SOLVING CONVEX SMOOTH FUNCTION CONSTRAINED OPTIMIZATION IS ALMOST AS EASY AS UNCONSTRAINED OPTIMIZATION

ZHENG ZHANG* and GUANGLUI LAN†

Abstract. Large-scale function-constrained optimization problems arise in many applications. The first-order method is an efficient numerical solution method due to its low per-iteration cost. However, one fundamental question remains less understood: the number of gradient evaluations required to find an $\epsilon$-optimal solution, i.e., the oracle complexity. A gap exists between the oracle complexities of the algorithms in the literature and the optimal oracle complexities for problems without function constraints. Moreover, the algorithms with better oracle complexities tend to require strong computation oracles like a solver for the quadratic program, or have strong assumptions like the number of constraints being small, which limits their application under the large-scale setting. In this paper, we close the gap by proposing two efficient algorithms. The simple Accelerated Constrained Gradient Descent (ACGD) method, which only adds some linear constraint to the descent step of the AGD method, achieves the same order of oracle complexity as the optimal complexity for the problems without function constraints. However, one drawback is the requirement for a strong computation oracle, i.e., a QP solver. We relax the strong computation oracle assumption by proposing the ACGD-S method which carries out only basic matrix-vector operations in each iteration. The ACGD-S method has the same optimal oracle complexity as the ACGD method, and its computation complexity, measured by the number of matrix-vector multiplications, is also optimal. Matching lower complexity bounds are also constructed to show the oracle complexity of both the ACGD and ACGD-S method to be unimprovable with respect to all problem parameters under certain optimality regimes for a general class of first-order methods.

Keywords: constrained optimization, nonlinear optimization, first-order algorithm, convex optimization, lower complexity.

1. Introduction. Consider convex smooth function-constrained optimization of the form

$$
\min_{x\in\mathbb{R}^n} f(x) + u(x)
\quad \text{s.t. } g(x) \leq 0,
$$

where both $f : \mathbb{R}^n \to \mathbb{R}$ and $g := [g_1, g_2, \ldots, g_m]^{\top} : \mathbb{R}^n \to \mathbb{R}^m$ are convex and differentiable with Lipschitz continuous gradients, and the domain $X$ is convex and closed. The regularization function $u(x)$ is assumed to be both simple [13] and $\alpha$-strongly convex for some $\alpha \geq 0$. Such a problem finds a wide range of applications in, for example, the Neyman-Pearson classification problem [20], the fairness constrained classification [23], and the risk-constrained portfolio optimization [4]. Since the dimensions $n$ and $m$ are large in many applications, we focus on first-order methods to find an approximation solution. Specifically, $(f, g)$ is assumed to be accessible only via a black-box first-order oracle, which returns $(f(x), g(x); \nabla f(x), \nabla g(x))$ when queried at some $x \in \mathbb{R}^n$, and the goal is to find an $(\epsilon; \epsilon/c)$-optimal solution:

$$
f(x^N) - f(x^*) \leq \epsilon \text{ and } \|g(x^N)\|_2 \leq \epsilon/c,
$$

where the scaling constant $c \geq 1$ represents the modeler’s preference for feasibility violation relative to sub-optimality.

This paper intends to develop fast methods and to understand the requisite computation cost. To ensure practical efficiency, special attention is paid to simple methods involving only oracle evaluations, projections onto $X$, and basic vector operations like matrix-vector multiplication. Since complicated constraints can be model by $g(x) \leq 0$, the projection onto $X$ is assumed to be efficiently computable. To study the computation cost, it is useful to consider two kinds of cost: a) the oracle complexity, i.e., the total number of queries to the first order oracle, and b) the computation complexity, i.e., the total number of matrix-vector multiplications (each costing at most $O(mn)$ FLOPs) and projections onto $X$. During implementation, these complexities translate to different computation burdens, so the exact problem instance determines which one dominates. For instance, if the constraint $g$ is complicated, say a large finite-sum function in the fairness constrained problem, the gradient evaluation of $g$ could be the bottleneck. On the other hand, if both $n$ and $m$ are large and $g$ is an affine function, the matrix-vector multiplication might be the bottleneck. Thus the ideal optimization method should excel in both directions.

*H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332. (email: jimmy_zhang@gatech.edu)
†H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332. (email: george.lan@isye.gatech.edu)
Table 1.1: Ideal Complexity for Solving (1.1)

| Case                  | Oracle Complexity | Computation Complexity |
|-----------------------|-------------------|------------------------|
| Non-strongly Convex $\alpha = 0$ | $O(1/\sqrt{\epsilon})$ | $O(1/\epsilon)$ |
| Strongly Convex $\alpha > 0$   | $O(\log(1/\epsilon))$ | $O(1/\sqrt{\epsilon})$ |

Since convex optimization has been studied extensively, e.g. [16], we can conjecture the best possible complexities for (1.1) by reducing it to simpler cases for which the optimal result is available. Recall from [14, 15], the number of oracle evaluations required to find an $\epsilon$-optimal solution to a smooth problem without function constraints is $\Theta(1/\sqrt{\epsilon})$ in general, and $\Theta(\sqrt{\kappa} \log(1/\epsilon))$ if the problem is also strongly convex, where $\kappa$ denotes the condition number in the latter case. One way to reduce (1.1) to such a case is to consider

$$
\min_{x \in X} \max_{\lambda \in \mathbb{R}^+_+} \{ \mathcal{L}(x; \lambda) := f(x) + u(x) + \lambda^T(g(x)) \},
$$

(1.3)

where $\lambda$ denotes the Lagrange multipliers to the inequality constraints. If $\lambda$ is fixed to the optimal dual multiplier $\lambda^*$, $\mathcal{L}(x; \lambda^*)$ is a smooth function without any function constraints, and one of its minimizer is the solution to (1.1). So, in the best case, finding an $(\epsilon; \epsilon/c)$ solution to (1.1) should have the same oracle complexity as approximately solving $\mathcal{L}(x; \lambda^*)$, i.e., $O(1/\sqrt{\epsilon})$ for the non-strongly case ($\alpha = 0$) and $O(\log(1/\epsilon))$ for the strongly convex case ($\alpha > 0$). On the other hand, Xu and Ouyang constructed large-scale linearly constrained quadratic programs in [19] to show the tight computation complexity for the linearly constrained smooth problem is $\Theta(1/\epsilon)$ if $\alpha = 0$, and $\Theta(1/\sqrt{\epsilon})$ if $\alpha > 0$. Since nonlinear constraint function includes affine function as a special case, i.e. $g(x) = Ax - b$, the computation complexity for (1.1) should be at least as expensive as the affine case. These conjectured complexities, summarized in Table 1.1, lead naturally to the research question:

Can we solve function-constrained problems with the same oracle complexities as those without function constraints, while maintaining the same computation complexities as solving linearly constrained problems?

However, despite much research effort on the subject from different directions, the question remains open. Broadly speaking, the current results can be grouped by the reformulation used (see the summary in Table 1.2). The methods based on the Lagrangian formulation (1.3) are usually simple to implement and have a low per-iteration cost. For instance, the ConEx method in [3] and the APD method in [5] are single-loop algorithms with the optimal computation complexities, but their oracle complexities are worse than the ideal ones by an order of magnitude. The current best method based on the augmented Lagrangian reformulation [21], a three loop algorithm built on the inexact augmented Lagrangian method, the ellipsoid method and the accelerated gradient descent (AGD) method, almost matches the ideal oracle complexities in Table 1.1. However, the overall algorithm in [21] scales poorly with the number of constraints because the ellipsoid method is utilized in the search for dual multiplier. In the numerical experiment [21], the method is advantageous to the APD method only when $m \leq 5$.

Another line of research [12, 17], called the level-set method, reformulates (1.1) to a root finding problem, which involves a certain mini-max problem as sub-problem. These methods typically require all constraint functions to share the same strong convexity modulus as the objective function, which is not satisfied in our set-up. Under such a uniform strong-convexity assumption, the method proposed by Nesterov in Section 2.3.5 of [17] can achieve an $O(\log(1/\epsilon))$ oracle complexity when $\alpha > 0$. However, one drawback is the need for the exact solution to a quadratic program (QP) in each iteration, and the need for the exact solution to a quadratic-constrained quadratic program (QCQP) from time to time, which could be computationally demanding under the large-scale setting. The expensive computation oracle assumption is relaxed to basic matrix-vector operations in [12], but the oracle complexity also becomes worse by an order of magnitude. To sum up, there exists two major deficiencies: current methods fail to match the general $O(\log(1/\epsilon))$ oracle complexity for the strongly convex case, and, for methods that are close, there always are some strong
In the strongly convex case with \( \alpha = 0 \), the lower bound shows the upper oracle complexity bound to be unimprovable for all problem parameters. In the non-strongly case with \( \alpha = 0 \), the lower bound shows the upper oracle complexity bound to be tight up to a factor of \( O(\frac{1}{\sqrt{c}}) \), i.e., the ACGD.

\[ \text{ACGD} \]
method has tight oracle complexity with respect to all problem parameters if $c \geq \|\lambda^*\|$. If $\lambda^*$ is interpreted as the shadow price of constraint violation around $x^*$, such a choice of $c$ ensures that the increase in objective value incurred by moving the solution infeasible solution $\bar{x}$ to feasibility is roughly $O(\epsilon)$.

For the more challenging task of developing a single method with both the ideal computation complexity and the ideal oracle complexity, we use the sliding technique $[7, 11]$ to extend the ACGD method to the ACGD with Sliding (ACGD-S) method. Given a linearization of $\mathcal{L}(\bar{x}' ; \lambda)$, ACGD-S solves the bilinear saddle point problem in (1.5) by iteratively performing $\lambda$-prox mapping and $x$-prox mapping (see (1.7)). Particularly, given the $t$th oracle evaluation, i.e., $\{\nabla g(x^t_i), \nabla f(x^t); f(x^t_i), g(x^t_i)\}$, the $x$-prox mapping and $\lambda$-prox mapping are repeated only $O(t)$ times if $\alpha = 0$, and $O(\sqrt{\theta}t)$ times for some $\theta > 1$ if $\alpha > 0$. They allow the ACGD-S method to achieve ideal complexities in Table 1.1, i.e., the same oracle complexity as the ACGD method and the same computation complexity as the ConEx method $[8]$. The intricate step-size choice to achieve both the optimal $O(\log(1/\epsilon))$ oracle evaluations (outer loops) and the optimal $O(1/\sqrt{t})$ matrix-vector multiplications (inner iterations) appears to be new to the sliding technique $[7, 11, 12]$, so it is of independent interest.

One potential concern for implementing ACGD and ACGD-S is that their stepsize selection depends on the assumption $\lambda_1(\bar{x}^t, \lambda^t) \approx 0$. This assumption implies that any $(\bar{x}, \lambda)$ with $\bar{x} \in X^*$ and $\lambda \in \Lambda^*$ is a KKT point to the Lagrangian defined in (1.3) throughout this paper.

Notations & Assumptions. The following assumptions and notations will be used throughout the paper.

- The set of optimal solutions to (1.1), $X^*$, is nonempty, and $x^*$ is an arbitrary optimal solution. $f_*$ denotes the optimal objective, $f(x^*)$. $R_0$ denotes an estimate of the distance from the initial point to $x^*$, i.e., $R_0 \geq \|x^0 - x^*\|$.
- $U_h$ denotes the Bregman distance function (see [1]) generated by a convex function $h$, i.e., $U_h(x^t_i; \pi) := h(\pi) - h(\bar{x}^t_i) - \langle h'(\bar{x}^t_i), \pi - \bar{x}^t_i \rangle$ where $g'(\bar{x}^t_i)$ is some fixed subgradient in $\partial h(\bar{x}^t_i)$. If $g$ is vector-valued, i.e., $g(\nu) := [g_1(\nu), g_2(\nu), \ldots, g_m(\nu)]$, $U_g$ is vector-valued with its $i$th component being the Bregman distance function generated by $g_i$, namely, $U_{g_i}$.

- We refer to the following computation as either a prox mapping or a projection:

$$
\bar{w} \leftarrow \arg \min_{w \in \mathcal{W}} \langle y, w \rangle + h(w) + \tau U(w; \bar{w}),
$$

where the vector $y$ represents some "descent direction" (the gradient for example), and $h(w)$ is a simple convex function $[8]$. Moreover, $U$ denotes the Bregman distance, $\bar{w}$ is a prox center, and $\tau$ is a stepsize parameter. Together they ensure the output $\bar{w}$ is close to $\bar{w}$. In particular, the following will referred to as the $X$-projection:

$$
\hat{x} \leftarrow \arg \min_{x \in X} \langle y, x \rangle + u(x) + \tau \|x - \bar{w}\|^2.
$$

2. The Accelerated Constrained Gradient Descent Method. In this section, we propose a novel nested Lagrangian reformulation to (1.1) to motivate the design of the ACGD method. We assume the existence of an KKT point to the Lagrangian defined in (1.3) throughout this paper.

Assumption 1. There exists a $\lambda^* \in \mathbb{R}_+^m$ and $x^* \in X$ such that $\nabla f(x^*) + \sum_{i=1}^m \lambda_i^* \nabla g_i(x^*) + u' \in N_X(x^*)$ for some $u' \in \partial u(x^*)$, $g(x^*) \leq 0$, and $\lambda_i^* g_i(x^*) = 0 \forall i$.

Since the assumption implies that any $(\bar{x}, \lambda)$ with $\bar{x} \in X^*$ and $\lambda \in \Lambda^*$ := arg max$_{\lambda \in \mathbb{R}_+^m}$ min$_{x \in X} \mathcal{L}(x; \lambda)$ also constitutes a KKT point (see Proposition 3.4.1 in [2]), $x^*$ and $\lambda^*$ should be interpreted as an arbitrary element in $X^*$ and $\Lambda^*$ for the rest of the paper.

4
If the optimal dual multiplier $\lambda^*$ is known, an optimal solution $x^*$ can be found by solving the following simplified problem under certain regularity conditions:

$$\min_{x \in X} \{ f(x) + \langle \lambda^*, g(x) \rangle + u(x) \}. \quad (2.1)$$

Since the function constraints are absent in (2.1), and the objective value is (strongly) convex and smooth, the optimal FO-oracle complexity is $(\mathcal{O}\{ \log(1/\epsilon) \}) \mathcal{O}\{ 1/\sqrt{\epsilon} \}$. One useful framework for designing such an optimal algorithm is to consider a bilinear saddle-point reformulation

$$\min_{x \in X} \max_{\nu \in V, \pi \in \Pi} \{ \langle x, \pi \rangle - f^*(\pi) + u(x) + \langle \lambda^*, \nu x - g^*(\nu) \rangle \}, \quad (2.2)$$

where $f^*$ is the Fenchel conjugate function to $f$ and $\Pi$ is its domain, namely, $\{ \pi \in \mathbb{R}^n : f^*(\pi) < \infty \}$, and $g^* := [g^*_1, \ldots, g^*_m]$ is the vector-valued Fenchel conjugate to $g$ and $V$ is its domain, namely, $\{ \nu \in \mathbb{R}^{m \times n} : g^*_i(\nu_i) < \infty \ \forall \ i \in [m] \}$. In particular, when applied to (2.2), the primal-dual algorithm in [8] achieves the desired oracle complexities. In our development, since the dual variables $\pi$ and $\nu$ are associated with a common primal variable $x$, it is sometimes helpful consider the following joint domain $[V, \Pi]$:

$$[V, \Pi] = \{ (\nabla g(x), \nabla f(x)) : x \in \mathbb{R}^n \}. \quad (2.3)$$

Returning back to the Lagrangian formulation in (1.3), since $\lambda^*$ is unknown in practice, we propose to consider the following nested Lagrangian reformulation which combines (1.3) and (2.2):

$$\min_{x \in X} \max_{\lambda \in \mathbb{R}^n, \pi \in \Pi, \nu \in V} \{ L(x; \lambda, \pi, \nu) := \langle x, \pi \rangle - f^*(\pi) + u(x) + \langle \lambda, \nu x - g^*(\nu) \rangle \}, \quad (2.4)$$

where $f^*$, $g^*$, $\Pi$ and $V$ are defined in the same way as (2.2). Notice a common notation $L$ is used for the nested Lagrangian and the ordinary Lagrangian, but the exact meaning should be clear from the context. Let $Z$ denote the joint domain, $Z := X \times \mathbb{R}^n \times \Pi \times V$ and $z^* := (x^*, \lambda^*, \nu^*, \pi^*) := (\nabla g(x^*), \nabla f(x^*))$. It is not hard to see that $z^*$ is saddle point to (2.4), and an useful criterion to measure the optimality of an iterate $z^i := (x^i, \lambda^i, \nu^i, \pi^i) \in Z$ is to compare it to some reference point $z \in Z$ in the following gap function:

$$Q(z^i; z) := L(x^i; \lambda^i, \nu^i, \pi^i) - L(x; \lambda^i, \nu^i, \pi^i). \quad (2.5)$$

Indeed, the saddle point $z^*$ satisfies $Q(z^*; z) \leq 0 \ \forall z \in Z$. For showing convergence to an $\epsilon$-feasible point, we can restrict the reference $\lambda$’s to a certain bounded set and $(\nu, \pi)$ to the joint domain $[V, \Pi]$ in (2.3) to obtain better constant dependence. The next lemma shows that the $Q$ function provides upper bounds for both the feasibility violation and the optimality gap with such a restricted domain.

**Lemma 2.1.** Let $z^i := (x^i, \lambda^i, \nu^i, \pi^i) \in Z$ be given and let $\Lambda_r$ denote a certain set of reference $\lambda$’s,

$$\Lambda_r = \{0\} \cup \{ \lambda^* + \lambda : \lambda \in B_r(0) \cap \mathbb{R}^n \}. \quad (2.6)$$

If $\max_{\lambda \in \Lambda_r, (\nu, \pi) \in [V, \Pi]} Q(z^i; (x^*; \lambda, \nu, \pi)) \leq \epsilon$, we have $f(x^i) - f(x^*) \leq \epsilon$ and $r \|[g(x^*)]_+\| \leq \epsilon$.

**Proof.** Consider Part a). Fixing $\hat{\pi} = \nabla f(x^i)$ and $\hat{\nu} = \nabla g(x^i)$, the given condition implies that $Q(z^i; (x^*; \lambda, \nu, \hat{\pi})) \leq \epsilon \ \forall \lambda \in \Lambda_r$. It follows from the conjugate duality relationship (??? in [11]) that

$$f(x^i) + \langle \lambda, g(x^i) \rangle - [f(x^*) + \langle \lambda, g(x^*) \rangle] \leq \epsilon \ \forall \lambda \in \Lambda_r.$$

Since $x^*$ is feasible, i.e., $g(x^*) \leq 0$, we have $\langle \lambda, g(x^*) \rangle \leq 0$. Then taking $\lambda = 0$ leads to

$$f(x^i) - f(x^*) \leq [f(x^i) + \langle \lambda, g(x^i) \rangle] \leq \epsilon.$$

Next since $(x^*, \lambda^*)$ is a saddle point to (1.3), we have $0 \leq f(x^i) + \langle \lambda^*, g(x^i) \rangle - [f(x^*) + \langle \lambda^*, g(x^*) \rangle]$. Setting $\hat{\lambda} = \lambda^* + r [g(x^i)]_+ / \|[g(x^i)]_+\| \in \Lambda_r$, we get

$$r \|[g(x^i)]_+\| \leq f(x^i) + \langle \lambda^* + r [g(x^i)]_+ / \|[g(x^i)]_+\|, g(x^i) \rangle - [f(x^*) + \langle \lambda^*, g(x^*) \rangle]$$

$$\leq Q(z^i; (x^*; \hat{\lambda}, \hat{\nu}, \hat{\pi})) \leq \epsilon.$$
Let us move on to the task of minimizing $Q$ so as to find a saddle point to (2.4). An essential feature of the nested Lagrangian function (2.4) is the trilinear term $\langle \lambda, \nu x - g^*(\nu) \rangle$. The problem cannot be simplified to a min–max saddle point problem by combining the $\nu$ and $\lambda$ into a single dual block, because their joint maximization is difficult to compute. Similar problems have been studied by Lan and Zhang in [11, 24, 25] and the key to handle the min–max–max trilinear structure is to decompose the $Q$ function into the sub-gap functions and optimize each sub-gap function sequentially. Consider the following decomposition of (2.4) into sub-gap functions associated with $x$, $\lambda$, $\nu$ and $\pi$:

$$Q(z^t; z) = Q_x(z^t; z) + Q_\lambda(z^t; z) + Q_\nu(z^t; z) + Q_\pi(z^t; z)$$

(2.7)

where

$$Q_\nu(z^t; z) := \mathcal{L}(x^t; \lambda, \nu, \pi) - \mathcal{L}(x^t; \lambda, \nu^t, \pi) - \sum_{i=1}^m \lambda_i (\nu_i, x^t) - g^*_i(\nu^t_i) - \gamma (\nu_i, x^t) - g^*_i(\nu^t_i),$$

$$Q_\pi(z^t; z) := \mathcal{L}(x^t; \lambda, \nu, \pi^t) - \mathcal{L}(x^t; \lambda, \nu^t, \pi^t) = \langle \lambda, \nu^t x^t - g^*(\nu^t) \rangle - \langle \lambda^t, \nu^t x^t - g^*(\nu^t) \rangle,$$

$$Q_x(z^t; z) := \mathcal{L}(x^t; \lambda^t, \nu^t, \pi^t) - \mathcal{L}(x^t; \lambda^t, \nu^t, \pi^t) = \langle \lambda^t, \nu^t x^t - g^*(\nu^t) \rangle - \langle \lambda^t, \nu^t x^t - g^*(\nu^t) \rangle.$$ 

Similar to the DRAO method in [11], each iteration of ACGD method consists of the following prox-mappings to reduce the boxed terms associated with $Q_\nu$ and $Q_\pi$, and $Q_x$ and $Q_\lambda$:

$$\tilde{x}^t \leftarrow x^{t-1} + \theta_t (x^{t-1} - x^{t-2});$$

$$\pi^t \leftarrow \arg\max_{\pi \in \Pi} \langle \pi, \tilde{x}^t \rangle - f^*(\pi) - \tau_t U_f^\ast(\pi; \pi^{t-1});$$

$$\nu_i^t \leftarrow \arg\max_{\nu_i \in V_i} \langle \nu_i, x^t \rangle - g^*_i(\nu_i) - \tau_t U_{g_i}^\ast(\nu_i; \nu_i^{t-1}) \forall i \in [m];$$

$$\lambda^t, \lambda^t \leftarrow \arg\min_{x^t} \arg\max_{\lambda^t \in \mathbb{R}_+^m} \langle \lambda^t, x^t \rangle + \langle \lambda^t, x^t - g^*(\nu^t) \rangle + u(x) - \eta_t \|x - x^{t-1}\|^2 / 2.$$ 

(2.8)

In the above listing, $\theta_t$, $\tau_t$ and $\eta_t$ are non-negative stepsize parameters, and $U_{g_i}^\ast$ and $U_f^\ast$ are Bregman distance functions generated by $g_i^*$ and $f^*$ respectively. Since $x^t$ is computed after $\nu^t$ and $\pi^t$, we use a proxy $\tilde{x}^t$ for the $\pi^t$ and $\nu^t$ computation. The point $\tilde{x}^t$ is generated by momentum extrapolation, and it helps us to achieve acceleration (e.g. see Section 3.4 in [8] for the connection).

The implementable version of the ACGD method is shown in Algorithm 1. It employs two simplifications compared to (2.8). First, we initialize the dual variables to some gradients, i.e., $\pi^0 = \nabla f(x^0)$ and $\nu^0 = \nabla g(x^0)$. With $U_{g_i}^\ast$ and $U_f^\ast$ selected as prox-functions, we can show recursively that $\pi^t$ and $\nu^t$ in (2.8) are same as the gradients at some averaged point (see Lemma 2 in [20]). Thus the $\nu_i^t$ and $\pi^t$ computation in (2.8) simplifies to gradient evaluations in Line 4 of Algorithm 1. Second, the $(x^t, \lambda^t)$-saddle point problem in (2.8) is formulated as a linearly constrained quadratic program in Line 5 of Algorithm 1. Since $\nu^t = \nabla g(x^t)$ in Algorithm 1 implies the relation $g_i(x^t) + g_i^*(\nu^t) = \langle x^t, \nu_i^t \rangle$ (see Theorem 4.20 in [1]), we have $g^*(\nu^t) = \nu_i^t x^t - g(x^t)$. Interestingly, other than the additional linear constraint associated with $g$ in the descent step (Line 5), Algorithm 1 is the same as Nesterov’s AGD method [18], hence the name Accelerated Constrained Gradient Descent method.
Next we present convergence results for the ACGD method. The detailed analysis is deferred to Subsection 2.2. The next proposition states some conditions required for the Algorithm 1 convergence results. We pick the set of reference multipliers \( \lambda^t \)'s under consideration, \( \Lambda \subset \mathbb{R}_+^n \). So it is useful to define an aggregate Lipschitz smoothness constant as a function of \( \Lambda \):

\[
L(\Lambda) := \max_{\lambda \in \Lambda} L_{\lambda}, \text{ where } L_{\lambda} = \lambda \bar{\nabla}_x \lambda \geq \|\nabla_x \mathcal{L}(x; \lambda) - \nabla_x \mathcal{L}(x; \lambda)\| \forall \bar{x}, \hat{x} \in \mathbb{R}^n. \tag{2.9}
\]

**Proposition 2.2.** Let a set of reference multipliers \( \Lambda \subset \mathbb{R}_+^n \) be given and let the aggregate smoothness constant \( L(\Lambda) \) be defined in (2.9). Consider the iterates \( x^t := (\lambda^t; \nu^t, \pi^t) \) generated by Algorithm 1, where \( \lambda^t \) is defined in (2.8). Suppose the following conditions are satisfied by the stepizes together with some non-negative weights \( \omega_t \geq 0 \) for all \( t \geq 2 \):

\[
\begin{align*}
\omega_t \eta_t & \leq \omega_{t-1}(\eta_{t-1} + \alpha), \tag{2.10} \\
\omega_t \tau_t & \leq \omega_{t-1}(\tau_{t-1} + 1), \tag{2.11} \\
\eta_N(\eta_N + 1) & \geq L(\Lambda), \eta_{t-1} \tau_t \geq \theta_t L(\Lambda) \text{ with } \theta_t := \omega_{t-1}/\omega_t. \tag{2.12}
\end{align*}
\]

Let the ergodic iterate \( \bar{x}^N := (\bar{x}^N; \bar{\lambda}^N, \bar{\nu}^N, \bar{\pi}^N) \) be specified according to

\[
\bar{x}^N := \sum_{t=1}^N \omega_t x^t / (\sum_{t=1}^N \omega_t), \quad \bar{\lambda}^N := \sum_{t=1}^N \omega_t \lambda^t / (\sum_{t=1}^N \omega_t), \quad \bar{\pi}^N := \sum_{t=1}^N \omega_t \pi^t / (\sum_{t=1}^N \omega_t), \quad \bar{\nu}^N := \frac{1}{\sum_{t=1}^N \omega_t \nu^t / (\sum_{t=1}^N \omega_t)} \text{ a.w. if } \lambda^t \neq 0 \forall t. \tag{2.13}
\]

Then the following convergence bound is valid for any reference point \( z = (x; \lambda, \nu, \pi) \in X \times \Lambda \times [V, \Pi] \) with \([V, \Pi]\) being defined in (2.3):

\[
(\sum_{t=1}^N \omega_t) Q(\bar{x}^N, z) + \omega_N(\eta_N + \alpha) \|x^N - x\|^2 / 2 \leq \omega_1 \eta_1 \|x^0 - x\|^2 + \omega_1 \tau_1 (U^*_f(\pi; \pi^0) + \langle \lambda, U^*_g(\nu; \nu^0) \rangle). \tag{2.14}
\]

Now we suggest some step-size choices satisfying the requirements in Proposition 2.2 to derive concrete convergence results. We pick the set of reference \( \lambda \)'s to be \( \Lambda_r \) in (2.6) to provide bound for feasibility violation.

**Theorem 2.3.** Let a smooth constrained optimization problem (1.1) be given and let \( L(\Lambda_r) \) be defined in (2.6) and (2.9). Denote its condition number by \( \kappa_r = L(\Lambda_r) / \alpha \) (We set \( \kappa_r = \infty \) if \( \alpha = 0 \).) Suppose the solution iterates \( \{x^t\} \) are generated by Algorithm 1 with the following stepizes for \( t \geq 1 \):

\[
\tau_t = \min\{\frac{t-1}{2}, \sqrt{\kappa_r}\}, \quad \eta_t = \frac{L(\Lambda_r)}{\tau_{t+1}}, \quad \theta_t = \frac{\tau_t}{\tau_{t-1} + 1}, \quad \omega_t = \begin{cases} \omega_{t-1}/\theta_t & \text{if } t \geq 2, \\ 1 & \text{if } t = 1. \end{cases} \tag{2.15}
\]

Then the ergodic average \( \bar{x}^N \) solution satisfies

\[
\max\{f(\bar{x}^N) - f^*, r \|g(\bar{x}^N)\|_+\} \leq \frac{2L(\Lambda_r)}{N(N + 1)} \|x^0 - x^\ast\|^2. \tag{2.16}
\]
Moreover, in the strongly convex case with $\alpha > 0$, $\bar{x}^N$ also satisfies

$$\max\{f(\bar{x}^N) - f_*, r \left\| g(\bar{x}^N) \right\|_1 \} \leq \frac{\sqrt{L(\Lambda_r)}\alpha \left\| x^0 - x^* \right\|^2}{(1 + 1/\sqrt{\kappa_r})N - 4 - 1},$$

(2.17)

$$\left\| \bar{x}^N - x^* \right\|^2 \leq \frac{2\sqrt{\kappa_r} \left\| x^0 - x^* \right\|^2}{(1 + 1/\sqrt{\kappa_r})N - 4 - 1}.$$  

(2.18)

As a consequence of the preceding theorem, we can derive upper bounds on the number of iterations required of the ACGD method to find an $(c, \epsilon/c)$-solution (see (1.2)). The next corollary focuses on the non-strongly convex case, i.e., $\alpha = 0$.

**Corollary 2.4.** Suppose $\{x^t\}$ are generated by Algorithm 1 using the stepsizes choice in (2.15), with $L(\Lambda_r) = L(\Lambda_c)$ (see (2.9)). The ergodic average solution $\bar{x}^N$ is an $(\epsilon, \epsilon/c)$-solution if

$$N \geq \sqrt{\frac{2L(\Lambda_c)}{\epsilon}} \left\| x^0 - x^* \right\|.$$  

As for the strongly convex problem with $\alpha > 0$, it is advantageous to choose a small $r$ to ensure a small aggregate smoothness constant $L(\Lambda_r)$ and hence a small condition number $\kappa_r$. We set it to $L(\Lambda_1)$ in the next corollary.

**Corollary 2.5.** Suppose $\{x^t\}$ are generated by Algorithm 1 using the stepsizes choice in (2.15), with $L(\Lambda_r) = L(\Lambda_1)$ (see (2.9)). Then the ergodic average solution $\bar{x}^N$ is an $(\epsilon, \epsilon/c)$-solution if

$$N \geq \min\{\sqrt{\frac{2\max\{c,1\}L(\Lambda_c)}{\epsilon}} \left\| x^0 - x^* \right\|, \left\lfloor \frac{L(\Lambda_c)}{\epsilon} + 1 \right\rfloor \log\left[ \frac{\max\{c,1\}L(\Lambda_c)\left\| x^0 - x^* \right\|^2}{\epsilon} \right] + 1 \} + 4 \}.$$  

Note that for a small $\lambda^*$, i.e., $\|\lambda^*\|$ is significantly less than 1, it may be worthwhile to choose $L(\Lambda_r) = L(\lambda^*)$ to further reduce the condition number from $\kappa_1 = L(\Lambda_1)/\alpha$ to $\kappa_1\|\lambda^*\| = L(\lambda^*)/\alpha$.

Since each iteration of the ACGD method requires only one gradient evaluation, preceding two corollaries establish the desired $O(1/\sqrt{\epsilon})$ and $O((\log(1/\epsilon))$ oracle complexities for the non-strongly convex and the strongly convex problems, respectively.

### 2.1. Online Search for $L(\Lambda_r)$

A crucial limitation to the ACGD method is that the desired aggregate smoothness constants, $L(\Lambda_r)$ and $L(\Lambda_1)$, may be unavailable for stepsize calculation in (2.15) during implementation. To address this issue, we propose a binary search outer-loop method to look for the suitable constant in an online fashion. Starting with a guess $L$, we run the ACGD method for $N(L)$ iterations and check if the ergodic average solution $\bar{x}^{N(L)}$ satisfies a certain termination criterion. The outer loop would be stopped if the termination criterion is met, otherwise the process would be repeated with the guess doubled, i.e., $2L$. To ensure the soundness of search routine, we need to choose the iteration limit function $N(L)$ such that the termination criterion would be met if $L$ were larger than the desired aggregate smoothness constant. Additionally, the termination criterion should also be both easily computable and valid, i.e., a test solution $\bar{x}^N$ which satisfies the criterion is $(\epsilon, c)$-optimal. Towards these ends, we need to make the following additional assumptions:

**Assumption 2.**

a) $X$ is bounded and we know an estimate $D_X$ of its radius, i.e., $D_X \geq \max_{x,y \in X} \left\| x - y \right\|$.

b) The dual multiplier to the quadratic program in Line 5 of Algorithm 2 $\lambda^t$, is available.

Given iterates $\{(x^t, \bar{x}^t), \lambda^t, \nu^t, \pi^t\}_{t=1}^N$ generated by the ACGD method, our termination criterion is the satisfaction of the following two conditions:

$$\left\| g(\bar{x}^N) \right\|_1 \leq \epsilon c \text{ and } f(\bar{x}^N) - \bar{f}^N \leq \epsilon,$$

(2.19)

where $\bar{x}^N$ is the ergodic average solution and $\bar{f}^N$ is given by

$$\bar{f}^N := \arg\min_{x \in X} \frac{1}{\sum_{t=1}^N \omega_t} \sum_{t=1}^N \omega_t [\pi^t, x - \bar{x}^t] + f(\bar{x}^t)]$$

$$\text{s.t. } \frac{1}{\sum_{t=1}^N \omega_t \lambda^t_i} \sum_{t=1}^N \omega_t \lambda^t_i (\nu^t_i, x - \bar{x}^t) + g_i(\bar{x}^t) \leq 0 \forall i \in [m].$$  

(2.20)
Notice that Assumption 2.a) ensures the finiteness of $f^N$, and that the dual solutions in Assumption 2.b) are used for constructing the linearized constraints above. Since its objective and constraints are lower linear approximation to $f$ and $g_1, g_2, \ldots, g_m$ respectively, (2.20) is a relaxation to (1.1). Its minimum value $\bar{f}^N$ is a lower bound to $f^*$. Therefore the satisfaction of (2.19) guarantees that $\bar{x}^N$ is an $(\epsilon, c\epsilon)$-optimal solution.

Algorithm 2 Binary Search for the Aggregate Smoothness Constant $L$.

**Input:** $\bar{x} \in X$, $\bar{L} > 0$, and an iteration limit function $N(L)$.

1: while true do
2:   Set stepsizes to (2.15) with $L(\Lambda_c)$ replaced by $\bar{L}$.
3:   Run the ACGD method for $N(\bar{L})$ iterations starting from $\bar{x}$.
4:   Set $\bar{x} \leftarrow \bar{x}^{N(\bar{L})}$, where $\bar{x}^{N(\bar{L})}$ is the ergodic average solution.
5:   if $\bar{x}$ satisfies the condition (2.19) then
6:     break
7:   else
8:     Set $\bar{L} \leftarrow 2\bar{L}$.
9:   end if
10: end while
11: return $\bar{x}$.

Putting together, the binary search outer loop is given in Algorithm 2. If it terminates, the output solution $\bar{x}$ is guaranteed to be $(\epsilon, c\epsilon)$-optimal. Now we suggest some concrete iteration limit functions, $N(L)$, and derive their total number of iterations, across all ACGD runs, required for termination. Their proof is deferred to Subsection 2.2.

**Theorem 2.6.** Consider a smooth constrained optimization problem of form (1.1). Suppose its aggregate smoothness constants, $L(\Lambda_c)$ and $L(\Lambda_1)$, are defined according to (2.9) and (2.6), and Assumption 2 is valid. Let it be initialized to some $\tilde{L}_0$, then Algorithm 2 must terminate finitely with the following choices of iteration limit function $N(L)$.

a) In the non-strongly convex case with $\alpha = 0$, if $N(L) = \lceil \sqrt{\frac{2L(\Lambda_c)}{\epsilon} D_X} \rceil$, the total number of ACGD iterations required for termination is upper bounded by

$$\max\{7\sqrt{\frac{L(\Lambda_c)}{\epsilon} D_X} + \lceil \log_2 \frac{L(\Lambda_c)}{L_0} \rceil, \sqrt{\frac{2L(\Lambda_c)}{\epsilon} D_X} + 1\}.$$  \hspace{1cm} (2.21)

b) In the strongly convex case with $\alpha > 0$, if

$$N(L) = \lceil (\sqrt{\frac{L(\Lambda_1)}{\epsilon}} + 1) \log\left(\frac{\max\{c,1\} \sqrt{\frac{L(\Lambda_1)}{L_0}} D_X^2}{\epsilon} + 1\right) + 4\rceil,$$

the total number of ACGD iterations required for termination is upper bounded by

$$\max\left\{9\sqrt{\frac{L(\Lambda_1)}{\epsilon}} \log\left(\frac{\max\{c,1\} \sqrt{\frac{2L(\Lambda_1)}{\epsilon} D_X^2}}{\epsilon} + 1\right) + 5\lceil \log_2 \frac{L(\Lambda_1)}{L_0} \rceil, \right\}.$$  \hspace{1cm} (2.22)

We make two remarks regarding the result. First, the proposed $N(L)$’s depend only on the target accuracy $(\epsilon, c)$ and some easy-to-estimate problem parameters, that is, $D_X$ in the non-strongly convex case, and $D_X$ and $\alpha$ in the strongly convex case. So Algorithm 2 is easy to implement. Second, the total numbers of ACGD iterations required by Algorithm 2 are nearly independent of the initial guess $L_0$ for all $\tilde{L}_0 \leq L(\Lambda_0)$. The FO oracle complexities of Algorithm 2 have the same orders as those in Corollary 2.4 and 2.5. So the total cost of the multiple ACGD runs in Algorithm 2 is on the same order as a single ACGD run with carefully selected stepsizes.
2.2. Convergence Analysis. We present the detailed proofs to convergence results of Algorithm 1.

Proof of Proposition 2.24 Let \( Q_\nu, Q_\lambda, Q_\psi, \) and \( Q_\sigma \) be defined in (2.27). It is useful to view the updates in Algorithm 1 from the perspective of prox-mappings in (2.8). First, let’s consider \( Q_\nu \). We have from the definition of \( \tilde{x}^t \) that

\[
\langle \nu_i - \nu_i^t, (\tilde{x}^t - x^t) \rangle = \langle \nu_i - \nu_i^t, (x^t - x^{t-1}) \rangle + \theta_i \langle \nu_i - \nu_i^{t-1}, (x^{t-1} - x^{t-2}) \rangle + \theta_i \langle \nu_i^{t-1} - \nu_i^t, (x^{t-1} - x^{t-2}) \rangle.
\]

The definition of \( \nu_i^t \) prox-mapping in (2.8) implies a three-point inequality (see Lemma 3.5 in [8]):

\[
\langle \nu_i - \nu_i^t, x^t \rangle + g_i^*(\nu_i^t) - g_i^*(\nu_i) + \langle \nu_i - \nu_i^t, \tilde{x}^t - x^t \rangle \\
\leq \tau_i U_{g_i^*}(\nu_i; \nu_i^{t-1}) - (\tau_i + \mu) U_{g_i^*}(\nu_i; \nu_i^t) - \tau_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}).
\]

So, combining the above two relations, taking the \( \omega_i \) weighted sum of the resulting inequalities and using the conditions that \( \omega_{i-1} = \omega_i \theta_i \) and \( \omega_i \tau_i = \omega_{i-1} (\tau_{i-1} + \mu) \), we obtain

\[
\sum_{t=1}^N \omega_i Q_\nu(z^t; z) \\
\leq -\omega_N (\tau_N + 1) \sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^t) - \omega_N \langle \sum_{i=1}^m \lambda_i (\nu_i - \nu_i^N), x^N - x^{N-1} \rangle \\
- \sum_{t=2}^N [\omega_t \tau_t \sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + \omega_{i-1} \langle \sum_{i=1}^m \lambda_i (\nu_i^{t-1} - \nu_i), x^{t-1} - x^{t-2} \rangle] \\
+ \omega_1 \tau_1 U_{g_1^*}(\nu_1; \nu_0^0).
\]

A \( \lambda_i \)-weighted sum of the above inequality leads to the desired \( Q_\nu \) convergence bound given by

\[
\sum_{i=1}^N \omega_i [Q_\nu(z^t; z) + Q_\sigma(z^t; z)] \\
\leq -\omega_N (\tau_N + 1) \sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^t) + U_{f^*}(\pi; \pi^t) - \omega_N \langle \pi - \pi^N + \sum_{i=1}^m \lambda_i (\nu_i - \nu_i^N), x^N - x^{N-1} \rangle \\
- \sum_{t=2}^N [\omega_t \tau_t \sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + U_{f^*}(\pi; \pi^{t-1})] + \omega_{i-1} \langle \pi - \pi^N + \sum_{i=1}^m \lambda_i (\nu_i^{t-1} - \nu_i), x^{t-1} - x^{t-2} \rangle \\
+ \omega_1 \tau_1 [\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^0) + U_{f^*}(\pi; \pi^0)].
\]

Since \( (\nu^t, \pi^t) = (\nabla g(x^t), \nabla f(x^t)) \) \( \forall t \), applying Lemma 6.1 with \( \tilde{\lambda} = [\lambda; 1] \) and \( \tilde{g} = [g; f] \) implies that

\[
\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i^t; \nu_i^{t-1}) + U_{f^*}(\pi; \pi^{t-1}) \geq \frac{1}{2\tilde{L}(\mathcal{A})} \|\pi^t - \pi^{t-1}\|^2 + \sum_{i=1}^m \lambda_i \|\nu_i^t - \nu_i^{t-1}\|^2.
\]

Similarly, with \( (\nu, \pi) \in [V_\nu, \Pi] \) (see (2.9)), we get

\[
\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^N) + U_{f^*}(\pi; \pi^N) \geq \frac{1}{2\tilde{L}(\mathcal{A})} \|\pi - \pi^N\|^2 + \sum_{i=1}^m \lambda_i \|\nu_i - \nu_i^N\|^2.
\]

Thus applying the Young’s inequality to (2.23) leads to

\[
\sum_{i=1}^N \omega_i [Q_\nu(z^t; z) + Q_\sigma(z^t; z)] \\
\leq \omega_N \frac{L(\mathcal{A})}{2(\tau_N + 1)} \|\nu^N - \nu^{N-1}\|^2 + \sum_{t=1}^{N-1} \omega_t \frac{\theta_i L(\mathcal{A})}{2\tau_N} \|x^t - x^{t-1}\|^2 + \omega_1 \tau_1 [\sum_{i=1}^m \lambda_i U_{g_i^*}(\nu_i; \nu_i^0) + U_{f^*}(\pi; \pi^0)].
\]
Now let us move onto $Q_x$ and $Q_{\lambda}$. Fix $\lambda^t$, the $x^t$-prox mapping implies a three-point inequality (Lemma 3.5 in [8]):

$$
\langle \pi^t + \sum_{i=1}^{m} \lambda_i^t \nu_i^t, x^t - x \rangle + u(x^t) - u(x) + \sum_{i=1}^{m} \frac{\kappa}{2} \|x^t - \hat{x}^t\|_2^2 \leq \frac{\kappa}{2} \|x^{t-1} - \hat{x}^t\|_2^2 - \frac{\kappa}{2} \|x^t - x^{t-1}\|_2^2.
$$

Fix $x^t$, the optimality of $\lambda^t$ implies that:

$$
\langle \lambda^t - \lambda^t, \nu^t x^t - g^*(\nu^t) \rangle \leq 0.
$$

So we have

$$
Q_x(z^t; z) + Q_{\lambda}(z^t; z) \leq \frac{\kappa}{2} \|x^{t-1} - x\|_2^2 - \frac{\kappa}{2} \|x^t - x^{t-1}\|_2^2.
$$

Summing across iterations with weight $\omega^t$ leads to

$$
\sum_{t=1}^{N} \omega_t [Q_x(z^t; z) + Q_{\lambda}(z^t; z)] + \frac{\kappa}{2} \|x^N - x\|_2^2 \leq \frac{\kappa}{2} \|x^0 - x\|_2^2 - \sum_{t=1}^{N} \omega_t \|x^t - x^{t-1}\|_2^2. \tag{2.25}
$$

Utilizing the stepsize assumption (2.12), we can add it to (2.24) to obtain a convergence bound for the $Q$ function

$$
\sum_{t=1}^{N} \omega_t Q(z^t; z) + \frac{\kappa}{2} \|x^N - x\|_2^2 \leq \frac{\kappa}{2} \|x^0 - x\|_2^2 + \omega_1 \tau_1 \left[ \sum_{t=1}^{m} \lambda_i U_{\nu_i^t} (\nu_i^t; \nu_i^0) + U_f (\pi; \pi^0) \right]. \tag{2.26}
$$

Moreover, the Jensen’s inequality implies that

$$
\sum_{t=1}^{N} \omega_t \mathcal{L}(x; \lambda^t, \nu^t, \pi^t) \geq \left( \sum_{t=1}^{N} \omega_t \right) \mathcal{L}((\bar{x}^N, \bar{\lambda}, \bar{\nu}, \bar{\pi})) \geq \sum_{t=1}^{N} \omega_t \mathcal{L}(x; \lambda, \nu, \pi). \tag{2.27}
$$

We have $(\sum_{t=1}^{N} \omega_t) Q(z^N; z) \leq \sum_{t=1}^{N} \omega_t Q(z^t; z)$, so the desired inequality in (2.14) follows from (2.26).

Next, the proof of Theorem 2.3 is a direct application of Proposition 2.2. The analysis is complicated by the switch from a diminishing stepsize to a constant stepsize in (2.15).

**Proof of Theorem 2.3** We apply Proposition 2.2 to obtain the results. First, we verify the requirements in (2.12) and (2.11) are satisfied by the stepsize choice in (2.15). Since $\theta_t = \tau_t / (\tau_{t-1} + 1)$ and $\eta_t = L(\Lambda_r) / \tau_{t+1}$, all requirements other than (2.10) hold automatically. Now let $T = [2 \sqrt{\kappa_r} + 1]$ denote the first iteration at which we switch to the $\tau_t = \sqrt{\kappa_r}$. Since other iterations and $\alpha = 0$ are straightforward to check, we focus on iteration $T - 1$ and $T$, and assume $\alpha > 0$. For $t = T$, we have

$$
\omega_T \eta_T = \omega_{T-1} \eta_T \tau_{T-1} t_{T-1} \leq \omega_{T-1} \frac{L(\Lambda_r)}{2 \sqrt{\kappa_r}} \leq \omega_{T-1} \alpha \sqrt{\kappa_r} \leq \omega_{T-1} (2 \sqrt{\kappa_r} + \alpha) \leq \omega_{T-1} (\eta_{T-1} + \alpha).
$$

For $t = T - 1$, we have

$$
\omega_{T-1} \eta_{T-1} = \omega_{T-2} \eta_{T-2} \tau_{T-2} t_{T-2} \leq \omega_{T-2} \frac{2 L(\Lambda_r)}{2 \sqrt{\kappa_r}} \leq \omega_{T-2} \frac{2 L(\Lambda_r)}{2 \sqrt{\kappa_r}} \leq \omega_{T-2} \eta_{T-2} \leq \omega_{T-2} (\eta_{T-2} + \alpha).
$$

Thus the requirements in Proposition 2.2 are satisfied, and we have

$$
Q(z^N, z) \leq L(\Lambda_r) \|x^0 - x\|_2^2 / (\sum_{t=1}^{N} \omega_t) \forall z = (x; \lambda, \nu, \pi) \in X \times \Lambda_r \times V \times \Pi. \tag{2.28}
$$

We provide a lower bound to $(\sum_{t=1}^{N} \omega_t)$. It is useful to show

$$
\omega_t \geq \max \{ t, (1 + 1/\sqrt{\kappa_r})^{t-5} \}.
$$

Since $\omega_t = 1 / (\prod_{t=2}^{N} \theta_t)$, the fact $\omega_t \geq t \forall t \geq 1$ is straightforward. Regarding the second lower bound, let us first consider $t \leq T = [2 \sqrt{\kappa_r} + 1]$. The algebraic fact in Lemma 6.2 implies that $\omega_{t+2} \geq t \geq (1 + 1/\sqrt{\kappa_r})^{t-3} \forall t \leq 2 \sqrt{\kappa_r}$, so $\omega_t \geq (1 + 1/\sqrt{\kappa_r})^{t-5} \forall t \leq T = [2 \sqrt{\kappa_r} + 1]$. For $t \geq T + 1$, the relation is also
valid since $1/\theta_t = (1 + 1/\sqrt{kr})$. Therefore we get $\omega_t \geq \max\{t, (1 + 1/\sqrt{kr})t^{-5}\} \forall t \geq 1$. Using these two lower bounds, it is easy to derive

$$\sum_{t=1}^{N} \omega_t \geq \max\{N(N+1)/2, \sqrt{kr}[1 + (1 + 1/\sqrt{kr})N^{-4} - 1]\}. \quad (2.29)$$

Substituting the preceding lower bound into (2.28) and applying Lemma 2.1 lead us to the convergence results in (2.19) and (2.17).

Now we deduce the convergence bound for $\|x^t - x^*\|$. Choosing $\tilde{z} = (x^*; \lambda^*, \nabla g(\tilde{x}^N), \nabla f(\tilde{x}^N))$ to be the reference point $z$ leads to

$$Q(\tilde{z}; \tilde{x}) \leq L(\Lambda_r) \|x^0 - x^*\|^2 / (\sum_{t=1}^{N} \omega_t). \quad (2.30)$$

Moreover, since $x^*$ minimizes the optimal Lagrangian, we get $(\nabla f(x^*) + (\lambda^*)^\top \nabla g(x^*) + \nu(x^*)', \tilde{x}^N - x^*) \geq 0$ for all $u'(x^*) \in \partial u(x^*)$. So we have

$$Q(\tilde{z}; \tilde{x}) \geq f(\tilde{x}^N) + (\lambda^*)^\top g(\tilde{x}^N) + u(\tilde{x}^N) - [f(x^*) + (\lambda^*)^\top g(x^*) + u(x^*)]$$

$$- (\nabla f(x^*) + (\lambda^*)^\top \nabla g(x^*) + \nu(x^*)', \tilde{x}^N - x^*)$$

$$= f(\tilde{x}^N) - f(x^*) - (\nabla f(x^*), \tilde{x}^N - x^*) + \sum_{t=1}^{m} \lambda_t^* [g_t(\tilde{x}^N) - g_t(x^*) - (\nabla g_t(x^*), \tilde{x}^N - x^*)]$$

$$+ [u(\tilde{x}^N) - u(x^*) - (\nu(x^*)', \tilde{x}^N - x^*)]$$

$$\geq \alpha \|\tilde{x}^N - x^*\|^2 / 2.$$

Combining the above two relations, we obtain the desired inequality in (2.19):

$$\|\tilde{x}^N - x^*\|^2 \leq 2\kappa r \|x^0 - x\|^2 / (\sum_{t=1}^{N} \omega_t). \quad (2.31)$$

We move on to provide proofs to the binary search outer-loop in Algorithm 2.

**Proof of Theorem 2.6.** First, let’s consider the non-strongly problem. We show the soundness of the outer search method, that is, the ergodic average solution from the ACGD method satisfies termination criterion (2.19) if the estimate $\tilde{L}$ is larger than the desired aggregate smoothness constant $L(\Lambda_c)$. Let $\tilde{N}$ be $\sqrt{2L/\epsilon D_X}$ such that $\tilde{x}^N$ denotes the ergodic average solution. Since $D_X \geq \|x^0 - x^*\|$, $\|g(\tilde{x}^N)\| \leq \epsilon/c$ is a direct consequence of Corollary 2.4. Now consider the objective criterion. Since $\tilde{L} \geq L(\Lambda_c)$, it follows from (2.26) that

$$\sum_{t=1}^{N} \omega_t Q(z^t; z) \leq \omega_1 \eta_1 \|x^0 - x\|^2 / 2 \leq \omega_1 \eta_1 D_X^2 / 2, \forall z = (x; \lambda, \nu, \pi) \in X \times L(\Lambda_c) \times V \times \Pi.$$

Substituting the stepsize choice in (2.16) then leads to

$$\sum_{t=1}^{N} \omega_t Q(z^t; z) \leq \frac{LD_X^2}{\sum_{t=1}^{N} \omega_t} \leq \epsilon, \forall z = (x; \lambda, \nu, \pi) \in X \times L(\Lambda_c) \times V \times \Pi.$$

In particular, fixing $\tilde{\pi} = \nabla f(\tilde{x}^N)$ and $\tilde{\nu} = \nabla g(\tilde{x}^N)$, we have

$$f(\tilde{x}^N) \leq (\sum_{t=1}^{N} \omega_t L(x^t; 0, \tilde{\nu}, \tilde{\pi}))(\sum_{t=1}^{N} \omega_t).$$

Moreover, since

$$\sum_{t=1}^{N} \frac{\omega_t}{\sum_{t=1}^{N} \omega_t} L(x; \lambda^t, \nu^t, \pi^t) = \frac{1}{\sum_{t=1}^{N} \omega_t} \sum_{t=1}^{N} \omega_t L(x^t; x - \tilde{x}^N) + f(x^t)$$

$$+ \sum_{t=1}^{m} \frac{\sum_{t=1}^{N} \omega_t \lambda_t^t}{\sum_{t=1}^{N} \omega_t} \sum_{t=1}^{N} \omega_t \lambda_t^t (x^t - \tilde{x}^N) + g_t(x^t)$$

$$= \frac{1}{\sum_{t=1}^{N} \omega_t} \sum_{t=1}^{N} \omega_t \lambda_t^t L(x^t; x^t - \tilde{x}^N) + f(x^t).$$
The preceding function of $x$ is a certain Lagrangian relaxation to (2.20), thus

$$
\min_{x \in X} \frac{\omega_1}{\sum_{t=1}^{N} \omega_t} \mathcal{L}(x; \lambda^t, \nu^t, \pi^t) \leq f^N.
$$

Combining these facts together, we get

$$
f(\bar{x}^N) - f^N \leq \max_{z \in \mathbb{X} \times L(\Lambda_1) \times V} \frac{\sum_{t=1}^{N} \omega_t Q(z^t, z)}{\sum_{t=1}^{N} \omega_t} \leq \epsilon.
$$

Thus $\bar{x}^N$ must satisfy the termination criterion in (2.19) if $\bar{L} \geq L(\Lambda_c)$. The total number of ACGD iterations follows from a straightforward algebraic calculation.

Next, the finite termination of Algorithm 2 for the strongly convex case can be deduced similarly. With the given choice of $N(L)$, we need to show the termination criterion (2.19) is satisfied for the ergodic solution $\bar{x}^N$ if the estimate $\bar{L}$ were larger than $L(\Lambda_1)$.

### 3. Lower Oracle Complexity Bound.

In this section, we provide lower complexity bounds for the first-order oracle, that is, the minimum number of queries to the FO oracle required to find an $(\epsilon, c)$-optimal solution. These results will illustrate that the ACGD method is optimal in certain ways. We assume, for the sake of simplicity, $X$ is radially invariant, e.g., $\mathbb{R}^n$, and $u(x) = x^2 / 2$ (see (1.1)).

Similar to Nesterov’s lower complexity computation model in [16], we consider the class of all first order methods, $\mathcal{F}$, verifying the following linear-span update requirement. Given a (finite) memory of reachable points $\mathcal{M}_{t-1}$ after the $t-1$th query to FO, the next memory after evaluating the FO at some $y \in \text{span}(\mathcal{M}_{t-1})$ needs to satisfy

$$
\mathcal{M}_t \subset \{ x + \eta \nabla f(y) + \sum_{i=1}^{m} \tau_i \nabla g_i(y) : x \in \text{span}(\mathcal{M}_{t-1}), \eta, \tau_i \in \mathbb{R} \}. \quad (3.1)
$$

The freedom to choose arbitrary elements from the linear span allows $\mathcal{F}$ to cover many first-order algorithms. For instance, consider the ACGD method. If the memory $\mathcal{M}_{t-1}$ contains all generated points up to iteration $t-1$, i.e., $\{x^0, x^1, \ldots, x^{t-1}\}$, in Algorithm 1 the evaluation point $x^t$ in Line 3 is inside $\text{span}(\mathcal{M}_{t-1})$. Moreover, the $x^t$-computation in Line 5 of Algorithm 1 can be expressed as

$$
x^t \leftarrow \arg \min_{x \in \mathcal{X}} \{ (\nabla f(x^t), x) + u(x) + \eta \| x - x^{t-1} \|^2 / 2 \text{ s.t. } \nabla g(x^t)(x - x^t) + g(x^t) \leq 0 \}
$$

$$(a) \Rightarrow x^t \leftarrow \arg \min_{x \in \mathcal{X}} \{ (\nabla f(x^t) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x^t), x) + \alpha \| x \|^2 / 2 + \eta \| x - x^{t-1} \|^2 / 2 \}
$$

$$(b) \Rightarrow x^t \leftarrow \arg \min_{x \in \mathcal{X}} \| x - \frac{1}{\eta + \alpha} x^{t-1} - \frac{1}{\eta} (\nabla f(x^t) + \sum_{i=1}^{m} \lambda_i \nabla g_i(x^t)) \|^2
$$

$$
\text{for some } \gamma > 0,
$$

where $\lambda_i$’s in (a) is the optimal dual multiplier to the quadratic program and (b) follows from $X$ being radially invariant. Thus $\mathcal{M}_t = \mathcal{M}_{t-1} \cup \{x^t\}$ satisfies the requirement (3.1) and the ACGD method belongs to $\mathcal{F}$. Indeed, similar arguments can be used to show that $\mathcal{F}$ covers both primal methods such as [12, 10] and primal-dual methods such as [5, 22].

Since the dependence on parameters of a smooth objective function are well established, e.g. [16], we will investigate how the lower complexity bounds depend on the constraint functions. We will focus on the case with an affine $f$ in the objective, i.e. $L_f = 0$, and study the complexity’s dependence on the norm of the optimal Lagrange multiplier $\|\lambda^*\|$ and the Lipschitz smoothness constant of the vector-valued constraint function $g$, $L_g$, i.e., $\|\nabla g(x) - \nabla g(y)\| \leq L_g \| x - y \| \forall x, y \in \mathbb{R}^n$. These parameters are used more often for convergence analysis in the literature, but we will relate them to the aggregate smoothness constant $L(\Lambda_r)$ (2.9) for the ACGD method in the coming discussion.
3.1. Non-strongly Convex Case. The next theorem considers the non-strongly convex problem with \( \alpha = 0 \).

**Theorem 3.1.** Let problem parameters \( L_g > 0, R_0 \geq 1, l > 0, c \geq 1 \) and \( \epsilon > 0 \) be given. For a large enough problem dimension, \( n > 2\lceil R_0 \sqrt{L_g/c} \rceil \), there exists a hard constrained optimization problem of form (1.1) with \( \|x^*\| = l, \|x^0 - x^*\| \leq R_0 \), \( g \) being \( L_g \)-Lipschitz smooth, and \( f \) being linear, i.e., \( L_f = 0 \), such that every first order method in \( F \) requires at least \( \Omega(\sqrt{L_g c R_0 / \epsilon}) \) queries to the first order oracle to find an \((\epsilon; c)\)-optimal solution.

**Proof.** Consider the following function-constrained optimization problem parameterized by \( \gamma > 0, \beta > 0 \), and \( k \in \mathbb{N}_+ \):

\[
\begin{align*}
\min_{x \in \mathbb{R}^{2k+1}} & -2l\gamma x_1 \\
\text{s.t.} & \quad g_1(x) := \beta |x_1^2 + \sum_{i=1}^{2k} (x_i - x_{i+1})^2 + x_{2k+1}^2| - \frac{(2k+1)}{2k+2} \gamma^2 \beta \leq 0 \\
& \quad g_2(x) := \beta |\bar{x}_1^2 + \sum_{i=1}^{2k} (x_i - x_{i+1})^2 + x_{2k+1}^2| + \frac{(2k+1)}{2k+2} \gamma^2 \beta \leq 0,
\end{align*}
\]

where \( l \) is the given parameter in the theorem statement. Without loss of generality, we take \( x^0 = 0 \) and \( M_0 = 0 \). Let \( K_i \) denote the subspace with non-zeros in only the first \( i \)th coordinates, i.e., \( \{ x \in \mathbb{R}^{2k+1} : x_j = 0 \forall j > i \} \). Given a first-order method satisfying (3.1), it is easy to show inductively that \( M_t \subset K_i \forall t \in [2k+1] \). This is because \( G_i = \{ \nabla f(\bar{x}), \nabla g_1(\bar{x}), \nabla g_2(\bar{x}) : \bar{x} \in K_i \} \) are non-zero only in the first \( i + 1 \) coordinates, i.e., \( G_t \subset K_{i+1} \). Thus in \( k \) iterations, we have the following lower bound on feasibility violation:

\[
\|[g(x^k)]_1\| \geq \min_{x \in K_k} g_2(x) \geq (\frac{1}{2k+2}) \beta \gamma^2.
\]

Now we study various problem parameters associated with (3.2). It is straightforward to verify via the KKT condition that the optimal solution and the optimal dual multiplier are respectively:

\[
\lambda^* = [l, 0], \quad x_i^* = \gamma [1 - \frac{i}{2k+2}].
\]

So \( \|x^0 - x^*\| \leq \gamma \sqrt{k+1} \), \( \|\lambda^*\| = l \), and the constraint function \( g \) have a smoothness constant of \( 12 \beta \). By selecting \( k = \lfloor \frac{1}{2\sqrt{c}} R_0 \rfloor - 1 \), \( \beta = \tilde{L}_g/12 \), \( \gamma = R_0 / \sqrt{k+1} \), the problem satisfies the requirements in the theorem statement. Moreover, (3.3) implies that in \( k = \lfloor \frac{1}{2\sqrt{c}} R_0 \rfloor - 1 = \Omega(\sqrt{\frac{L_c}{c} R_0}) \) iterations, the feasibility violation are lower bounded by

\[
\|(g(x^k))_1\| \geq (\frac{1}{2k+2}) \beta \gamma^2 = (\frac{1}{2k+2})(12 \beta)(\gamma^2(k + 1))(\frac{1}{(k+1)^2}) \geq (\frac{1}{2k+2})(\tilde{L}_g) R_0^{2.5 \epsilon} \geq \frac{\epsilon}{\sqrt{c}}.
\]

This shows that \( \Omega(\sqrt{\frac{L_c}{c} R_0}) \) iterations are necessary for finding an \((\epsilon, c)\)-optimal solution. Since the choice of an method among \( F \) is arbitrary, the lower complexity bound is valid for all first order method in \( F \).

Comparing the above lower bound of \( \Omega(\sqrt{\frac{L_c}{c} R_0} \|x^* - x^0\|) \) to the upper bound of \( O(\sqrt{\frac{L(\Lambda_c)}{\epsilon} \|x^* - x^0\|}) \), we see the dependences of the oracle complexity of the ACGD method on \( \epsilon \) and \( \|\|x^* - x^0\|\) are not improvable. Only the dependence on the smoothness constant \( L_g \) might be sub-optimal. Specifically, the (big-O) factor of sub-optimality can be characterized by the following function of \( \epsilon \):

\[
H(\epsilon) := O(\sqrt{\frac{L(\Lambda_c)}{\epsilon} \epsilon}).
\]

Since we take \( L_f = 0 \) in this section, one useful inequality is \( L(\Lambda_c) \leq \tilde{L}_g(\|\lambda^*\| + c) \) (see (2.3)). The big-O relationship among these quantities is plotted in Figure 3.1.

Clearly \( H(\epsilon) \) has two distinct regions. When \( c \in [1, \|\lambda^*\|] \), we have \( 1 \leq H(\epsilon) \leq \sqrt{\|\lambda^*\| / \epsilon} \), i.e., the oracle complexity of the ACGD method can be sub-optimal with a factor of \( \sqrt{\|\lambda^*\| / \epsilon} \) in the worst case. However,
choose to focus on the convergence of $\min, \text{that is, they only have different constants inside the \"log\". So, with the loss of generality, we linear convergence can be expected, the difference in complexity bounds of different optimality criteria is

and

$f(1.1)$

dimensional hard constrained optimization problem of form

infeasible point with a smaller objective value. In that case, our ACGD method is tight. So we should perhaps always choose $c \geq \|\lambda^*\|$ to reflect the preference for a feasible point compared to an infeasible point with a smaller objective value. In that case, our ACGD method is tight.

3.2. Strongly Convex Case. Now we move on to the strongly convex problem with $\alpha > 0$. Since linear convergence can be expected, the difference in complexity bounds of different optimality criteria is minimal, that is, they only have different constants inside the \"log\". So, with the loss of generality, we choose to focus on the convergence of $\|x^t - x^*\|$ in the next theorem.

**Theorem 3.2.** Let problem parameters $\bar{L}_g > 0$, $l \geq 1$, and $0 \leq \alpha \leq \bar{L}_g l$ be given. There exists an infinite dimensional hard constrained optimization problem of form (1.1) with $\|\lambda^*\| = l$, $g$ being $\bar{L}_g$-Lipschitz smooth, and $f$ being linear, i.e., $L_f = 0$, such that every first order method in $\mathcal{F}$ requires at least $\Omega(\sqrt{\bar{L}_g l / \alpha \log(1/\epsilon)})$ queries to the first order oracle to find an $x^t$ with $\|x^t - x^*\|^2 \leq \epsilon$ for all $\epsilon > 0$.

**Proof.** Let $\gamma = \alpha/(\bar{L}_g l)$, $\Delta = (1 - \sqrt{\gamma})/(1 + \sqrt{\gamma})$ and $\bar{x} = [\Delta, \Delta^2, \ldots, \Delta^i, \ldots] \in \mathbb{R}^\infty$. Consider the following hard problem:

$$\min_{x \in \mathbb{R}^\infty} - \frac{\beta - \alpha}{4} x_1 + \frac{\alpha}{2} \|x\|^2$$

s.t. $h(x) - h(\bar{x}) \leq 0$ with $h(x) := \frac{\beta - \alpha}{8} [x_1^2 + \sum_{i=1}^{\infty} (x_i - x_{i+1})^2]$.

Clearly, the objective has a strong convexity modulus of $\alpha$. It is straightforward to verify that $\lambda^* = l$ and $x^* = \bar{x}$ satisfies the KKT condition:

$$g(x^*) = 0 \text{ and } \frac{\beta - \alpha}{4} \begin{bmatrix} 2(1 + \gamma) & -1 & 2(1 + \gamma) & -1 & \cdots \\ -1 & 2(1 + \gamma) & -1 & \cdots \\ \vdots & \vdots & \ddots & \ddots \\ \vdots & \vdots & \ddots & \ddots \end{bmatrix} \begin{bmatrix} x_1^2 \\ x_2^2 \\ x_3^2 \\ \vdots \\ \vdots \end{bmatrix} = \frac{\beta - \alpha}{4} \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix}.$$
Starting from $\mathcal{M}_0 = \{0\}$, the solution $x_t$ generated by any first order method satisfying (3.1) in $t$ iterations have non-zeros in only the first $t$ coordinates. Thus we have

$$\|x^t - x^*\|^2 \geq \sum_{i=1}^{\infty} (x^t_i)^2 = \left(\frac{1 - \sqrt{\gamma}}{1 + \sqrt{\gamma}}\right)^{2t} \sum_{i=1}^{\infty} (\frac{1 - \sqrt{\gamma}}{1 + \sqrt{\gamma}})^{2i} = \left(\frac{1 - \sqrt{\gamma}}{1 + \sqrt{\gamma}}\right)^{2t} \|x^0 - x^*\|^2 \geq (1 - \sqrt{\gamma})^{2t} \|x^0 - x^*\|^2,$$

(3.5)

where the last inequality follows from $\gamma \leq 1$. Therefore we require at least $t = \Omega(\sqrt{Lg}/\alpha \log(1/\epsilon))$ iterations to find $x^t$ with $\|x^t - x^*\|^2 \leq \epsilon$.

For $\|\lambda^*\| \geq 1$, the preceding theorem implies that the $\mathcal{O}(\sqrt{L(L_1)/\alpha \log(1/\epsilon)})$ upper complexity bound in Corollary 2.5 is not improvable since $L(L_1) \leq (1 + \|\lambda^*\|)L_g = \mathcal{O}(\|\lambda^*\| L_g)$. Notice that the optimal condition number is indeed dependent on $\|\lambda^*\|$. So, unlike the non-strongly convex case where the ACGD method is optimal only when $c \geq \|\lambda^*\|$, the ACGD method is always optimal in the strongly convex case.

4. The ACGD-S method. Despite its optimal oracle complexities, the ACGD method may be lacking in computation efficiency for large scale problems where both the problem dimension $n$ and the number of constraints $m$ are large. The bottleneck to Algorithm 1 is in Line 5:

$$x^t \leftarrow \arg\min_{x \in X} \langle \pi^t, x \rangle + u(x) + \eta_t \|x - x^{t-1}\|^2 / 2 \quad \text{s.t.} \quad \nu^t_i(x - x^t_i) + g_i(x^t_i) \leq 0 \forall i \in [m].$$

(4.1)

It involves a quadratic program (QP) if $X$ is linearly constrained, say a box, and a quadratic constrained quadratic program (QCQP) if $X$ is an Euclidean ball. So computing the exact $x^t$ to (4.1) could be prohibitively expensive under the large-scale setting. In this section, we address the bottleneck by replacing the large scale quadratic program with a sequence of basic matrix-vector operations requiring only $\mathcal{O}(mn)$ FLOPs. The proposed method, termed the accelerated nested Lagrangian algorithm with sliding (ACGD-S) method, is able to achieve the same oracle complexity as the ACGD method. It is also computationally efficient in the sense that it requires the same order of matrix-vector operations as solving a single linear constrained problem, that is, $g(x)$ is an affine function in (1.1), to an $\epsilon$-accuracy.

Towards that end, we need to assume for this section the projection onto $X$ is easy, i.e., the following operation can be computed with $\mathcal{O}(n)$ FLOPs for any $\pi, \bar{x} \in \mathbb{R}^n$ and $\eta_t \geq 0$.

$$x^t \leftarrow \arg\min_{x \in X} \langle \pi, x \rangle + u(x) + \eta_t \|x - \bar{x}\|^2 / 2.$$  

(4.2)

For instance, if $u(x) = \alpha \|x\|^2 / 2$, the computation simplifies, respectively, to component-wise thresholding or vector scaling if $X$ is a box or an Euclidean ball. If $X$ is more complicated but admits function-constrained representation $h(x) \leq 0$, $h(x)$ can be included as a part of $g$ in (1.1).

4.1. The ACGD-S Method and its Convergence Results. The ACGD-S method is listed in Algorithm 4 It has two loops. For clarity, we will call an outer iteration a phase, and an inner iteration a iteration for the rest of rest section. The outer loop is indexed by $t$. In phase $t$, the outer updates happen in Line 3, 4, and 12: they are identical to the ACGD method except that $x^t$ is computed instead as some average of inner iterates. The others, Line 5-11, constitute the sliding subroutine. Its goal is to solve the Lagrangian reformulation to (4.1), or the $(\lambda^t, x^t)$ saddle point problem in (2.8), inexacty:

$$(x^t, \lambda^t) \leftarrow \arg\min_{y \in X} \arg\max_{\lambda \in \mathbb{R}^m} \langle \pi^t, y \rangle + \langle \lambda, \nu^t y - x^t \rangle + u(y) + \eta_t \|y - x^{t-1}\|^2 / 2.$$ 

(4.3)

To avoid confusion, we use the dummy variable $y$ to emphasize it being used only in the inner loop. Specifically, Line 5 calculates the stepsizes parameters and iteration number $S_t$. Line 7-9 carry out primal-dual type updates for $S_t$ iterations: a momentum extrapolation term $\tilde{h}^{(t)_s}$ in Line 7, the primal proximal mapping of $y^{(t)}$ with respect to $h^{(t)}$ in Line 8 and the dual proximal mapping of $\lambda^{(t)}$ with respect to $y^{(t)}$ in Line 9. After that, Line 11 prepares the initialization points for the inner loop in the next phase.

We highlight a few features of the ACGD-S method essential for achieving the desired computation efficiency. First, rather than being pre-specified, the inner loop step-size parameters and iteration limit $S_t$ are calculated in an online fashion in Line 5. This allows the method to adjust dynamically to the difficulty.
Algorithm 3 The ACGD-S Method

Input: $x^{-1} = x^0 = y^0(1) = x^0 \in X$, stepsizes $\{\theta_t\}$, $\{\eta_t\}$, $\{\tau_t\}$, and weights $\{\omega_t\}$.

1. Set $\pi^0 = \nabla f(x^0)$, $\nu^0 = \nabla g(x^0)$, $\lambda_{0(1)}^0 = 0$.
2. for $t = 1, 2, \ldots, N$ do
   3. Set $x^t \leftarrow (\pi^t x^{t-1} + \tilde{x}^t)/(1 + \tau_t)$ where $\tilde{x}^t = x^{t-1} + \theta_t(x^{t-1} - x^{t-2})$.
   4. Set $\pi^t \leftarrow \nabla f(\pi^t)$ and $\nu^t \leftarrow \nabla g(\nu^t)$.
   5. Calculate inner loop iteration limit $\{S_t\}$ stepsizes $\{\beta_s(t)\}$ and $\{\gamma_s(t)\}$, and weights $\{\delta(t)\}$.
   6. for $s = 1, 2, \ldots, S_t$ do
      7. Set $\tilde{h}(t,s) = \begin{cases} (\nu^t)\pi_0^0 + \rho_1^0(\nu^{t-1})\pi_0^0 + \lambda_0^0 - \lambda_{1-1}^0), & \text{if } s = 1 \\ (\nu^t)\pi_{s-1}^0 + \rho_1^0(\nu^{t-1})\pi_{s-1}^0 + \lambda_{s-1}^0 - \lambda_{s-2}^0), & \text{o.w.} \end{cases}$
      8. Set $y_s(t) \leftarrow \arg\min_{y \in X} \tilde{h}(t,s) + \pi^t x^t + u(y) + \eta_t \frac{y - y^{t-1}}{2} + \beta_s(t) \frac{y - y_{s-1}}{2}$.
      9. Set $\lambda_s(t) = \arg\max_{\lambda \in \mathbb{R}^p} \langle \lambda, \nu^t(y_s(t) - x^t) + g(y_s(t)) \rangle + \gamma_s(t) \frac{\lambda - \lambda_{s-1}^0}{2}$.
   10. end for
   11. Set $\lambda_{s(t+1)}^t = \lambda_{s(t-1)}^t = \lambda_{s(t)}^t$, $y_0(t) = y_{S_t}$
   12. Set $x^t = \sum_{s=1}^{S_t} \delta_s(t) y_s(t)/(\sum_{s=1}^{S_t} \delta_s(t))$ and $\lambda^t = \sum_{s=1}^{S_t} \delta_s(t) \lambda_s(t)/(\sum_{s=1}^{S_t} \delta_s(t))$.
   13. end for
14. return $\bar{x}^N := \sum_{t=1}^{N} \omega_t x^t/(\sum_{t=1}^{N} \omega_t)$.

of the current saddle point problem (4.3) in each phase. Second, the last operator $\nu^{t-1}$, rather than $\nu^t$, is used for calculating the momentum extrapolation term at the first iterations $s = 1$ in Line 7. This is characteristic of the sequential dual type algorithms [24, 25, 11] for solving the trilinear saddle point problem (2.4). Third, two primal iterates, $x^t$ and $y_{S_t}(t)$, are stored after the inner loop in each phase to help the next inner loop. This is common to sliding algorithms [7, 10, 11].

Now we suggest certain stepsize choices to obtain convergence rates for Algorithm 3. Their convergence proofs are deferred to Subsection 4.3. The next theorem consider the non-strongly convex case and the strongly convex case separately.

**Theorem 4.1.** Consider a non-strongly convex function constrained problem of form (1.1). Let the aggregate smoothness constant $L(\Lambda)$ and the reference multiplier set $\Lambda$, be defined in (2.4) and (2.6) respectively. Suppose Algorithm 3 is performed with the following stepsize. The outer-loop stepsize are

$$
\tau_t = \frac{1}{2} \quad \eta_t = \frac{L(\Lambda)}{\tau_{t+1}}, \quad \omega_t = t, \quad \theta_{t+1} = \omega_{t+1}/\omega_t \quad \forall t \geq 1.
$$

(4.4)

Additionally, with $M_t = \|\nu^t\|$ and $d(\Lambda_r) = \|\lambda^t\| + r$, the inner loop parameters in phase $t$ are calculated according to

$$
S_t = \lfloor M_t \Delta t \rfloor, \quad \bar{M}_t = \frac{S_t}{\Delta t}, \quad \rho_s^t(t) = \begin{cases} \bar{M}_t, & \text{if } s = 1 \\ 1, & \text{if } s \geq 2, \end{cases} \quad \beta_s^t(t) = \frac{\bar{M}_t d(\Lambda)}{\|x^t - x^t\|} \quad \gamma_s^t(t) = \frac{\bar{M}_t^2}{\rho_s^t(t)} \quad \delta_s^t = 1 \quad \forall s \geq 1,
$$

(4.5)

where $\Delta > 0$ is some fixed stepsize parameter. Then $M_t \leq \bar{M} \forall t$, where $\bar{M}$ is an upper bound for $\|\nabla g(x)\|$ for $x$ in some bounded ball around $x^*$. And the ergodic average solution $\bar{x}^N$ satisfies

$$
\max\{f(\bar{x}^N) - f(x^*), c \|g(\bar{x}^N)\|\} \leq \frac{1}{N(N+1)} \left( \frac{2d(\Lambda)}{\Delta} \right) \Delta + L(\Lambda) \|x^0 - x^*\|^2.
$$

(4.6)

**Corollary 4.2.** Under the setting of Theorem 4.1, if $\Delta = \frac{d(\Lambda)}{\|x^0 - x^*\| L(\Lambda)}$, the numbers of operations are required by the ACGD-S method to find an $(\epsilon; \epsilon/\epsilon) \text{-optimal solution are bounded by:}$

- $N_e = \lceil \frac{2d(\Lambda)}{\epsilon} \|x^0 - x^*\| \rceil$ FO-oracle evaluations.
- $C_e = \mathcal{O}\left( \left\lceil \frac{d(\Lambda) \Delta}{\epsilon} \|x^0 - x^*\| \right\rceil \right)$ matrix-vector multiplications.
Three remarks are in order regarding the above results. First, the oracle complexity of the ACGD-S method matches that of the ACGD method, while its computation complexity, measured by the number of matrix-vector multiplications, matches the lower bound for solving a single linearly constrained problem [19]. Second, the stepsize choice in (4.3) and (4.5) only requires the estimate of $L(\Lambda_r)$ to be accurate. Misspecification of $\|x^0 - x^*\|$ or $d(\Lambda_r)$ or $\Delta$ would still lead to an $O(1/\sqrt{r})$ oracle complexity and an $O(1/\epsilon)$ computation complexity. Third, the iteration limit function $S_t$ in (4.4) in an online fashion adapts to the difficulty of the saddle point sub-problem (4.3) from each phase. Since a large $\|\nu\|$ implies that the subproblem is harder to solve, and a large $t$ implies a higher accuracy is expected of the inexact solution, $S_t$ scales in proportion to both parameters.

**Theorem 4.3.** Consider an $\alpha$-strongly convex function constrained problem of form (1.1). Let the aggregate smoothness constant $L(\Lambda_r)$ and the reference multiplier set $\Lambda_r$ be defined in (2.4) and (2.6) respectively, and let $\kappa_r := L(\Lambda_r)/\alpha$ be the condition number. Suppose Algorithm 3 is performed with the following stepizes. The outer-loop stepizes are

$$\tau_t = \min\{\frac{t}{\tau_t}, \sqrt{2K_r}\}, \eta_t = \frac{L(\Lambda_r)}{\tau_t+1}, \text{and } \theta_t = \frac{\tau_t}{\tau_t+1} \forall t \geq 1. \quad \omega_t = \begin{cases} \omega_{t-1}/\theta_t & \text{if } t \geq 2, \\ 1 & \text{if } t = 1. \end{cases} \quad (4.7)$$

For the inner loops, let $M_t = \|\nu\|$, and let $\Delta > 0$ denote a given step-size parameter. At the beginning, the iteration limit is set to $S = \min\{S \in \mathbb{N}_+: \sum_{j=1}^S s \geq \omega_1 M_t^2 \Delta\}$, and the stepizes for all $s \in [S]$ are set to

$$\delta_s^{(1)} = \frac{\eta_t}{\alpha}, \quad \delta_s^{(1)} = \frac{\eta_t}{\alpha}, \quad \delta_s^{(1)} = \frac{\eta_t}{\alpha}, \quad \delta_s^{(1)} = \frac{\eta_t}{\alpha}, \quad W_2 = \frac{M_t^2}{\delta_s^{(1)}}. \quad (4.8)$$

where $\Gamma_1$ is the non-negative root to $\sum_{s=1}^{S_t} s = \Gamma_1^2(\omega_1 M_t^2 \Delta)$. Then for phase $t \geq 2$, the iteration limit $S_t$ and the parameter $\Gamma_1 \geq 0$ are specified to satisfy

$$S_t = \min\{S \in \mathbb{N}_+: \sum_{j=1}^{S_t} W_j M_j + (s-1) \geq \omega_t M_t^2 \Delta, \sum_{s=1}^{S_t} \Gamma_j (W_j M_j + (s-1) = \Gamma_1^2 \omega_t M_t^2 \Delta, \text{and the stepizes are chosen according to}\}

$$\delta_s^{(t)} = \frac{\eta_t}{\alpha}, \quad \delta_s^{(t)} = \frac{\eta_t}{\alpha}, \quad \delta_s^{(t)} = \frac{\eta_t}{\alpha}, \quad \delta_s^{(t)} = \frac{\eta_t}{\alpha}, \quad W_{t+1} = \frac{M_t^2}{\delta_s^{(t)}}. \quad (4.9)$$

Then we have $M_t \leq \bar{M} \forall t$, where $\bar{M}$ is an upper bound for $\|\nabla g(x)\|$ for $x$ in some bounded ball around $x^*$. The ergodic average solution $\bar{x}_N$ satisfies

$$\max\{f(\bar{x}_N) - f(x^*), \|g(\bar{x}_N)\|\}, \|\bar{x}_N - x^*\|^2 \leq \frac{2}{W_N}(\frac{\alpha}{\alpha\Delta}[d(\Lambda_r)]^2 + L(\Lambda_r)\|x^0 - x^*\|^2), \quad (4.10)$$

where $d(\Lambda_r) = \|\lambda^*\| + r$ and the denominator satisfies $W_N \geq \max\{N(N+1)/2, \sqrt{2\kappa_r}(1 + 1/\sqrt{2\kappa_r})^{-1}\}$.

**Corollary 4.4.** Under the setting of Theorem 4.3, if $\Delta = \frac{|d(\Lambda_r)|^2}{L(\Lambda_r)\|x^0 - x^*\|^2\alpha}$, the numbers of FO-oracle evaluations $N_r$ and of matrix-vector multiplications $C_r$ required by the ACGD-S method to find an $(\epsilon; \epsilon/c)$-optimal solution are bounded by:

- $N_r \leq \min\{\sqrt{\frac{4L(\Lambda_r)}{\epsilon}}\frac{\max\{c/r, 1\}}{\alpha}, \sqrt{\frac{2M(\Lambda_r)}{\kappa_r}} + 1 \log(\frac{3\max\{c/r, 1\}}{\epsilon}\frac{\max\{c/r, 1\}}{\alpha} + 1) + 1\}.$
- $C_r = \mathcal{O}\{\sqrt{\frac{\max\{c/r, 1\}}{\alpha\epsilon}}[d(\Lambda_r)]^2 \|x^0 - x^*\|^2\} + N_r\}.$

**Corollary 4.5.** Under the setting of Theorem 4.3, if the goal is to find an $\epsilon$-close solution, i.e., $\|\bar{x}_N - x^*\| \leq \epsilon$, we can choose $c = r = 1$ and $\Delta = \frac{|d(\Lambda_r)|^2}{L(\Lambda_r)\|x^0 - x^*\|^2\alpha}$ such that the numbers of required operations can be bounded by
• $N_t = \mathcal{O}\left(\sqrt{2L(\Lambda^t)} \log(\frac{\sqrt{\kappa_1}\|x^0 - x^*\|^2}{\epsilon})\right)$

• $C_t = \mathcal{O}\left(\frac{d(\Lambda^t)M}{\alpha \sqrt{\epsilon}} + \sqrt{\frac{2L(\Lambda^t)}{\alpha}} \log(\frac{\sqrt{\kappa_1}\|x^0 - x^*\|^2}{\epsilon})\right)$.

Again, we make three remarks regarding the results. First, to find an $\epsilon$-close solution, Corollary (4.5) implies that the ACGD-S method has the same oracle complexity as the ACGD method, and has the same computation complexity as that of the lower computation complexity bound for solving a single strongly-convex linearly constrained problem [19]. Second, the iteration limit function $S_t$ is again adaptive to the difficulty of the saddle-point subproblem (4.3) in each phase. Third, the rather complicated inner-loop stepsize choice in (4.8) and (4.9) is the first among sliding algorithms, e.g., [7, 9, 10, 11], to achieve both optimal inner and optimal outer complexities for solving smooth and strongly convex type problems without restarting. It is unclear if the same effect is achievable with simpler stepsize choices. Since $\mathcal{M}_t = \|\nu^t\|$ can be calculated in an online fashion, only the prior knowledge of $\alpha$ and $L(\Lambda_t)$ is required during implementation.

4.2. Online Search for $L(\Lambda_t)$ and $d(\Lambda_t)$. To enhance its implementability, we propose a binary search scheme to find (upper) approximations to these hard-to-estimate problem parameters required by the ACGD-S method. Since both quantities are proportional to $\Lambda_t$, it is convenient to look for a common upper bound:

$$\tilde{H} \geq \max\{L(\Lambda_t), d(\Lambda_t)\}.$$ 

The search scheme has the same logic as Algorithm 2, given a guess $\tilde{H}$, we first run the ACGD-S method for $N(\tilde{H})$ phases, with stepsizes specified according to $\tilde{H}$, to generate $\bar{x}^{N(\tilde{H})}$, then we check if $\bar{x}^{N(\tilde{H})}$ satisfies certain termination criterion, and update our guess $\tilde{H}$ accordingly. The following assumption is required by the termination criterion.

ASSUMPTION 3. The feasible region $X$ is bounded and we know an estimate $D_X$ of its radius, i.e., $D_X \geq \max_{x,y \in X} \|x - y\|$.

Compared to Assumption 2, we no longer require the access to the dual solution $\lambda^t$ (Line 5 of Algorithm 1). This is because the sliding subroutine (Line 12 of Algorithm 3) provides a feasible dual iterate $\lambda^t$ automatically. So the new termination criterion, termed (2.19)-(4.11), requires the following modification to (2.19)

$$\text{Change } \lambda^t \text{ in (2.20) to } \lambda^t_0. \quad (4.11)$$

Clearly, $f^N$ in (2.19)-(4.11) bounds $f^*$ from below, so the criterion can be satisfied only if the point $\bar{x}^{N(\tilde{H})}$ is $(\epsilon, \epsilon/c)$-optimal. Accordingly, we modify Algorithm 2 by switching the ACGD-S method and the termination criterion (2.19)-(4.11) to arrive at Algorithm 4.

Algorithm 4 Binary Search Outer Loop to ACGD-S.

**Input:** $\bar{x} \in X$, $\tilde{H} > 0$, and a phase-limit function $N(\tilde{H})$.

1. **while** true **do**
   2. With $D_X$, $\tilde{H}$, and $\tilde{H}$ in place of $\|x^0 - x^*\|$, $L(\Lambda_t)$, and $d(\Lambda_t)$, set stepsizes to (4.3) and (4.4) if $\alpha = 0$, and to (4.7), (4.8) and (4.9) if $\alpha > 0$.
   3. Run the ACGD-S method for $N(\tilde{H})$ phases starting from $\bar{x}$.
   4. Set $\bar{x} \leftarrow \bar{x}^{N(\tilde{H})}$, where $\bar{x}^{N(\tilde{H})}$ is the ergodic average solution.
   5. **if** $\bar{x}$ satisfies the condition (2.19)-(4.11) **then**
      6. **break**
   7. **else**
      8. Set $\tilde{H} \leftarrow 2\tilde{H}$.
   9. **end if**
10. **end while**
11. **return** $\bar{x}$. 

19
Clearly, the above algorithm terminates with an \((\epsilon, \epsilon/c)\)-optimal solution. We only need to specify the phase-limit function \(N(H)\) to make it implementable. The next theorem provides some choices of \(N(H)\) and the corresponding complexities. Its derivation is similar to Theorem 2.6.

**Theorem 4.6.** Consider a smooth constrained optimization problem of form (1.1). Assume assumption 3 holds and \(M := \max_{x \in X} \|\nabla g(x)\|\). Suppose the reference set \(\Lambda_r\) is defined according to (2.6), and that \(L(\Lambda_r)\) and \(d(\Lambda_r) = \|\lambda^*\| + r\) denote, respectively, the aggregate smoothness constant \(2.9\) and the radius. Let \(\hat{H}\) being initialized to some \(\hat{H}_0\), then Algorithm 4 terminates finitely with an \((\epsilon, \epsilon/c)\)-optimal solution under the following choices of phase limit function \(N(H)\).

a) In the non-strongly convex case with \(\alpha = 0\), if \(N(H) = \lceil \sqrt{\frac{3L}{\epsilon}} D_X \rceil\), the total numbers of oracle evaluations (phases) and matrix-vector multiplications required for termination are bounded respectively by

\[
O(\sqrt{\frac{\max\{L(\Lambda_s), d(\Lambda_s), \hat{H}_0\}}{\epsilon} D_X}), \quad \text{and} \quad O\left(\max\{L(\Lambda_s), d(\Lambda_s), \hat{H}_0\} D_X M\right).
\]

b) In the strongly convex case with \(\alpha > 0\), if \(N(H) = \lceil (\sqrt{\frac{2M}{\alpha}} + 1) \log(\frac{3\max\{c, 1\} \sqrt{M\sigma D_X^2} + 1}{\epsilon}) \rceil + 4\), the total numbers of oracle evaluations (phases) and matrix-vector multiplications required for termination are bounded respectively by

\[
O(\frac{\max\{\hat{H}_0, L(\Lambda_1), d(\Lambda_1)\}}{\alpha} \log \frac{\max\{\hat{H}_0, L(\Lambda_1), d(\Lambda_1)\}}{\epsilon}), \quad \text{and} \quad O\left(\frac{\sqrt{\max\{\hat{H}_0, L(\Lambda_1), d(\Lambda_1)\}}}{\sqrt{\alpha}} D_X \right).
\]

We remark that both the oracle complexity and the computation complexity across the multiple ACGD-S runs in Algorithm 4 have the optimal order for all \(\alpha \geq 0\). If the Lipschitz smoothness constants \(L_f\) and \(L_g\) for \(f\) and \(g\) are known, we can search instead for \(d(\Lambda_r) = \|\lambda^*\| + r\) and utilize \(L_f + d(\Lambda_r) L_g\) as an upper bound to \(L(\Lambda_r)\), then the resulting search scheme would have the same complexities, with matching constant dependences, as those of Corollary 4.2 and 4.3.

**4.3. The Convergence Analysis.** We first prove a generic result for the \(Q\)-function (2.7) useful for both the non-strongly convex and the strongly convex cases.

**Proposition 4.7.** Consider an \(\alpha\)-strongly convex function-constrained optimization problem of form (1.1). Let a set of reference multipliers \(\Lambda \in \mathbb{R}^n_m\) be given and let the aggregate smoothness constant \(L(\Lambda)\) be defined in (2.9). Let iterates \(z^t := \{x^t, \lambda^t, \nu^t, \pi^t\}\) be generated by Algorithm 3. Suppose the following stepsize requirements are met. For all \(t \geq 1\), the outer-loop stepsize requirements are

\[
\omega_t \eta_t \leq \omega_{t-1}(\eta_{t-1} + \alpha/2),
\]

\[
\omega_t \tau_t \leq \omega_{t-1}(\tau_{t-1} + 1),
\]

\[
\eta_N(\tau_N + 1) \geq L(\Lambda), \quad \eta_{t-1} \tau_t \geq \theta_t L(\Lambda) \quad \text{with} \quad \theta_t := \omega_{t-1}/\omega_t.
\]

For all \(t \geq 1\), the intra-phase stepsize requirements are

\[
\delta_s^{(t)}(\beta_s^{(t)} + \alpha/2) \geq \delta_{s+1}^{(t)} \beta_{s+1}^{(t)},
\]

\[
\delta_s^{(t)} \gamma_s^{(t)} \geq \delta_{s+1}^{(t)} \gamma_{s+1}^{(t)},
\]

\[
\gamma_s^{(N)}(\beta_s^{(N)} + \alpha/2) \geq \|\nu^{N}\|^2, \quad \gamma_s^{(t)} \beta_s^{(t)} \gamma_{s+1}^{(t)} \geq \rho_{s+1}^{(t)} ||\nu^{t}||^2, \quad \rho_{s+1}^{(t)} = \delta_s^{(t)}/\delta_{s+1}^{(t)}.
\]

For all \(t \geq 1\), the inter-phase stepsize requirements are

\[
\hat{w}_s^{(t)}(\beta_s^{(t)} + \alpha/2) \geq \hat{w}_1^{(t+1)} \beta_1^{(t+1)},
\]

\[
\hat{w}_s^{(t)} \gamma_s^{(t)} \geq \hat{w}_1^{(t+1)} \gamma_1^{(t+1)},
\]

\[
\gamma_s^{(t)} \beta_1^{(t+1)} \geq \rho_1^{(t+1)} ||\nu^{t-1}||^2, \quad \rho_1^{(t+1)} = \hat{w}_s^{(t)}/\hat{w}_1^{(t+1)};
\]

\[
(4.14)
\]
where \( \bar{w}_s^{(t)} := \omega_1 \delta_s^{(t)} / (\sum_{s=1}^{S_t} \delta_s^{(t)}) \) denotes the aggregate weights. Then for any \( z = (x; \lambda, \nu, \pi) \) any reference point \( z = (x; \lambda, \nu, \pi) \in X \times \Lambda \times [V, \Pi] \) with \([V, \Pi]\) being defined in (2.3), we have
\[
\sum_{i=1}^{N} \omega_i Q(z^i; z) + \frac{\omega_N}{T}(\eta_N + \frac{\omega}{T}) \|x^N - x\|^2 \leq \frac{\bar{w}_s^{(t)} \rho_s^{(t)} + \omega_1 \eta_1}{2} \|x^0 - x\|^2 + \frac{\bar{w}_s^{(t)} \gamma_s^{(t)}}{2} \|\lambda_s^{(t)} - \lambda\|^2
\]
\[+ \omega_1 \tau_1[U_f,(\pi; \pi^0) + \lambda^T U_f,(\nu; \nu^0)]]. \tag{4.15} \]

Proof. We first establish a convergence bound the inner loop within a phase. Fix \( t \geq 1 \). Consider the convergence of \( y_s^{(t)} \). Since \( u(y) + \eta_t \|y - x^{t-1}\|^2 / 2 \) has a strong convexity modulus of \( \alpha + \eta_t \), the \( y \)-proximal mapping in Line 8 of Algorithm 3 leads to a three point inequality (see Lemma 3.1 of [8]):
\[
\langle y_s^{(t)} - x, \tilde{h}^{(t),x} \rangle + u(y_s^{(t)}) - u(x) + \frac{\eta_t}{2} \left( \|y_s^{(t)} - x^{t-1}\|^2 - \|x - x^{t-1}\|^2 \right)
\]
\[+ \frac{1}{2} \left( (\beta_s^{(t)} + \alpha + \eta_t) \|y_s^{(t)} - x\|^2 + \beta_s^{(t)} \|y_s^{(t)} - y_s^{(t-1)}\|^2 - \beta_s^{(t)} \|y_s^{(t-1)} - x\|^2 \right) \leq 0. \tag{4.16} \]

Equivalently, we have
\[
\langle y_s^{(t)} - x, \tilde{h}^{(t),x} \rangle + \frac{1}{2} \left( (\beta_s^{(t)} + \alpha + \eta_t) \|y_s^{(t)} - x\|^2 + \beta_s^{(t)} \|y_s^{(t)} - y_s^{(t-1)}\|^2 - \beta_s^{(t)} \|y_s^{(t-1)} - x\|^2 \right)
\]
\[+ u(y_s^{(t)}) - u(x) + \frac{1}{2} \eta_t \left( y_s^{(t)} - x^{t-1}\| - \right) (\eta_t + \alpha / 2) \|y_s^{(t)} - x\|^2 - \eta_t \|x - x^{t-1}\|^2 \leq 0. \tag{4.16} \]

In particular, the definition of \( \tilde{h}^{(t),x} \) in Line 7 of Algorithm 3 implies
\[
\langle y_s^{(t)} - x, \tilde{h}^{(t),x} \rangle = (y_s^{(t)} - x, \sum_{i=1}^{m} \lambda^{(t)}_{s,i} \nu_i^{(t)}) - (y_s^{(t)} - x, \sum_{i=1}^{m} (\lambda^{(t)}_{s,-i} - \lambda^{(t)}_{s,-i} \nu_i^{(t)})
\]
\[+ \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} (\lambda^{(t)}_{s,-i} - \lambda^{(t)}_{s,-i} \nu_i^{(t)})
\]
\[+ \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} (\lambda^{(t)}_{s,-i} - \lambda^{(t)}_{s,-i} \nu_i^{(t)})
\]
\[= \langle y_s^{(t)} - x, \sum_{i=1}^{m} \lambda^{(t)}_{s,i} \nu_i^{(t)} \rangle - \langle y_s^{(t)} - x, \sum_{i=1}^{m} \lambda^{(t)}_{s,-1} \nu_i^{(t)} \rangle + \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} \lambda^{(t)}_{s,-i} \nu_i^{(t)}
\]
\[+ \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} (\lambda^{(t)}_{s,i} - \lambda^{(t)}_{s,-1} \nu_i^{(t)})
\]
\[+ \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} (\lambda^{(t)}_{s,i} - \lambda^{(t)}_{s,-1} \nu_i^{(t)})
\]
\[= \langle y_s^{(t)} - x, \sum_{i=1}^{m} \lambda^{(t)}_{s,i} \nu_i^{(t)} \rangle - \langle y_s^{(t)} - x, \sum_{i=1}^{m} \lambda^{(t)}_{s,-1} \nu_i^{(t)} \rangle + \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} \lambda^{(t)}_{s,-i} \nu_i^{(t)}
\]
\[+ \rho^{(t)}_s (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} (\lambda^{(t)}_{s,i} - \lambda^{(t)}_{s,-1} \nu_i^{(t)})
\]

So, substituting them into (4.16), summing up the resulting inequality with weight \( \delta_s^{(t)} \), noting the stepsizes conditions in (4.13), and utilizing Young’s inequality, we get
\[
\sum_{s=1}^{S_t} \delta_s^{(t)} \left( L(y_s^{(t)}; \lambda^{(t)}, \nu^t, \pi^t) - L(x; \lambda^{(t)}, \nu^t, \pi^t) + \frac{\eta_t}{2} \|y_s^{(t)} - x^{t-1}\|^2 + \frac{\eta_t}{2} \|y_s^{(t)} - x\|^2 \right)
\]
\[+ \delta_s^{(t)} \rho_s^{(t)} (y_s^{(t)} - x, \sum_{i=1}^{m} (\lambda^{(t)}_{s,i} - \lambda^{(t)}_{s,-1} \nu_i^{(t)}) - \beta_s^{(t)} (y_s^{(t)} - y_s^{(t-1)}, \sum_{i=1}^{m} (\lambda^{(t)}_{s,i} - \lambda^{(t)}_{s,-1} \nu_i^{(t)})
\]
\[\leq \sum_{s=1}^{S_t} \delta_s^{(t)} \frac{\beta_s^{(t)}}{2} \|\lambda^{(t)}_{s-1} - \lambda^{(t)}_{s-2}\|^2 + \frac{\delta_s^{(t)} \rho_s^{(t)} }{2} \|\lambda^{(t)} - \lambda^{(t-1)}\|^2
\]
\[+ \frac{\delta_s^{(t)}}{2} \|\beta_s^{(t)} \| \cdot \eta_t \|x - x^{t-1}\|^2 - \beta_s^{(t)} \|y_s^{(t)} - x\|^2 \leq \delta_s^{(t)} \gamma_s^{(t)} \|\lambda^{(t)} - \lambda^{(t-1)}\|^2 \leq 0. \tag{4.17} \]

Next, consider the convergence of \( \lambda_s^{(t)} \). The \( \lambda \)-proximal mapping in Line 9 of Algorithm 3 implies
\[
L(y_s^{(t)}; \lambda, \nu^t, \pi^t) - L(y_s^{(t)}; \lambda^{(t)}, \nu^t, \pi^t) + \frac{\gamma_t}{2} \left( \|\lambda - \lambda^{(t)}\|^2 + \|\lambda^{(t)} - \lambda^{(t-1)}\|^2 - \|\lambda - \lambda^{(t-1)}\|^2 \right) \leq 0.
\]

Due to the stepsizes conditions in (4.13), the \( \delta_s^{(t)} \) weighted sum satisfies
\[
\sum_{s=1}^{S_t} \delta_s^{(t)} \left( L(y_s^{(t)}; \lambda, \nu^t, \pi^t) - L(y_s^{(t)}; \lambda^{(t)}, \nu^t, \pi^t) + \frac{\gamma_t}{2} \|\lambda - \lambda^{(t)}\|^2 \right)
\]
\[+ \sum_{s=1}^{S_t} \delta_s^{(t)} \gamma_s^{(t)} \|\lambda^{(t)} - \lambda^{(t-1)}\|^2 \leq \delta_1 \gamma_1 \|\lambda - \lambda^{(t)}\|^2.
\]

21
Then, combining it with the \( y \) convergence bound in (4.17), we get

\[
\sum_{s=1}^{S_t} \delta_s(t) \left( L(y^{(t)}_s; \lambda, \nu^t, \pi^t) - L(x; \lambda^t, \nu^t, \pi^t) + \frac{\eta}{2} \|y^{(t)}_s - x^{t-1}\|^2 + (\eta + \alpha/2) \|y^{(t)}_s - x\|^2 \|x^{t-1} - x\|^2 \right) \\
+ \delta_1(t) \rho_1(t) (y_0 - x, \sum_{i=1}^m \nu_{i}^{-1}(\lambda_0 - \lambda^{(t)}_{-1,i})) - \delta_1(S_t, y^{(t)}_{S_t} - x, \sum_{i=1}^m \nu_{i}^{(t)}(\lambda^{(t)}_{S_t,i} - \lambda^{(t)}_{-1,i})) \\
\leq \frac{\delta_1(t)}{2} \left\| \lambda - \lambda_0 \right\|^2 + \left( \delta_1(t) \rho_1(t) \left\| y^{(t)}_s - x \right\|^2 \right) \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2 + \frac{\delta_1(t) \rho_1(t) \left\| y^{(t)}_s - x \right\|^2}{2 \beta_1(t)} \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2 \\
- \frac{4}{5} \left( \delta_1(t) \beta_1(t) + \alpha/2 \right) \left\| y^{(t)}_s - x \right\|^2 - \delta_1(t) \right\| y_0 - x \right\|^2 \right].
\]

Moreover, since \( L(y^{(t)}_s; \lambda, \pi^t), \left\| y^{(t)}_s - x^{t-1}\right\|^2 \) are convex with respect to \( y^{(t)}_s \) and applying the Jensen’s inequality leads to

\[
\omega_t \left( L(x^t; \lambda, \nu^t, \pi^t) - L(x; \lambda^t, \nu^t, \pi^t) + \frac{\eta}{2} \|x^{t-1} - x\|^2 \right) + \omega_t \left( \frac{\eta + \alpha/2}{2} \|x^{t-1} - x\|^2 \right) \\
+ \omega_t \left( \frac{\eta + \alpha/2}{2} \|x - x^{t-1}\|^2 \right) \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2 + \omega_0 \frac{\eta}{2} \left\| x^{t-1} - x^{t-2} \right\|^2 \right. \\
\leq \frac{\omega_t}{2} \left( \left\| \lambda_0 - \lambda \right\|^2 + \rho_1(t) \left\| y_0 - x^{t-1}\right\|^2 \right) + \frac{\omega_t}{2} \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2.
\]

where \( \tilde{w}_s(t) = \omega_t \delta_s(t) \sum_{s=1}^{S_t} \delta_s(t) \) represents the aggregate weight for the inner iterates.

Next, we consider inner loops among different phases. The inner-phase stepsize condition in (4.14) implies the sum of preceding inequality across \( t \) satisfies

\[
\sum_{t=1}^{T} \omega_t \left[ Q_x(z^t; z) + Q_p(z^t; z) + \sum_{t=1}^{T} \frac{\omega_t}{2} \left\| x^t - x^{t-1}\right\|^2 + \omega_0 \frac{\eta}{2} \left\| x^{t-1} - x^t \right\|^2 \right] \\
\leq \frac{\omega_0}{2} \left( \left\| \lambda_0 - \lambda \right\|^2 + \rho_1(t) \left\| y_0 - x^{t-1}\right\|^2 \right) + \frac{\omega_0}{2} \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2.
\]

Observe that (2.23) is almost identical to the \( Q_x \) and \( Q_\lambda \) inequality in (2.25). Thus a similar argument as Proposition 2.2 and the outer-loop stepsize requirements in (4.12) leads to the desired convergence result in (4.15).

Now we leverage the preceding proposition to prove the convergence of the ACCGD-S method under the non-strongly convex setting.

**Proof to Theorem 4.1 and Corollary 4.2** It is straightforward to verify that the outer loop stepsize in (4.4) satisfies the condition (4.12), and the adaptive inner loop stepsize in (4.5) satisfies both the iner-phase condition (4.13) and the inner-phase condition (4.14). So it follows from (4.15) that

\[
\sum_{t=1}^{T} \omega_t Q(z^t; z) + \omega_0 \omega_t \left\| x^{t-1} - x\right\|^2 \leq \frac{\omega_0}{2} \left( \left\| \lambda_0 - \lambda \right\|^2 + \rho_1(t) \left\| y_0 - x^{t-1}\right\|^2 \right) + \frac{\omega_0}{2} \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2.
\]

Now consider setting the reference point to \( z^* = (x^*, \lambda^*, \nu^* = \nabla g(x^*), \pi^* = \nabla f(x^*)) \) such that \( Q(z^*, z^*) \geq 0 \forall t \). The preceding inequality implies \( \left\| x^N - x^* \right\|^2 \leq \omega_0 \omega_t \left( \left\| \lambda_0 - \lambda \right\|^2 + \omega_0 \omega_t \left\| x^0 - x^* \right\|^2 \right) + \omega_t \left( \left\| \lambda_0 - \lambda^{(t)}_{-1,i} \right\|^2 \right) \forall N \geq 1 \). So \( x^t \), being the convex combination of \( \{x^t\}_s \), remains in a bounded ball around \( x^* \), and \( \nu^t = \nabla g(x^t) \) is bounded for all \( t \geq 1 \).

Next, setting the ergodic average solution as \( \bar{z}^N = \frac{1}{T} \sum_{t=1}^{T} \omega_t \bar{x}^t / (\sum_{t=1}^{T} \omega_t), \bar{y}^N = \frac{1}{T} \sum_{t=1}^{T} \omega_t \bar{y}^t / (\sum_{t=1}^{T} \omega_t) \) with

\[
\bar{\pi}^N := \frac{1}{T} \sum_{t=1}^{T} \omega_t \pi^t / (\sum_{t=1}^{T} \omega_t), \bar{\nu}^N := \frac{\nabla g_0(x^0)}{T} \text{ o.w. if } \lambda_0 = 0 \forall t,
\]

(4.20)
a similar application of the Jensen’s inequality as \((2.27)\) leads to
\[
Q(\tilde{z}^N; (x^*; \lambda, \nu, \pi)) \leq \frac{1}{N(N+1)} \frac{2d(\Lambda_\alpha)\|x^0-x^*\|}{\Delta} + L(\Lambda_\alpha)\|x^0-x^*\|^2, \forall \lambda \in \Lambda_c, (\nu, \pi) \in [V, \Pi]. \tag{4.21}
\]
The convergence in both the optimality gap and the feasibility violation in \((4.6)\) then follows from Lemma \((2.1)\).

Consider now the given choice of \(\Delta\) in Corollary \((4.2)\). It follows from \((4.6)\) that at most \(N_\epsilon = \lceil \sqrt{\frac{4L(\Lambda_\alpha)}{\epsilon} \|x^0-x^*\|} \rceil\) phases are required to find an \((\epsilon; \epsilon/c)\)-optimal solution. Since each phase requires one gradient evaluation for \(f\) and \(g\), the oracle complexity of the ACGD-S method is \(N_\epsilon\). Moreover, since each inner iteration requires less than three matrix-vector multiplication, the total number of matrix-vector multiplication across \(N_\epsilon\) phases can bounded as
\[
C_\epsilon = 3 \sum_{t=1}^{N_\epsilon} S_t \leq N_\epsilon + N_\epsilon^2 \tilde{M} \Delta = O(\sqrt{\frac{L(\Lambda_\alpha)}{\epsilon} \|x^0-x^*\|} + \frac{d(\Lambda_\alpha)\|x^0-x^*\|}{\epsilon}).
\]

The next proof considers the strongly convex case.

**Proof to Theorem 4.3, Corollary 4.4 and Corollary 4.5** It is straightforward to check that the outer-loop stepsize in \((4.7)\) satisfies the condition \((4.12)\), and the adaptive inner-loop stepsize in \((4.8)\) and \((4.9)\) satisfy the intra-phase condition \((4.13)\). Now we verify the inter-phase condition in \((4.14)\). Consider a fixed \(t \geq 2\), we have
\[
\tilde{w}_1^{(t)} = \frac{\omega_t \delta^{(t)}_1}{\sum_{s=1}^{S_t} \delta^{(t)}_s} = \frac{\omega_t \delta^{(t)}_1}{\omega_t M_t^2 \Gamma_1 \Delta} = \frac{M_t^2}{\omega_t M_t^2 \Gamma_1 \Delta},
\]
\[
\tilde{w}_{S_t-1}^{(t-1)} = \frac{\omega_{t-1} \delta^{(t-1)}_{S_t-1}}{\sum_{s=1}^{S_t-1} \delta^{(t-1)}_s} = \frac{\omega_{t-1} \delta^{(t-1)}_{S_t-1}}{\omega_{t-1} M_{t-1}^2 \Gamma_{t-1} \Delta} = \frac{M_{t-1}^2 \Gamma_{t-1}}{\omega_{t-1} M_{t-1}^2 \Gamma_{t-1} \Delta}.
\]
Thus
\[
\tilde{w}_{S_t-1}^{(t-1)} (\beta_{S_t-1}^{(t-1)} + \alpha/2) = \frac{W_t}{M_{t-1}^2 \Gamma_{t-1}} \tilde{w}_{S_t-1}^{(t-1)} (\beta_{S_t-1}^{(t-1)} + \alpha/2) = \frac{\omega_t M_t \Gamma_t W_t}{M_{t-1}^2 \Gamma_{t-1} \Delta} = \frac{\omega_t}{\Delta} \frac{M_t \Gamma_t W_t}{M_{t-1}^2 \Gamma_{t-1} \Delta} = \frac{\omega_t}{\Delta} \frac{M_t \Gamma_t W_t}{M_{t-1}^2 \Gamma_{t-1} \Delta} = \tilde{w}_1^{(t)} \beta_1^{(t)}.
\]
\[
\tilde{w}_{S_t-1}^{(t-1)} \gamma_{S_t-1}^{(t-1)} = \frac{\delta^{(t-1)}_{S_t-1}}{M_{t-1}^2 \Gamma_{t-1}} (\beta_{S_t-1}^{(t-1)} + \alpha/2) = \frac{\delta^{(t-1)}_{S_t-1}}{M_{t-1}^2 \Gamma_{t-1}} \frac{M_{t-1}^2 \Gamma_{t-1}}{\delta^{(t-1)}_{S_t-1}} = \tilde{w}_1^{(t)} \gamma_1^{(t)},
\]
\[
\beta_1^{(t)} \gamma_{S_t-1}^{(t-1)} = \frac{M_t \Gamma_t W_t}{M_{t-1}^2 \Gamma_{t-1} \Delta} \beta_1^{(t)} \gamma_{S_t-1}^{(t-1)} = \frac{M_t \Gamma_t W_t}{M_{t-1}^2 \Gamma_{t-1} \Delta} \beta_1^{(t)} \gamma_{S_t-1}^{(t-1)} = \beta_1^{(t)} \beta_1^{(t)},
\]
where the inequality in \((a)\) holds because we have \(\Gamma_{t-1} \geq 1\) as a consequence of its definition. Thus all the requirements in Proposition \((4.7)\) are satisfied. We get from \((4.15)\) that
\[
\sum_{t=1}^{N} \omega_t Q(z^t; z) + \frac{\epsilon}{\sqrt{2\epsilon}} \|x^N - x^*\|^2 \leq \frac{\tilde{w}_1^{(t)} \delta^{(t)}_{1} + \epsilon \omega_t \|x^0 - x^*\|^2}{\Delta} + \frac{\tilde{w}_1^{(t)} \delta^{(t)}_{1}}{\Delta} \|x^0 - x^*\|^2.
\tag{4.22}
\]
Similar arguments as that of Theorem \((4.1)\) imply the boundedness of \(M_t\), and that the ergodic average solution \(\tilde{z}^N\) defined according to \((4.13)\) satisfies
\[
Q(\tilde{z}^N; (x^*; \lambda, \nu, \pi)) \leq \frac{1}{N} \sum_{t=1}^{N} \omega_t \left(\frac{2d(\Lambda_\alpha)\|x^0-x^*\|}{\Delta} + L(\Lambda_\alpha)\|x^0-x^*\|^2\right), \forall \lambda \in \Lambda_c, (\nu, \pi) \in [V, \Pi]. \tag{4.23}
\]
Since \(\sum_{t=1}^{N} \omega_t \geq \max\{N(N+1)/2, \sqrt{\frac{2\nu}{\|x^N\|} (1 + \sqrt{\frac{2\nu}{\|x^N\|}})^4 - 1}\} \) (see \((2.20)\)), we get the optimality gap and feasibility violation convergence bound in \((4.10)\). Moreover, since \(\tilde{z}^N\) converges to \(x^*\) in \((4.10)\) also follows from \((4.23)\).

Next we show Corollary 4.4. Fix a \(c \geq 1\), and set \(\Delta = \frac{\|x^0-x^*\|^2}{L(\Lambda_\alpha)\|x^0-x^*\|^2} \). For any
\[
N \geq \min\{\sqrt{\frac{6L(\Lambda_\alpha)\max\{c/r, 1\}}{\epsilon} \|x^0-x^*\|^2}, \sqrt{\frac{2L(\Lambda_\alpha)}{\epsilon} + 1} \log(\frac{3\max\{c/r, 1\} L(\Lambda_\alpha)\|x^0-x^*\|^2}{\epsilon} + 1) + 4\},
\]
we get
\[ Q(\overline{x}^N; (x^*; \lambda, \nu, \pi)) \leq \min \{1, \overline{\epsilon} \} \epsilon, \forall \lambda \in \Lambda_r, (\nu, \pi) \in [V, \Pi], \Rightarrow \max \{ f(x^N) - f^*, r \|g(x^N)\|_\infty \} \leq \min \{1, \overline{\epsilon} \} \epsilon, \]
so that \( \overline{x}^N \) is an \( (\epsilon; \epsilon/c) \) solution. Therefore, the least number of phases \( N_e \), required for such a solution admits the upper bounded in the corollary statement.

Now we consider the corresponding number of matrix-vector multiplications \( C_e \) in the \( N_e \) phases, i.e.,
\[ C_e = \sum_{i=1}^{N_e} S_t. \]
We can deduce from the preceding argument that
\[ \sum_{i=1}^{N_e} \omega_i \leq \frac{6 \max \{1, \overline{\epsilon} \} L(\Lambda_r) \| x^0 - x^* \|^2}{\epsilon}. \]
Calculating the sum of \( S_t \) directly is challenging, so we consider an easier quantity \( R_t := [S_t - 3]_+ \). An useful algebraic relation is
\[ \sum_{i=1}^{t-1} R_t \leq \bar{M} W_t. \] (4.24)
For \( t = 2 \), the relation clearly holds since \( W_2 \bar{M} \geq W_2 M_1 = S_1 \geq R_1 \). For \( t \geq 2 \), we have
\[ W_{t+1} = \frac{W_t M_t + \frac{1}{M_t} (S_t - 1)}{M_t} \geq \frac{W_t + \frac{S_t - 1}{M_t}}{M_t} \geq \frac{1}{M_t} \left( \sum_{i=1}^{t-1} R_i + [S_t - 3]_+ \right) = \frac{1}{M_t} \sum_{i=1}^{t-1} [R_i], \]
where (a) follows from the algebraic fact \( \frac{S_t}{M_t} \geq S_t - 2 \) (see Lemma 6.3), and that \( S_t - 1 \geq 0 \) and (b) follows from the induction hypothesis. Thus the principle of induction implies that (4.24) holds. Consequently, for \( R_e = \sum_{i=1}^{N_e} R_t \), we have
\[ R_e^2 / 2 \leq \sum_{s=1}^{R_e} s = \sum_{t=1}^{N_e} \sum_{s=1}^{R_t} [(\sum_{j=1}^{t-1} R_j) + s] \leq \sum_{t=1}^{N_e} \sum_{s=1}^{R_t} [\bar{M} W_t + s] = \sum_{t=1}^{N_e} \sum_{s=1}^{R_t} [\bar{M} W_t + (s - 1)] \leq \bar{M} \sum_{t=1}^{N_e} \sum_{s=1}^{R_t} [W_t + \frac{(s-1)}{M_t}] \]
\[ \leq \bar{M} \sum_{t=1}^{N_e} M_t \omega_t \Delta \leq N_e \bar{M}^2 \omega_t \Delta \leq \frac{6 \max \{1, \overline{\epsilon} \} L(\Lambda_r) \| x^0 - x^* \|^2}{\epsilon} \frac{|d(\Lambda_r)|^2}{L(\Lambda_r) \| x^0 - x^* \|^2} \bar{M}^2 \]
\[ \leq \frac{6 \max \{1, \overline{\epsilon} \} \bar{M}^2 |d(\Lambda_r)|^2}{\epsilon}, \]
where (a) follows from the fact that \( \sum_{s=1}^{R_t} W_t M_t + (s - 1) \leq \omega_t M_t^2 \Delta \). Therefore we get \( R_e \leq \sqrt{\frac{12}{\epsilon}} \bar{M} d(\Lambda_r). \)
Since \( \sum_{t=1}^{N_e} S_t \leq R_t + 3 N_e \), the big-O bound on \( C_e \) follows immediately.

The complexity bounds in Corollary 4.15 can be derived similarly.

5. Conclusion. To sum up, this paper proposes two efficient method for the large-scale function constrained optimization. The ACGD method has the optimal oracle complexity, but requires the access to an QP solver. The ACGD-S method has both the optimal oracle complexity and the optimal computation complexity. Lower complexity bounds are provided to illustrate the oracle complexity of both ACGD and ACGD-S to be unimprovable for a general case of first-order methods, thus establishing the tight oracle complexity for function constrained optimization under certain accuracy regimes.

REFERENCES

[1] A. Beck, First-order methods in optimization, vol. 25, SIAM, 2017.
[2] D. Bertsekas, Convex optimization theory, vol. 1, Athena Scientific, 2009.
[3] D. Boob, Q. Deng, and G. Lan, Stochastic first-order methods for convex and nonconvex functional constrained optimization, Mathematical Programming, (2022), pp. 1–65.
[4] R. Gandy, Portfolio optimization with risk constraints, PhD thesis, Universität Ulm, 2005.
[5] E. Y. Hamedani and N. S. Aybat, A primal-dual algorithm with line search for general convex-concave saddle point problems, SIAM Journal on Optimization, 31 (2021), pp. 1299–1329.
[6] ———, A primal-dual algorithm with line search for general convex-concave saddle point problems, SIAM Journal on Optimization, 31 (2021), pp. 1299–1329.

24
Lemma 6.1. Let \( \Lambda \subset \mathbb{R}^n \) and a convex vector-valued function \( g : \mathbb{R}^n \to \mathbb{R}^m \) be given. If \( \sum_{i=1}^m \lambda_i g_i \) is \( \mathcal{L} \)-smooth for all \( \lambda \in \Lambda \), i.e., \( \| \sum_{i=1}^m \lambda_i \nabla g_i(x) - g_i(\bar{x}) \| \leq \mathcal{L} \| x - \bar{x} \| \) \( \forall x, \bar{x} \in \mathbb{R}^n \), \( \forall \lambda \in \Lambda \), the Bregman distance function generated by its (vector-valued) conjugate function \( U_g^* \) satisfies

\[
\langle \lambda, U_g^*(\nu; \bar{v}) \rangle \geq \left\| \sum_{i=1}^m \lambda_i (\nu_i - \bar{v}_i) \right\|^2 / (2\mathcal{L}) \quad \forall \nu, \bar{v} \in \{\nabla g(x) : x \in \mathbb{R}^n\}.
\]

Proof. Let \( \lambda \in \Lambda \), \( \nu = \nabla g(x) \), and \( \bar{v} = \nabla g(\bar{x}) \) be given. Consider the function \( \hat{g} : \sum_{i=1}^m \lambda_i \hat{g}_i \). Clearly, \( \hat{g} \) is \( \mathcal{L} \)-smooth such that the Bregman distance function generated by its conjugate satisfies \( U_{\hat{g}}^*(\nu; \bar{v}) \geq \| \nu - \bar{v} \|^2 / (2\mathcal{L}) \). Since \( \nabla \hat{g}(x) = \lambda^T \nabla g(x) \), we have

\[
\mathcal{L}^T U_{\hat{g}}^*(\nu; \bar{v}) \overset{(a)}{=} \sum_{i=1}^m \lambda_i U_{\hat{g}_i}^*(\nu_i; \bar{v}_i) = \sum_{i=1}^m \lambda_i U_{g_i}(\bar{x}; x) = \sum_{i=1}^m \lambda_i [g_i(\bar{x}) - g_i(x) - \langle \nabla g_i(x), \bar{x} - x \rangle] = \hat{g}(\bar{x}) - \hat{g}(x) - \langle \nabla \hat{g}(x), \bar{x} - x \rangle \overset{(b)}{=} U_{\hat{g}}(\bar{x}; x) = U_{\hat{g}}^*(\lambda^T \nu; \lambda^T \bar{v}) \geq \left\| \sum_{i=1}^m \lambda_i (\nu_i - \bar{v}_i) \right\|^2 / (2\mathcal{L}),
\]

where (a) and (b) follows from the algebraic identity between Bregman distance functions generated by Fenchel conjugates \( (h, h^*) \), \( U_h(y; \bar{y}) = U_{h^*}(\nabla h(\bar{y}); \nabla h(y)) \).

The next two lemmas provide some basic algebraic identifies useful for deriving complexity bounds.

Lemma 6.2. Given an \( x > 0 \), the following algebraic relation is valid:

\[
h(y) := (1 + 1/x)^{y-3} \leq y \quad \forall \quad 2x \geq y \geq 1.
\] (6.1)
Proof. First, we show the relation for \( y = 2x \), i.e. \( h(2x) = (1 + 1/x)^{2x-3} \leq 2x \forall x > 0 \). Let’s consider two cases. If \( x \geq 4 \), we have

\[
(1 + 1/x)^{2x-3} \leq [(1 + 1/x)^x]^2 \leq \exp(2) \leq 2x.
\]

If \( 0 < x < 4 \), we have

\[
[(1 + 1/x)^{x-1.5}]^2 \leq [1/(1 - \frac{x-1.5}{x})]^2 = (x/1.5)^2 = (x/2.25) \times x \leq 2x.
\]

Thus \( h(2x) \leq 2x \). Since \( h(1) \leq 1 \), the relation in (6.1) follows from the convexity of \( h \) with respect to \( y \).

Lemma 6.3. Given non-negative parameters \( \Delta_t \), \( H \), and \( h \in \{0,1\} \), suppose \( S_t = \min \{ S \in \mathbb{N}_+ : \sum_{s=1}^S H + (s - h) \geq \Delta_t \} \), and \( \Gamma \) is the non-negative root of \( \sum_{s=1}^S [\Gamma H + (s - h)] = \Gamma^2 \Delta_t \), then \( S_t \) satisfies

\[
S_t/\Gamma \geq S_t - 2.
\]

Proof. Suppose for the sake of contradiction that \( S_t/\Gamma < S_t - 2 \). On the one hand, the definition of \( S_t \) implies that

\[
\Delta_t = H(S_t/\Gamma) + \sum_{s=1}^{S_t-1} \frac{(s-h)}{S_t} = H(S_t/\Gamma) + \frac{1}{2} \frac{(S_t-h+1)}{S_t} S_t \geq H(S_t/\Gamma) + \frac{1}{2} \frac{(S_t)}{S_t}^2.
\]

On the other hand, the choice of \( S_t \) implies that \( \sum_{s=1}^{S_t-1} [H + (s - h)] < \Delta_t \), thus

\[
H(S_t - 1) + \frac{1}{2}(S_t - 1)(S_t - h - 1) < \Delta_t
\]

\[
\Rightarrow H(S_t/\Gamma) + \frac{1}{2} \frac{(S_t)}{S_t}^2 < \Delta_t.
\]

These two relations leads to the desired contradiction.