Journal of Machine Learning Research 26 (2025) 1-45 Submitted 2/24; Revised 4/25; Published 9/25

An Adaptive Parameter-free and Projection-free Restarting Level Set Method for Constrained Convex Optimization Under the Error Bound Condition

Qihang Lin QIHANG-LIN@UIOWA.EDU
The University of Iowa, Tippie College of Business, Iowa City, IA 52242

Negar Soheili[*] NAZAD@UIC.EDU
University of Illinois at Chicago, College of Business Administration, Chicago, IL 60607

Runchao Ma RUNCHAO-MA@UIOWA.EDU
The University of Iowa, Tippie College of Business, Iowa City, IA 52242

Selvaprabu Nadarajah SELVAN@UIC.EDU
University of Illinois at Chicago, College of Business Administration, Chicago, IL 60607

Editor: Martin Jaggi

Abstract

Recent efforts to accelerate first-order methods have focused on convex optimization problems that satisfy a geometric property known as error-bound condition, which covers a broad class of problems, including piece-wise linear programs and strongly convex programs. Parameter-free first-order methods that employ projection-free updates have the potential to broaden the benefit of acceleration. Such a method has been developed for unconstrained convex optimization but is lacking for general constrained convex optimization. We propose a parameter-free level-set method for the latter constrained case based on projection-free subgradient method that exhibits accelerated convergence for problems that satisfy an error-bound condition. Our method maintains a separate copy of the level-set sub-problem for each level parameter value and restarts the computation of these copies based on objective function progress. Applying such a restarting scheme in a level-set context is novel and results in an algorithm that dynamically adapts the precision of each copy. This property is key to extending prior restarting methods based on static precision that have been proposed for unconstrained convex optimization to handle constraints. We report promising numerical performance relative to benchmark methods.

Keywords: level set method, parameter free, projection free, accelerated methods, constrained convex optimization, error bound condition

1. Introduction

In this paper, we consider a convex optimization problem with inequality constraints:

$$f^* := \min_{\mathbf{x}\in\mathcal{X}} \{ f(\mathbf{x}) := f_0(\mathbf{x}) \quad \text{s.t.} \quad g(\mathbf{x}) := \max_{i=1,\ldots,m} f_i(\mathbf{x}) \le 0 \}, \tag{1}$$

where f_i for $i = 0, 1, \ldots, m$ are convex real-valued functions and $\mathcal{X} \subset \mathbb{R}^n$ is a closed convex set with an efficiently computable projection mapping. Given $\epsilon > 0$, we call $\bar{\mathbf{x}}$ an *ϵ-optimal* solution to (1) if $f(\bar{\mathbf{x}}) - f^* \le \epsilon$ and *ϵ-feasible* if $\bar{\mathbf{x}} \in \mathcal{X}$ and $g(\bar{\mathbf{x}}) \le \epsilon$. There has been a large volume of literature on constrained convex optimization problems in deterministic settings (Yu et al., 2017; Bayandina et al., 2018; Grimmer, 2018; Lin et al., 2018a,b; Wei et al., 2018; Aravkin et al., 2019; Fercoq et al., 2019; Xu, 2020, 2021a,b; Ene et al., 2021; Ito et al., 2023; Sujanani and Monteiro, 2025; Deng et al., 2024) and significant, but comparatively less, activity in stochastic (Lan and Zhou, 2016; Yu et al., 2017; Lin et al., 2020) settings.

Recent developments have focused on accelerating first order methods with linear convergence rates (without assuming strong convexity) based on a more general geometric property known as the "error bound condition (EBC)", which is

$$\frac{\operatorname{dist}(\mathbf{x}, \mathcal{X}^*)^d}{G} \leq \max\{f(\mathbf{x}) - f^*, g(\mathbf{x})\}, \quad \forall \mathbf{x} \in \mathcal{X}, \tag{2}$$

for some parameters $G > 0$ and $d \geq 1$, where $\mathcal{X}^*$ is the set of optimal solutions to (1) and $\operatorname{dist}(\mathbf{x}, \mathcal{X}^*)$ denotes the Euclidean distance of $\mathbf{x}$ to $\mathcal{X}^*$, which we refer to as *solution distance*. EBC provides a lower bound on the maximum of the optimality $(f(x) - f^*)$ and feasibility $(g(x))$ gaps as a function of the solution distance, d, and G. A larger such lower bound is desirable, as it implies that additional progress in reducing solution distance yields meaningful improvements in optimality and feasibility. In contrast, if this lower bound is very small, iterations that move $\mathbf{x}$ closer to $\mathcal{X}^*$ may not result in significant optimality or feasibility progress. Smaller values of G lead to larger lower bounds all else being the same and are thus preferred. For a fixed G, we prefer a smaller d because it corresponds to more favorable geometry closer to the optimal solution set, which is typically when the progress of first-order methods slows down. Specifically, when $\mathbf{x}$ is close to the set of optimal solutions (i.e., is $\operatorname{dist}(\mathbf{x}, \mathcal{X}^*) < 1$), smaller d leads to a larger lower bound. Many constrained convex optimization problems satisfy EBC with exponents $d = 1$ or $d = 2$. For example, piecewise linear programs typically yield $d = 1$, while strongly convex problems yield $d = 2$. Although less commonly discussed in the literature, the EBC framework can also accommodate exponents larger than two. A concrete example and bound are provided in Appendix A.

We refer to an algorithm as *adaptive* if its convergence rate improves (i.e., it accelerates) as d and G become smaller[1]. Adaptive methods for unconstrained or simply constrained[2] convex optimization problems have been explored. These methods use the knowledge of parameters d, G, and f^* in their step length (Polyak, 1969; Iouditski and Nesterov, 2014; Bolte et al., 2017; Davis et al., 2018; Wei et al., 2018; Xu and Yang, 2018; Zhang et al., 2021; Grimmer, 2019; Necoara et al., 2019; Johnstone and Moulin, 2020; Zhang, 2020; Renegar and Zhou, 2021; Grimmer, 2023; Wang and Liu, 2022; Grimmer, 2024) or in the frequency with which they restart (Xu et al., 2016; Liu and Yang, 2017; Xu et al., 2017a,b,c; Yang and Lin, 2018; Davis et al., 2024; Fercoq and Qu, 2019; Yan et al., 2019; Fercoq and Qu, 2020; Charisopoulos and Davis, 2024; Fercoq, 2023). Such parameters are unknown in general, and their accurate estimation is challenging. Adaptive algorithms that are also parameter-free (i.e., do not rely on the knowledge of unknown parameters) would therefore broaden or ease the applicability of accelerated methods.

While a range of parameter-free adaptive methods exist for unconstrained problems, analogous results for constrained settings remain limited. To the best of our knowledge,

1. We note that algorithms that exhibit acceleration for only specific values of d (e.g., under strong convexity, which corresponds to $d = 2$) would not be considered adaptive under our definition.
2. Here, simply constrained means $m = 0$ in (1) and $\mathcal{X}$ is a simple set, e.g., $\mathbb{R}^n$, a box or a ball.

Roulet and d'Aspremont (2017), Díaz and Grimmer (2023), Ito and Fukuda (2021), and Renegar and Grimmer (2022) are the only parameter-free adaptive methods that handle unconstrained problems. The restarted method in Roulet and d'Aspremont (2017) utilizes a logarithmic grid search over d and G to determine the optimal frequency of restart without knowing d and G parameters in advance. While effective, their method is only applicable to unconstrained problems and does not extend to constrained settings. The method in Renegar and Grimmer (2022) employs a subgradient scheme to solve K copies of the problem with termination precisions $2^k\epsilon$ for $k = -1, 0, \dots, K-1$. Copy k communicates its current solution to copy $k-1$ when a restart is triggered. Since each copy operates at a pre-defined precision, this restarting subgradient method (RSG) can be interpreted as a static precision-based restarting scheme. A key feature of RSG is that it tracks the reduction in the objective function (i.e., progress made) to determine when to trigger a restart, as opposed to relying on the distance from f^* (i.e., remaining progress). This structure enables the method to operate without requiring prior knowledge of problem parameters. However, RSG is not designed to handle convex optimization problems with (potentially) complicated constraints. This method has been adapted to bundle methods in Díaz and Grimmer (2023), where similar restart logic is used in a subgradient bundle framework. Ito and Fukuda (2021) apply a similar idea for smooth unconstrained convex optimization problems where the restarts are triggered based on the norm of proximal gradients instead of function values.

Several works, including Nesterov (2013), Lin and Xiao (2015), Lan et al. (2023), and Sujanani and Monteiro (2025), study parameter-free methods for smooth, strongly convex, unconstrained problems that satisfy the EBC with exponent $d = 2$. These methods employ accelerated gradient schemes combined with backtracking line search to estimate the gradient Lipschitz constant, using a guess-and-check procedure to achieve optimal complexity without prior knowledge of G. In contrast, Davis and Jiang (2024) propose a subgradient method for unconstrained, non-smooth convex problems under the same EBC assumption ($d = 2$), which achieves *local* nearly linear convergence. Our work extends these results by providing *global* convergence guarantees for general constrained problems.

There are limited adaptive algorithms that solve the constrained convex optimization problem (1); see Renegar (2016), Yang et al. (2017), and Adcock et al. (2025). The algorithm proposed in Renegar (2016) addresses the case of $d = 1$, but it requires prior knowledge of the optimal value f^*. The method in Yang et al. (2017) applies to general d, but assumes knowledge of both G and d, among other problem parameters. Moreover, it requires projection onto the set $\mathcal{X} \cap \{\mathbf{x} \mid g(\mathbf{x}) \leq 0\}$, which is often computationally intractable. Adcock et al. (2025) propose a restarted method that uses a grid search approach - similar to that of Roulet and d'Aspremont (2017) - to estimate both d and G. Unlike Roulet and d'Aspremont (2017), however, they introduce a scheduling criterion function to determine the order in which the grid estimates are applied. Their method can be extended to constrained problems by incorporating an exact penalty term with a sufficiently large penalty parameter. However, it remains unclear how such a parameter can be selected while maintaining the method's parameter-free nature. Furthermore, even with this extension, the complexity of the method in Adcock et al. (2025) is higher than ours. For instance, in the non-smooth case with $d = 1$, our method achieves a complexity of $\mathcal{O}\left(\log^2(1/\epsilon)\right)$, while theirs is $\mathcal{O}\left(c^{-1}\log^{2+c}(1/\epsilon)\right)$ for any $c > 0$.

The goal of our paper is to develop a first method that is simultaneously adaptive, parameter-free, and projection-free for general convex constraints. [3]. Extending RSG to handle general convex constraints would be the natural first avenue to consider to achieve our goal. Indeed, the k-th copy in such an extension will solve the problem of finding an $2^k\epsilon$-optimal and $2^k\epsilon$-feasible solution, which entails balancing optimality and feasibility by defining a single objective. Defining such an objective is possible given optimal Lagrange multipliers or f^*, which are both unknown. Thus, extending RSG in this manner appears to be challenging.

EBC	Smooth Functions	Algorithm Parameters	Complexity (Function/Gradient Evaluations)
General [This paper]	No	None	$\mathcal{O}\left((mM^2G^{(2/d)})/\epsilon^{(2-2/d)}\log^2(1/\epsilon)\right)$
	Yes	None	$\mathcal{O}\left(m\sqrt{L}\max\left\{G^{1/d}/\epsilon^{(1/2-1/d)},1\right\}\log^2(1/\epsilon)\right)$
General (Yang et al., 2017), where m = 1[††]	No	d, G, M	$\mathcal{O}\left((M^4G^{(2/d)})/\epsilon^{(2-2/d)}\log(1/\epsilon)\right)$
	Yes	d, G, M, L	$\mathcal{O}\left((M^2G^{(1/d)})/\epsilon^{(1-1/d)}\log(1/\epsilon) + (\sqrt{ML}G^{(1/d)})/\epsilon^{(1-1/d)}\right)$
$d=2$ (Lin et al., 2018b)	No	M, μ	$\mathcal{O}\left(mM^2/\mu\epsilon\right)$
	Yes	M, L, μ	$\mathcal{O}\left(m\left(\sqrt{\log(m)}M/\sqrt{\mu\epsilon}+\sqrt{L/\mu}\log(1/\epsilon)\right)\right)$
$d=2$ (Xu, 2021b)	Yes	M, L, μ	$\mathcal{O}\left(m\left((M+\sqrt{ML})/\epsilon+\sqrt{L/\epsilon}\right)\log(1/\epsilon)\right)$
$d=1$ (Renegar, 2016)	No	f^*	$\mathcal{O}\left(mG^2\log(1/\epsilon)\right)$

Table 1: Complexity of subgradient-based methods that accelerate under the error bound condition (2) for smooth and non-smooth constrained convex optimization problems. The bounds reflect the total number of subgradient, gradient, or function evaluations (including those from constraint functions) required to obtain an ϵ-optimal and ϵ-feasible solution. Here, M denotes an upper bound on the subgradient norm, and L and μ are the smoothness (Lipschitz) and strong convexity parameters associated with the functions f_i, $i=0,1,\dots,m$. [††] indicates that the method requires projection onto $\mathcal{X}\cap\{\mathbf{x}\mid g(\mathbf{x})\le 0\}$.

3. As is common in the literature when using the term projection-free, we do allow projections onto $\mathcal{X}$, which are inexpensive to compute and often available in closed form.

We show that our goal can be achieved by developing a restarting level set method. Level-set methods in Aravkin et al. (2019) and Lin et al. (2018a,b, 2020) convert the solution of the constrained optimization problem (1) into the solution of a sequence of unconstrained non-smooth optimization problems that depend on a scalar, referred to as the level parameter. The level parameter is chosen by approximately solving a one-dimensional root-finding problem[4]. Aravkin et al. (2019) provides a detailed discussion and complexity analysis of the level set approach. Subgradient algorithms to solve the aforementioned level-set subproblems are desirable as they are easy to implement. However, level set methods that leverage subgradient algorithms depend on unknown parameters, which may include an upper bound on the subgradient norm and/or a smoothness parameter (Lin et al., 2018a,b, 2020). Our contributions are the following:

- We introduce a restarting level set method (RLS) that is both parameter-free and projection-free. It maintains K copies, each distinguished by a level parameter and its associated sub-problem. Each restart of copy k results in an update of the level parameters associated with copies $k, k+1, \ldots, K$. The restart of a copy is triggered by the progress made on the sub-problem objective. To the best of our knowledge, this is the first level set method based on projection-free subgradient oracles that is parameter-free, which may also be of broader interest within the literature on level set methods.

- RLS can be interpreted as a restarting scheme with dynamic precision, which is conceptually different from RSG. To elaborate, a fixed level parameter implicitly sets a precision for a subproblem. Since RLS updates the level parameter of multiple copies in every restart, it dynamically and implicitly updates the precision associated with copies. Our key algorithmic insights are that (i) triggering restarts based on the progress of the subproblem objective and (ii) dynamically changing the precision of subproblems at each restart by updating the level parameters side step the issues discussed earlier of extending RSG to handle general constrained convex optimization.

- We establish that RLS is adaptive in addition to being parameter-free, that is, RLS exhibits acceleration under EBC. Its iteration complexity for non-smooth problems is $\mathcal{O}((M^2G^{2/d})/\epsilon^{(2-2/d)}\log^2(1/\epsilon))$. The adaptive method in Yang et al. (2017) has a complexity of $\mathcal{O}((M^4G^{2/d})/\epsilon^{(2-2/d)}\log(1/\epsilon))$ but it depends on unknown parameters and requires a non-trivial projection onto $\mathcal{X} \cap \{\mathbf{x}|g(x) \leq 0\}$. The additional $\log\left(\frac{1}{\epsilon}\right)$ factor in the RLS complexity can be interpreted as the cost of both relaxing parameter dependence and employing simple subgradient oracles. Despite this worsening, the dependence of RLS on M is better. Table 1 shows that an analogous property holds in the smooth setting when comparing RLS with the approach in Yang et al. (2017). Although perhaps unfair to RLS, Table 1 also compares its computational complexity to algorithms designed for specific values of d, all of which depend on unknown parameters. RLS worsens by the same logarithmic factor of $\log\left(\frac{1}{\epsilon}\right)$ relative to these

4. An alternative approach to handle constraints is using the radial duality framework of Grimmer (2018); Renegar (2016), where a constrained convex optimization problem is converted to an unconstrained convex optimization problem, but solving this converted problem relies on being able to do exact line searches to evaluate the gauge of the feasible set.

more specialized algorithms and sometimes improves on the dependence with respect to G.

- We evaluate the performance of RLS on two types of constrained classification problems: (i) fairness-constrained classification, where we consider both non-smooth and smooth formulations, and (ii) non-smooth Neyman–Pearson classification with constraints on the Type II error control. In the fairness-constrained setting, we compare RLS with three parameter-free but non-adaptive benchmarks in the non-smooth case - the feasible level set method from Lin et al. (2018b), the subgradient method from Yu et al. (2017), and the switching subgradient method from Bayandina et al. (2018) - and with the feasible level set method and the constraint extrapolation (ConEx) method from Boob et al. (2023) in the smooth case. For the Neyman–Pearson classification problem, we compare RLS against the same three non-smooth baselines. These experiments include large-scale real-world datasets with up to 72,000 data points and 47,000 features. Through the experiments, we find that the adaptive nature of RLS enables faster convergence to an ϵ-optimal and ϵ-feasible solution than existing benchmarks in various regimes.

This paper is organized as follows: In Section 2, we discuss a general level-set method for solving (1) and discuss its convergence rate. In Section 3, we present RLS and analyze its oracle complexity. In Section 4, we provide first order oracles for use with RLS in the non-smooth and smooth settings, also analyzing the total complexity of RLS in each case. In Section 5, we numerically compare RLS to other benchmarks. We conclude in Section 6.

2. Level Set Method

Throughout the paper, we assume that there exists a computable feasible solution to (1) that strictly satisfies the constraint $g(\mathbf{x}) \leq 0$. This assumption is formalized below.

Assumption 1 *There exists a computable solution $\tilde{\mathbf{x}}$ such that $\tilde{\mathbf{x}} \in \mathcal{X}$ and $g(\tilde{\mathbf{x}}) < 0$.*

We define the *level-set function* corresponding to (1) as:

$$H(r) := \min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r), \tag{3}$$

where $r \in \mathbb{R}$ is called a *level parameter* and

$$P(\mathbf{x}; r) := \max\{f_0(\mathbf{x}) - r, f_1(\mathbf{x}), \dots, f_m(\mathbf{x})\} = \max\{f(\mathbf{x}) - r, g(\mathbf{x})\}. \tag{4}$$

Below, we summarize some of the well-known properties of $H(r)$ that we use in our algorithm and analyses.

Lemma 1 (Lin et al. (2018b); Nesterov (2018)) *The function $H(r)$ defined in (3) has the following properties:*

1. *$H(r)$ is non-increasing and convex in r;*
2. *$H(f^*) = 0$;*
3. *$H(r) > 0$ when $r < f^*$ and $H(r) < 0$ when $r > f^*$.*

Algorithm 1 Level Set Method

```
1: Input: $\alpha \in (0,1)$, $\epsilon > 0$, and $r_0 < f^*$.
2: Initialize: $k = -1$.
3: do
4:     $k = k + 1$.
5:     Solve $\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r_k)$ to obtain $\mathbf{x}_k \in \mathcal{X}$ such that
           $\alpha P(\mathbf{x}_k; r_k) < f^* - r_k.$      (5)
6:     Set $r_{k+1} = r_k + \alpha P(\mathbf{x}_k; r_k)$.
7: while $P(\mathbf{x}_k; r_k) > \epsilon$.
8: Output: $\mathbf{x}_k$.
```

4. $H(r) - \delta \le H(r+\delta) \le H(r)$ for any $\delta \ge 0$.

Lemma 1 indicates that f^* is the unique root to the root-finding problem $H(r) = 0$ under Assumption 1. Notice that Assumption 1 guarantees the existence of strong duality for problem (1). Without the assurance of strong duality, the root of $H(r)$ might not lead to f^* (refer to Estrin and Friedlander (2020) for further insights). Moreover, as shown in Lemma 2, it is possible to find an ϵ-feasible solution and use the level parameter r to bound the optimality gap.

Lemma 2 *Given $\epsilon > 0$, if $\mathbf{x} \in \mathcal{X}$ satisfies $P(\mathbf{x}; r) \le \epsilon$ for some r, then $\mathbf{x}$ is $(r - f^* + \epsilon)$-optimal and ϵ-feasible. Consequently, when $r < f^*$, this solution $\mathbf{x}$ is ϵ-optimal and ϵ-feasible.*

Hence to tackle (1), a level-set method applies a root-finding scheme to solve $H(r) = 0$ with the goal of finding $r = f^*$. Once such r is found, then $P(\mathbf{x}; r) \le \epsilon$ implies that $\mathbf{x}$ is a nearly optimal and nearly feasible solution to (1). Notice that for any $\mathbf{x}^* \in \mathcal{X}^* := \arg\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; f^*)$ we have $P(\mathbf{x}^*; f^*) = 0$.

Algorithm 1 presents a level set method that generates a sequence of level parameters $\{r_k\}_{k\ge 0}$ converging to f^*. Given $\alpha \in (0,1)$, each update of $r_{k+1} = r_k + \alpha P(\mathbf{x}_k; r_k)$ requires a vector $\mathbf{x}_k \in \mathcal{X}$ satisfying $\alpha P(\mathbf{x}_k; r_k) < f^* - r_k$, that is, condition (5). The convergence of Algorithm 1 relies on $r_k < f^*$. Under this condition, it follows from Lemma 1(4) and $\alpha < 1$ that $\mathbf{x}_k$ can be computed by solving $\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r_k)$ because $\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r_k) = H(r_k) \le f^* - r_k < (f^* - r_k)/\alpha$. Moreover, the level parameter r_k converges to f^* from below causing the quantity $P(\mathbf{x}_k; r_k) \ge 0$ to converge to zero. Thus, Algorithm 1 terminates and the final solution is both ϵ-optimal and ϵ-feasible by Lemma 2.

Next, we present analysis that shows the correctness and complexity of Algorithm 1. This analysis depends on a condition measure that is defined as

$$\theta := -\lim_{r\to f^{*-}} \frac{H(r) - H(f^*)}{r - f^*} = \lim_{r\to f^{*-}} \frac{H(r)}{f^* - r}. \tag{6}$$

The following lemma sheds light on some geometric properties of the condition measure θ. We skip proving this lemma as it directly follows from the definition of θ in (6) and the first and fourth properties in Lemma 1.

Lemma 3 *It holds that* $0 < \theta \le \dfrac{H(r)}{f^* - r} \le 1$ *for any* $r < f^*$ *and* $0 \le \dfrac{H(r)}{f^* - r} \le \theta$ *for any* $r > f^*$.

Lemma 3 establishes bounds on $H(r)$. In particular, when r_k is less than f^*, there is a natural lower bound of zero.

If $r_0 < f^*$, Lemma 4 shows that r_k remains less than f^* at each iteration of Algorithm 1 and that it converges to f^* at a geometric rate.

Lemma 4 *Suppose* $\mathbf{x}_k \in \mathcal{X}$ *and* r_k *satisfy* $r_k < f^*$ *and* (5). *The update* $r_{k+1} = r_k + \alpha P(\mathbf{x}_k; r_k)$ *in Algorithm 1 guarantees (i)* $r_{k+1} < f^*$; *and (ii)* $f^* - r_{k+1} \le (1 - \alpha\theta)(f^* - r_k)$.

Proof Fix k. Given r_k and $\mathbf{x}_k$ satisfying $\alpha P(\mathbf{x}_k; r_k) < f^* - r_k$, we have

$$r_{k+1} = r_k + \alpha P(\mathbf{x}_k; r_k) < r_k + f^* - r_k = f^*.$$

In addition, it follows that

$$f^* - r_{k+1} = f^* - r_k - \alpha P(\mathbf{x}_k; r_k) \le f^* - r_k - \alpha H(r_k) \le (1 - \alpha\theta)\,(f^* - r_k), \tag{7}$$

where the first inequality holds because $0 \le H(r_k) \le P(\mathbf{x}_k; r_k)$ and the second one because $H(r_k) \ge \theta(f^* - r_k)$ for $r_k < f^*$ by Lemma 3. ■

In Theorem 5, we characterize the number of outer iterations needed by Algorithm 1 to find an ϵ-optimal and ϵ-feasible solution to (1).

Theorem 5 *Given* $\epsilon > 0$, $\alpha \in (0,1)$, *and* $r_0 < f^*$, *Algorithm 1 terminates in at most*

$$\hat{K} := \left\lceil \frac{1}{\alpha\theta} \ln\left(\frac{f^* - r_0}{\alpha\epsilon}\right) \right\rceil, \tag{8}$$

iterations and returns an ϵ*-optimal and* ϵ*-feasible solution to* (1).

Proof Since Algorithm 1 starts with $r_0 < f^*$, Lemma 4 indicates that $r_k < f^*$ and (7) holds at each iteration $k \ge 0$. Since $\alpha\theta \in (0,1)$, recursively applying (7) for $k = 0, 1, \dots$ shows that $0 \le f^* - r_k \le (1 - \alpha\theta)^k (f^* - r_0)$ for any $k \ge 0$. Hence, it follows from inequality (5) that $P(\mathbf{x}_k; r_k) < \frac{1}{\alpha}(f^* - r_k) \le \frac{1}{\alpha}(1 - \alpha\theta)^k (f^* - r_0)$. When $k = \left\lceil \frac{1}{\alpha\theta} \ln\left(\frac{f^* - r_0}{\alpha\epsilon}\right)\right\rceil$, we get $P(\mathbf{x}_k; r_k) \le \epsilon$ and the algorithm stops. Since $r_k < f^*$, Lemma 2 then guarantees that the solution $\mathbf{x}_k$ returned at termination is an ϵ-optimal and ϵ-feasible solution. ■

Remark 6 *The* $1/\theta$ *factor in the definition of* $\hat{K}$ *arises because we leverage the natural lower bound of zero on* $H(r)$, *as discussed after Lemma 3. This choice results in the specific form of condition* (5), *which is a simplification of the analogous condition in Aravkin et al. (2019), where the authors use non-trivial lower and upper bounds on* $H(r)$. *The lower bound used in Aravkin et al. (2019) is easily computable only when the problem has certain structures. While the number of iterations in the algorithm from Aravkin et al. (2019) becomes independent of* θ *when such bounds are efficiently calculable, obtaining such bounds for a broader class of problems poses a formidable challenge. We opt for our simpler approach, which proves implementable across a more general class of problems and contributes to a clearer exposition of the fundamental concepts underlying RLS in the subsequent section.*

3. Restarting Level Set Method

We define a sequence of pairs $\{(\mathbf{x}_k, r_k)\}_{k=0}^{K}$ with $r_0 < f^*$ and $\mathbf{x}_k \in \mathcal{X}$ as a *level set sequence* of length $K+1$ if $r_{k+1} = r_k + \alpha P(\mathbf{x}_k; r_k)$ holds for some $\alpha \in (0,1)$ and each $k = 0, 1, \ldots, K-1$. It follows from Theorem 5 that the level set sequence generated in Algorithm 1 converges to an ϵ-optimal and ϵ-feasible solution when (i) $\mathbf{x}_k$ satisfies (5) and (ii) $K \geq \hat{K}$, where $\hat{K}$ is defined in (8). One way to find $\mathbf{x}_k \in \mathcal{X}$ satisfying (5) is to solve the *level-set subproblem* (3) with $r = r_k$, which we denote by $\mathtt{LSP}(r_k)$:

$$\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r_k). \tag{9}$$

Let $\mathtt{fom}$ represent the first-order method applied to (9). Although there are many choices for $\mathtt{fom}$, it is difficult to numerically verify (5) because f^* is unknown. As a result, we are not able to terminate $\mathtt{fom}$ at the right time. If $\mathtt{fom}$ is terminated too soon, the returned solution $\mathbf{x}_k$ will not satisfy (5) and Algorithm 1 may not converge. If $\mathtt{fom}$ is terminated too late, Algorithm 1 will converge but consume longer run time than it actually needs.

To address this issue, we parallelize the sequential approach in Algorithm 1 by simultaneously solving multiple level-set subproblems with potentially different level parameters and restarting the solution of each such subproblem if a predetermined amount of progress in reducing its objective function has been made. We refer to this procedure as the *restarting level set* (RLS) method and describe its core idea next. It requires maintaining $K+1$ (with $K \geq \hat{K}$) first order method instances denoted by $\mathtt{fom}_0, \mathtt{fom}_1, \ldots, \mathtt{fom}_K$. Given $\mathbf{x}_{\text{ini}} \in \mathcal{X}$ and $r_{\text{ini}} < f^*$, we initialize the level set sequence $\{(\mathbf{x}_k^{(0)}, r_k)\}_{k=0}^{K}$ with $\mathbf{x}_k^{(0)} = \mathbf{x}_{\text{ini}}$ for $k = 0, 1, \ldots, K$, $r_0 = r_{\text{ini}}$, and $r_{k+1} = r_k + \alpha P(\mathbf{x}_k^{(0)}; r_k)$ for $k = 1, \ldots, K-1$. We then apply $\mathtt{fom}_k$ to solve $\mathtt{LSP}(r_k)$ starting from $\mathbf{x}_k^{(0)}$ for $k = 0, 1, \ldots, K$ and denote by $\mathbf{x}_k^{(t_k)}$ the solution computed by $\mathtt{fom}_k$ after t_k iterations. The $\mathtt{fom}$ instances communicate with each other via restarts. We initiate a *restart* at instance k' when $P(\mathbf{x}_{k'}^{(0)}; r_{k'}) \geq 0$, and the objective function of $P(\mathbf{x}, r_{k'})$ experiences a significant reduction. Specifically, the restart occurs when the solution $\mathbf{x}_{k'}^{(t_{k'})}$ satisfies the condition

$$P(\mathbf{x}_{k'}^{(t_{k'})}; r_{k'}) \leq BP(\mathbf{x}_{k'}^{(0)}; r_{k'}), \tag{10}$$

where $B \in (0,1)$ is a constant factor. The sequence of steps associated with a restart is shown in Definition 7.

Definition 7 (Restart at k') *Set $\mathbf{x}_{k'}^{(0)} = \mathbf{x}_{k'}^{(t_{k'})}$ and $r_{k+1} = r_k + \alpha P(\mathbf{x}_k^{(0)}; r_k)$ for all $k = k', k'+1, \ldots, K-1$. Restart $\mathtt{fom}_k$ from $\mathbf{x}_k^{(0)}$ for $k = k', k'+1, \ldots, K$.*

In other words, initiating a restart at $k' < K$, we reset $\mathtt{fom}_{k'}$ to begin at an updated initial solution equal to $\mathbf{x}_{k'}^{(t_{k'})}$ and solve the same level set subproblem $\mathtt{LSP}(r_{k'})$ as before the restart because $r_{k'}$ is not updated. In contrast, for indices $k = k'+1, \ldots, K$, $\mathtt{LSP}(r_k)$ changes because the restart changes r_k. This new level set subproblem is solved by $\mathtt{fom}_k$ starting from the same initial solution $\mathbf{x}_k^{(0)}$ that was used before the restart. If a restart is initiated at K, we only perform the update $\mathbf{x}_K^{(0)} = \mathbf{x}_K^{(t_K)}$. Restarts initiated at index k' have no effect on $\mathtt{fom}_k$ for $k = 0, 1, \ldots, k'-1$, so those $\mathtt{fom}$s will continue their iterations. Lemma 8 follows from our restarting updates.

Lemma 8 *The level set sequence $\{(\mathbf{x}_k^{(0)}, r_k)\}_{k=0}^K$ remains a level set sequence after each restart.*

Remark 9 *We highlight that restarts dynamically modify the precision with which subproblems need to be solved by* `fom` *instances to satisfy* (5)*. Specifically, the precision of a solution at instance k can be viewed as the deviation of $P(\mathbf{x}_k^{(t_k)}; r_k)$ from the optimal subproblem objective of $H(r_k)$. This difference satisfies*

$$P(\mathbf{x}_k^{(t_k)}; r_k) - H(r_k) \leq \frac{1}{\alpha}(f^* - r_k) - \theta(f^* - r_k) \leq (1 - \alpha\theta)P(\mathbf{x}_k^{(0)}; r_k),$$

where the first inequality follows from Lemma 3 and the fact that the solution $x_k^{(t_k)}$ satisfies (5) *and the second inequality holds since the initial solution $\mathbf{x}_k^{(0)}$ does not satisfy* (5)*. i.e. $\alpha P(\mathbf{x}_k^{(0)}; r_k) > f^* - r_k$. It follows from these inequalities that the precision $P(\mathbf{x}_k^{(t_k)}; r_k) - H(r_k)$ is $\Theta(P(\mathbf{x}_k^{(0)}; r_k))$. A restart at k' changes $\mathbf{x}_k^{(0)}$ or r_k of instances $k \geq k'$ and hence the value of $P(\mathbf{x}_k^{(0)}; r_k)$ at these instances. Therefore, RLS can be interpreted as an approach that dynamically modifies the precision required to satisfy* (5) *at a subset of instances, each time a restart is initiated as the result of there being sufficient progress in the subproblem objective function at some instance, that is, condition* (10) *is satisfied.*

Well-known choices of `fom` (e.g., subgradient method) can be shown to meet the restart condition (10) after a finite number of iterations starting from $\mathbf{x}^{(0)}$ when $P(\mathbf{x}^{(0)}; r) \geq 0$ is sufficiently large. In particular, when $P(\mathbf{x}^{(0)}; r) > \epsilon$, the number of iterations taken by `fom` to find a desirable solution can be bounded by an integer $n^{\texttt{fom}}$ that is independent of r and $\mathbf{x}^{(0)}$. We summarize this property in Assumption 2 and take it to be true in this section. We will present `fom` choices that satisfy this assumption in §4.

Assumption 2 *Consider α, B, ϵ, r, and $\mathbf{x}^{(0)} \in \mathcal{X}$ such that $0 < \alpha < B < 1$, $\epsilon > 0$, $r < f^*$, and $P(\mathbf{x}^{(0)}; r) > \epsilon$. There exists an* `fom` *initiated at $\mathbf{x}^{(0)}$ which guarantees that the inequality $P(\mathbf{x}^{(t)}; r) \leq BP(\mathbf{x}^{(0)}; r)$ holds after $t \leq n^{\texttt{fom}}$ iterations if* (5) *is not satisfied by then, where $n^{\texttt{fom}}$ is an integer depending on α, B and ϵ and independent of $\mathbf{x}^{(0)}$ and r.*

To interpret RLS, it is useful to think of instance $\texttt{fom}_k$ as being a proxy for the first-order method applied to solve $\texttt{LSP}(r_k)$ in iteration k of Algorithm 1. By Lemma 8, the relationship between the r_k values across iterations k in Algorithm 1 are also maintained for the r_k values across instances $\texttt{fom}_k$ in RLS but some r_k may not be computed based on a solution that satisfies (5). As a result, even though the initial level set parameter is less than f^*, subsequent updates can push $r_k \geq f^*$. Theorem 10 states key properties of RLS that allow it to nevertheless converge to a near-optimal and near-feasible solution, as we will discuss shortly.

Theorem 10 *Given $\epsilon > 0$ and $\alpha \in (0, 1)$, consider a level set sequence $\{(\mathbf{x}_k^{(0)}, r_k)\}_{k=0}^K$ with $K \geq \hat{K}$ where $\hat{K}$ is defined in* (8)*. At least one of the following two statements holds:*

A. There exists an index $k^* \in \{0, 1, \ldots, K\}$ such that $r_{k^*} < f^*$ and $\alpha P(\mathbf{x}_{k^*}^{(0)}; r_{k^*}) \geq f^* - r_{k^*}$;

B. The solution $\mathbf{x}_K^{(0)}$ is an ϵ-optimal and ϵ-feasible solution.

Moreover, if statement A holds, it follows that the index k^* is unique and we have $r_0 < \cdots < r_{k^*} < f^*$ and $\alpha P(\mathbf{x}_k^{(0)}; r_k) < f^* - r_k$ for $k = 1, 2, \ldots, k^* - 1$.

Proof Suppose statement A does not hold for any index $k \leq K$. We show the statement B holds. By the definition of a level set sequence, we have $r_0 < f^*$ and $r_{k+1} = r_k + \alpha P(\mathbf{x}_k^{(0)}; r_k)$ for all k. Since $\alpha P(\mathbf{x}_k^{(0)}; r_k) < f^* - r_k$ (by our assumption that A does not hold), Lemma 4 guarantees that $r_1 < f^*$. Using the same argument and by induction, Lemma 4 implies $r_k < f^*$ and $f^* - r_k \leq (1 - \alpha\theta)^k (f^* - r_0)$ for any k. Therefore, we have $P(\mathbf{x}_K^{(0)}; r_K) < 1/\alpha(f^* - r_K) \leq \frac{1}{\alpha}(1 - \alpha\theta)^K (f^* - r_0)$. Since $K \geq \hat{K}$ with $\hat{K}$ defined in (8), we have $P(\mathbf{x}_K^{(0)}; r_K) \leq \epsilon$ which indicates $\mathbf{x}_K^{(0)}$ is an ϵ-optimal and ϵ-feasible solution by Lemma 2.

Next, we prove the rest of the conclusion. Let k^* denote the smallest index satisfying the conditions of statement A. If $k^* = 0$ the conclusion is trivial since $r_0 < f^*$. Suppose $k^* > 0$. Since $r_0 < f^*$, following the definition of k^*, we must have $\alpha P(\mathbf{x}_0^{(0)}; r_0) < f^* - r_0$. Lemma 4 then guarantees that $r_1 < f^*$. Applying the same argument, we can show $r_2, ..., r_{k^*-1}$ are all less than f^* (i.e. $r_k < f^*$) and hence the inequality $\alpha P(\mathbf{x}_k^{(0)}; r_k) < f^* - r_k$ must hold for any $k < k^*$. In addition, recall that $P(\mathbf{x}; r) \geq H(r) > 0$ for any $r < f^*$. Hence, the relationship $r_{k+1} = r_k + \alpha P(\mathbf{x}_k^{(0)}; r_k)$ shows $r_{k+1} > r_k$ for any $k < k^*$. The index k^* in A is unique because it is straightforward to see $r_{k^*+1} \geq f^*$ which follows from the definition of r_{k^*+1} and the inequality $\alpha P(x_{k^*}^{(0)}, r_{k^*}) \geq f^* - r_{k^*}$. Moreover, once there exists an index k such that $r_k \geq f^*$, we must have $r_i \geq f^*$ for all $i \geq k$ since

$$r_{i+1} - f^* = r_i - f^* + \alpha P(\mathbf{x}_i^{(0)}; r_i) \geq r_i - f^* + \alpha H(r_i) \geq (1 - \alpha\theta)(r_i - f^*) \geq 0,$$

where the first inequality holds by the definition of $H(\cdot)$ and the second by Lemma 3. Therefore, given that $r_{k^*+1} \geq f^*$, we must have $r_k \geq f^*$ for any $k \geq k^* + 1$. ■

Given a level set sequence, we refer to the unique index k^* in Theorem 10 as the *critical index*. At this index, we have $r_{k^*} < f^*$, but condition (5) does not hold. For all indices k that precede the critical index, we have $r_k < f^*$ and condition (5) holds. Moreover, if $K \geq \hat{K}$ and k^* does not exist (intuitively $k^* \geq K + 1$), it implies the solution $\mathbf{x}_K^{(0)}$ is an ϵ-optimal and ϵ-feasible. Therefore, we would like to increase the critical index with restarts so that it eventually exceeds K. Ideally, if one could initiate restarts at the critical index k^* sufficiently many times, it would reduce $P(\mathbf{x}_{k^*}^{(0)}; r_{k^*})$ by updating $\mathbf{x}_{k^*}^{(0)}$ while ensuring $r_{k^*} < f^*$ (since r_{k^*} is not updated when a restart is initiated at k^* by Definition 7). Then one would expect condition (5) will hold at k^*. Moreover, when this happens, $r_{k^*+1} < f^*$ by Lemma 4. Hence, repeated restarts at the critical index should intuitively increase its value.

Unfortunately, this intuition is of little algorithmic use since the critical index is defined in terms of f^*, which is unknown. Therefore, RLS instead executes restarts at the smallest index at which condition (10) is satisfied. This index could be smaller than or greater than the critical index k^*. We refer to as *desirable restarts* the ones initiated at index k less than or equal to k^* because such restarts may update r_{k^*} or $\mathbf{x}_{k^*}^{(0)}$. By Assumption 2 and Theorem 10, a desirable restart must be initiated at $\texttt{fom}_k$ with some $k \leq k^*$ in no more than $n^{\texttt{fom}}$ iterations, unless $P(\mathbf{x}_{k^*}^{(0)}; r_{k^*}) \leq \epsilon$. In the latter case the solution $\mathbf{x}_{k^*}^{(0)}$ is ϵ-optimal and ϵ-feasible. The number of desirable restarts can be bounded by $\mathcal{O}(\log^2(1/\epsilon))$ as we will establish later. Such restarts that occur before k^* may already increase the critical index. If this does not happen, desirable restarts will start happening at k^* and increase the critical index.

Algorithm 2 formalizes the steps of RLS. The inputs to this algorithm include the number of `fom` instances K, a total budget I on the number of `fom` iterations, a level parameter $r_{\text{ini}} < f^*$, a strictly feasible solution $\mathbf{x}_{\text{ini}} \in \mathcal{X}$, and parameters α, B, and ϵ. By Assumption 1, we can set $\mathbf{x}_{\text{ini}} = \tilde{\mathbf{x}}$. For K, an ideal value to use is $\hat{K}$ defined in (8). However, since the parameters θ and f^* in this bound are unknown, we compute a bound on $\hat{K}$ by using a strictly feasible solution. In particular, we use

$$\tilde{K} := \left\lceil \frac{1}{\alpha\tilde{\theta}(r)} \ln\left(\frac{\tilde{r} - r_0}{\alpha\epsilon}\right)\right\rceil, \tag{11}$$

with $\tilde{r} := f(\tilde{\mathbf{x}}) - g(\tilde{\mathbf{x}})$ and $\tilde{\theta}(r) := g(\tilde{\mathbf{x}})/(r - \tilde{r})$ for $r < f^*$ and $\tilde{\mathbf{x}}$ from Assumption 1. We show below $\tilde{K}$ is an upper bound on $\hat{K}$. We set $r = r_{\text{ini}}$ in definition of $\tilde{\theta}(r)$ when executing Algorithm 2.

Lemma 11 *Let $\tilde{\mathbf{x}}$ be the strictly feasible solution in Assumption 1 and $\tilde{r} := f(\tilde{\mathbf{x}}) - g(\tilde{\mathbf{x}})$ and $\tilde{\theta}(r) := g(\tilde{\mathbf{x}})/(r - \tilde{r})$ for $r < f^*$. Then $\tilde{K} \geq \hat{K}$ for $\tilde{K}$ and $\hat{K}$ respectively defined in* (11) *and* (8)*.*

The proof of this lemma is provided in the Appendix to maintain a coherent flow here.

The initialization step in Algorithm 2 assigns $\mathbf{x}_{\text{ini}}$ as starting solutions to all `fom` instances (i.e. $\mathbf{x}_k^{(0)} = \mathbf{x}_{\text{ini}}$ for $k = 0, \ldots, K$) and sets the level parameters used in each $\texttt{fom}_k$ using $r_{k+1} = r_k + \alpha P(\mathbf{x}_k^{(0)}; r_k)$ starting at $r_0 = r_{\text{ini}}$. Algorithm 2 runs for a pre-specified number of `fom` iterations. At each iteration i, it runs $K+1$ `fom` iterations simultaneously, one for each instance, until the condition (10) holds for some index. It then finds a smallest index k' for which $P(\mathbf{x}_{k'}^{(0)}; r_{k'}) \geq 0$ and (10) hold. A restart is executed at k'. If the solution $\mathbf{x}_{k'}^{(t_{k'})}$ is ϵ-feasible and has a better objective function value than the best ϵ-feasible solution at hand, this solution is updated. Once the total of I iterations have been reached, the ϵ-feasible solution with the least $f_0(\mathbf{x})$ is output.

By Lemma 8, the sequence $\{(\mathbf{x}_k^{(0)}, r_k)\}_{k=0}^K$ generated at each iteration of Algorithm 2 is a level set sequence. In addition, since $K = \tilde{K} \geq \hat{K}$, Theorem 10 guarantees that the critical index k^* must exist unless an ϵ-optimal and ϵ-feasible solution is obtained. The correctness and complexity of Algorithm 2 rely on the following lemma, which we will prove in the Appendix.

Lemma 12 *The critical index k^* is non-decreasing throughout Algorithm 2.*

We count the total number of `fom` iterations taken by the RLS algorithm as the total number of iterations taken by `fom` instances between two consecutive desirable restarts times the total number of desirable restarts. Lemma 12 shows the critical index never decreases after desirable restarts. Since the critical index k^* is at most $\tilde{K}$ (by its definition), this index varies between one to $\tilde{K}$ in an increasing order until it reaches $\tilde{K}$. If the latter case happens, Assumption 2 and Theorem 10 ensure that in finite number of iterations condition (5) holds

Algorithm 2 Restarting Level Set Method

```
1: Input: $K = \tilde{K}, I \in \mathcal{Z}_+$, $\alpha \in (0,1)$, $B \in (\alpha, 1)$, $\epsilon > 0$, $r_{\text{ini}} < f^*$ and a strictly feasible $\mathbf{x}_{\text{ini}} \in \mathcal{X}$.
2: Initialization: Set $\mathbf{x}_k^{(0)} = \mathbf{x}_{\text{ini}}$, $t_k = 0$ for $k = 1, \ldots, K$, $r_0 = r_{\text{ini}}$, $r_{k+1} = r_k + \alpha P(\mathbf{x}_k^{(0)}; r_k)$ for $k = 0, 1, \ldots, K-1$, and $\mathbf{x}_{\text{best}} = \mathbf{x}_{\text{ini}}$.
3: for $i = 1, 2, \ldots, \lceil I/(K+1) \rceil$ do
   ▷ —— Execute $K$ iterations, one in each fom instance ——
4:     Run fom$_k$, $k = 0, 1, \ldots, K$, for one iteration each to obtain $\mathbf{x}_1^{(t_1+1)}, \mathbf{x}_2^{(t_2+1)}, \ldots, \mathbf{x}_K^{(t_K+1)}$.
5:     $t_k = t_k + 1$, $k = 1, \ldots, K$.
       ▷ —— Check if (10) holds and find index $k'$ to initiate restart ——
6:     if $P(\mathbf{x}_k^{(0)}; r_k) \geq 0$ and $P(\mathbf{x}_k^{(t_k)}; r_k) \leq BP(\mathbf{x}_k^{(0)}; r_k)$ for some $k = 1, \ldots, K$ then
7:         Find the smallest index $k' \geq 1$ such that $P(\mathbf{x}_{k'}^{(0)}; r_{k'}) \geq 0$ and $P(\mathbf{x}_{k'}^{(t_{k'})}; r_{k'}) \leq BP(\mathbf{x}_{k'}^{(0)}; r_{k'})$.
8:         Set $\mathbf{x}'_{k'} := \arg\min_{\mathbf{x} \in \left\{\mathbf{x}_{k'}^{(t_{k'})}, \mathbf{x}_1^{(0)}, \mathbf{x}_2^{(0)}, \ldots, \mathbf{x}_K^{(0)}\right\}} P(\mathbf{x}; r_{k'})$.
9:     end if
       ▷ —— Execute restart at $k'$ ——
10:    $\mathbf{x}_{k'}^{(0)} \leftarrow \mathbf{x}'_{k'}$.
11:    $t_k = 0$ for $k = k', k'+1, \ldots, K$.
12:    for $k = k', ..., K-1$ do
13:        Set $\mathbf{x}_k^{(0)} := \arg\min_{\mathbf{x} \in \left\{\mathbf{x}_1^{(0)}, \mathbf{x}_2^{(0)}, \ldots, \mathbf{x}_K^{(0)}\right\}} P(\mathbf{x}; r_k)$.
14:        $r_{k+1} \leftarrow r_k + \frac{\alpha}{2} P(\mathbf{x}_k^{(0)}; r_k)$.
15:    end for
       ▷ —— Update best $\epsilon$-feasible solution ——
16:    If $\mathbf{x}'_{k'}$ is $\epsilon$-feasible and $f_0(\mathbf{x}'_{k'}) < f_0(\mathbf{x}_{\text{best}})$ then $\mathbf{x}_{\text{best}} = \mathbf{x}'_{k'}$.
17: end for
18: Output: $\mathbf{x}_{\text{best}}$.
```

at all `fom` instances and hence an ϵ-optimal and ϵ-feasible solution must be found. Let D denote the total number of desirable restarts before an ϵ-optimal and ϵ-feasible solution is found. Using the argument above, we can compute D as follows. Suppose D_{k,k^*} shows the number of desirable restarts starting at $\texttt{fom}_k$ before an ϵ-optimal and ϵ-feasible solution is found with $k \leq k^*$, assuming k^* is the current critical index. Then

$$D \;:= \textstyle\sum_{k^*=0}^{\tilde{K}} D_{k^*,k^*} + \sum_{k^*=0}^{\tilde{K}} \sum_{k=0}^{k^*-1} D_{k,k^*}. \tag{12}$$

The first sum in the above equation accounts for the total number of desirable restarts that occur at each $\texttt{fom}_{k^*}$ for all possible critical index $k^* = 0, 1, \ldots, \tilde{K}$. The second term however is the total number of desirable restarts that start at each $\texttt{fom}_k$ instance with $k < k^*$, where k^* is the critical index and varies from zero to $\tilde{K}$. Notice that some of the values between zero and $\tilde{K}$ in the above sums cannot be critical index. This could happen if the critical index is strictly positive at the beginning of Algorithm 2. In this case, all the smaller indices will never be a critical index since by Lemma 12, critical index does not decrease throughout the algorithm. We assume $D_{kk^*} = D_{k^*k^*} = 0$ for such cases. The equation (12) can be re-written as

$$D \;\; := \textstyle\sum_{k^*=0}^{\tilde{K}} D_{k^*,k^*} + \sum_{k=0}^{\tilde{K}-1} \sum_{k^*=k+1}^{\tilde{K}} D_{k,k^*}. \tag{13}$$

The equivalence between the second sum in (12) and the second one in (13) can be explained by considering each D_{k,k^*} as an element of an upper-triangle matrix where k is a row index and k^* is a column index. Then the sums of elements over rows and columns can be exchanged.

We provide upper bounds for D_{k^*,k^*} and $\sum_{k^*=k+1}^{\tilde{K}} D_{k,k^*}$ respectively in Lemma 26 and Lemma 25 in the Appendix. Using these bounds in (13), we obtain the total complexity of Algorithm 2 in Theorem 13.

Theorem 13 *Consider α, β, ϵ, and r_{ini} such that $0 < \alpha < B < 1$, $\epsilon > 0$, and $r_{ini} < f^*$. Suppose Algorithm 2 is executed for $K = \tilde{K}$ where $\tilde{K}$ is defined in (11) with $r = r_{ini}$. This algorithm returns an ϵ-optimal and ϵ-feasible solution in at most*

$$(\tilde{K}+1) \cdot n^{\texttt{fom}} \cdot \tilde{D} = \mathcal{O}\left(n^{\texttt{fom}} \log^2\left(\frac{1}{\epsilon}\right)\right), \tag{14}$$

`fom` *iterations where $n^{\texttt{fom}}$ is as in Assumption 2 and $\tilde{D}$ is an upper bound on D that can be written as*

$$\tilde{D} := \left(\tilde{K}+1\right) \ln(2/\alpha\theta)/\ln(1/B) + \tilde{K} \cdot \ln\left(4 - \alpha\theta/3\alpha\theta\right)/\ln\left(1 + \alpha(1-B)\theta/4\right) = \mathcal{O}\left(\log\left(\frac{1}{\epsilon}\right)\right). \tag{15}$$

Proof Suppose C and D respectively denote the number of `fom` iterations between two consecutive desirable restarts and the total number of desirable restarts performed by Algorithm 2 until an ϵ-optimal and ϵ-feasible solution is found. It then follows that the total number of `fom` iterations required by Algorithm 2 is $C \cdot D$, where D is expressed in (13). To obtain the iteration complexity, we only require to bound D and C.

An upper bound on C: Consider a desirable restart and assume the critical index is k^* after this restart. Suppose an ϵ-optimal and ϵ-feasible solution has not been found. We must have $P(x_{k^*}^{(0)}; r_{k^*}) > \epsilon$ by Lemma 2. Assumption 2 indicates that in $t_{k^*} \le n^{\texttt{fom}}$ iterations, the inequality $P(x_{k^*}^{(t_{k^*})}; r_{k^*}) \le BP(x_{k^*}^{(0)}; r_{k^*})$ holds. This means that the next desirable restart must occur at one of the $\tilde{K}+1$ `fom` instances in at most $n^{\texttt{fom}}$ iterations, and $n^{\texttt{fom}}$ is independent of $\mathbf{x}^{(0)}$ and r by Assumption 2. Since each iteration of Algorithm 2 simultaneously runs $\tilde{K}+1$ `fom` iterations, we get

$$C \le (\tilde{K}+1) \cdot n^{\texttt{fom}}. \tag{16}$$

An upper bound on D: We claim that for any $k = 0, 1, \ldots, \tilde{K}-1$,

$$\sum_{k^*=k+1}^{\tilde{K}} D_{k,k^*} \le \ln\left(4 - \alpha\theta/3\alpha\theta\right)/\ln\left(1 + \alpha(1-B)\theta/4\right) \tag{17}$$

and for any $k^* = 0, 1, \ldots, \tilde{K}$,

$$D_{k^*,k^*} \leq \ln(2/\alpha\theta)/\ln(1/B). \tag{18}$$

We formally prove these claims in Lemma 25 and Lemma 26 in the Appendix. From the above inequalities, it follows that the total number of desirable restarts can be bounded by

$$\begin{aligned} D &= \sum_{k^*=0}^{\tilde{K}} D_{k^*,k^*} + \sum_{k=0}^{\tilde{K}-1} \sum_{k^*=k+1}^{\tilde{K}} D_{k,k^*} \\ &\leq \left(\tilde{K}+1\right) \ln(2/\alpha\theta)/\ln(1/B) + \tilde{K} \cdot \ln\left(4 - \alpha\theta/3\alpha\theta\right)/\ln\left(1 + \alpha(1-B)\theta/4\right) = \tilde{D}. \end{aligned} \tag{19}$$

The bound (14) then can be obtained by multiplying (16) and (19). ■

The key remaining question to implement Algorithm 2 is the following: What `fom` satisfies Assumption 2 and what is $n^{\texttt{fom}}$? In the following sections, we will present different `foms` that satisfy Assumption 2 for the non-smooth and smooth problems and provide a total complexity bound for Algorithm 2 under each case.

4. First Order Subroutines

In this section, we provide two different first order methods (`foms`) for smooth and non-smooth problems that can be used as subroutines in Algorithm 2 to solve $\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r)$. In particular, when the functions f_i, $i = 0, 1, \ldots, m$ defining $P(\mathbf{x}; r)$ are non-smooth, we use the standard subgradient method as an `fom`, which is an optimal algorithm for solving general non-smooth problems. When the functions f_i, $i = 0, 1, \ldots, m$ are smooth, we still need to solve the minimization of a non-smooth function since $P(\mathbf{x}; r)$ is non-smooth because of the $\max\{\cdot\}$ operator in its definition. In this case, we discuss the accelerated proximal linear method (Drusvyatskiy and Paquette, 2019), which is an extension of proximal gradient algorithm by Beck and Teboulle (2009) and Nesterov (2013). This algorithm harnesses the inherent smoothness of the functions enclosed within the max operator and yields a better convergence result.

Since all $\texttt{fom}_k$, $k = 0, 1, \ldots, \tilde{K}$ follow the same steps, we drop the index k in $\mathbf{x}_k^{(t_k)}$ and r_k used in the first order methods described in this section to simplify our notations.

4.1 Non-Smooth Case

First, let's assume the functions f_i, $i = 0, 1, \ldots, m$ are non-smooth and $\partial f_i(\mathbf{x})$ denotes the set of subgradient of f_i at $\mathbf{x}$. Suppose Assumption 1 and the error bound condition (2) hold. We next show that the subgradient method using a specific step length rule satisfies Assumption 2 and can be used to solve the level-set subproblem $\min\limits_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}; r)$.

Let $\mathbf{x}^{(0)} \in \mathcal{X}$ be an initial solution, the subgradient updates can be presented as

$$\mathbf{x}^{(t+1)} = \text{Proj}_{\mathcal{X}}\left(\mathbf{x}^{(t)} - \eta^{(t)}\boldsymbol{\xi}_r^{(t)}\right), \quad t = 0, 1, \ldots, \tag{20}$$

where $\mathrm{Proj}_{\mathcal{X}}(\cdot)$ denotes the projection onto $\mathcal{X}$, $\eta^{(t)} > 0$ a step size, and $\boldsymbol{\xi}_r^{(t)} \in \partial P(\mathbf{x}^{(t)}; r)$ a subgradient of $P(\mathbf{x}^{(t)}; r)$ with respect to $\mathbf{x}^{(t)}$. Recall that the projection mapping $\mathrm{Proj}_{\mathcal{X}}(\cdot)$ is easily computable since we assume the set $\mathcal{X}$ is simple (e.g., $\mathbb{R}^n$, a box, or a ball). The output of the subgradient method can be chosen as the historically best iterate, i.e.,

$$\bar{\mathbf{x}}^{(t)} := \operatorname*{arg\,min}_{s=0,1,\ldots,t} P(\mathbf{x}^{(s)}; r). \tag{21}$$

Proposition 14 below presents a well-known convergence result of the subgradient method.

Proposition 14 (See Theorem 3.2.2 in Nesterov (2018)) *Consider $r < f^*$. Let $\mathbf{x}^* = \mathit{Proj}_{\mathcal{X}^*}(\mathbf{x}^{(0)})$ and $\bar{\mathbf{x}}^{(t)}$ be defined as in (21). The subgradient method in (20) guarantees that for any $t \geq 0$,*

$$P(\bar{\mathbf{x}}^{(t)}; r) - P(\mathbf{x}^*; r) \leq \frac{\mathit{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)^2 + \sum_{s=0}^{t} (\eta^{(s)})^2 \|\boldsymbol{\xi}_r^{(s)}\|^2}{2\sum_{s=0}^{t} \eta^{(s)}}. \tag{22}$$

Notice that $\mathbf{x}^*$ in the above proposition could be the projection of any $\mathbf{x}^{(t)}$, for $t \geq 0$, onto $\mathcal{X}^*$ since by definition of $\mathcal{X}^*$ and $P(\mathbf{x}; r)$, it follows that $P(\mathrm{Proj}_{\mathcal{X}^*}(\mathbf{x}^{(t)}); r) = f^* - r$ for all $t \geq 0$.

We will shortly show that the subgradient method discussed above can be used as an `fom` in Algorithm 2 if the stepsizes are carefully chosen. However, before proceeding with that, we make the following assumptions to ensure the rate of change of the function $P(\cdot; r)$ does not become arbitrarily steep at the points generated by the subgradient algorithm.

Assumption 3 *Consider the initial solution $\mathbf{x}^{(0)} \in \mathcal{X}$, there exists M such that $\max\limits_{\boldsymbol{\xi} \in \partial f_i(\mathbf{x})} \|\boldsymbol{\xi}\| \leq M$ for $i = 0, 1, \ldots, m$ and any $\mathbf{x} \in \mathcal{X}$ that satisfies $\mathit{dist}(\mathbf{x}, \mathcal{X}^*) \leq \mathit{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)$.*

Notice that this assumption is more lenient than the standard assumption of bounded subgradients across the entire domain which is commonly used in the literature. This standard assumption implies that $\mathrm{dist}(\mathbf{x}, \mathcal{X}^*)$ is bounded for any $\mathbf{x} \in \mathcal{X}$ when $d > 1$. In contrast, we only require subgradients to be bounded when we are in proximity to the set of optimal solutions.

Proposition 15 below shows that the inequality $\mathrm{dist}(\mathbf{x}^{(t)}, \mathcal{X}^*) \leq \mathrm{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)$ holds at any solution $\mathbf{x}^{(t)}$ generated by the subgradient algorithm. Therefore, under Assumption 3, it easily follows that $\left\|\boldsymbol{\xi}_r^{(t)}\right\| \leq M$ for any $\boldsymbol{\xi}_r^{(t)} \in \partial P(\mathbf{x}^{(t)}; r)$ and any $r < f^*$.

Proposition 15 *Consider α, B, ϵ, r, and $\mathbf{x}^{(0)} \in \mathcal{X}$ such that $0 < \alpha < B < 1$, $\epsilon > 0$, $r < f^*$, and $P(\mathbf{x}^{(0)}; r) > \epsilon$. Let $\eta^{(t)} := \frac{(B-\alpha)P(\mathbf{x}^{(0)}; r)}{\|\boldsymbol{\xi}_r^{(t)}\|^2}$. Suppose $\mathbf{x}^{(0)}$ does not satisfy (5), i.e. $\alpha P(\mathbf{x}^{(0)}; r) > f^* - r$. At any iteration t of the subgradient method, we have*

$$\mathit{dist}(\mathbf{x}^{(t)}, \mathcal{X}^*) \leq \mathit{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*),$$

until a desirable solution is found. i.e. $P(\bar{\mathbf{x}}^{(t)}; r) \leq BP(\mathbf{x}^{(0)}; r)$.

Proof Consider $t \geq 1$ and suppose $P(\bar{\mathbf{x}}^{(t)}; r) > BP(\mathbf{x}^{(0)}; r)$ which indicates that $P(\mathbf{x}^{(s)}; r) > BP(\mathbf{x}^{(0)}; r)$ for $s = 0, \ldots, t$. Let $\mathbf{x}^* = \mathrm{Proj}_{\mathcal{X}^*}(\mathbf{x}^{(t-1)}) \in \mathcal{X}^*$. Since $r < f^*$, from definition of $P(\mathbf{x}; r)$ it follows that

$$P(\mathbf{x}^*; r) = f^* - r. \tag{23}$$

Hence, for any $s = 0, \dots, t$, we have

$$P(\mathbf{x}^*; r) = f^* - r < \alpha P(\mathbf{x}^{(0)}; r) < \frac{\alpha}{B} P(\mathbf{x}^{(s)}; r). \tag{24}$$

By the standard analysis of subgradient methods, we have

$$\begin{aligned}
\mathrm{dist}(\mathbf{x}^{(t)}, \mathcal{X}^*)^2 &\leq \mathrm{dist}(\mathbf{x}^{(t-1)}, \mathcal{X}^*)^2 - 2\eta^{(t-1)} \left\langle \boldsymbol{\xi}_r^{(t-1)}, \mathbf{x}^{(t-1)} - \mathbf{x}^* \right\rangle + \left(\eta^{(t-1)}\right)^2 \left\|\boldsymbol{\xi}_r^{(t-1)}\right\|^2 \\
&\leq \mathrm{dist}(\mathbf{x}^{(t-1)}, \mathcal{X}^*)^2 - 2\eta^{(t-1)} \left(P(\mathbf{x}^{(t-1)}; r) - P(\mathbf{x}^*; r)\right) + \left(\eta^{(t-1)}\right)^2 \left\|\boldsymbol{\xi}_r^{(t-1)}\right\|^2 \\
&\leq \mathrm{dist}(\mathbf{x}^{(t-1)}, \mathcal{X}^*)^2 - 2\eta^{(t-1)}(B-\alpha)P(\mathbf{x}^{(0)}; r) + \eta^{(t-1)}(B-\alpha)P(\mathbf{x}^{(0)}; r) \\
&= \mathrm{dist}(\mathbf{x}^{(t-1)}, \mathcal{X}^*)^2 - \eta^{(t-1)}\,(B-\alpha)\,P(\mathbf{x}^{(0)}; r) \\
&\leq \mathrm{dist}(\mathbf{x}^{(t-1)}, \mathcal{X}^*)^2 - (B-\alpha)^2 \frac{P(\mathbf{x}^{(0)}; r)^2}{\|\boldsymbol{\xi}_r^{(t-1)}\|^2},
\end{aligned}$$

where the second inequality holds by the convexity of $P(\mathbf{x}; r)$ at $\mathbf{x}$ and the third inequality by (24) and the definition of $\eta^{(t)}$. Since $(B-\alpha)^2 \frac{P(\mathbf{x}^{(0)};r)^2}{\|\boldsymbol{\xi}_r^{(t-1)}\|^2} \geq 0$, the above inequality indicates that $\mathrm{dist}(\mathbf{x}^{(t)}, \mathcal{X}^*)^2 \leq \mathrm{dist}(\mathbf{x}^{(t-1)}, \mathcal{X}^*)^2 \leq \ldots \leq \mathrm{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)^2$. ∎

Proposition 16 indicates that the subgradient method with a specific choice of stepsize satisfies Assumption 2 and therefore it can be used as an `fom` in Algorithm 2. Notice that some of the arguments used in the proof of Proposition 16 are borrowed from the proof of Proposition 3.2 in Freund and Lu (2018).

Proposition 16 *Consider α, B, ϵ, r, and $\mathbf{x}^{(0)} \in \mathcal{X}$ such that $0 < \alpha < B < 1$, $\epsilon > 0$, $r < f^*$, and $P(\mathbf{x}^{(0)}; r) > \epsilon$. Let $\eta^{(t)} := \frac{(B-\alpha)P(\mathbf{x}^{(0)};r)}{\|\boldsymbol{\xi}_r^{(t)}\|^2}$. The subgradient method satisfies Assumption 2 with*

$$n^{\texttt{fom}} = \left\lceil \frac{M^2 G^{2/d}}{(B-\alpha)^2 \epsilon^{2-2/d}} \right\rceil - 1. \tag{25}$$

Proof Suppose $\mathbf{x}^{(0)}$ does not satisfy (5), i.e. $\alpha P(\mathbf{x}^{(0)}; r) > f^* - r$. We show $P(\bar{\mathbf{x}}^{(t)}; r) \leq BP(\mathbf{x}^{(0)}; r)$ for $t \leq n^{\texttt{fom}}$.

Let $\mathbf{x}^* = \mathrm{Proj}_{\mathcal{X}^*}(\mathbf{x}^{(0)}) \in \mathcal{X}^*$. Plugging the definition of $\eta^{(t)}$ into (22) we get

$$\begin{aligned}
P(\bar{\mathbf{x}}^{(t)}; r) &\leq P(\mathbf{x}^*; r) + \frac{1}{2\sum_{s=0}^t \eta^{(s)}} \left[\mathrm{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)^2 + \sum_{s=0}^t \eta^{(s)} \frac{(B-\alpha)P(\mathbf{x}^{(0)}; r)}{\|\boldsymbol{\xi}_r^{(t)}\|^2} \|\boldsymbol{\xi}_r^{(s)}\|^2\right] \\
&\leq P(\mathbf{x}^*; r) + \frac{\mathrm{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)^2}{2(B-\alpha)P(\mathbf{x}^{(0)}; r)\sum_{s=0}^t \|\boldsymbol{\xi}_r^{(s)}\|^{-2}} + \frac{B-\alpha}{2} P(\mathbf{x}^{(0)}; r) \\
&\leq f^* - r + \frac{M^2 \mathrm{dist}(\mathbf{x}^{(0)}, \mathcal{X}^*)^2}{2(t+1)(B-\alpha)P(\mathbf{x}^{(0)}; r)} + \frac{B-\alpha}{2} P(\mathbf{x}^{(0)}; r) \\
&\leq f^* - r + \frac{M^2 G^{2/d} P(\mathbf{x}^{(0)}; f^*)^{2/d}}{2(t+1)(B-\alpha)P(\mathbf{x}^{(0)}; r)} + \frac{B-\alpha}{2} P(\mathbf{x}^{(0)}; r) \\
&\leq f^* - r + \frac{M^2 G^{2/d} P(\mathbf{x}^{(0)}; r)^{2/d-1}}{2(t+1)(B-\alpha)} + \frac{B-\alpha}{2} P(\mathbf{x}^{(0)}; r),
\end{aligned} \tag{26}$$

where the third inequality follows from (23) and the inequality $\|\boldsymbol{\xi}_r^{(t)}\| \leq M$ which holds by Assumption 3. The fourth inequality follows from the error bound condition inequality (2) and definition of $P(\mathbf{x};r)$, and the last from the inequality $P(\mathbf{x}^{(0)};f^*) \leq P(\mathbf{x}^{(0)};r)$ which holds because $r \leq f^*$. Let

$$t = \left\lceil \frac{M^2 G^{2/d}}{(B-\alpha)^2 P(\mathbf{x}^{(0)};r)^{2-2/d}} \right\rceil - 1. \tag{27}$$

Using (27) in (26), we get $P(\bar{\mathbf{x}}^{(t)};r) \leq f^* - r + \frac{B-\alpha}{2}P(\mathbf{x}^{(0)};r) + \frac{B-\alpha}{2}P(\mathbf{x}^{(0)};r) \leq BP(\bar{\mathbf{x}}^{(0)};r)$, where the second inequality is by the assumption that $\alpha P(\mathbf{x}^{(0)};r) > f^* - r$. Since $P(\mathbf{x}^{(0)};r) > \epsilon$, we have $t \leq n^{\texttt{fom}}$. ■

Remark 17 *The step length $\eta = (B-\alpha)P(\mathbf{x}^{(0)};r)/\|\boldsymbol{\xi}_r\|^2$ used in subgradient algorithm is likely to reduce at the instance that is restarted since $P(\mathbf{x}^{(0)};r)$ becomes smaller as a result of updating the initial solution with a solution that satisfies* (10)*. For all the subsequent instances, the step length after the restart would likely be larger since $P(\mathbf{x}^{(0)};r)$ is a decreasing function of r.*

Corollary 18 below provides the overall complexity of Algorithm 2 by counting the total number of subgradient and function evaluations required in this algorithm.

Corollary 18 *Suppose α, β, ϵ, and r_{ini} are such that $0 < \alpha < B < 1$, $\epsilon > 0$, and $r_{ini} < f^*$. Consider $\tilde{K}$ defined in* (11) *for $r = r_{ini}$. Suppose we execute Algorithm 2 with subgradient method as $\texttt{fom}_k$ instances for $k = 0, 1, \ldots, \tilde{K}$. This algorithm finds an ϵ-optimal and ϵ-feasible solution in at most*

$$(m+1)\cdot(\tilde{K}+1)\cdot\left\lceil \frac{M^2 G^{2/d}}{(B-\alpha)^2 \epsilon^{2-2/d}} \right\rceil \cdot \tilde{D} = \mathcal{O}\left(\log^2\left(\frac{1}{\epsilon}\right)\frac{1}{\epsilon^{2-2/d}}\right),$$

subgradient or function evaluations, where $\tilde{D}$ is defined in (15)*.*

Proof Theorem 13 guarantees that an ϵ-feasible and ϵ-optimal solution can be found in at most

$$(\tilde{K}+1)\cdot n^{\texttt{fom}}\cdot\tilde{D}, \tag{28}$$

iterations where $\tilde{K}$, $\tilde{D}$, and $n^{\texttt{fom}}$ are respectively defined in (11), (15), and (25).

It is easy to verify that the subgradient algorithm requires $m+1$ subgradients and function evaluations (one function evaluation for each function in $P(\mathbf{x};r)$). Hence, the expression (28) indicates that the total number of subgradient computations required in Algorithm 2 is

$$(m+1)\cdot(\tilde{K}+1)\cdot n^{\texttt{fom}}\cdot\tilde{D}. \tag{29}$$

The proof is then completed using (25) in (29). ■

4.2 Smooth Case

In this section, we assume Assumption 1 and the error bound condition (2) hold. Moreover, we make an additional assumption about the smoothness of the functions f_i, $i = 0, 1, \ldots, m$ in (1).

Assumption 4 *Functions f_i, $i = 0, 1, \ldots, m$, are L-smooth on $\mathcal{X}$ for some $L \geq 0$. In other words, the functions f_i, $i = 0, 1, \ldots, m$, are differentiable and $f_i(\mathbf{x}) \leq f_i(\mathbf{y}) + \langle \nabla f_i(y), \mathbf{x} - \mathbf{y} \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2$ for any $\mathbf{x}$ and $\mathbf{y}$ in $\mathcal{X}$.*

It is important to note that while Assumption 2 guarantees the smoothness of the functions f_i, the function $P(\mathbf{x}; r)$ may not possess this property due to the presence of the $\max\{\cdot\}$ operator in its definition. To address this concern, we use the prox-linear method introduced in Drusvyatskiy and Paquette (2019). In particular, we define a composite linear approximation of $P(\mathbf{x}; r)$ at a specific point $\mathbf{y} \in \mathcal{X}$ that is:

$$P(\mathbf{x}; \mathbf{y}, r) \;\; := \max_{i=1,\ldots,m} \left\{ f_0(\mathbf{y}) + \langle \mathbf{x} - \mathbf{y}, \nabla f_0(\mathbf{y}) \rangle - r, f_i(\mathbf{y}) + \langle \mathbf{x} - \mathbf{y}, \nabla f_i(\mathbf{y}) \rangle \right\}. \quad (30)$$

Given the L-smoothness of the functions f_i, $i = 0, 1, \ldots, m$ in Assumption 4 and the convexity of these functions in $\mathbf{x}$, it is straightforward to verify that for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$,

$$P(\mathbf{x}; \mathbf{y}; r) \leq P(\mathbf{x}; r) \leq P(\mathbf{x}; \mathbf{y}; r) + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2. \quad (31)$$

This inequality implies that to solve $\min_{\mathbf{x} \in \mathcal{X}} P(\mathbf{x}; r)$, one can apply a majorization-minimization procedure known as the *prox-linear method*, which solves the following subproblem at each iteration:

$$\mathbf{x}^{(t+1)} \in \arg\min_{\mathbf{x} \in \mathcal{X}} P(\mathbf{x}; \mathbf{x}^{(t)}, r) + \frac{1}{2\eta}\|\mathbf{x} - \mathbf{x}^{(t)}\|^2. \quad (32)$$

Here, $\eta > 0$ is a step size parameter (typically chosen as $\eta = \frac{1}{L}$). We assume that this strongly convex subproblem can be efficiently solved. There are various methods available to tackle this problem; for instance, one can employ the simplex method (Wolfe, 1959) or utilize the algorithm outlined in Chambolle and Pock (2011).

While the basic prox-linear method in (32) can be used to minimize $P(\mathbf{x}; r)$, it does not achieve the optimal convergence rate. A faster alternative is the *accelerated prox-linear method*, which introduces two auxiliary sequences $\mathbf{y}^{(t)}$ and $\mathbf{z}^{(t)}$, in addition to the primary sequence $\mathbf{x}^{(t)}$.

Starting from an initial point $\mathbf{x}^{(0)} \in \mathcal{X}$ and setting $\mathbf{z}^{(0)} = \mathbf{x}^{(0)}$, the algorithm proceeds in iterations. At each iteration t, it computes an extrapolated point: $\mathbf{y}^{(t)} = a^{(t)}\mathbf{z}^{(t)} + (1 - a^{(t)})\mathbf{x}^{(t)}$, where $a^{(t)} \in (0, 1]$ is a weighting parameter. The next iterate $\mathbf{x}^{(t+1)}$ is then obtained by solving the prox-linear subproblem:

$$\mathbf{x}^{(t+1)} \in \arg\min_{\mathbf{x} \in \mathcal{X}} P(\mathbf{x}; \mathbf{y}^{(t)}, r) + \frac{1}{2\eta}\|\mathbf{x} - \mathbf{y}^{(t)}\|^2, \quad (33)$$

where $\eta = 1/L$. Finally, the auxiliary sequence is updated as $\mathbf{z}^{(t+1)} = \mathbf{x}^{(t)} + \frac{1}{a^{(t)}}(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)})$, enabling the method to exploit acceleration.

In practice, the step size $\eta = \frac{1}{L}$ may not be known a priori or may be conservatively estimated. A standard remedy is to incorporate a backtracking line search (Beck and Teboulle, 2009; Nesterov, 2013), as shown in Algorithm 3, which dynamically adjusts η. The procedure starts with a guess $\eta^{(t)}$ for $1/L$, solves (33) for $\eta = \eta^{(t)}$, and checks whether the inequality

$$P(\mathbf{x}^{(t+1)}; r) \leq P(\mathbf{x}^{(t+1)}; \mathbf{y}^{(t)}, r) + \frac{1}{2\eta^{(t)}} \|\mathbf{x}^{(t+1)} - \mathbf{y}^{(t)}\|^2,$$

holds. By inequality (31), this condition is satisfied whenever $\eta^{(t)} \leq \frac{1}{L}$. If the condition fails, then $\eta^{(t)} > \frac{1}{L}$, and the estimate is decreased by a factor $\beta_{\text{dec}} < 1$. The subproblem is resolved with the new $\eta^{(t)}$, and the process continues until the condition is met.

Once the line search is successful, $\mathbf{x}^{(t+1)}$ and the corresponding $\eta^{(t)}$ are returned. Additionally, to allow for potentially larger step sizes in future iterations, the step size is increased by a factor $\beta_{\text{inc}} > 1$ and used as the initial estimate in the next iteration. This is implemented in Line 3 of Algorithm 4.

Algorithm 3 BacktrackingLineSearch

```
Require: Outer iterate $\mathbf{x}^{(t)}$, extrapolated point $\mathbf{z}^{(t)}$, previous weight $a^{(t-1)}$, level parameter
    $r < f^*$, previous step size $\eta^{(t-1)}$, initial trial step size $\bar{\eta}$, backtracking parameter
    $\beta_{\text{dec}} \in (0,1)$.
1: Initialize: $\eta \leftarrow \beta_{\text{dec}}^{-1}\bar{\eta}$
2: repeat
3:     $\eta \leftarrow \beta_{\text{dec}} \cdot \eta$
4:     Compute $a$ by solving:
           $\frac{(a^{(t-1)})^2}{\eta^{(t-1)}}(1-a) = \frac{a^2}{\eta}$
5:     Compute extrapolated point:
           $\mathbf{y} = a \cdot \mathbf{z}^{(t)} + (1-a) \cdot \mathbf{x}^{(t)}$
6:     Compute candidate update:
           $\mathbf{x}^+ = \arg\min_{\mathbf{x} \in \mathcal{X}} \left\{ P(\mathbf{x}; \mathbf{y}, r) + \frac{1}{2\eta} \|\mathbf{x} - \mathbf{y}\|^2 \right\}$
7: until $P(\mathbf{x}^+; r) \leq P(\mathbf{x}^+; \mathbf{y}, r) + \frac{1}{2\eta} \|\mathbf{x}^+ - \mathbf{y}\|^2$
8: return $(\mathbf{x}^+, \mathbf{y}, \eta, a)$
```

Theorem 19 *Given $\eta^{(ini)} > 0$, $\beta_{inc} > 1$, $\beta_{dec} \in (0,1)$, and $r < f^*$, Algorithm 4 guarantees that for any $\mathbf{x} \in \mathcal{X}$ and $t \geq 0$,*

$$P(\mathbf{x}^{(t)}; r) - P(\mathbf{x}; r) \leq \frac{4 \max\left\{\frac{1}{\eta^{(ini)}}, \frac{L}{\beta_{dec}}\right\}}{(t+1)^2} \left\{ \eta^{(ini)} \left[P(\mathbf{x}^{(0)}; r) - P(\mathbf{x}; r) \right] + \frac{1}{2} \left\| \mathbf{x}^{(0)} - \mathbf{x} \right\|^2 \right\}. \tag{34}$$

```
Algorithm 4 Accelerated Prox-Linear Method with Backtracking Line Search
Require: Level parameter $r \le f^*$, initial point $\mathbf{x}^{(0)} \in X$, initial step size $\eta^{(\mathrm{ini})} > 0$, and
    backtracking parameters $\beta_{\mathrm{dec}} \in (0,1)$ and $\beta_{\mathrm{inc}} > 1$.
1: Initialize: set $\mathbf{z}^{(0)} \leftarrow \mathbf{x}^{(0)}$, $a^{(-1)} \leftarrow 1$ and $\eta^{(-1)} \leftarrow \eta^{(\mathrm{ini})}$.
2: for $t = 0, 1, 2, \dots$ do
3:     Set $\bar{\eta} \leftarrow \min\{\beta_{\mathrm{inc}}\, \eta^{(t-1)}, \eta^{(\mathrm{ini})}\}$.
4:     $(\mathbf{x}^{(t+1)}, \mathbf{y}^{(t)},\ \eta^{(t)},\ a^{(t)}) \leftarrow$ BacktrackingLineSearch$(\mathbf{x}^{(t)}, \mathbf{z}^{(t)}, a^{(t-1)}, r, \eta^{(t-1)}, \bar{\eta}, \beta_{\mathrm{dec}})$.
5:     Update the extrapolated solution:
           $$\mathbf{z}^{(t+1)} = \mathbf{x}^{(t)} + \frac{1}{a^{(t)}}\Big(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}\Big).$$
6: end for
7: return $\mathbf{x}^{(t+1)}$.
```

Moreover, the total number of prox-linear subproblems solved in Algorithm 3 is at most

$$\left[1 + \frac{\ln(\beta_{inc})}{\ln(\beta_{dec}^{-1})}\right] t + \frac{1}{\ln(\beta_{dec}^{-1})} \ln\left(\max\left\{1, \frac{\eta^{(ini)} L}{\beta_{dec}}\right\}\right) = \mathcal{O}(t).$$

Proof Let $t \ge 0$. In this proof, we denote by $\eta^{(t)}$, $a^{(t)}$, $\mathbf{x}^{(t+1)}$, and $\mathbf{y}^{(t)}$ the final step size, weight, solution, and extrapolated point returned by the backtracking line search at the t-th outer iteration of Algorithm 4, i.e., after the internal subroutine, Algorithm 3, has terminated.

Since $\mathbf{y}^{(t)} = a^{(t)}\mathbf{z}^{(t)} + (1 - a^{(t)})\mathbf{x}^{(t)}$, we have $\mathbf{z}^{(t)} - \mathbf{y}^{(t)} = -\frac{1-a^{(t)}}{a^{(t)}}(\mathbf{x}^{(t)} - \mathbf{y}^{(t)})$. Let $\hat{\mathbf{x}} = a^{(t)}\mathbf{x} + (1 - a^{(t)})\mathbf{x}^{(t)}$ for $\mathbf{x} \in \mathcal{X}$. Then: $\hat{\mathbf{x}} - \mathbf{y}^{(t)} = a^{(t)}(\mathbf{x} - \mathbf{y}^{(t)}) + (1 - a^{(t)})(\mathbf{x}^{(t)} - \mathbf{y}^{(t)})$, which implies

$$\hat{\mathbf{x}} - \mathbf{y}^{(t)} = a^{(t)}(\mathbf{x} - \mathbf{z}^{(t)}). \tag{35}$$

Also, using $\mathbf{z}^{(t+1)} = \mathbf{x}^{(t)} + \frac{1}{a^{(t)}}(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)})$ and $\hat{\mathbf{x}} = a^{(t)}\mathbf{x} + (1 - a^{(t)})\mathbf{x}^{(t)}$, it follows that:

$$\|\hat{\mathbf{x}} - \mathbf{x}^{(t+1)}\|^2 = (a^{(t)})^2 \|\mathbf{x} - \mathbf{z}^{(t+1)}\|^2. \tag{36}$$

Next, by the stopping condition in Algorithm 3, the definition of $\mathbf{x}^{(t+1)}$, and strong convexity of the subproblem, we obtain:

$$\begin{aligned} P(\mathbf{x}^{(t+1)}; r) &\le P(\mathbf{x}^{(t+1)}; \mathbf{y}^{(t)}, r) + \frac{1}{2\eta^{(t)}}\|\mathbf{x}^{(t+1)} - \mathbf{y}^{(t)}\|^2 \\ &\le P(\hat{\mathbf{x}}; \mathbf{y}^{(t)}, r) + \frac{1}{2\eta^{(t)}}\|\hat{\mathbf{x}} - \mathbf{y}^{(t)}\|^2 - \frac{1}{2\eta^{(t)}}\|\hat{\mathbf{x}} - \mathbf{x}^{(t+1)}\|^2. \end{aligned}$$

Using convexity of $P(\cdot; \mathbf{y}^{(t)}, r)$, (35), and (36), we get

$$\begin{aligned} P(\mathbf{x}^{(t+1)}; r) \le\ & (1 - a^{(t)}) P(\mathbf{x}^{(t)}; \mathbf{y}^{(t)}, r) + a^{(t)} P(\mathbf{x}; \mathbf{y}^{(t)}, r) \\ & + \frac{(a^{(t)})^2}{2\eta^{(t)}}\|\mathbf{x} - \mathbf{z}^{(t)}\|^2 - \frac{(a^{(t)})^2}{2\eta^{(t)}}\|\mathbf{x} - \mathbf{z}^{(t+1)}\|^2 \end{aligned}$$

$$\leq (1-a^{(t)})P(\mathbf{x}^{(t)};r) + a^{(t)}P(\mathbf{x};r) + \frac{(a^{(t)})^2}{2\eta^{(t)}}\|\mathbf{x}-\mathbf{z}^{(t)}\|^2 - \frac{(a^{(t)})^2}{2\eta^{(t)}}\|\mathbf{x}-\mathbf{z}^{(t+1)}\|^2,$$

where the last inequality uses (31). By adding $P(\mathbf{x};r)$ to both sides and rearranging the terms, we obtain

$$\frac{\eta^{(t)}}{(a^{(t)})^2}\Big[P(\mathbf{x}^{(t+1)};r) - P(\mathbf{x};r)\Big] \leq \frac{\eta^{(t)}(1-a^{(t)})}{(a^{(t)})^2}\Big[P(\mathbf{x}^{(t)};r) - P(\mathbf{x};r)\Big] + \frac{1}{2}\|\mathbf{x}-\mathbf{z}^{(t)}\|^2 - \frac{1}{2}\|\mathbf{x}-\mathbf{z}^{(t+1)}\|^2.$$

In addition, by construction (Algorithm 3, Line 4), we have:

$$\begin{aligned}\frac{\eta^{(t)}}{(a^{(t)})^2}\Big[P(\mathbf{x}^{(t+1)};r) - P(\mathbf{x};r)\Big] \leq{}& \frac{\eta^{(t-1)}}{(a^{(t-1)})^2}\Big[P(\mathbf{x}^{(t)};r) - P(\mathbf{x};r)\Big] \\ &+ \frac{1}{2}\|\mathbf{x}-\mathbf{z}^{(t)}\|^2 - \frac{1}{2}\|\mathbf{x}-\mathbf{z}^{(t+1)}\|^2.\end{aligned}$$

Summing the inequality above over $t = 0,\ldots,T-1$ (with $a^{(-1)} = 1$, $\eta^{(-1)} = \eta^{(\mathrm{ini})}$, and $\mathbf{z}^{(0)} = \mathbf{x}^{(0)}$), we get:

$$P(\mathbf{x}^{(t)};r) - P(\mathbf{x};r) \leq \frac{(a^{(t-1)})^2}{\eta^{(t-1)}}\left(\eta^{(\mathrm{ini})}\Big[P(\mathbf{x}^{(0)};r) - P(\mathbf{x};r)\Big] + \frac{1}{2}\|\mathbf{x}-\mathbf{x}^{(0)}\|^2\right). \quad (37)$$

Since Algorithm 3 is terminated if $\eta^{(t)} \leq \frac{1}{L}$, it must hold that $\eta^{(t)} \geq \beta_{\mathrm{dec}}/L$ if Algorithm 3 runs for more than one iteration. If Algorithm 3 is terminated after one iteration, we have $\eta^{(t)} = \min\{\beta_{\mathrm{inc}}\,\eta^{(t-1)}, \eta^{(\mathrm{ini})}\}$ (Algorithm 4, Line 3). Combining these two cases, we have:

$$\eta^{(t)} \geq \min\left\{\eta^{(\mathrm{ini})}, \frac{\beta_{\mathrm{dec}}}{L}\right\} \qquad \text{for } t = -1, 0, 1, \ldots$$

Also, the recurrence relation $\frac{\sqrt{\eta^{(t)}}}{a^{(t)}} = \frac{\sqrt{\eta^{(t)}}}{2} + \frac{1}{2}\sqrt{\eta^{(t)} + \frac{4\eta^{(t-1)}}{(a^{(t-1)})^2}} \geq \frac{\sqrt{\eta^{(t)}}}{2} + \frac{\sqrt{\eta^{(t-1)}}}{a^{(t-1)}}$ leads to:

$$\frac{(a^{(t-1)})^2}{\eta^{(t-1)}} \leq \frac{4}{(t+1)^2}\max\left\{\frac{1}{\eta^{(\mathrm{ini})}}, \frac{L}{\beta_{\mathrm{dec}}}\right\}.$$

Substituting this into (37) completes the proof of (34). Finally, the total number of prox-linear subproblems solved follows by the same argument as in Lemma 4 of Nesterov (2013). ■

We next show Algorithm 4 can be used as an `fom` in Algorithm 2.

Proposition 20 *Consider α, B, ϵ, r, $\eta^{(ini)}$, β_{dec} and $\mathbf{x}^{(0)} \in \mathcal{X}$ such that $0 < \alpha < B < 1$, $\epsilon > 0$, $r < f^*$, $\eta^{(ini)} > 0$, $\beta_{dec} \in (0,1)$, and $P(\mathbf{x}^{(0)};r) > \epsilon$. The accelerated prox-linear method described in Algorithm 4 satisfies Assumption 2 with*

$$n^{\mathtt{fom}} := \max\left\{\sqrt{\frac{2\max\left\{\frac{1}{\eta^{(ini)}}, \frac{L}{\beta_{dec}}\right\}G^{2/d}}{\alpha(B-\alpha)\epsilon^{1-2/d}}}, \sqrt{\frac{4\max\left\{1, \frac{\eta^{(ini)}L}{\beta_{dec}}\right\}}{B-\alpha}}\right\} - 1. \quad (38)$$

Proof Suppose $\mathbf{x}^{(0)}$ does not satisfy (5), i.e. $\alpha P(\mathbf{x}^{(0)}; r) > f^* - r$. We only need to show $P(\bar{\mathbf{x}}^{(t)}; r) \le BP(\mathbf{x}^{(0)}; r)$ for $t \le n^{\texttt{fom}}$. Let $\mathbf{x}^* = \text{Proj}_{\mathcal{X}^*}(\mathbf{x}^{(0)})$, and consider the approximation bound from Theorem 19 for any $t \ge 1$ and $\mathbf{x} = \mathbf{x}^*$:

$$\begin{aligned} P(\mathbf{x}^{(t)}; r) &\le P(\mathbf{x}^*; r) + \frac{4C_1}{(t+1)^2}\left[P(\mathbf{x}^{(0)}; r) - P(\mathbf{x}^*; r)\right] + \frac{2C_2 G^{2/d} \cdot P(\mathbf{x}^{(0)}; f^*)^{2/d}}{(t+1)^2}, \\ &\le f^* - r + \frac{4C_1}{(t+1)^2}\left[P(\mathbf{x}^{(0)}; r) - (f^* - r)\right] + \frac{2C_2 G^{2/d} \cdot P(\mathbf{x}^{(0)}; r)^{2/d}}{(t+1)^2}, \end{aligned} \tag{39}$$

where for simplifying the equations, we defined

$$C_1 := \max\left\{1, \frac{\eta^{(\text{ini})} L}{\beta_{\text{dec}}}\right\}, \qquad C_2 := \max\left\{\frac{1}{\eta^{(\text{ini})}}, \frac{L}{\beta_{\text{dec}}}\right\}.$$

The first inequality follows from the error bound condition (2) and the last from (23) and the fact that $P(\mathbf{x}^{(0)}; f^*) \le P(\mathbf{x}^{(0)}; r)$ which holds because $r < f^*$. Note that $P(\mathbf{x}^{(0)}; r) - (f^* - r) \ge (1/\alpha - 1)(f^* - r) \ge 0$. Now, define

$$t := \max\left\{\sqrt{\frac{2\, C_2\, G^{2/d}}{\alpha\,(B\, -\, \alpha)\, P(\mathbf{x}^{(0)}; r)^{1-2/d}}}, \sqrt{\frac{4\, C_1}{B\, -\, \alpha}}\right\} - 1 \ge 0. \tag{40}$$

Plugging this t into (39) gives:

$$\begin{aligned} P(\mathbf{x}^{(t)}; r) \le& f^* - r + (B - \alpha)[P(\mathbf{x}^{(0)}; r) - (f^* - r)] + \alpha(B - \alpha)P(\mathbf{x}^{(0)}; r) \\ =& (1 - B + \alpha)(f^* - r) + (B - \alpha)P(\mathbf{x}^{(0)}; r) + \alpha(B - \alpha)P(\mathbf{x}^{(0)}; r) \\ \le& (1 - B + \alpha)\alpha P(\mathbf{x}^{(0)}; r) + (B - \alpha)P(\mathbf{x}^{(0)}; r) + \alpha(B - \alpha)P(\mathbf{x}^{(0)}; r) \\ =& BP(\mathbf{x}^{(0)}; r), \end{aligned}$$

where the second inequality holds by the assumption that $\alpha P(\mathbf{x}^{(0)}; r) > f^* - r$. Our proof is complete because $t \le n^{\texttt{fom}}$ which follows from our assumption that $P(\mathbf{x}^{(0)}; r) > \epsilon$. ■

Remark 21 *When $d < 2$ and ϵ is sufficiently small, $n^{\texttt{fom}}$ drops to $\mathcal{O}(1)$. That is because the condition $P(\mathbf{x}^{(t)}; r) \le BP(\mathbf{x}^{(0)}; r)$ is satisfied after a constant number of prox-linear steps - as long as $P(\mathbf{x}^{(0)}; r) > \epsilon$. This phenomenon is due to the error bound condition: when $d < 2$, the landscape of $P(\mathbf{x}; r)$ becomes sharp enough near the optimal solutions set, and the first term in the maximum in (40) decays rapidly as a function of $P(\mathbf{x}^{(0)}; r)$. As a result, the overall complexity bound in (14) remains valid and logarithmic in $1/\epsilon$.*

Corollary 22 *Suppose α, B, γ, ϵ, and r_{ini} are such that $\alpha < B < 1$, $\gamma > 1$, $\epsilon > 0$, and $r_{ini} < f^*$. Consider $\tilde{K}$ defined in (11) for $r = r_{ini}$ and let $\eta^{(ini)} > 0$, $L > 0$, and $\beta_{dec} \in (0, 1)$. Suppose Algorithm 2 is executed with the accelerated prox-linear method described in Algorithm 4 as $\texttt{fom}_k$ instances for $k = 0, 1, \ldots, \tilde{K}$. Then, the algorithm finds an ϵ-optimal and ϵ-feasible solution in at most*

$$(m+1)(\tilde{K}+1) \cdot \mathcal{O}\left(\max\left\{\sqrt{\frac{\max\left\{\frac{1}{\eta^{(ini)}}, \frac{L}{\beta_{dec}}\right\} G^{2/d}}{\alpha(B - \alpha)\epsilon^{1-2/d}}}, \sqrt{\frac{\max\left\{1, \frac{\eta^{(ini)} L}{\beta_{dec}}\right\}}{B - \alpha}}\right\}\right) \cdot \tilde{D},$$

$$= \mathcal{O}\left(\log^2\left(\frac{1}{\epsilon}\right)\max\left\{\frac{1}{\epsilon^{1/2-1/d}}, 1\right\}\right),$$

gradient and function evaluations, where $\tilde{D}$ is defined in (15).

Proof The proof of this corollary is similar to the proof of Corollary 18 except that (38) is used as $n^{\texttt{fom}}$ and follows from the number of prox-linear subproblems to solve, which equals $\mathcal{O}(n^{\texttt{fom}})$ (see the second conclusion of Theorem 19). ■

5. Numerical Experiments

We assess the numerical effectiveness of Algorithm 2 on two types of binary classification problems: those with fairness constraints and those with Type II error constraints. This evaluation serves as a benchmark, comparing our proposed method against existing approaches in the literature. For the fairness-constrained setting, we consider both (1) a non-smooth formulation, using the standard subgradient algorithm as the `fom` in Algorithm 2, and (2) a smooth formulation, using Algorithm 4 as the `fom`. For the Type II error-constrained problems, we focus on the non-smooth formulation and use the subgradient method as the `fom`, with the goal of evaluating performance on larger-scale instances.

5.1 Non-smooth Classification Problem With Fairness Constraints

Consider a set of n data points $\mathcal{D} = \{(\mathbf{a}_i, b_i)\}_{i=1}^n$ where $\mathbf{a}_i \in \mathbb{R}^p$ denotes a feature vector, and $b_i \in \{1, -1\}$ represents the class label for $i = 1, 2, \ldots, n$. Let $\mathcal{D}_M \subset \mathbb{R}^p$ and $\mathcal{D}_F \subset \mathbb{R}^p$ be two different sensitive groups of instances. We want to find a linear classifier $\mathbf{x} \in \mathbb{R}^p$ that not only predicts the labels well (i.e., minimizes a loss function) but also treats each sensitive instance from $\mathcal{D}_M$ and $\mathcal{D}_F$ fairly. Such classification problems are suitable for applications such as loan approval or hiring decisions. For example, in the context of loan approval, we can ensure that loans are equally provided for different applicants independent of their group memberships (e.g., home-ownership or their gender). A correct classifier $\mathbf{x}$ satisfies $b_i\mathbf{a}_i^\top\mathbf{x} > 0$ for all i. One can train such a classifier by solving the following optimization problems that minimizes a non-increasing convex loss function $\phi(\cdot)$ subject to fairness constraints:

$$\min_{\mathbf{x}} \quad \frac{1}{n}\sum_{i=1}^n \phi(-b_i\mathbf{a}_i^\top\mathbf{x}) \tag{41}$$

$$\text{s.t.} \quad \frac{1}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F}\sigma(\mathbf{a}^\top\mathbf{x}) \geq \frac{\kappa}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M}\sigma(\mathbf{a}^\top\mathbf{x}), \tag{42}$$

$$\frac{1}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M}\sigma(\mathbf{a}^\top\mathbf{x}) \geq \frac{\kappa}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F}\sigma(\mathbf{a}^\top\mathbf{x}), \tag{43}$$

where $\kappa \in (0, 1]$ is a constant, n_M and n_F are respectively the number of instances in D_M and D_F, and finally $\sigma(\mathbf{a}^\top\mathbf{x}) := \max\{0, \min\{1, \{0.5 + \mathbf{a}^\top\mathbf{x}\}\} \in [0, 1]$ is the probability of predicting $\mathbf{a}$ in class $+1$. The constraint (42) guarantees that the percentage of the instances in $\mathcal{D}_F$ that are predicted in class $+1$ is at least a $\kappa > 0$ fraction of that in $\mathcal{D}_M$. The constraint (43) has a similar interpretation for instances in $\mathcal{D}_M$. Choosing an appropriate κ

ensures that the obtained classifier is fair to both $\mathcal{D}_M$ and $\mathcal{D}_F$ groups. An analogous model was considered in Goh et al. (2016).

Constraints (42) and (43) are non-convex, so we approximate this problem by a convex optimization problem following the approach described in Lin et al. (2020). In particular, we reformulate the first constraint as $\frac{\kappa}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M}\sigma(\mathbf{a}^\top\mathbf{x})+\frac{1}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F}\sigma(-\mathbf{a}^\top\mathbf{x})\leq 1$ by using $\sigma(\mathbf{a}^\top\mathbf{x})=1-\sigma(-\mathbf{a}^\top\mathbf{x})$. In addition, the function $\sigma(\mathbf{a}^\top\mathbf{x})$ can be approximated by $(0.5+\mathbf{a}^\top\mathbf{x})_+=\max\{0,0.5+\mathbf{a}^\top\mathbf{x}\}$. Hence, we can rewrite the constraint (42) as a convex constraint: $\frac{\kappa}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M}(\mathbf{a}^\top\mathbf{x}+0.5)_++\frac{1}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F}(-\mathbf{a}^\top\mathbf{x}+0.5)_+\leq 1$. Applying a similar convex approximation to (43), we obtain the following convex reformulation of (41):

$$\begin{aligned}\min_{\mathbf{x}}\quad & \frac{1}{n}\sum_{i=1}^{n}\phi(-b_i\mathbf{a}_i^\top\mathbf{x}) \\ \text{s.t.}\quad & \frac{\kappa}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M}(\mathbf{a}^\top\mathbf{x}+0.5)_++\frac{1}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F}(-\mathbf{a}^\top\mathbf{x}+0.5)_+\leq 1, \\ & \frac{\kappa}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F}(\mathbf{a}^\top\mathbf{x}+0.5)_++\frac{1}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M}(-\mathbf{a}^\top\mathbf{x}+0.5)_+\leq 1.\end{aligned}\tag{44}$$

The above reformulation contains piece-wise linear functions (non-smooth) in its objective and constraints. Hence, this problem satisfies the error bound condition (2) with $d=1$ and an unknown value of G. We implemented RLS (Algorithm 2) to solve (44) using subgradient method in each $\texttt{fom}_k$ instance to solve the subproblem $\min_{\mathbf{x}\in\mathcal{X}}P(\mathbf{x}_k,r_k)$. We compare the performance of our method with three parameter-free but non-adaptive benchmarks: (1) the feasible level-set (FLS) method described in Lin et al. (2018b); (2) the stochastic subgradient method by Yu, Neely, and Wei (YNW) in Yu et al. (2017); and (3) the switching gradient (SWG) method in Bayandina et al. (2018) which is based on a mirror descent algorithm and a switching sub-gradient scheme [5]. We choose these benchmarks as they are the most representative first-order methods that can solve "non-smooth" convex "constrained" optimization problems. The FLS is a level-set method that guarantees the feasibility of the solutions generated at each iteration, but does not exploit the error bound condition as our method does. Hence, it has a higher complexity bound for the class of problems that satisfy the error bound condition (2). YNW is a primal-dual method with dual variables updated implicitly, while SWG is a pure primal method. We used the number of equivalent data passes performed by each algorithm as a measure of complexity.

Instances: To check the performance of the above mentioned methods, we considered three classification problems using (i) "LoanStats" data set from lending club[6] which is a platform that allows individuals lend to other individuals; (ii) "COMPAS" data set from ProPublica[7]; and (iii) "German" data set from UCI Machine Learning Repository[8].

The LoanStats data set contains information of 128,375 loans issued on the loan club platform in the fourth quarter of 2018. The goal in this application is to predict whether a

5. We do not benchmark against adaptive methods for constrained convex optimization as they require knowledge of unknown parameters. Recall that an adaptive algorithm can harness the EBC when it holds, leading to improved convergence results.
6. `https://www.lendingclub.com/info/statistics.action`
7. `https://github.com/propublica/compas-analysis`
8. `https://archive.ics.uci.edu/ml/datasets/statlog+(german+credit+data)`

loan request will be approved or rejected. After creating dummy variables, each loan in this data set is represented by a vector of 250 features. We randomly partitioned this dataset into two sets: one of $63,890$ examples to construct the objective function and another of $64,485$ examples to construct the constraints. The set D_M in this application denotes the set of instances with "home-ownership = Mortgage". All the other instances are considered in the set D_F. The fairness constraints in this application guarantee that customers with mortgages will not be disproportionately affected in receiving new loans.

The COMPAS dataset includes criminal history, jail and prison time, demographics, and other factors for 6,172 instances. The goal of this application is to predict whether a person will be rearrested within two years after their initial arrest. This information, for example, helps judges make bail decisions by predicting the criminal recidivism of defendants. We used $4,115$ examples from this dataset to formulate the objective function, and the remaining examples to formulate the constraints. We chose sets D_M and D_F to respectively be the sets of male and female instances to ensure fair treatment among different genders.

The German credit dataset describes financial details of customers and is used to determine whether the customer should be granted credit (i.e., is a good customer) or not (i.e., is a bad customer). This dataset contains 1,000 instances and 50 variables. We used 667 of the examples to formulate the objective function and the remaining to formulate the constraints. Similar to COMPAS data, the sets D_M and D_F respectively denote male and female examples.

In all above applications, we chose $\kappa = 0.9$, and $\phi(z) = (1 - z)_+$. We terminated RLS and the other three benchmarks when the number of data passes reached 20,000. However, we ran the two level-set methods, i.e., RLS and FLS, longer for 100,000 data passes to find the optimal value f^*. In particular, we selected the smallest objective value among all feasible solutions as a close approximation of f^*. We used this value to compute and plot $P(x; f^*)$.

Algorithmic configurations: We next describe how we selected each of the parameters used in RLS, FLS, YNW, and SWG.

- RLS: We choose a value for ϵ from $\{1, 0.1, 0.01, 0.001\}$, for α from $\{0.4, 0.5, 0.7, 0.9\}$ and for B from $\{0.5, 0.9, 0.95, 0.99\}$ to minimize the value of $P(\mathbf{x}; f^*)$ at termination (we only consider the combination with $\alpha \leq B$). Since the optimal value of (44) is positive, we set $r_{\text{ini}} = 0$ to satisfy the condition $r_{\text{ini}} < f^*$. Although the vector of all zeros could be used as an initial solution $\mathbf{x}_{\text{ini}}$, we used a different value in our algorithm. In particular, the complexity bound of RLS shows that the algorithm requires fewer iterations if $g(\mathbf{x}_{\text{ini}}) < 0$ and is as small as possible. Hence, we followed a heuristic to obtain a better quality solution. We applied the subgradient algorithm to minimize the function $g(\mathbf{x})$ and used the returned solution after 40 iterations. Finding such a solution accounted for less than 1% of our total-run time. Finally, we used $K = \tilde{K}$ where $\tilde{K}$ is computed as in (11) for $r = r_{\text{ini}}$.

- FLS: The FLS algorithm described in Lin et al. (2018b) requires two input parameters: $r_{\text{ini}} > f^*$ and $\alpha > 1$. To obtain $r_{\text{ini}} > f^*$, we first followed the same step as in RLS to obtain a feasible solution $\mathbf{x}_{\text{ini}}$ such that $g(\mathbf{x}_{\text{ini}}) < 0$ and then set $r_{\text{ini}} = f(\mathbf{x}_{\text{ini}})$. Since $\mathbf{x}_{\text{ini}}$ is feasible, using Lemma 1 we can guarantee that $r_{\text{ini}} > f^*$. We set $D = 5\|\mathbf{x}_{\text{ini}}\|$ and chose α from $\{100, 200, 500\}$ that produces the smallest $P(\mathbf{x}; f^*)$ at termination.

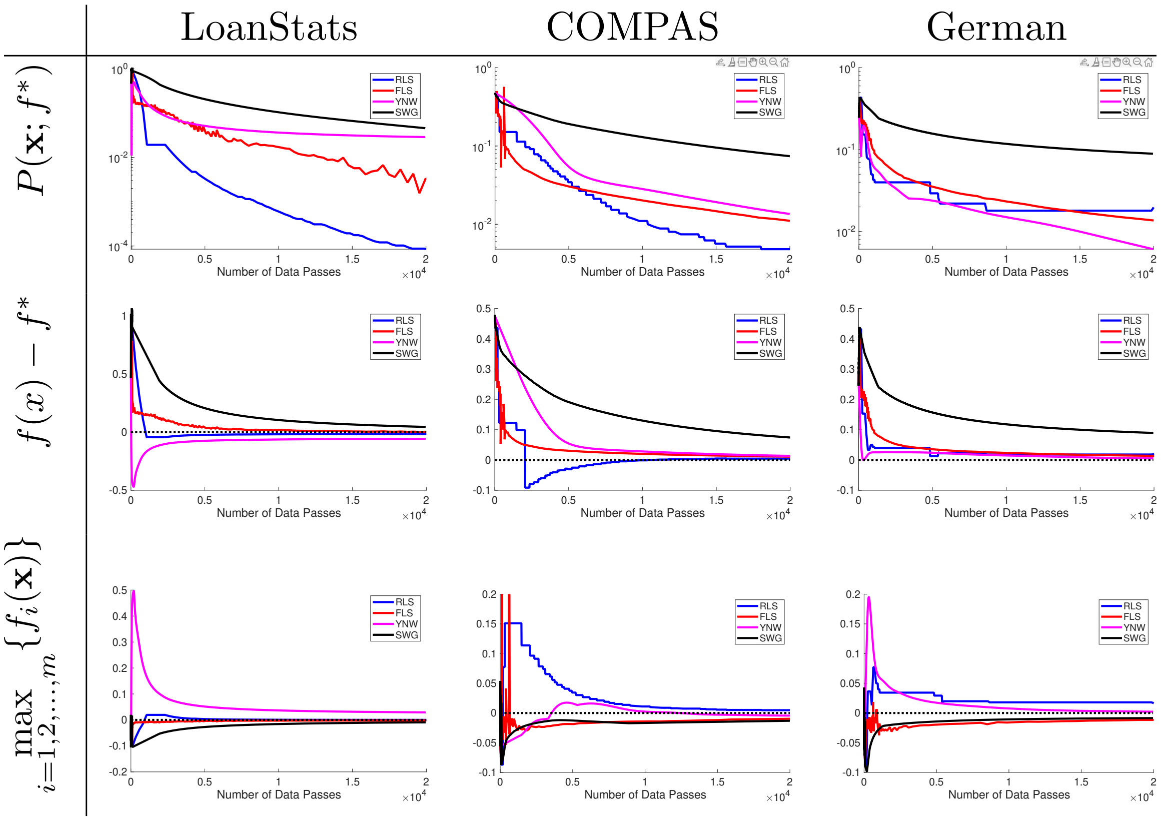

Figure 1: Performance of RLS, FLS, YNW, and SWG for solving non-smooth binary classification problems with fairness constraints.

- YNW: We followed the guidance in Yu et al. (2017) to setup YNW. Specifically, we chose the control parameters V and α as $V = \sqrt{T}$ and $\alpha = T$, respectively, as a function of the total number of iterations T, where V is the weight of the gradient of the objective function and α is the weight of the proximal term in the updating equation of $\mathbf{x}$ in YNW. We chose the number of iterations T such that the total number of data passes is $20,000$. We used the all-zero vector as an initial solution to YNW.

- SWG: The only input of SWG algorithm is the precision ϵ. We selected ϵ from $\{0.01, 0.001, 0.0001\}$ to minimize the value of $P(x; f^*)$ achieved by SWG at termination. The initial solution is chosen as a feasible solution $\mathbf{x}_{\text{ini}}$ found by the same step as in RLS and FLS. Of course, for a fair comparison, the number of data passes that RLS, FLS and SWG spend in searching for the initial solution is included when comparing the performance of these algorithms.

Results: Figure 1 displays the performance of RLS, FLS, YNW and SWG as a function of data passes. In particular, the x-axis in this figure shows the number of data passes each algorithm performed while the y-axis represents the feasibility and optimality of solutions returned by each method. More specifically, the y-axis in the first row of this plot represents $P(\mathbf{x}; f^*)$. As explained before, this quantity is zero when the solution $\mathbf{x}$ is both feasible and optimal and positive otherwise. The speed at which $P(\mathbf{x}; f^*)$ converges to zero indicates how fast the solution $\mathbf{x}$ becomes feasible and optimal at each iteration. The y-axis in the second row represents $f(\mathbf{x}) - f^*$ that measures the optimality gap at $\mathbf{x}$ and in the third row, $g(\mathbf{x}) = \max_{i=1,\ldots,m}\{f_i(\mathbf{x})\}$ checks whether the solution $\mathbf{x}$ is feasible or not (if $g(\mathbf{x}) < 0$ then $\mathbf{x}$ is feasible).

We observe that RLS achieves the fastest reduction in the function $P(\mathbf{x}; f^*)$ across two out of the three datasets. Recall that $P(\mathbf{x}; f^*)$ equals the maximum of infeasibility and suboptimality of $\mathbf{x}$. Our computational results reveal that our algorithm compares favorably with the benchmarks on these two datasets. However, on the German credit dataset, RLS doesn't exhibit the best performance, possibly due to the problem satisfying the error bound condition ($d = 1$) with a large G, causing our method to have higher complexity than FLS and YNW. It can also be seen that RLS and YNW may generate infeasible solutions before convergence, while FLS and SWG maintain feasibility at each iteration.

5.2 Smooth Classification Problem With Fairness Constraints

In this section, we revisit the binary classification problem with fairness constraints outlined in (41)-(43). However, this time, we employ a different approximation for the function $\sigma(\mathbf{a}^\top\mathbf{x})$ by using $\log_4(1 + \exp(\mathbf{a}^\top\mathbf{x}))$, and adopt $\phi(z) = \ln(1 + \exp(-z))$. The choice of the base-four logarithm ensures that $\log_4(1 + \exp(\mathbf{a}^\top\mathbf{x})) \geq 0.5$ when $\mathbf{a}^\top\mathbf{x} \geq 0$, aligning with our prior selection of $\sigma(\mathbf{a}^\top\mathbf{x}) = (0.5 + \mathbf{a}^\top\mathbf{x})_+$.

Therefore, we can formulate our classification problem with fairness constraints as

$$\begin{aligned}
\min_{\mathbf{x}} \quad & \frac{1}{n}\sum_{i=1}^{n} \ln(1 + \exp(-b_i\mathbf{a}_i^\top\mathbf{x})) \qquad (45)\\
\text{s.t.} \quad & \frac{\kappa}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M} \log_4(1 + \exp(\mathbf{a}^\top\mathbf{x})) + \frac{1}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F} \log_4(1 + \exp(-\mathbf{a}^\top\mathbf{x})) \leq 1,\\
& \frac{\kappa}{n_F}\sum_{\mathbf{a}\in\mathcal{D}_F} \log_4(1 + \exp(\mathbf{a}^\top\mathbf{x})) + \frac{1}{n_M}\sum_{\mathbf{a}\in\mathcal{D}_M} \log_4(1 + \exp(-\mathbf{a}^\top\mathbf{x})) \leq 1.
\end{aligned}$$

The convex optimization problem presented above features differentiable objectives and constraints with Lipschitz continuous gradients. While the satisfaction of the error bound condition (2) is generally uncertain, our RLS method remains applicable and automatically leverages the error bound condition for any values of d and G, when (2) is valid, thanks to its adaptive nature.

Notably, even in the absence of a strongly convex regularization term like $\frac{1}{2}\|\mathbf{x}\|_2^2$ in the objective and constraints, when the dataset size significantly exceeds the dimension of $\mathbf{a}$, it is plausible that both the objective and constraints exhibit μ-strong convexity. This implies satisfaction of (2) with $d = 2$ and $G = \mu^{-1}$. In such instances, our method boasts a theoretical complexity of $\tilde{\mathcal{O}}(1/\sqrt{\epsilon})$ without necessitating knowledge of d and μ.

In contrast, existing non-adaptive methods require prior knowledge of d and μ to achieve a similar complexity. Without this information, they are limited to a complexity of $\tilde{\mathcal{O}}(1/\epsilon)$.

We employed the Restarting Level Set (RLS) algorithm, detailed in Algorithm 2, to solve the convex and smooth problem specified in (45). In each instance $\texttt{fom}_k$, we utilized the accelerated proximal linear method outlined in Algorithm 4 to solve the subproblem $\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}, r_k)$, where the subproblem in Step 6 in Algorithm 3 is solved by applying `quadprog` function in MATLAB to solve its dual problem, a convex quadratic program in $\mathbb{R}^3$, and recovering the primal solution.

Our method's performance is benchmarked against two non-adaptive approaches: (1) Feasible Level Set (FLS) from Lin et al. (2018b), introduced previously; (2) Constraint Extrapolation (ConEx) method from Boob et al. (2023), a single-loop primal-dual algorithm considered a variant of well-known primal-dual accelerated gradient methods for min-max optimization (Chambolle and Pock, 2011; Hamedani and Aybat, 2021). We selected these benchmarks as they represent prominent first-order methods capable of solving "smooth" convex "constrained" optimization problems, achieving optimal complexity for both convex and strongly convex problems.

Instances: To evaluate these methods, we tackled three classification problems using the same datasets as in the non-smooth case. Across all applications, we set $\kappa = 0.9$. Termination criteria for RLS and the two benchmarks were based on reaching 80,000 data passes. However, we extended the runs for the two level-set methods, RLS and FLS, to 100,000 data passes to pinpoint the optimal value f^*. Specifically, we selected the smallest objective value among all feasible solutions as a close approximation of f^*, using this value to compute and plot $P(x; f^*)$.

Algorithmic configurations: The parameters for RLS and FLS methods were chosen by the same procedure as in the non-smooth case, except for RLS, where we employed the accelerated proximal linear method with line search outlined in Algorithm 4 to solve the subproblem $\min_{\mathbf{x}\in\mathcal{X}} P(\mathbf{x}, r_k)$. We set $\eta^{(\text{ini})} = 1$, $\beta_{\text{dec}} = 0.8$ and $\beta_{\text{inc}} = 1.2$ in Algorithm 4.

For the ConEx method, without assuming strong convexity, we adhered to the parameter settings recommended in Theorem 2 of Boob et al. (2023). Specifically, we chose $\gamma_t = 1$, $\theta_t = 1$, $\tau_t = \tau$, and $\eta_t = \eta$ for each iteration t in ConEx. The values of τ and η were selected from the set $\{1, 10, 100\}$ to minimize the value of $P(\mathbf{x}; f^*)$ at termination.

Results: Figure 2 illustrates the performance of RLS, FLS, and ConEx over varying data passes. The axes' meanings remain consistent with Figure 1.

Notably, RLS demonstrates the swiftest reduction in the function $P(\mathbf{x}; f^*)$ to a small scale (e.g. 10^{-5}) on the three datasets. FLS stands out as the sole algorithm maintaining feasibility at each iteration, while ConEx diminishes the infeasibility faster than RLS on the COMPAS dataset and the German credit dataset. RLS, despite not being the fastest in reducing infeasibility, promptly achieves a high-quality solution in the sense of having a small $P(\mathbf{x}; f^*)$, showcasing its favorable comparison with existing approaches in the literature.

5.3 Non-smooth Neyman–Pearson Classification with Type II Error Control

In this section, we evaluate the performance of our RLS method on a Neyman-Pearson classification problem, where the goal is to minimize the Type I error (i.e., the misclassification of negative samples as positive) subject to a Type II error constraint (i.e., ensuring that the misclassification rate for positive samples remains below a given threshold). In contrast to

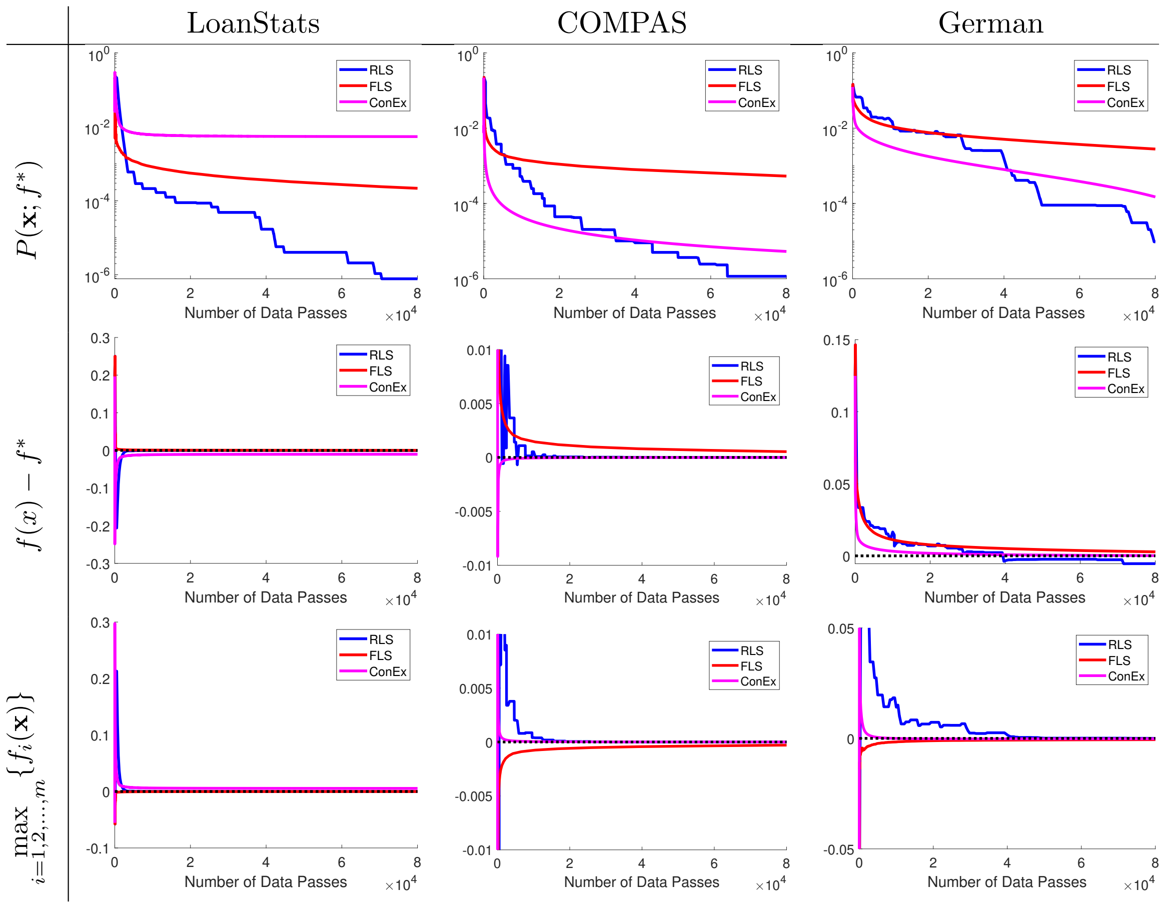

Figure 2: Performance of RLS, FLS, and ConEx for solving smooth binary classification problems with fairness constraints.

our previous experiments on fairness constraints, this instance emphasizes the control of Type II error and involves larger problem instances.

Consider a training set of n data points $\mathcal{D} = \{(\mathbf{a}_i, b_i)\}_{i=1}^n$, where $\mathbf{a}_i \in \mathbb{R}^p$ denotes a feature vector and $b_i \in \{1, -1\}$ is the corresponding class label. Let $\mathcal{D}_+ = \{(\mathbf{a}, b) \in \mathcal{D} \mid b = 1\}$ and $\mathcal{D}_- = \{(\mathbf{a}, b) \in \mathcal{D} \mid b = -1\}$. The Neyman–Pearson linear classification problem aims to find a vector $\mathbf{x} \in \mathbb{R}^p$ that minimizes the empirical Type I error (computed over $\mathcal{D}_-$) while ensuring that the empirical Type II error (computed over $\mathcal{D}_+$) does not exceed a specified threshold κ. By approximating both error types with a non-increasing convex surrogate loss function $\phi(\cdot)$, the problem is formulated as

$$\min_{\|\mathbf{x}\|_2 \le R} \left\{ \frac{1}{n_-} \sum_{(\mathbf{a},b)\in\mathcal{D}_-} \phi(-\mathbf{a}^\top \mathbf{x}) \right\} \quad \text{s.t.} \quad \frac{1}{n_+} \sum_{(\mathbf{a},b)\in\mathcal{D}_+} \phi(\mathbf{a}^\top \mathbf{x}) - \kappa \le 0, \tag{46}$$

where $R > 0$ is a regularization parameter, $n_+ = |\mathcal{D}_+|$, $n_- = |\mathcal{D}_-|$, and the constraint enforces a bound on the Type II error. Problem (46) is an instance of the general constrained convex optimization model discussed in Section 4.1 (with $m = 1$ and $\mathcal{X} = \{\mathbf{x} : \|\mathbf{x}\|_2 \leq R\}$).

We consider three large-scale datasets: (i) "gisette" (Guyon et al., 2004), (ii) "rcv1.binary" (Lewis et al., 2004), and (iii) "real-sim"[9]. These datasets were obtained from the LIBSVM library[10]. The gisette dataset involves handwritten digit recognition and comprises $6,000$ data points with $5,000$ features; the rcv1.binary dataset, used for text categorization, has $20,242$ training examples with $47,236$ features; and the real-sim dataset consists of $72,309$ data points with $20,958$ features. Each feature vector is normalized to have unit Euclidean norm. For all instances, we set $\kappa = 0.5$, choose $\phi(z) = (1 - z)_+$, and set $R = 1$ for gisette and $R = 5$ for rcv1.binary and real-sim.

Since all these instances are non-smooth, we implemented Algorithm 2 using a subgradient method in each $\texttt{fom}_k$ instance. Similar to Section 5.1, we compare RLS with the FLS, the YNW method, and the switching subgradient method (SWG) on these larger instances. In our experiments, the parameters of all methods are chosen by the same procedure as in Section 5.1. All methods are terminated when the total number of data passes reaches 40,000, while RLS is run for 100,000 data passes to obtain an accurate approximation of the optimal value f^*. Specifically, the smallest objective value among all feasible solutions produced by RLS is selected as an approximation of f^* and is used to compute and plot the residual function $P(\mathbf{x}; f^*)$.

Results: Figure 3 illustrates the performance of RLS, FLS, YNW, and SWG over varying data passes. The axes' meanings remain consistent with Figures 1 and 2. The plots illustrate that RLS achieves the fastest reduction of the metric $P(\mathbf{x}; f^*)$ compared to YNW and SWG across all datasets. Although FLS initially reduces $P(\mathbf{x}; f^*)$ more quickly when $P(\mathbf{x}; f^*) > 10^{-2}$, RLS outperforms FLS in the high-accuracy regime (i.e., when $P(\mathbf{x}; f^*) \leq 10^{-2}$), ultimately achieving significantly smaller values of $P(\mathbf{x}; f^*)$. Moreover, while YNW exhibits substantial optimality gaps on the rcv1.binary and real-sim datasets, both FLS and SWG maintain feasibility throughout the iterations; among these, FLS demonstrates a faster reduction of the optimality gap than SWG. These numerical results, obtained on larger-scale classification problems, confirm that RLS is highly effective in simultaneously reducing the optimality gap and constraint violation, thereby yielding an accurate and feasible solution.

6. Conclusion

We develop an adaptive level-set method that is both parameter-free and projection-free for constrained convex optimization, that is, it accelerates under the error bound condition and does not require knowledge of unknown parameters or challenging projections for its execution. This method finds an ϵ-optimal and ϵ-feasible solution by considering a sequence of level-set subproblems that are solved in parallel using standard subgradient oracles with simple updates. These oracles restart based on objective function progress and communicate information between them upon each restart. We show that the iteration complexity of our restarting level set method is only worse by a log-factor in both the smooth and non-smooth

9. `https://people.cs.umass.edu/~mccallum/code-data.html`
10. `https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

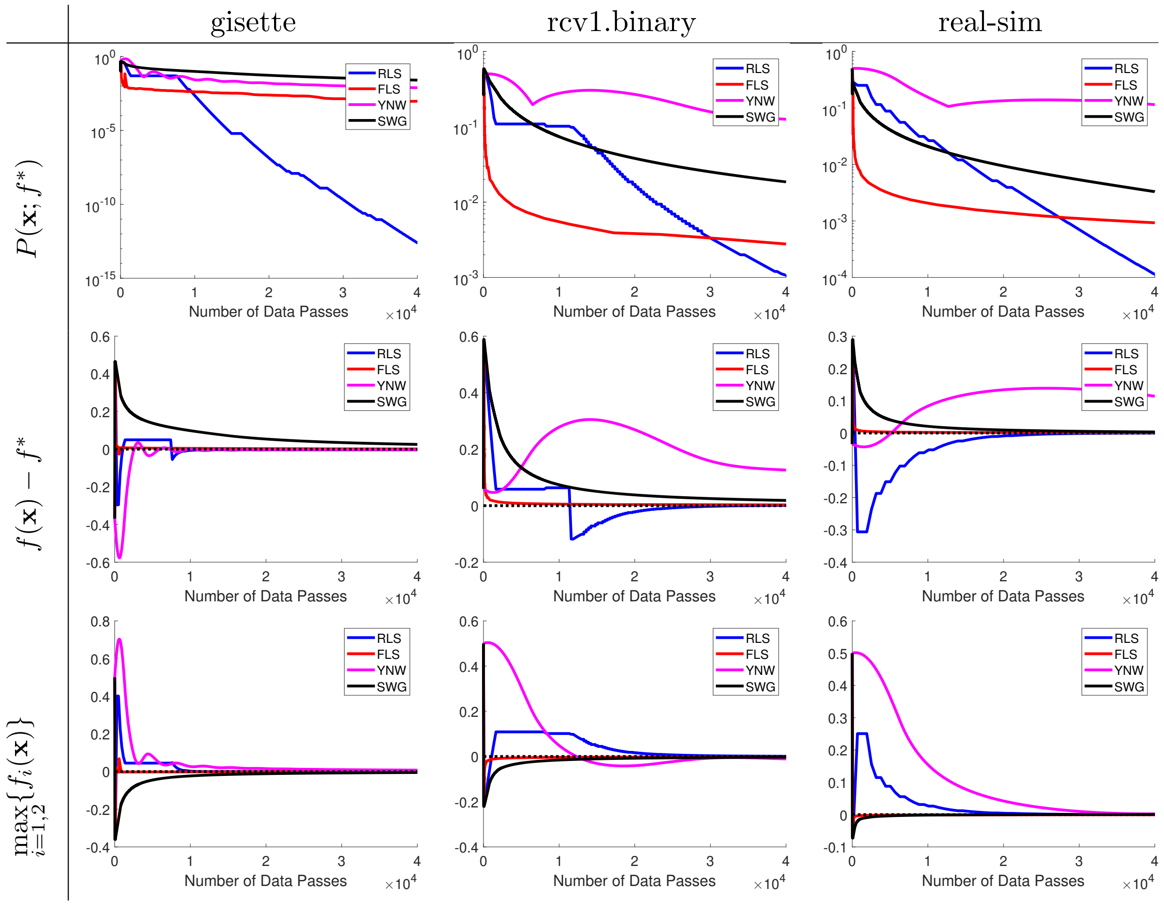

Figure 3: Performance of RLS, FLS, YNW, and SWG for solving non-smooth Neyman–Pearson classification problems with Type I error control on larger-scale datasets.

settings compared to existing accelerated methods for constrained convex optimization under the error bound condition, all of which rely on either unknown parameters or sophisticated oracles (e.g., involving difficult projection or exact line search). Numerical experiments show that the proposed method exhibits promising performance relative to benchmarks on constrained classification problems.

While slightly beyond the primary scope of this paper, we explore an extension of our analysis in Appendix C. In this extension, we showcase how our methodology can adapt to scenarios where linear equality constraints are integrated into (1). It's worth noting that in this extension, our adaptive algorithm is no longer parameter-free, as it requires knowledge of parameter M, an upper bound on the norm of sub-gradients. Despite this requirement, given that M is readily computable in many applications, we believe this extension could be of interest to certain readers.

Appendix A. An Example of Error Bound Condition with Exponent $d > 2$

To demonstrate that the error bound condition can hold with exponent $d > 2$, we present an illustrative example adapted from Li et al. (2015). Consider a constrained convex polynomial optimization problem in which the feasible region $\mathcal{X}$ is given by

$$\mathcal{X} := \{\mathbf{x} \in \mathbb{R}^n \mid h_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, \ell\},$$

where each h_i is a convex polynomial of degree at most q, and suppose that $\mathcal{X}$ is compact. Let each objective and constraint function f_i for $i = 0, \ldots, m$ also be a convex polynomial of degree at most q. The optimization problem (1) becomes

$$\min_{\mathbf{x} \in \mathcal{X}} \{f_0(\mathbf{x}) \mid f_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, m\}.$$

The corresponding optimal solution set is

$$\mathcal{X}^* := \{\mathbf{x} \in \mathcal{X} \mid \max\{f(\mathbf{x}) - f^*, g(\mathbf{x}), h(\mathbf{x})\} \leq 0\},$$

where $f(\mathbf{x}) := f_0(\mathbf{x})$, $g(\mathbf{x}) := \max_{i=1,\ldots,m} f_i(\mathbf{x})$, and $h(\mathbf{x}) := \max_{i=1,\ldots,\ell} h_i(\mathbf{x})$.

Under these assumptions, Corollary 3.8 in Li et al. (2015) implies that there exists a constant $c > 0$ such that for all $\mathbf{x} \in \mathcal{X}$,

$$\begin{aligned}\operatorname{dist}(\mathbf{x}, \mathcal{X}^*) &\leq c \left([f_0(\mathbf{x}) - f^*]_+ + \sum_{i=1}^{m} [f_i(\mathbf{x})]_+ + \sum_{i=1}^{\ell} [h_i(\mathbf{x})]_+ \right)^{\tau} \\ &\leq c(m+1)^{\tau} \left(\max\{f(\mathbf{x}) - f^*, g(\mathbf{x})\}\right)^{\tau},\end{aligned}$$

where $[\cdot]_+ := \max\{\cdot, 0\}$ and

$$\tau := \max\left\{ \frac{1}{(q+1)(3q)^{n+m+\ell}}, \frac{1}{q(6q-3)^{n+m+\ell}} \right\}.$$

This demonstrates that the EBC equation (2) holds with exponent $d = 1/\tau > 2$, demonstrating that values of $d > 2$ can naturally arise even in well-structured convex programs.

Appendix B. Proofs

This section contains the proof of lemmas 11 and 12 and presents Lemma 25 and Lemma 26 that formally prove the inequalities (17) and (18) used in the proof of Theorem 13.

Proof of Lemma 11: We prove this lemma by showing $\tilde{r} > f^*$ and $0 < \tilde{\theta}(r) \leq \theta$ for any $r < f^*$. Since $\tilde{\mathbf{x}} \in \mathcal{X}$ and $g(\tilde{\mathbf{x}}) < 0$, by the definitions of $\tilde{r}$ and $H(\cdot)$, we have

$$H(\tilde{r}) \leq \max\{f(\tilde{\mathbf{x}}) - \tilde{r}, g(\tilde{\mathbf{x}})\} = g(\tilde{\mathbf{x}}) < 0. \tag{47}$$

By property 3 in Lemma 1, we must have $\tilde{r} > f^*$ which indicates $\tilde{\theta}(r) > 0$ for any $r < f^*$. It follows from the convexity of $H(r)$ and definition of θ in (6) that

$$H(r) \geq H(f^*) - \theta(r - f^*) = -\theta(r - f^*), \quad \forall r. \tag{48}$$

This further implies that, for $r < f^*$, $\tilde{\theta}(r) = \frac{-g(\tilde{\mathbf{x}})}{\tilde{r}-r} \leq \frac{-H(\tilde{r})}{\tilde{r}-r} \leq \frac{\theta(\tilde{r}-f^*)}{\tilde{r}-r} < \frac{\theta(\tilde{r}-r)}{\tilde{r}-r} = \theta$, where the first inequality is from (47), the second from (48) with $r = \tilde{r}$, and the last from $r < f^* < \tilde{r}$. ■

In the proof of the lemmas below we require the result of Proposition 23.

Proposition 23 *The function* $\dfrac{\alpha}{2}P(\mathbf{x};r)+r$ *is an increasing function in* r *for any* $\mathbf{x}\in\mathcal{X}$.

Proof The proof of this proposition follows from the convexity of $\frac{\alpha}{2}P(\mathbf{x};r)$ in r. In particular, let ξ_r denote the sub-gradient of $\frac{\alpha}{2}P(\cdot;r)$ with respect to r. It is straightforward to see $\xi_r\in[-1,0]$ for any r. Hence, from convexity of $\frac{\alpha}{2}P(\mathbf{x};r)$ in r, it follows that for $r\geq r'$, we have $\frac{\alpha}{2}P(\mathbf{x};r)\geq\frac{\alpha}{2}P(\mathbf{x};r')+\xi_r(r-r')\geq\frac{\alpha}{2}P(\mathbf{x};r')-1\cdot(r-r')$ which indicates $\frac{\alpha}{2}P(\mathbf{x};r)+r\geq\frac{\alpha}{2}P(\mathbf{x};r')+r'$. ■

For the remaining results included in this section, we need to introduce slightly different notations representing $\mathbf{x}_k^{(t_k)}$ and r_k to account for the number of restarts in each `fom` instance. More precisely, we use the notation $\mathbf{x}_k^{(l_k,t_k)}$ and $r_k^{(l_k)}$ to respectively represent the solution in the t_kth iteration of $\texttt{fom}_k$ after l_k restarts and the level parameter in $\texttt{fom}_k$ after l_k restarts.

Let k' be the smallest index at which the inequalities $P(\mathbf{x}_{k'}^{(l_{k'},t_{k'})};r_{k'}^{(l_{k'})})\leq BP(\mathbf{x}_{k'}^{(l_{k'},0)};r_{k'}^{(l_{k'})})$ and $P(\mathbf{x}_{k'}^{(l_{k'},0)};r_{k'}^{(l_{k'})})\geq 0$ hold (i.e. k' is the index found in Line 7 of Algorithm 2). RLS then initiates a restart at this index and also restarts all instances whose indices are greater than k'. Suppose this is the l_kth restart at $\texttt{fom}_k$ for $k\geq k'$. To understand how level parameters change after each restart, we reply on the following lemma where we show $r_k^{(l_k-1)}\geq r_k^{(l_k)}$ for each $k\geq k'$.

Lemma 24 *Suppose* α *and* B *are such that* $0<\alpha<B<1$. *Let* k' *be the index found in Line 7 of Algorithm 2. In addition, assume the level parameters* $r_k^{(l_k)}$ *for* $k\geq k'$ *are updated as in Line 14, i.e.,* $r_k^{(l_k)}=r_{k-1}^{(l_{k-1})}+\frac{\alpha}{2}P(\mathbf{x}_{k-1}^{(l_{k-1},0)};r_{k-1}^{(l_{k-1})})$ *for* $k\geq k'+1$. *We then have*

$$r_{k'}^{(l_{k'}-1)}-r_{k'}^{(l_{k'})}=0, \tag{49}$$

and

$$r_k^{(l_k-1)}-r_k^{(l_k)}\geq 0 \quad \textit{for } k\geq k'+1. \tag{50}$$

Proof The equation $r_{k'}^{(l_{k'})}=r_{k'}^{(l_{k'}-1)}$ directly follows from the steps of Algorithm 2. In particular, the level parameters are only updated for $k\geq k'+1$ after each restart at index k'. Recall that the index k' is selected such that $P(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})\geq 0$ and $P(\mathbf{x}_{k'}^{(l_{k'}-1,t_{k'})};r_{k'}^{(l_{k'}-1)})\leq BP(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})$. Since $\mathbf{x}_{k'}^{(l_{k'},0)}=\arg\min_{\mathbf{x}\in\left\{\mathbf{x}_{k'}^{(l_{k'}-1,t_{k'})},\mathbf{x}_1^{(l_1-1,0)},\ldots,\mathbf{x}_K^{(l_K-1,0)}\right\}}P(\mathbf{x};r_{k'}^{(l_{k'}-1)})$ and $r_{k'}^{(l_{k'})}=r_{k'}^{(l_{k'}-1)}$, we get

$$P(\mathbf{x}_{k'}^{(l_{k'},0)};r_{k'}^{(l_{k'})})\leq BP(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)}). \tag{51}$$

Now let's consider the index $k'+1$:

$$\begin{aligned}
r_{k'+1}^{(l_{k'+1}-1)}-r_{k'+1}^{(l_{k'+1})} &= r_{k'}^{(l_{k'}-1)}+\frac{\alpha}{2}P(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})-r_{k'}^{(l_{k'})}-\frac{\alpha}{2}P(\mathbf{x}_{k'}^{(l_{k'},0)};r_{k'}^{(l_{k'})})\\
&= \frac{\alpha}{2}P(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})-\frac{\alpha}{2}P(\mathbf{x}_{k'}^{(l_{k'},0)};r_{k'}^{(l_{k'})})\\
&\geq \frac{\alpha}{2}P(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})-\frac{\alpha}{2}BP(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})\\
&= \frac{\alpha}{2}(1-B)P(\mathbf{x}_{k'}^{(l_{k'}-1,0)};r_{k'}^{(l_{k'}-1)})\geq 0
\end{aligned} \tag{52}$$

where we used the definitions of $r_{k'+1}$ to obtain the first equality; (49) for the second; and (51) for the first inequality and the inequality $P(\mathbf{x}_{k'}^{(l_{k'}-1,0)}; r_{k'}^{(l_{k'}-1)}) \geq 0$ for the last; Similarly for $k \geq k'+2$, we show

$$\begin{aligned} r_k^{(l_k-1)} - r_k^{(l_k)} &= r_{k-1}^{(l_{k-1}-1)} + \frac{\alpha}{2} P(\mathbf{x}_{k-1}^{(l_{k-1}-1,0)}; r_{k-1}^{(l_{k-1}-1)}) - r_{k-1}^{(l_{k-1})} - \frac{\alpha}{2} P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_{k-1}^{(l_{k-1})}) \\ &\geq r_{k-1}^{(l_{k-1}-1)} + \frac{\alpha}{2} P(\mathbf{x}_{k-1}^{(l_{k-1}-1,0)}; r_{k-1}^{(l_{k-1}-1)}) - \left(r_{k-1}^{(l_{k-1})} + \frac{\alpha}{2} P(\mathbf{x}_{k-1}^{(l_{k-1}-1,0)}; r_{k-1}^{(l_{k-1})}) \right) \geq 0, \end{aligned}$$

where the first inequality follows from Line 13 of Algorithm 2 which indicates that

$$P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_{k-1}^{(l_{k-1})}) \leq P(\mathbf{x}_{k-1}^{(l_{k-1}-1,0)}; r_{k-1}^{(l_{k-1})}).$$

The last inequality form Proposition 23 and the fact that $r_{k-1}^{(l_{k-1}-1)} \geq r_{k-1}^{(l_{k-1})}$ for $k \geq k'+2$ which can be easily verified by (52). ■

Proof of Lemma 12: To prove this lemma, we claim that if the inequality (5) ($\alpha P(\mathbf{x}_k^{(0)}; r_k) \leq f^* - r_k$) is satisfied at some k, it will continue to be satisfied after each restart. By Theorem 10, it will then follows that none of the indices that are smaller than the current critical index can become a critical index after the restarts. Therefore, the critical index cannot decrease throughout the algorithm. Our claim can be proved by defining the quantity $V_k\left(l_k\right)$, $k = 0, 1, 2, \ldots$ as follows:

$$V_k\left(l_k\right) := f^* - r_k^{(l_k)} - \alpha P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}). \tag{53}$$

We show $V_k(l_k)$ is a non-decreasing function of l_k when the inequality (5) holds at index k. More precisely, we show the inequality $V_k(l_k) \geq V_k(l_k - 1)$ holds for such k. Therefore, assuming (5) is satisfied in $l_k - 1$ restarts, it will continue to hold after next restart since $V_k(l_k) \geq V_k(l_k - 1) \geq 0$.

Fix an index $k \in [0, \tilde{K}]$ for which we have $\alpha P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}) < f^* - r_k^{(l_k-1)}$. The restarts that occur at larger indices than k do not change $V_k(l_k - 1)$ since $\mathbf{x}_k^{(l_k-1,0)}$ and $r_k^{(l_k-1)}$ remain the same. Therefore, we only focus on the restarts that either happen (i) at index k or (ii) any smaller index than k.

Case (i): When the restart occurs at index k, it means we have found a solution $\mathbf{x}_k^{(l_k-1,t_k)}$ such that $P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}) \geq 0$ and $P(\mathbf{x}_k^{(l_k-1,t_k)}; r_k^{(l_k-1)}) \leq BP(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)})$. Since $\mathbf{x}_k^{(l_k,0)} = \mathbf{x}_k^{(l_k-1,t_k)}$ and $r_k^{(l_k)} = r_k^{(l_k-1)}$ which follows from the restarting steps, we get $P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \leq BP(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)})$. Using this inequality and the fact that $r_k^{(l_k)} = r_k^{(l_k-1)}$, it is straightforward to see $V_k(l_k) \geq V_k(l_k-1) + \alpha(1-B)P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)})$. This inequality further implies that

$$\begin{aligned} V_k(l_k) &\geq V_k(l_k-1) + \alpha(1-B)P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}) \geq V_k(l_k-1) + \alpha(1-B)\theta \left(f^* - r_k^{(l_k-1)} \right) \\ &> V_k(l_k-1) + \alpha(1-B)\theta(f^* - r_k^{(l_k-1)} - \alpha P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)})) \\ &= \left(1 + \alpha(1-B)\theta\right) V_k(l_k-1) > V_k(l_k-1), \end{aligned} \tag{54}$$

where the second inequality follows from $0 \leq \alpha P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}) \leq f^* - r_k^{(l_k-1)}$ and Lemma 3 that shows $\theta \leq \frac{H(r_k^{(l_k-1)})}{f^* - r_k^{(l_k-1)}} \leq \frac{P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)})}{f^* - r_k^{(l_k-1)}}$, and the third from the inequality $P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}) \geq 0$ which holds since the restart has occurred at index k.

Case (ii): In this case we have

$$V_k(l_k) = f^* - r_k^{(l_k)} - \alpha P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \geq f^* - r_k^{(l_k-1)} - \alpha P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}) = V_k(l_k - 1). \tag{55}$$

To obtain the above inequality we used Line 13 of Algorithm 2 which indicates that $P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \leq P(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k)})$. In addition, following the proof of Proposition 23, one can show that the function $f^* - r - \alpha P(\mathbf{x}; r)$ is decreasing in r for any $\mathbf{x} \in \mathcal{X}$. Therefore, the inequality (55) holds since $r_k^{(l_k-1)} \geq r_k^{(l_k)}$ which follows from Lemma 24. Considering (54) and (55), our proof is complete. ■

In the following lemmas we prove the inequalities (17) and (18) used in the proof of Theorem 13 in Section 3.

Lemma 25 *Suppose α, B, and ϵ are such that $0 < \alpha < B < 1$ and $\epsilon > 0$. Consider an index k and $\tilde{K}$ as defined in (11) for $r = r_{ini}$. Let D_k denote the total number of desirable restarts starting at $\texttt{fom}_k$ with $k < k^*$ until an ϵ-optimal and ϵ-feasible solution in found, assuming k^* is the critical index at the time of restart. Then we have*

$$D_k = \sum_{k^*=k+1}^{\tilde{K}} D_{kk^*} \leq \ln\left(\frac{4 - \alpha\theta}{3\alpha\theta}\right) / \ln\left(1 + \alpha(1-B)\theta/4\right),$$

where D_{kk^*} is the total number of desirable restarts starting at $\texttt{fom}_k$ with $k < k^*$.

Proof Using the definitions of D_k and D_{kk^*}, it is easy to see that $D_k = \sum_{k^*=k+1}^{\tilde{K}} D_{kk^*}$ because by Lemma 12, the critical index k^* is non-decreasing and can vary between $k+1$ and $\tilde{K}$. Let's define

$$V_k'(l_k) := \frac{P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)})}{f^* - r_k^{(l_k)} - \frac{\alpha}{4} P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)})},$$

where the denominator is a modification of the quantity $V_k(l_k)$ in (53).

We show $V_k'(l_k)$ is a non-increasing function of l_k. More precisely, we show the inequality $V_k'(l_k) \leq V_k'(l_k - 1)$ holds for each $k < k^*$. The restarts that occur at larger indices than k do not change $V_k'(l_k - 1)$ since $\mathbf{x}_k^{(l_k-1,0)}$ and $r_k^{(l_k-1)}$ remain the same. Therefore, we only focus on the restarts that either happen at index k or any smaller index than k.

Since $k < k^*$, Theorem 10 indicates that for all $l_k \geq 0$ we have $r_k^{(l_k)} < f^*$ and the inequality (5) holds at k. Hence, we can easily verify that $P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \geq H(r_k^{(l_k)}) > 0$, $V_k'(l_k) > 0$, $r_k^{(l_k)} \geq r_0^{(l_0)}$, and

$$P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \leq P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_k^{(l_k)}) \leq P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_{k-1}^{(l_{k-1})}). \tag{56}$$

The above inequality follows from Line 13 of Algorithm 2, Lemma 24, and the fact that the function $P(\cdot; r)$ is decreasing in r. Following the same proof as in the proof of Lemma 12, one can also show

$$f^* - r_k^{(l_k)} - \frac{\alpha}{4} P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \geq (1 + \alpha(1-B)\theta/4) \left(f^* - r_k^{(l_k-1)} - \frac{\alpha}{4} P\left(\mathbf{x}_k^{(l_k-1,0)}; r_k^{(l_k-1)}\right)\right), \tag{57}$$

if the restarts occurs at index k and

$$f^* - r_k^{(l_k)} - \frac{\alpha}{4}P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \geq f^* - r_k^{(l_{k-1})} - \frac{\alpha}{4}P(\mathbf{x}_k^{(l_{k-1},0)}; r_k^{(l_{k-1})}), \tag{58}$$

if the restarts occurs at an index smaller than k. Combining (57) and (58) with the inequality (56), we can respectively show

$$\begin{cases} V_k'(l_k) \leq \dfrac{1}{(1+\alpha(1-B)\theta/4)} V_k'(l_k - 1) & \text{if the restarts occurs at index } k, \\ V_k'(l_k) \leq V_k'(l_k - 1) & \text{if the restarts occurs at an index smaller than } k. \end{cases}$$

Suppose $\hat{l}_k$ denotes the smallest restarting iteration at which the index k becomes smaller than a critical index. By recursively applying the definition of V', we obtain

$$\begin{aligned} V_k'(l_k) &\leq \frac{P(\mathbf{x}_k^{(\hat{l}_k,0)}; r_k^{(\hat{l}_k)})}{(1+\alpha(1-B)\theta/4)^{D_k}\left(f^* - r_k^{(\hat{l}_k)} - \frac{\alpha}{4}P(\mathbf{x}_k^{(\hat{l}_k,0)}; r_k^{(\hat{l}_k)})\right)} \\ &\leq \frac{1}{(1+\alpha(1-B)\theta/4)^{D_k}} \cdot \frac{(1/\alpha)(f^* - r_k^{(\hat{l}_k)})}{(3/4)(f^* - r_k^{(\hat{l}_k)})} = \frac{1}{(1+\alpha(1-B)\theta/4)^{D_k}} \cdot \frac{4}{3\alpha}, \end{aligned} \tag{59}$$

where the second inequality holds because for any $k < k^*$, we have $P(\mathbf{x}_k^{(\hat{l}_k,0)}; r_k^{(\hat{l}_k)}) \leq (f^* - r_k^{(\hat{l}_k)})/\alpha$.

In addition since $P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \geq H(r_k^{(l_k)}) \geq \theta(f^* - r_k^{(l_k)})$ which holds by Lemma 3, we can easily verify that

$$V_k'(l_k) \geq \frac{\theta}{1 - \frac{\alpha\theta}{4}}. \tag{60}$$

Then from the inequalities (59) and (60) it follows that $\dfrac{\theta}{1-\frac{\alpha\theta}{4}} \leq \dfrac{1}{(1+\alpha(1-B)\theta/4)^{D_k}} \cdot \dfrac{4}{3\alpha}$, We can then obtain our desirable bound on D_k by taking a logarithmic transformation on both sides of the above inequality and organizing terms. ■

Lemma 26 *Suppose α, B, and ϵ are such that $0 < \alpha < B < 1$ and $\epsilon > 0$. Consider a critical index k^* and $\tilde{K}$ as defined in (11) for $r = r_{ini}$. Let $D_{k^*k^*}$ denote the total number of desirable restarts starting at $\texttt{fom}_{k^*}$ before the critical index increases or an ϵ-feasible and ϵ-optimal solution is found. Then $D_{k^*k^*} \leq \ln(2/\alpha\theta)/\ln(1/B)$.*

Proof Suppose k^* is the critical index while there is no ϵ-feasible and ϵ-optimal solution is found. Consider $k \leq k^*$. Using the updates of $\mathbf{x}_k$ and r_k in Algorithm 2 and the fact that the function $P(\cdot; r_k^{(l_k)})$ is non-increasing in $r_k^{(l_k)}$ we can obtain

$$P(\mathbf{x}_k^{(l_k,0)}; r_k^{(l_k)}) \leq P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_k^{(l_k)}) \leq P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_{k-1}^{(l_{k-1})}). \tag{61}$$

Notice that we used the inequality $r_{k-1}^{(l_{k-1})} \leq r_k^{(l_k)}$ from Theorem 10 to obtain the second inequality above. In addition,

$$f^* - r_k^{(l_k)} = f^* - r_{k-1}^{(l_{k-1})} - \frac{\alpha}{2}P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_{k-1}^{(l_{k-1})}) \geq \frac{1}{2}\left(f^* - r_{k-1}^{(l_{k-1})}\right), \tag{62}$$

where the inequality holds since $k \le k^*$ and hence Theorem 10 indicates that $P(\mathbf{x}_{k-1}^{(l_{k-1},0)}; r_{k-1}^{(l_{k-1})}) \le (f^* - r_{k-1}^{(l_{k-1})})/\alpha$. Define $U_{k^*}(l_{k^*}) := \dfrac{P(\mathbf{x}_{k^*}^{(l_{k^*},0)}; r_{k^*}^{(l_{k^*})})}{f^* - r_{k^*}^{(l_{k^*})}}$. We show this quantity is non-increasing at each desirable restart. Especially, it reduces with a factor of B if desirable restarts occur at $\texttt{fom}_{k^*}$. More precisely, we will prove

$$\begin{cases} U_{k^*}(l_{k^*}) \le U_{k^*}(l_{k^*}-1) & \text{if } \texttt{fom}_{k^*} \text{ instance is updated by a restart at } \texttt{fom}_k \text{ with } k < k^*, \\ U_{k^*}(l_{k^*}) \le B \cdot U_{k^*}(l_{k^*}-1) & \text{if } \texttt{fom}_{k^*} \text{ instance is updated by a restart at } \texttt{fom}_{k^*}. \end{cases}$$

Finally, by providing a lower bound on $U_{k^*}(l_{k^*})$ for any l_{k^*}, we show that this quantity can only be reduced by a finite number of desirable restarts.

First, let's consider a case where $\mathbf{x}_{k^*}^{(l_{k^*},0)}$ and $r_{k^*}^{(l_{k^*})}$ are generated by a restart at $\texttt{fom}_k$ with $k < k^*$. We then have

$$U_{k^*}(l_{k^*}) = \frac{P(\mathbf{x}_{k^*}^{(l_{k^*},0)}; r_{k^*}^{(l_{k^*})})}{f^* - r_{k^*}^{(l_{k^*})}} \le \frac{P(\mathbf{x}_{k^*}^{(l_{k^*}-1,0)}; r_{k^*}^{(l_{k^*}-1)}) + r_{k^*}^{(l_{k^*}-1)} - r_{k^*}^{(l_{k^*})}}{f^* - r_{k^*}^{(l_{k^*}-1)} + r_{k^*}^{(l_{k^*}-1)} - r_{k^*}^{(l_{k^*})}}$$

$$\le \frac{P(\mathbf{x}_{k^*}^{(l_{k^*}-1,0)}; r_{k^*}^{(l_{k^*}-1)})}{f^* - r_{k^*}^{(l_{k^*}-1)}} = U_{k^*}(l_{k^*}-1), \tag{63}$$

where the first inequality follows from the definition of $P(\mathbf{x}; r)$ and Line 13 of Algorithm 2, and the second from $r_{k^*}^{(l_{k^*}-1)} - r_{k^*}^{(l_{k^*})} \ge 0$ (see Lemma 24) and the fact that k^* is a critical index and hence $P(\mathbf{x}_{k^*}^{(l_{k^*}-1,0)}, r_{k^*}^{(l_{k^*}-1)}) \ge \frac{f^* - r_{k^*}^{(l_{k^*}-1)}}{\alpha} \ge f^* - r_{k^*}^{(l_{k^*}-1)}$. Now suppose $\mathbf{x}_{k^*}^{(l_{k^*},0)}$ and $r_{k^*}^{(l_{k^*})}$ are generated by a restart at $\texttt{fom}_{k^*}$, Since every desirable restart from $\texttt{fom}_{k^*}$ happens when $P(\mathbf{x}_{k^*}^{(l_{k^*}-1,t_{k^*})}; r_{k^*}^{(l_{k^*}-1)}) \le BP(\mathbf{x}_{k^*}^{(l_{k^*}-1,0)}; r_{k^*}^{(l_{k^*})})$, we get $U_{k^*}(l_{k^*}) \le \frac{P(\mathbf{x}_{k^*}^{(l_{k^*},0)}; r_{k^*}^{(l_{k^*})})}{f^* - r_{k^*}^{(l_{k^*})}} \le \frac{P(\mathbf{x}_{k^*}^{(l_{k^*}-1,t_{k^*})}; r_{k^*}^{(l_{k^*}-1)})}{f^* - r_{k^*}^{(l_{k^*}-1)}} \le \frac{BP(\mathbf{x}_{k^*}^{(l_{k^*}-1,0)}; r_{k^*}^{(l_{k^*}-1)})}{f^* - r_{k^*}^{(l_{k^*}-1)}} = BU_{k^*}(l_{k^*}-1)$, where we used $r_{k^*}^{(l_{k^*})} = r_{k^*}^{(l_{k^*}-1)}$. Using this inequality and (63) recursively, we get

$$U_{k^*}(l_{k^*}) = \frac{P(\mathbf{x}_{k^*}^{(l_{k^*},0)}; r_{k^*}^{(l_{k^*})})}{f^* - r_{k^*}^{(l_{k^*})}} \le \frac{B^{D_{k^*k^*}} P(\mathbf{x}_{k^*}^{(\hat{l}_{k^*},0)}; r_{k^*}^{(\hat{l}_{k^*})})}{f^* - r_{k^*}^{(\hat{l}_{k^*})}} \le \frac{B^{D_{k^*k^*}} P\left(\mathbf{x}_{k^*}^{(\hat{l}_{k^*-1},0)}; r_{k^*-1}^{(\hat{l}_{k^*-1})}\right)}{\frac{1}{2}\left(f^* - r_{k^*-1}^{(\hat{l}_{k^*-1})}\right)}$$

$$\le \frac{2B^{D_{k^*k^*}}}{\alpha}, \tag{64}$$

where the first inequality holds since $r_{k^*}^{(\hat{l}_{k^*})} \ge r_{k^*}^{(l_{k^*})}$ which follows from Lemma 24. The second holds by (61) and (62) applied to $k = k^*$, and the third by Theorem 10 which indicates that $P(\mathbf{x}_{k^*-1}^{(\hat{l}_{k^*-1},0)}; r_{k^*-1}^{(\hat{l}_{k^*-1})}) \le (f^* - r_{k^*-1}^{(\hat{l}_{k^*-1})})/\alpha$.

Moreover, since $P(\mathbf{x}_{k^*}^{(l_{k^*},0)}; r_{k^*}^{(l_{k^*})}) \ge H(r_{k^*}^{(l_{k^*})}) \ge \theta(f^* - r_{k^*}^{(l_{k^*})})$ which holds by Lemma 3, we get $U_{k^*}(l_{k^*}) = \frac{P(\mathbf{x}_{k^*}^{(l_{k^*},0)}; r_{k^*}^{(l_{k^*})})}{f^* - r_{k^*}^{(l_{k^*})}} \ge \frac{\theta(f^* - r_{k^*}^{(l_{k^*})})}{f^* - r_{k^*}^{(l_{k^*})}} = \theta$. This inequality and (64) together imply $\theta \le \frac{2B^{D_{k^*k^*}}}{\alpha}$, which results in the following upper bound on $D_{k^*k^*}$: $D_{k^*k^*} \le \ln(2/\alpha\theta)/\ln(1/B)$.■

Appendix C. Linear Equality Constraints

As discussed in Section 3, the restarting level set method, described in Algorithm 2, requires $K+1$ instances of the first-order method. Here, K (defined in (8)) depends on certain unknown parameters f^* and θ. Under Assumption 1, we established a lower bound on the condition number θ and an upper bound on f^*. Utilizing these established bounds, we were able to formulate an approximate value for K, termed $\tilde{K}$, seen in (11). This $\tilde{K}$ is something we can calculate and apply in implementing Algorithm 2. Notably, our subsequent analysis remains applicable even when Assumption 1 is not met exactly where linear equality constraints are in play. This situation renders Assumption 1 inapplicable due to the equality constraints. Despite this, we attempt to find a computable lower bound on θ in this context. It's important to highlight that, in this particular case, the lower bound, denoted as $\tilde{\theta}$, hinges on having knowledge of the parameter M. This does introduce a limitation. Suppose

$$f^* := \min_{\mathbf{x}\in\mathcal{X}} \left\{ f(\mathbf{x}) := f_0(\mathbf{x}) \quad \text{s.t.} \quad g(\mathbf{x}) := \max_{i=1,\ldots,m} f_i(\mathbf{x}) \le 0, \mathbf{A}\mathbf{x} = \mathbf{b} \right\}, \tag{65}$$

where f_i for $i = 0, 1, \ldots, m$ are convex real-valued functions, $\mathcal{X} \subset \mathbb{R}^n$ is a closed convex set onto which the projection mapping is easy to compute, $\mathbf{A} \in \mathbb{R}^{l\times n}$ is a full row rank matrix of size $l \times n$ and $\mathbf{b} \in \mathbb{R}^l$ is a l-dimensional vector. In this case, $P(\mathbf{x}; r)$ is defined as

$$P(\mathbf{x}; r) := \max \left\{ f(\mathbf{x}) - r, g(\mathbf{x}), \|\mathbf{A}\mathbf{x} - \mathbf{b}\|_\infty \right\}. \tag{66}$$

In this section, we consider a modification of Assumption 1 as describe below as well as Assumption 3.

Assumption 5 *There exists a computable solution $\tilde{\mathbf{x}} \in \operatorname{int}(\mathcal{X})$ such that $\mathbf{A}\tilde{\mathbf{x}} = \mathbf{b}$ and $g(\tilde{\mathbf{x}}) < 0$.*

Lemma 27 *Let $\tilde{\mathbf{x}}$ be the feasible solution in Assumption 5 and θ be the condition measure defined in* (6). *It holds that*

$$\theta \ge \left(1 + \frac{f(\tilde{\mathbf{x}}) - r_{ini}}{-g(\tilde{\mathbf{x}})} + \sqrt{l} \left\| (\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{A} \right\| \left(\frac{M(f(\tilde{\mathbf{x}}) - r_{ini} - g(\tilde{\mathbf{x}}))}{-g(\tilde{\mathbf{x}})} + \frac{f(\tilde{\mathbf{x}}) - r_{ini}}{\operatorname{dist}(\tilde{\mathbf{x}}, \partial\mathcal{X})} \right) \right)^{-1}.$$

Proof Let $\mathbf{x}^*$ be the optimal solution of (65). Under Assumption 5, there exist Lagrangian multipliers $\lambda^* \ge 0$ and $\gamma^* \in \mathbb{R}^l$ such that $\lambda^* g(\mathbf{x}^*) = 0$ and

$$\boldsymbol{\zeta}_f + \lambda^* \boldsymbol{\zeta}_g + \mathbf{A}^\top \gamma^* + \mathbf{u}^* = \mathbf{0}, \tag{67}$$

where $\boldsymbol{\zeta}_f \in \partial f(\mathbf{x}^*)$, $\boldsymbol{\zeta}_g \in \partial g(\mathbf{x}^*)$, $\mathbf{u}^* \in \mathcal{N}_{\mathcal{X}}(\mathbf{x}^*)$ and $\mathcal{N}_{\mathcal{X}}(\mathbf{x}^*)$ is the normal cone of $\mathcal{X}$ at $\mathbf{x}^*$. Let $\bar{\gamma}^+ \in \mathbb{R}^l_+$ and $\bar{\gamma}^- \in \mathbb{R}^l_+$ are defined such that $\bar{\gamma}_i^+ = \gamma_i^*$ and $\bar{\gamma}_i^- = 0$ if $\gamma_i^* \ge 0$ and $\bar{\gamma}_i^+ = 0$ and $\bar{\gamma}_i^- = -\gamma_i^*$ if $\gamma_i^* < 0$ for $i = 1, \ldots, l$. Let $\mu := 1 + \lambda^* + \|\bar{\gamma}^+\|_1 + \|\bar{\gamma}^-\|_1$. Then we can normalize (67) as

$$\frac{1}{\mu} \left(\boldsymbol{\zeta}_f + \lambda^* \boldsymbol{\zeta}_g + \mathbf{A}^\top \bar{\gamma}^+ - \mathbf{A}^\top \bar{\gamma}^- \right) + \mathbf{v}^* = \mathbf{0}, \tag{68}$$

where $\mathbf{v}^* = \dfrac{1}{\mu}\mathbf{u} \in \mathcal{N}_{\mathcal{X}}(\mathbf{x}^*)$. Let Δ denotes the non-negative probability simplex. From the definition of $H(r)$ it then follows that for any r,

$$\begin{aligned} H(r) &= \min_{\mathbf{x}\in\mathcal{X}} \max\left\{f(\mathbf{x}) - r, g(\mathbf{x}), \max(\mathbf{A}\mathbf{x}-\mathbf{b}), \max(-\mathbf{A}\mathbf{x}+\mathbf{b})\right\} \\ &= \min_{\mathbf{x}\in\mathcal{X}} \max_{(\alpha,\lambda,\gamma^+,\gamma^-)\in\Delta} \left\{\alpha(f^*-r) + \lambda g(x) + \gamma^+(\mathbf{A}\mathbf{x}-\mathbf{b}) + \gamma^-(-\mathbf{A}\mathbf{x}+\mathbf{b})\right\} \\ &\geq \frac{1}{\mu}\min_{\mathbf{x}\in\mathcal{X}} \Big\{ f(\mathbf{x}^*) + \boldsymbol{\zeta}_f^\top(\mathbf{x}-\mathbf{x}^*) - r + \lambda^*(g(\mathbf{x}^*) + \boldsymbol{\zeta}_g^\top(\mathbf{x}-\mathbf{x}^*)) + (\mathbf{x}-\mathbf{x}^*)^\top\mathbf{A}^\top\gamma^+ \\ &\qquad -(\mathbf{x}-\mathbf{x}^*)^\top\mathbf{A}^\top\gamma^- \Big\} \geq \frac{1}{\mu}\left(f(\mathbf{x}^*) - r + \lambda^* g(\mathbf{x}^*)\right) = H(f^*) - \frac{1}{\mu}\left(r - f^*\right), \end{aligned}$$

where the first inequality is established through the convexity of f and g, along with the conditions $\mathbf{A}\mathbf{x}^* = \mathbf{b}$ and $\frac{1}{\mu}\cdot(1, \lambda^*, \bar{\gamma}^+, \bar{\gamma}^-) \in \Delta$. The second inequality is a result of (68). The final equality can be deduced from the observations that $f(\mathbf{x}^*) = f^*$, $\lambda^* g(\mathbf{x}^*) = 0$, and $H(f^*) = 0$. This indicates that $-\mu^{-1}$ acts as the subgradient of $H(r)$ at $r = f^*$. By referring to the definition of θ in (6), we ascertain that $\theta \geq \mu^{-1}$. Consequently, a lower bound for θ can be established by bounding λ^* and $|\bar{\gamma}^+|_1 + |\bar{\gamma}^-|_1 = |\gamma^*|_1$ from above. To achieve this, we borrow some of the analysis from the proof of Lemma 3 in Lin et al. (2022).

An upper bound on λ^*: From the convexity of g and the equality (67), it follows that

$$\lambda^* g(\tilde{\mathbf{x}}) \geq \lambda^*\left[g(\mathbf{x}^*) + (\tilde{\mathbf{x}}-\mathbf{x}^*)^\top\boldsymbol{\zeta}_g\right] = -(\tilde{\mathbf{x}}-\mathbf{x}^*)^\top(\boldsymbol{\zeta}_f + \mathbf{A}^\top\gamma^* + \mathbf{u}^*) = -(\tilde{\mathbf{x}}-\mathbf{x}^*)^\top(\boldsymbol{\zeta}_f + \mathbf{u}^*), \quad (69)$$

where the first inequality and the last equality hold due to the non-negativity of λ^* and $\mathbf{A}\tilde{\mathbf{x}} = \mathbf{A}\mathbf{x}^* = \mathbf{b}$, respectively.

In the case of $u^* = 0$, inequality (69) suggests that

$$-\lambda^* g(\tilde{\mathbf{x}}) \leq (\tilde{\mathbf{x}}-\mathbf{x}^*)^\top\boldsymbol{\zeta}_f \leq f(\tilde{\mathbf{x}}) - f^* \leq f(\tilde{\mathbf{x}}) - r_{\text{ini}} \quad (70)$$

where the second inequality arises from convexity of f and the third from $r_{\text{ini}} \leq f^*$. The validity of this inequality also holds when $u^* \neq 0$, and we shortly proceed to demonstrate it. As such, λ^* can be bounded above as follows

$$\lambda^* \leq \frac{f(\tilde{\mathbf{x}}) - r_{\text{ini}}}{-g(\tilde{\mathbf{x}})}. \quad (71)$$

To establish (70) in the scenario of $u^* \neq 0$, consider that $\mathbf{x}^* \in \partial\mathcal{X}$. Let $\mathcal{H}$ represent the supporting hyperplane of $\mathcal{X}$ at $\mathbf{x}^*$, i.e. $\left\{\mathbf{x} \in \mathbb{R}^n \,|\, (\mathbf{x}-\mathbf{x}^*)^\top\mathbf{u}^* = 0\right\}$. Given that $\text{dist}(\tilde{\mathbf{x}}, \mathcal{H}) = |(\mathbf{x}^*-\tilde{\mathbf{x}})^\top\mathbf{u}^*|/\|\mathbf{u}^*\|$ and $\text{dist}(\tilde{\mathbf{x}}, \mathcal{H}) \geq \text{dist}(\tilde{\mathbf{x}}, \partial\mathcal{X}) > 0$, we get $(\mathbf{x}^*-\tilde{\mathbf{x}})^\top\mathbf{u}^* = |(\mathbf{x}^*-\tilde{\mathbf{x}})^\top\mathbf{u}^*| = \text{dist}(\tilde{\mathbf{x}}, \mathcal{H})\|\mathbf{u}^*\| \geq \text{dist}(\tilde{\mathbf{x}}, \partial\mathcal{X})\|\mathbf{u}^*\|$, where the first equality follows from $\mathbf{u}^* \in \mathcal{N}_{\mathcal{X}}(\mathbf{x}^*)$. Applying this inequality alongside (69) yields $-\lambda^* g(\tilde{\mathbf{x}}) + \text{dist}(\tilde{\mathbf{x}}, \partial\mathcal{X})\|\mathbf{u}^*\| \leq (\tilde{\mathbf{x}}-\mathbf{x}^*)^\top\boldsymbol{\zeta}_f \leq f(\tilde{\mathbf{x}}) - f^* \leq f(\tilde{\mathbf{x}}) - r_{\text{ini}}$, and subsequently,

$$-\lambda^* g(\tilde{\mathbf{x}}) \leq f(\tilde{\mathbf{x}}) - r_{\text{ini}} \text{ and } \|\mathbf{u}^*\| \leq \frac{f(\tilde{\mathbf{x}}) - r_{\text{ini}}}{\text{dist}(\tilde{\mathbf{x}}, \partial\mathcal{X})}. \quad (72)$$

An upper bound on γ^*: Utilizing equation (67) along with the full row rank of $\mathbf{A}$, we deduce that $\gamma^* = -(\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{A}\left(\boldsymbol{\zeta}_f + \lambda^*\boldsymbol{\zeta}_g + \mathbf{u}^*\right)$, which, together with (71), leads to

$$\|\gamma^*\|_1 \leq \sqrt{l}\|\gamma^*\| \leq \sqrt{l}\|(\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{A}\|\left(M(1+\lambda^*) + \|\mathbf{u}^*\|\right)$$

$$\leq\sqrt{l}\|(\mathbf{A}\mathbf{A}^\top)^{-1}\mathbf{A}\|\left(\frac{M(f(\tilde{\mathbf{x}})-r_{\text{ini}}-g(\tilde{\mathbf{x}}))}{-g(\tilde{\mathbf{x}})}+\frac{f(\tilde{\mathbf{x}})-r_{\text{ini}}}{\text{dist}(\tilde{\mathbf{x}},\partial\mathcal{X})}\right).$$

In the above inequalities, we incorporated (72) and made use of Assumption 3. By applying this upper bound of $\|\gamma^*\|_1$ and the upper bound of λ^* from (71) to μ^{-1}, we successfully establish the desired lower bound for θ. ■

Remark 28 *Note that Lemma 27 encompasses Lemma 11 as a special case. In particular, in cases where linear equality constraints are absent, the introduction of γ^* in* (67) *becomes unnecessary. Then the aforementioned lemma implies $\theta \geq (1+\lambda^*)^{-1}$. Given that* (71) *remains applicable, we still have $\theta \geq (1+\lambda^*)^{-1} = \frac{g(\tilde{\mathbf{x}})}{f^*-f(\tilde{\mathbf{x}})+g(\tilde{\mathbf{x}})} \geq \tilde{\theta}(r)$ for any $r < f^*$, where $\tilde{\theta}(r)$ is defined in Lemma 11.*

References

B. Adcock, M. J. Colbrook, and M. Neyra-Nesterenko. Restarts subject to approximate sharpness: a parameter-free and optimal scheme for first-order methods. *Foundations of Computational Mathematics*, pages 1–56, 2025.

A. Aravkin, J. Burke, D. Drusvyatskiy, M. Friedlander, and S. Roy. Level-set methods for convex optimization. *Mathematical Programming*, 174(1-2):359–390, 2019.

A. Bayandina, P. Dvurechensky, A. Gasnikov, F. Stonyakin, and A. Titov. *Mirror descent and convex optimization problems with non-smooth inequality constraints*, pages 181–213. Springer, 2018.

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

J. Bolte, P. Nguyen, J. Peypouquet, and W. Suter. From error bounds to the complexity of first-order descent methods for convex functions. *Mathematical Programming*, 165(2): 471–507, 2017.

D. Boob, Q. Deng, and G. Lan. Stochastic first-order methods for convex and nonconvex functional constrained optimization. *Mathematical Programming*, 197(1):215–279, 2023.

A. Chambolle and T. Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40:120–145, 2011.

V. Charisopoulos and D. Davis. A superlinearly convergent subgradient method for sharp semismooth problems. *Mathematics of Operations Research*, 49(3):1678–1709, 2024.

D. Davis and L. Jiang. A local nearly linearly convergent first-order method for nonsmooth functions with quadratic growth. *Foundations of Computational Mathematics*, pages 1–82, 2024.

D. Davis, D. Drusvyatskiy, K. J. MacPhee, and C. Paquette. Subgradient methods for sharp weakly convex functions. *Journal of Optimization Theory and Applications*, 179(3): 962–982, 2018.

D. Davis, D. Drusvyatskiy, and V. Charisopoulos. Stochastic algorithms with geometric step decay converge linearly on sharp functions. *Mathematical Programming*, 207(1):145–190, 2024.

Q. Deng, G. Lan, and Z. Lin. Uniformly optimal and parameter-free first-order methods for convex and function-constrained optimization. *arXiv preprint arXiv:2412.06319*, 2024.

M. Díaz and B. Grimmer. Optimal convergence rates for the proximal bundle method. *SIAM Journal on Optimization*, 33(2):424–454, 2023.

D. Drusvyatskiy and C. Paquette. Efficiency of minimizing compositions of convex functions and smooth maps. *Mathematical Programming*, 178:503–558, 2019.

A. Ene, H. L. Nguyen, and A. Vladu. Adaptive gradient methods for constrained convex optimization and variational inequalities. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(8):7314–7321, 2021.

R. Estrin and M. P. Friedlander. A perturbation view of level-set methods for convex optimization. *Optimization Letters*, 14(8):1989–2006, 2020.

O. Fercoq. Quadratic error bound of the smoothed gap and the restarted averaged primal-dual hybrid gradient. *Open Journal of Mathematical Optimization*, 4:1–34, 2023.

O. Fercoq and Z. Qu. Adaptive restart of accelerated gradient methods under local quadratic growth condition. *IMA Journal of Numerical Analysis*, 39(4):2069–2095, 2019.

O. Fercoq and Z. Qu. Restarting the accelerated coordinate descent method with a rough strong convexity estimate. *Computational Optimization and Applications*, 75(1):63–91, 2020.

O. Fercoq, A. Alacaoglu, I. Necoara, and V. Cevher. Almost surely constrained convex optimization. In *International Conference on Machine Learning*, pages 1910–1919. PMLR, 2019.

R. Freund and H. Lu. New computational guarantees for solving convex optimization problems with first order methods, via a function growth condition measure. *Mathematical Programming*, 170(2):445–477, 2018.

G. Goh, A. Cotter, M. Gupta, and M. P. Friedlander. Satisfying real-world goals with dataset constraints. *Advances in Neural Information Processing Systems*, 29, 2016.

B. Grimmer. Radial subgradient method. *SIAM Journal on Optimization*, 28(1):459–469, 2018.

B. Grimmer. Convergence rates for deterministic and stochastic subgradient methods without lipschitz continuity. *SIAM Journal on Optimization*, 29(2):1350–1365, 2019.

B. Grimmer. General hölder smooth convergence rates follow from specialized rates assuming growth bounds. *Journal of Optimization Theory and Applications*, 197(1):51–70, 2023.

B. Grimmer. On optimal universal first-order methods for minimizing heterogeneous sums. *Optimization Letters*, 18(2):427–445, 2024.

I. Guyon, S. Gunn, A. Ben-Hur, and G. Dror. Result analysis of the nips 2003 feature selection challenge. *Advances in neural information processing systems*, 17, 2004.

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(2):1299–1329, 2021.

A. Iouditski and Y. Nesterov. Primal-dual subgradient methods for minimizing uniformly convex functions. *arXiv preprint arXiv:1401.1792*, 2014.

M. Ito and M. Fukuda. Nearly optimal first-order methods for convex optimization under gradient norm measure: An adaptive regularization approach. *Journal of Optimization Theory and Applications*, 188(3):770–804, 2021.

M. Ito, Z. Lu, and C. He. A parameter-free conditional gradient method for composite minimization under hölder condition. *Journal of Machine Learning Research*, 24(166): 1–34, 2023.

P. R. Johnstone and P. Moulin. Faster subgradient methods for functions with hölderian growth. *Mathematical Programming*, 180(1):417–450, 2020.

G. Lan and Z. Zhou. Algorithms for stochastic optimization with expectation constraints. *arXiv preprint arXiv:1604.03887*, 2016.

G. Lan, Y. Ouyang, and Z. Zhang. Optimal and parameter-free gradient minimization methods for convex and nonconvex optimization. *arXiv preprint arXiv:2310.12139*, 2023.

D. D. Lewis, Y. Yang, T. G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5(Apr):361–397, 2004.

G. Li, B. S. Mordukhovich, and T. Pham. New fractional error bounds for polynomial systems with applications to hölderian stability in optimization and spectral theory of tensors. *Mathematical Programming*, 153(2):333–362, 2015.

Q. Lin and L. Xiao. An adaptive accelerated proximal gradient method and its homotopy continuation for sparse optimization. *Computational Optimization and Applications*, 60 (3):633–674, 2015.

Q. Lin, R. Ma, and T. Yang. Level-set methods for finite-sum constrained convex optimization. In *International Conference on Machine Learning*, pages 3112–3121, 2018a.

Q. Lin, S. Nadarajah, and N. Soheili. A level-set method for convex optimization with a feasible solution path. *SIAM Journal on Optimization*, 28(4):3290–3311, 2018b.

Q. Lin, S. Nadarajah, N. Soheili, and T. Yang. A data efficient and feasible level set method for stochastic convex optimization with expectation constraints. *Journal of Machine Learning Research*, 2020.

Q. Lin, R. Ma, and Y. Xu. Complexity of an inexact proximal-point penalty method for constrained smooth non-convex optimization. *Computational Optimization and Applications*, 82(1):175–224, 2022.

M. Liu and T. Yang. Adaptive accelerated gradient converging method under hölderian error bound condition. In *Advances in Neural Information Processing Systems*, pages 3104–3114, 2017.

I. Necoara, Y. Nesterov, and F. Glineur. Linear convergence of first order methods for non-strongly convex optimization. *Mathematical Programming*, 175(1-2):69–107, 2019.

Y. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

Y. Nesterov. *Lectures on Convex Optimization*, volume 137. Springer, 2018.

B. T. Polyak. Minimization of unsmooth functionals. *USSR Computational Mathematics and Mathematical Physics*, 9(3):14–29, 1969.

J. Renegar. "efficient" subgradient methods for general convex optimization. *SIAM Journal on Optimization*, 26(4):2649–2676, 2016.

J. Renegar and B. Grimmer. A simple nearly optimal restart scheme for speeding up first-order methods. *Foundations of Computational Mathematics*, 22(1):211–256, 2022.

J. Renegar and S. Zhou. A different perspective on the stochastic convex feasibility problem. *arXiv preprint arXiv:2108.12029*, 2021.

V. Roulet and A. d'Aspremont. Sharpness, restart and acceleration. In *Advances in Neural Information Processing Systems*, pages 1119–1129, 2017.

A. Sujanani and R. D. Monteiro. Efficient parameter-free restarted accelerated gradient methods for convex and strongly convex optimization: A. sujanani, rdc monteiro. *Journal of Optimization Theory and Applications*, 206(2):52, 2025.

T. Wang and H. Liu. Convergence results of a new monotone inertial forward–backward splitting algorithm under the local hölder error bound condition. *Applied Mathematics & Optimization*, 85(2):7, 2022.

X. Wei, H. Yu, Q. Ling, and M. Neely. Solving non-smooth constrained programs with lower complexity than $O(1/\epsilon)$: A primal-dual homotopy smoothing approach. In *Advances in Neural Information Processing Systems*, pages 3995–4005, 2018.

P. Wolfe. The simplex method for quadratic programming. *Econometrica: Journal of the Econometric Society*, pages 382–398, 1959.

Y. Xu. Primal-dual stochastic gradient method for convex programs with many functional constraints. *SIAM Journal on Optimization*, 30(2):1664–1692, 2020.

Y. Xu. First-order methods for constrained convex programming based on linearized augmented lagrangian function. *INFORMS Journal on Optimization*, 3(1):89–117, 2021a.

Y. Xu. Iteration complexity of inexact augmented lagrangian methods for constrained convex programming. *Mathematical Programming*, 185(1):199–244, 2021b.

Y. Xu and T. Yang. Frank-wolfe method is automatically adaptive to error bound condition. *arXiv preprint arXiv:1810.04765*, 2018.

Y. Xu, Y. Yan, Q. Lin, and T. Yang. Homotopy smoothing for non-smooth problems with lower complexity than $\mathcal{O}(1/\epsilon)$. In *Advances In Neural Information Processing Systems*, pages 1208–1216, 2016.

Y. Xu, Q. Lin, and T. Yang. Adaptive svrg methods under error bound conditions with unknown growth parameter. In *Advances in Neural Information Processing Systems*, pages 3277–3287, 2017a.

Y. Xu, Q. Lin, and T. Yang. Stochastic convex optimization: Faster local growth implies faster global convergence. In *International Conference on Machine Learning*, pages 3821–3830, 2017b.

Y. Xu, M. Liu, Q. Lin, and T. Yang. Admm without a fixed penalty parameter: Faster convergence with new adaptive penalization. In *Advances in Neural Information Processing Systems*, pages 1267–1277, 2017c.

Y. Yan, Y. Xu, Q. Lin, L. Zhang, and T. Yang. Stochastic primal-dual algorithms with faster convergence than $\mathcal{O}(1/\sqrt{T})$ for problems without bilinear structure. *arXiv preprint arXiv:1904.10112*, 2019.

T. Yang and Q. Lin. Rsg: Beating subgradient method without smoothness and strong convexity. *The Journal of Machine Learning Research*, 19(1):236–268, 2018.

T. Yang, Q. Lin, and L. Zhang. A richer theory of convex constrained optimization with reduced projections and improved rates. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3901–3910. JMLR. org, 2017.

H. Yu, M. Neely, and X. Wei. Online convex optimization with stochastic constraints. *Advances in Neural Information Processing Systems*, 30, 2017.

H. Zhang. New analysis of linear convergence of gradient-type methods via unifying error bound conditions. *Mathematical Programming*, 180(1):371–416, 2020.

H. Zhang, Y.-H. Dai, L. Guo, and W. Peng. Proximal-like incremental aggregated gradient method with linear convergence under bregman distance growth conditions. *Mathematics of Operations Research*, 46(1):61–81, 2021.