpha_K c(x_K)$.

From (3.24) and the fact that that $\beta^K = \mathcal{O}((\epsilon^{(g)})^{-1})$, we have,

$$\epsilon^{(g)} = \bar{\epsilon}^{(g)} \log(64(\mathcal{T}_K J_K + 1)) = \bar{\epsilon}^{(g)} \Theta(\log(\max\{(\epsilon^{(H)})^{-3}(\epsilon^{(g)})^{-3}(\bar{\epsilon}^{(g)})^{-1}, (\epsilon^{(H)})^{-1}(\epsilon^{(g)})^{-3}(\bar{\epsilon}^{(g)})^{-2}\}))$$

Then, from Lemma A.4, we have for any constant $r > 0$ that

$$\tilde{\mathcal{O}}((\bar{\epsilon}^{(g)})^{-r}) = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-r}). \tag{4.7}$$

Following the same approach as in (3.29) and (4.1), and using (4.7) we have

$$\begin{aligned} \mathcal{W}_{SPDR,0}^{(p)} &= \tilde{\mathcal{O}}\left((\bar{\epsilon}^{(g)})^{-3} + (\bar{\epsilon}^{(g)})^{-2}(\epsilon^{(H)})^{-2} + (\epsilon^{(H)})^{-5} + (\bar{\epsilon}^{(g)})^{-1}(\epsilon^{(H)})^{-3} \right) \\ &= \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-3} + (\epsilon^{(g)})^{-2}(\epsilon^{(H)})^{-2} + (\epsilon^{(H)})^{-5} + (\epsilon^{(g)})^{-1}(\epsilon^{(H)})^{-3} \right). \end{aligned} \tag{4.8}$$

Case (i): By the same reasoning as in (4.2) and using (4.7), we have

$$\sum_{k=1}^{K} \mathcal{W}_{SPDR,k}^{(p)} = \tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5}\right). \tag{4.9}$$

Substituting (4.8) and (4.9) in (3.19) completes the proof.

Case (ii): For probabilistic bounds, by following the same approach as that for (4.3) and using (4.7), we have

$$\mathbb{P}\left(\sum_{k=1}^{K} \mathcal{W}_{\mathcal{M},k}^{(p)} = \tilde{\mathcal{O}}((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-2} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})\right) \geq p^K. \tag{4.10}$$

We use (4.8) and (4.10) to complete the proof as in (3.32).

□

Remark 4.6. *We note that the sample complexity of MESCAL-SPIDER matches that of MESCAL-SG-HV-NC. However, we have an additional mild restriction on the parameters $\epsilon^{(g)}, \epsilon^{(H)}$ such that $\epsilon^{(H)}$ cannot decrease exponentially faster than $\epsilon^{(g)}$.*

5 Numerical Experiments

In this section, we demonstrate the empirical performance of the proposed MESCAL framework for solving a fairness constrained problem and a Neyman-Pearson classification problem. To solve the primal subproblem in Algorithm 1 of MESCAL, we consider the three solver choices discussed in Section 4: SG-HV-NC [6], Natasha2 [3], and SPIDER [24].

We make a few modifications to the MESCAL framework in Algorithm 1 to enhance its computational efficiency in practical implementations. The practical variant differs from Algorithm 1 in a few respects.

- **Primal variable update:** To conduct Step 3 of Algorithm 1, we terminate the primal solver and set $x_k := x_{k-1,t}$ after t iterations of the primal problem subsolver, if $x_{k-1,t}$ satisfies the conditions in Algorithm 1. We also impose a maximum number of inner iterations $t \leq t^*$. If x_{k-1,t^*} does not satisfy the conditions in Algorithm 1, then $x_k := x_{k-1,t^*}$. Finally, we impose a maximum number of outer iterations K, such that the outer loop terminates when $k = K$.

- **Dual variable update:** Instead of the bounded dual update as provided in Line 7 of Algorithm 1, we employ the full dual update by setting $\gamma_k = \infty$ for all $k \in \mathbb{N}_0$. Although the full dual update increases the theoretical computational complexity of expectation-constrained optimization [1], it leads to superior practical performance.

- **Subsampled gradient estimator**: Theoretically, in Section 3 and Section 4, we utilize a sample average approximation of the augmented Lagrangian gradient estimator in (3.1) for mini-batch gradient estimators. For improved practical performance, we instead use the modified estimator

$$\nabla\hat{\mathcal{L}}_\alpha(x,\lambda,S^{(p)}) := \frac{\sum_{\theta_i\in S^{(p)}}\nabla F(x,\xi_i)}{|S^{(p)}|} + \frac{\sum_{\theta_i\in S^{(p)}}\nabla C(x,\zeta_i^{(1)})}{|S^{(p)}|}^\top \left(\lambda + \alpha\frac{\sum_{\theta_i\in S^{(p)}} C(x,\zeta_i^{(2)})}{|S^{(p)}|}\right), \tag{5.1}$$

  where $S^{(p)} = \{(\xi_i,\zeta_i^{(1)},\zeta_i^{(2)})\}_{i=1}^{|S^{(p)}|}$, and $\zeta_{i_1}^{(1)},\zeta_{i_2}^{(2)} \overset{\text{i.i.d.}}{\sim} \mathcal{P}_\zeta$ for all i_1, i_2. We note that this new estimator is also unbiased. Moreover, the additional cross-product terms reduce the variance relative to subsampling the estimator in (3.1). We prove this variance reduction in Lemma A.3.

- **Subsampled Hessian estimator**: Theoretically, we utilize a sample average approximation of the augmented Lagrangian Hessian estimator in (3.2) for mini-batch Hessian estimators. For improved practical performance, we first consider a modified estimator, similar to (5.1), that benefits from variance reduction:

$$\nabla^2 \ddot{\mathcal{L}}_\alpha(x,\lambda,S^{(p)}) := \frac{\sum_{\theta_i \in S^{(p)}} \nabla^2 F(x,\xi_i)}{|S^{(p)}|} + \sum_{j=1}^{m} \left(\lambda^{(j)} + \alpha \frac{\sum_{\theta_i \in S^{(p)}} C_j(x,\zeta_i^{(2)})}{|S^{(p)}|} \right) \frac{\sum_{\theta_i \in S^{(p)}} \nabla^2 C_j(x,\zeta_i^{(1)})}{|S^{(p)}|}^\top + \alpha \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(1)})^\top}{|S^{(p)}|} \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(2)})}{|S^{(p)}|}. \tag{5.2}$$

  As discussed above, the cross-product terms provide variance reduction. However, this estimator is not symmetric because of the term involving the product of constraint Jacobians. Since nonsymmetric Hessian estimates can increase computational overhead in floating-point implementations, we symmetrize this Gram matrix term as follows:

$$\frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(1)})^\top}{|S^{(p)}|} \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(2)})}{|S^{(p)}|} \rightarrow \frac{1}{2} \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(1)})^\top}{|S^{(p)}|} \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(2)})}{|S^{(p)}|} + \frac{1}{2} \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(2)})^\top}{|S^{(p)}|} \frac{\sum_{\theta_i \in S^{(p)}} \nabla C(x,\zeta_i^{(1)})}{|S^{(p)}|} \tag{5.3}$$

  This modification does not introduce bias. However, the resulting estimator $\nabla^2 \hat{\mathcal{L}}_\alpha(x,\lambda,S^{(p)})$ obtained by using (5.3) in (5.2) may have a higher variance compared to estimator $\nabla^2 \ddot{\mathcal{L}}_\alpha(x,\lambda,S^{(p)})$. We use the symmetrized estimator in our practical implementations because it retains the variance-reduction benefit of the cross-product terms while improving computational efficiency through symmetry.

- **Handling inequality constraints:** Typically, slack variables are handled implicitly in augmented Lagrangian methods using implicit slack formulations, e.g. see [54]. However, we handle inequality constraints explicitly, by augmenting primal variable x with slack variables s, and setting $\mathcal{X} = \mathbb{R}^d \times \mathbb{R}_+^{\dim(s)}$ with non-negativity constraints in the projection set. We record three values at every outer iteration of the primal solver: 1) the first order stationarity error $\|\text{proj}_{T_\mathcal{X}(x,s)}(-\nabla_{(x,s)}\mathcal{L}_\alpha(x,s,\lambda))\|$ corresponding to (1.4)(i), 2) the feasibility error $\|c(x)+s\|$ corresponding to (1.4)(ii), and 3) the minimum eigenvalue of the Hessian of the augmented Lagrangian function $\sigma_{\min}(\nabla^2_{(x,s)}\mathcal{L}_\alpha(x,s,\lambda))$ which provides a lower bound on (1.4)(iii). We provide Remark A.2 in the Appendix on how the first-order stationarity error in (1.4)(i) encodes primal feasibility, dual feasibility and complementary slackness for standard inequality constraints, and why we utilize the minimum eigenvalue of the Hessian of the augmented Lagrangian function instead of utilizing (1.4)(iii).

We implemented the practical variant described above, together with the three primal solvers, in Python on a shared Ubuntu 22.04.5 LTS system with CUDA support.[1] All computations were performed on a machine with an Intel Xeon Gold 6426Y processor with 64 CPU cores across two NUMA nodes and an NVIDIA L40S GPU with driver version 550.120 and CUDA 12.4. Each run was allocated 4 CPU cores and 16 GB of RAM. The system has 251 GB of total memory and 15 GB of swap space, although actual memory usage varied across runs. Each solver has several hyperparameters, as described in Section 4. For each problem, we first selected reasonable values for these parameters based on the conditions recommended in the respective references and preliminary empirical results. We then performed a local grid search to further improve the performance of each solver. A complete fine-tuning of all solver hyperparameters is beyond

[1] Our code will be made publicly available upon publication of the manuscript.

the scope of this paper due to the size of the resulting parameter space and the high computational cost of individual runs.

5.1 Nonconvex Fairness Constrained Problem

We first consider a machine learning task involving a data-driven fairness constraint, where we fit a classification model with a fairness-based constraint that ensures a target group receives a sufficient rate of positive outcomes. Let $\mathcal{D} = \{(a_i, y_i)\}_{i=1}^{|\mathcal{D}|}$ where $a_i \in \mathbb{R}^d$ represents the feature vector of a particular realization and $y_i \in \{-1, 1\}$ represents the class label; let $\mathcal{A} = \{a_j\}_{j=1}^{|\mathcal{A}|}$ denote a possibly unlabeled set representing the general population; and let $\mathcal{A}_{\text{tar}} \subseteq \mathcal{A}$ denote the target group. The goal is to minimize the empirical risk over $\mathcal{D}$ while ensuring the average predicted probability for the target group is at least a fraction $\tau_{\mathcal{A}}$ of the average predicted probability for the entire population $\mathcal{A}$. This problem can be formulated as the following constrained optimization problem:

$$\begin{aligned} \min_{x \in \mathbb{R}^d} \; & f(x) := \frac{1}{|\mathcal{D}|} \sum_{i:(a_i,y_i)\in\mathcal{D}} \phi_\rho(\ell(x; a_i, y_i)), \\ \text{s.t. } & c(x) := \frac{\tau_{\mathcal{A}}}{|\mathcal{A}|} \sum_{i:a_i\in\mathcal{A}} \sigma(a_i^\top x) - \frac{1}{|\mathcal{A}_{\text{tar}}|} \sum_{i:a_i\in\mathcal{A}_{\text{tar}}} \sigma(a_i^\top x) \le 0, \end{aligned} \tag{5.4}$$

where $x \in \mathbb{R}^d$ denotes the learnable parameters of the model, $\ell(x; a_i, y_i) := \log(1 + \exp(-y_i a_i^\top x))$ is the standard logistic loss function, and $\phi_\rho(u) := \rho \log\left(1 + \frac{u}{\rho}\right)$ is a concave truncation function used to truncate the loss, where ρ is a scaling parameter. The function $\sigma(z) := (1 + \exp(-z))^{-1}$ represents the predicted probability of a sample belonging to the positive class, and $\tau_{\mathcal{A}}$ denotes the fairness fraction parameter; see [44] for further details. We use two datasets to evaluate the performance of our algorithm on this problem:

- *bank [22]:* $d = 81$, $(|\mathcal{D}|, |\mathcal{A}|, |\mathcal{A}_{tar}|) = (22605, 22605, 6320)$, and $\tau_{\mathcal{A}} = 0.4$.
- *loan [25]:* $d = 250$, $(|\mathcal{D}|, |\mathcal{A}|, |\mathcal{A}_{tar}|) = (63890, 64485, 31966)$, and $\tau_{\mathcal{A}} = 0.6$.

To solve (5.4) using MESCAL, we reformulate its inequality constraint to an equality constraint $c(x)+s = 0$, where $s \ge 0$ is enforced by projections after each iterate update. This reformulation has stationarity conditions equivalent to those of the original problem (5.4) as shown in [38]. In all experiments, we set the tolerances to $\epsilon = \epsilon^{(g)} = \epsilon^{(H)} = 0.01$. The algorithm terminates if the first-order stationarity, feasibility, and a lower bound on second-order stationarity conditions in (1.4) for the equality-constrained reformulation of (5.4) are all satisfied to within ϵ. For all subroutines, at the k-th iteration of MESCAL, we set $\alpha_k = 1.1^{k+1}$. All methods are initialized at $x_{-1} = \frac{20}{\sqrt{d}}[11\ldots]^\top$ where $[11\ldots]^\top$ represents a vector of ones.

Figure 1 plots the second-order stationarity error, first-order stationarity error, and feasibility error against the total work. Here, the total work is defined as the total number of stochastic augmented Lagrangian gradient evaluations plus twice the total number of stochastic augmented Lagrangian Hessian vector product evaluations. The factor of two reflects the convention that a Hessian vector product evaluation requires approximately twice the computational effort of a gradient evaluation. As established in Lemma 3.7, this measure of work accounts for the total stochastic objective gradient, constraint function, constraint gradient, objective Hessian vector product, and constraint Hessian vector product evaluations.

We observe that SPIDER requires approximately one order of magnitude less work than SG-HV-NC and two orders of magnitude less work than Natasha2 to reach the desired accuracy. Although appropriate tuning of the hyperparameters may slightly change the relative performance of these solvers, SPIDER consistently performed well in our experiments. This behavior can be attributed, at least in part, to its use of SARAH-based gradient updates and its relatively infrequent negative curvature searches. For the problems considered, the iterates ($\{x_k\}$) rapidly move out of regions of negative curvature, reducing the need for repeated negative curvature searches. Consequently, SPIDER avoids frequent Hessian vector product evaluations and achieves lower overall computational work. All methods converge to the prescribed tolerances,

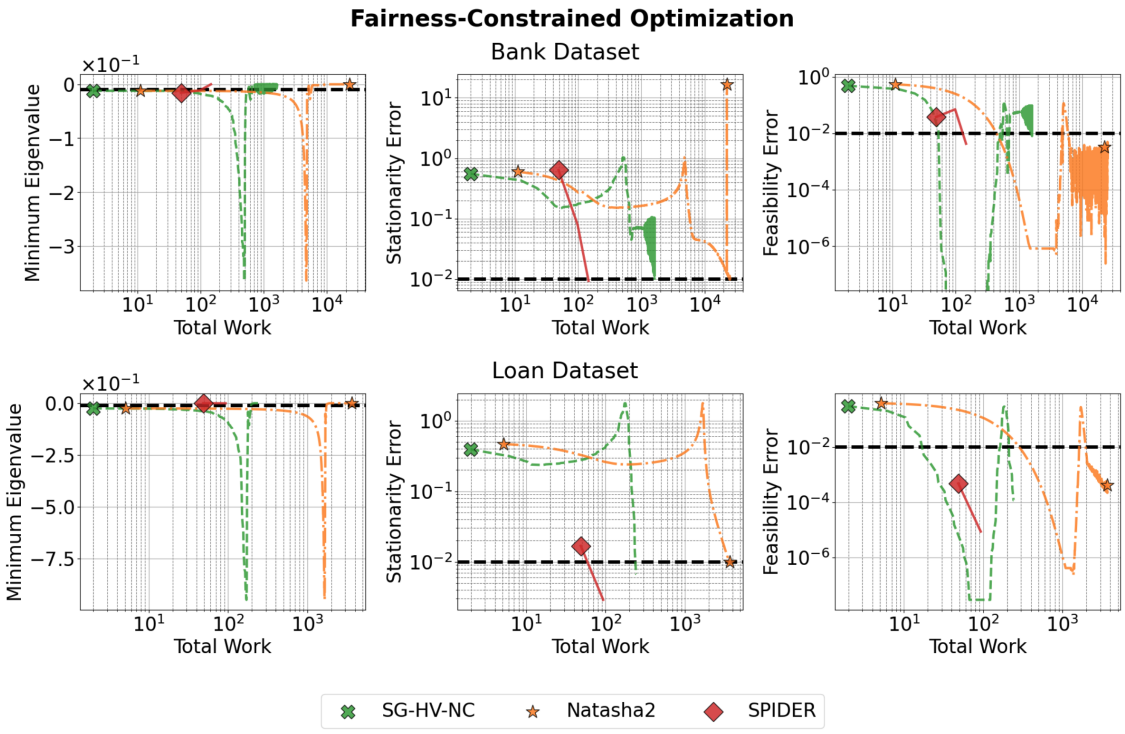

Figure 1: Results of Algorithm 1 on the nonconvex fairness-constrained optimization problem utilizing the *bank* [22] and the *loan* [25] dataset. From left to right, the plots report the minimum eigenvalue of the Hessian of the augmented Lagrangian, the first-order stationarity error (1.4)(i), and the feasibility error (1.4)(ii). Results are shown for the three primal subproblem solvers SG-HV-NC, Natasha2, and SPIDER. The x-axis represents the total computational work, measured by the number of stochastic augmented Lagrangian gradient and Hessian vector product evaluations. Markers indicate the beginning of each MESCAL iteration for the corresponding solver. All solvers use the same initial point; the initial point is not shown because the x-axis is on a logarithmic scale.

although the stationarity and feasibility errors for SG-HV-NC and Natasha2 exhibit some transient increases before converging to the prescribed tolerances.

5.2 Nonconvex Neyman-Pearson Classification

We now consider a nonconvex Neyman-Pearson classification problem [52, 68], where the goal is to minimize the false-negative error rate while ensuring that the false-positive error rate does not exceed a prescribed maximum value. This problem can be formulated as the following constrained optimization problem:

$$\begin{aligned} \min_{x\in\mathbb{R}^d} \; & f(x) := \frac{1}{N^+}\sum_{i=1}^{N^+} \phi(x^\top a_i^+), \\ \text{s.t.} \; & c(x) := \frac{1}{N^-}\sum_{j=1}^{N^-} \phi(-x^\top a_j^-) - \tau_{FP} \le 0, \end{aligned} \tag{5.5}$$

where $\{a_i^+\}_{i=1}^{N^+}$ and $\{a_j^-\}_{j=1}^{N^-}$ denotes the positive-class samples and negative-class samples in the training data set, respectively. The parameter τ_{FP} specifies the maximum allowable false-positive error rate. In (5.5), we set $\phi(\cdot)$ to be the sigmoid function: $\phi(u) = (1+\exp(u))^{-1}$. Similar to the approach in Section 5.1, we

reformulate the inequality constraint in (5.5) as the equality constraint $c(x) + s = 0$, with $s \geq 0$, such that (5.5) is of the form (1.1), where $\mathcal{X} = \{x \in \mathbb{R}^d, s \in \mathbb{R}_+\}$, and the random variables ζ and ξ in (1.1) correspond to uniformly sampling from the N^+ positive-class samples and N^- negative-class samples, respectively. We use two datasets to evaluate the performance of our algorithm on this problem:

- *spambase [22]:* $d = 57$, $(N^+, N^-) = (1813, 2788)$, and $\tau_{FP} = 0.2$.
- *madelon [27]:* $d = 500$, $(N^+, N^-) = (1300, 1300)$, and $\tau_{FP} = 0.4$.

Following [68], we preprocess each dataset by first standardizing each feature to zero mean and unit variance, and subsequently scaling each feature vector to unit Euclidean norm. The tolerance was set to $\epsilon = \epsilon^{(g)} = \epsilon^{(H)} = 0.01$ in all tests on the *spambase* and *madelon* problem settings. The method is set to terminate if the first-order stationarity, feasibility, and a lower bound on second-order stationarity conditions (see (1.4)) of the equality-constrained reformulation of (5.5) are all satisfied to within ϵ. For all subroutines, at the k-th outer iteration of MESCAL, we set $\alpha_k = 1.1^{k+1}$. All methods are initialized at $x_{-1} = \mathbf{0}$.

Figure 2 plots the second-order stationarity error, first-order stationarity error, and feasibility error against the total work. As in the fairness-constrained problem, SPIDER requires lower computational work than SG-HV-NC and Natasha2, requiring approximately one order of magnitude less work for the *madelon* dataset and two orders of magnitude less work for the *spambase* dataset. We attribute this difference, as discussed above, to the relatively infrequent negative curvature searches performed by SPIDER.

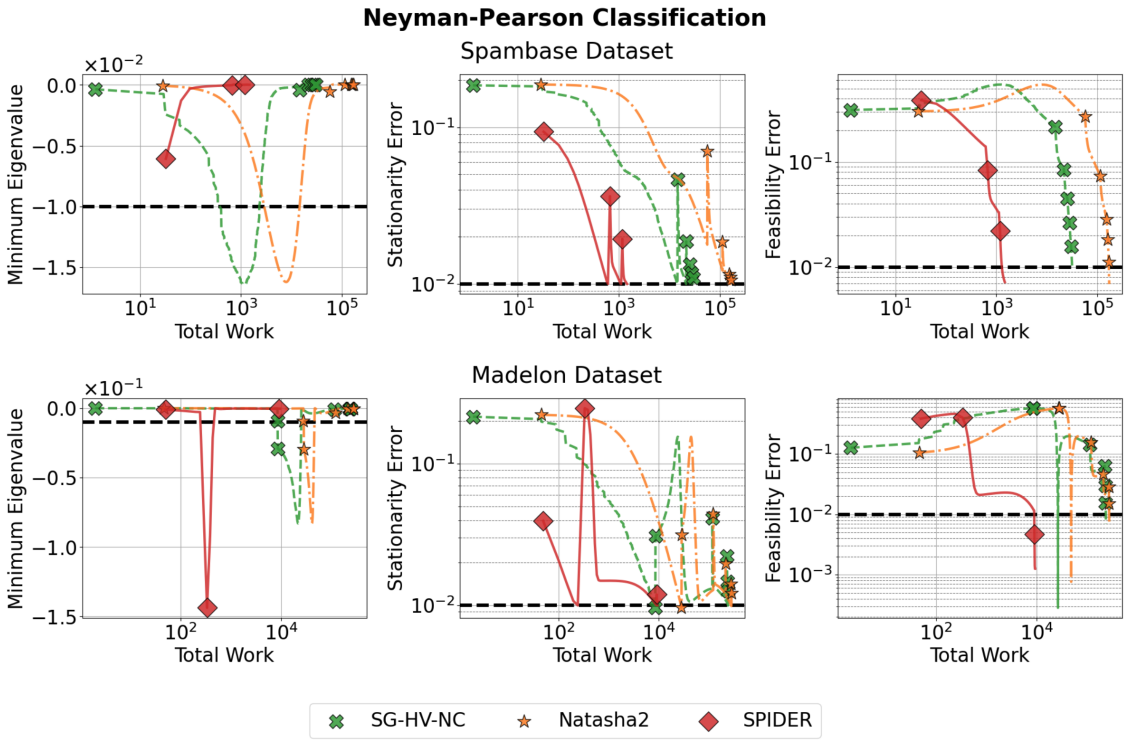

Figure 2: Results of Algorithm 1 on the Neyman-Pearson classification problem (5.5) utilizing the *spambase* [22] and the *madelon* [27] dataset. From left to right, the plots report the minimum eigenvalue of the Hessian of the augmented Lagrangian, the first-order stationarity error (1.4)(i), and the feasibility error (1.4)(ii). Results are shown for the three primal subproblem solvers SG-HV-NC, Natasha2, and SPIDER. The x-axis represents the total computational work, measured by the number of stochastic augmented Lagrangian gradient and Hessian vector product evaluations. Markers indicate the beginning of each MESCAL iteration for the corresponding solver. All solvers use the same initial point; the initial point is not shown because the x-axis is on a logarithmic scale.

Despite the difference in total work, the error trajectories remain comparable across the three solvers. For *spambase* dataset, the feasibility error consistently decreases, while the second-order stationarity error initially increases before decreasing, and the first-order stationarity error increases after each dual variable update. SG-HV-NC and Natasha2 also require more outer iterations than SPIDER. For *madelon*, all three solvers exhibit closely matching error trajectories. After the first iteration, the iterates enter region with negative curvature, reflected by a large second-order stationarity error, along with large first-order stationarity and feasibility errors. Subsequently, all three errors decrease rapidly before gradually converging to the prescribed tolerances. The nonmonotonic behavior of the errors is not unexpected due to the interplay among the first-order gradient step, second-order negative curvature step, and dual variable update step.

6 Final Remarks

We developed and analyzed MESCAL, an augmented Lagrangian framework for solving stochastic nonconvex optimization problems with expectation constraints over a closed and convex set. We proposed second-order inexactness conditions in expectation and with prescribed probability for solving the primal subproblems inexactly and established corresponding iteration complexity results. Under reasonable assumptions on the theoretical properties of a general primal solver, we established a sample complexity result of $\mathcal{O}((\epsilon^{(g)})^{-5} + (\epsilon^{(g)})^{-4}(\epsilon^{(H)})^{-5})$ for obtaining an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate second-order stationary point satisfying (1.5) or (1.6). We further incorporated three existing stochastic second-order subproblem solvers into the framework under the restriction $\mathcal{X} = \mathbb{R}^d$ and established corresponding deterministic sample complexity results to achieve an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate p-probabilistic second-order stationary point, with sample complexity bounds of the form $\tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5} + (\epsilon^{(g)})^{-6}(\epsilon^{(H)})^{-2}\right)$ or $\tilde{\mathcal{O}}\left((\epsilon^{(g)})^{-5}(\epsilon^{(H)})^{-5} + (\epsilon^{(g)})^{-7}(\epsilon^{(H)})^{-1}\right)$ depending on the chosen primal solver. The corresponding probabilistic bounds improve the dependence on $\epsilon^{(g)}$ in the solver-specific terms by a factor of $\mathcal{O}((\epsilon^{(g)})^{-1})$. To the best of our knowledge, sample complexity guarantees for obtaining approximate second-order stationary points in nonconvex expectation-constrained optimization have not been established previously. Our numerical experiments on two nonconvex machine learning problems further illustrate the practical performance of MESCAL with the considered subproblem solvers.

Our analysis also highlights several open problems in stochastic nonconvex optimization that are directly relevant to extending the scope of the proposed framework. Based on our literature review, we are unaware of existing methods and/or complexity results for the following settings:

- Second-order solvers for expectation-constrained optimization problems over general closed and convex constraint sets.
- Complexity results for obtaining second-order stationary points in expectation for unconstrained or convex-constrained stochastic optimization. In particular, Berahas et al. [9] provide stationarity guarantees for obtaining a second-order stationary point in expectation for unconstrained optimization, but do not establish corresponding sample complexity results.
- Algorithms for computing negative curvature directions that lie within a convex constraint set. In particular, we are unaware of a projected variant of Oja's algorithm for identifying a negative curvature direction subject to convex constraints.
- Single-loop second-order solvers for expectation-constrained optimization problems. Developing such methods is nontrivial, as discussed in [1].

Developing methods for these problems would directly expand the class of subproblem solvers that can be incorporated into the proposed framework and could lead to stronger theoretical guarantees. These directions provide several avenues for future work.

Acknowledgments

The authors thank Zichong Li for helpful initial discussions that stimulated this work.

References

[1] Ahmet Alacaoglu and Stephen J. Wright. Complexity of Single Loop Algorithms for Nonlinear Programming with Stochastic Objective and Constraints. In *Proceedings of The 27th International Conference on Artificial Intelligence and Statistics*, pages 4627–4635. PMLR, April 2024.

[2] Zeyuan Allen-Zhu. How To Make the Gradients Small Stochastically: Even Faster Convex and Nonconvex SGD. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[3] Zeyuan Allen-Zhu. Natasha 2: Faster Non-Convex Optimization Than SGD, June 2018. arXiv:1708.08694.

[4] Zeyuan Allen-Zhu and Yuanzhi Li. Follow the Compressed Leader: Faster Online Learning of Eigenvectors and Faster MMWU. In *Proceedings of the 34th International Conference on Machine Learning*, pages 116–125. PMLR, July 2017.

[5] Zeyuan Allen-Zhu and Yuanzhi Li. NEON2: Finding Local Minima via First-Order Oracles. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[6] Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Ayush Sekhari, and Karthik Sridharan. Second-Order Information in Non-Convex Stochastic Optimization: Power and Limitations. In *Proceedings of Thirty Third Conference on Learning Theory*, pages 242–299. PMLR, July 2020.

[7] Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization. *Mathematical Programming*, 199(1):165–214, May 2023.

[8] Shamsulhaq Basir and Inanc Senocak. Physics and equality constrained artificial neural networks: Application to forward and inverse problems with multi-fidelity data fusion. *Journal of Computational Physics*, 463:111301, August 2022.

[9] Albert S. Berahas, Raghu Bollapragada, and Wanping Dong. Exploiting negative curvature in conjunction with adaptive sampling: theoretical results and a practical algorithm. *Computational Optimization and Applications*, 94(1):1–37, May 2026.

[10] Dimitri P. Bertsekas. On Penalty and Multiplier Methods for Constrained Minimization. *SIAM Journal on Control and Optimization*, 14(2):216–235, February 1976.

[11] Raghu Bollapragada and Shagun Gupta. On the convergence and complexity of proximal gradient and accelerated proximal gradient methods under adaptive gradient estimation. *Computational Optimization and Applications*, May 2026.

[12] Raghu Bollapragada, Cem Karamanli, Brendan Keith, Boyan Lazarov, Socratis Petrides, and Jingyi Wang. An adaptive sampling augmented Lagrangian method for stochastic optimization with deterministic constraints. *Computers & Mathematics with Applications*, 149:239–258, November 2023.

[13] Raghu Bollapragada and Yash Kumar. Augmented Lagrangian Methods for Expectation-Constrained Optimization via Adaptive Sampling Strategies. Manuscript in preparation, 2026.

[14] Digvijay Boob, Qi Deng, and Guanghui Lan. Stochastic first-order methods for convex and nonconvex functional constrained optimization. *Mathematical Programming*, 197(1):215–279, January 2023.

[15] Coralia Cartis, Nicholas I. M. Gould, and Philippe L. Toint. Adaptive cubic regularisation methods for unconstrained optimization. Part II: worst-case function- and derivative-evaluation complexity. *Mathematical Programming*, 130(2):295–319, December 2011.

[16] Coralia Cartis, Nicholas I. M. Gould, and Philippe L. Toint. Complexity bounds for second-order optimality in unconstrained optimization. *Journal of Complexity*, 28(1):93–108, February 2012.

[17] Frank E. Curtis, Vyacheslav Kungurtsev, Daniel P. Robinson, and Qi Wang. A Stochastic-Gradient-Based Interior-Point Algorithm for Solving Smooth Bound-Constrained Optimization Problems. *SIAM Journal on Optimization*, 35(2):1030–1059, June 2025.

[18] Ashok Cutkosky and Harsh Mehta. High-probability Bounds for Non-Convex Stochastic Optimization with Heavy Tails. In *Advances in Neural Information Processing Systems*, volume 34, pages 4883–4895. Curran Associates, Inc., 2021.

[19] Ashok Cutkosky and Francesco Orabona. Momentum-Based Variance Reduction in Non-Convex SGD. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[20] Ibrahim Dincer, Marc A. Rosen, and Pouria Ahmadi. *Optimization of Energy Systems*. John Wiley & Sons, Ltd, 1 edition, 2017.

[21] Michele Donini, Luca Oneto, Shai Ben-David, John S. Shawe-Taylor, and Massimiliano Pontil. Empirical Risk Minimization Under Fairness Constraints. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[22] Dheeru Dua and Graff Casey. UCI Machine Learning Repository, 2019.

[23] Francisco Facchinei and Vyacheslav Kungurtsev. Stochastic Approximation for Expectation Objective and Expectation Inequality-Constrained Nonconvex Optimization, July 2025. arXiv:2307.02943 [math].

[24] Cong Fang, Chris J. Li, Zhouchen Lin, and Tong Zhang. SPIDER: Near-Optimal Non-Convex Optimization via Stochastic Path Integrated Differential Estimator, October 2018. arXiv:1807.01695 [math].

[25] Nathan George. All Lending Club loan data. tex.howpublished: Kaggle.

[26] Saeed Ghadimi and Guanghui Lan. Stochastic First- and Zeroth-Order Methods for Nonconvex Stochastic Programming. *SIAM Journal on Optimization*, 23(4):2341–2368, January 2013.

[27] Isabelle Guyon, Steve Gunn, Asa Ben-Hur, and Gideon Dror. Result Analysis of the NIPS 2003 Feature Selection Challenge. In *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

[28] Magnus R. Hestenes. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5):303–320, November 1969.

[29] Magnus R. Hestenes. *Optimization theory : the finite dimensional case*. Pure and applied mathematics. Wiley, New York, NY, USA, 1975.

[30] Yankun Huang and Qihang Lin. Oracle Complexity of Single-Loop Switching Subgradient Methods for Non-Smooth Weakly Convex Functional Constrained Optimization. *Advances in Neural Information Processing Systems*, 36:61327–61340, December 2023.

[31] Aditi Krishnapriyan, Amir Gholami, Shandian Zhe, Robert Kirby, and Michael Mahoney. Characterizing possible failure modes in physics-informed neural networks. In *Advances in Neural Information Processing Systems*, volume 34, pages 26548–26560. Curran Associates, Inc., 2021.

[32] Guanghui Lan and Zhiqiang Zhou. Algorithms for stochastic optimization with function or expectation constraints. *Computational Optimization and Applications*, 76(2):461–498, June 2020.

[33] Steven M. LaValle. *Planning algorithms*, volume 1. Cambridge university press Cambridge, UK, 2006.

[34] Sébastien Lengagne, Joris Vaillant, Eiichi Yoshida, and Abderrahmane Kheddar. Generation of whole-body optimal dynamic multi-contact motions. *The International Journal of Robotics Research*, 32(9-10):1104–1119, August 2013.

[35] Thomas Lew, Riccardo Bonalli, and Marco Pavone. Sample Average Approximation for Stochastic Programming with Equality Constraints. *SIAM Journal on Optimization*, 34(4):3506–3533, December 2024.

[36] Fei Li and Zheng Qu. An inexact proximal augmented Lagrangian framework with arbitrary linearly convergent inner solver for composite convex optimization. *Mathematical Programming Computation*, 13(3):583–644, September 2021.

[37] Zhize Li and Jian Li. A Simple Proximal Stochastic Gradient Method for Nonsmooth Nonconvex Optimization. In *Proceedings of the 32nd Conference on Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[38] Zichong Li, Pin-Yu Chen, Sijia Liu, Songtao Lu, and Yangyang Xu. Rate-improved inexact augmented Lagrangian method for constrained nonconvex optimization. In *Proceedings of The 24th International Conference on Artificial Intelligence and Statistics*, pages 2170–2178. PMLR, March 2021.

[39] Zichong Li, Pin-Yu Chen, Sijia Liu, Songtao Lu, and Yangyang Xu. Stochastic inexact augmented Lagrangian method for nonconvex expectation constrained optimization. *Computational Optimization and Applications*, 87(1):117–147, January 2024.

[40] Yuqing Liang and Dongpo Xu. Almost sure convergence rates of stochastic proximal gradient descent algorithm. *Optimization*, 73(8):2413–2446, August 2024.

[41] Qihang Lin, Runchao Ma, and Yangyang Xu. Complexity of an inexact proximal-point penalty method for constrained smooth non-convex optimization. *Computational Optimization and Applications*, 82(1):175–224, May 2022.

[42] Wei Liu and Yangyang Xu. A SPIDER-Type Stochastic Subgradient Method for Expectation-Constrained Nonconvex Nonsmooth Optimization. *SIAM Journal on Optimization*, 36(2):1125–1153, June 2026.

[43] Songtao Lu. A Single-Loop Gradient Descent and Perturbed Ascent Algorithm for Nonconvex Functional Constrained Optimization. In *Proceedings of the 39th International Conference on Machine Learning*, pages 14315–14357. PMLR, June 2022.

[44] Runchao Ma, Qihang Lin, and Tianbao Yang. Quadratically Regularized Subgradient Methods for Weakly Convex Optimization with Weakly Convex Constraints. In *Proceedings of the 37th International Conference on Machine Learning*, pages 6554–6564. PMLR, November 2020.

[45] Friedrich Menhorn, Florian Augustin, Hans-Joachim Bungartz, and Youssef M. Marzouk. A trust-region method for derivative-free nonlinear constrained stochastic optimization, January 2022. arXiv:1703.04156 [math].

[46] Aryan Mokhtari, Asuman Ozdaglar, and Ali Jadbabaie. Escaping Saddle Points in Constrained Optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[47] Jean-Jacques Moreau. Décomposition orthogonale d'un espace hilbertien selon deux cônes mutuellement polaires. *Comptes Rendus Hebdomadaires des Séances de l'Académie des Sciences*, 255:238–240, 1962.

[48] Lam M. Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A Novel Method for Machine Learning Problems Using Stochastic Recursive Gradient. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2613–2621. PMLR, July 2017.

[49] Jorge Nocedal and Stephen J. Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[50] Luca Oneto and Silvia Chiappa. Fairness in Machine Learning. In Luca Oneto, Nicolò Navarin, Alessandro Sperduti, and Davide Anguita, editors, *Recent Trends in Learning From Data: Tutorials from the INNS Big Data and Deep Learning Conference (INNSBDDL2019)*, pages 155–196. Springer International Publishing, Cham, 2020.

[51] Michael J. D. Powell. A method for nonlinear constraints in minimization problems. *Optimization*, pages 283–298, 1969.

[52] Philippe Rigollet and Xin Tong. Neyman-Pearson Classification, Convexity and Stochastic Constraints. *Journal of Machine Learning Research*, 12(86):2831–2855, 2011.

[53] R. Tyrrell Rockafellar and Roger J. B. Wets. *Variational Analysis*, volume 317 of *Grundlehren der mathematischen Wissenschaften*. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[54] Ralph T. Rockafellar. A dual approach to solving nonlinear programming problems by unconstrained optimization. *Mathematical Programming*, 5(1):354–373, December 1973.

[55] Farbod Roosta-Khorasani and Michael W. Mahoney. Sub-sampled Newton methods. *Mathematical Programming*, 174(1):293–326, March 2019.

[56] Clément W. Royer, Michael O'Neill, and Stephen J. Wright. A Newton-CG algorithm with complexity guarantees for smooth unconstrained optimization. *Mathematical Programming*, 180(1):451–488, March 2020.

[57] Abdurakhmon Sadiev, Marina Danilova, Eduard Gorbunov, Samuel Horváth, Gauthier Gidel, Pavel Dvurechensky, Alexander Gasnikov, and Peter Richtárik. High-Probability Bounds for Stochastic Optimization and Variational Inequalities: the Case of Unbounded Variance. In *Proceedings of the 40th International Conference on Machine Learning*, pages 29563–29648. PMLR, July 2023.

[58] Mehmet F. Sahin, Armin Eftekhari, Ahmet Alacaoglu, Fabian Latorre, and Volkan Cevher. An Inexact Augmented Lagrangian Framework for Nonconvex Optimization with Nonlinear Constraints. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[59] Ohad Shamir. Convergence of Stochastic Gradient Descent for PCA. In *Proceedings of The 33rd International Conference on Machine Learning*, pages 257–265. PMLR, June 2016.

[60] Haoming Shen, Yang Zeng, and Baoyu Zhou. Sequential Quadratic Optimization for Solving Expectation Equality Constrained Stochastic Optimization Problems, March 2025. arXiv:2503.09490 [math].

[61] Nilesh Tripuraneni, Mitchell Stern, Chi Jin, Jeffrey Regier, and Michael Jordan. Stochastic Cubic Regularization for Fast Nonconvex Optimization. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[62] Joel A. Tropp. An Introduction to Matrix Concentration Inequalities. *Foundations and Trends in Machine Learning*, 8(1-2):1–230, May 2015.

[63] Allen J. Wood and Bruce F. Wollenberg. Power generation operation and control — 2nd edition. *Fuel and Energy Abstracts*, 37(3):195, May 1996.

[64] Nachuan Xiao, Kuangyu Ding, Xiaoyin Hu, and Kim-Chuan Toh. Developing Lagrangian-Based Methods for Nonsmooth Nonconvex Optimization. *Mathematics of Operations Research*, January 2026.

[65] Yue Xie and Stephen J. Wright. Complexity of Proximal Augmented Lagrangian for Nonconvex Optimization with Nonlinear Equality Constraints. *Journal of Scientific Computing*, 86(3):38, February 2021.

[66] Peng Xu, Fred Roosta, and Michael W. Mahoney. Newton-type methods for non-convex optimization under inexact Hessian information. *Mathematical Programming*, 184(1):35–70, November 2020.

[67] Yangyang Xu. Iteration complexity of inexact augmented Lagrangian methods for constrained convex programming. *Mathematical Programming*, 185(1):199–244, January 2021.

[68] Yonggui Yan and Yangyang Xu. Adaptive primal-dual stochastic gradient method for expectation-constrained convex stochastic programs. *Mathematical Programming Computation*, 14(2):319–363, June 2022.

[69] Shuoguang Yang, Xudong Li, and Guanghui Lan. Data-Driven Minimax Optimization with Expectation Constraints. *Operations Research*, 73(3):1345–1365, May 2025.

[70] Hao Yu, Michael Neely, and Xiaohan Wei. Online Convex Optimization with Stochastic Constraints. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[71] Liwei Zhang, Yule Zhang, Jia Wu, and Xiantao Xiao. Solving Stochastic Optimization with Expectation Constraints Efficiently by a Stochastic Augmented Lagrangian-Type Algorithm. *INFORMS Journal on Computing*, 34(6):2989–3006, November 2022.

A Additional Results

Lemma A.1. *Suppose $\mathcal{X}$ is a closed and convex set. For any vectors a and b, it holds that*

$$\|\mathrm{proj}_{\mathcal{X}}(a+b)\| \le \|\mathrm{proj}_{\mathcal{X}}(a)\| + \|b\|.$$

Proof. By the triangle inequality, and the non-expansive property of the projection operator $\mathrm{proj}_{\mathcal{X}}$, we have

$$\begin{aligned}\|\mathrm{proj}_{\mathcal{X}}(a+b)\| &\le \|\mathrm{proj}_{\mathcal{X}}(a+b) - \mathrm{proj}_{\mathcal{X}}(a)\| + \|\mathrm{proj}_{\mathcal{X}}(a)\| \\ &\le \|(a+b)-a\| + \|\mathrm{proj}_{\mathcal{X}}(a)\| = \|b\| + \|\mathrm{proj}_{\mathcal{X}}(a)\|.\end{aligned}$$

□

Lemma A.2. *Let $p \in (0,1)$ and $r \in \mathbb{R}$ be constants. As $\epsilon \to 0^+$,*

$$\left(\log\left(\frac{1}{1-p^{1/\log(1/\epsilon)}}\right)\right)^r = \Theta\left((\log\log(1/\epsilon))^r\right) = \widetilde{\mathcal{O}}(1).$$

Proof. Let $x := \log(1/\epsilon)$. As $\epsilon \to 0^+$, we have $x \to \infty$. Since $p^{1/x} = \exp\left(-\frac{\ln(1/p)}{x}\right)$, a first-order Taylor expansion gives

$$1 - p^{1/x} = \frac{\ln(1/p)}{x} + \mathcal{O}(x^{-2}) = \frac{\ln(1/p)}{x}\left(1 + \mathcal{O}(x^{-1})\right).$$

Hence,

$$\log\left(\frac{1}{1-p^{1/x}}\right) = \log x - \log\ln(1/p) - \log\left(1+\mathcal{O}(x^{-1})\right) = \log x + \mathcal{O}(1).$$

Writing $U := \log x$, we have $U \to \infty$, and thus $\log\left(\frac{1}{1-p^{1/x}}\right) = U\left(1+\mathcal{O}(U^{-1})\right)$. As r is constant, $\left(\log\left(\frac{1}{1-p^{1/x}}\right)\right)^r = U^r\left(1+\mathcal{O}(U^{-1})\right)^r = U^r\left(1+\mathcal{O}(U^{-1})\right)$.

Substituting $U = \log\log(1/\epsilon)$ proves

$$\left(\log\left(\frac{1}{1-p^{1/\log(1/\epsilon)}}\right)\right)^r = \Theta\left((\log\log(1/\epsilon))^r\right) = \widetilde{\mathcal{O}}(1).$$

□

Lemma A.3. *For $x \in \mathcal{X}$, let $\{\zeta_i^{(1)}\}_{i=1}^n$ and $\{\zeta_i^{(2)}\}_{i=1}^n$ be two independent samples, each consisting of i.i.d. draws from $\mathcal{P}_\zeta$. Define, for $j \in \{1, 2, \cdots, m\}$,*

$$X_i^{(j)} := \nabla C^{(j)}(x, \zeta_i^{(1)}) \in \mathbb{R}^d, \qquad Y_i^{(j)} := C^{(j)}(x, \zeta_i^{(2)}) \in \mathbb{R}.$$

Assume $n \geq 2$, $\mathbb{E}[\|X_i^{(j)}\|^2] < \infty$, $\mathbb{E}[(Y_i^{(j)})^2] < \infty$. Then,

$$Var\left(\frac{1}{n}\sum_{i=1}^n X_i^{(j)} Y_i^{(j)}\right) \geq Var\left(\left(\frac{1}{n}\sum_{i=1}^n X_i^{(j)}\right)\left(\frac{1}{n}\sum_{i=1}^n Y_i^{(j)}\right)\right) \quad \forall j \in \{1, 2, \cdots, m\}.$$

Proof. Consider

$$\mu_X^{(j)} := \mathbb{E}[X_i^{(j)}], \qquad \mu_Y^{(j)} := \mathbb{E}[Y_i^{(j)}], \qquad v_X^{(j)} := Var(X_i^{(j)}), \qquad v_Y^{(j)} := Var(Y_i^{(j)}) \quad \forall i, j.$$

For any independent random vector X and scalar Y with finite second moments, independence gives

$$\begin{aligned} Var(XY) &= \mathbb{E}\big[\|XY\|^2\big] - \|\mathbb{E}[XY]\|^2 \\ &= \mathbb{E}[Y^2]\,\mathbb{E}\big[\|X\|^2\big] - (\mathbb{E}[Y])^2\|\mathbb{E}[X]\|^2 \\ &= Var(X)Var(Y) + Var(X)(\mathbb{E}[Y])^2 + Var(Y)\|\mathbb{E}[X]\|^2. \end{aligned} \tag{A.1}$$

Since the vectors $X_i^{(j)} Y_i^{(j)}$ are i.i.d., it follows from (A.1) and linearity of variance for independent random variables,

$$Var\left(\frac{1}{n}\sum_{i=1}^n X_i^{(j)} Y_i^{(j)}\right) = \frac{v_X^{(j)} v_Y^{(j)} + v_X^{(j)}(\mu_Y^{(j)})^2 + v_Y^{(j)}\|\mu_X^{(j)}\|^2}{n}.$$

Now, let

$$\bar{X}^{(j)} := \frac{1}{n}\sum_{i=1}^n X_i^{(j)}, \qquad \bar{Y}^{(j)} := \frac{1}{n}\sum_{i=1}^n Y_i^{(j)}.$$

The independence of the two samples implies that $\bar{X}^{(j)}$ and $\bar{Y}^{(j)}$ are independent. Moreover,

$$\mathbb{E}[\bar{X}^{(j)}] = \mu_X^{(j)}, \qquad \mathbb{E}[\bar{Y}^{(j)}] = \mu_Y^{(j)}, \qquad Var(\bar{X}^{(j)}) = \frac{v_X^{(j)}}{n}, \qquad Var(\bar{Y}^{(j)}) = \frac{v_Y^{(j)}}{n}.$$

Applying the product-variance identity from (A.1) yields

$$Var(\bar{X}^{(j)}\bar{Y}^{(j)}) = \frac{v_X^{(j)} v_Y^{(j)}}{n^2} + \frac{v_X^{(j)}(\mu_Y^{(j)})^2}{n} + \frac{v_Y^{(j)}\|\mu_X^{(j)}\|^2}{n}.$$

Consequently,

$$Var\left(\frac{1}{n}\sum_{i=1}^n X_i^{(j)} Y_i^{(j)}\right) - Var(\bar{X}^{(j)}\bar{Y}^{(j)}) = \frac{n-1}{n^2} v_X^{(j)} v_Y^{(j)} \geq 0,$$

where the inequality follows from $n \geq 2$. □

Lemma A.4. *Let $\epsilon_1, \epsilon_2, \epsilon_3 \in (0, 1)$ and let $r, r_1, r_2, r_3 > 0$ be constants. Suppose*

$$\epsilon_1 = \Theta\big(\epsilon_2 \log\big(\epsilon_1^{-r_1}\epsilon_2^{-r_2}\epsilon_3^{-r_3}\big)\big).$$

Furthermore, if there exists a constant $q > 0$ such that $\log(1/\epsilon_1) \geq \log^q(1/\epsilon_3)$. Then

$$\tilde{\mathcal{O}}(\epsilon_1^{-r}) = \tilde{\mathcal{O}}(\epsilon_2^{-r}).$$

Proof. Set $a = \log(1/\epsilon_1)$, $b = \log(1/\epsilon_2)$, $d = \log(1/\epsilon_3)$, and $r_q = \max\{1, 1/q\}$. Since $d \le a^{1/q}$, the assumed relation gives

$$e^{b-a} = \Theta(r_1 a + r_2 b + r_3 d) = \mathcal{O}(a^{r_q} + b).$$

Taking logarithms, for sufficiently large a,

$$b \le a + r_q \log a + \log(1 + b) + \mathcal{O}(1).$$

Using $\log(1 + b) \le b/2 + \mathcal{O}(1)$ yields $b = \mathcal{O}(a)$. Consequently,

$$\frac{\epsilon_1}{\epsilon_2} = \Theta(r_1 a + r_2 b + r_3 d) \le \mathcal{O}(a^{r_q}). \tag{A.2}$$

The assumption $\epsilon_1/\epsilon_2 = \Theta(r_1 a + r_2 b + r_3 d)$ and the fact that for some constant $t' > 0$, $t' r_1 a \le t'(r_1 a + r_2 b + r_3 d)$ means

$$\Omega(a) = \epsilon_1/\epsilon_2. \tag{A.3}$$

Combining (A.2) and (A.3) means there exist constants $t_{\min}, t_{\max} > 0$ such that $t_{\min} a \le \frac{\epsilon_1}{\epsilon_2} \le t_{\max} a^{r_q}$. As $\epsilon_1 \to 0$, we have $a = \log(1/\epsilon_1) \to \infty$, so eventually $t_{\min} a \ge 1$. Raising to the power $r > 0$ gives

$$1 \le \tfrac{\epsilon_1^r}{\epsilon_2^r} \le t_{\max}^r a^{r_q r}.$$

Thus, for sufficiently small ϵ_1, multiplying by ϵ_1^{-r} and substituting the value of a gives,

$$\epsilon_1^{-r} \le \epsilon_2^{-r} = \mathcal{O}\left(\epsilon_1^{-r} [\log(1/\epsilon_1)]^{r_q r}\right).$$

Absorbing polylogarithmic factors proves $\tilde{\mathcal{O}}(\epsilon_1^{-r}) = \tilde{\mathcal{O}}(\epsilon_2^{-r})$. □

A.1 Proof of Lemma 3.1

Proof. From Assumption 2.1 and Assumption 3.3, we have

$$\|\nabla F(x, \xi)\| \le \|\nabla f(x)\| + \|\nabla F(x, \xi) - \nabla f(x)\| \le \kappa_{\nabla f} + \sigma_{\nabla f}, \quad \mathcal{P}_\xi\text{-a.s.}$$

Similarly, we can prove

$$\begin{aligned}
&\|C(x, \zeta)\| \le \|c(x)\| + \|C(x, \zeta) - c(x)\| \le \kappa_c + \sigma_c, \quad \mathcal{P}_\xi\text{-a.s.},\\
&\|\nabla C(x, \zeta)\| \le \|\nabla c(x)\| + \|\nabla C(x, \zeta) - \nabla c(x)\| \le \kappa_{\nabla c} + \sigma_{\nabla c}, \quad \mathcal{P}_\xi\text{-a.s., and}\\
&\|\nabla^2 C(x, \zeta)\| \le \|\nabla^2 c(x)\| + \|\nabla^2 C(x, \zeta) - \nabla^2 c(x)\| \le \kappa_{\nabla^2 c} + \sigma_{\nabla^2 c}, \quad \mathcal{P}_\xi\text{-a.s.}
\end{aligned}$$

By the mean value theorem, along the line segment between any two points $x, y \in \mathcal{X}$ yields some z such that $\mathcal{P}_\zeta$-almost surely,

$$|C^{(j)}(x) - C^{(j)}(y)| = |\nabla C^{(j)}(z)^T (x - y)| \le \|\nabla C^{(j)}(z)\| \|x - y\| \le (\kappa_{\nabla c} + \sigma_{\nabla c}) \|x - y\|.$$

□

A.2 Proof of Lemma 4.1

Proof. Iterate x_k satisfying probabilistic conditions (2.5) with probability at least $p_k = 5/8$ is a direct consequence of [6, Theorem 5]. First, we establish the conditions required for 1) the stochastic maps of $\mathcal{L}_{\alpha_k}(\cdot, \lambda_k)$ in (3.1) and (3.2) to belong to [6, Oracle $(\mathsf{O}_F^2, P_z) \in \overline{\mathcal{O}}(F, \sigma_1, \bar{\sigma}_2)$ for $F \in \mathcal{F}_2(\Delta, L_1, L_2)$], and 2) for convergence of Algorithm 2, as defined in [6, Theorem 5].

Condition 1: From (3.13) in Lemma 3.6, it holds that $\mathcal{L}_{\alpha_k}(x_k, \lambda_k) - \mathcal{L}_{\alpha_k}(x_k^*, \lambda_k) = \Delta_k \leq \Delta_k^* < \infty$ for all $k \in \mathbb{N}_0$, where $x_k^* \in \min_{x \in \mathcal{X}} \mathcal{L}_{\alpha_k}(x, \lambda_k)$. This corresponds to the assumption provided in [6, Section 2 (Page 6)].

Condition 2: From Lemma 3.2, it holds that at each iteration $k \in \mathbb{N}_0$, the function $\nabla \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L_{\mathcal{L},k}^{(g)}$-Lipschitz continuous on $\mathcal{X}$ almost surely, i.e. $\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L_{\mathcal{L},k}^{(g)} \|x - y\|$, for all $k \in \mathbb{N}_0$, for all $x, y \in \mathcal{X}, \mathcal{P}_\theta$-almost surely By Jensen's inequality, Lemma 3.2 directly implies the required condition $\|\nabla \mathcal{L}_{\alpha_k}(x, \lambda_k) - \nabla \mathcal{L}_{\alpha_k}(y, \lambda_k)\| \leq L_{\mathcal{L},k}^{(g)} \|x - y\|$ for all $k \in \mathbb{N}_0$. This corresponds to the assumption provided in [6, Section 2 (Page 6)].

Condition 3: From Lemma 3.3, it holds that at each iteration $k \in \mathbb{N}_0$, the function $\nabla^2 \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L_{\mathcal{L},k}^{(H)}$-Lipschitz continuous on $\mathcal{X}$ almost surely, i.e. $\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L_{\mathcal{L},k}^{(H)} \|x - y\|$, for all $k \in \mathbb{N}_0$ for all $x, y \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. By Jensen's inequality, Lemma 3.3 directly implies the required condition $\|\nabla^2 \mathcal{L}_{\alpha_k}(x, \lambda_k) - \nabla^2 \mathcal{L}_{\alpha_k}(y, \lambda_k)\| \leq L_{\mathcal{L},k}^{(H)} \|x - y\|$ for all $k \in \mathbb{N}_0$. This corresponds to the assumption provided in [6, Section 2 (Page 6)].

Condition 4: From Lemma 3.4, it holds that at each iteration $k \in \mathbb{N}_0$, the stochastic gradient estimator $\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ satisfy uniform error bounds $\mathcal{P}_\theta$-almost surely, i.e. $\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\|^2 \leq (\sigma_{\mathcal{L},k}^{(g)})^2$ for all $k \in \mathbb{N}_0$, for all $x \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. By Jensen's inequality, Lemma 3.4 directly implies the required condition $\mathbb{E}_\theta[\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\|^2] \leq (\sigma_{\mathcal{L},k}^{(g)})^2$ for all $k \in \mathbb{N}_0$. This corresponds to the assumption provided in [6, Section 2 (Page 6)].

Condition 5: From Lemma 3.5, it holds that at each iteration $k \in \mathbb{N}_0$, the stochastic Hessian estimator $\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ satisfy uniform error bounds $\mathcal{P}_\theta$-almost surely, i.e. $\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \mathcal{L}_{\alpha_k}(x, \lambda_k)\|^2 \leq (\sigma_{\mathcal{L},k}^{(H)})^2$ for all $k \in \mathbb{N}_0$, for all $x \in \mathcal{X}, \mathcal{P}_\theta$-a.s.. This corresponds to the assumption provided in [6, Section 4.1 (Page 11)].

Condition 6: Finally, $(\epsilon_k^{(g)}, \epsilon_k^{(H)})$ satisfies the constraints on [6, (ϵ, γ)] as required in [6, Section 2 (Page 6)].

By **Conditions 1-6**, [6, Theorem 5] holds. Therefore, x_k satisfies probabilistic conditions (2.5). Furthermore, this establishes the work complexity by [6, Theorem 5] as per Definition 3.2 to be

$$\mathcal{W}_{SG,k}^{(p)} = \tilde{\mathcal{O}}\left(\frac{\Delta_k \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(g)})^3} + \frac{\Delta_k L_{\mathcal{L},k}^{(H)} \sigma_{\mathcal{L},k}^{(g)} \sigma_{\mathcal{L},k}^{(H)}}{(\epsilon_k^{(H)})^2 (\epsilon_k^{(g)})^2} + \frac{\Delta_k (L_{\mathcal{L},k}^{(H)})^2 (\sigma_{\mathcal{L},k}^{(H)} + L_{\mathcal{L},k}^{(g)})^2}{(\epsilon_k^{(H)})^5} + \frac{\Delta_k L_{\mathcal{L},k}^{(g)}}{(\epsilon_k^{(g)})^2} \right).$$

By the argument in Lemma 3.7, this is the work complexity of the primal solver at each iteration $k \in \mathbb{N}_0$. □

A.3 Proof of Lemma 4.3

Proof. Iterate x_k satisfying probabilistic conditions (2.5) with probability at least $p_k = 2/3$ is a direct consequence of [3, Theorem 2]. First, we establish the conditions required for 1) the stochastic maps of $\mathcal{L}_{\alpha_k}(\cdot, \lambda_k)$ in (3.1) and (3.2) to belong to [3, function class f_i], and 2) for convergence of Algorithm 4, as defined in [3, Theorem 2].

Condition 1: From (3.13) in Lemma 3.6, it holds that $\mathcal{L}_{\alpha_k}(x_k, \lambda_k) - \mathcal{L}_{\alpha_k}(x_k^*, \lambda_k) = \Delta_k \leq \Delta_k^* < \infty$ for all $k \in \mathbb{N}_0$, where $x_k^* \in \min_{x \in \mathcal{X}} \mathcal{L}_{\alpha_k}(x, \lambda_k)$. This corresponds to the assumption stated in [3, Theorem 2].

Condition 2: From Lemma 3.2, it holds that at each iteration $k \in \mathbb{N}_0$, the function $\nabla \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L_{\mathcal{L},k}^{(g)}$-Lipschitz continuous on $\mathcal{X}$ almost surely, i.e. $\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L_{\mathcal{L},k}^{(g)} \|x - y\|$, for all $k \in \mathbb{N}_0$, for all $x, y \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. This corresponds to [3, Assumption (A2)].

Condition 3: From Lemma 3.3, it holds that at each iteration $k \in \mathbb{N}_0$, the function $\nabla^2 \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L^{(H)}_{\mathcal{L},k}$-Lipschitz continuous on $\mathcal{X}$ almost surely, i.e. $\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L^{(H)}_{\mathcal{L},k}\|x - y\|$, for all $k \in \mathbb{N}_0$, for all $x, y \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. Lemma 3.3 directly implies the required condition $\|\nabla^2 \mathcal{L}_{\alpha_k}(x, \lambda_k) - \nabla^2 \mathcal{L}_{\alpha_k}(y, \lambda_k)\| \leq L^{(H)}_{\mathcal{L},k}\|x - y\|$ for all $k \in \mathbb{N}_0$. This corresponds to [3, Assumption (A4)].

Condition 4: From Lemma 3.4, it holds that at each iteration $k \in \mathbb{N}_0$, the stochastic gradient estimator $\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ satisfy uniform error bounds $\mathcal{P}_\theta$-almost surely, i.e. $\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\|^2 \leq (\sigma^{(g)}_{\mathcal{L},k})^2$, for all $k \in \mathbb{N}_0$, for all $x \in \mathcal{X}, \mathcal{P}_\theta$-a.s.. Lemma 3.4 directly implies the required condition $\mathbb{E}_\theta[\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\|^2] \leq (\sigma^{(g)}_{\mathcal{L},k})^2$, for all $k \in \mathbb{N}_0$. This corresponds to [3, Assumption (A1)].

Condition 5: Finally, $(\epsilon^{(g)}_k, \epsilon^{(H)}_k)$ satisfies the constraints on $[3, (\varepsilon, 3\delta)]$ in [3, Theorem 2].

By **Conditions 1-5**, [3, Theorem 2] holds. Therefore, x_k satisfies probabilistic conditions (2.5). Furthermore, this establishes the work complexity by [3, Theorem 2] as per Definition 3.2 to be

$$\mathcal{W}^{(p)}_{Nat2,k} = \tilde{\mathcal{O}}\left(\frac{(\sigma^{(g)}_{\mathcal{L},k})^4}{(\epsilon^{(g)}_k)^2} + \frac{(L^{(H)}_{\mathcal{L},k})^2 (L^{(g)}_{\mathcal{L},k})^2 \Delta_k}{(\epsilon^{(H)}_k)^5} + \frac{L^{(H)}_{\mathcal{L},k} \Delta_k (L^{(g)}_{\mathcal{L},k})^2}{\epsilon^{(g)}_k (\epsilon^{(H)}_k)^3} + \frac{L^{(H)}_{\mathcal{L},k} \Delta_k (\sigma^{(g)}_{\mathcal{L},k})^2}{(\epsilon^{(g)}_k)^3 \epsilon^{(H)}_k} + \frac{(L^{(g)}_{\mathcal{L},k})^3 \Delta_k}{\epsilon^{(g)}_k (\epsilon^{(H)}_k)^2 \sigma^{(g)}_{\mathcal{L},k}}\right).$$

By the argument in Lemma 3.7, this is the work complexity of the primal solver at each iteration $k \in \mathbb{N}_0$. □

A.4 Proof of Lemma 4.5

Proof. Iterate x_k satisfying probabilistic conditions (2.5) with probability at least $p_k = 1/2$ is a direct consequence of [24, Theorem 6]. First, we establish the conditions required for 1) the stochastic maps of $\mathcal{L}_{\alpha_k}(\cdot, \lambda_k)$ in (3.1) and (3.2) to belong to [24, function class f_i], and 2) for convergence of Algorithm 6, as defined in [24, Theorem 6].

Condition 1: From (3.13) in Lemma 3.6, it holds that $\mathcal{L}_{\alpha_k}(x_k, \lambda_k) - \mathcal{L}_{\alpha_k}(x^*_k, \lambda_k) = \Delta_k \leq \Delta^*_k < \infty$ for all $k \in \mathbb{N}_0$, where $x^*_k \in \min_{x \in \mathcal{X}} \mathcal{L}_{\alpha_k}(x, \lambda_k)$. This corresponds to [24, Assumption 1(i)].

Condition 2: From Lemma 3.2, it holds that at each iteration $k \in \mathbb{N}_0$, the function $\nabla \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L^{(g)}_{\mathcal{L},k}$-Lipschitz continuous on $\mathcal{X}$ almost surely, i.e. $\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L^{(g)}_{\mathcal{L},k}\|x - y\|$, for all $k \in \mathbb{N}_0$, for all $x, y \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. This corresponds to [24, Assumption 2(ii')].

Condition 3: From Lemma 3.3, it holds that at each iteration $k \in \mathbb{N}_0$, the function $\nabla^2 \bar{\mathcal{L}}(x, \lambda_k; \theta)$ is $L^{(H)}_{\mathcal{L},k}$-Lipschitz continuous on $\mathcal{X}$ almost surely, i.e. $\|\nabla^2 \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla^2 \bar{\mathcal{L}}_{\alpha_k}(y, \lambda_k; \theta)\| \leq L^{(H)}_{\mathcal{L},k}\|x - y\|$, for all $k \in \mathbb{N}_0$, for all $x, y \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. This corresponds to [24, Assumption 3].

Condition 4: From Lemma 3.4, it holds that at each iteration $k \in \mathbb{N}_0$, the stochastic gradient estimator $\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta)$ satisfy uniform error bounds $\mathcal{P}_\theta$-almost surely, i.e. $\|\nabla \bar{\mathcal{L}}_{\alpha_k}(x, \lambda_k; \theta) - \nabla \mathcal{L}_{\alpha_k}(x, \lambda_k)\|^2 \leq (\sigma^{(g)}_{\mathcal{L},k})^2$, for all $k \in \mathbb{N}_0$, for all $x \in \mathcal{X}, \mathcal{P}_\theta$-almost surely. This corresponds to [24, Assumption 2(iii')].

Condition 5: Finally, primal tolerance parameter $\bar{\epsilon}^{(g)}_k$ satisfies the condition on $[24, 10\epsilon]$ and subproblem tolerance $\epsilon^{(H)}_k$ satisfies the constraints on $[24, 3\delta]$ in [24, Theorem 6].

By **Conditions 1-5**, [24, Theorem 6] holds. Therefore, x_k satisfies probabilistic conditions (2.5) Furthermore, this establishes the work complexity by [24, Theorem 6] as per Definition 3.2 to be

$$\mathcal{W}^{(p)}_{SPDR,k} = \tilde{\mathcal{O}}\left(\frac{\Delta_k L^{(g)}_{\mathcal{L},k} \sigma^{(g)}_{\mathcal{L},k}}{(\bar{\epsilon}^{(g)}_k)^3} + \frac{\Delta_k L^{(g)}_{\mathcal{L},k} L^{(H)}_{\mathcal{L},k} \sigma^{(g)}_{\mathcal{L},k}}{(\bar{\epsilon}^{(g)}_k)^2 (\epsilon^{(H)}_k)^2} + \frac{\Delta_k (L^{(g)}_{\mathcal{L},k})^2 (L^{(H)}_{\mathcal{L},k})^2}{(\epsilon^{(H)}_k)^5} + \frac{\Delta_k (L^{(g)}_{\mathcal{L},k})^2 L^{(H)}_{\mathcal{L},k}}{\bar{\epsilon}^{(g)}_k (\epsilon^{(H)}_k)^3} + \frac{(\sigma^{(g)}_{\mathcal{L},k})^2}{(\bar{\epsilon}^{(g)}_k)^2} + \frac{L^{(g)}_{\mathcal{L},k} \sigma^{(g)}_{\mathcal{L},k} \epsilon^{(H)}_k}{(\bar{\epsilon}^{(g)}_k)^2 L^{(H)}_{\mathcal{L},k}}\right)$$

By Lemma 3.7, this is the work complexity of the primal solver at each iteration $k \in \mathbb{N}_0$. □

A.5 Additional Remarks

Remark A.1. *We summarize several key properties of the projection operator onto tangent cones that may be useful in convergence and complexity analysis. Throughout this discussion, let $\mathcal{X} \subset \mathbb{R}^d$ be a nonempty, closed, and convex set. The projection operator $\mathrm{proj}_{T_{\mathcal{X}}(x)}(\cdot)$ satisfies the following properties:*

- ***Closedness and Convexity:*** *For any $x \in \mathcal{X}$, the tangent cone $T_{\mathcal{X}}(x)$ is a closed convex cone. Consequently, $\mathrm{proj}_{T_{\mathcal{X}}(x)}$ inherits all standard properties of projection onto closed convex sets, including non-expansiveness.*
- ***Moreau Decomposition and Duality:*** *The tangent cone $T_{\mathcal{X}}(x)$ and the normal cone $N_{\mathcal{X}}(x)$ are polar to each other, i.e., $N_{\mathcal{X}}(x) = (T_{\mathcal{X}}(x))^\circ$. By Moreau's Decomposition Theorem [47, 53], any vector $y \in \mathbb{R}^d$ satisfies $y = \mathrm{proj}_{T_{\mathcal{X}}(x)}(y) + \mathrm{proj}_{N_{\mathcal{X}}(x)}(y)$ with $\langle \mathrm{proj}_{T_{\mathcal{X}}(x)}(y), \mathrm{proj}_{N_{\mathcal{X}}(x)}(y) \rangle = 0$. This yields the dual distance identity: $\|\mathrm{proj}_{T_{\mathcal{X}}(x)}(y)\| = \mathrm{dist}(y, N_{\mathcal{X}}(x))$, for all $y \in \mathbb{R}^d$.*
- ***Norm Bound:*** *Since $0 \in T_{\mathcal{X}}(x)$ for all $x \in \mathcal{X}$, applying Lemma A.1 directly yields a uniform norm bound: $\|\mathrm{proj}_{T_{\mathcal{X}}(x)}(v)\| \le \|v\|$, for all $v \in \mathbb{R}^d$.*
- ***Positive Homogeneity:*** *As a consequence of $T_{\mathcal{X}}(x)$ being a cone, the projection operator is positively homogeneous of degree one: $\mathrm{proj}_{T_{\mathcal{X}}(x)}(\lambda v) = \lambda \mathrm{proj}_{T_{\mathcal{X}}(x)}(v), \quad \forall \lambda > 0, v \in \mathbb{R}^d$.*

Remark A.2. *We make the following remarks regarding the analysis of inequality constraints:*

- *As stated earlier, to convert an inequality constraint into an equality constraint, a non-negative slack variable is added to the constraint with the non-negativity constraints embedded into the projection set $\mathcal{X}$. We note that the first-order stationarity error term (1.4)(i), encodes dual feasibility and complementary slackness, in addition to the first-order stationarity error.*

  Consider $x' := (x, s) \in \mathcal{X} := \mathbb{R}^d \times \mathbb{R}^m_+$. The constraint function is updated to $c(x) + s = 0$. Now, the standard first-order stationarity conditions for the problem

$$\begin{aligned} \min_{(x,s)\in\mathcal{X}} \quad & f(x) \\ \text{s.t.} \quad & c(x) + s = 0, \\ & s \ge 0, \end{aligned}$$

  are

$$\begin{aligned} \nabla f(x) + \nabla c(x)^\top \lambda &= 0, \\ c(x) + s &= 0, \\ \lambda &\ge 0, \\ \lambda \cdot s &= 0. \end{aligned}$$

  Under the projection settings, the optimality conditions are

$$\begin{aligned} \|\mathrm{proj}_{T_{\mathcal{X}}(x')}(-\nabla f(x) - \nabla c(x)^\top \lambda\| &= 0, \\ c(x) + s &= 0. \end{aligned}$$

  The first condition corresponds to

$$\|\mathrm{proj}_{T_{\mathcal{X}}((x,s))}(-\nabla f(x) - \nabla c(x)^\top \lambda\| := \sqrt{\|\nabla f(x) + \nabla c(x)^\top \lambda\|^2 + \sum_{j=1}^{m} \|\mathrm{proj}_{T_{\mathbb{R}_+}(s^{(j)})}(-\lambda^{(j)})\|^2}.$$

The first term within the square root corresponds to the first-order stationarity term. The second term within the square root can be simplified as

$$\text{proj}_{T_{\mathbb{R}_+}(s^{(j)})}(-\lambda^{(j)}) = \begin{Bmatrix} -\lambda^{(j)}, & s^{(j)} > 0 \\ \max(-\lambda^{(j)}, 0), & s^{(j)} = 0 \end{Bmatrix}.$$

Setting the norm of this term to zero exactly corresponds to the dual feasibility and complementary slackness condition.

- *We found no algorithms for checking the minimum eigenvalue corresponding to a critical cone of vectors as required. Thus, instead of ensuring the minimum eigenvalue over the critical cone is greater than $\epsilon^{(H)}$, we ensure $\sigma_{\min}(\nabla^2_{(x,s)(x,s)}\mathcal{L}_\alpha((x,s),\lambda)) \geq -\epsilon^{(H)}$. This allows us to use Oja's and Neon2 algorithm to search for negative curvature directions. It is not necessary to ensure this, as the negative curvature direction could correspond to a direction that is not within the critical cone. The iterate could have already converged to an $(\epsilon^{(g)}, \epsilon^{(H)})$-accurate point earlier, but this is a sufficient condition for second-order stationarity. Our empirical observations, within the considered problems, found $\sigma_{\min}(\nabla^2_{(x,s)(x,s)}\mathcal{L}_\alpha) \approx \sigma_{\min}(\nabla^2_{xx}\mathcal{L}_\alpha)$. We will consider developing algorithms to search for eigenvectors corresponding to a closed and convex set $\mathcal{X} \subset \mathbb{R}^n$ in the future.*